Installing Debian GNU/Linux 2.2 For Alpha

Bruce Perens
Sven Rudolph
Igor Grobman
James Treacy
Adam Di Carlo

version 2.2.20, 25 November, 2000

Abstract

This document contains installation instructions for the Debian GNU/Linux 2.2 system, for the Alpha ("alpha") architecture. It also contains pointers to more information and information on how to make the most of your new Debian system. The procedures in this document are *not* to be used for users upgrading existing systems; if you are upgrading, see the Release Notes for Debian 2.2 (`http://www.debian.org/releases/2.2/alpha/release-notes/`).

Copyright Notice

The table of Alpha architectures was derived from information contributed by Jay Estabrook, with kind permission.

Contents

Chapter 1

Welcome to Debian

We are delighted that you have decided to try Debian, and sure that you will find that Debian's GNU/Linux distribution is unique. Debian GNU/Linux brings together high–quality free software from around the world, integrating it into a coherent whole. We believe that you will find that the result is truly more than the sum of the parts.

This chapter provides an overview of the Debian Project and Debian GNU/Linux. If you already know about the Debian Project's history and the Debian GNU/Linux distribution, feel free to skip to the next chapter.

1.1 What is Debian?

Debian is an all–volunteer organization dedicated to developing free software and promoting the ideals of the Free Software Foundation. The Debian Project began in 1993, when Ian Murdock issued an open invitation to software developers to contribute to a complete and coherent software distribution based on the relatively new Linux kernel. That relatively small band of dedicated enthusiasts, originally funded by the Free Software Foundation (`http://www.gnu.org/fsf/fsf.html`) and influenced by the GNU (`http://www.gnu.org/`) philosophy, has grown over the years into an organization of around 500 *Debian Developers*.

Debian Developers are involved in a variety of activities, including Web (`http://www.debian.org/`) and FTP (`ftp://ftp.debian.org/`) site administration, graphic design, legal analysis of software licenses, writing documentation, and, of course, maintaining software packages.

In the interest of communicating our philosophy and attracting developers who believe in the principles that Debian stands for, the Debian Project has published a number of documents that outline our values and serve as guides to what it means to be a Debian Developer:

The Debian Social Contract (`http://www.debian.org/social_contract`) is a statement of Debian's commitments to the Free Software Community. Anyone who agrees to abide to the Social Contract may become a maintainer (`http://www.debian.org/doc/maint-guide/`). Any maintainer can introduce new software into Debian — provided that the software meets our criteria for being free, and the package follows our quality standards.

The Debian Free Software Guidelines (`http://www.debian.org/social_contract#guidelines`) are a clear and concise statement of Debian's criteria for free software. The DFSG is a very influential document in the Free Software Movement, and was the foundation of the Open Source Free Software Guidelines (`http://opensource.org/osd.html`).

The Debian Policy Manual (`http://www.debian.org/doc/debian-policy/`) is an extensive specification of the Debian Project's standards of quality.

Debian developers are also involved in a number of other projects; some specific to Debian, others involving some or all of the Linux community. Some examples include:

The Linux Standard Base (`http://www.linuxbase.org/`) (LSB) is a project aimed at standardizing the basic GNU/Linux system, which will enable third–party software and hardware developers to easily design programs and device drivers for Linux–in–general, rather than for a specific GNU/Linux distribution.

The Filesystem Hierarchy Standard (`http://www.pathname.com/fhs/`) (FHS) is an effort to standardize the layout of the Linux filesystem. The FHS will allow software developers to concentrate their efforts on designing programs, without having to worry about how the package will be installed in different GNU/Linux distributions.

Debian Jr. (`http://www.debian.org/devel/debian-jr/`) is an internal project, aimed at making sure Debian has something to offer to our youngest users.

For more general information about Debian, see the Debian FAQ (`http://www.debian.org/doc/FAQ/`).

1.2 What is GNU/Linux?

The GNU Project has developed a comprehensive set of free software tools for use with Unix^{TM} and Unix–like operating systems such as Linux. These tools enable users to perform tasks ranging from the mundane (such as copying or removing files from the system) to the arcane (such as writing and compiling programs or doing sophisticated editing in a variety of document formats).

An operating system consists of various fundamental programs which are needed by your computer so that it can communicate and receive instructions from users; read and write data to hard disks, tapes, and printers; control the use of memory; and run other software. The most important part of an operating system is the kernel. In a GNU/Linux system, Linux is the kernel component. The rest of the system consists of other programs, many of which were written by or for the GNU Project. Because the Linux kernel alone does not form a working operating system, we prefer to use the term "GNU/Linux" to refer to systems that many people casually refer to as "Linux".

The Linux kernel (`http://www.kernel.org/`) first appeared in 1991, when a Finnish computing science student named Linus Torvalds announced an early version of a replacement kernel for Minix to the Usenet newsgroup `comp.os.minix`. See Linux International's Linux History Page (`http://www.li.org/linuxhistory.php`).

Linus Torvalds continues to coordinate the work of several hundred developers with the help of a few trusty deputies. An excellent weekly summary of discussions on the `linux-kernel` mailing list is Kernel Traffic (`http://kt.linuxcare.com/kernel-traffic/`). More information about the `linux-kernel` mailing list can be found on the linux-kernel mailing list FAQ (`http://www.tux.org/lkml/`).

1.3 What is Debian GNU/Linux?

The combination of Debian's philosophy and methodology and the GNU tools, the Linux kernel, and other important free software, form a unique software distribution called Debian GNU/Linux. This distribution is made up of a large number of software *packages*. Each package in the distribution contains executables, scripts, documentation, and configuration information, and has a *maintainer* who is primarily responsible for keeping the package up–to–date, tracking bug reports, and communicating with the upstream author(s) of the packaged software. Our extremely large user base, combined with our bug tracking system ensures that problems are found and fixed quickly.

Debian's attention to detail allows us to produce a high–quality, stable, and scalable distribution. Installations can be easily configured to serve many roles, from stripped–down firewalls to desktop scientific workstations to high–end network servers.

The feature that most distinguishes Debian from other GNU/Linux distributions is its package management system. These tools give the administrator of a Debian system complete control over the packages installed on that system, including the ability to install a single package or automatically update the entire operating system. Individual packages can also be protected from being updated. You can even tell the package management system about software you have compiled yourself and what dependencies it fulfills.

To protect your system against "trojan horses" and other malevolent software, Debian's servers verify that uploaded packages come from their registered Debian maintainers. Debian packagers also take great care to configure their packages in a secure manner. When security problems in shipped packages do appear, fixes are usually available very quickly. With Debian's simple update options, security fixes can be downloaded and installed automatically across the Internet.

The primary, and best, method of getting support for your Debian GNU/Linux system and communicating with Debian Developers is through the many mailing lists maintained by the Debian Project (there are more than 90 at this writing). The easiest way to subscribe to one or more of these lists is visit Debian's mailing list subscription page (`http://www.debian.org/MailingLists/subscribe`) and fill out the form you'll find there.

1.4 What is Debian GNU/Hurd?

Debian GNU/Hurd is a Debian GNU system that replaces the Linux monolithic kernel with the GNU Hurd — a set of servers running on top of the GNU Mach microkernel. The Hurd is still unfinished, and is unsuitable for day–to–day use, but work is continuing. The Hurd is currently only being developed for the i386 architecture, although ports to other architectures will be made once the system becomes more stable.

For more information, see the Debian GNU/Hurd ports page (`http://www.debian.org/ports/hurd/`) and the `<debian-hurd@lists.debian.org>` mailing list.

1.5 Getting Debian

For information on how to download Debian GNU/Linux from the Internet or from whom official Debian CDs can be purchased, see the distribution web page (`http://www.debian.org/distrib/`). The list of Debian mirrors (`http://www.debian.org/distrib/ftplist`) contains a full set of official Debian mirrors.

Debian can be upgraded after installation very easily. The installation procedure will help setup up the system so that you can make those upgrades once installation is complete, if need be.

1.6 Getting the Newest Version of This Document

This document is constantly being revised. Be sure to check the Debian 2.2 pages (`http://www.debian.org/releases/2.2/`) for any last–minute information about the 2.2 release of the Debian GNU/Linux system. Updated versions of this installation manual are also available from the official Install Manual pages (`http://www.debian.org/releases/2.2/alpha/install`).

1.7 Organization of This Document

This document is meant to serve as a manual for first–time Debian users. It tries to make as few assumptions as possible about your level of expertise. However, we do assume that you have a general understanding of how the hardware in your computer works.

Expert users may also find interesting reference information in this document, including minimum installation sizes, details about the hardware supported by the Debian installation system, and so on. We encourage expert users to jump around in the document.

In general, this manual is arranged in a linear fashion, walking you through the installation process from start to finish. Here are the steps in installing Debian GNU/Linux, and the sections of this document which correlate with each step:

1. Determine whether your hardware meets the requirements for using the installation system, in ‘System Requirements’ on page 7.
2. Backup your system, and perform any necessary planning and hardware configuration prior to installing Debian, in ‘Before You Start’ on page 13.
3. Getting the partitions on your system set up correctly is very important, because once you’ve done the install, you may have to live with your choices for a long time.
4. In ‘Methods for Installing Debian’ on page 21, several different ways to install Debian are presented and discussed. Select your favorite method and prepare your installation media as described.
5. ‘Booting the Installation System’ on page 33, describes booting into the installation system. This chapter also discusses troubleshooting procedures in case you have problems with this step.

6. Perform the initial system configuration, which is discussed in ‘Using `dbootstrap` for Initial System Configuration’ on page 43 (Sections ‘Introduction to `dbootstrap`’ on page 43 through “‘Configure the Network”’ on page 49).
7. “‘Install the Base System”’ on page 50.
8. Boot into your newly installed base system and run through some additional configuration tasks, from ‘The Moment of Truth’ on page 51.
9. Install the rest of the system, using `dselect` or `apt-get`, in ‘Installing the Rest of Your System’ on page 54.

Once you’ve got your system installed, you can read ‘Next Steps and Where to Go From Here’ on page 55. That chapter explains where to look to find more information about Unix and Debian, and how to replace your kernel. If you want to build your own install system from source, be sure to read ‘Technical Information on the Boot Floppies’ on page 59.

Finally, information about this document and how to contribute to it may be found in ‘Administrivia’ on page 63.

1.8 WARNING: This Document Has Known Problems

This document is still in a rather rough form. It is known to be incomplete, and probably also contains errors, grammatical problems, and so forth. If you see the words “FIXME” or “TODO”, you can be sure we already know that section is not complete. As usual, *caveat emptor* (buyer beware). Any help, suggestions, and, especially, patches, would be greatly appreciated.

The non–x86 versions of this document may be particularly incomplete, inaccurate, and untested. Your help is definitely wanted!

Working versions of this document can be found at `http://www.debian.org/releases/2.2/alpha/install`. There you will find a list of all the different architectures and languages for which this document is available.

Source is also available publicly; look for more information concerning how to contribute in ‘Administrivia’ on page 63. We welcome suggestions, comments, patches, and bug reports (use the package `boot-floppies`, but check first to see if the problem is already reported).

1.9 About Copyrights and Software Licenses

We’re sure that you’ve read some of the licenses that come with most commercial software — they usually say that you can only use one copy of the software on a single computer. The Debian GNU/Linux system’s license isn’t like that at all. We encourage you to put a copy of Debian GNU/Linux on every computer in your school or place of business. Lend your installation media to your friends and help them install it on their computers! You can even make thousands of copies and *sell* them — albeit with a few restrictions. Your freedom to install and use the system comes directly from Debian being based on *free software*.

Calling software “free” doesn’t mean that the software isn’t copyrighted, and it doesn’t mean that CDs containing that software must be distributed at no charge. Free software, in part, means that the licenses of individual programs do not require you to pay for the privilege of distributing or using those programs. Free software also means that not only may anyone extend, adapt, and modify the software, but that they may distribute the results of their work as well.[1]

Many of the programs in the system are licensed under the *GNU General Public License*, often simply referred to as “the GPL”. The GPL requires you to make the *source code* of the programs available whenever you distribute a binary copy of the program; that provision of the license ensures that any user will be able to modify the software. Because of this provision, the source code for all such programs is available in the Debian system.[2]

There are several other forms of copyright statements and software licenses used on the programs in Debian. You can find the copyrights and licenses for every package installed on your system by looking in the file `/usr/doc/`*`package-name`*`/copyright` once you’ve installed a package on your system.

For more information about licenses and how Debian determines whether software is free enough to be included in the main distribution, see the Debian Free Software Guidelines (`http://www.debian.org/social_contract#guidelines`).

The most important legal notice is that this software comes with *no warranties*. The programmers who have created this software have done so for the benefit of the community. No guarantee is made as to the suitability of the software for any given purpose. However, since the software is free, you are empowered to modify that software to suit your needs — and to enjoy the benefits of the changes made by others who have extended the software in this way.

[1]Note that the Debian project, as a pragmatic concession to its users, does make some packages available that do not meet our criteria for being free. These packages are not part of the official distribution, however, and are only available from the `contrib` or `non-free` areas of Debian mirrors or on third–party CD–ROMs; see the Debian FAQ (`http://www.debian.org/doc/FAQ/`), under “The Debian FTP archives”, for more information about the layout and contents of the archives.

[2]For information on how to locate, unpack, and build binaries from Debian source packages, see the Debian FAQ (`http://www.debian.org/doc/FAQ/`), under “Basics of the Debian Package Management System”.

Chapter 2

System Requirements

This section contains information about what hardware you need to get started with Debian. You will also find links to further information about hardware supported by GNU and Linux.

2.1 Supported Hardware

Debian does not impose hardware requirements beyond the requirements of the Linux kernel and the GNU tool–sets. Therefore, any architecture or platform to which the Linux kernel, libc, `gcc`, etc. have been ported, and for which a Debian port exists, can run Debian.

There are, however, some limitations in our boot floppy set with respect to supported hardware. Some Linux–supported platforms might not be directly supported by our boot floppies. If this is the case, you may have to create a custom rescue disk (see 'Replacing the Rescue Floppy Kernel' on page 59), or investigate network installations.

Rather than attempting to describe all the different hardware configurations which are supported for Alpha, this section contains general information and pointers to where additional information can be found.

2.1.1 Supported Architectures

Debian 2.2 supports six architectures: Intel x86–based architectures; Motorola 680x0 machines such as Atari, Amiga, and Macintoshes; DEC Alpha machines; Sun SPARC machines; ARM and StrongARM machines; and some IBM/Motorola PowerPC machines, including CHRP, PowerMac and PReP machines. These are referred to as *i386*, *m68k*, *alpha*, *sparc*, *arm*, and *powerpc*, respectively.

This document covers installation for the *alpha* architecture. If you look for information on other architectures take a look at the Debian-Ports (`http://www.debian.org/ports/`) pages.

2.1.2 CPU, Mainboards, and Video Support

Complete information regarding supported DEC Alphas can be found at Linux Alpha HOWTO (`http://www.linuxdoc.org/HOWTO/Alpha-HOWTO.html`). The purpose of this section is to describe the systems supported by the boot disks.

Alpha machines are subdivided into different system types because there are a number of generations of motherboard and supporting chip–sets. Different systems (“sub–architectures”) often have radically different engineering and capabilities. Therefore, the process of installing and, more to the point, booting, can vary from system to system.

The following table lists the system types supported by the Debian installation system. The table also indicates the *code name* for these system types. You’ll need to know this code name when you actually begin the installation process:

```
Family/Model                     Code Name
=============                    =========
ALPHAbook 1                      book1

ALCOR
  AS 600                         alcor
  AS 500 5/3xx                   alcor
  AS 500 5/5xx                   alcor
  XL-300/366/433                 xlt

AVANTI
  AS 200 4/*                     avanti
  AS 205 4/*                     avanti
  AS 250 4/*                     avanti
  AS 255 4/*                     avanti
  AS 300 4/*                     avanti
  AS 400 4/*                     avanti

DP264                            dp264
  DP264                          dp264
  AS DP10                        dp264
  AS DP20                        dp264
  AS ES40                        dp264
  AS XP1000                      dp264
  UP2000                         dp264

EB164                            eb164
  AlphaPC164                     pc164
  AlphaPC164-LX                  lx164
  AlphaPC164-SX                  sx164

EB64+
  EB64+                          eb64p
  AlphaPC64                      cabriolet
  AlphaPCI-64                    cabriolet

EB66                             eb66

EB66+                            eb66p

JENSEN
  DECpc 150                      jensen
  DEC 2000 Model 300             jensen

MIKASA
  AS 1000 4/xxx                  mikasa
  AS 1000 5/xxx                  mikasa-p

NAUTILUS
  UP1000                         nautilus

NONAME
  AXPpci33                       noname
  UDB                            noname

NORITAKE
  AS 1000A 4/xxx                 noritake
  AS 1000A 5/xxx                 noritake-p
  AS 600A 5/xxx                  noritake-p
  AS 800 5/xxx                   noritake-p

Personal Workstation
  PWS 433a or 433au              miata
  PWS 500a or 500au              miata
  PWS 600a or 600au              miata

RAWHIDE
  AS 4000                        rawhide
  AS 4100                        rawhide
  AS 1200                        rawhide

RUFFIAN
  Deskstation RPX164-2           ruffian
  Samsung AlphaPC164-UX/BX       ruffian

SABLE
  AS 2100 4/xxx                  sable
  AS 2000 4/xxx                  sable
  AS 2100 5/xxx                  sable-g
  AS 2000 5/xxx                  sable-g

TAKARA                           takara

XL
  XL-233/266                     xl
```

2.1.3 Multiple Processors

Multi–processor support — also called “symmetric multi–processing” or SMP — is supported for this architecture. However, the standard Debian 2.2 kernel image does not support SMP. This should not prevent installation, since the standard, non–SMP kernel should boot on SMP systems; the kernel will simply use the first CPU.

In order to take advantage of multiple processors, you’ll have to replace the standard Debian kernel. You can find a discussion of how to do this in ‘Compiling a New Kernel’ on page 56. At this time (kernel version 2.2.18pre21) the way you enable SMP is to select “symmetric multi–processing” in the “General” section of the kernel config. If you compile software on a multiprocessor system, look for the `-j` flag in the documentation on `make(1)`.

2.2 Installation Media

There are four different media which can be used to install Debian: floppies, CD–ROMs, local disk partitions, or the network. Different parts of the same Debian installation can mix and match these options; we’ll go into that in ‘Methods for Installing Debian’ on page 21.

Floppy disk installation is a common option, although generally, the least desirable. In many cases, you’ll have to do your first boot from floppies, using the Rescue Floppy. Generally, all you will need is a high–density (1440 kilobytes) 3.5 inch floppy drive.

CD–ROM based installation is also supported for some architectures. On machines which support bootable CD–ROMs, you should be able to do a completely floppy–less installation. Even if your system doesn’t support booting from a CD–ROM, you can use the CD–ROM in conjunction with the other techniques to install your system, once you’ve booted up by other means; see ‘Installing from a CD–ROM’ on page 35.

Installation from local disk is another option. If you have free space on partitions other than the partitions you’re installing to, this is definitely a good option. Some platforms even have local installers, i.e., for booting from AmigaOS, TOS, or MacOS.

The last option is network installation. You can install your base system via HTTP or NFS. You can also *boot* your system over the network. Diskless installation, using network booting and NFS–mounting of all local filesystems, is another option — you’ll probably need at least 16MB of RAM for this option. After your base system is installed, you can install the rest of your system via any sort of network connection (including PPP), via FTP, HTTP, or NFS.

More complete descriptions of these methods, and helpful hints for picking which method is best for you, can be found in 'Methods for Installing Debian' on page 21. Please be sure to continue reading to make sure the device you intend to boot and install from is supported by the Debian installation system.

2.2.1 Supported Storage Systems

The Debian boot disks contain a kernel which is built to maximize the number of systems it runs on. Unfortunately, this makes for a larger kernel, with a lot of drivers which will never be used (see 'Compiling a New Kernel' on page 56 to learn how to build your own). However, support for the widest possible range of devices is desirable in order to ensure that Debian can be installed on the widest array of hardware. Any storage system supported by the Linux kernel is also supported by the boot system. The following SCSI drivers are supported in the default kernel:

- Qlogic ISP
- NCR and Symbios 53c8xx
- Adaptec AIC7xxx

IDE disks are also supported. Note, however, that on many systems, the SRM console is unable to boot from IDE drives, and the Jensen is unable to boot from floppies. (see `http://www.alphalinux.org/faq/FAQ-9.html` for more information on booting the Jensen)

2.3 Memory and Disk Space Requirements

You must have at least 16MB of memory and 100MB of hard disk. If you want to install a reasonable amount of software, including the X Window System, and some development programs and libraries, you'll need at least 300MB. For a more or less complete installation, you'll need around 800MB. To install *everything* available in Debian, you'll probably need around 2 GB. Actually, installing everything doesn't even make sense, since some packages conflict with others.

2.4 Peripherals and Other Hardware

Linux supports a large variety of hardware devices such as mice, printers, scanners, modems, network cards, PCMCIA devices, etc. However, none of these devices are required while installing the system. This section contains information about peripherals specifically *not* supported by the installation system, even though they may be supported by Linux.

Any network interface card (NIC) supported by the Linux kernel should also be supported by the boot disks. Support for the built–in DECChip (Tulip) Ethernet on many Alpha models is compiled directly into the kernel. (FIXME: the kernel maintainer screwed up and forgot this, make sure this is fixed for the release) For other cards, you may need to load your network driver as a module.

2.5 Purchasing Hardware Specifically for GNU/Linux

There are several vendors, now, who ship systems with Debian or other distributions of GNU/Linux pre–installed. You might pay more for the privilege, but it does buy a level of peace of mind, since you can be sure that the hardware is well–supported by GNU/Linux.

Whether or not you are purchasing a system with Linux bundled, or even a used system, it is still important to check that your hardware is supported by the Linux kernel. Check if your hardware is listed in the references found above. Let your salesperson (if any) know that you're shopping for a Linux system. Support Linux–friendly hardware vendors.

2.5.1 Avoid Proprietary or Closed Hardware

Some hardware manufacturers simply won't tell us how to write drivers for their hardware. Others won't allow us access to the documentation without a non–disclosure agreement that would prevent us from releasing the Linux source code. One example is the IBM laptop DSP sound system used in recent ThinkPad systems — some of these systems also couple the sound system to the modem. Another example is the proprietary hardware in the older Macintosh line.

Since we haven't been granted access to the documentation on these devices, they simply won't work under Linux. You can help by asking the manufacturers of such hardware to release the documentation. If enough people ask, they will realize that the free software community is an important market.

Chapter 3

Before You Start

3.1 Backups

Before you start, make sure to back up every file that is now on your system. The installation procedure can wipe out all of the data on a hard disk! The programs used in installation are quite reliable and most have seen years of use; still, a false move can cost you. Even after backing up be careful and think about your answers and actions. Two minutes of thinking can save hours of unnecessary work.

Even if you are installing a multi–boot system, make sure that you have on hand the distribution media of any other present operating systems. Especially if you repartition your boot drive, you might find that you have to reinstall your operating system's boot loader, or in some cases (i.e., Macintosh), the whole operating system itself.

3.2 Information You Will Need

Besides this document, you'll need the cfdisk (`cfdisk.txt`) manual page, the fdisk (`fdisk.txt`) manual page, the dselect Tutorial (`dselect-beginner`), and the Linux/Alpha FAQ (`http://www.alphalinux.org/faq/FAQ.html`).

If your computer is connected to a network 24 hours a day (i.e., an Ethernet or equivalent connection — not a PPP connection), you should ask your network's system administrator for this information:

- Your host name (you may be able to decide this on your own).
- Your domain name.
- Your computer's IP address.
- The IP address of your network.
- The netmask to use with your network.
- The broadcast address to use on your network.

The IP address of the default gateway system you should route to, if your network *has* a gateway.

The system on your network that you should use as a DNS (Domain Name Service) server.

Whether you connect to the network using Ethernet.

If your computer's only network connection is via a serial line, using PPP or an equivalent dialup connection, you are probably not installing the base system over a network. You don't need to worry about getting your network setup until your system is already installed. See 'Setting up PPP' on page 53 below for information on setting up PPP under Debian.

3.3 Pre–installation Hardware and Operating System Setup

There is sometimes some tweaking to your system that must be done prior to installation. The x86 platform is the most notorious of these; pre–installation hardware setup on other architectures is considerably simpler.

This section will walk you through pre–installation hardware setup, if any, that you will need to do prior to installing Debian. Generally, this involves checking and possibly changing firmware settings for your system. The "firmware" is the core software used by the hardware; it is most critically invoked during the bootstrap process (after power–up).

3.3.1 Over–Clocking your CPU

Many people have tried operating their 90 MHz CPU at 100 MHz, etc. It sometimes works, but is sensitive to temperature and other factors and can actually damage your system. One of the authors of this document over–clocked his own system for a year, and then the system started aborting the `gcc` program with an unexpected signal while it was compiling the operating system kernel. Turning the CPU speed back down to its rated value solved the problem.

3.3.2 Bad Memory Modules

The `gcc` compiler is often the first thing to die from bad memory modules (or other hardware problems that change data unpredictably) because it builds huge data structures that it traverses repeatedly. An error in these data structures will cause it to execute an illegal instruction or access a non–existent address. The symptom of this will be `gcc` dying from an unexpected signal.

Chapter 4

Partitioning Your Hard Drive

4.1 Background

Partitioning your disk simply refers to the act of breaking up your disk into sections. Each section is then independent of the others. It's roughly equivalent to putting up walls in a house; if you add furniture to one room it doesn't affect any other room.

If you already have an operating system on your system (Windows95, Windows NT, OS/2, MacOS, Solaris, FreeBSD, . . .) and want to stick Linux on the same disk, you will probably need to repartition the disk. In general, changing a partition with a filesystem already on it will destroy any information there. Thus you should always make backups before doing any repartitioning. Using the analogy of the house, you would probably want to move all the furniture out of the way before moving a wall or you risk destroying it.

At a bare minimum, GNU/Linux needs one partition for itself. You can have a single partition containing the entire operating system, applications, and your personal files. Most people feel that the swap partition is also a necessity, although it's not strictly true. "Swap" is scratch space for an operating system, which allows the system to use cheap disk storage as "virtual memory". By putting swap on a separate partition, Linux can make much more efficient use of it. It is possible to force Linux to use a regular file as swap, but it is not recommended.

Most people choose to give GNU/Linux more than the minimum number of partitions, however. There are two reasons you might want to break up the filesystem into a number of smaller partitions. The first is for safety. If something happens to corrupt the file system, generally only one partition is affected. Thus, you only have to replace (from the backups you've been carefully keeping) a portion of your system. At a bare minimum, you should consider creating what is commonly called a "root partition". This contains the most essential components of the system. If any other partitions get corrupted, you can still boot into GNU/Linux to fix the system. This can save you the trouble of having to reinstall the system from scratch.

The second reason is generally more important in a business setting, but it really depends on your use of the machine. Suppose something runs out of control and starts eating disk space. If the process causing the problem happens to have root privileges (the system keeps a percentage of the disk away from users), you could suddenly find yourself out of disk space. This is not good as the OS needs to use real files (besides swap space) for many things. It may not even be a problem of local origin. For example, getting spammed with e–mail can easily fill a partition. By using more partitions, you protect the system from many of these problems. Using mail as an example again, by putting `/var/spool/mail` on its own partition, the bulk of the system will work even if you get spammed.

The only real drawback to using more partitions is that it is often difficult to know in advance what your needs will be. If you make a partition too small then you will either have to reinstall the system or you will be constantly moving things around to make room in the undersized partition. On the other hand, if you make the partition too big, you will be wasting space that could be used elsewhere. Disk space is cheap nowadays, but why throw your money away?

4.1.1 The Directory Tree

The following list describes some important directories. It should help you to find out what your partitioning scheme should be. If this is too confusing for you, just ignore it and reread it when you read the rest of the installation manual.

`/`: root represents the starting point of the directory hierarchy. It contains the essential programs that the computer can boot. This includes the kernel, system libraries, configuration files in `/etc` and various other needed files. Typically 30–50 MB are needed but this may vary.

Note: do *not* partition `/etc`, `/bin`, `/sbin`, `/lib` or `/dev` as its own partition; you won't be able to boot.

`/dev`: this directory contains the various device files which are interfaces to the various hardware components. For more information see 'Device Names in Linux' on page 18.

`/usr`: all user programs (`/usr/bin`), libraries (`/usr/lib`), documentation (`/usr/share/doc`), etc., are in this directory. This part of the filesystem needs most of the space. You should at least provide 300–500 MB of disk space. If you want to install more packages you should increase the amount of space you give this directory.

`/home`: every user will put his data into a subdirectory of this directory. The size of this depends on how many users will be using the system and what files are to be stored in their directories. Depending on your planned usage you should reserve about 100 MB for each user, but adapt this value to your needs.

`/var`: all variable data like news articles, e–mails, websites, APT's cache, etc. will be placed under this directory. The size of this directory depends greatly on the usage of your computer, but for most people will be dictated by the package management tool's overhead. If you are going to do a full installation of just about everything Debian has to offer, all in one session, setting aside 2 or 3 gigabytes of space for `/var` should be sufficient. If you are going to install in pieces (that is to say, install services and utilities, followed by text stuff, then X, . . .), you can get away with 2–5 hundred megabytes of room in `/var`. If harddrive space is at a premium and you don't plan on using APT, at least not for major updates, you can get by with as little as 30 or 40 megabytes in `/var`.

`/tmp`: if a program creates temporary data it will most likely go in there. 20–50 MB should be usually enough.

`/proc`: this is a virtual file system which doesn't reside on the harddisk. Thus no harddisk space is needed. It provides interesting and also vital information about the running system.

4.2 Planning Use of the System

It is important to decide what type of machine you are creating. This will determine disk space requirements and affect your partitioning scheme.

This has changed for Potato — we need to update it. There are a number of common task applications What does this need to be called? which Debian offers for your convenience (see 'Select and Install Profiles' on page 53). Common task applications are simply sets of package selections which make it easier for you, in that a number of packages are automatically marked for installation.

Each given common task application has a size of the resulting system after installation is complete. Even if you don't use these common task applications, this discussion is important for planning, since it will give you a sense of how large your partition or partitions need to be.

The following are some of the available common task applications and their sizes: The various applications and sizes should probably go here.

Server_std This is a small server profile, useful for stripped down server which does not have a lot of niceties for shell users. It basically has an FTP server, a web server, DNS, NIS, and POP. It will take up around 50MB. Of course, this is just size of the software; any data you serve up would be additional.

Dialup A standard desktop box, including the X window system, graphics applications, sound, editors, etc. Size of the packages will be around 500MB.

Work_std A more stripped–down user machine, without the X window system or X applications. Possibly suitable for a laptop or mobile computer. The size is around 140MB. (Note that the author has a pretty simple laptop setup including X11 in even less, around 100MB).

Devel_comp A desktop setup with all the development packages, such as Perl, C, C++, etc. Size is around 475MB. Assuming you are adding X11 and some additional packages for other uses, you should plan around 800MB for this type of machine.

Remember that these sizes don't include all the other materials which are usually to be found, such as user files, mail, and data. It is always best to be generous when considering the space for your own files and data. Notably, the Debian `/var` partition contains a lot of state information. The `dpkg` files (with information on all installed packages) can easily consume 20MB; with logs and the rest, you should usually allocate at least 50MB for `/var`.

4.3 Device Names in Linux

Linux disks and partition names may be different from other operating systems. You need to know the names that Linux uses when you create and mount partitions. Here's the basic naming scheme:

> The first floppy drive is named “/dev/fd0”.
>
> The second floppy drive is named “/dev/fd1”.
>
> The first SCSI disk (SCSI ID address–wise) is named “/dev/sda”.
>
> The second SCSI disk (address–wise) is named “/dev/sdb”, and so on.
>
> The first SCSI CD–ROM is named “/dev/scd0”, also known as “/dev/sr0”.
>
> The master disk on IDE primary controller is named “/dev/hda”.
>
> The slave disk on IDE primary controller is named “/dev/hdb”.
>
> The master and slave disks of the secondary controller can be called “/dev/hdc” and “/dev/hdd”, respectively. Newer IDE controllers can actually have two channels, effectively acting like two controllers.

The partitions on each disk are represented by appending a decimal number to the disk name: “sda1” and “sda2” represent the first and second partitions of the first SCSI disk drive in your system.

Here is a real–life example. Let's assume you have a system with 2 SCSI disks, one at SCSI address 2 and the other at SCSI address 4. The first disk (at address 2) is then named “sda”, and the second “sdb”. If the “sda” drive has 3 partitions on it, these will be named “sda1”, “sda2”, and “sda3”. The same applies to the “sdb” disk and its partitions.

Note that if you have two SCSI host bus adapters (i.e., controllers), the order of the drives can get confusing. The best solution in this case is to watch the boot messages, assuming you know yourself the drive models.

4.4 Recommended Partitioning Scheme

As described above, you should definitely have a separate smaller root partition, and a larger `/usr` partition, if you have the space. For examples, see below. For most users, the two partitions initially mentioned are sufficient. This is especially appropriate when you have a single small disk, since breaking out lots of partitions can waste space.

In some cases, you might need a separate `/usr/local` partition if you plan to install many programs that are not part of the Debian distribution. If your machine will be a mail server, you might need to make `/var/spool/mail` a separate partition. Often, putting `/tmp` on its own partition, for instance 20 to 32MB, is a good idea. If you are setting up a server with lots of user accounts, it's generally good to have a separate, large `/home` partition. In general, the partitioning situation varies from computer to computer depending on its uses.

For very complex systems, you should see the Multi Disk HOWTO (`http://www.linuxdoc.org/HOWTO/Multi-Disk-HOWTO.html`). This contains in–depth information, mostly of interest to ISPs and people setting up servers.

With respect to the issue of swap partition size, there are many views. One rule of thumb which works well is to use as much swap as you have system memory, although there probably isn't much point in going over 64MB of swap for most users. It also shouldn't be smaller than 16MB, in most cases. Of course, there are exceptions to these rules. If you are trying to solve 10000 simultaneous equations on a machine with 256MB of memory, you may need a gigabyte (or more) of swap.

On 32–bit architectures (i386, m68k, 32–bit SPARC, and PowerPC), the maximum size of a swap partition is 2GB (on Alpha and SPARC64, it's so large as to be virtually unlimited). This should be enough for nearly any installation. However, if your swap requirements are this high, you should probably try to spread the swap across different disks (also called "spindles") and, if possible, different SCSI or IDE channels. The kernel will balance swap usage between multiple swap partitions, giving better performance.

4.5 Example Partitioning

As an example, one of the authors' home machine has 32MB of RAM and a 1.7GB IDE drive on `/dev/hda`. There is a 500MB partition for another operating system on `/dev/hda1` (should have made it 200MB as it never gets used). A 32MB swap partition is used on `/dev/hda3` and the rest (about 1.2GB on `/dev/hda2`) is the Linux partition.

4.6 Partitioning Prior to Installation

There are two different times that you can partition: prior to the installation of Debian, or during installation of Debian. If your computer will be solely dedicated to Debian, you should partition as part of the installation process ("'Partition a Hard Disk"' on page 45). If you have a machine with more than one operating system on it, you generally should let the native operating system create its own partitions.

The following sections contain information regarding partitioning in your native operating system prior to installation. Note that you'll have to map between how the other operating system names partitions, and how Linux names partitions; see 'Device Names in Linux' on the facing page.

4.6.1 Partitioning in Tru64 UNIX

Tru64 UNIX, formerly known as Digital UNIX, which is in turn formerly known as OSF/1, uses the partitioning scheme similar to the BSD 'disk label', which allows for up to eight partitions per disk drive. The partitions are numbered '1' through to '8' in Linux and "lettered" 'a' through to 'h' in UNIX. Linux kernels 2.2 and higher always correspond '1' to 'a', '2' to 'b' and so on. For example, `rz0e` in Tru64 UNIX would most likely be called `sda5` in Linux.

Partitions in the disk label may overlap. Moreover, the 'c' partition is required to span the entire disk (thus overlapping all other non–empty partitions). Under Linux this makes `sda3` identical to `sda` (`sdb3` to `sdb`, if present, and so on). Apart from satisfying this requirement, there is, however, not much point in creating overlapping partitions.

Another conventional requirement is for the ‘a’ partition to start from the beginning of the disk, so that it always includes the boot block with the disk label.

Disks can be partitioned with the graphical disk configuration tool that is accessible through the Application Manager, or with the command–line `disklabel` utility. Partition type for the Linux file system should be set to ‘resrvd8’. This can only be done via `disklabel`; however, all other configuration can easily be performed with the graphical tool.

It is possible, and indeed quite reasonable, to share a swap partition between UNIX and Linux. In this case it will be needed to do a `mkswap` on that partition every time the system is rebooted from UNIX into Linux, as UNIX will damage the swap signature. You may want to run `mkswap` from the Linux start–up scripts before adding swap space with `swapon -a`.

If you want to mount UNIX partitions under Linux, note that Digital UNIX can use two different file system types, UFS and AdvFS, of which Linux only understands the former.

4.6.2 Partitioning in Windows NT

Windows NT uses the PC–style partition table. If you are manipulating existing FAT or NTFS partitions, it is recommended that you use the native Windows NT tools (or, more conveniently, you can also repartition your disk from the AlphaBIOS setup menu). Otherwise, it is not really necessary to partition from Windows; the Linux partitioning tools will generally do a better job. Note that when you run NT, the Disk Administrator may offer you to write a “harmless signature” on non–Windows disks if you have any. *Never* let it do that, as this signature will destroy the partition information.

If you plan to boot Linux from an ARC/AlphaBIOS/ARCSBIOS console, you will need a (small) FAT partition for MILO. One megabyte is quite sufficient. If Windows NT is installed, its 6 Mb bootstrap partition can be employed for this purpose.

Chapter 5

Methods for Installing Debian

You can install Debian from a variety of sources, both local (CD, hard disk, floppies) and remote (FTP, NFS, PPP, HTTP). Debian also supports various hardware configurations, so you may still have a few choices to make before you get going. This chapter lays out the choices and some suggestions for how to make them.

You can make different choices for different steps in the installation. For example, you may start the installation by booting off diskettes, but then feed later steps in the install process files from your hard disk.

As the installation progresses you will move from a scrawny, incapable system which lives only in RAM to a full–featured Debian GNU/Linux system installed on the hard disk. One of the key goals of the early installation steps is to increase the variety of hardware (e.g., interface cards) and software (e.g., network protocols and file system drivers) the system supports. Consequently, later installation steps can use a broader range of sources than earlier ones.

The easiest route for most people will be to use a set of Debian CDs. If you have such a set, and if your machine supports booting directly off the CD, great! Simply insert your CD, reboot, and proceed to the next chapter. If it turns out the standard installation doesn't work for your hardware, you can come back here to see about alternate kernels and installation methods which may work for you. In particular, note that some CD sets provide different kernels on different CDs, so that booting off some CD other than the first may work for you.

5.1 Overview of the Installation Process

This overview highlights the points for which you must choose an installation media, or make a choice which will affect which sources you can choose later. The following steps will occur:

1. You begin by booting the installation system.
2. You answer a series of questions to perform the initial system configuration.
3. You provide a media source for the kernel and drivers.

4. You select which drivers to load.
5. You provide a media source for the base system.
6. You reboot the system and then do some final configuration.
7. You install additional software, packages, at your discretion.

In making your choices, you need to bear a few factors in mind. The first involve your choice of kernel. The kernel that you pick for the initial system boot is the same kernel that your fully configured system will use. Since drivers are kernel–specific, you must pick a package containing drivers which go with your kernel. We'll turn shortly to the details of picking the right kernel, or rather, installation set.

Different kernels also have different networking abilities out of the box, and so also expand or limit your source choices, particularly early in the install process.

Finally, the particular drivers that you choose to load can enable additional hardware (e.g., network interface cards, hard drive controllers) or file systems (e.g., NTFS or NFS). This therefore widens the choices of installation source media.

5.2 Choosing the Right Installation Set

Your hardware will dictate your choice of installation. Choose the appropriate sub–architecture directory, review the documentation there, and proceed.

If you are booting from CD, different CDs use different installation sets. Consult your CD documentation for more information. Details on kernel arrangement for specific CDs needed.

5.3 Installation Sources for Different Installation Stages

This section indicates the type of hardware which *may*, and usually *will*, work at different stages of the installation. It is not a guarantee that all hardware of the indicated type will work with all kernels. For example, RAID disks generally will not be accessible until you install the appropriate drivers.

5.3.1 Booting the Initial Installation System

The initial boot of the installation system is perhaps the most idiosyncratic step. The next chapter provides additional details, but your choices generally include

the Rescue Floppy (note that on installations using `MILO` as a bootloader, you will need several disks to boot – see 'Booting the Installation System' on page 33 for more information)

a bootable CD–ROM

over the network, using TFTP

5.3.2 Source Media and Installation Stages

The following table indicates which media sources you can use at each stage of the installation process. The columns indicate different install stages, ordered from left to right in the sequence which they occur. The far right column is the installation media. A blank cell indicates that given source media is not available at that installation stage; Y indicates that it is, and S means that it is in some cases.

Boot	Kernel Image	Drivers	Base System	Packages	media
S					tftp
S	Y	Y	Y		diskette
S	Y	Y	Y	Y	CD-ROM
S	Y	Y	Y	Y	hard disk
	Y	Y	Y	Y	NFS
		S	Y	Y	LAN
				Y	PPP

For example, the table shows that only use for PPP in the installation process is the installation of packages.

Note that you will only be prompted for a source for the kernel images and drivers in some installation methods. If you boot off a CD–ROM, it will automatically pick those items off the CD. The important point is that *as soon as you boot off a diskette, you can immediately switch to some superior installation source*. Remember, though, that you *must* not mix up the different install sets, i.e., using a Rescue Floppy from one subarchitecture and Driver Floppies from another.

The ‘Boot’ column is all Ss because media support for booting varies widely for different architectures.

The ‘LAN’ and ‘PPP’ rows refer to Internet–based file transfer (FTP, HTTP, and the like) over Ethernet or phone lines. In general this is not available, but certain kernels may permit you to do this earlier. Experts can also use these connections to mount disks and perform other operations to accelerate the process. Providing help in such cases is beyond the scope of this document.

5.3.3 Recommendations

Get a set of Debian GNU/Linux CDs. Boot off them if you can.

Since you’ve read this far, you probably couldn’t or wouldn’t. If your problem is simply that your CD drive is not bootable, you can pull the files you need for the initial boot off the CD and use them to make floppies or do a boot from alternate operating system.

Failing this, you may have an existing operating system with some free disk space. The early installation system can read many filesystems (NTFS being a prominent exception — you must load the appropriate driver). If it can read yours, you should download documentation, initial boot images, and utilities. Then get the appropriate drivers archive as a single file, and the base system as a single file. Perform your initial boot, and then point the installation program at the files you have downloaded when it asks for the appropriate source.

These are only suggestions. You should choose whatever sources are most convenient for you. Floppies are neither convenient nor reliable, so we urge you to get off them as soon as possible. However, compared to booting off an existing operating system they may provide a cleaner environment and an easier path, so they are appropriate for the initial boot, if your system supports them.

5.4 Description of Installation System Files

This section contains an annotated list of files you will find in the `disks-alpha` directory. You may not need to download these at all; it all depends on the booting and base system installation media you have chosen.

Most files are floppy disk images; that is, a single file which can be written to a disk to create the necessary floppy disk. These images are, obviously, dependent on the size of the target floppy. For instance, 1.44MB is the normal quantity of data which is what fits on standard 3.5 inch floppies. This is the only floppy size supported on your architecture. The images for 1.44MB floppy disks can be found in the `images-1.44` directory.

If you are using a web browser on a networked computer to read this document, you can probably retrieve the files by selecting their names in your web browser. Depending on your browser you may need to take special action to download directly to a file, in raw binary mode. For example, in Netscape you need to hold the shift key when clicking on the URL to retrieve the file. Files can be downloaded from the URLs in this document, or you can retrieve them from `http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/`, or the corresponding directory on any of the Debian mirror sites (`http://www.debian.org/distrib/ftplist`).

5.4.1 Documentation

Installation Manual:

`install.en.txt`

`install.en.html`

`install.en.pdf` This file you are now reading, in plain ASCII, HTML or PDF format.

Partitioning Program Manual Pages:

`fdisk.txt`

`cfdisk.txt` Instructions for using your available partitioning programs.

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base-cont`
Listing of the contents of the base system.

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/md5sum.tx`
: List of MD5 checksums for the binary files. If you have the `md5sum` program, you can ensure that your files are not corrupt by running `md5sum -v -c md5sum.txt`.

5.4.2 Files for the Initial System Boot

Rescue Floppy images:

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/jensen/im`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/nautilus/`
: These are the Rescue Floppy disk images. The Rescue Floppy is used for initial setup and for emergencies, such as when your system doesn't boot for some reason. Therefore it is recommended you write the disk image to the floppy even if you are not using floppies for installation.

Select the floppy image for your supported sub–architecture, as indicated in 'CPU, Mainboards, and Video Support' on page 8.

Root image(s):

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.`
: This file contains an image of a temporary filesystem that gets loaded into memory when you boot from the Rescue Floppy. This is used for installations from CD–ROM, hard disk and floppies.

Linux kernel:

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/linux`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/jensen/li`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/nautilus/`
: This is the Linux kernel image to be used for hard disk and CD installations. You don't need it if you are installing from floppies.

TFTP boot images

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/tftpboot.`
: Boot images used for network booting, see 'Booting from TFTP' on page 35. Generally, they contain the Linux kernel and the `root.bin` root filesystem.

5.4.3 Driver Files

These files contain kernel modules, or drivers, for all kinds of hardware that are not necessary for initial booting. Getting the drivers you want is a two step process: first you identify an archive of drivers you want to use, and then you select which particular drivers you want.

Remember that your driver archive must be consistent with your initial kernel choice.

Driver Floppies images:

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/jensen/im`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/nautilus/`

These are the Driver Floppies disk images.

Driver Floppies archive

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/drivers.t`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/jensen/dr`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/nautilus/`

If you are not limited to diskettes, choose one of these files.

5.4.4 Base System Files

The “Debian base system” is a core set of packages which are required to run Debian in a minimal, stand–alone fashion. Once you have configured and installed the base system, your machine can “stand on its own”.

Base system images:

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base2_2.t`

or

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.`

`http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.`

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/images-1.

These files contain the base system which will be installed on your Linux partition during the installation process. This is the bare minimum necessary for you to be able to install the rest of the packages. The `\path{http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base2_2.tgz}` file is for installation from non–floppy media, i.e., CD–ROM, harddisk, or NFS.

We turn now to concerns specific to particular kind of sources. For convenience, they appear in the same order as the rows in the earlier table discussing different installation sources.

5.5 TFTP

Booting from the network requires that you have a network connection supported by the boot floppies, a BOOTP server, and a TFTP server. This installation method is described in 'Booting from TFTP' on page 35.

5.6 Diskettes

5.6.1 Floppy Disk Reliability

The biggest problem for people installing Debian for the first time seems to be floppy disk reliability.

The Rescue Floppy is the floppy with the worst problems, because it is read by the hardware directly, before Linux boots. Often, the hardware doesn't read as reliably as the Linux floppy disk driver, and may just stop without printing an error message if it reads incorrect data. There can also be failures in the Driver Floppies and the base floppies, most of which indicate themselves with a flood of messages about disk I/O errors.

If you are having the installation stall at a particular floppy, the first thing you should do is re–download the floppy disk image and write it to a *different* floppy. Simply reformatting the old floppy may not be sufficient, even if it appears that the floppy was reformatted and written with no errors. It is sometimes useful to try writing the floppy on a different system.

One user reports he had to write the images to floppy *three* times before one worked, and then everything was fine with the third floppy.

Other users have reported that simply rebooting a few times with the same floppy in the floppy drive can lead to a successful boot. This is all due to buggy hardware or firmware floppy drivers.

5.6.2 Booting from Floppies

Booting from floppies is supported for most platforms.

To boot from floppies, simply download the Rescue Floppy image and the Driver Floppies image.

If you need to, you can also modify the Rescue Floppy; see 'Replacing the Rescue Floppy Kernel' on page 59.

The Rescue Floppy couldn't fit the root filesystem image, so you'll need the root image to be written to a disk as well. You can create that floppy just as the other images are written to floppies. Once the kernel has been loaded from the Rescue Floppy, you'll be prompted for the root disk. Insert that floppy and continue.

On Alpha, if you choose to boot from ARC console firmware using `MILO`, you will also need to obtain MILO and LINLOAD.EXE and write them to a DOS FAT formatted floppy disk. See 'Alpha Console Firmware' on page 38 for more information on Alpha firmware and bootloaders.

MILO binaries are platform–specific. See 'CPU, Mainboards, and Video Support' on page 8 to determine the appropriate MILO image for your Alpha.

5.6.3 Installing Base from Floppies

NOTE: This is not a recommended way of installing Debian, because floppies are generally the least reliable type of media. This is only recommended if you have no extra, pre–existing filesystems on any of the hard drives on your system.

Complete these steps:

1. Obtain these disk images (these files are described in greater detail in 'Description of Installation System Files' on page 24):

   a Rescue Floppy image

   MILO and LINLOAD.EXE binaries and a DOS FAT–formatted disk, if you choose to boot from the ARC console. LINLOAD.EXE is platform–independent, but MILO is specific to the particular Alpha model used. See 'CPU, Mainboards, and Video Support' on page 8 for more information on the different Alpha platforms.

   the Driver Floppies images

   the base system disk images, i.e., `base-1.bin`, `base-2.bin`, etc.

   and a root filesystem image

2. Locate sufficient floppies for all the images you need to write.
3. Create the floppies, as discussed in 'Creating Floppies from Disk Images' on this page.
4. Insert the Rescue Floppy into your floppy drive, and reboot the computer.
5. Skip down to 'Booting the Installation System' on page 33.

5.6.4 Creating Floppies from Disk Images

Disk images are files containing the complete contents of a floppy disk in *raw* form. Disk images, such as `rescue.bin`, cannot simply be copied to floppy drives. A special program is used to write the image files to floppy disk in *raw* mode. This is required because these images are raw representations of the disk; it is required to do a *sector copy* of the data from the file onto the floppy.

There are different techniques for creating floppies from disk images, which depend on your platform. This section describes how to create floppies from disk images for different platforms.

No matter which method you use to create your floppies, you should remember to flip the tab on the floppies once you have written them, to ensure they are not damaged unintentionally.

Writing Disk Images From a Linux or Unix System

To write the floppy disk image files to the floppy disks, you will probably need root access to the system. Place a good, blank floppy in the floppy drive. Next, use the command

```
dd if=file of=/dev/fd0 bs=1024 conv=sync ; sync
```

where *file* is one of the floppy disk image files. `/dev/fd0` is a commonly used name of the floppy disk device, it may be different on your workstation (on Solaris, it is `/dev/fd/0`). The command may return to the prompt before Unix has finished writing the floppy disk, so look for the disk–in–use light on the floppy drive and be sure that the light is out and the disk has stopped revolving before you remove it from the drive. On some systems, you'll have to run a command to eject the floppy from the drive (on Solaris, use `eject`, see the manual page).

Some systems attempt to automatically mount a floppy disk when you place it in the drive. You might have to disable this feature before the workstation will allow you to write a floppy in *raw mode*. Unfortunately, how to accomplish this will vary based on your operating system. On Solaris, you can work around volume management to get raw access to the floppy. First, make sure that the floppy is automounted (using `volcheck` or the equivalent command in the file manager). Then use a `dd` command of the form given above, just replace `/dev/fd0` with `/vol/rdsk/\textit{floppy_name}`, where *floppy_name* is the name the floppy disk was given when it was formatted (unnamed floppies default to the name `unnamed_floppy`). On other systems, ask your system administrator.

5.7 CD–ROM

CD–ROM booting is one of the easiest ways to install. If you're unlucky and the kernel on the CD–ROM doesn't work for you, you'll have to fall back to another technique.

Installing from CD–ROM is described in 'Installing from a CD–ROM' on page 35.

Note that certain CD drives may require special drivers, and so be inaccessible in the early installation stages.

5.8 Hard Disk

Booting from an existing operating system is often a convenient option; for some systems it is the only supported method of installation. This method is described in 'Booting from a Hard Disk' on page 34.

Exotic hardware or filesystems may render files on the hard disk inaccessible early in the installation process. If they aren't supported by the Linux kernel, they may be inaccessible even at the end!

5.9 Installing from NFS

Due to the nature of this method of installation, only the base system can be installed via NFS. You will need to have the Rescue Floppy and the Driver Floppies available locally using one of the above methods. To install the base system via NFS, you'll have to go through the regular installation as explained in 'Using `dbootstrap` for Initial System Configuration' on page 43. Do not forget to insert the module (driver) for your Ethernet card, and the file system module for NFS.

When `dbootstrap` asks you where the base system is located ("'Install the Base System"' on page 50), you should choose NFS, and follow the instructions.

Chapter 6

Booting the Installation System

This chapter begins with some general information about booting Debian GNU/Linux, then moves to individual sections on particular installation methods, and concludes with some troubleshooting advice.

6.1 Boot Parameter Arguments

Boot parameters are Linux kernel parameters which are generally used to make sure that peripherals are dealt with properly. For the most part, the kernel can auto–detect information about your peripherals. However, in some cases you'll have to help the kernel a bit.

Depending on the console firmware from which you will be bootstrapping, different methods apply for passing parameters to the kernel. These methods will be described below, separately for each bootstrap procedure. Full information on boot parameters can be found in the Linux BootPrompt HOWTO (`http://www.linuxdoc.org/HOWTO/BootPrompt-HOWTO.html`); this section contains only a sketch of the most salient parameters.

If this is the first time you're booting the system, try the default boot parameters (i.e., don't try setting arguments) and see if it works correctly. It probably will. If not, you can reboot later and look for any special parameters that inform the system about your hardware.

When the kernel boots, a message `Memory:` *`avail`*`k/`*`total`*`k available` should be emitted early in the process. *total* should match the total amount of RAM, in kilobytes. If this doesn't match the actual of RAM you have installed, you need to use the `mem=`*`ram`* parameter, where *ram* is set to the amount of memory, suffixed with "k" for kilobytes, or "m" for megabytes. For example, both `mem=65536k` and `mem=64m` mean 64MB of RAM.

If your monitor is only capable of black–and–white, use the `mono` boot argument. Otherwise, your installation will use color, which is the default.

If you are booting with a serial console, generally the kernel will autodetect this. If you have a videocard (framebuffer) and a keyboard also attached to the computer which you wish to boot via serial console, you may have to pass the `console=`*`device`* argument to the kernel, where *device* is your serial device, which is usually something like "ttyS0".

Again, full details on boot parameters can be found in the Linux BootPrompt HOWTO (`http://www.linuxdoc.org/HOWTO/BootPrompt-HOWTO.html`), including tips for obscure hardware. Some common gotchas are included below in 'Troubleshooting the Boot Process' on page 41.

6.1.1 `dbootstrap` Arguments

The installation system recognizes a few arguments which may be useful.

quiet This will cause the installation system to suppress confirmation messages and try to do the right thing without fuss. If you are familiar and comfortable with what the installation system is going to expect, this is a nice option to quieten the process.

verbose Ask even more questions than usual.

debug Emit additional debug messages to the installation system log (see 'Using the Shell and Viewing the Logs' on page 43), including every command run.

bootkbd=... Pre–select the keyboard you want to use, e.g., `bootkbd=qwerty/us`

mono Use monochrome rather than color mode.

6.2 Interpreting the Kernel Startup Messages

During the boot sequence, you may see many messages in the form `can't find something`, or `something not present`, `can't initialize something`, or even `this driver release depends on something`. Most of these messages are harmless. You see them because the kernel for the installation system is built to run on computers with many different peripheral devices. Obviously, no one computer will have every possible peripheral device, so the operating system may emit a few complaints while it looks for peripherals you don't own. You may also see the system pause for a while. This happens when it is waiting for a device to respond, and that device is not present on your system. If you find the time it takes to boot the system unacceptably long, you can create a custom kernel later (see 'Compiling a New Kernel' on page 56).

6.3 Booting from a Hard Disk

In some cases, you may wish to boot from an existing operating system. You can also boot into the installation system using other means, but install the base system from disk.

6.3.1 Installing from a Linux Partition

You can install Debian from an ext2fs partition or from a Minix partition. This installation technique may be appropriate if you are completely replacing your current Linux system with Debian, for instance.

Note that the partition you are installing *from* should not be the same as the partitions you are installing Debian *to* (e.g., `/`, `/usr`, `/lib`, etc.).

To install from an already existing Linux partition, follow these instructions.

1. Get the following files and place them in a directory on your Linux partition:

   a Rescue Floppy image, see 'Files for the Initial System Boot' on page 25

   one of the Driver Floppies archives from 'Driver Files' on page 26

   `http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base2_2.tgz`

2. You can use any other functional boot method when installing from a partition. The following assumes you are booting with floppies; however, any boot installation can be used.

3. Create the Rescue Floppy as discussed in 'Creating Floppies from Disk Images' on page 29. Note that you won't need the Driver Floppies.

4. Insert the Rescue Floppy into your floppy drive, and reboot the computer.

5. Skip down to 'Booting the Installation System' on page 33.

6.4 Installing from a CD–ROM

If you have a CD which is bootable, and if your architecture and system supports booting from a CD–ROM, you don't need any floppies. Booting from CD–ROM on Alpha is somewhat more involved than on i386. However, the reduction in the number of floppies required makes it worthwhile. See 'Booting the Installation System' on page 33 for more information on booting Alpha systems from CD and floppy.

Even if you cannot boot from CD–ROM, you can install the base Debian system from CD–ROM. Simply boot using a different media, such as floppies. When it is time to install the base system and any additional packages, point the installation system at the CD–ROM drive as described in "'Install the Base System"' on page 50.

6.5 Booting from TFTP

You need to setup a BOOTP server and a TFTP server.

BOOTP is an IP protocol that informs a computer of its IP address and where on the network to obtain a boot image. Unlike the Open Firmware found on Sparc and PowerPC machines, the SRM console will *not* use RARP to obtain its IP address, and therefore you must use BOOTP for netbooting your Alpha. You can also enter the IP configuration for network interfaces directly in the SRM console. [1]

[1] Alpha systems can also be net–booted using the DECNet MOP (Maintenance Operations Protocol), but this is not covered here. Presumably, your local OpenVMS operator will be happy to assist you should you have some burning need to use MOP to boot Linux on your Alpha.

The Trivial File Transfer Protocol (TFTP) is used to serve the boot image to the client. Theoretically, any server, on any platform, which implements these protocols, may be used. In the examples in this section, we shall provide commands for SunOS 4.x, SunOS 5.x (a.k.a. Solaris), and GNU/Linux.

6.5.1 Setting up BOOTP server

There are two BOOTP servers available for GNU/Linux, the CMU bootpd and the ISC dhcpd, which are contained in the `bootp` and `dhcp` packages on Debian GNU/Linux.

To use CMU bootpd, you must first uncomment (or add) the relevant line in `/etc/inetd.conf`. On Debian GNU/Linux, you can run `update-inetd --enable bootps`, then `/etc/init.d/inetd reload` to do so. Elsewhere, the line in question should look like:

```
    bootps          dgram   udp     wait    root    /usr/sbin/bootpd        boot
i -t 120
```

Now, you must create an `/etc/bootptab` file. This has the same sort of familiar and cryptic format as the good old BSD `printcap(5)`, `termcap(5)`, and `disktab(5)` files. See the `bootptab(5)` manual page for more information. For CMU bootpd, you will need to know the hardware (MAC) address of the client.

By contrast, setting up BOOTP with ISC `dhcpd` is really easy, because it treats BOOTP clients as a moderately special case of DHCP clients. You don't really need to know the hardware (MAC) address of the client unless you wish to specify some options such as boot image filename or NFS root path on a client–by–client basis, or unless you wish to assign fixed addresses to your machines using BOOTP and/or DHCP. Simply add the `allow bootp` directive to the configuration block for the subnet containing the client, and restart `dhcpd` with `/etc/init.d/dhcpd restart`.

6.5.2 Enabling the TFTP Server

To get the TFTP server ready to go, you should first make sure that `tftpd` is enabled. This is usually enabled by having the following line in `/etc/inetd.conf`:

```
    tftp dgram udp wait root /usr/etc/in.tftpd in.tftpd /tftpboot
```

Look in that file and remember the directory which is used as the argument of `in.tftpd`; you'll need that below. The `-l` argument enables some versions of `in.tftpd` to log all requests to the system logs; this is useful for diagnosing boot errors. If you've had to change `/etc/inetd.conf`, you'll have to notify the running `inetd` process that the file has changed. On a Debian machine, run `/etc/init.d/netbase reload` (for potato/2.2 and newer systems use `/etc/init.d/inetd reload`); on other machines, find out the process ID for `inetd`, and run `kill -HUP` *`inetd-pid`*.

6.5.3 Move TFTP Images Into Place

Next, place the TFTP boot image you need, as found in 'Description of Installation System Files' on page 24, in the `tftpd` boot image directory. Generally, this directory will be `/tftpboot`. Next you'll have to make a link from that file to the file which `tftpd` will use for booting a particular client. Unfortunately, the file name is determined by the TFTP client, and there are no strong standards.

Often, the file that the TFTP client will look for is *client–ip–in–hexclient–architecture*. To compute *client–ip–in–hex*, take each byte of the client IP address and translate it into hexadecimal notation. If you have a machine handy with the `bc` program, you can use the program. First issue the `obase=16` command to set the output to hex, then enter the individual components of the client IP one at a time. As for *client–architecture*, try out some values.

On Alpha, you must specify the filename (as a relative path to the boot image directory) using the `-file` argument to the SRM `boot` command, or by setting the `BOOT_FILE` environment variable. Alternatively, the filename can be given via BOOTP (in ISC dhcpd, use the `filename` directive). Unlike Open Firmware, there is *no default filename* on SRM, so you *must* specify a filename by either one of these methods.

Once you've determined the name, make the link like this: `ln /boot/tftpboot.img /boot/`*`file-name`*.

Now you should be ready to actually boot your system. In SRM, Ethernet interfaces are named with the `ewa` prefix, and will be listed in the output of the `show dev` command, like this (edited slightly):

```
>>>show dev
ewa0.0.0.9.0                EWA0              08-00-2B-86-98-65
ewb0.0.0.11.0               EWB0              08-00-2B-86-98-54
ewc0.0.0.2002.0             EWC0              00-06-2B-01-32-B0
```

So, to boot from the first Ethernet interface, you would type:

```
>>>boot ewa0
```

If you wish to use a serial console, you *must* pass the `console=` parameter to the kernel. This can be done using the `-flags` argument to the SRM `boot` command. The serial ports are named the same as their corresponding files in `/dev`. For example, to boot from `ewa0` and use a console on the first serial port, you would type:

```
>>>boot ewa0 -flags console=ttyS0
```

NOT YET WRITTEN

6.5.4 Installing with TFTP and NFS Root

It is closer to "tftp install for lowmem. . . " because you don't want to load the ramdisk anymore but boot from the newly created nfs–root fs. You then need to replace the symlink to the tftpboot image by a symlink to the kernel image (eg. linux–a.out). My experience on booting over the network was based exclusively on RARP/TFTP which requires all daemons running on the same server (the sparc workstation is sending a tftp request back to the server that replied to its previous rarp request). However, Linux supports BOOTP protocol, too, but I don't know how to set it up :–((Does it have to be documented as well in this manual?

6.6 Alpha Console Firmware

Console firmware is stored in a flash ROM and started when an Alpha system is powered up or reset. There are two different console specifications used on Alpha systems, and hence two classes of console firmware available:

SRM console, based on the Alpha Console Subsystem specification, which provides an operating environment for OpenVMS, Tru64 UNIX, and Linux operating systems.

ARC, AlphaBIOS, or ARCSBIOS console, based on the Advanced RISC Computing (ARC) specification, which provides an operating environment for Windows NT.

From the user's perspective, the most important difference between SRM and ARC is that the choice of console constrains the possible disk–partitioning scheme for the hard disk which you wish to boot off of.

ARC requires that you use an MS–DOS partition table (as created by `cfdisk`) for the boot disk. Therefore MS–DOS partition tables are the "native" partition format when booting from ARC. In fact, since AlphaBIOS contains a disk partitioning utility, you may prefer to partition your disks from the firmware menus before installing Linux.

Conversely, SRM is *incompatible* with MS–DOS partition tables. [2] Since Tru64 Unix uses the BSD disklabel format, this is the "native" partition format for SRM installations.

Because GNU/Linux is the only operating system on Alpha that can be booted from both console types, the choice will also depend on what other operating systems you wish to run on the same machine. All other Unix–like operating systems (Tru64 Unix, FreeBSD, OpenBSD, and NetBSD) and OpenVMS can only boot from SRM, whereas Windows NT can only boot from ARC.

The following table summarizes available and supported system type/console combinations (see 'CPU, Mainboards, and Video Support' on page 8 for the system type names). The word 'ARC' below denotes any of the ARC–compliant consoles.

[2] Specifically, the bootsector format required by the Console Subsystem Specification conflicts with the placement of the DOS partition table.

System Type	Console Type Supported
alcor	ARC or SRM
avanti	ARC or SRM
book1	SRM only
cabriolet	ARC or SRM
dp264	SRM only
eb164	ARC or SRM
eb64p	ARC or SRM
eb66	ARC or SRM
eb66p	ARC or SRM
jensen	SRM only
lx164	ARC or SRM
miata	ARC or SRM
mikasa	ARC or SRM
mikasa-p	SRM only
nautilus	ARC only (see motherboard manual)
noname	ARC or SRM
noritake	SRM only
noritake-p	SRM only
pc164	ARC or SRM
rawhide	SRM only
ruffian	ARC only
sable	SRM only
sable-g	SRM only
sx164	ARC or SRM
takara	ARC or SRM
xl	ARC only
xlt	ARC or SRM

Generally, none of these consoles can boot Linux directly, so the assistance of an intermediary bootloader is required. There are two mainstream Linux loaders: `MILO` and `aboot`.

`MILO` is itself a console, which replaces ARC or SRM in memory. `MILO` can be booted from both ARC and SRM and is the only way to bootstrap Linux from the ARC console. `MILO` is platform–specific (a different `MILO` is needed for each system type) and exist only for those systems, for which ARC support is shown in the table above. See also the (unfortunately outdated) MILO HOWTO (`http://www.linuxdoc.org/HOWTO/MILO-HOWTO.html`).

`aboot` is a small, platform–independent bootloader, which runs from SRM only. See the (also unfortunately outdated) SRM HOWTO (`http://www.linuxdoc.org/HOWTO/SRM-HOWTO/`) for more information on `aboot`.

Thus, three scenarios are generally possible, depending on the system's console firmware and whether or not `MILO` is available:

```
SRM -> aboot
SRM -> MILO
ARC -> MILO
```

The UP1000 motherboard (subarchitecture name ‘nautilus’) from Alpha Processor, Inc. is different from all the others, in that it uses an API–specific bootloader that runs under AlphaBIOS firmware. There are no install disks (yet) for the UP1000, but you should be able to install by booting a ‘generic’ or ‘nautilus’ kernel with the root.bin from the install disks, following the instructions in the manual.

Because `MILO` is not available for any of the Alpha systems currently in production (as of February 2000), and because it is no longer necessary to buy an OpenVMS or Tru64 Unix license to have SRM firmware on your older Alpha, it is recommended that you use SRM and `aboot` on new installations of GNU/Linux, unless you wish to dual–boot with Windows NT, or you have existing DOS–partitioned disks.

The majority of AlphaServers and all current server and workstation products contain both SRM and AlphaBIOS in their firmware. For “half–flash” machines such as the various evaluation boards, it is possible to switch from one version to another by reflashing the firmware. Also, once SRM is installed, it is possible to run ARC/AlphaBIOS from a floppy disk (using the ‘arc’ command).

As on other architectures, you should install the newest available revision of the firmware [3] before installing Debian. For Alpha, firmware updates can be obtained from Alpha Firmware Updates (`http://ftp.digital.com/pub/DEC/Alpha/firmware/`).

6.7 Booting From the SRM Console

At the SRM prompt (>>>), issue the following command:

```
>>> boot dva0 -flags 0
```

possibly replacing `dva0` with the actual device name. Usually, `dva0` is the floppy; type

```
>>> show dev
```

to see the list of devices (e.g., if you want to boot from a CD). Note that if you are booting via MILO, `-flags` argument is ignored, so you can just type `boot dva0`.

If everything works OK, you will eventually see the Linux kernel boot.

If you want to specify kernel parameters when booting via `aboot`, use the following command:

```
>>> boot dva0 -file linux.gz -flags
"root=/dev/fd0 load_ramdisk=1 arguments"
```

[3] Except on Jensen, where Linux is not supported on firmware versions newer than 1.7 – see `http://www.alphalinux.org/faq/FAQ-9.html` for more information

(typed on one line), substituting, if necessary, the actual SRM boot device name for `dva0`, the Linux boot device name for `fd0`, and the desired kernel parameters for `arguments`.

If you want to specify kernel parameters when booting via `MILO`, you will have to interrupt bootstrap once you get into MILO. See ‘Booting with MILO’ on the current page.

6.8 Booting from the ARC or AlphaBIOS Console

In the OS Selection menu, set `linload.exe` as the boot loader, and `milo` as the OS Path. Bootstrap using the newly created entry.

6.9 Booting with MILO

To bootstrap the installation system, enter the following command at the MILO prompt:

```
MILO> boot fd0:linux.gz root=/dev/fd0 load_ramdisk=1
```

If you are booting from something other than a floppy, substitute `fd0` in the above example with the appropriate device name in Linux notation. The `help` command would give you a brief MILO command reference.

6.10 Troubleshooting the Boot Process

If you have problems and the kernel hangs during the boot process, doesn’t recognize peripherals you actually have, or drives are not recognized properly, the first thing to check is the boot parameters, as discussed in ‘Boot Parameter Arguments’ on page 33.

Often, problems can be solved by removing add–ons and peripherals, and then trying booting again.

If you still have problems, please submit a bug report. Send an email to `<submit@bugs.debian.org>`. You *must* include the following as the first lines of the email:

```
Package: boot-floppies
Version: version
```

Make sure you fill in *version* with the version of the boot–floppies set that you used. If you don’t know the *version*, use the date you downloaded the floppies, and include the distribution you got them from (e.g., “stable”, “frozen”).

You should also include the following information in your bug report:

```
architecture:  alpha
model:         your general hardware vendor and model
memory:        amount of RAM
scsi:          SCSI host adapter, if any
cd-rom:        CD-ROM model and interface type, e.g., ATAPI
network card:  network interface card, if any
pcmcia:        details of any PCMCIA devices
```

Depending on the nature of the bug, it also might be useful to report whether you are installing to IDE or SCSI disks, other peripheral devices such as audio, disk capacity, and the model of video card.

In the bug report, describe what the problem is, including the last visible kernel messages in the event of a kernel hang. Describe the steps that you did which brought the system into the problem state.

Chapter 7

Using `dbootstrap` for Initial System Configuration

7.1 Introduction to `dbootstrap`

`dbootstrap` is the name of the program which is run after you have booted into the installation system. It is responsible for initial system configuration and the installation of the "base system".

The main job of `dbootstrap`, and the main purpose of your initial system configuration, is to configure essential elements of your system. For instance, you may need to use certain "kernel modules", which are drivers which are linked into the kernel. These modules include storage hardware drivers, network drivers, special language support, and support for other peripherals which are not automatically built in to the kernel you are using.

Disk partitioning, disk formatting, and networking setup are also handled by `dbootstrap`. This fundamental setup is done first, since it is often necessary for the proper functioning of your system.

`dbootstrap` is a simple, character–based application, designed for maximum compatability in all situations (such as installation over a serial line). It is very easy to use. It will guide you through each step of the installation process in a linear fashion. You can also go back and repeat steps if you find you have made a mistake.

Navigation within `dbootstrap` is accomplished with the arrow keys, *Enter*, and *Tab*.

7.1.1 Using the Shell and Viewing the Logs

If you are an experienced Unix or Linux user, press *Left Alt–F2* to get to the second *virtual console*. That's the *Alt* key on the left–hand side of the space bar, and the *F2* function key, at the same time. This is a separate window running a Bourne shell clone called `ash`. At this point you are booted from the RAM disk, and there is a limited set of Unix utilities available for your use. You can see what programs are available with the command `ls /bin /sbin /usr/bin /usr/sbin`. Use the menus to perform any task that they are able to do — the shell and commands are only there in case something goes wrong. In particular, you should always use the menus, not the shell, to activate your swap partition, because the menu software can’t detect that you’ve done this from the shell. Press *Left Alt–F1* to get back to menus. Linux provides up to 64 virtual consoles, although the Rescue Floppy only uses a few of them.

Error messages are redirected to the third virtual terminal (known as `tty3`). You can access this terminal by pressing *Left Alt–F3* (hold the *Alt* key while pressing the *F3* function key); get back to `dbootstrap` with *Left Alt–F1*.

These messages can also be found in `/var/log/messages`. After installation, this log is copied to `/var/log/installer.log` on your new system.

7.2 “Release Notes”

The first screen `dbootstrap` will present you with is the “Release Notes”. This screen presents the version information for the `boot-floppies` software you are using, and gives a brief introduction to Debian developers.

7.3 “Debian GNU/Linux Installation Main Menu”

You may see a dialog box that says “The installation program is determining the current state of your system and the next installation step that should be performed.”. On some systems, this will go by too quickly to read. You’ll see this dialog box between steps in the main menu. The installation program, `dbootstrap`, will check the state of the system in between each step. This checking allows you to re–start the installation without losing the work you have already done, in case you happen to halt your system in the middle of the installation process. If you have to restart an installation, you will have to configure your keyboard, re–activate your swap partition, and re–mount any disks that have been initialized. Anything else that you have done with the installation system will be saved.

During the entire installation process, you will be presented with the main menu, entitled “Debian GNU/Linux Installation Main Menu”. The choices at the top of the menu will change to indicate your progress in installing the system. Phil Hughes wrote in the Linux Journal (`http://www.linuxjournal.com/`) that you could teach a *chicken* to install Debian! He meant that the installation process was mostly just *pecking* at the *Enter* key. The first choice on the installation menu is the next action that you should perform according to what the system detects you have already done. It should say “Next”, and at this point the next step in installing the system will be taken.

7.4 “Configure the Keyboard”

Make sure the highlight is on the “Next” item, and press *Enter* to go to the keyboard configuration menu. Select a keyboard that conforms to the layout used for your national language, or select something close if the keyboard layout you want isn’t represented. Once the system installation is complete, you’ll be able to select a keyboard layout from a wider range of choices (run `kbdconfig` as root when you have completed the installation).

Move the highlight to the keyboard selection you desire and press *Enter*. Use the arrow keys to move the highlight — they are in the same place in all national language keyboard layouts, so they are independent of the keyboard configuration.

If you are installing a diskless workstation, the next few steps will be skipped, since there are no local disks to partition. In that case, your next step will be "'Configure the Network"' on page 49. After that, you will be prompted to mount your NFS root partition in "'Mount a Previously–Initialized Partition"' on page 47.

7.5 Preload Drivers

In certain unusual situations, you may wish to pre–load kernel modules from the floppy drive. Generally, you can ignore this alternate.

7.6 Last Chance!

Did we tell you to back up your disks? Here's your first chance to wipe out all of the data on your disks, and your last chance to save your old system. If you haven't backed up all of your disks, remove the floppy from the drive, reset the system, and run backups.

7.7 "Partition a Hard Disk"

If you have not already partitioned your disks for Linux native and Linux swap filesystems, i.e., as described in 'Partitioning Prior to Installation' on page 19, the next step will be "Partition a Hard Disk". If you have already created at least one Linux native and one Linux swap disk partition, the "Next" menu selection will be "Initialize and Activate a Swap Partition", or you may even skip that step if your system had low memory and you were asked to activate the swap partition as soon as the system started. Whatever the "Next" menu selection is, you can use the down–arrow key to select "Partition a Hard Disk".

The "Partition a Hard Disk" menu item presents you with a list of disk drives you can partition, and runs a partitioning application. You must create at least one "Linux native" (type 83) disk partition, and you probably want at least one "Linux swap" (type 82) partition, as explained in 'Partitioning Your Hard Drive' on page 15. If you are unsure how to partition your system, go back and read that chapter.

Depending on your architecture, there are different programs which can be used. These are the program or programs available on your architecture:

`fdisk` The original Linux disk partitioner, good for gurus; read the fdisk manual page (`fdisk.txt`).

`cfdisk` A simple–to–use, full–screen disk partitioner for the rest of us; read the cfdisk manual page (`cfdisk.txt`).

One of these programs will be run by default when you select “Partition a Hard Disk”. If the one which is run by default isn’t the one you want, quit the partitioner, go to the shell (tty2), and manually type in the name of the program you want to use (and arguments, if any). Then skip the “Partition a Hard Disk” step in `dbootstrap` and continue to the next step.

A swap partition is strongly recommended, but you can do without one if you insist, and if your system has more than 16MB RAM. If you wish to do this, please select the “Do Without a Swap Partition” item from the menu.

If you have chosen to boot from the SRM console, you must use `fdisk` to partition your disk, as it is the only partitioning program that can manipulate the BSD disklabels required by `aboot` (remember, the SRM boot block is incompatible with MS–DOS partition tables – see ‘Alpha Console Firmware’ on page 38). `dbootstrap` will run `fdisk` by default if you have not booted from `MILO`.

If the disk that you have selected for partitioning already contains a BSD disklabel, `fdisk` will default to BSD disklabel mode. Otherwise, you must use the ‘b’ command to enter disklabel mode.

Unless you wish to use the disk you are partitioning from Tru64 Unix or one of the free 4.4BSD–Lite derived operating systems (FreeBSD, OpenBSD, or NetBSD), it is suggested that you do *not* make the third partition contain the whole disk. This is not required by `aboot`, and in fact, it may lead to confusion since the `swriteboot` utility used to install `aboot` in the boot sector will complain about a partition overlapping with the boot block.

Also, because `aboot` is written to the first few sectors of the disk (currently it occupies about 70 kilobytes, or 150 sectors), you *must* leave enough empty space at the beginning of the disk for it. In the past, it was suggested that you make a small partition at the beginning of the disk, to be left unformatted. For the same reason mentioned above, we now suggest that you do not do this on disks that will only be used by GNU/Linux.

For ARC installations, you should make a small FAT partition at the beginning of the disk to contain `MILO` and `linload.exe` – a megabyte should be sufficient, see ‘Partitioning Prior to Installation’ on page 19. You will have to copy these files manually at present, by mounting the FAT partition under Linux, or by using `mtools`. You can use `mkdosfs` to create FAT partitions from a shell on the installation disk.

7.8 “Initialize and Activate a Swap Partition”

This will be the next step once you have created one disk partition. You have the choice of initializing and activating a new swap partition, activating a previously–initialized one, and doing without a swap partition. It’s always permissible to re–initialize a swap partition, so select “Initialize and Activate a Swap Partition” unless you are sure you know what you are doing.

This menu choice will first present you with a dialog box reading “Please select the partition to activate as a swap device.”. The default device presented should be the swap partition you’ve already set up; if so, just press *Enter*.

Next, there is a confirmation message, since initialization destroys any data previously on the partition. If all is well, select “Yes”. The screen will flash as the initialization program runs.

7.9 "Initialize a Linux Partition"

At this point, the next menu item presented should be "Initialize a Linux Partition". If it isn't, it is because you haven't completed the disk partitioning process, or you haven't made one of the menu choices dealing with your swap partition.

You can initialize a Linux partition, or alternately you can mount a previously–initialized one. Note that `dbootstrap` will *not* upgrade an old system without destroying it. If you're upgrading, Debian can usually upgrade itself, and you won't need to use `dbootstrap`. For help on upgrading to Debian 2.2, see the upgrade instructions (`http://www.debian.org/releases/2.2/alpha/release-notes/`).

Thus, if you are using old disk partitions that are not empty, i.e., if you want to just throw away what is on them, you should initialize them (which erases all files). Moreover, you must initialize any partitions that you created in the disk partitioning step. About the only reason to mount a partition without initializing it at this point would be to mount a partition upon which you have already performed some part of the installation process using this same set of installation floppies.

Select "Initialize a Linux Partition" to initialize and mount the `/` disk partition. The first partition that you mount or initialize will be the one mounted as `/` (pronounced "root").

You will be asked whether to preserve "Pre–2.2 Linux Kernel Compatibility?". Saying "No" here means that you cannot run 2.0 or earlier Linux kernels on your system, since the file systems enable some features not supported in the 2.0 kernel. If you know you'll never need to run a 2.0 or earlier vintage kernel, then you can achieve some minor benefits by saying "No" here. The default is "Yes" in the name of compatibility.

You will also be asked about whether to scan for bad blocks. The default here is to skip the bad block scan, since the scan can be very time consuming, and modern disk drive controllers internally detect and deal with bad blocks. However, if you are at all unsure about the quality of your disk drive, or if you have a rather old system, you should probably do the bad block scan.

The next prompts are just confirmation steps. You will be asked to confirm your action, since initializing is destructive to any data on the partition, and you will be informed that the partition is being mounted as `/`. (Technically, it's being mounted at `/target`; when you reboot into the system itself, that will become `/`.)

Once you've mounted the `/` partition, if you have additional file systems that you wish to initialize and mount, you should use the "Alternate" menu item. This is for those who have created separate partitions for `/boot`, `/var`, `/usr` or others, which ought to be initialized and mounted at this time.

7.10 "Mount a Previously–Initialized Partition"

An alternative to "'Initialize a Linux Partition"' on the current page is the "Mount a Previously–Initialized Partition" step. Use this if you are resuming an installation that was broken off, or if you want to mount partitions that have already been initialized or have data on it which you wish to preserve.

If you are installing a diskless workstation, at this point, you want to NFS mount your root partition from the remote NFS server. Specify the path to the NFS server in standard NFS syntax, namely, *`server-`*

`name-or-IP``:`*`server-share-path`*. If you need to mount additional filesystems as well, you can do that at this time.

If you have not already setup your network as described in "'Configure the Network"' on the facing page, then selecting an NFS install will prompt you to do so.

7.11 "Install Operating System Kernel and Modules"

The next step is to install a kernel and kernel modules onto your new system.

You will be offered a menu of devices from which you can install the kernel. Choose the appropriate device from which to install the kernel and modules. Remember that you can use any devices which is available to you, and that you are not restricted to using the same media you used to mount with (see 'Methods for Installing Debian' on page 21).

Note that the options presented to you will vary based on what hardware `dbootstrap` has detected. If you are installing from an official CD–ROM, the software should do the right thing automatically.

If you are installing from a local filesystem, you have a choice between two options. Select "harddisk" if the disk partition is not yet mounted; select "mounted" if it is. In both cases, you will be prompted to "Choose Debian archive path" — this is the directory within the disk where you have placed the required installation files discussed in 'Booting from a Hard Disk' on page 34. If you have a Debian archive mirrored locally, you can use that by giving the directory where that exists, which is often `/archive/debian`. Such archives are characterized by directory structures such as `debian/dists/stable/main/disks-alpha/current`. You can type in the path manually, or use the <`...`> button to browse through the filesystem tree.

Continuing the discusssion on installation from a local disk or similar medium (such as NFS), you will next be prompted for the actual directory containing the needed files (which may be based on your subarchitecture). Note that the system may be quite insistent that the files appear in the precise location indicated, including the subdirectories, if any. See the logs in tty3 (see 'Using the Shell and Viewing the Logs' on page 43) where `dbootstrap` will log the location of the files it's looking for.

If the "default" option appears, then you should use that. Otherwise, try the "list" option to let `dbootstrap` try to find the actual files on its own (but note that this can be very slow if you're mounting over NFS). As a last resort, use the "manual" option to specify the directory manually.

If you're installing from floppies, you'll need to feed in the Rescue Floppy (which is probably already in the drive), followed by the Driver Floppies.

If you wish to install the kernel and modules over the network, you can do this using the "network" (HTTP) or "nfs" options. Your networking interfaces must be supported by the standard kernel (see 'Peripherals and Other Hardware' on page 11). If these "nfs" options don't appear, you need to select "Cancel", then go back and select the "Configure the Network" step (see "'Configure the Network"' on the next page), and then re–run this step.

7.11.1 NFS

Select the “nfs” option, and then tell `dbootstrap` your NFS server name and path. Assuming you’ve put the Rescue Floppy and Driver Floppies images on the NFS server in the proper location, these files should be available to you for installing the kernel and modules. The NFS filesystem will be mounted under `/instmnt`. Select the location of the files as for “harddisk” or “mounted”.

7.11.2 Network

Select the “network” option, and then tell `dbootstrap` the URL and path to the Debian archive. The default will usually work fine, and in any case, the path part is probably correct for any official Debian mirror, even if you edit the server part. You may choose to pull the files in through a proxy server; just enter the server **. . . this sentence isn’t finished. . .**

7.11.3 NFS Root

If you are installing a diskless workstation, you should have already configured your networking as described in “‘Configure the Network”’ on this page. You should be given the option to install the kernel and modules from NFS. Proceed using the “nfs” option described above.

Other steps may need to be taken for other installation media.

7.12 “Configure Device Driver Modules”

Select the “Configure Device Driver Modules” menu item and look for devices that are on your system. Configure those device drivers, and they will be loaded whenever your system boots.

You don’t have to configure all your devices at this point; what is crucial is that any device configuration required for the installation of the base system is done here. This includes Ethernet drivers.

At any point after the system is installed, you can reconfigure your modules with the `modconf` program.

7.13 “Configure the Network”

You’ll have to configure the network even if you don’t have a network, but you’ll only have to answer the first two questions — “Choose the Host name”, and “Is your system connected to a network?”.

If you are connected to a network, you’ll need the information you collected from ‘Information You Will Need’ on page 13. However, if your primary connection to the network will be PPP, you should choose *NOT* to configure the network.

`dbootstrap` will ask you a number of questions about your network; fill in the answers from ‘Information You Will Need’ on page 13. The system will also summarize your network information and ask you for confirmation. Next, you need to specify the network device that your primary network connection uses. Usually, this will be “eth0” (the first Ethernet device).

Some technical details you might, or might not, find handy: the program assumes the network IP address is the bitwise–AND of your system’s IP address and your netmask. It will guess the broadcast address is the bitwise OR of your system’s IP address with the bitwise negation of the netmask. It will guess that your gateway system is also your DNS server. If you can’t find any of these answers, use the system’s guesses — you can change them once the system has been installed, if necessary, by editing `/etc/network/interfaces`.

7.14 “Install the Base System”

During the “Install the Base System” step, you’ll be offered a menu of devices from which you may install the base system. You should select the appropriate device.

If you choose to install from a filesystem on the harddisk or from CD–ROM, you will be prompted to specify the path to the `http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base2_2.tgz` file. If you have official media, the default value should be correct. Otherwise, enter the path where the base system can be found, relative to the media’s mount point. As with the “Install Operating System Kernel and Modules” step, you can either let `dbootstrap` find the file itself or type in the path at the prompt.

If you choose to install from floppy disk, feed in the base floppies in order, as requested by `dbootstrap`. If one of the base floppies is unreadable, you’ll have to create a replacement floppy and feed all floppies into the system again. Once the floppies have all been read, the system will install the files it had read from the floppies. This could take 10 minutes or more on slow systems, less on faster ones.

If you are installing the base system from NFS, then choose NFS and continue. You’ll be prompted to specify the server, the share on the server, and the subdirectory within that share where the `\path{http://http.us.debian.org/debian/dists/potato/main/disks-alpha/current/base2_2.tgz}` file can be found. If you have problems mounting NFS, make sure that the system time on the NFS server more or less agrees with the system time on the client. You can set your date on `tty2` using the `date` command; you’ll have to set it by hand. See the `date(1)` manual page.

7.15 “Configure the Base System”

At this point you’ve read in all of the files that make up a minimal Debian system, but you must perform some configuration before the system will run.

You’ll be asked to select your time zone. There are many ways to specify your time zone; we suggest you go to the “Directories:” pane and select your country (or continent). That will change the available time zones, so go ahead and select your geographic locality (i.e., country, province, state, or city) in the “Timezones:” pane.

Next, you’ll be asked if your system clock is to be set to GMT or local time. Select GMT (i.e., “Yes”) if you will only be running Unix on your computer; select local time (i.e., “No”) if you will be running another operating system as well as Debian. Unix (and Linux is no exception) generally keeps GMT time on the system clock and converts visible time to the local time zone. This allows the system to keep track of daylight savings time and leap years, and even allows users who are logged in from other time zones to individually set the time zone used on their terminal.

7.16 “Make Linux Bootable Directly From Hard Disk”

If you elect to make the hard disk boot directly to Linux, and you are *not* installing a diskless workstation, you will be asked to install a master boot record. If you aren’t using a boot manager (and this is probably the case if you don’t know what a boot manager is) and you don’t have another different operating system on the same machine, answer “Yes” to this question. If you answer “Yes”, the next question will be whether you want to boot Linux automatically from the hard disk when you turn on your system. This sets Linux root partition to be the *bootable partition* — the one that will be loaded from the hard disk.

Note that multiple operating system booting on a single machine is still something of a black art. This document does not even attempt to document the various boot managers, which vary by architecture and even by subarchitecture. You should see your boot manager’s documentation for more information. Remember: when working with the boot manager, you can never be too careful.

If you have booted from SRM, if you select this option, the installer will write `aboot` to the first sector of the disk on which you installed Debian. Be *very* careful – it is *not* possible to boot multiple operating systems (e.g. GNU/Linux, Free/Open/NetBSD, OSF/1 a.k.a. Digital Unix a.k.a. Tru64 Unix, or OpenVMS) from the same disk. If you also have a different operating system installed on the disk where you have installed Debian, you will have to boot GNU/Linux from a floppy instead.

If you are installing a diskless workstation, obviously, booting off the local disk isn’t a meaningful option, and this step will be skipped.

7.17 The Moment of Truth

You system’s first boot on its own power is what electrical engineers call the “smoke test”. If you have any floppies in your floppy drive, remove them. Select the “Reboot the System” menu item.

If are booting directly into Debian, and the system doesn’t start up, either use your original installation boot media (for instance, the Rescue Floppy), or insert the Custom Boot floppy if you created one, and reset your system. If you are *not* using the Custom Boot floppy, you will probably need to add some boot arguments. If booting with the Rescue Floppy or similar technique, you need to specify `rescue root=`*`root`*, where *root* is your root partition, such as “/dev/sda1”.

Debian should boot, and you should see the same messages as when you first booted the installation system, followed by some new messages.

7.18 Set the Root Password

The *root* account is also called the *super–user*; it is a login that bypasses all security protection on your system. The root account should only be used to perform system administration, and only used for as short a time as possible.

Any password you create should contain from 6 to 8 characters, and should contain both upper– and lower–case characters, as well as punctuation characters. Take extra care when setting your root password, since it is such a powerful account. Avoid dictionary words or use of any personal information which could be guessed.

If anyone ever tells you they need your root password, be extremely wary. You should normally never give your root account out, unless you are administering a machine with more than one system administrator.

7.19 Create an Ordinary User

The system will ask you to create an ordinary user account. This account should be your main personal log–in. You should *not* use the root account for daily use or as your personal login.

Why not? Well, one reason to avoid using root's privileges is that it is very easy to do irreparable damage as root. Another reason is that you might be tricked into running a *Trojan–horse* program — that is a program that takes advantage of your super–user powers to compromise the security of your system behind your back. Any good book on Unix system administration will cover this topic in more detail — consider reading one if it is new to you.

Name the user account anything you like. If your name is John Smith, you might use "smith", "john", "jsmith" or "js".

7.20 Shadow Password Support

Next, the system will ask whether you want to enable shadow passwords. This is a system in which your Linux system is made to be a bit more secure. In a system without shadow passwords, passwords are stored (encrypted) in a world–readable file, `/etc/passwd`. This file has to be readable to anyone who can log in because it contains vital user information, for instance, how to map between numeric user identifiers and login names. Therefore, someone could conceivably grab your `/etc/passwd` file and run a brute force attack (i.e. run an automated test of all possible password combinations) against it to try to determine passwords.

If you have shadow passwords enabled, passwords are instead stored in `/etc/shadow`, which is readable and writable only by root, and readable by group shadow. Therefore, we recommend that you enable shadow passwords.

Reconfiguration of the shadow password system can be done at any time with the `shadowconfig` program. After installation, see `/usr/share/doc/passwd/README.debian.gz` for more information.

7.21 Select and Install Profiles

The system will now ask you if you want to use the pre–rolled software configurations offered by Debian. You can always choose, package by package, what do you want to install on your new machine. This is the purpose of the `dselect` program, described below. But this can be a long task with around 3750 packages available in Debian!

So, you have the ability to choose *tasks* or *profiles* instead. A *task* is a work you will do with the machine such as “Perl programming” or “HTML authoring” or “Chinese word processing”. You can choose several tasks. A *profile* is a category your machine will be a member of such as “Network server” or “Personal workstation”. Unlike the tasks, you can choose only one profile.

In summary, if you are in a hurry, choose one profile. If you have more time, choose the Custom profile and select a set of tasks. If you have plenty of time and want very precise control on what is or is not installed, skip this step and use the full power of `dselect`.

Soon, you will enter into `dselect`. If you selected tasks or profiles, remember to skip the “Select” step of `dselect`, since the selections have already been made.

A word of warning about the size of the tasks as they are displayed: the size shown for each task is the sum of the sizes of its packages. If you choose two tasks that share some packages, the actual disk requirement will be less than the sum of the sizes for the two tasks.

Once you’ve added both logins (root and personal), you’ll be dropped into the `dselect` program. The dselect Tutorial (`dselect-beginner`) is required reading before you run `dselect`. `dselect` allows you to select *packages* to be installed on your system. If you have a CD–ROM or hard disk containing the additional Debian packages that you want to install on your system, or you are connected to the Internet, this will be useful to you right away. Otherwise, you may want to quit `dselect` and start it later, once you have transported the Debian package files to your system. You must be the super–user (root) when you run `dselect`.

7.22 Log In

After you’ve quit `dselect`, you’ll be presented with the login prompt. Log in using the personal login and password you selected. Your system is now ready to use.

7.23 Setting up PPP

NOTE: In case you are installing from CD–ROM and/or are connected directly to the network, you can safely skip this section. The installation system will only prompt you for this information if the network hasn’t been configured yet.

The base system includes a full `ppp` package. This package allows you to connect to your ISP using PPP. Below are some basic instructions for setting up your PPP connection. The boot disks contain a program called `pppconfig` which will help you set up PPP. *Make sure, when it asks you for the name of your dialup connection, that you name it “provider”.*

Hopefully, the `pppconfig` program will walk you through a pain–free PPP connection setup. However, if it does not work for you, see below for detailed instructions.

In order to setup PPP, you'll need to know the basics of file viewing and editing in Linux. To view files, you should use `more`, and `zmore` for compressed files with a `.gz` extension. For example, to view `README.debian.gz`, type `zmore README.debian.gz`. The base system comes with two editors: `ae`, which is very simple to use, but does not have a lot of features, and `elvis-tiny`, a limited clone of `vi`. You will probably want to install more full–featured editors and viewers later, such as `nvi`, `less`, and `emacs`.

Edit `/etc/ppp/peers/provider` and replace "/dev/modem" with "/dev/ttyS#" where # stands for the number of your serial port. In Linux, serial ports are counted from 0; your first serial port is `/dev/ttyS0` under Linux. The next step is to edit `/etc/chatscripts/provider` and insert your provider's phone number, your user–name and password. Please do not delete the "\q" that precedes the password. It hides the password from appearing in your log files.

Many providers use PAP or CHAP for login sequence instead of text mode authentication. Others use both. If your provider requires PAP or CHAP, you'll need to follow a different procedure. Comment out everything below the dialing string (the one that starts with "ATDT") in `/etc/chatscripts/provider`, modify `/etc/ppp/peers/provider` as described above, and add `user` *`name`* where *name* stands for your user–name for the provider you are trying to connect to. Next, edit `/etc/ppp/pap-secrets` or `/etc/ppp/chap-secrets` and enter your password there.

You will also need to edit `/etc/resolv.conf` and add your provider's name server (DNS) IP addresses. The lines in `/etc/resolv.conf` are in the following format: `nameserver` *`xxx.xxx.xxx.xxx`* where the *x*s stand for numbers in your IP address. Optionally, you could add the `usepeerdns` option to the `/etc/ppp/peers/provider` file, which will enable automatic choosing of appropriate DNS servers, using settings the remote host usually provides.

Unless your provider has a login sequence different from the majority of ISPs, you are done! Start the PPP connection by typing `pon` as root, and monitor the process using `plog` command. To disconnect, use `poff`, again, as root.

Read `/usr/share/doc/ppp/README.Debian.gz` file for more information on using PPP on Debian.

7.24 Installing the Rest of Your System

Information about the installation of the rest of your Debian system is contained in a separate document, the dselect Tutorial (`dselect-beginner`). Remember to skip the "Select" step in `dselect` if you are using the profiles and tasks from 'Select and Install Profiles' on the page before.

Chapter 8

Next Steps and Where to Go From Here

8.1 If You Are New to Unix

If you are new to Unix, you probably should go out and buy some books and do some reading. The Unix FAQ (`ftp://rtfm.mit.edu/pub/usenet/news.answers/unix-faq/faq/`) contains a number of references to books and Usenet news groups which should help you out. You can also take a look at the User-Friendly Unix FAQ (`http://www.camelcity.com/~noel/usenet/cuuf-FAQ.htm`).

Linux is an implementation of Unix. The Linux Documentation Project (LDP) (`http://www.linuxdoc.org/`) collects a number of HOWTOs and online books relating to Linux. Most of these documents can be installed locally; just install the `doc-linux-html` package (HTML versions) or the `doc-linux-text` package (ASCII versions), then look in `/usr/doc/HOWTO`. International versions of the LDP HOWTOs are also available as Debian packages.

Information specific to Debian can be found below.

8.2 Orienting Yourself to Debian

Debian is a little different from other distributions. Even if you're familiar with Linux in other distributions, there are things you should know about Debian to help you to keep your system in a good, clean state. This chapter contains material to help you get oriented; it is not intended to be a tutorial for how to use Debian, but just a very brief glimpse of the system for the very rushed.

The most important concept to grasp is the Debian packaging system. In essence, large parts of your system should be considered under the control of the packaging system. These include:

`/usr` (excluding `/usr/local`)

`/var` (you could make `/var/local` and be safe in there)

`/bin`

`/sbin`

`/lib`

For instance, if you replace `/usr/bin/perl`, that will work, but then if you upgrade your `perl` package, the file you put there will be replaced. Experts can get around this by putting packages on "hold" in `dselect`.

8.3 Further Reading and Information

If you need information about a particular program, you should first try `man` *`program`*, or `info` *`program`*.

There is lots of useful documentation in `/usr/doc` as well. In particular, `/usr/doc/HOWTO` and `/usr/doc/FAQ` contain lots of interesting information.

The Debian web site (`http://www.debian.org/`) contains a large quantity of documentation about Debian. In particular, see the Debian FAQ (`http://www.debian.org/doc/FAQ/`) and the Debian Mailing List Archives (`http://www.debian.org/Lists-Archives/`). The Debian community is self–supporting; to subscribe to one or more of the Debian mailing lists, see the Mail List Subscription (`http://www.debian.org/MailingLists/subscribe`) page.

8.4 Compiling a New Kernel

Why would someone want to compile a new kernel? It is often not necessary since the default kernel shipped with Debian handles most configurations. However, it is useful to compile a new kernel in order to:

- handle special hardware needs, or hardware conflicts with the pre–supplied kernels
- handle hardware or options not included in the stock kernel, such as APM or SMP
- optimize the kernel by removing useless drivers to speed up boot time
- use options of the kernel which are not supported by the default kernel (such as network firewalling)
- run a updated or development kernel
- impress your friends, try new things

Don't be afraid to try compiling the kernel. It's fun and profitable.

To compile a kernel the Debian way, you need some packages: `kernel-package`, `kernel-source-2.2.18pre21` (the most recent version at the time of this writing), `fakeroot` and a few others which are probably already installed (see `/usr/share/doc/kernel-package/README.gz` for the complete list).

Note that you don't *have* to compile your kernel the "Debian way"; but we find that using the packaging system to manage your kernel is actually safer and easier. In fact, you can get your kernel sources right from Linus instead of `kernel-source-2.2.18pre21`, yet still use the `kernel-package` compilation method.

Note that you'll find complete documentation on using `kernel-package` under `/usr/share/doc/kernel-package`. This section just contains a brief tutorial.

Hereafter, we'll assume your kernel source will be located in `/usr/local/src` and that your kernel version is 2.2.18pre21. As root, create a directory under `/usr/local/src` and change the owner of that directory to your normal non–root account. As your normal non–root account, change your directory to where you want to unpack the kernel sources (`cd /usr/local/src`), extract the kernel sources (`tar xIf /usr/src/kernel-source-2.2.18pre21.tar.bz2`), change your directory to it (`cd kernel-source-2.2.18pre21/`). Now, you can configure your kernel. Run `make xconfig` if X11 is installed, configured and being run, `make menuconfig` otherwise (you'll need `ncurses-dev` installed). Take the time to read the online help and choose carefully. When in doubt, it is typically better to include the device driver (the software which manages hardware peripherals, such as Ethernet cards, SCSI controllers, and so on) you are unsure about. Be careful: other options, not related to a specific hardware, should be left at the default value if you do not understand them. Do not forget to select "Kernel module loader" in "Loadable module support" and "Enhanced Real Time Clock Support" in "Character devices" (they are not selected by default). If not included, your Debian installation will experience problems. If you don't need the full power of the Real Time Clock driver, you may want (and are encouraged) to apply the Light-weight RTC patch (`ftp://genie.ucd.ie/pub/alpha/patches-2.2.x/rtclight-2.2.12.diff`) and then select the light–weight option in the kernel configuration.

Clean the source tree and reset the `kernel-package` parameters. To do that, do `make-kpkg clean`.

Now, compile the kernel: `fakeroot make-kpkg --revision=custom.1.0 kernel_image`. The version number of "1.0" can be changed at will; this is just a version number that you will use to track your kernel builds. Likewise, you can put any word you like in place of "custom" (e.g., a host name). Kernel compilation may take quite a while, depending on the power of your machine.

Once the compilation is complete, you can install your custom kernel like any package. As root, do `dpkg -i ../kernel-image-2.2.18pre21-`*`subarch`*`_custom.1.0_alpha.deb`. The *subarch* part is an optional sub–architecture, depending on what kernel options you set. `dpkg -i kernel-image...` will install the kernel, along with some other nice supporting files. For instance, the `System.map` will be properly installed (helpful for debugging kernel problems), and `/boot/config-2.2.18pre21` will be installed, containing your current configuration set. Your new `kernel-image-2.2.18pre21` package is also clever enough to automatically use you're platform's boot–loader to run an update on the booting, allowing you to boot without re–running the boot loader. If you have created a modules package, e.g., if you have PCMCIA, you'll need to install that package as well.

It is time to reboot the system: read carefully any warning that the above step may have produced, then `shutdown -r now`.

For more information on `kernel-package`, read documentation in `/usr/doc/kernel-package`.

Chapter 9

Technical Information on the Boot Floppies

9.1 Source Code

The `boot-floppies` package contains all of the source code and documentation for the installation floppies.

9.2 Rescue Floppy

The Rescue Floppy has an Ext2 filesystem (or a FAT filesystem, depending on your architecture), and you should be able to access it from anything else that can mount Ext2 or FAT disks. The Linux kernel is in the file `linux`. The file `root.bin` is a `gzip`–compressed disk image of a 1.4MB Minix or Ext2 filesystem, and will be loaded into the RAM disk and used as the root filesystem.

9.3 Replacing the Rescue Floppy Kernel

If you find it necessary to replace the kernel on the Rescue Floppy, you must configure your new kernel with these features linked in, not in loadable modules:

- RAM disk support (`CONFIG_BLK_DEV_RAM`)
- Initial RAM disk (initrd) support (`CONFIG_BLK_DEV_INITRD`)
- Kernel support for ELF binaries (`CONFIG_BINFMT_ELF`)
- Loop device support (`CONFIG_BLK_DEV_LOOP`)
- FAT, Minix, and Ext2 filesystems (some architectures don't need FAT and/or Minix filesystems — see the source)

Documentation not complete, text missing.

You’ll also want to replace the `modules.tgz` file on the Driver Floppies. This file simply contains a `gzip`–compressed tar file of `/lib/modules/`*`kernel-ver`*; make it from the root filesystem so that all leading directories are in the tar file as well.

9.4 The Base Floppies

The base floppies contain a 512–byte header followed by a portion of a gzip–compressed `tar` archive. If you strip off the headers and then concatenate the contents of the base floppies, the result should be the compressed tar archive. The archive contains the base system that will be installed on your hard disk.

Once this archive is installed, you must go through the steps described in “‘Configure the Base System”’ on page 50, and other `dbootstrap` menu items to configure the network, and you must install the operating system kernel and modules on your own. Once you have done that, the system should be usable.

As for the post–installation tasks, those are mostly handled by the `base-config` package.

Chapter 10

Appendix

10.1 Further Information and Obtaining Debian GNU/Linux

10.1.1 Further Information

A general source of information on Linux is the Linux Documentation Project (`http://www.linuxdoc.org/`). There you will find the HOWTOs and pointers to other very valuable information on parts of a GNU/Linux system.

10.1.2 Obtaining Debian GNU/Linux

If you want to buy a CD set to install Debian GNU/Linux system from CD–ROM you should look at the CD vendors page (`http://www.debian.org/distrib/vendors`). There you get a list of addresses which sell Debian GNU/Linux on CD–ROMs. The list is sorted by country so you shouldn't have a problem to find a vendor near you.

10.1.3 Debian Mirrors

If you live outside of the USA and you want to download Debian packages, you can also use one of many mirrors which reside outside the USA. A list of countries and mirrors can be found at the Debian FTP server website (`http://www.debian.org/distrib/ftplist`).

10.1.4 GPG, SSH and other Security Software

United States laws place restrictions on the export of defense articles, which, unfortunately, includes some types of cryptographic software. PGP and ssh, among others, fall into this category. It is legal however, to import such software into the US.

To prevent anyone from taking unnecessary legal risks, some Debian packages are available from a server outside the US which serves the various cryptographic programs: Debian non-US Server (`ftp://nonus.debian.org/debian-non-US/`).

This text is taken from the README.non–US file, which you can find on any Debian FTP archive mirror. It also contains a list of mirrors of the non–US server.

10.2 Linux Devices

In Linux you have various special files in `/dev`. These files are called devices files. In the Unix world accessing hardware is different. There you have a special file which actually runs a driver which in turn accesses the hardware. The device file is an interface to the actual system component. Files under `/dev` also behave differently than ordinary files. Below are the most important device files listed.

```
fd0 1. Floppy Drive
fd1 2. Floppy Drive

hda IDE Harddisk / CD-ROM on the first IDE port (Master)
hdb IDE Harddisk / CD-ROM on the first IDE port (Slave)
hdc IDE Harddisk / CD-ROM on the second IDE port (Master)
hdd IDE Harddisk / CD-ROM on the second IDE port (Slave)
hda1 1. partition of the first IDE harddisk
hdd15 15. partition of the fourth IDE harddisk

sda SCSI Harddisk with lowest SCSI ID (e.g. 0)
sdb SCSI Harddisk with next higher SCSI ID (e.g. 1)
sdc SCSI Harddisk with next higher SCSI ID (e.g. 2)
sda1 1. partition of the first SCSI harddisk
sdd10 10. partition of the fourth SCSI harddisk

sr0     SCSI CD-ROM with the lowest SCSI ID
sr1     SCSI CD-ROM with the next higher SCSI ID

ttyS0    Serial port 0, COM1 under DOS
ttyS1    Serial port 1, COM2 under DOS
psaux    PS/2 mouse device
gpmdata  Pseudo device, repeater data from GPM (mouse) daemon

cdrom Symbolic link to the CD-ROM drive
mouse Symbolic link to the mouse device file

null everything pointed to this device will disappear
zero one can endlessly read zeros out of this device
```

Chapter 11

Administrivia

11.1 About This Document

This document is written in SGML, using the “DebianDoc” DTD. Output formats are generated by programs from the `debiandoc-sgml` package.

In order to increase the maintainability of this document, we use a number of SGML features, such as entities and marked sections. These play a role akin to variables and conditionals in programming languages. The SGML source to this document contains information for each different architecture — marked sections are used to isolate certain bits of text as architecture–specific.

11.2 Contributing to This Document

If you have problems or suggestions regarding this document, you should probably submit them as a bug report against the package `boot-floppies`. See the `bug` or `reportbug` package or read the online documentation of the Debian Bug Tracking System (`http://www.debian.org/Bugs/`). It would be nice if you could check the open bugs against boot-floppies (`http://www.debian.org/Bugs/db/pa/lboot-floppies.html`) to see whether your problem has already been reported. If so, you can supply addition corroboration or helpful information to `<XXXX@bugs.debian.org>`, where *XXXX* is the number for the already–reported bug.

Better yet, get a copy of the SGML source for this document, and produce patches against it. The SGML source can be found in the `boot-floppies`; try to find the newest revision in the unstable (`ftp://ftp.debian.org/debian/dists/unstable/`) distribution. You can also browse the source via CVSweb (`http://cvs.debian.org/boot-floppies/`); for instructions on how to check out the sources via CVS, see README-CVS (`http://cvs.debian.org/~checkout~/boot-floppies/README-CVS?tag=HEAD%26content-type=text/plain`) from the sources.

Please do *not* contact the authors of this document directly. There is also a discussion list for `boot-floppies`, which includes discussions of this manual. The mailing list is `<debian-boot@lists.debian.org>`. Instructions for subscribing to this list can be found at the Debian Mailing List Subscription (`http://`

`www.debian.org/MailingLists/subscribe`) page; an online browsable copy can be found at the Debian Mailing List Archives (`http://www.debian.org/Lists-Archives/`).

11.3 Major Contributions

Many, many Debian users and developers contributed to this document. Particular note must be made for Michael Schmitz (m68k support), Frank Neumann (original author of the Debian Installation Instructions for Amiga (`http://www.informatik.uni-oldenburg.de/~amigo/debian_inst.html`)), Arto Astala, Eric Delaunay/Ben Collins (SPARC information), Tapio Lehtonen, and Stéphane Bortzmeyer for numerous edits and text.

Extremely helpful text and information was found in Jim Mintha's HOWTO for network booting (no URL available), the Debian FAQ (`http://www.debian.org/doc/FAQ/`), the Linux/m68k FAQ (`http://www.linux-m68k.org/faq/faq.html`), the Linux for SPARC Processors FAQ (`http://www.ultralinux.org/faq.html`), the Linux/Alpha FAQ (`http://www.alphalinux.org/faq/FAQ.html`), amongst others. The maintainers of these freely available and rich sources of information must be recognized.

11.4 Trademark Acknowledgement

All trademarks are property of their respective trademark owners.