Installing Debian GNU/Linux 3.0 For SPARC

Bruce Perens
Sven Rudolph
Igor Grobman
James Treacy
Adam Di Carlo

version 3.0.23, 15 May, 2002

Abstract

This document contains installation instructions for the Debian GNU/Linux 3.0 system, for the SPARC ("sparc") 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 3.0 (`http://www.debian.org/releases/woody/sparc/release-notes/`).

Copyright Notice

This document may be distributed and modified under the terms of the GNU General Public License.
© 1996 Bruce Perens
© 1996, 1997 Sven Rudolph
© 1998 Igor Grobman, James Treacy
© 1998–2002 Adam Di Carlo

This manual is free software; you may redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This manual is distributed in the hope that it will be useful, but *without any warranty*; without even the implied warranty of merchantability or fitness for a particular purpose. See the GNU General Public License for more details.

A copy of the GNU General Public License is available as `/usr/share/common-licenses/GPL` in the Debian GNU/Linux distribution or on the World Wide Web at the GNU website (`http://www.gnu.org/copyleft/gpl.html`). You can also obtain it by writing to the Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

We require that you properly attribute Debian and the authors of this document on any materials derived from this document. If you modify and improve this document, we request that you notify the authors of this document, via `<debian-boot@lists.debian.org>`.

Contents

Chapter 1

Welcome to Debian

We are delighted that you have decided to try Debian, and are 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.fsf.org/fsf/fsf.html`) and influenced by the GNU (`http://www.gnu.org/gnu/the-gnu-project.html`) philosophy, has grown over the years into an organization of around 800 *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 The Open Source Definition (http://opensource.org/docs/definition_plain.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 file system. 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™ 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.zork.net/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 3.0 pages (http://www.debian.org/releases/woody/) for any last-minute information about the 3.0 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/woody/sparc/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, perform any necessary planning and hardware configuration prior to installing Debian, in ‘Before Installing Debian GNU/Linux’ on page 13. If you are preparing a multi-boot system, you may need to create partition-able space on your hard disk for Debian to use.

3. In ‘Obtaining System Installation Media’ on page 23, you will obtain the necessary installation files for your method of installation.

4. ‘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.

5. Setting up the Linux partitions for your Debian system is explained in ‘Partitioning for Debian’ on page 41.
6. Install the kernel and configure peripheral driver modules in ‘Installing the Kernel and Base Operating System’ on page 49. Configure your network connection so that remaining installation files can be obtained directly from a Debian server, if you are not installing from a CD.
7. Initiate automatic download/install/setup of a minimal working system in “‘Install the Base System”’ on page 52.
8. Boot into your newly installed base system and run through some additional configuration tasks, from ‘Booting Into Your New Debian System’ on page 53.
9. Install additional software in ‘Package Installation: Simple or Advanced’ on page 57. Use `tasksel` to install groups of packages which form a computer ‘task’, `dselect` to select individual packages from a long list, or `apt-get` to install individual packages when you already know the package names you want.

Once you’ve got your system installed, you can read ‘Next Steps and Where to Go From Here’ on page 61. 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 67.

Finally, information about this document and how to contribute to it may be found in ‘Administrivia’ on page 77.

1.8 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.

Working versions of this document can be found at `http://www.debian.org/releases/woody/sparc/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 77. 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/share/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. Please refer to the Ports pages at `http://www.debian.org/ports/sparc/` for more details on sparc architecture systems which have been tested with Debian.

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

2.1.1 Supported Architectures

Debian 3.0 supports eleven major architectures and several variations of each architecture known as 'flavors'.

```
Architecture          | Debian Designation / Flavor
----------------------+----------------------------
Intel x86-based       | i386
                      |   - vanilla
                      |   - idepci
                      |   - compact
                      |   - bf2.4 (experimental)
                      |
Motorola 680x0:       | m68k
```

```
  - Atari             |   - atari
  - Amiga             |   - amiga
  - 68k Macintosh     |   - mac
  - VME               |   - bvme6000
                      |   - mvme147
                      |   - mvme16x
                      |
DEC Alpha             | alpha
                      |   - generic
                      |   - jensen
                      |   - nautilus
                      |
Sun SPARC             | sparc
                      |   - sun4cdm
                      |   - sun4u
                      |
ARM and StrongARM     | arm
                      |   - netwinder
                      |   - riscpc
                      |   - shark
                      |   - lart
                      |
IBM/Motorola PowerPC  | powerpc
  - CHRP              |   - chrp
  - PowerMac          |   - powermac, new-powermac
  - PReP              |   - prep
  - APUS              |   - apus
                      |
HP PA-RISC            | hppa
  - PA-RISC 1.1       |   - 32
  - PA-RISC 2.0       |   - 64
                      |
Intel ia64-based      | ia64
                      |
MIPS (big endian)     | mips
  - SGI Indy/I2       |  - r4k-ip22
                      |
MIPS (little endian)  | mipsel
  - DEC Decstation    |  - r4k-kn04
                      |  - r3k-kn02
                      |
IBM S/390             | s390
                      |  - tape
                      |  - vmrdr
                      |
----------------------+-----------------------------
```

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

2.1.2 CPU, Main Boards, and Video Support

Currently the *sparc* port supports several types of Sparc systems. The most common identifiers for Sparc systems are sun4, sun4c, sun4m, sun4d and sun4u. Currently we do not support very old sun4 hardware. However, the other systems are supported. Sun4d has been tested the least of these, so expect possible problems with regard to the kernel stability. Sun4c and Sun4m, the most common of the older Sparc hardware, includes such systems as SparcStation 1, 1+, IPC, IPX and the SparcStation LX, 5, 10, and 20, respectively. The UltraSPARC class systems fall under the sun4u identifier, and are supported using the sun4u set of install images. Some systems that fall under these supported identifiers are known to not be supported. Known unsupported systems are the AP1000 multicomputer and the Tadpole Sparcbook 1. See the Linux for SPARCProcessors FAQ (http://www.ultralinux.org/faq.html) for complete information.

Memory Configuration

Some older Sun workstations, notably the Sun IPX and Sun IPC have memory banks located at fixed locations in physical memory. Thus if the banks are not filled gaps will exist in the physical memory space. The Linux installation requires a contiguous memory block into which to load the kernel and the initial RAMdisk. If this is not available a 'Data Access Exception' will result.

Thus you must configure the memory so that the lowest memory block is contiguous for at least 8Mb. In the IPX and IPC cited above, memory banks are mapped in at 16Mb boundaries. In effect this means that you must have a sufficiently large SIMM in bank zero to hold the kernel and RAMdisk. In this case 4Mb is *not* sufficient.

Example: In a Sun IPX you have a 16Mb SIMM and a 4Mb SIMM. There are four SIMM banks (0,1,2,3). [Bank zero is that furthest away from the SBUS connectors]. You must therefore install the 16Mb SIMM in bank 0; it is then recommended to install the 4Mb SIMM in bank 2.

Graphics Configuration

Especially in the case of older Sun workstations, it is very common for there to be an onboard framebuffer which has been superseded (for example the bwtwo on a sun IPC), and an SBUS card containing a later probably accelerated buffer is then plugged in to an SBUS slot. Under Solaris/SunOS this causes no problems because both cards are initialised.

However with Linux this can cause a problem, in that the boot PROM monitor may display its output on this additional card; however the linux kernel boot messages may then be directed to the original on board framebuffer, leaving *no* error messages on the screen, with the machine apparently stuck loading the RAMdisk.

To avoid this problem, connect the monitor (if required) to the video card in the lowest numbered SBUS slot (on motherboard card counts as below external slots). Alternatively it is possible to use a serial console.

Graphics Card

Debian's support for graphical interfaces is determined by the underlying support found in XFree86's X11 system. The newer AGP video slots are actually a modification on the PCI specification, and most AGP video cards work under XFree86. Details on supported graphics buses, cards, monitors, and pointing devices can be found at `http://www.xfree86.org/`. Debian 3.0 ships with X11 revision 4.1.0.

2.1.3 Multiple Processors

Multi-processor support — also called "symmetric multi-processing" or SMP — is supported for this architecture. However, the standard Debian 3.0 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 63. At this time (kernel version 2.2.20) the way you enable SMP is to select "symmetric multi-processing" in the "General" section of the kernel config.

2.2 Installation Media

In many cases, you'll have to do your first boot from floppy disks, 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 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 'Booting from a CD-ROM' on page 34.

Installation system booting from a hard disk is another option for many architectures. Although the SPARC does not allow booting from SunOS (Solaris), you can install from a SunOS partiton (UFS slices).

You can also *boot* your system over the network. Diskless installation, using network booting from a local area network and NFS-mounting of all local filesystems, is another option — you'll probably need at least 16MB of RAM for a diskless installation. After the operating system kernel is installed, you can install the rest of your system via any sort of network connection (including PPP after installation of the base system), via FTP, HTTP, or NFS.

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, which includes many drivers that won't be used for your machine (see 'Compiling a New Kernel' on page 63 to learn how to build your own kernel). Support for the widest possible range of devices is desirable in general, 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:

- Sparc ESP
- PTI Qlogic,ISP
- Adaptec AIC7xxx
- NCR and Symbios 53C8XX

IDE systems (such as the UltraSPARC 5) are also supported. See Linux for SPARC Processors FAQ (`http://www.ultralinux.org/faq.html`) for more information on SPARC hardware supported by the Linux kernel.

2.3 Memory and Disk Space Requirements

You must have at least 12MB of memory and 110MB of hard disk space. For a minimal console-based system (all standard packages), 250MB is required. 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 400MB. 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 Network Connectivity Hardware

The following network interface cards (NICs) are supported from the bootable kernel directly:

- Sun LANCE
- Sun Happy Meal

The following network interface cards are supported as modules. They can be enabled once the drivers are installed during the setup. However, due to the magic of OpenPROM, you still should be able to boot from these devices:

- Sun BigMAC
- Sun QuadEthernet
- MyriCOM Gigabit Ethernet

2.5 Peripherals and Other Hardware

Linux supports a large variety of hardware devices such as mice, printers, scanners, PCMCIA and USB devices. However, most of these devices are not 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.

2.6 Purchasing Hardware Specifically for GNU/Linux

There are several vendors, 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.

2.6.1 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.

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 Installing Debian GNU/Linux

3.1 Overview of the Installation Process

Here’s a road map for the steps you will take during the installation process.

1. Create partition-able space for Debian on your hard disk
2. Locate and/or download kernel and driver files (except Debian CD users)
3. Set up boot floppies or place boot files (except most Debian CD users can boot from one of the CDs)
4. Boot the installation system
5. Configure the keyboard
6. Create and mount Debian partitions
7. Point the installer to the location of the kernel and drivers
8. Select which peripheral drivers to load
9. Configure the network interface
10. Initiate automatic download/install/setup of the base system
11. Configure Linux or multi-system boot loading
12. Boot the newly installed system and do some final configuration
13. Install additional tasks and packages, at your discretion

3.2 Back Up Your Existing Data!

Before you start, make sure to back up every file that is now on your system. If this is the first time a non-native operating system has been installed on your computer, it's quite likely you will need to re-partition your disk to make room for Debian GNU/Linux. Anytime you partition your disk, you should count on losing everything on the disk, no matter what program you use to do it. The programs used in installation are quite reliable and most have seen years of use; but they are also quite powerful and 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.

If you are creating a multi-boot system, make sure that you have the distribution media of any other present operating systems on hand. Especially if you repartition your boot drive, you might find that you have to reinstall your operating system's boot loader, or in many cases the whole operating system itself and all files on the affected partitions.

3.3 Information You Will Need

3.3.1 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.

dselect Tutorial (`dselect-beginner.en.html`) Tutorial for using the `dselect` program. This is one means of installing addition packages onto your system after the basic install is complete.

Linux for SPARC Processors FAQ (`http://www.ultralinux.org/faq.html`)

Partitioning Program Manual Pages:

`fdisk.txt` Manual pages for the partitioning software used during the installation process.

.../current/md5sum.txt (`../md5sum.txt`) 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`.

3.3.2 Finding Sources of Hardware Information

Hardware information can be gathered from:

- The manuals that come with each piece of hardware.

- The BIOS setup screens of your computer. You can view these screens when you start your computer by pressing a combination of keys. Check your manual for the combination. Often, it is the Delete key.
- The cases and boxes for each piece of hardware.
- System commands or tools in another operating system, including file manager displays. This source is especially useful for information about RAM and hard drive memory.
- Your system administrator or Internet Service Provider. These sources can tell you the settings you need to set up your networking and e-mail.

Hardware Information Needed for an Install

Hardware	Information You Might Need
Hard Drives	* How many you have. * Their order on the system. * Whether IDE or SCSI (most computers are IDE). * Available free space. * Partitions. * Partitions where other operating systems are installed.
Monitor	* Model and manufacturer. * Resolutions supported. * Horizontal refresh rate. * Vertical refresh rate. * Color depth (number of colors) supported. * Screen size.
Mouse	* Type: serial, PS, or USB. * Port. * Manufacturer. * Number of buttons.
Network	* Model and manufacturer. * Type of adapter.
Printer	* Model and manufacturer. * Printing resolutions supported.
Video Card	* Model and manufacturer. * Video RAM available. * Resolutions and color depths supported (these should be checked against your monitor’s capabilities).

3.3.3 Hardware Compatibility

Many brand name products work without trouble on Linux. Moreover, hardware for Linux is improving daily. However, Linux still does not run as many different types of hardware as some operating systems.

You can check hardware compatibility by:

- Checking manufacturers' web sites for new drivers.
- Looking at web sites or manuals for information about emulation. Lesser known brands can sometimes use the drivers or settings for better-known ones.
- Checking hardware compatibility lists for Linux on web sites dedicated to your architecture.
- Searching the Internet for other users' experiences.

3.3.4 Network Settings

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. On the other hand, if your administrator tells you that a DHCP server is available and is recommended, then you don't need this information because the DHCP server will provide it directly to your computer during the installation process.

- 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 will not be able to install the base system over the network. To install the system in this case, you must use a CD, pre-load the base packages on an existing hard disk partition, or prepare floppy disks containing the base packages. See 'Setting Up PPP' on page 55 below for information on setting up PPP under Debian once the system is installed.

3.4 Planning Use of the System

It is important to decide what type of machine you are creating. This will determine the disk space requirements for your Debian system.

3.5 Meeting Minimum Hardware Requirements

Once you have gathered information about your computer's hardware, check that your hardware will let you do the type of installation that you want to do.

Depending on your needs, you might manage with less than some of the recommended hardware listed in the table below. However, most users risk being frustrated if they ignore these suggestions.

Recommended Minimum System Requirements

Install Type	RAM	Hard Drive
No desktop	16 megabytes	450 megabytes
With Desktop	64 megabytes	1 gigabyte
Server	128 megabytes	4 gigabytes

Here is a sampling of some common Debian system configurations. You can also get an idea of the disk space used by related groups of programs by referring to 'Disk Space Needed for Tasks' on page 73.

Standard Server This is a small server profile, useful for a stripped down server which does not have a lot of niceties for shell users. It includes an FTP server, a web server, DNS, NIS, and POP. For these 50MB of disk space would suffice, and then you would need to add space for any data you serve up.

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

Work Console 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.

Developer 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`.

3.6 Pre-Partitioning for Multi-Boot Systems

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 and want to stick Linux on the same disk, you will need to repartition the disk. Debian requires its own hard disk partitions. It cannot be installed on Windows or MacOS partitions. It may be able to share some partitions with other Linux systems, but that's not covered here. At the very least you will need a dedicated partition for the Debian root.

You can find information about your current partition setup by using a partitioning tool for your current operating system . Partitioning tools always provide a way to show existing partitions without making changes.

In general, changing a partition with a file system 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.

If your computer has more than one hard disk, you may want to dedicate one of the hard disks completely to Debian. If so, you don't need to partition that disk before booting the installation system; the installer's included partitioning program can handle the job nicely.

If your machine has only one hard disk, and you would like to completely replace the current operating system with Debian GNU/Linux, you also can wait to partition as part of the installation process ('Partitioning for Debian' on page 41), after you have booted the installation system. However this only works if you plan to boot the installer system from floppies, CD-ROM or files on a connected machine. Consider: if you boot from files placed on the hard disk, and then partition that same hard disk within the installation system, thus erasing the boot files, you'd better hope the installation is successful the first time around. At the least in this case, you should have some alternate means of reviving your machine like the original system's installation floppies or CDs.

If your machine already has multiple partitions, and enough space can be provided by deleting and replacing one or more of them, then you too can wait and use the Debian installer's partitioning program. You should still read through the material below, because there may be special circumstances like the order of the existing partitions within the partition map, that force you to partition before installing anyway.

In all other cases, you'll need to partition your hard disk before starting the installation to create partitionable space for Debian. If some of the partitions will be owned by other operating systems, you should create those partitions using native operating system partitioning programs. We recommend that you do *not* attempt to create Debian Linux partitions using another operating system's tools. Instead, you should just create the native operating system's partitions you will want to retain.

If you are going to install more than one operating system on the same machine, you should install all other system(s) before proceeding with Linux installation. Windows and other OS installations may destroy your ability to start Linux, or encourage you to reformat non-native partitions.

You can recover from these actions or avoid them, but installing the native system first saves you trouble.

If you currently have one hard disk with one partition (a common setup for desktop computers), and you want to multi-boot the native operating system and Debian, you will need to:

1. Back up everything on the computer.
2. Boot from the native operating system installer media such as CD-ROM or floppies.
3. Use the native partitioning tools to create native system partition(s). Leave either a place holder partition or free space for Debian GNU/Linux.
4. Install the native operating system on its new partition.
5. Boot back into the native system to verify everything's OK, and to download the Debian installer boot files.
6. Boot the Debian installer to continue installing Debian.

3.6.1 Partitioning from SunOS

It's perfectly fine to partition from SunOS; in fact, if you intend to run both SunOS and Debian on the same machine, it is recommended that you partition using SunOS prior to installing Debian. The Linux kernel understands Sun disk labels, so there are no problems there. Just make sure you leave room for the Debian root partition within the first 1GB area of the boot disk. You can also place the kernel image on a UFS partition if that is easier than putting the root partition there. SILO supports booting Linux and SunOS from either EXT2 (Linux), UFS (SunOS), romfs and iso9660 (CDROM) partitions.

3.6.2 Partitioning from Linux or another OS

Whatever system you are using to partition, make sure you create a "Sun disk label" on your boot disk. This is the only kind of partition scheme that the OpenBoot PROM understands, and so it's the only scheme from which you can boot. In `fdisk`, the *s* key is used to create Sun disk labels. You only need to do this on drives that do not already have a Sun disk label. If you are using a drive that was previously formatted using a PC (or other architecture) you must create a new disk label, or problems with the disk geometry will most likely occur.

You will probably be using `SILO` as your boot loader (the small program which runs the operating system kernel). `SILO` has certain requirements for partition sizes and location; see 'Partitioning for Debian' on page 41.

3.7 Pre-Installation Hardware and Operating System Setup

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). Known hardware issues affecting the reliability of Debian GNU/Linux on your system are also highlighted.

3.7.1 Invoking OpenBoot

OpenBoot provides the basic functions needed to boot the SPARC architecture. This is rather similar in function to the BIOS in the x86 architecture, although much nicer. The Sun boot PROMs have a built-in forth interpreter which lets you do quite a number of things with your machine, such as diagnostics, simple scripts, etc.

To get to the boot prompt you need to hold down the *Stop* key (on older type 4 keyboards, use the *L1* key, if you have a PC keyboard adapter, use the *Break* key) and press the *A* key. The boot PROM will give you a prompt, either `ok` or >. It is preferred to have the `ok` prompt. So if you get the old style prompt, hit the 'n' key to get the new style prompt.

3.7.2 Boot Device Selection

You can use OpenBoot to boot from specific devices, and also to change your default boot device. However, you need to know some details about how OpenBoot names devices; it's much different from Linux device naming, described in 'Device Names in Linux' on page 44. Also, the command will vary a bit, depending on what version of OpenBoot you have. More information about OpenBoot can be found in the Sun OpenBoot Reference (`http://docs.sun.com/ab2/coll.216.1/@Ab2CollView?Ab2Lang=C%26Ab2Enc=iso-8859-1%26DwebQuery=OpenBOOT`).

Typically, with newer revisions, you can use OpenBoot device such as "floppy", "cdrom", "net", "disk", or "disk2". These have the obvious meanings; the "net" device is for booting from the network. Additionally, the device name can specify a particular partition of a disk, such as "disk2:a" to boot disk2, first partition. Full OpenBoot device names have the form

driver-name@unit-address:device-arguments

. In older revisions of OpenBoot, device naming is a bit different: the floppy device is called "/fd", and SCSI disk devices are of the form "sd(*controller*, *disk-target-id*, *disk-lun*)". The command `show-devs` in newer OpenBoot revisions is useful for viewing the currently configured devices. For full information, whatever your revision, see the Sun OpenBoot Reference (`http://docs.sun.com/ab2/coll.216.1/@Ab2CollView?Ab2Lang=C%26Ab2Enc=iso-8859-1%26DwebQuery=OpenBOOT`).

To boot from a specific device, use the command `boot` *`device`*. You can set this behavior as the default using the `setenv` command. However, the name of the variable to set changed between OpenBoot revisions. In OpenBoot 1.x, use the command `setenv boot-from` *`device`*. In later revisions of OpenBoot, use the command `setenv boot-device` *`device`*. Note, this is also configurable using the `eeprom` command on Solaris, or modifying the appropriate files in `/proc/openprom/options/`, for example under Linux:

```
echo disk1:1 >/proc/openprom/options/boot-device
```

and under Solaris:

```
eeprom boot-device=disk1:1
```

3.7.3 Hardware Issues to Watch Out For

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.

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

Obtaining System Installation Media

4.1 Official Debian GNU/Linux CD-ROM Sets

By far the easiest way to install Debian GNU/Linux is from an Official Debian CD-ROM Set (see the CD vendors page (`http://www.debian.org/CD/vendors/`)). You may also download the CD-ROM images from the Debian server and make your own set, if you have a fast network connection and a CD burner. If you have a Debian CD set and CDs are bootable on your machine, you can skip right to 'Booting from a CD-ROM' on page 34; much effort has been expended to ensure the files most people need are there on the CD.

If your machine doesn't support CD booting, but you do have a CD set, you can use an alternative strategy (floppy disk, hard disk, or net boot) to initially boot the system installer. The files you need for booting by another means are also on the CD; the Debian network archive and CD folder organization are identical. So when archive file paths are given below for particular files you need for booting, look for those files in the same directories and subdirectories on your CD.

Once the installer is booted, it will be able to obtain all the other files it needs from the CD.

If you don't have a CD set, then you will need to download the installer system files and place them either on your hard disk, floppy disk or a connected computer so they can be used to boot the installer.

4.2 Downloading Files from Debian Mirrors

When downloading files from a Debian mirror, be sure to download the files in *binary* mode, not text or automatic mode. It's important to replicate the directory structure you find on the mirror to create a local 'sub-mirror'. It isn't really necessary to do this if you place all the installation files on floppies; but it still makes it easier to find the files when you need them. You should start your local directory structure at the level under `disks-sparc`, for example:

```
current/subarchitecture/images-1.44/flavor/rescue.bin
```

You don't need to download every file under that level, just those that apply to you (you'll find out which ones apply as you read on). Just name the directories the same as the mirror's, and keep the files in their proper directories.

If your machine is set up to automatically decompress/decode files you download, you must turn that feature off when downloading the installation system files. They will be decompressed just-in-time by the installer. Decompressing in your current system will waste space and time, and if the original compressed archives are deleted by the decompression program, they won't be there later when the installer needs them.

4.2.1 Installation Options

Files you may need fall into three categories:

1. Files needed to boot into the installation system (for example, `rescue.bin`, `linux.bin`, and `root.bin`)
2. Files the installation system will need access to after it has been booted in order to install the operating system kernel and peripheral drivers (for example, `rescue.bin` and `drivers.tgz`)
3. Base system installation files (for example, `basedebs.tar`)

If you have a working Ethernet connection on the computer, and your Ethernet card is of one of the types compiled into the installation kernel, you may only need the install system boot files. The installer is capable of installing the kernel and drivers over the network for many common Ethernet cards.

If you have an Ethernet connection for which the installer doesn't have built-in support, you may need both the install system boot files and the kernel and peripheral driver installation files.

If you are installing on a system without a working network connection, or if your network connection is via PPP (using a modem) rather than Ethernet, you will need to obtain all three types of files before starting the installation.

If you're not sure which files you need, just start with the install system boot files. If your first attempt to configure the network within the installer fails, you can just quit, get the extra files you need, and re-start the installation.

The base system installation file `basedebs.tar` is currently about 27M. If you are able to use a CD, or configure your network before installing the base system, it is better to do so; in that case you won't need this file. The network location is listed in the appendix ('Debian Base System Installation Files' on page 71).

To use a current debian system to assemble a `basedebs.tar` from the debian archives, first install `debootstrap` (`apt-get install debootstrap`). Then use the following command:

```
debootstrap binary-basedebs SUITE=woody VERSION=3.0 \
  MIRROR="http://ftp.debian.org/debian" ARCHES="sparc"
```

4.2.2 Choosing the Right Installation Set

Installation files include kernel images, which are available for various "subarchitectures". Each subarchitecture supports a different set of hardware. The subarchitectures available for SPARC are:

'sun4cmd' This is the kernel required for older SPARC hardware. For specific models supported, see 'CPU, Main Boards, and Video Support' on page 9.

'sun4u' UltraSPARC machines.

The kernel config files for these subarchitectures can be found in their respective directories in a file named `kernel-config`.

4.2.3 Where to Find Installation Files

The network locations of installation files for each sparc flavor are listed in the Appendix. These include:

.../current/sun4cdm/images-1.44/rescue.bin (`../sun4cdm/images-1.44/rescue.bin`)

.../current/sun4u/images-1.44/rescue.bin (`../sun4u/images-1.44/rescue.bin`) rescue image

.../current/images-1.44/root.bin (`../images-1.44/root.bin`) root image(s) or tarball

'Linux Kernel Files' on page 71 kernel binary

'Driver Files' on page 71 driver images or tarball

'Debian Base System Installation Files' on page 71 base system images or tarball

The rescue image contains a compressed Linux boot kernel. It is used for both floppy disk booting (when transferred to a floppy) and as the source for the Linux kernel when the kernel is being installed on your machine. The kernel binary `linux.bin` is an uncompressed binary kernel. It is used when booting the installer from the hard disk or CD-ROM, and is not needed for floppy installer booting.

Refer to 'Creating Floppies from Disk Images' on the next page for important information on properly creating floppy disks from floppy images.

The root floppy image contains a compressed RAMdisk filesystem which gets loaded into memory after you boot the installer.

The peripheral drivers may be downloaded as a series of floppy images or as a tarball (`drivers.tgz`). The installer system will need access to the drivers file during installation. If you have a hard drive partition or connected computer which will be accessible to the installer (see below), the tarball will be more convenient to handle. The floppy image files are needed only if you must install the drivers from floppies.

When downloading files, you should also pay attention to the type of file system you are downloading them *to*, unless you will use floppies for the kernel and drivers. The installer can read files from many kinds of file systems, including FAT, HFS, ext2fs, and Minix. When downloading files to a *nix file system, choose the largest possible files from the archive.

During the installation, you will erase the partition(s) on which you are installing Debian before beginning the installation. All downloaded files must be placed on partitions *other* than those on which you are planning to install the system.

4.3 Creating Floppies from Disk Images

Bootable floppy disks are commonly used to boot the installer system for machines with a floppy drive. Floppies can also be used for installation of the kernel and modules on most systems.

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 on 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.

4.3.1 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 auto-mounted (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/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.

4.3.2 Writing Disk Images From DOS, Windows, or OS/2

If you have access to an i386 machine, you can use one of the following programs to copy images to floppies.

The FDVOL, WrtDsk or RaWrite3 programs can be used under MS-DOS.

`http://www.minix-vmd.org/pub/Minix-vmd/dosutil/`

To use these programs, first make sure that you are booted into DOS. Trying to use these programs from within a DOS box in Windows, or double-clicking on these programs from the Windows Explorer is *not* expected to work. If you don't know how to boot into DOS, just hit *F8* while booting.

`NTRawrite` is an attempt to create a contemporary version of `Rawrite/Rawrite3` that is consistently compatible with WinNT, Win2K and Win95/98. It is a self-explanatory GUI application; you select the disk drive to write to, browse to the disk image you want to place there and hit the Write button.

`http://sourceforge.net/projects/ntrawrite/`

4.3.3 Modifying the Rescue Floppy to Support National Language

The messages shown by the rescue floppy (before loading the Linux kernel) can be shown in your mother tongue. To achieve this if you are not an English speaker, after writing the image file, you must copy the provided message files and a font to the floppy. For MS-DOS and Windows users there is a batch file `setlang.bat` in the `dosutils` directory, which copies the correct files. Simply enter this directory (e.g.

```
cd
c:\debian\dosutils
```

) within a command prompt window, and run `setlang` *`lang`*, where *lang* is a two-letter code of your language in lower case, for example `setlang pl` to set the language to Polish. Currently these language codes are available:

```
ca cs da de eo es fi fr gl hr hu it ko ja pl pt ru sk sv tr zh_CN
```

Note that the descriptions in this manual assume that you use non localized (English) installation; otherwise the names of menus and buttons will differ from what you will see on your screen.

4.4 Preparing Files for TFTP Net Booting

If your machine is connected to a local area network, you may be able to boot it over the network from another machine, using TFTP. If you intend to boot the installation system from another machine, the boot files will need to be placed in specific locations on that machine, and the machine configured to support booting of your specific machine.

You need to setup a TFTP server, and for CATS machines, a BOOTP server , or RARP server, or DHCP server.

The Reverse Address Resolution Protocol (RARP) is one way to tell your client what IP address to use for itself. Another way is to use the BOOTP protocol. BOOTP is an IP protocol that informs a computer of its IP address and where on the network to obtain a boot image. The DHCP (Dynamic Host Configuration Protocol) is a more flexible, backwards-compatible extension of BOOTP. Some systems can only be configured via DHCP.

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.

4.4.1 Setting up RARP server

To setup RARP, you need to know the Ethernet address of the client (a.k.a. the MAC address). If you don't know this information, you can pick it off the initial OpenPROM boot messages, use the OpenBoot `.enet-addr` command, or boot into "Rescue" mode (e.g., from the rescue floppy) and use the command `/sbin/ifconfig eth0`.

On systems using a Linux 2.2.x kernel, you need to populate the kernel's RARP table. To do this, run the following commands:

```
/sbin/rarp -s client-hostname client-enet-addr
/usr/sbin/arp -s client-ip client-enet-addr
```

If you get

```
SIOCSRARP: Invalid argument
```

you probably need to load the RARP kernel module or else recompile the kernel to support RARP. Try `modprobe rarp` and then try the `rarp` command again.

On systems using a Linux 2.4.x kernel, there is no RARP module, and you should instead use the `rarpd` program. The procedure is similar to that used under SunOS in the following paragraph.

Under SunOS, you need to ensure that the Ethernet hardware address for the client is listed in the "ethers" database (either in the `/etc/ethers` file, or via NIS/NIS+) and in the "hosts" database. Then you need to start the RARP daemon. In SunOS 4, issue the command (as root): `/usr/etc/rarpd -a`; in SunOS 5, use `/usr/sbin/rarpd -a`.

4.4.2 Setting up BOOTP server

There are two BOOTP servers available for GNU/Linux, the CMU `bootpd` and the other is actually a DHCP server, ISC `dhcpd`, which are contained in the `bootp` and `dhcp` packages in 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        bootpd
```

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. Here is an example `/etc/bootptab`:

```
client:\
        hd=/tftpboot:\
        bf=tftpboot.img:\
        ip=192.168.1.90:\
        sm=255.255.255.0:\
        sa=192.168.1.1:\
        ha=0123456789AB:
```

You will need to change at least the “ha” option, which specifies the hardware address of the client. The “bf” option specifies the file a client should retrieve via TFTP; see ‘Move TFTP Images Into Place’ on the following page for more details.

By contrast, setting up BOOTP with ISC `dhcpd` is really easy, because it treats BOOTP clients as a moderately special case of DHCP clients. Some architectures require a complex configuration for booting clients via BOOTP. If yours is one of those, read the section ‘Setting up a DHCP server’ on the current page. Otherwise, you will probably be able to get away with simply adding the `allow bootp` directive to the configuration block for the subnet containing the client, and restart `dhcpd` with `/etc/init.d/dhcpd restart`.

4.4.3 Setting up a DHCP server

At the time of this writing, there is only one DHCP server which is free software, namely ISC `dhcpd`. In Debian GNU/Linux, this is available in the `dhcp` package. Here is a sample configuration file for it (usually `/etc/dhcpd.conf`):

```
option domain-name "example.com";
option domain-name-servers ns1.example.com;
option subnet-mask 255.255.255.0;
default-lease-time 600;
max-lease-time 7200;
server-name "servername";

subnet 192.168.1.0 netmask 255.255.255.0 {
```

```
    range 192.168.1.200 192.168.1.253;
    option routers 192.168.1.1;
  }

  host clientname {
    filename "/tftpboot/tftpboot.img";
    server-name "servername";
    next-server servername;
    hardware ethernet 01:23:45:67:89:AB;
    fixed-address 192.168.1.90;
  }
```

In this example, there is one server *"servername"* which performs all of the work of DHCP, server, TFTP server, and network gateway. You will almost certainly need to change the domain-name options, as well as the server name and client hardware address. The *"filename"* option should be the name of the file which will be retrieved via TFTP. After you have edited the `dhcpd` configuration file, restart it with `/etc/init.d/dhcpd restart`.

4.4.4 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`*.

4.4.5 Move TFTP Images Into Place

Next, place the TFTP boot image you need, as found in 'Description of Installation System Files' on page 69, in the `tftpd` boot image directory. Generally, this directory will be `/tftpboot`. 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.

SPARC architectures for instance use the subarchitecture names, such as “SUN4M” or “SUN4C”; in some cases, the architecture is left blank, so the file the client looks for is just `client-ip-in-hex`. Thus, if your system subarchitecture is a SUN4C, and its IP is 192.168.1.3, the filename would be `C0A80103.SUN4C`.

You can also force some sparc systems to look for a specific file name by adding it to the end of the OpenPROM boot command, such as `boot net my-sparc.image`. This must still reside in the directory that the TFTP server looks in.

4.4.6 TFTP Installation for Low-Memory Systems

On some systems, the standard installation RAMdisk, combined with the memory requirements of the TFTP boot image, cannot fit in memory. In this case, you can still install using TFTP, you’ll just have to go through the additional step of NFS mounting your root directory over the network as well. This type of setup is also appropriate for diskless or dataless clients.

First, follow all the steps above in ‘Preparing Files for TFTP Net Booting’ on page 27.

1. Copy the Linux kernel image on your TFTP server using the `a.out` image for the architecture you are booting.

2. Untar the root archive on your NFS server (can be the same system as your TFTP server):

   ```
   # cd /tftpboot
   # tar xvzf root.tar.gz
   ```

   Be sure to use the GNU `tar` (other tar programs, like the SunOS one, badly handle devices as plain files).

3. Export your `/tftpboot/debian-sparc-root` directory with root access to your client. E.g., add the following line to `/etc/exports` (GNU/Linux syntax, should be similar for SunOS):

   ```
   /tftpboot/debian-sparc-root client(rw,no_root_squash)
   ```

   NOTE: “client” is the host name or IP address recognized by the server for the system you are booting.

4. Create a symbolic link from your client IP address in dotted notation to `debian-sparc-root` in the `/tftpboot` directory. For example, if the client IP address is 192.168.1.3, do

   ```
   # ln -s debian-sparc-root 192.168.1.3
   ```

NOT YET WRITTEN

4.4.7 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 file system. You then need to replace the symlink to the tftpboot image by a symlink to the kernel image (for example, `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?

To boot the client machine, go to ‘Booting from TFTP’ on page 35.

4.5 Automatic Installation

For installing on multiple computers it’s possible to use the fully automatic installation called `FAI`. The Debian package `fai` has to be installed on a computer called the install server. Then all install clients boot from their network card or floppy disk and automatically install Debian on their local disks.

Chapter 5

Booting the Installation System

Subject to limitations in some cases, you may boot the installation system from a Debian GNU/Linux CD-ROM, floppy disks, a partition on a hard disk, or from another machine via a local area network.

5.1 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.

Full information on boot parameters can be found in the Linux BootPrompt HOWTO (`http://www.tldp.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:
availk/totalk 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 “ttya” or “ttyb” for SPARC, or otherwise something like “ttyS0”. Alternatively, set the *input-device* and *output-device* OpenPROM variables to “ttya”.

Again, full details on boot parameters can be found in the Linux BootPrompt HOWTO (http://www.tldp.org/HOWTO/BootPrompt-HOWTO.html), including tips for obscure hardware. Some common gotchas are included below in ‘Troubleshooting the Install Process’ on the next page.

5.1.1 `dbootstrap` Arguments

The installation system recognizes a few boot arguments which may be useful. The effects of `quiet` and `verbose` are listed in ‘Effects of Verbose and Quiet’ on page 74.

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 38), 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.

nolangchooser Some architectures use the kernel framebuffer to offer installation in a number of languages. If framebuffer causes a problem on your system you can use this option to disable the feature.

5.2 Booting from a CD-ROM

The easiest route for most people will be to use a set of Debian CDs (http://www.debian.org/CD/vendors/). If you have a CD set, and if your machine supports booting directly off the CD, great! Simply insert your CD, reboot, and proceed to the next chapter. Most OpenBoot versions support the `boot cdrom` command which is simply an alias to boot from the SCSI device on ID 6 (or the secondary master for IDE based systems). You may have to use the actual device name for older OpenBoot versions that don’t support this special command.

Note that some problems have been reported on Sun4m (e.g., Sparc 10s and Sparc 20s) systems booting from CD-ROM.

Note that certain CD drives may require special drivers, and thus be inaccessible in the early installation stages. If it turns out the standard way of booting off a CD doesn’t work for your hardware, revisit this chapter and read about alternate kernels and installation methods which may work for you.

Even if you cannot boot from CD-ROM, you can probably install the Debian system components and any packages you want from CD-ROM. Simply boot using a different media, such as floppies. When

it's time to install the operating system, base system, and any additional packages, point the installation system at the CD-ROM drive.

If you have problems booting, see 'Troubleshooting the Install Process' on this page.

5.3 Booting from Floppies

Be warned that the newer Sun4u (ultra) architecture does not support floppy booting. Furthermore, a number of Sun4c models (such as the IPX) do not support the compressed images found on the disks, so also are not supported.

Several Sparcs (e.g. Ultra 10) have an OBP bug that prevents them from booting (instead of not supporting booting at all). The appropriate OBP update can be downloaded as product ID 106121 from http://sunsolve.sun.com.

To boot from floppy on a Sparc, use

```
Stop-A -> OpenBoot: "boot floppy"
```

If you have problems booting, see 'Troubleshooting the Install Process' on the current page.

5.4 Booting from NFS

To install the system via NFS, simply select `NFS` for the location of the images and files and follow the instructions provided. You will be prompted for the `server: /directory` where the images are located.

If you have problems booting, see 'Troubleshooting the Install Process' on this page.

5.5 Booting from TFTP

Booting from the network requires that you have a network connection supported by the boot floppies, including either a static network address or a DHCP server, a RARP or a BOOTP server, and a TFTP server. The installation method to support TFTP booting is described in 'Preparing Files for TFTP Net Booting' on page 27. On machines with OpenBoot, simply enter the boot monitor on the machine you are installing to (see 'Invoking OpenBoot' on page 20), and use the command `boot net`. Some older OpenBoot revisions require using the device name, such as `boot le()`.

5.6 Troubleshooting the Install Process

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 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 Boot Configuration

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.

If you are booting with your own kernel instead of the one supplied with the installer, be sure that `CONFIG_DEVFS` is not set in your kernel. The installer is not compatible with `CONFIG_DEVFS`.

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

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 67), or investigate network installations.

If you have a large amount of memory installed in your machine, more than 512M, and the installer hangs when booting the kernel, you may need to include a boot argument to limit the amount of memory the kernel sees, such as `mem=512m`.

If you cannot boot because you get messages about a problem with "IDPROM", then it's possible that your NVRAM battery, which holds configuration information for you firmware, has run out. See the Sun NVRAM FAQ (`http://www.squirrel.com/sun-nvram-hostid.faq.html`) for more information.

If you are booting from the floppy, and you seem messages such as

```
Fatal error: Cannot read partition
Illegal or malformed device name
```

then it is possible that floppy booting is simply not supported on your machine. See 'Booting from Floppies' on the page before.

5.6.3 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 63).

5.6.4 `dbootstrap` Problem Report

If you get through the initial boot phase but cannot complete the install, `dbootstrap`'s 'Report a Problem' menu choice may be helpful. It creates `dbg_log.tgz` on a floppy, hard disk or nfs-mounted filesystem. `dbg_log.tgz` details the system's state (`/var/log/messages`, `/proc/cpuinfo` etc.). `dbg_log.tgz` may provide clues as to what went wrong and how to fix it. If you are submitting a bug report you may want to attach this file to the bug report.

5.6.5 Submitting Bug Reports

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", "woody").

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

```
architecture:  sparc
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.

5.7 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", 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 facilitated 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 compatibility 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.

To navigate within `dbootstrap`, use:

- The right arrow or Tab key to move 'forward', and left arrow or Shift-Tab to move 'backward' between buttons and selections in the current screen.
- The up and down arrow to select different items within a scrollable list, and to scroll the list itself.
- The space bar to select an item such as a checkbox.
- The *Enter* to activate choices.

5.7.1 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.

During the Base installation, package unpacking and setup messages are redirected to `tty4`. You can access this terminal by pressing *Left Alt-F4*; get back to `dbootstrap` with *Left Alt-F1*.

The unpack/setup messages generated by debootstrap are saved in `/target/tmp/debootstrap.log` when the installation is performed over a serial console.

5.8 "Release Notes"

The first screen that `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.

5.9 "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.

5.10 "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. An 'extended' keyboard is one with F1 through F10 keys along the top row.

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 51. After that, you will be prompted to mount your NFS root partition in "'Mount a Previously-Initialized Partition"' on page 46.

5.11 Last Chance!

Did we tell you to back up your disks? Here's 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.

Chapter 6

Partitioning for Debian

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.

6.1 Deciding on Debian Partitions and Sizes

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 a separate 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 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 file system 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/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?

6.2 The Directory Tree

Debian GNU/Linux adheres to the Filesystem Hierarchy Standard (http://www.pathname.com/fhs/) for directory and file naming. This standard allows users and software programs to predict the location of files and directories. The root level directory is represented simply by the slash /. At the root level, all Debian systems include these directories:

```
bin       Essential command binaries
boot      Static files of the boot loader
dev       Device files
etc       Host-specific system configuration
home      User home directories
lib       Essential shared libraries and kernel modules
mnt       Mount point for mounting a file system temporarily
proc      Virtual directory for system information
root      Home directory for the root user
sbin      Essential system binaries
tmp       Temporary files
usr       Secondary hierarchy
var       Variable data
opt       Add-on application software packages
```

The following is a list of important considerations regarding directories and partitions.

- The root partition / must always physically contain /etc, /bin, /sbin, /lib and /dev, otherwise you won't be able to boot. Typically 100 MB is needed for the root partition, but this may vary.
- /usr: all user programs (/usr/bin), libraries (/usr/lib), documentation (/usr/share/doc), etc., are in this directory. This part of the file system needs most of the space. You should provide at least 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, web sites, 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 300 - 500 megabytes of in `/var`. If hard drive 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 `/tmp`. 20-50 MB should be usually enough.

6.3 Recommended Partitioning Scheme

For new users, personal Debian boxes, home systems, and other single-user setups, a single `/` partition (plus swap) is probably the easiest, simplest way to go. It is possible to have problems with this idea, though, with larger (20GB) disks. Based on limitations in how ext2 works, avoid any single partition greater than 6GB or so.

For multi-user systems, it's best to put `/usr`, `/var`, `/tmp`, and `/home` each on their own partitions separate from the `/` partition.

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/mail` a separate partition. Often, putting `/tmp` on its own partition, for instance 20 to 50MB, 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.tldp.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. 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.

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.

For more examples, see Partitioning Strategies (`http://www.tldp.org/HOWTO/mini/Partition/partition-5.html#SUBMITTED`). For an idea of the space taken by tasks you might be interested in adding after your system installation is complete, check 'Disk Space Needed for Tasks' on page 73.

6.4 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 the drive models and/or capacities.

Sun disk partitions allow for 8 separate partitions (or slices). The third partition is usually (and is preferred to have) the "Whole Disk" partition. This partition references all of the sectors of the disk, and is used by the boot loader (either SILO, or Sun's).

6.5 Debian Partitioning Programs

Several varieties of partitioning programs have been adapted by Debian developers to work on various types of hard disks and computer architectures. Following is a list of the program(s) applicable for your architecture.

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

Be careful if you have existing FreeBSD partitions on your machine. The installation kernels include support for these partitions, but the way that `fdisk` represents them (or not) can make the device names differ. See the Linux+FreeBSD HOWTO (`http://www.tldp.org/HOWTO/mini/Linux+FreeBSD-2.html`).

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`) by pressing `Alt` and `F2` keys together, 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.

Make sure you create a "Sun disk label" on your boot disk. This is the only kind of partition scheme that the OpenBoot PROM understands, and so it's the only scheme from which you can boot. The *s* key is used in `fdisk` to create Sun disk labels.

Furthermore, on SPARC disks, make sure your first partition on your boot disk starts at cylinder 0. While this is required, it also means that the first partition will contain the partition table and the boot block, which are the first two sectors of the disk. You must *not* put swap on the first partition of the boot drive, since swap partitions do not preserve the first few sectors of the partition. You can put Ext2 or UFS partitions there; these will leave the partition table and the boot block alone.

It is also advised that the third partition should be of type "Whole disk" (type 5), and contain the entire disk (from the first cylinder to the last). This is simply a convention of Sun disk labels, and helps the `SILO` boot loader keep its bearings.

6.6 "Initialize and Activate a Swap Partition"

This will be the next step once you have created disk partitions. You have the choice of initializing and activating a new swap partition, activating a previously-initialized one, or 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.

A swap partition is strongly recommended, but you can do without one if you insist, and if your system has more than 12MB RAM. If you wish to do this, please select the "Do Without a Swap Partition" item from the menu.

6.7 “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 3.0, see the upgrade instructions (`http://www.debian.org/releases/woody/sparc/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.

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 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 `/`, the root partition.[1]

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.

6.8 “Mount a Previously-Initialized Partition”

An alternative to “‘Initialize a Linux Partition”’ on this 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,

[1] Technically, it’s being mounted at `/target`; when you reboot into the system itself, that will become `/`.

server-name-or-IP:*server-share-path*

. If you need to mount additional file systems as well, you can do that at this time.

If you have not already setup your network as described in "'Configure the Network"' on page 51, then selecting an NFS install will prompt you to do so.

6.9 Mounting Partitions Not Supported by dbootstrap

In some special situations, dbootstrap might not know how to mount your file systems (whether root or otherwise). It may be possible, if you're an experienced GNU/Linux user, to simply go to tty2 by pressing Alt and F2 keys together, and manually run the commands you need to run in order to mount the partition in question.

If you are mounting a root partition for your new system, just mount it to /target, the go back to dbootstrap and continue (perhaps running the "View the Partition Table" step to cause dbootstrap to re-compute where it is in the installation process.

For non-root partitions, you'll have to remember to manually modify your new fstab file so that when you reboot the partition will be mounted. Wait for that file (/target/etc/fstab) to be written by dbootstrap, of course, before editing it.

Chapter 7

Installing the Kernel and Base Operating System

7.1 “Install Kernel and Driver 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, and an option to install using the network. You can use any available device, you are not restricted to using the same media you used to mount (see ‘Obtaining System Installation Media’ on page 23).

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, not even prompting you for a device to install from (unless you boot with the `verbose` argument). When prompted for the CD-ROM, be sure to insert the first CD-ROM in the drive.

If you are installing from a local file system, you have a choice between two options. Select “hard disk” if the disk partition is not yet mounted; select “mounted” if it is. In both cases, the system will first look for some files in `dists/woody/main/disks-sparc/current`. If it doesn’t find those files, you will be prompted to “Select Debian Archive path” — this is the directory within the disk where you have placed the required installation files. 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/woody/main/disks-sparc /current`. You can type in the path manually, or use the <...> button to browse through the file system tree.

Continuing the discussion 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 38) 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 12). 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.

7.2 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 file system will be mounted under `/instmnt`. Select the location of the files as for “hard disk” or “mounted”.

7.3 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. . .**

7.4 NFS Root

If you are installing a diskless workstation, you should have already configured your networking as described in “‘Configure the Network”’ on the facing 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.

7.5 “Configure Device Driver Modules”

Select the “Configure Device Driver Modules” menu item to configure device drivers, that is, kernel modules.

You will first be prompted if you would like to load additional kernel modules from a vendor-supplied floppy. Most can skip this step, since it is only useful if there are some additional proprietary or non-standard modules which are needed for your hardware (for instance, for a specific SCSI controller). It will look for modules in the floppy in locations such as `/lib/modules/misc` (where *misc* can be

any standard kernel module section). Any such files will be copied to the disk you're installing to, so that they can be configured in the next step.

Next, the `modconf` program will be run, which is a simple program which displays the kernel modules sections and allows you to step through the various kernel sections, picking out what modules you would like to install.

We recommend that you *only* configure devices which are required for the installation process and not already detected by the kernel. Many people do not need to configure any kernel modules at all.

For instance, you may need to explicitly load a network interface card driver from the `net` section, a SCSI disk driver in the `scsi` section, or a driver for a proprietary CD-ROM in the `cdrom` section. The devices you configure will be loaded automatically whenever your system boots.

Some modules may require parameters. To see what parameters are relevant, you'll have to consult the documentation for that kernel driver.

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

7.6 "Configure the Network"

If the installation system does not detect that you have a network device available, you will be presented with the "Configure the Hostname" option. Even if you don't have a network, or if your network connection dynamically goes up and down (e.g., uses dialup) your machine must have a name to call itself.

If the installation system does detect a network device, you'll be presented with the "Configure the Network" step. If the system does not allow you to run this step, then that means it cannot see any network devices present. If you have a network device, that means you probably missed configuring the network device back in "'Configure Device Driver Modules"' on the preceding page. Go back to that step and look for `net` devices.

As you enter the "Configure the Network" step, if the system detects that you have more than one network device, you'll be asked to choose which device you wish to configure. You may only configure one. After installation, you may configuration additional interfaces — see the `interfaces(5)` man page.

`dbootstrap` will next ask you whether you wish to use a DHCP or BOOTP server to configure your network. If you can, you should say "Yes", since it allows you to skip all the rest of the next section. You should hopefully see the reply "The network has been successfully configured using DHCP/BOOTP.". Jump forward to "'Install the Base System"' on the following page. If configuration fails, check your wires and the log on tty3, or else move on and configure the network manually.

To manually configure the network, `dbootstrap` will ask a number of questions about your network; fill in the answers from 'Information You Will Need' on page 14. 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`. Alternatively, you can install `etherconf`, which will step you through your network setup.

7.7 "Install the Base System"

The next step is to install the base system. The base system is a minimal set of packages which provides a working, basic, self-contained system. It's under 70MB in size.

During the "Install the Base System" step, if you're not installing from a CD-ROM, you'll be offered a menu of devices from which you may install the base system. You should select the appropriate installation media. If you are installing from an official CD-ROM, you will simply be prompted to insert it.

If you are installing the base system over the network, note that some steps may take a significant amount of time, and progress may not be evident. In particular, the initial retrieve of `Packages.gz`, and the installs for base and essential packages may seem to be stalled; give them some extra time. You can use `df -h` in console 2 to assure yourself that the contents of your disk are indeed changing.

However, if the install bogs down right away retrieving a file called `Release`, you may assume that the network archive has not been found, or there is a problem with it.

If you are installing the base system from your hard disk, just point the installer to the `basedebs.tar` disk location, similar to the procedure for installing the kernel and modules.

Chapter 8

Booting Into Your New Debian System

8.1 “Make System Bootable”

The standard sparc boot loader is called “silo”. It is documented in `/usr/share/doc/silo/`. `SILO` is similar in configuration and usage to `LILO`, with a few exceptions. First of all, `SILO` allows you to boot any kernel image on your drive, even if it is not listed in `/etc/silo.conf`. This is because `SILO` can actually read Linux partitions. Also, `/etc/silo.conf` is read at boot time, so there is no need to rerun `silo` after installing a new kernel like you would with `LILO`. `SILO` can also read UFS partitions, which means it can boot SunOS/Solaris partitions as well. This is useful if you want to install GNU/Linux along side an existing SunOS/Solaris install.

If you are installing a diskless workstation, obviously, booting off the local disk isn’t a meaningful option, and this step will be skipped. You may wish to set the OpenBoot to boot from the network by default; see ‘Boot Device Selection’ on page 20.

8.2 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.

8.3 Debian Post-Boot (Base) Configuration

After booting, you will be prompted to complete the configuration of your basic system, and then to select what additional packages you wish to install. The application which guides you through this process is called `base-config`. If you wish to re-run the `base-config` at any point after installation is complete, as root run `base-config`.

8.4 Configuring your Time Zone

You will first be prompted to configure your time zone. After selecting local vs. GMT hardware clock setting, you will select a region and then a city within that region which is in the same time zone you are. When making selections in these lists, you can type a single letter to take you to the section of the list beginning with that letter.

8.5 MD5 Passwords

You will next be prompted whether to install MD5 passwords. This is an alternate method of storing passwords on your system which is more secure than the standard means (called “crypt”).

The default is “No”, but if you do not require NIS support and are very concerned about security on this machine, you may say “Yes”.

8.6 Shadow Passwords

Unless you said “Yes” to MD5 passwords, the system will ask whether you want to enable shadow passwords. This is a system in which your GNU/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.

8.7 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.

8.8 Create an Ordinary User

The system will ask you whether you wish to create an ordinary user account at this point. 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". You will also be prompted for the full name of the user, and, like before, a password.

If at any point after installation you would like to create another account, use the `adduser` command.

8.9 Setting Up PPP

You will next be asked whether you wish to install the rest of the system using PPP. If you are installing from CD-ROM and/or are connected directly to the network, you can safely say "No" and skip this section.

If you do choose to configure PPP at this point, a program named `pppconfig` will be run. This program helps you configure your PPP connection. *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 GNU/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 an editor named `nano`, which is very simple to use, but does not have a lot of features. You will probably want to install more full-featured editors and viewers later, such as `jed`, `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.

For static SLIP connections, you will need to add the `slattach` command (from the `net-tools` package) into `/etc/init.d/network`. Dynamic SLIP will require the `gnudip` package.

8.10 Configuring APT

The main means that people use to install packages on their system is via a program called `apt-get`, from the `apt` package.[1] APT must be configured, however, so that it knows where to retrieve packages from. The helper application which assists in this task is called `apt-setup`.

The next step in your configuration process is to tell APT where other Debian packages can be found. Note that you can re-run this tool at any point after installation by running `apt-setup`, or by manually editing `/etc/apt/sources.list`.

If you are booting from an official CD-ROM, then that CD-ROM should automatically be configured as an apt source without prompting. You will notice this because you will see the CD-ROM being scanned,

[1] Note that the actual program that installs packages is called `dpkg`. However, this package is more of a low-level tool. `apt-get` will invoke `dpkg` as appropriate; it is a higher-level too, however, because it knows to install other packages which are required for the package you're trying to install, as well as how to retrieve the package from your CD, the network, or wherever.

and then asked if you want to configure another CD-ROM. If you have a multiple CD-ROM set — and most people will — then you should go ahead and scan each of them one by one.

For users without an official CD-ROM, you will be offered an array of choices for how Debian packages are accessed: FTP, HTTP, CD-ROM, or a local file system. For CD-ROM users, you can get to this step by specifically asking to add another source.

You should know that it's perfectly acceptable to have a number of different APT sources, even for the same Debian archive. `apt-get` will automatically pick the package with the highest version number given all the available versions. Or, for instance, if you have both an HTTP and a CD-ROM APT source, `apt-get` should automatically use the local CD-ROM when possible, and only resort to HTTP if a newer version is available there. However, it is not a good idea to add unnecessary APT sources, since this will tend to slow down the process of checking the network archives for new versions.

8.10.1 Configuring Network Package Sources

If you plan on installing the rest of your system via the network, the most common option is to select the "http" source. The "ftp" source is also acceptable, but tends to be a little slower making connections.

Next you will be asked whether you wish to have any non-free software. That refers to commercial software or any other software whose licensing does not comply with the Debian Free Software Guidelines (`http://www.debian.org/social_contract#guidelines`). It's fine to say "Yes", but be careful when installing such software, because you will need to ensure that you are using the software in compliance with its license.

The next step during the configuration of network packages sources is to tell `apt-setup` which country you live in. This configures which of the official Debian Internet mirror network you connect to. Depending on which country you select, you will be given a list of possible machines. Its generally fine to pick the one on the top of the list, but any of them should work.

If you are installing via HTTP, you will be asked to configure your proxy server. This is sometimes required by people behind firewalls, on corporate networks, etc.

Finally, your new network package source will be tested. If all goes well, you will be prompted whether you want to do it all over again with another network source.

8.11 Package Installation: Simple or Advanced

You will next be prompted whether you wish to install packages the simple way, or the more fine-grained, advanced way. We recommend you start with the simple way, since you can always run the more advanced way at any time.

You should know that for simple installation, `base-config` is merely invoking the `tasksel` program. For advanced package installation, the `dselect` program is being run. Either of these can be run at any time after installation to install more packages. If you are looking for a specific single package, after installation is complete, simply run `apt-get install` *`package`*, where *package* is the name of the package you are looking for.

8.12 Simple Package Selection — The Task Installer

If you chose "simple" installation, you will next be thrown into the Task Installer (`tasksel`). This technique offers you a number of pre-rolled software configurations offered by Debian. You could always choose, package by package, what 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 7950 packages available in Debian!

So, you have the ability to choose *tasks* first, and then add on more individual packages later. These tasks loosely represent a number of different jobs or things you want to do with your computer, such as 'desktop environment', 'development in C', or 'file server'.

For each task, you can highlight that task and select "Task Info" to see more information on that task. This will show you an extended description and the list of packages which will be installed for that task. A table showing approximate sizes of the various tasks for planning purposes is in 'Disk Space Needed for Tasks' on page 73.

Once you've selected your tasks, select "Finish". At this point, `apt-get` will install the packages you've selected. Note, if you did not select any tasks at all, any standard, important, or required priority packages that are not yet present on your system will be installed. This functionality is the same as running tasksel -s at the command line, and currently involves a download of about 37M of archives. You will be shown the number of packages to be installed, and how many kilobytes of packages, if any, need to be downloaded.

Of the 7950 packages available in Debian, only a small minority are covered by tasks offered in the Task Installer. To see information on more packages, either use `apt-cache search` *`search-string`* for some given search string (see the `apt-cache(8)` man page), or run `dselect` as described below.

8.13 Advanced Package Selection with `dselect`

If you selected "advanced" package selection, you'll be dropped into the `dselect` program. The dselect Tutorial (`dselect-beginner.en.html`) is required reading before you run `dselect`. `dselect` allows you to select *packages* to be installed on your system. You must be the super-user (root) when you run `dselect`.

8.14 Prompts During Software Installation

Each package you selected with either `tasksel` and/or `dselect` is unpacked and then installed in turn by the `apt-get` and `dpkg` programs. If a particular program needs more information from the user, it will prompt you during this process. You might also want to keep an eye on the output during the process, to watch for any installation errors (although you will be asked to acknowledge errors which prevented a package's installation).

8.15 Log In

After you've installed packages, 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.

If you are a new user, you may want to explore the documentation which is already installed on your system as you start to use it. There are currently several documentation systems, work is proceeding on integrating the different types of documentation. Here are a few starting points.

Documentation accompanying programs you have installed is in `/usr/share/doc/`, under a subdirectory named after the program. For example, the APT User's Guide for using `apt` to install other programs on your system, is located in `/usr/share/doc/apt/guide.html/index.html`.

In addition, there are some special folders within the `/usr/share/doc/` hierarchy. Linux HOWTOs are installed in .gz format, in `/usr/share/doc/HOWTO/en-txt/` and `/usr/share/doc/HOWTO/en-txt/mini/`. The `/usr/share/doc/HTML/index.html` contains browse-able indexes of documentation installed by `dhelp`.

One easy way to view these documents is to `cd /usr/share/doc/`, and type `lynx` followed by a space and a dot (the dot stands for the current directory).

You can also type `info (command)` or `man (command)` to see documentation on most commands available at the command prompt. Typing `help` will display help on shell commands. And typing a command followed by `--help` will usually display a short summary of the command's usage. If a command's results scroll past the top of the screen, type `| more` after the command to cause the results to pause before scrolling past the top of the screen. To see a list of all commands available which begin with a certain letter, type the letter and then two tabs.

For a more complete introduction to Debian and GNU/Linux, see `/usr/share/doc/debian-guide/html/noframes/index.html`.

Chapter 9

Next Steps and Where to Go From Here

9.1 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.tldp.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/share/doc/HOWTO`. International versions of the LDP HOWTOs are also available as Debian packages.

Information specific to Debian can be found below.

9.2 Shutting Down the System

To shut down a running Linux system, you must not reboot with the reset switch on the front or back of your computer, or just turn off the computer. Linux must be shut down in a controlled manner, otherwise files may be lost and disk damage incurred. You can press the key combination Ctrl-Alt-Del . You may also log in as root and type `shutdown -h now`, `reboot`, or `halt` if either of the key combinations do not work or you prefer to type commands.

9.3 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.

9.3.1 Debian Packaging System

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`.

One of the best installation methods is apt. You can use it as a method from dselect, or you can use the command line version (info apt-get). Note apt will also let you merge main, contrib, and non-free so you can have export-restricted packages as well as standard versions.

9.3.2 Application Version Management

Alternative versions of applications are managed by update-alternatives. If you are maintaining multiple versions of your applications, read the update-alternatives man page.

9.3.3 Cron Job Management

Any jobs under the purview of the system administrator should be in `/etc`, since they are configuration files. If you have a root cron job for daily, weekly, or nightly runs, put them in `/etc/cron.{daily,weekly,month` These are invoked from `/etc/crontab`, and will run in alphabetic order, which serializes them.

On the other hand, if you have a cron job that (a) needs to run as a special user, or (b) needs to run at a special time or frequency, you can use either `/etc/crontab`, or, better yet, `/etc/cron.d/whatever`. These particular files also have an extra field that allows you to stipulate the user under which the cron job runs.

In either case, you just edit the files and cron will notice them automatically. There is no need to run a special command. For more information see cron(8), crontab(5), and `/usr/share/doc/cron/README.Debian`.

9.4 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/share/doc` as well. In particular, `/usr/share/doc/HOWTO` and `/usr/share/doc/FAQ` contain lots of interesting information. To submit bugs, look at `/usr/share/doc/debian/bug*`. To read about Debian-specific issues for particular programs, look at `/usr/share/doc/(packagename)/README.Debian`.

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://lists.debian.org/`). 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.

9.5 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

9.5.1 Kernel Image Management

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-2.2.20` (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).

This method will make a .deb of your kernel source, and, if you have non-standard modules, make a synchronized dependent .deb of those too. It's a better way to manage kernel images; `/boot` will hold the kernel, the System.map, and a log of the active config file for the build.

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-2.2.20`, yet still use the `kernel-package` compilation method. Although the 2.2.20 kernel is still used in Woody for installs, more-recent 2.4 kernels are available as kernel-images.

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

If you are compiling a kernel for UltraSPARC you will need to be sure you have installed the `egcs64` package. This is the preferred compiler for 64bit SPARC kernels. The default `gcc` will also compile 64bit kernels, but is not as stable. Plus, if you do not use `egcs64` and you encounter kernel problems, you will most likely be asked to recompile the kernel using `egcs64` in order to verify your problem still exists. After installing `egcs64` be sure to run `update-alternatives --config sparc64-linux-gcc` as root, and be sure that `egcs64` is being used for this program.

Hereafter, we'll assume your kernel source will be located in `/usr/local/src` and that your kernel version is 2.2.20. 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-2.2.20.tar.bz2`), change your directory to it (`cd kernel-source-2.2.20/`). 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" (it is not selected by default). If not included, your Debian installation will experience problems.

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-2.2.20-`*`subarchitecture`*`_custom.1.0_sparc.deb`. The *subarchitecture* 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-2.2.20` will be installed, containing your current configuration set. Your new `kernel-image-2.2.20` package is also clever enough to automatically use your platform's bootloader 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 the fine documentation in `/usr/share/doc/kernel-package`.

Chapter 10

Technical Information on the Boot Floppies

10.1 Source Code

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

10.2 Rescue Floppy

The rescue floppy has an Ext2 file system (or a FAT file system, 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.bin`. The file `root.bin` is a `gzip`-compressed disk image of a 1.4MB Minix or Ext2 file system, and will be loaded into the RAM disk and used as the root file system.

10.3 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 file systems (some architectures don’t need FAT and/or Minix file systems — see the source)

- Socket filtering for DHCP (`CONFIG_FILTER`)
- Packet socket, also for DHCP (`CONFIG_PACKET`)
- Unix domain sockets for system logging (`CONFIG_UNIX`)

Be sure that the kernel you plan to use does *NOT* have `CONFIG_DEVFS` set. `CONFIG_DEVFS` is not compatible with the installer.

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-version`*; make it from the root file system so that all leading directories are in the tar file as well.

If you want to roll your own TFTP image, you'll need some of the tools found in the `sparc-utils` package.

Chapter 11

Appendix

11.1 Further Information

11.1.1 Further Information

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

11.2 Obtaining Debian GNU/Linux

11.2.1 Official Debian GNU/Linux CD Sets

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/CD/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.

11.2.2 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).

11.2.3 Description of Installation System Files

This section contains an annotated list of files you will find in the `disks-sparc` directory. Which files you need to download will depend on the installation boot option and operating 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, which are within the www server's .../current/ (`http://http.us.debian.org/debian/dists/woody/main/disks-sparc/current/`) directory, or you can retrieve them via ftp from `ftp://ftp.debian.org/debian/dists/woody/main/disks-sparc/current/`. You can also use the corresponding directory on any of the Debian mirror sites (`http://www.debian.org/distrib/ftplist`).

Files for the Initial System Boot

Rescue floppy images:

.../current/sun4cdm/images-1.44/rescue.bin (`../sun4cdm/images-1.44/rescue.bin`)

.../current/sun4u/images-1.44/rescue.bin (`../sun4u/images-1.44/rescue.bin`) 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. The UltraSPARC platform uses the `sun4u` images; generally all other supported SPARCs will use the `sun4cdm` images.

Root image(s):

.../current/images-1.44/root.bin (`../images-1.44/root.bin`) This file contains an image of a temporary file system that gets loaded into memory when you boot from the rescue floppy. This is used for installations from CD-ROM, hard disk and floppies.

TFTP boot images

.../current/sun4cdm/tftpboot.img (`../sun4cdm/tftpboot.img`)

.../current/sun4u/tftpboot.img (`../sun4u/tftpboot.img`) Boot images used for network booting, see 'Preparing Files for TFTP Net Booting' on page 27. Generally, they contain the Linux kernel and the `root.bin` root file system.

The `tftpboot.img` contains both the sun4cdm kernel and the sun4u kernel to provide a single image for booting all the supported systems. TILO will automatically select the correct image.

Linux Kernel Files

This is the Linux kernel image to be used for hard disk installations. You don't need it if you are installing from floppies.

.../current/sun4cdm/linux-a.out (`../sun4cdm/linux-a.out`)

.../current/sun4u/linux-a.out (`../sun4u/linux-a.out`) Linux kernel files.

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.

The driver archive floppies are not used until after the hard drive has been partitioned and the kernel has been installed. If you need a particular driver for initial booting, for your subarchitecture, or to access the hard drive, choose a kernel with the necessary driver compiled in and supply the correct boot parameter arguments. Please see 'Boot Parameter Arguments' on page 33.

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

driver floppies images:

.../current/sun4cdm/images-1.44/driver-1.bin (`../sun4cdm/images-1.44/driver-1.bin`)

.../current/sun4u/images-1.44/driver-1.bin (`../sun4u/images-1.44/driver-1.bin`)

.../current/sun4u/images-1.44/driver-2.bin (`../sun4u/images-1.44/driver-2.bin`) These are the driver floppies disk images.

driver floppies archive

.../current/sun4cdm/drivers.tgz (`../sun4cdm/drivers.tgz`)

.../current/sun4u/drivers.tgz (`../sun4u/drivers.tgz`) If you are not limited to diskettes, choose one of these files.

Debian Base System Installation Files

These files are needed only for computers without a working network connection, or those with unsupported network hardware. They contain the programs needed for the most basic GNU/Linux operating system. Often the contents of these files can be obtained automatically by the installer over a working network connection.

Base System archive tarball

.../base-images-current/basedebs.tar (`http://http.us.debian.org/debian/dists/woody/main/dis`
If you are not limited to diskettes, choose this file.

11.3 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 First Floppy Drive
fd1 Second Floppy Drive

hda IDE Hard disk / CD-ROM on the first IDE port (Master)
hdb IDE Hard disk / CD-ROM on the first IDE port (Slave)
hdc IDE Hard disk / CD-ROM on the second IDE port (Master)
hdd IDE Hard disk / CD-ROM on the second IDE port (Slave)
hda1 First partition of the first IDE hard disk
hdd15 Fifteenth partition of the fourth IDE hard disk

sda SCSI Hard disk with lowest SCSI ID (e.g. 0)
sdb SCSI Hard disk with next higher SCSI ID (e.g. 1)
sdc SCSI Hard disk with next higher SCSI ID (e.g. 2)
sda1 First partition of the first SCSI hard disk
sdd10 Tenth partition of the fourth SCSI hard disk

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 MS-DOS
ttyS1 Serial port 1, COM2 under MS-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
```

11.3.1 Setting Up Your Mouse

The mouse can be used in both the Linux console (with gpm) and the X window environment. The two uses can be made compatible if the gpm repeater is used to allow the signal to flow to the X server as shown:

```
mouse => /dev/psaux  => gpm => /dev/gpmdata -> /dev/mouse => X
         /dev/ttyS0             (repeater)        (symlink)
         /dev/ttyS1
```

Set the repeater protocol to be raw (in `/etc/gpm.conf`) while setting X to the original mouse protocol in `/etc/X11/XF86Config` or `/etc/X11/XF86Config-4`.

This approach to use gpm even in X has advantages when the mouse is unplugged inadvertently. Simply restarting gpm with

```
user@debian:# /etc/init.d/gpm restart
```

will re-connect the mouse in software without restarting X.

If gpm is disabled or not installed with some reason, make sure to set X to read directly from the mouse device such as /dev/psaux. For details, refer to the 3-Button Mouse mini-Howto at `/usr/share/doc/HOWTO/en-txt/mini/3-Button-Mouse.gz`, `man gpm`, `/usr/share/doc/gpm/FAQ.gz`, and README.mouse (http://www.xfree86.org/current/mouse.html).

11.4 Disk Space Needed for Tasks

The base woody installation on the author's computer required 117MB. The installed size for all standard packages was 123MB, with a download size of 38MB; so 278MB of space was needed to install the base and all standard packages.

The following table lists sizes reported by aptitude (a very nice program, by the way) for the tasks listed in tasksel. The system for which the figures were reported already had all standard packages installed. Note that some tasks have overlapping constituents, so the total installed size for two tasks together may be less than the total obtained by adding the numbers up.

Task	Installed Size (MB)	Download Size (MB)	Space Needed To Install (MB)
desktop environment	345	118	463
X window system	78	36	114
games	49	14	63
Debian Jr.	340	124	464
dialup system	28	8	36
laptop system	3	1	4
scientific applications	110	30	140
C and C++	32	15	47
Python	103	30	133
Tcl/Tk	37	11	48
fortran	10	4	14

file server	1	-	1
mail server	4	3	7
usenet news server	6	2	8
print server	48	18	66
conventional unix server	55	19	74
web server	4	1	5
TeX/LaTeX environment	171	64	235
simplified Chinese environment	80	29	109
traditional Chinese environment	166	68	234
Cyrillic environment	29	13	42
French environment	60	18	78
German environment	31	9	40
Japanese environment	110	53	163
Korean environment	178	72	250
Polish environment	58	27	85
Russian environment	12	6	18
Spanish environment	15	4	19

11.5 Effects of Verbose and Quiet

These are the effects of the `verbose` boot argument for woody:

- For LiveCD, allow choice of alternate install media
- When mounting volumes, always ask which mount point
- Warn that earlier kernels do not support newer file systems
- Warn that pre-2.4.1 kernels do not support ReiserFS 3.6
- Confirm install files path even if only one path found

These are the effects of the `quiet` boot argument for woody:

- Suppress confirm before writing the aboot boot loader
- Suppress confirm before overwriting master boot record
- Suppress 'Important Information about installed MBR'
- No invitation to install additional modules from floppy
- Don't mention that s390 doesn't support reboot

- Suppress confirmation that detected interface is PCMCIA
- Suppress message about successful DHCP configuration
- Suppress long message about Lilo and large disk support
- Suppress long message about PALO and large disk support
- Suppress SGI disk label note from Dvhtool
- Don’t chatter about how much disk space ReiserFS uses
- Don’t explain what Apple_Bootstrap is
- Mount the first initialized partition on / without asking
- Don’t offer to scan for bad blocks
- Don’t ask before initializing as XFS, ext2/3, ReiserFS, swap
- Avoid trying to persuade that a swap partition is good
- Don’t lecture before rebooting the system

Chapter 12

Administrivia

12.1 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.

12.2 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://bugs.debian.org/`). It would be nice if you could check the open bugs against boot-floppies (`http://bugs.debian.org/boot-floppies`) 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/cgi-bin/viewcvs.cgi/~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 browse-able copy can be found at the Debian Mailing List Archives (http://lists.debian.org/).

12.3 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://linux.iol.unh.edu/linux/alpha/faq/), amongst others. The maintainers of these freely available and rich sources of information must be recognized.

12.4 Trademark Acknowledgement

All trademarks are property of their respective trademark owners.