Installing Debian GNU/Linux 2.2 For Intel x86

Bruce Perens
Sven Rudolph
Igor Grobman
James Treacy
Adam Di Carlo

version 2.2.27, 14 Listopad, 2001

Abstract

This document contains installation instructions for the Debian GNU/Linux 2.2 system, for the Intel x86 (“i386”) architecture. It also contains pointers to more information and information on how to make the most of your new Debian system. The procedures in this document are *not* to be used for users upgrading existing systems; if you are upgrading, see the Release Notes for Debian 2.2 (`http://www.debian.org/releases/2.2/i386/release-notes/`).

Copyright Notice

Contents

Chapter 1

Welcome to Debian

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

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

1.1 What is Debian?

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

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

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

- The Debian Social Contract (`http://www.debian.org/social_contract`) is a statement of Debian's commitments to the Free Software Community. Anyone who agrees to

abide to the Social Contract may become a maintainer (http://www.debian.org/doc/maint-guide/). Any maintainer can introduce new software into Debian — provided that the software meets our criteria for being free, and the package follows our quality standards.
- The Debian Free Software Guidelines (http://www.debian.org/social_contract#guidelines) are a clear and concise statement of Debian's criteria for free software. The DFSG is a very influential document in the Free Software Movement, and was the foundation of the Open Source Free Software Guidelines (http://opensource.org/osd.html).
- The Debian Policy Manual (http://www.debian.org/doc/debian-policy/) is an extensive specification of the Debian Project's standards of quality.

Debian developers are also involved in a number of other projects; some specific to Debian, others involving some or all of the Linux community. Some examples include:
- The Linux Standard Base (http://www.linuxbase.org/) (LSB) is a project aimed at standardizing the basic GNU/Linux system, which will enable third-party software and hardware developers to easily design programs and device drivers for Linux-in-general, rather than for a specific GNU/Linux distribution.
- The Filesystem Hierarchy Standard (http://www.pathname.com/fhs/) (FHS) is an effort to standardize the layout of the Linux filesystem. The FHS will allow software developers to concentrate their efforts on designing programs, without having to worry about how the package will be installed in different GNU/Linux distributions.
- Debian Jr. (http://www.debian.org/devel/debian-jr/) is an internal project, aimed at making sure Debian has something to offer to our youngest users.

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

1.2 What is GNU/Linux?

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

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

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

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

1.3 What is Debian GNU/Linux?

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

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

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

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

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

1.4 What is Debian GNU/Hurd?

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

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

1.5 Getting Debian

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

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

1.6 Getting the Newest Version of This Document

This document is constantly being revised. Be sure to check the Debian 2.2 pages (`http://www.debian.org/releases/2.2/`) for any last-minute information about the 2.2 release of the Debian GNU/Linux system. Updated versions of this installation manual are also available from the official Install Manual pages (`http://www.debian.org/releases/2.2/i386/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 9.

2. Backup your system, and perform any necessary planning and hardware configuration prior to installing Debian, in 'Before You Start' on page 17.

3. Getting the partitions on your system set up correctly is very important, because once you've done the install, you may have to live with your choices for a long time.

4. In 'Methods for Installing Debian' on page 31, several different ways to install Debian are presented and discussed. Select your favorite method and prepare your installation media as described.

5. 'Booting the Installation System' on page 47, describes booting into the installation system. This chapter also discusses troubleshooting procedures in case you have problems with this step.

6. Perform the initial system configuration, which is discussed in 'Using `dbootstrap` for Initial System Configuration' on page 55 (Sections 'Introduction to `dbootstrap`' on page 55 through "'Configure the Network"' on page 63).

7. "'Install the Base System"' on page 64.

8. Boot into your newly installed base system and run through some additional configuration tasks, from 'The Moment of Truth' on page 66.

9. Install the rest of the system, using `dselect` or `apt-get`, in 'Package Installation: Simple or Advanced' on page 71.

Once you've got your system installed, you can read 'Next Steps and Where to Go From Here' on page 73. 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 79.

Finally, information about this document and how to contribute to it may be found in 'Administrivia' on page 85.

1.8 WARNING: This Document Has Known Problems

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

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

Working versions of this document can be found at `http://www.debian.org/releases/2.2/i386/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 85. 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

[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.

ensures that any user will be able to modify the software. Because of this provision, the source code for all such programs is available in the Debian system.[2]

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

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

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

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

Chapter 2

System Requirements

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

2.1 Supported Hardware

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

There are, however, some limitations in our boot floppy set with respect to supported hardware. Some Linux-supported platforms might not be directly supported by our boot floppies. If this is the case, you may have to create a custom rescue disk (see 'Replacing the Rescue Floppy Kernel' on page 79), or investigate network installations.

Rather than attempting to describe all the different hardware configurations which are supported for Intel x86, this section contains general information and pointers to where additional information can be found.

2.1.1 Supported Architectures

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

This document covers installation for the *i386* architecture. If you look for information on other architectures take a look at the Debian-Ports (`http://www.debian.org/ports/`) pages.

2.1.2 CPU, Mainboards, and Video Support

Complete information concerning supported peripherals can be found at Linux Hardware Compatibility HOWTO (http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html). This section merely outlines the basics.

CPU

Nearly all x86-based processors are supported; this includes AMD and Cyrix processors as well. Also the new processors like Athlon and the K6-2 or K6-3, respectively, are supported. However, Linux will *not* run on 286 or earlier processors.

I/O Bus

The system bus is the part of the motherboard which allows the CPU to communicate with peripherals such as storage devices. Your computer must use the ISA, EISA, PCI, the Microchannel Architecture (MCA, used in IBM's PS/2 line), or VESA Local Bus (VLB, sometimes called the VL bus).

Graphics Card

You should be using a VGA-compatible display interface for the console terminal. Nearly every modern display card is compatible with VGA. Ancient standards such CGA, MDA, or HGA should also work, assuming you do not require X11 support. Note that X11 is not used during the installation process described in this document.

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 2.2 ships with X11 revision 3.3.6.

Laptops

Laptops are also supported. Laptops are often specialized or contain proprietary hardware. To see if your particular laptop works well with GNU/Linux, see the Linux Laptop pages (http://www.cs.utexas.edu/users/kharker/linux-laptop/).

2.1.3 Multiple Processors

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

In order to take advantage of multiple processors, you'll have to replace the standard Debian kernel. You can find a discussion of how to do this in 'Compiling a New Kernel' on page 75. At this time (kernel version 2.2.19) the way you enable SMP is to select "symmetric multi-processing" in the "General" section of the kernel config. If you compile software on a multiprocessor system, look for the `-j` flag in the documentation on `make(1)`.

2.2 Installation Media

There are four different media which can be used to install Debian: floppies, CD-ROMs, local disk partitions, or the network. Different parts of the same Debian installation can mix and match these options; we'll go into that in 'Methods for Installing Debian' on page 31.

Floppy disk installation is a common option, although generally, the least desirable. In many cases, you'll have to do your first boot from floppies, using the Rescue Floppy. Generally, all you will need is a high-density (1440 kilobytes) 3.5 inch floppy drive. Double-density, 5.25 inch installation floppies (1200 k) are also provided.

CD-ROM based installation is also supported for some architectures. On machines which support bootable CD-ROMs, you should be able to do a completely floppy-less installation. Even if your system doesn't support booting from a CD-ROM, you can use the CD-ROM in conjunction with the other techniques to install your system, once you've booted up by other means; see 'Booting and/or Installing from a CD-ROM' on page 50.

Both SCSI and IDE/ATAPI CD-ROMs are supported. In addition, all non-standard CD interfaces supported by Linux are supported by the boot disks (such as Mitsumi and Matsushita drives). However, these models might require special boot parameters or other massaging to get them to work, and booting off these non-standard interfaces is unlikely. The Linux CD-ROM HOWTO (`http://www.linuxdoc.org/HOWTO/CDROM-HOWTO.html`) contains in-depth information on using CD-ROMs with Linux.

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

The last option is network installation. You can install your base system via HTTP or NFS. Diskless installation, using network booting and NFS-mounting of all local filesystems, is another option – you'll probably need at least 16MB of RAM for this option. After your base system is installed,

you can install the rest of your system via any sort of network connection (including PPP), via FTP, HTTP, or NFS.

More complete descriptions of these methods, and helpful hints for picking which method is best for you, can be found in 'Methods for Installing Debian' on page 31. Please be sure to continue reading to make sure the device you intend to boot and install from is supported by the Debian installation system.

2.2.1 Supported Storage Systems

The Debian boot disks contain a kernel which is built to maximize the number of systems it runs on. Unfortunately, this makes for a larger kernel, with a lot of drivers which will never be used (see 'Compiling a New Kernel' on page 75 to learn how to build your own). However, support for the widest possible range of devices is desirable in order to ensure that Debian can be installed on the widest array of hardware.

Generally, the Debian installation system includes support for floppies, IDE drives, IDE floppies, parallel port IDE devices, SCSI controllers and drives. The file systems supported include MINIX, FAT, Win-32 FAT extensions (VFAT), among others (note that NTFS is not supported by the installation system; you can add it later, as described in 'Compiling a New Kernel' on page 75).

Rather than attempting to describe the supported hardware, it is much easier to describe the Linux supported hardware which is *not* supported by the Debian boot system.

The disk interfaces that emulate the "AT" hard disk interface which are often called MFM, RLL, IDE, or ATA are supported. Very old 8 bit hard disk controllers used in the IBM XT computer are supported only as a module. SCSI disk controllers from many different manufacturers are supported. See the Linux Hardware Compatibility HOWTO (`http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html`) for more details.

Not supported are IDE SCSI drives and some SCSI controllers, including

- EATA-DMA protocol compliant SCSI Host Adapters like the SmartCache III/IV, SmartRAID controller families and the DPT PM2011B and PM2012B controllers.
- The 53c7 NCR family of SCSI controllers (but 53c8 and 5380 controllers are supported)

2.3 Memory and Disk Space Requirements

You must have at least 12MB of memory and 64MB of hard disk. If you want to install a reasonable amount of software, including the X Window System, and some development programs and libraries, you'll need at least 300MB. For a more or less complete installation, you'll need around 800MB. To install *everything* available in Debian, you'll probably need around 2 GB. Actually, installing everything doesn't even make sense, since some packages conflict with others.

2.4 Peripherals and Other Hardware

Linux supports a large variety of hardware devices such as mice, printers, scanners, modems, network cards, PCMCIA devices, etc. However, none of these devices are required while installing the system. This section contains information about peripherals specifically *not* supported by the installation system, even though they may be supported by Linux. Again, see the Linux Hardware Compatibility HOWTO (http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html) to determine whether your specific hardware is supported by Linux.

Some network interface cards (NICs) are not supported by the Debian installation disks (although a custom Linux kernel can use them), such as AX.25 cards and protocols; 3Com EtherLink Plus (3c505) and EtherLink16 (3c507); NI5210 cards; generic NE2100 cards; NI6510 and NI16510 EtherBlaster cards; SEEQ 8005 cards; Schneider & Koch G16 cards; Ansel Communications EISA 3200; and the Zenith Z-Note built-in network card. Microchannel (MCA) network cards are not supported by the standard installation system, but see Linux on MCA disk images (ftp://ns.gold-link.com/pub/LinuxMCA/) for unofficial images, and the Linux MCA discussion archives (http://www.dgmicro.com/linux_frm.htm). FDDI networks are also not supported by the installation disks, both cards and protocols.

As for ISDN, the D-channel protocol for the (old) German 1TR6 is not supported; Spellcaster BRI ISDN boards are also not supported by the boot-floppies.

Sound devices are not supported by default. But as already mentioned above: if you want to use an own kernel please go to 'Compiling a New Kernel' on page 75 for further information.

2.5 Purchasing Hardware Specifically for GNU/Linux

There are several vendors, now, who ship systems with Debian or other distributions of GNU/Linux pre-installed. You might pay more for the privilege, but it does buy a level of peace of mind, since you can be sure that the hardware is well-supported by GNU/Linux. If you do have to buy a machine with Windows bundled, carefully read the software license that comes with Windows; you may be able to reject the license and obtain a rebate from your vendor. See http://www.linuxmall.com/refund/ for complete details.

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

2.5.1 Avoid Proprietary or Closed Hardware

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

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

2.5.2 Windows-specific Hardware

A disturbing trend is the proliferation of Windows-specific modems and printers. In some cases these are specially designed to be operated by the Microsoft Windows operating system and bear the legend "WinModem" or "Made especially for Windows-based computers". This is generally done by removing the embedded processors of the hardware and shifting the work they do over to a Windows driver that is run by your computer's main CPU. This strategy makes the hardware less expensive, but the savings are often *not* passed on to the user and this hardware may even be more expensive than equivalent devices that retain their embedded intelligence.

You should avoid Windows-specific hardware for two reasons. The first is that the manufacturers do not generally make the resources available to write a Linux driver. Generally, the hardware and software interface to the device is proprietary, and documentation is not available without a non-disclosure agreement, if it is available at all. This precludes its being used for free software, since free software writers disclose the source code of their programs. The second reason is that when devices like these have had their embedded processors removed, the operating system must perform the work of the embedded processors, often at *real-time* priority, and thus the CPU is not available to run your programs while it is driving these devices. Since the typical Windows user does not multi-process as intensively as a Linux user, the manufacturers hope that the Windows user simply won't notice the burden this hardware places on their CPU. However, any multi-processing operating system, even Windows 95 or NT, suffers from degraded performance when peripheral manufacturers skimp on the embedded processing power of their hardware.

You can help this situation by encouraging these manufacturers to release the documentation and other resources necessary for us to program their hardware, but the best strategy is simply to avoid this sort of hardware until it is listed as working in the Linux Hardware Compatibility HOWTO (`http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html`).

2.5.3 Fake or "Virtual" Parity RAM

If you ask for Parity RAM in a computer store, you'll probably get *virtual parity* memory modules instead of *true parity* ones. Virtual parity SIMMs can often (but not always) be distinguished because they only have one more chip than an equivalent non-parity SIMM, and that one extra chip is smaller than all the others. Virtual-parity SIMMs work exactly like non-parity memory. They can't tell you when you have a single-bit RAM error the way true-parity SIMMs do in a motherboard that implements parity. Don't ever pay more for a virtual-parity SIMM than a non-parity one. Do expect to pay a little more for true-parity SIMMs, because you are actually buying one extra bit of memory for every 8 bits.

If you want complete information on Intel x86 RAM issues, and what is the best RAM to buy, see the PC Hardware FAQ (`ftp://rtfm.mit.edu/pub/usenet-by-hierarchy/comp/sys/ibm/pc/hardware/systems/`).

Chapter 3

Before You Start

3.1 Backups

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

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

3.2 Information You Will Need

Besides this document, you'll need the cfdisk (`cfdisk.txt`) manual page, the fdisk (`fdisk.txt`) manual page, the dselect Tutorial (`dselect-beginner`), and the Linux Hardware Compatibility HOWTO (`http://www.linuxdoc.org/HOWTO/Hardware-HOWTO.html`).

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

- Your host name (you may be able to decide this on your own).
- Your domain name.

- Your computer's IP address.
- The IP address of your network.
- The netmask to use with your network.
- The broadcast address to use on your network.
- The IP address of the default gateway system you should route to, if your network *has* a gateway.
- The system on your network that you should use as a DNS (Domain Name Service) server.
- Whether you connect to the network using Ethernet.
- Whether your Ethernet interface is a PCMCIA card; if so, the type of PCMCIA controller you have.

If your computer's only network connection is via a serial line, using PPP or an equivalent dialup connection, you are probably not installing the base system over a network. You don't need to worry about getting your network setup until your system is already installed. See 'Setting Up PPP' on page 68 below for information on setting up PPP under Debian.

3.3 Pre-installation Hardware and Operating System Setup

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

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

3.3.1 Invoking the BIOS Set-Up Menu

BIOS provides the basic functions needed to boot your machine to allow your operating system to access your hardware. Your system probably provides a BIOS set-up menu, which is used to configure the BIOS. Before installing, you *must* ensure that your BIOS is setup correctly; not doing so can lead to intermittent crashes or an inability to install Debian.

The rest of this section is lifted from the PC Hardware FAQ (`ftp://rtfm.mit.edu/pub/usenet-by-hierar`
`comp/sys/ibm/pc/hardware/systems/`), answering the question, "How do I enter the CMOS

configuration menu?". How you access the BIOS (or "CMOS") configuration menu depends on who wrote your BIOS software:

[From: burnesa@cat.com (Shaun Burnet)]

AMI BIOS Del key during the POST (power on self test)

Award BIOS Ctrl-Alt-Esc, or Del key during the POST

DTK BIOS Esc key during the POST

IBM PS/2 BIOS Ctrl-Alt-Ins after Ctrl-Alt-Del

Phoenix BIOS Ctrl-Alt-Esc or Ctrl-Alt-S

[From: mike@pencom.com (Mike Heath)] Some 386 machines don't have a CMOS configuration menu in the BIOS. They require a software CMOS setup program. If you don't have the Installation and/or Diagnostics diskette for your machine, you can try using a shareware/freeware program. Try looking in `ftp://ftp.simtelnet.net/pub/simtelnet/msdos/`.

3.3.2 Boot Device Selection

Many BIOS set-up menus allow you to select the devices that will be used to bootstrap the system. Set this to look for a bootable operating system on `A:` (the first floppy disk), then optionally the first CD-ROM device (possibly appearing as `D:` or `E:`), and then from `C:` (the first hard disk). This setting enables you to boot from either a floppy disk or a CD-ROM, which are the two most common boot devices used to install Debian.

If you have a newer SCSI controller and you have a CD-ROM device attached to it, you are usually able to boot from the CD-ROM. All you have to do is enabling booting from a CD-ROM in the SCSI-BIOS of your controller. Additionally you have to be able to boot from a floppy disk. This is set up in the PC-BIOS.

If your system can't boot directly from CD-ROM, or you simply can't seem to get it to work, don't despair; you can simply run `E:\install\boot.bat` under DOS (replace `E:` with whatever drive letter DOS assigns to your CD-ROM drive) to start the installation process. See 'Booting and/or Installing from a CD-ROM' on page 50 below for details.

Also, if you're going to be installing from a FAT (DOS) partition, you won't need any floppies at all. See 'Booting from a DOS partition' on page 49 below for more information on installing via this method.

3.3.3 CD-ROM Settings

Some BIOSes (such as Award BIOS) allows you to automatically set the CD speed. You should avoid that, and instead set it to, say, the lowest speed. If you get `seek failed` error messages, this may be your problem.

3.3.4 Extended vs. Expanded Memory

If your system provides both ex*ten*ded and ex*pan*ded memory, set it so that there is as much extended and as little expanded memory as possible. Linux requires extended memory and cannot use expanded memory.

3.3.5 Virus Protection

Disable any virus-warning features your BIOS may provide. If you have a virus-protection board or other special hardware, make sure it is disabled or physically removed while running GNU/Linux. These aren't compatible with GNU/Linux; moreover, due to the file system permissions and protected memory of the Linux kernel, viruses are almost unheard of.[1]

3.3.6 Shadow RAM

Your motherboard may provide *shadow RAM* or BIOS caching. You may see settings for "Video BIOS Shadow", "C800-CBFF Shadow", etc. *Disable* all shadow RAM. Shadow RAM is used to accelerate access to the ROMs on your motherboard and on some of the controller cards. Linux does not use these ROMs once it has booted because it provides its own faster 32-bit software in place of the 16-bit programs in the ROMs. Disabling the shadow RAM may make some of it available for programs to use as normal memory. Leaving the shadow RAM enabled may interfere with Linux access to hardware devices.

3.3.7 Advanced Power Management

If your motherboard provides Advanced Power Management (APM), configure it so that power management is controlled by APM. Disable the doze, standby, suspend, nap, and sleep modes, and disable the hard disk's power-down timer. Linux can take over control of these modes, and can do a better job of power-management than the BIOS. The version of the operating system kernel on the installation floppies does not, however, use APM, because we've had reports of one

[1] After installation you can enable Boot Sector protection if you want. There is no need to tamper with Master Boot Record (MBR) after the boot manager has been set up. This offers no additional security in Linux but if you also do Windows it may prevent a catastrophe.

laptop system crashing when the Linux APM driver is configured. Once you've installed Linux, you can build a custom-configured version of the Linux kernel; see 'Compiling a New Kernel' on page 75 for instructions how.

3.3.8 The Turbo Switch

Many systems have a *turbo* switch that controls the speed of the CPU. Select the high-speed setting. If your BIOS allows you to disable software control of the turbo switch (or software control of CPU speed), do so and lock the system in high-speed mode. We have one report that on a particular system, while Linux is auto-probing (looking for hardware devices) it can accidentally touch the software control for the turbo switch.

3.3.9 Over-Clocking your CPU

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

3.3.10 Bad Memory Modules

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

The very best motherboards support parity RAM and will actually tell you if your system has a single-bit error in RAM. Unfortunately, they don't have a way to fix the error, thus they generally crash immediately after they tell you about the bad RAM. Still, it's better to be told you have bad memory than to have it silently insert errors in your data. Thus, the best systems have motherboards that support parity and true-parity memory modules; see 'Fake or "Virtual" Parity RAM' on page 15.

If you do have true-parity RAM and your motherboard can handle it, be sure to enable any BIOS settings that cause the motherboard to interrupt on memory parity errors.

3.3.11 Cyrix CPUs and Floppy Disk Errors

Many users of Cyrix CPUs have had to disable the cache in their systems during installation, because the floppy disk has errors if they do not. If you have to do this, be sure to re-enable your

cache when you are finished with installation, as the system runs *much* slower with the cache disabled.

We don't think this is necessarily the fault of the Cyrix CPU. It may be something that Linux can work around. We'll continue to look into the problem. For the technically curious, we suspect a problem with the cache being invalid after a switch from 16-bit to 32-bit code.

3.3.12 Miscellaneous BIOS Settings to Watch Out For

If your BIOS offers something like "15-16 MB Memory Hole", please disable that. Linux expects to find memory there if you have that much RAM.

We have a report of an Intel Endeavor motherboard on which there is an option called "LFB" or "Linear Frame Buffer". This had two settings: "Disabled" and "1 Megabyte". Set it to "1 Megabyte". When disabled, the installation floppy was not read correctly, and the system eventually crashed. At this writing we don't understand what's going on with this particular device – it just worked with that setting and not without it.

3.3.13 Peripheral Hardware Settings to Watch Out For

In addition to your BIOS settings, you may have to change some settings on the actual cards. Some cards have setup menus, while others rely on jumpers. This document cannot hope to provide complete information on every hardware device; what it hopes to provide is useful tips.

If any cards provide "mapped memory", the memory should be mapped somewhere between 0xA0000 and 0xFFFFF (from 640K to just below 1 megabyte) or at an address at least 1 megabyte greater than the total amount of RAM in your system.

3.3.14 More than 64 MB RAM

The Linux Kernel can not always detect what amount of RAM you have. If this is the case please look at 'Boot Parameter Arguments' on page 47.

Chapter 4

Partitioning Your Hard Drive

4.1 Background

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

If you already have an operating system on your system (Windows95, Windows NT, OS/2, MacOS, Solaris, FreeBSD, ...) and want to stick Linux on the same disk, you will probably need to repartition the disk. In general, changing a partition with a filesystem already on it will destroy any information there. Thus you should always make backups before doing any repartitioning. Using the analogy of the house, you would probably want to move all the furniture out of the way before moving a wall or you risk destroying it. Luckily, there is an alternative for some users; see 'Lossless Repartitioning When Starting From DOS, Win-32 or OS/2' on page 29.

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 cheap disk storage as "virtual memory". By putting swap on a separate partition, Linux can make much more efficient use of it. It is possible to force Linux to use a regular file as swap, but it is not recommended.

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

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

Another reason applies to you only if you have a large IDE disk, and are using neither LBA addressing, overlay drivers (sometimes provided by hard disk manufacturers), nor a new (post 1998) BIOS that supports large disk access extensions. In this case, you will have to put the boot partition into the first 1024 cylinders of your hard drive (usually around 524 megabytes, without BIOS translation).

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

4.1.1 The Directory Tree

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

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

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

- `/dev`: this directory contains the various device files which are interfaces to the various hardware components. For more information see 'Device Names in Linux' on page 27.

- `/usr`: all user programs (`/usr/bin`), libraries (`/usr/lib`), documentation (`/usr/share/doc`), etc., are in this directory. This part of the filesystem needs most of the space. You should 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, websites, APT's cache, etc. will be placed under this directory. The size of this directory depends greatly on the usage of your computer, but for most people will be dictated by the package management tool's overhead. If you are going to do a full installation of just about everything Debian has to offer, all in one session, setting aside 2 or 3 gigabytes of space for `/var` should be sufficient. If you are going to install in pieces (that is to say, install services and utilities, followed by text stuff, then X, ...), you can get away with 300 - 500 megabytes of in `/var`. If harddrive space is at a premium and you don't plan on using APT, at least not for major updates, you can get by with as little as 30 or 40 megabytes in `/var`.
- `/tmp`: if a program creates temporary data it will most likely go in `/tmp`. 20-50 MB should be usually enough.

4.2 Planning Use of the System

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

Debian offers the `tasksel` tool to assist the user during installation. (see 'Simple Package Selection – The Task Installer' on page 71). Tasks are collections of packages which are automatically marked for installation as a group, to implement a given type of Linux installation. Checking the sizes of various tasks will give you a sense of how large your partition or partitions need to be for your intended usage.

Link to a page dynamically generated using current tasksel to list tasks with associated sizes. Delete following outdated list.

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

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

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

Devel_comp A desktop setup with all the development packages, such as Perl, C, C++, etc. Size is around 475 MB. Assuming you are adding X11 and some additional packages for other uses, you should plan around 800 MB 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 20 MB; with logs and the rest, you should usually allocate at least 50 MB for `/var`.

4.2.1 PC Disk Limitations

The PC BIOS generally adds additional constraints for disk partitioning. There is a limit to how many "primary" and "logical" partitions a drive can contain. Additionally, with pre 1994-98 BIOS, there are limits to where on the drive the BIOS can boot from. More information can be found in the Linux Partition HOWTO (`http://www.linuxdoc.org/HOWTO/mini/Partition/`) and the Phoenix BIOS FAQ (`http://www.phoenix.com/pcuser/BIOS/biosfaq2.htm`), but this section will include a brief overview to help you plan most situations.

"Primary" partitions are the original partitioning scheme for PC disks. However, there can only be four of them. To get past this limitation, "extended" and "logical" partitions were invented. By setting one of your primary partitions as an extended partition, you can subdivide all the space allocated to that partition into logical partitions. You can create up to 60 logical partitions per extended partition; however, you can only have one extended partition per drive.

Linux limits the partitions per drive to 15 partitions for SCSI disks (3 usable primary partitions, 12 logical partitions), and 63 partitions on an IDE drive (3 usable primary partitions, 60 logical partitions).

The last issue about the PC BIOS which you need to know is that your boot partition, that is, the partition containing your kernel image, needs to be contained within the first 1024 cylinders of the drive, *unless* you have a BIOS newer than around 1995-98 (depending on the manufacturer) that supports the "Enhanced Disk Drive Support Specification". Both Lilo, the Linux loader, and Debian's alternative `mbr` must use the BIOS to read the kernel from the disk into RAM. If the BIOS int 0x13 large disk access extensions are found to be present, they will be utilized. Otherwise, the legacy disk access interface is used as a fallback, and it cannot be used to address any location on the disk higher than the 1023rd cylinder. Once Linux is booted, no matter what BIOS your computer has, these restrictions no longer apply, since Linux does not use the BIOS for disk access.

If you have a large disk, you might have to use cylinder translation techniques, which you can set from your BIOS setup program, such as LBA (Logical Block Addressing) or CHS translation mode ("Large"). More information about issues with large disks can be found in the Large Disk HOWTO (`http://www.linuxdoc.org/HOWTO/Large-Disk-HOWTO.html`). If you are using

a cylinder translation scheme, and the BIOS does not support the large disk access extensions, then your boot partition has to fit within the *translated* representation of the 1024th cylinder.

The recommended way of accomplishing this is to create a small (5-10 MB should suffice) partition at the beginning of the disk to be used as the boot partition, and then create whatever other partitions you wish to have, in the remaining area. This boot partition *must* be mounted on `/boot`, since that is the directory where the Linux kernel(s) will be stored. This configuration will work on any system, regardless of whether LBA or large disk CHS translation is used, and regardless of whether your BIOS supports the large disk access extensions.

4.3 Device Names in Linux

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

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

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

Here is a real-life example. Let's assume you have a system with 2 SCSI disks, one at SCSI address 2 and the other at SCSI address 4. The first disk (at address 2) is then named "sda", and the second

"sdb". If the "sda" drive has 3 partitions on it, these will be named "sda1", "sda2", and "sda3". The same applies to the "sdb" disk and its partitions.

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

Linux represents the primary partitions as the drive name, plus the numbers 1 through 4. For example, the first primary partition on the first IDE drive is `/dev/hda1`. The logical partitions are numbered starting at 5, so the first logical partition on that same drive is `/dev/hda5`. Remember that the extended partition, that is, the primary partition holding the logical partitions, is not usable by itself. This applies to SCSI disks as well as IDE disks.

4.4 Recommended Partitioning Scheme

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

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

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

With respect to the issue of swap partition size, there are many views. One rule of thumb which works well is to use as much swap as you have system memory, although there probably isn't much point in going over 64 MB of swap for most users. It also shouldn't be smaller than 16 MB, in most cases. Of course, there are exceptions to these rules. If you are trying to solve 10000 simultaneous equations on a machine with 256 MB 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 2 GB (on Alpha and SPARC64, it's so large as to be virtually unlimited). This should be enough for nearly any installation. However, if your swap requirements are this high, you should probably try to spread the swap across different disks (also called "spindles") and, if possible, different SCSI or IDE channels. The kernel will balance swap usage between multiple swap partitions, giving better performance.

4.5 Example Partitioning

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

4.6 Partitioning Prior to Installation

There are two different times that you can partition: prior to the installation of Debian, or during installation of Debian. If your computer will be solely dedicated to Debian, you should partition as part of the installation process ("'Partition a Hard Disk"' on page 57). If you have a machine with more than one operating system on it, you generally should let the native operating system create its own partitions.

The following sections contain information regarding partitioning in your native operating system prior to installation. Note that you'll have to map between how the other operating system names partitions, and how Linux names partitions; see 'Device Names in Linux' on page 27.

4.6.1 Partitioning From DOS or Windows

If you are manipulating existing FAT or NTFS partitions, it is recommended that you either use the scheme below or native Windows or DOS tools. Otherwise, it is not really necessary to partition from DOS or Windows; the Linux partitioning tools will generally do a better job.

4.7 Lossless Repartitioning When Starting From DOS, Win-32 or OS/2

One of the most common installations is onto a system that already contains DOS (including Windows 3.1), Win32 (such as Windows 95, 98, NT), or OS/2, and it is desired to put Debian onto the same disk without destroying the previous system. As explained in the 'Background' on page 23, decreasing the size of an existing partition will almost certainly damage the data on that partition unless certain precautions are taken. The method described here, while not guaranteed to protect your data, works extremely well in practice. As a precaution, you should *make a backup*.

Before going any further, you should have decided how you will be dividing up the disk. The method in this section will only split a partition into two pieces. One will contain the original OS and the other will be used for Debian. During the installation of Debian, you will be given the opportunity to use Debian portion of the disk as you see fit, i.e., as swap or as a filesystem.

The idea is to move all the data on the partition to the beginning, before changing the partition information, so that nothing will be lost. It is important that you do as little as possible between the data movement and repartitioning to minimize the chance of a file being written near the end of the partition as this will decrease the amount of space you can take from the partition.

The first thing needed is a copy of `fips` which is available in the `tools/` directory on your nearest Debian mirror. Unzip the archive and copy the files `RESTORRB.EXE`, `FIPS.EXE` and `ERRORS.TXT` to a bootable floppy. A bootable floppy can be created using the command `sys a:` under DOS. `fips` comes with very good documentation which you may want to read. You will definitely need to read the documentation if you use a disk compression driver or a disk manager. Create the disk and read the documentation *before* you defragment the disk.

The next thing needed is to move all the data to the beginning of the partition. `defrag`, which comes standard with DOS 6.0 and later can easily do the job. See the `fips` documentation for a list of other software that may do the trick. Note that if you have Windows 95, you must run `defrag` from there, since DOS doesn't understand VFAT, which is used to support for long filenames, used in Windows 95 and higher.

After running the defragmenter (which can take a while on a large disk), reboot with the `fips` disk you created in the floppy drive. Simply type `a:\fips` and follow the directions.

Note that there are many other other partition managers out there, in case `fips` doesn't do the trick for you.

4.8 Partitioning for DOS

If you are partitioning for DOS drives, or changing the size of DOS partitions, using Linux tools, many people experience problems working with the resulting FAT partitions. For instance, some have reported slow performance, consistent problems with `scandisk`, or other weird errors in DOS or Windows.

Apparently, whenever you create or resize a partition for DOS use, it's a good idea to fill the first few sectors with zeros. Do this prior to running DOS's `format` command, from Linux:

```
dd if=/dev/zero of=/dev/hdXX bs=512 count=4
```

Chapter 5

Methods for Installing Debian

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

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

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

The easiest route for most people will be to use a set of Debian CDs. If you have such a set, and if your machine supports booting directly off the CD, great! Simply configure your system to boot off the CD as described in 'Boot Device Selection' on page 19, insert your CD, reboot, and proceed to the next chapter. If it turns out the standard installation doesn't work for your hardware, you can come back here to see about alternate kernels and installation methods which may work for you. In particular, note that some CD sets provide different kernels on different CDs, so that booting off some CD other than the first may work for you.

5.1 Overview of the Installation Process

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

1. You begin by booting the installation system.
2. You answer a series of questions to perform the initial system configuration.
3. You provide a media source for the kernel and drivers.
4. You select which drivers to load.
5. You provide a media source for the base system.
6. You reboot the system and then do some final configuration.
7. You install additional software, packages, at your discretion.

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

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

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

5.2 Choosing the Right Installation Set

Kernel images are available in various "flavors", each of which supports a different set of hardware. The flavors available for Intel x86 are:

'vanilla' The standard kernel package available in Debian. This includes almost all drivers supported by Linux built as modules, which includes drivers for network devices, SCSI devices, sound cards, Video4Linux devices, etc. The 'vanilla' flavor includes one Rescue Floppy, one root and three Driver Floppies.

'udma66' Very similar to 'vanilla', except it includes Andre Hedrick's IDE patches to support UDMA66 devices.

'compact' Like 'vanilla', but with many of the less-frequently-use drivers removed (sound, v4l, etc). In addition, it has built in support for several popular PCI Ethernet devices — NE2000, 3com 3c905, Tulip, Via-Rhine and Intel EtherExpress Pro100. These built in drivers allow you to take full advantage of the Debian installer's net install feature to install the Driver Floppies

and/or base system over the network so that only the root and Rescue Floppy disks need to be made. Finally, 'compact' also supports several common RAID controllers: DAC960, and Compaq's SMART2 RAID controllers. The 'compact' flavor includes one Rescue Floppy, one root and one driver disk.

'idepci' Kernel that supports only IDE and PCI devices (and a very small number of ISA devices). This kernel should be used if the SCSI drivers in the other flavors cause your system to hang on startup (probably because of resource conflicts, or a misbehaving driver/card in your system.) The 'idepci' flavor also has a built-in ide-floppy driver so that you can install from LS120 or ZIP devices.

Although we have described above how many 1.44MB diskettes the different sets occupy, you may still choose different methods of installation.

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

5.3 Installation Sources for Different Installation Stages

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

5.3.1 Booting the Initial Installation System

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

- the Rescue Floppy
- a bootable CD-ROM
- a hard drive, via a boot loader running in another operating system

5.3.2 Source Media and Installation Stages

The following table indicates which media sources you can use at each stage of the installation process. The columns indicate different install stages, ordered from left to right in the sequence which they occur. The far right column is the installation media. A blank cell indicates that given

source media is not available at that installation stage; Y indicates that it is, and S means that it is in some cases.

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

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

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

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

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

5.3.3 Recommendations

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

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

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

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

5.4 Description of Installation System Files

This section contains an annotated list of files you will find in the `disks-i386` directory. You may not need to download these at all; it all depends on the booting and base system installation media you have chosen.

Most files are floppy disk images; that is, a single file which can be written to a disk to create the necessary floppy disk. These images are, obviously, dependent on the size of the target floppy. For instance, 1.44MB is the normal quantity of data which is what fits on standard 3.5 inch floppies. 1.2MB is the amount of data which normally fits on 5.25 inch floppy disks, so use this image size if you have such a floppy drive. The images for 1.44MB floppy disks can be found in the `images-1.44` directory. Images for 1.2MB floppy disks can be found in the `images-1.20` directory. Images for 2.88MB disks, which are generally only used for CD-ROM booting and the like, are found in the `images-2.88` directory.

If you are using a web browser on a networked computer to read this document, you can probably retrieve the files by selecting their names in your web browser. Depending on your browser you may need to take special action to download directly to a file, in raw binary mode. For example, in Netscape you need to hold the shift key when clicking on the URL to retrieve the file. Files can be downloaded from the URLs in this document, or you can retrieve them from `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/`, or the corresponding directory on any of the Debian mirror sites (`http://www.debian.org/distrib/ftplist`).

5.4.1 Documentation

Installation Manual:

`install.en.txt`

`install.en.html`

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

Partitioning Program Manual Pages:

`fdisk.txt`

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

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/basecont.txt`
Listing of the contents of the base system.

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/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`.

5.4.2 Files for the Initial System Boot

Rescue Floppy images:

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/r`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/r`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/i`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/s`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-2.88/r`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-2.88/c`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-2.88/i`

`http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-2.88/u`
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.

Root image(s):

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/r

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/r

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/i

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u
This file contains an image of a temporary filesystem that gets loaded into memory when you boot from the Rescue Floppy. This is used for installations from hard disk and floppies.

Linux kernel:

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/linux

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/compact/linux

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/idepci/linux

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/udma66/linux
This is the Linux kernel image to be used for hard disk and CD installations. You don't need it if you are installing from floppies.

Linux boot loader for DOS:

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/dosutils/loa
You will need this boot loader if you are installing from a DOS partition or from a CD-ROM.
See 'Booting from a DOS partition' on page 49.

DOS Installer Batch Files:

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/install.bat

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/compact/insta

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/idepci/instal

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/udma66/instal
DOS batch file for starting Debian installation from DOS. This batch file is used in installations from hard disk or CD-ROM. See 'Booting from a DOS partition' on page 49.

5.4.3 Driver Files

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

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

Driver Floppies images:

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.20/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/c

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/i

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/s

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/u

These are the Driver Floppies disk images.

Driver Floppies archive

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/drivers.tgz

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/compact/drive

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/idepci/driver

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/udma66/driver
If you are not limited to diskettes, choose one of these files.

5.4.4 Base System Files

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

Base system images:

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz

or

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/b

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/images-1.44/
These files contain the base system which will be installed on your Linux partition during the installation process. This is the bare minimum necessary for you to be able to install the rest of the packages. The `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz` file is for installation from non-floppy media, i.e., CD-ROM, harddisk, or NFS.

5.4.5 Utilities

http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/dosutils/raw
This is a DOS utility to write a floppy disk image to a floppy. You should not copy images to the floppy, but instead use this utility to "raw write" them.

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

5.5 Diskettes

5.5.1 Floppy Disk Reliability

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

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

If you are having the installation stall at a particular floppy, the first thing you should do is redownload 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.

5.5.2 Booting from Floppies

Booting from floppies is supported for most platforms.

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

If you need to, you can also modify the Rescue Floppy; see 'Replacing the Rescue Floppy Kernel' on page 79.

The Rescue Floppy couldn't fit the root filesystem image, so you'll need the root image to be written to a disk as well. You can create that floppy just as the other images are written to floppies. Once the kernel has been loaded from the Rescue Floppy, you'll be prompted for the root disk. Insert that floppy and continue. See also 'Booting With the Rescue Floppy' on page 51.

5.5.3 Installing Base from Floppies

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

Complete these steps:

1. Obtain these disk images (these files are described in greater detail in 'Description of Installation System Files' on page 35):
   - a Rescue Floppy image
   - the Driver Floppies images
   - the base system disk images, i.e., `base-1.bin`, `base-2.bin`, etc.
   - and a root filesystem image
2. Locate sufficient floppies for all the images you need to write.
3. Create the floppies, as discussed in 'Creating Floppies from Disk Images' on the facing page.
4. If you are not an English speaker, see 'Modifying the Rescue Floppy to Support National Language' on page 44 to have the Rescue Floppy speak your language.
5. Insert the Rescue Floppy into your floppy drive, and reboot the computer.
6. Skip down to 'Booting the Installation System' on page 47.

5.5.4 Creating Floppies from Disk Images

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

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

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

Writing Disk Images From a Linux or Unix System

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

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

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

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

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

You'll find the `rawrite2.exe` program in the same directory as the floppy disk images. There's also a `rawrite2.txt` file containing instructions for using `rawrite2`.

To write the floppy disk image files to the floppy disks, first make sure that you are booted into DOS. Many problems have been reported when trying to use `rawrite2` from within a DOS box from within Windows. Double-clicking on `rawrite2` from within the Windows Explorer is also reported to not work. If you don't know how to boot into DOS, just hit *F8* while booting.

Once you've booted into plain DOS, use the command

```
rawrite2 -f file -d drive
```

where *file* is one of the floppy disk image files, and *drive* is either 'a:' or 'b:', depending on which floppy drive you are writing to.

5.5.5 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: `cs de eo es fi fr hr hu it ja pl pt ru sk sv tr`

Writing Disk Images on Atari Systems

You'll find the `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/rawwrite.ttp` program in the same directory as the floppy disk images. Start the program by double clicking on the program icon, and type in the name of the floppy image file you want written to the floppy at the TOS program command line dialog box.

Writing Disk Images From MacOS

To create floppies from the distribution floppy images on a MacOS system, you can use the MacOS utility `Disk Copy` or the freeware utility `suntar`. The `root.bin` file is an example of a floppy image. First, locate `root.bin` on the offical Debian GNU/Linux CD, or download it from your favorite Debian mirror in *binary* mode. Do not allow any automatic extraction of the file after downloading. The '.bin' extension does not stand for Macbinary, but rather just 'binary' floppy image files. Then use one of the following methods to create a floppy from the floppy image.

Writing Disk Images with `Disk Copy`

1. If you are creating the floppy image from files which were originally on the official Debian GNU/Linux CD, then the Type and Creator are already set correctly. These `Creator-Changer` steps are only necessary if you downloaded the image files.

   (a) Obtain Creator-Changer (`ftp://uiarchive.uiuc.edu/mirrors/ftp/ftp.info-mac.org/info-mac/disk/creator-changer-284.hqx`) and use it to open the `root.bin` file.

   (b) Change the Creator to `ddsk` (Disk Copy), and the Type to `DDim` (binary floppy image). The case is sensitive for these fields.

   (c) *Important:* In the Finder, use `Get Info` to display the Finder information about the floppy image, and 'X' the `File Locked` checkbox so that MacOS will be unable to remove the boot blocks if the image is accidentally mounted.

2. Obtain `Disk Copy`; if you have a MacOS system or CD it will very likely be there already, otherwise try `http://asu.info.apple.com/swupdates.nsf/artnum/n11162`.

3. Run `Disk Copy`, and select 'Make a Floppy' from the `Utilities` menu, then select the *locked* image file from the resulting dialog. It will ask you to insert a floppy, then ask if you really want to erase it. When done it should eject the floppy.

Writing Disk Images with `suntar`

1. Obtain `suntar` from `http://hyperarchive.lcs.mit.edu/HyperArchive/Archive/cmp/suntar-223.hqx`. Start the `suntar` program and select 'Overwrite Sectors...' from the `Special` menu.

2. Insert the floppy disk as requested, then hit return (start at sector 0).

3. Select the `root.bin` file in the file-opening dialog.

4. After the floppy has been created successfully, select 'Eject' from the `File` menu. If there are any errors writing the floppy, simply toss that floppy and try another.

Before using the floppy you created, *set the write protect tab*! Otherwise if you accidently mount it in MacOS, MacOS will helpfully ruin it.

5.6 CD-ROM

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

Installing from CD-ROM is described in 'Booting and/or Installing from a CD-ROM' on page 50.

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

5.7 Hard Disk

Booting from an existing operating system is often a convenient option; for some systems it is the only supported method of installation. This method is described in 'Booting from a Hard Disk' on page 49.

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

5.8 Installing from NFS

Due to the nature of this method of installation, only the base system can be installed via NFS. You will need to have the Rescue Floppy and the Driver Floppies available locally using one of the above methods. To install the base system via NFS, you'll have to go through the regular installation as explained in 'Using `dbootstrap` for Initial System Configuration' on page 55. Do not forget to insert the module (driver) for your Ethernet card, and the file system module for NFS.

When `dbootstrap` asks you where the base system is located ("'Install the Base System"' on page 64), you should choose NFS, and follow the instructions.

Chapter 6

Booting the Installation System

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

Note that on some machines, `Control-Alt-Delete` does not properly reset the machine, so a "hard" reboot is recommended. If you are installing from an existing operating system (e.g., from a DOS box) you don't have a choice. Otherwise, please do a hard boot when booting.

6.1 Boot Parameter Arguments

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

If you are booting from the Rescue Floppy or from CD-ROM you will be presented with the boot prompt, `boot:`. Details about how to use boot parameters with the Rescue Floppy can be found in 'Booting With the Rescue Floppy' on page 51. If you are booting from an existing operating system, you'll have to use other means to set boot parameters. For instance, if you are installing from DOS, you can edit the `install.bat` file with any text editor. Full information on boot parameters can be found in the Linux BootPrompt HOWTO (`http://www.linuxdoc.org/HOWTO/BootPrompt-HOWTO.html`); this section contains only a sketch of the most salient parameters.

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

When the kernel boots, a message `Memory: 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.

Some systems have floppies with "inverted DCLs". If you receive errors reading from the floppy, even when you know the floppy is good, try the parameter `floppy=thinkpad`.

On some systems, such as the IBM PS/1 or ValuePoint (which have ST-506 disk drivers), the IDE drive may not be properly recognized. Again, try it first without the parameters and see if the IDE drive is recognized properly. If not, determine your drive geometry (cylinders, heads, and sectors), and use the parameter `hd=`*`cylinders`*`,`*`heads`*`,`*`sectors`*.

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

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

Again, full details on boot parameters can be found in the Linux BootPrompt HOWTO (`http://www.linuxdoc.org/HOWTO/BootPrompt-HOWTO.html`), including tips for obscure hardware. Some common gotchas are included below in 'Troubleshooting the Boot Process' on page 52.

6.1.1 `dbootstrap` Arguments

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

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

verbose Ask even more questions than usual.

debug Emit additional debug messages to the installation system log (see 'Using the Shell and Viewing the Logs' on page 55), including every command run.

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

mono Use monochrome rather than color mode.

6.2 Interpreting the Kernel Startup Messages

During the boot sequence, you may see many messages in the form `can't find something`, or `something not present`, `can't initialize something`, or even `this driver release`

`depends on something`. Most of these messages are harmless. You see them because the kernel for the installation system is built to run on computers with many different peripheral devices. Obviously, no one computer will have every possible peripheral device, so the operating system may emit a few complaints while it looks for peripherals you don't own. You may also see the system pause for a while. This happens when it is waiting for a device to respond, and that device is not present on your system. If you find the time it takes to boot the system unacceptably long, you can create a custom kernel later (see 'Compiling a New Kernel' on page 75).

6.3 Booting from a Hard Disk

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

6.3.1 Booting from a DOS partition

It is possible to install Debian from an already installed DOS partition on the same machine. You have two alternatives: either try the floppy-less installation, or boot from the Rescue Floppy but install base from the local disk.

To try floppyless booting, follow these directions:

1. Get the following files from your nearest Debian FTP mirror and put them into a directory on your DOS partition. Be sure to retain their subdirectory structure, e.g., `images-1.44\compact\rescue.`

   - One of the Rescue Floppy images, one of the root images, one of the Linux kernel files, and one of the DOS batch files from 'Files for the Initial System Boot' on page 36. See 'Choosing the Right Installation Set' on page 32 for help deciding which kernel to use.
   - One of the Driver Floppies archives from 'Driver Files' on page 38; it must correspond to the kernel flavor you chose above.
   - `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz` (see 'Base System Files' on page 40)
   - `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/dosutils/loadlin.exe` (see 'Files for the Initial System Boot' on page 36)

2. Boot into DOS (not Windows) without any drivers being loaded. To do this, you have to press *F8* at exactly the right moment (and optionally select the 'safe mode command prompt only' option).

3. Enter the subdirectory for the flavor you chose, e.g., `cd c:\debian\compact`. Next, execute `install.bat`.

4. Skip down to 'Using `dbootstrap` for Initial System Configuration' on page 55.

If you want to boot from floppies, but install base from a DOS partition, then simply download and create the Rescue Floppy and Driver Floppies as described in 'Creating Floppies from Disk Images' on page 43. Download `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz` and place that file somewhere on a DOS partition.

6.3.2 Installing from a Linux Partition

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

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

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

1. Get the following files and place them in a directory on your Linux partition. Use the largest possible files for your architecture:
   - a Rescue Floppy image, see 'Files for the Initial System Boot' on page 36
   - one of the Driver Floppies archives from 'Driver Files' on page 38
   - `http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz`
2. You can use any other functional boot method when installing from a partition. The following assumes you are booting with floppies; however, any boot installation can be used.
3. Create the Rescue Floppy as discussed in 'Creating Floppies from Disk Images' on page 43. Note that you won't need the Driver Floppies.
4. Insert the Rescue Floppy into your floppy drive, and reboot the computer.
5. Skip down to 'Using `dbootstrap` for Initial System Configuration' on page 55.

6.4 Booting and/or Installing from a CD-ROM

If you have a CD which is bootable, and if your architecture and system supports booting from a CD-ROM, you don't need any floppies. Often, it's as simple as puting the CD-ROM in the CD drive and booting. You may need to configure your hardware as indicated in 'Boot Device

Selection' on page 19. Then put the CD-ROM into the drive, and reboot. The system should boot up, and you should be presented with the `boot:` prompt. Here you can enter your boot arguments, or just hit *enter*.

Note that official Debian CD-ROM sets for Intel x86 will boot different "flavors" depending on which CD-ROM you boot from. See 'Choosing the Right Installation Set' on page 32 for a discussion of the different flavors. Here's how the flavors are laid out on the different CD-ROMs:

CD 1 Boots the 'vanilla' flavor.

CD 2 Boots the 'compact' flavor.

CD 3 Boots the 'idepci' flavor (2.2r3 or better only)

CD 4 Boots the 'udma66' flavor (2.2r3 or better only)

So, if you want to boot from one of the above flavors, put that CD in the drive for booting.

If your hardware does not support bootable CD-ROMs, you should boot into DOS, and execute the `boot.bat` file which is located in the `\boot` directory on your CD. Then, skip down to 'Using `dbootstrap` for Initial System Configuration' on page 55.

Even if you cannot boot from CD-ROM, you can install the base Debian system from CD-ROM. Simply boot using a different media, such as floppies. When it is time to install the base system and any additional packages, point the installation system at the CD-ROM drive as described in "'Install the Base System"' on page 64.

6.5 Booting With the Rescue Floppy

Booting from the Rescue Floppy is easy: place the Rescue Floppy in the primary floppy drive, and reset the system by pressing *reset*, or by turning the system off and on. As mentioned above, doing a "hard reboot" is recommended. The floppy disk should be accessed, and you should then see a screen that introduces the Rescue Floppy and ends with the `boot:` prompt.

If you are using an alternative way to boot the system, follow the instructions, and wait for the `boot:` prompt to come up. If you boot from floppies smaller than 1.44MB, or, in fact, whenever you boot from floppy on your architecture, you have to use a ramdisk boot method, and you will need the Root Disk.

You can do two things at the `boot:` prompt. You can press the function keys *F1* through *F10* to view a few pages of helpful information, or you can boot the system.

Information on boot parameters which might be useful can be found by pressing *F4* and *F5*. If you add any parameters to the boot command line, be sure to type the boot method (the default is

`linux`) and a space before the first parameter (e.g., `linux floppy=thinkpad`). If you simply press *Enter*, that's the same as typing `linux` without any special parameters.

The disk is called the Rescue Floppy because you can use it to boot your system and perform repairs if there is ever a problem that makes your hard disk unbootable. Thus, you should save this floppy after you've installed your system. Pressing *F3* will give further information on how to use the Rescue Floppy.

Once you press *Enter*, you should see the message `Loading...`, followed by `Uncompressing Linux...`, and then a screenful or so of information about the hardware in your system. More information on this phase of the boot process can be found below.

If you choose a non-default boot method, e.g., "ramdisk" or "floppy", you will be prompted to insert the Root Floppy. Insert the Root Floppy into the first disk drive and press *Enter*. (If you choose floppy1 insert the Root Floppy into the second disk drive.)

6.6 Troubleshooting the Boot Process

If you have problems and the kernel hangs during the boot process, doesn't recognize peripherals you actually have, or drives are not recognized properly, the first thing to check is the boot parameters, as discussed in 'Boot Parameter Arguments' on page 47.

Often, problems can be solved by removing add-ons and peripherals, and then trying booting again. Internal modems, sound cards, and Plug-n-Play devices can be especially problematic.

If you have a very old machine, and the kernel hangs after saying `Checking 'hlt' instruction...`, then you should try the `no-hlt` boot argument, which disables this test.

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

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

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

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

```
flavor:        flavor of image you are using
architecture:  i386
model:         your general hardware vendor and model
```

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

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

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

Chapter 7

Using `dbootstrap` for Initial System Configuration

7.1 Introduction to `dbootstrap`

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

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

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

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

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

7.1.1 Using the Shell and Viewing the Logs

If you are an experienced Unix or Linux user, press *Left Alt-F2* to get to the second *virtual console*. That's the *Alt* key on the left-hand side of the space bar, and the *F2* function key, at the same time.

This is a separate window running a Bourne shell clone called ash. At this point you are booted from the RAM disk, and there is a limited set of Unix utilities available for your use. You can see what programs are available with the command ls /bin /sbin /usr/bin /usr/sbin. Use the menus to perform any task that they are able to do – the shell and commands are only there in case something goes wrong. In particular, you should always use the menus, not the shell, to activate your swap partition, because the menu software can't detect that you've done this from the shell. Press *Left Alt-F1* to get back to menus. Linux provides up to 64 virtual consoles, although the Rescue Floppy only uses a few of them.

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

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

7.2 "Release Notes"

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

7.3 "Debian GNU/Linux Installation Main Menu"

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

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

already done. It should say "Next", and at this point the next step in installing the system will be taken.

7.4 "Configure the Keyboard"

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

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

If you are installing a diskless workstation, the next few steps will be skipped, since there are no local disks to partition. In that case, your next step will be "'Configure the Network"' on page 63. After that, you will be prompted to mount your NFS root partition in "'Mount a Previously-Initialized Partition"' on page 60.

7.5 Last Chance!

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

7.6 "Partition a Hard Disk"

If you have not already partitioned your disks for Linux native and Linux swap filesystems, i.e., as described in 'Partitioning Prior to Installation' on page 29, the next step will be "Partition a Hard Disk". If you have already created at least one Linux native and one Linux swap disk partition, the "Next" menu selection will be "Initialize and Activate a Swap Partition", or you may even skip that step if your system had low memory and you were asked to activate the swap partition as soon as the system started. Whatever the "Next" menu selection is, you can use the down-arrow key to select "Partition a Hard Disk".

The "Partition a Hard Disk" menu item presents you with a list of disk drives you can partition, and runs a partitioning application. You must create at least one "Linux native" (type 83) disk partition, and you probably want at least one "Linux swap" (type 82) partition, as explained in

'Partitioning Your Hard Drive' on page 23. If you are unsure how to partition your system, go back and read that chapter.

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

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

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. Be careful, and see the Linux+FreeBSD HOWTO (http://www.linuxdoc.org/HOWTO/mini/Linux+FreeBSD-2.html).

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

Note that cfdisk doesn't understand FreeBSD partitions at all, and, again, device names may differ as a result.

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

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

Remember to mark your boot partition as "Bootable".

7.7 "Initialize and Activate a Swap Partition"

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

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

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

7.8 "Initialize a Linux Partition"

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

You can initialize a Linux partition, or alternately you can mount a previously-initialized one. Note that dbootstrap will *not* upgrade an old system without destroying it. If you're upgrading, Debian can usually upgrade itself, and you won't need to use dbootstrap. For help on upgrading to Debian 2.2, see the upgrade instructions (http://www.debian.org/releases/2.2/i386/release-notes/).

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

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

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

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

The next prompts are just confirmation steps. You will be asked to confirm your action, since initializing is destructive to any data on the partition, and you will be informed that the partition is being mounted as /, 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.

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

7.9 "Mount a Previously-Initialized Partition"

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

If you are installing a diskless workstation, at this point, you want to NFS mount your root partition from the remote NFS server. Specify the path to the NFS server in standard NFS syntax, namely, *server-name-or-IP*:*server-share-path*. If you need to mount additional filesystems as well, you can do that at this time.

If you have not already setup your network as described in "'Configure the Network"' on page 63, then selecting an NFS install will prompt you to do so.

7.10 Mounting Partitions Not Supported by dbootstrap

In some special situations, dbootstrap might not know how to mount your filesystems (whether root or otherwise). It may be possible, if you're an experienced Linux user, to simply go to tty2 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.

7.11 "Install Operating System Kernel and Modules"

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

You will be offered a menu of devices from which you can install the kernel. Choose the appropriate device from which to install the kernel and modules. Remember that you can use any devices which is available to you, and that you are not restricted to using the same media you used to mount with (see 'Methods for Installing Debian' on page 31).

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 filesystem, you have a choice between two options. Select "harddisk" if the disk partition is not yet mounted; select "mounted" if it is. In both cases, the system will first look for some files in `dists/potato/main/disks-i386/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 discussed in 'Booting from a Hard Disk' on page 49. 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/potato/main/disks-i386/current`. You can type in the path manually, or use the `<...>` button to browse through the filesystem tree.

Continuing the discusssion on installation from a local disk or similar medium (such as NFS), you will next be prompted for the actual directory containing the needed files (which may be based on your subarchitecture). Note that the system may be quite insistent that the files appear in the precise location indicated, including the subdirectories, if any. See the logs in tty3 (see 'Using the Shell and Viewing the Logs' on page 55) 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 13). 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 page 63), and then re-run this step.

7.11.1 NFS

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

7.11.2 Network

Select the "network" option, and then tell `dbootstrap` the URL and path to the Debian archive. The default will usually work fine, and in any case, the path part is probably correct for any official

Debian mirror, even if you edit the server part. You may choose to pull the files in through a proxy server; just enter the server **. . . this sentence isn't finished. . .**

7.11.3 NFS Root

If you are installing a diskless workstation, you should have already configured your networking as described in "'Configure the Network"' on 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.

7.12 "Configure PCMCIA Support"

There is an alternate step, *before* the "Configure Device Driver Modules" menu selection, called "Configure PCMCIA Support". This menu is used to enable PCMCIA support.

If you do have PCMCIA, but are not installing your Debian system using it (e.g., installation with a PCMCIA Ethernet card), then you need not configure PCMCIA at this point. You can easily configure and enable PCMCIA at a later point, after installation is complete. However, if you are installing by way of a PCMCIA network device, this alternate must be selected, and PCMCIA support must be configured prior to configuring the network.

If you need to install PCMCIA, select the alternate, below "Configure Device Driver Modules". You will be asked which PCMCIA controller your system contains. In most cases, this will be `i82365`. In some cases, it will be `tcic`; your laptop's vendor-supplied specifications should provide the information if in doubt. You can generally leave the next few sets of options blank. Again, certain hardware has special needs; the Linux PCMCIA HOWTO (`http://www.linuxdoc.org/HOWTO/PCMCIA-HOWTO.html`) contains plenty of information in case the default doesn't work.

In some unusual cases, you may also need to read and edit `/etc/pcmcia/config.opts`. You can open your second virtual terminal (*Left Alt-F2*) and edit the file there, and then reconfigure your PCMCIA, or manually forcing a reload of the modules using `insmod` and `rmmod`.

Once PCMCIA is properly configured and installed, you should jump back up and configure your device drivers as described in the next section.

7.13 "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.

7.14 "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.

If `dbootstrap` detects that you configured PCMCIA ("'Configure PCMCIA Support"' on the facing page), you will be asked to confirm that your network card is a PCMCIA card. This affects

how and where the network configuration is set.

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 current 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 17. The system will also summarize your network information and ask you for confirmation. Next, you need to specify the network device that your primary network connection uses. Usually, this will be "eth0" (the first Ethernet device).

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

7.15 "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 choose to install from a filesystem on the harddisk or from a non-official CD-ROM, you will be prompted to specify the path to the http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz file. If you have official media, the default value should be correct. Otherwise, enter the path where the base system can be found, relative to the media's mount point. As with the "Install Operating System Kernel and Modules" step, you can either let dbootstrap find the file itself or type in the path at the prompt.

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

If you are installing the base system from NFS, then choose NFS and continue. You'll be prompted to specify the server, the share on the server, and the subdirectory within that share where the http://http.us.debian.org/debian/dists/potato/main/disks-i386/current/base2_2.tgz file can be found. If you have problems mounting NFS, make sure that the system time on the NFS server more or less agrees with the system time on the client. You can set your date on tty2 using the date command; you'll have to set it by hand. See the date(1) manual page.

7.16 "Configure the Base System"

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

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

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

7.17 "Make Linux Bootable Directly From Hard Disk"

If you elect to make the hard disk boot directly to Linux, and you are *not* installing a diskless workstation, you will be asked to install a master boot record. If you aren't using a boot manager (and this is probably the case if you don't know what a boot manager is) and you don't have another different operating system on the same machine, answer "Yes" to this question. Note that if you answer "Yes", you won't be able to boot into DOS normally on your machine, for instance. Be careful, and see 'Reactivating DOS and Windows' on page 74. If you answer "Yes", the next question will be whether you want to boot Linux automatically from the hard disk when you turn on your system. This sets Linux root partition to be the *bootable partition* – the one that will be loaded from the hard disk.

Note that multiple operating system booting on a single machine is still something of a black art. This document does not even attempt to document the various boot managers, which vary by architecture and even by subarchitecture. You should see your boot manager's documentation

for more information. Remember: when working with the boot manager, you can never be too careful.

The standard i386 boot loader is called "LILO". It is a complex program which offers lots of functionality, including DOS, NT, and OS/2 boot management. Please carefully read the instructions in the directory `/usr/share/doc/lilo/` if you have special needs; also see the LILO mini-HOWTO (`http://www.linuxdoc.org/HOWTO/mini/LILO.html`).

You can skip this step for now, and set the bootable partition later with the Linux `fdisk` or `activate` programs.

If you mess up and can no longer boot into DOS, you'll need to use a DOS boot disk and use the `fdisk /mbr` command to reinstall the DOS master boot record – however, this means that you'll need to use some other way to get back into Debian! For more information on this please read 'Reactivating DOS and Windows' on page 74.

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

7.18 "Make a Boot Floppy"

You may wish to make a boot floppy even if you intend to boot the system from the hard disk. The reason for this is that it's possible for the hard disk bootstrap to be mis-installed, but a boot floppy will almost always work. Select "Make a Boot Floppy" from the menu and feed the system a blank floppy as directed. Make sure the floppy isn't write-protected, as the software will format and write it. Mark this the "Custom Boot" floppy and write-protect it once it has been written.

This floppy will contain a kernel and a simple filesystem, with a directive to use your new root filesystem.

7.19 The Moment of Truth

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

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

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

7.20 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 `base-config` at any point after installation is complete, as root run `dpkg-reconfigure base-config`.

7.21 MD5 Passwords

You will first 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".

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

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

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

7.23 Set the Root Password

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

used for as short a time as possible.

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

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

7.24 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.

7.25 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 Linux. To view files, you should use more, and zmore for compressed files with a .gz extension. For example, to view README.debian.gz, type zmore README.debian.gz. The base system comes with two editors: ae, which is very simple to use, but does not have a lot of features, and elvis-tiny, a limited clone of vi. You will probably want to install more full-featured editors and viewers later, such as nvi, less, and emacs.

Edit /etc/ppp/peers/provider and replace "/dev/modem" with "/dev/ttyS#" where # stands for the number of your serial port. In Linux, serial ports are counted from 0; your first serial port (i.e., COM1) is /dev/ttyS0 under Linux. The next step is to edit /etc/chatscripts/provider and insert your provider's phone number, your user-name and password. Please do not delete the "\q" that precedes the password. It hides the password from appearing in your log files.

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

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

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

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

7.26 Removing PCMCIA

If you have no use for PCMCIA, you can choose to remove it at this point. This will make your startup cleaner; also, it will make it easier to replace your kernel (PCMCIA requires a lot of correlation between the version of the PCMCIA drivers, the kernel modules, and the kernel itself).

7.27 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.[2] 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, 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 filesystem. 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.

7.27.1 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.

For any of the network package sources, you will be prompted whether you wish to use "non-US software". You will generally wish to say "yes", because otherwise you won't be able to install cryptographically secure software, such as the popular `ssh` program.

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

[2] 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 whereever.

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.

7.28 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.

7.29 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 do you want to install on your new machine. This is the purpose of the dselect program, described below. But this can be a long task with around 3900 packages available in Debian!

So, you have the ability to choose *tasks* instead. These loosely represent a number of different jobs or things you want to do with your computer, such as "Samba" for SAMBA servers, or "Gnome Desktop" for the GNOME desktop environment.

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 included for that task.

Once you've selected your tasks, select "Finish". At this point, apt-get will be run to install the packages you've selected. You will be shown the number of packages to be installed, and how many kilobytes of packages, if any, need to be downloaded.

There are two caveats to be mentioned at this point. Firstly, of the 3900 packages available in Debian, only a small minority of those 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.

The second caveat is that some so-called "standard" packages are not installed by default. Thus, some software, which we consider basic to any Linux system, may not be installed.[3] In order to install that software, simply run tasksel -s, without selecting any packages, then select "Finish".

7.30 Advanced Package Selection with dselect

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

7.31 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.

[3] This is due to a bug in base-config which we have fixed for the next release. We decided not to change this after Potato release, since it was a rather large change, and too likely to cause problems.

Chapter 8

Next Steps and Where to Go From Here

8.1 If You Are New to Unix

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

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

Information specific to Debian can be found below.

8.2 Orienting Yourself to Debian

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

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

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

- `/var` (you could make `/var/local` and be safe in there)
- `/bin`
- `/sbin`
- `/lib`

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

8.3 Reactivating DOS and Windows

After installing the base system and writing to the *Master Boot Record*, you will be able boot Linux, but probably nothing else. This depends what you have chosen during the installation. This chapter will describe how you can reactivate your old systems so that you can also boot your DOS or Windows again.

`LILO` is a boot manager with which you can also boot other operating systems than Linux, which complies to PC conventions. The boot manager is configured via `/etc/lilo.conf` file. Whenever you edited this file you have to run `lilo` afterwards. The reason for this is that the changes will take place only when you call the program.

Important parts of the `lilo.conf` file are the lines containing the `image` and `other` keywords, as well as the lines following those. They can be used to describe a system which can be booted by `LILO`. Such a system can include a kernel (`image`), a root partition, additional kernel parameters, etc. as well as a configuration to boot another, non-Linux (`other`) operating system. These keywords can also be used more than once. The ordering of these systems within the configuration file is important because it determines which system will be booted automatically after, for instance, a timeout (`delay`) presuming `LILO` wasn't stopped by pressing the *shift*-key.

After a fresh install of Debian, just the current system is configured for booting with `LILO`. If you want to boot another Linux kernel, you have to edit the configuration file `/etc/lilo.conf` to add the following lines:

```
image=/boot/vmlinuz.new
  label=new
  append="mcd=0x320,11"
  read-only
```

For a basic setup just the first two lines are necessary. If you want to know more about the other two options please have a look at the `LILO` documentation. This can be found in `/usr`

`/share/doc/lilo/`. The file which should be read is `Manual.txt`. To have a quicker start into the world of booting a system you can also look at the `LILO` manpages `lilo.conf(5)` for an overview of configuration keywords and `lilo(8)` for description of the installation of the new configuration into the boot sector.

Notice that there are other boot loaders available in Debian GNU/Linux, such as GRUB (in `grub` package), CHOS (in `chos` package), Extended-IPL (in `extipl` package), loadlin (in `loadlin` package) etc.

8.4 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/doc` as well. In particular, `/usr/doc/HOWTO` and `/usr/doc/FAQ` contain lots of interesting information.

The Debian web site (`http://www.debian.org/`) contains a large quantity of documentation about Debian. In particular, see the Debian FAQ (`http://www.debian.org/doc/FAQ/`) and the Debian Mailing List Archives (`http://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.

8.5 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

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-2.2.19` (the most recent version at the time of this writing), `fakeroot` and a few others which are probably already installed (see `/usr/share/doc/kernel-package/README.gz` for the complete list).

Note that you don't *have* to compile your kernel the "Debian way"; but we find that using the packaging system to manage your kernel is actually safer and easier. In fact, you can get your kernel sources right from Linus instead of `kernel-source-2.2.19`, yet still use the `kernel-package` compilation method.

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

Hereafter, we'll assume your kernel source will be located in `/usr/local/src` and that your kernel version is 2.2.19. 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-2.2.19.tar.bz2`), change your directory to it (`cd kernel-source-2.2.19/`). 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.

If you require PCMCIA support, you'll also need to install the `pcmcia-source` package. Unpack the gzipped tar file as root in the directory `/usr/src` (it's important that modules are found where they are expected to be found, namely, `/usr/src/modules`). Then, as root, do `make-kpkg modules_image`.

Once the compilation is complete, you can install your custom kernel like any package. As root, do `dpkg -i ../kernel-image-2.2.19-`*`subarch`*`_custom.1.0_i386.deb`. The *subarch* part is an optional sub-architecture, such as "i586", 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-2.2.19` will be installed, containing your current configuration set. Your new `kernel-image-2.2.19` package is also clever enough to automatically use you're platform's boot-loader to run an update on the booting, allowing you to boot without re-running the boot loader. If you have created a modules package, e.g., if you have PCMCIA, you'll need to install that package as well.

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

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

Chapter 9

Technical Information on the Boot Floppies

9.1 Source Code

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

9.2 Rescue Floppy

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

9.3 Replacing the Rescue Floppy Kernel

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

- RAM disk support (`CONFIG_BLK_DEV_RAM`)
- Initial RAM disk (initrd) support (`CONFIG_BLK_DEV_INITRD`)
- Kernel support for ELF binaries (`CONFIG_BINFMT_ELF`)

- Loop device support (`CONFIG_BLK_DEV_LOOP`)
- FAT, Minix, and Ext2 filesystems (some architectures don't need FAT and/or Minix filesystems – see the source)
- Socket filtering for DHCP (`CONFIG_FILTER`)
- Packet socket, also for DHCP (`CONFIG_PACKET`)
- Unix domain sockets for syslogging – it is provided as a module in the vanilla flavor (`CONFIG_UNIX`)

Copy your new kernel to the file `linux` on the Rescue Floppy, and then run the shell script `rdev.sh` that you'll find on the floppy. The `rdev.sh` script assumes the kernel is in the current directory, or else `/mnt/linux`. If not, you should supply the path as an argument to the script.

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

9.4 The Base Floppies

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

Once this archive is installed, you must go through the steps described in "'Configure the Base System"' on page 65, and other `dbootstrap` menu items to configure the network, and you must install the operating system kernel and modules on your own. Once you have done that, the system should be usable.

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

Chapter 10

Appendix

10.1 Further Information and Obtaining Debian GNU/Linux

10.1.1 Further Information

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

10.1.2 Obtaining Debian GNU/Linux

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

10.1.3 Debian Mirrors

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

10.1.4 GPG, SSH and other Security Software

United States laws place restrictions on the export of defense articles, which, unfortunately, includes some types of cryptographic software. PGP and ssh, among others, fall into this category.

It is legal however, to import such software into the US.

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

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

10.2 Linux Devices

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

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

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

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

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

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

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

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

Chapter 11

Administrivia

11.1 About This Document

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

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

11.2 Contributing to This Document

If you have problems or suggestions regarding this document, you should probably submit them as a bug report against the package `boot-floppies`. See the `bug` or `reportbug` package or read the online documentation of the Debian Bug Tracking System (`http://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/~checkout~/boot-floppies/README-CVS?tag=HEAD%26content-type=text/plain`) from the sources.

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

11.3 Major Contributions

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

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

11.4 Trademark Acknowledgement

All trademarks are property of their respective trademark owners.