Debian GNU/Linux Installation Guide

Debian GNU/Linux Installation Guide

This document contains installation instructions for the Debian GNU/Linux 3.1 system (codename “sarge”), 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.

> **Note:** Although this installation guide for i386 is mostly up-to-date, we plan to make some changes and reorganize parts of the manual after the official release of sarge. A newer version of this manual may be found on the Internet at the `debian-installer` home page (http://www.debian.org/devel/debian-installer/). You may also be able to find additional translations there.

Table of Contents

List of Tables

Installing Debian GNU/Linux 3.1 For i386

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

We understand that many of you want to install Debian without reading this manual, and the Debian installer is designed to make this possible. If you don't have time to read the whole Installation Guide right now, we recommend that you read the Installation Howto, which will walk you through the basic installation process, and links to the manual for more advanced topics or for when things go wrong. The Installation Howto can be found in Appendix A.

With that said, we hope that you have the time to read most of this manual, and doing so will lead to a more informed and likely more successful installation experience.

Chapter 1. Welcome to Debian

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

1.1. What is Debian?

Debian is an all-volunteer organization dedicated to developing free software and promoting the ideals of the Free Software Foundation. The Debian Project began in 1993, when Ian Murdock issued an open invitation to software developers to contribute to a complete and coherent software distribution based on the relatively new Linux kernel. That relatively small band of dedicated enthusiasts, originally funded by the Free Software Foundation (http://www.fsf.org/) and influenced by the GNU (http://www.gnu.org/gnu/the-gnu-project.html) philosophy, has grown over the years into an organization of around 900 *Debian Developers*.

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

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

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

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

- The Linux Standard Base (http://www.linuxbase.org/) (LSB) is a project aimed at standardizing the basic GNU/Linux system, which will enable third-party software and hardware developers to easily design programs and device drivers for Linux-in-general, rather than for a specific GNU/Linux distribution.
- The Filesystem Hierarchy Standard (http://www.pathname.com/fhs/) (FHS) is an effort to standardize the layout of the Linux file system. The FHS will allow software developers to concentrate their efforts on designing programs, without having to worry about how the package will be installed in different GNU/Linux distributions.

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

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

1.2. What is GNU/Linux?

Linux is an operating system: a series of programs that let you interact with your computer and run other programs.

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".

Linux is modelled on the Unix operating system. From the start, Linux was designed to be a multi-tasking, multi-user system. These facts are enough to make Linux different from other well-known operating systems. However, Linux is even more different than you might imagine. In contrast to other operating systems, nobody owns Linux. Much of its development is done by unpaid volunteers.

Development of what later became GNU/Linux began in 1984, when the Free Software Foundation (http://www.gnu.org/) began development of a free Unix-like operating system called GNU.

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).

While many groups and individuals have contributed to Linux, the largest single contributor is still the Free Software Foundation, which created not only most of the tools used in Linux, but also the philosophy and the community that made Linux possible.

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://www.kerneltraffic.org/kernel-traffic/index.html). More information about the **linux-kernel** mailing list can be found on the linux-kernel mailing list FAQ (http://www.tux.org/lkml/).

Linux users have immense freedom of choice in their software. For example, Linux users can choose from a dozen different command line shells and several graphical desktops. This selection is often bewildering to users of other operating systems, who are not used to thinking of the command line or desktop as something that they can change.

Linux is also less likely to crash, better able to run more than one program at the same time, and more secure than many operating systems. With these advantages, Linux is the fastest growing operating system in the server market. More recently, Linux has begun to be popular among home and business users as well.

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.

Debian is especially popular among advanced users because of its technical excellence and its deep commitment to the needs and expectations of the Linux community. Debian also introduced many features to Linux that are now commonplace.

For example, Debian was the first Linux distribution to include a package management system for easy installation and removal of software. It was also the first Linux distribution that could be upgraded without requiring reinstallation.

Debian continues to be a leader in Linux development. Its development process is an example of just how well the Open Source development model can work — even for very complex tasks such as building and maintaining a complete operating system.

The feature that most distinguishes Debian from other 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 160 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. 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, so you can easily find the nearest one.

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

1.5. Getting the Newest Version of This Document

This document is constantly being revised. Be sure to check the Debian 3.1 pages (http://www.debian.org/releases/sarge/) for any last-minute information about the 3.1 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/sarge/i386/).

1.6. 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 Chapter 2.
2. Backup your system, perform any necessary planning and hardware configuration prior to installing Debian, in Chapter 3. If you are preparing a multi-boot system, you may need to create partition-able space on your hard disk for Debian to use.
3. In Chapter 4, you will obtain the necessary installation files for your method of installation.
4. Chapter 5 describes booting into the installation system. This chapter also discusses troubleshooting procedures in case you have problems with this step.
5. Perform the actual installation according to Chapter 6. This involves choosing your language, configuring peripheral driver modules, configuring your network connection, so that remaining installation files can be obtained directly from a Debian server (if you are not installing from a CD), partitioning your hard drives and installation of minimal working system. (Some background about setting up the partitions for your Debian system is explained in Appendix B.)
6. Boot into your newly installed base system and run through some additional configuration tasks, from Chapter 7.
7. Install additional software in Section 7.2.5.

Once you've got your system installed, you can read Chapter 8. That chapter explains where to look to find more information about Unix and Debian, and how to replace your kernel.

Finally, information about this document and how to contribute to it may be found in Appendix D.

1.7. 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. This system's license

isn't like that at all. We encourage you to put a copy of 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.

> **Note:** 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.

Many of the programs in the system are licensed under the *GNU General Public License*, often simply referred to as "the GPL". The GPL requires you to make the *source code* of the programs available whenever you distribute a binary copy of the program; that provision of the license ensures that any user will be able to modify the software. Because of this provision, the source code[1] for all such programs is available in the Debian system.

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

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

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

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

Chapter 2. System Requirements

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

2.1. Supported Hardware

Debian does not impose hardware requirements beyond the requirements of the Linux kernel and the GNU tool-sets. Therefore, any architecture or platform to which the Linux kernel, libc, **gcc**, etc. have been ported, and for which a Debian port exists, can run Debian. Please refer to the Ports pages at http://www.debian.org/ports/i386/ for more details on Intel x86 architecture systems which have been tested with Debian.

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 3.1 supports eleven major architectures and several variations of each architecture known as “flavors”.

Architecture	Debian Designation	Subarchitecture	Flavor
Intel x86-based	i386		vanilla
			speakup
			linux26
Motorola 680x0	m68k	Atari	atari
		Amiga	amiga
		68k Macintosh	mac
		VME	bvme6000
			mvme147
			mvme16x
DEC Alpha	alpha		
Sun SPARC	sparc		sun4cdm
			sun4u
ARM and StrongARM	arm		netwinder
			riscpc
			shark
			lart
IBM/Motorola PowerPC	powerpc	CHRP	chrp

Architecture	Debian Designation	Subarchitecture	Flavor
		PowerMac	pmac
		PReP	prep
		APUS	apus
HP PA-RISC	hppa	PA-RISC 1.1	32
		PA-RISC 2.0	64
Intel ia64-based	ia64		
MIPS (big endian)	mips	SGI Indy/Indigo 2	r4k-ip22
			r5k-ip22
		Broadcom BCM91250A (SWARM)	sb1-swarm-bn
MIPS (little endian)	mipsel	Cobalt	cobalt
		DECstation	r4k-kn04
			r3k-kn02
		Broadcom BCM91250A (SWARM)	sb1-swarm-bn
IBM S/390	s390	IPL from VM-reader and DASD	generic
		IPL from tape	tape

This document covers installation for the *Intel x86* architecture. If you are looking for information on any of the other Debian-supported architectures take a look at the Debian-Ports (http://www.debian.org/ports/) pages.

2.1.2. CPU, Main Boards, and Video Support

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

2.1.2.1. CPU

Nearly all x86-based processors are supported; this includes AMD and VIA (former Cyrix) processors as well. Also the new processors like Athlon XP and Intel P4 Xeon are supported. However, Linux will *not* run on 286 or earlier processors.

2.1.2.2. 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).

2.1.3. 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. Most AGP, PCI and PCIe video cards work under XFree86. Details on supported graphics buses, cards, monitors, and pointing devices can be found at http://www.xfree86.org/. Debian 3.1 ships with XFree86 version 4.3.0.

2.1.4. 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.linux-laptop.net/)

2.1.5. Multiple Processors

Multi-processor support — also called "symmetric multi-processing" or SMP — is supported for this architecture, and is supported by a precompiled Debian kernel image. Depending on your install media, this SMP-capable kernel may or may not be installed by default. 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 should check to see if a kernel package that supports SMP is installed, and if not, choose an appropriate kernel package. You can also build your own customized kernel to support SMP. You can find a discussion of how to do this in Section 8.5. At this time (kernel version 2.6.8) the way you enable SMP is to select "Symmetric multi-processing support" in the "Processor type and features" section of the kernel config.

2.2. Installation Media

This section will help you determine which different media types you can use to install Debian. For example, if you have a floppy disk drive on your machine, it can be used to install Debian. There is a whole chapter devoted media, Chapter 4, which lists the advantages and disadvantages of each media type. You may want to refer back to this page once you reach that section.

2.2.1. Floppies

In some cases, you'll have to do your first boot from floppy disks. Generally, all you will need is a high-density (1440 kilobytes) 3.5 inch floppy drive.

2.2.2. CD-ROM/DVD-ROM

Note: Whenever you see "CD-ROM" in this manual, it applies to both CD-ROMs and DVD-ROMs, because both technologies are really the same from the operating system's point of view, except for some very old nonstandard CD-ROM drives which are neither SCSI nor IDE/ATAPI.

CD-ROM based installation is supported for some architectures. On machines which support bootable CD-ROMs, you should be able to do a completely floppy-less installation. Even if your system doesn't support booting from a CD-ROM, you can use the CD-ROM in conjunction with the other techniques to install your system, once you've booted up by other means; see Chapter 5.

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.tldp.org/HOWTO/CDROM-HOWTO.html) contains in-depth information on using CD-ROMs with Linux.

USB CD-ROM drives are also supported, as are FireWire devices that are supported by the ohci1394 and sbp2 drivers.

2.2.3. Hard Disk

Booting the installation system directly from a hard disk is another option for many architectures. This will require some other operating system to load the installer onto the hard disk.

2.2.4. USB Memory Stick

Many Debian boxes need their floppy and/or CD-ROM drives only for setting up the system and for rescue purposes. If you operate some servers, you will probably already have thought about omitting those drives and using an USB memory stick for installing and (when necessary) for recovering the system. This is also useful for small systems which have no room for unnecessary drives.

2.2.5. Network

You can also *boot* your system over the network.

Diskless installation, using network booting from a local area network and NFS-mounting of all local filesystems, is another option.

After the operating system kernel is installed, you can install the rest of your system via any sort of network connection (including PPP after installation of the base system), via FTP or HTTP.

2.2.6. Un*x or GNU system

If you are running another Unix-like system, you could use it to install Debian GNU/Linux without using the `debian-installer` described in the rest of the manual. This kind of install may be useful for users with otherwise unsupported hardware or on hosts which can't afford downtime. If you are interested in this technique, skip to the Section C.4.

2.2.7. Supported Storage Systems

The Debian boot disks contain a kernel which is built to maximize the number of systems it runs on. Unfortunately, this makes for a larger kernel, which includes many drivers that won't be used for your machine (see Section 8.5 to learn how to build your own kernel). Support for the widest possible range of devices is desirable in general, to ensure that Debian can be installed on the widest array of hardware.

Generally, the Debian installation system includes support for floppies, IDE drives, IDE floppies, parallel port IDE devices, SCSI controllers and drives, USB, and FireWire. The file systems supported include FAT, Win-32 FAT extensions (VFAT), and NTFS, among others.

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.tldp.org/HOWTO/Hardware-HOWTO.html) for more details.

2.3. Peripherals and Other Hardware

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

USB hardware generally works fine, only some USB keyboards may require additional configuration (see Section 3.6.4.4).

Again, see the Linux Hardware Compatibility HOWTO (http://www.tldp.org/HOWTO/Hardware-HOWTO.html) to determine whether your specific hardware is supported by Linux.

2.4. Purchasing Hardware Specifically for GNU/Linux

There are several vendors, who ship systems with Debian or other distributions of GNU/Linux pre-installed (http://www.debian.org/distrib/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. Searching the Internet for "windows refund" may get you some useful information to help with that.

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.

2.4.1. Avoid Proprietary or Closed Hardware

Some hardware manufacturers simply won't tell us how to write drivers for their hardware. Others won't allow us access to the documentation without a non-disclosure agreement that would prevent us from releasing the Linux source code.

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

2.4.2. 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 2000 or XP, 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.tldp.org/HOWTO/Hardware-HOWTO.html).

2.4.3. 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 (http://www.faqs.org/faqs/pc-hardware-faq/part1/).

2.5. Memory and Disk Space Requirements

You must have at least 32MB of memory and 110MB of hard disk space. For a minimal console-

based system (all standard packages), 250MB is required. If you want to install a reasonable amount of software, including the X Window System, and some development programs and libraries, you'll need at least 400MB. For a more or less complete desktop system, you'll need a few gigabytes.

2.6. Network Connectivity Hardware

Most PCI and many older ISA network cards are supported. Some network interface cards are not supported by most Debian installation disks, such as AX.25 cards and protocols; NI16510 EtherBlaster cards; Schneider & Koch G16 cards; 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 (http://www.dgmicro.com/mca/general-goods.html) for some (old) instructions. 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 `debian-installer`.

Chapter 3. Before Installing Debian GNU/Linux

This chapter deals with the preparation for installing Debian before you even boot the installer. This includes backing up your data, gathering information about your hardware, and locating any necessary information.

3.1. Overview of the Installation Process

First, just a note about re-installations. With Debian, a circumstance that will require a complete re-installation of your system is very rare; perhaps mechanical failure of the hard disk would be the most common case.

Many common operating systems may require a complete installation to be performed when critical failures take place or for upgrades to new OS versions. Even if a completely new installation isn't required, often the programs you use must be re-installed to operate properly in the new OS.

Under Debian GNU/Linux, it is much more likely that your OS can be repaired rather than replaced if things go wrong. Upgrades never require a wholesale installation; you can always upgrade in-place. And the programs are almost always compatible with successive OS releases. If a new program version requires newer supporting software, the Debian packaging system ensures that all the necessary software is automatically identified and installed. The point is, much effort has been put into avoiding the need for re-installation, so think of it as your very last option. The installer is *not* designed to re-install over an existing system.

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

1. Back up any existing data or documents on the hard disk where you plan to install.
2. Gather information about your computer and any needed documentation, before starting the installation.
3. Create partition-able space for Debian on your hard disk.
4. Locate and/or download the installer software and any specialized driver files your machine requires (except Debian CD users).
5. Set up boot tapes/floppies/USB sticks, or place boot files (most Debian CD users can boot from one of the CDs).
6. Boot the installation system.
7. Select installation language.
8. Activate the ethernet network connection, if available.
9. Create and mount the partitions on which Debian will be installed.
10. Watch the automatic download/install/setup of the *base system*.
11. Install a *boot loader* which can start up Debian GNU/Linux and/or your existing system.
12. Load the newly installed system for the first time, and make some initial system settings.
13. Install additional software (*tasks* and/or *packages*), at your discretion.

If you have problems during the installation, it helps to know which packages are involved in which steps. Introducing the leading software actors in this installation drama:

The installer software, `debian-installer`, is the primary concern of this manual. It detects hardware and loads appropriate drivers, uses `dhcp-client` to set up the network connection, and runs `debootstrap` to install the base system packages. Many more actors play smaller parts in this process, but `debian-installer` has completed its task when you load the new system for the first time.

Upon loading the new base system, `base-config` supervises adding users, setting a time zone (via `tzsetup`), and setting up the package installation system (using `apt-setup`). It then launches `tasksel` which can be used to select large groups of related programs, and in turn can run `aptitude` which allows you to choose individual software packages.

When `debian-installer` finishes, before the first system load, you have only a very basic command line driven system. The graphical interface which displays windows on your monitor will not be installed unless you select it during the final steps, with either `tasksel` or `aptitude`. It's optional because many Debian GNU/Linux systems are servers which don't really have any need for a graphical user interface to do their job.

Just be aware that the X system is completely separate from `debian-installer`, and in fact is much more complicated. Installation and trouble shooting of the X window installation is not within the scope of this manual.

3.2. Back Up Your Existing Data!

Before you start, make sure to back up every file that is now on your system. If this is the first time a non-native operating system has been installed on your computer, it's quite likely you will need to re-partition your disk to make room for Debian GNU/Linux. Anytime you partition your disk, you should count on losing everything on the disk, no matter what program you use to do it. The programs used in installation are quite reliable and most have seen years of use; but they are also quite powerful and a false move can cost you. Even after backing up be careful and think about your answers and actions. Two minutes of thinking can save hours of unnecessary work.

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

3.3. Information You Will Need

3.3.1. Documentation

3.3.1.1. Installation Manual

This document you are now reading, in plain ASCII, HTML or PDF format.

- install.en.txt
- install.en.html
- install.en.pdf

3.3.1.2. Hardware documentation

Often contains useful information on configuring or using your hardware.

- Linux Hardware Compatibility HOWTO (http://www.tldp.org/HOWTO/Hardware-HOWTO.html)

3.3.2. Finding Sources of Hardware Information

In many cases, the installer will be able to automatically detect your hardware. But to be prepared, we do recommend familiarizing yourself with your hardware before the install.

Hardware information can be gathered from:

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

Table 3-1. Hardware Information Needed for an Install

Hardware	Information You Might Need
Hard Drives	How many you have.
	Their order on the system.
	Whether IDE or SCSI (most computers are IDE).
	Available free space.
	Partitions.
	Partitions where other operating systems are installed.
Monitor	Model and manufacturer.
	Resolutions supported.
	Horizontal refresh rate.
	Vertical refresh rate.
	Color depth (number of colors) supported.
	Screen size.
Mouse	Type: serial, PS/2, or USB.
	Port.

Hardware	Information You Might Need
	Manufacturer.
	Number of buttons.
Network	Model and manufacturer.
	Type of adapter.
Printer	Model and manufacturer.
	Printing resolutions supported.
Video Card	Model and manufacturer.
	Video RAM available.
	Resolutions and color depths supported (these should be checked against your monitor's capabilities).

3.3.3. Hardware Compatibility

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

In particular, Linux usually cannot run hardware that requires a running version of Windows to work.

Although some Windows-specific hardware can be made to run on Linux, doing so usually requires extra effort. In addition, Linux drivers for Windows-specific hardware are usually specific to one Linux kernel. Therefore, they can quickly become obsolete.

So called win-modems are the most common type of this hardware. However, printers and other equipment may also be Windows-specific.

You can check hardware compatibility by:

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

3.3.4. Network Settings

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

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

- Your domain name.
- Your computer's IP address.
- The netmask to use with 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.

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

If you use a wireless network, you should also find out:

- ESSID of your wireless network.
- WEP security key (if applicable).

3.4. Meeting Minimum Hardware Requirements

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

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

A Pentium 100 is the minimum recommended for desktop systems, and a Pentium II-300 for a Server.

Table 3-2. Recommended Minimum System Requirements

Install Type	RAM	Hard Drive
No desktop	24 megabytes	450 megabytes
With Desktop	64 megabytes	1 gigabyte
Server	128 megabytes	4 gigabytes

Here is a sampling of some common Debian system configurations. You can also get an idea of the disk space used by related groups of programs by referring to Section C.3.

Standard Server

This is a small server profile, useful for a stripped down server which does not have a lot of niceties for shell users. It includes an FTP server, a web server, DNS, NIS, and POP. For these 100MB of disk space would suffice, and then you would need to add space for any data you serve up.

Desktop

A standard desktop box, including the X window system, full desktop environments, sound, editors, etc. You'll need about 2GB using the standard desktop task, though it can be done in far less.

Work Console

A more stripped-down user machine, without the X window system or X applications. Possibly suitable for a laptop or mobile computer. The size is around 140MB.

Developer

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

Remember that these sizes don't include all the other materials which are usually to be found, such as user files, mail, and data. It is always best to be generous when considering the space for your own files and data. Notably, the `/var` partition contains a lot of state information specific to Debian in addition to its regular contents like logfiles. The **dpkg** files (with information on all installed packages) can easily consume 20MB. Also, **apt-get** puts downloaded packages here before they are installed. You should usually allocate at least 100MB for `/var`.

3.5. Pre-Partitioning for Multi-Boot Systems

Partitioning your disk simply refers to the act of breaking up your disk into sections. Each section is then independent of the others. It's roughly equivalent to putting up walls inside 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 (Windows 9x, Windows NT/2000/XP, OS/2, MacOS, Solaris, FreeBSD, ...) and want to stick Linux on the same disk, you will need to repartition the disk. Debian requires its own hard disk partitions. It cannot be installed on Windows or MacOS partitions. It may be able to share some partitions with other Linux systems, but that's not covered here. At the very least you will need a dedicated partition for the Debian root.

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

In general, changing a partition with a file system already on it will destroy any information there. Thus you should always make backups before doing any repartitioning. Using the analogy of the house, you would probably want to move all the furniture out of the way before moving a wall or you risk destroying it.

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

If your machine has only one hard disk, and you would like to completely replace the current operating system with Debian GNU/Linux, you also can wait to partition as part of the installation process (Section 6.3.2.1), after you have booted the installation system. However this only works if you plan to boot the installer system from tapes, CD-ROM or files on a connected machine. Consider: if you boot from files placed on the hard disk, and then partition that same hard disk within the installation system, thus erasing the boot files, you'd better hope the installation is successful the first time around. At the least in this case, you should have some alternate means of reviving your machine like the original system's installation tapes or CDs.

If your machine already has multiple partitions, and enough space can be provided by deleting and replacing one or more of them, then you too can wait and use the Debian installer's partitioning program. You should still read through the material below, because there may be special circumstances

like the order of the existing partitions within the partition map, that force you to partition before installing anyway.

If your machine has a FAT or NTFS filesystem, as used by DOS and Windows, you can wait and use Debian installer's partitioning program to resize the filesystem.

If none of the above apply, you'll need to partition your hard disk before starting the installation to create partition-able space for Debian. If some of the partitions will be owned by other operating systems, you should create those partitions using native operating system partitioning programs. We recommend that you do *not* attempt to create partitions for Debian GNU/Linux using another operating system's tools. Instead, you should just create the native operating system's partitions you will want to retain.

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

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

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

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

3.5.1. 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.

But 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, then you must locate your Debian boot partition carefully. 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). This may require that you move an existing FAT or NTFS partition.

3.5.1.1. 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, Me, NT, 2000, XP), or OS/2, and it is desired to put Debian onto the same disk without destroying the previous system. Note that the installer supports resizing of FAT and NTFS filesystems as used by DOS and Windows. Simply start the installer, select the option

to Manually edit partition table, select the partition to resize, and specify its new size. So in most cases you should not need to use the method described below.

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 the Debian portion of the disk as you see fit, i.e., as swap or as a file system.

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 9x, 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 partition managers out there, in case **fips** doesn't do the trick for you.

3.5.1.2. 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
```

3.6. Pre-Installation Hardware and Operating System Setup

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

during the bootstrap process (after power-up). Known hardware issues affecting the reliability of Debian GNU/Linux on your system are also highlighted.

3.6.1. 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 http://www.faqs.org/faqs/pc-hardware-faq/part1/, 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:

AMI BIOS

Delete key during the POST (power on self test)

Award BIOS

Ctrl-**Alt**-**Esc**, or **Delete** key during the POST

DTK BIOS

Esc key during the POST

IBM PS/2 BIOS

Ctrl-**Alt**-**Insert** after **Ctrl**-**Alt**-**Delete**

Phoenix BIOS

Ctrl-**Alt**-**Esc** or **Ctrl**-**Alt**-**S** or **F1**

Information on invoking other BIOS routines can be found in http://www.tldp.org/HOWTO/Hard-Disk-Upgrade/install.html.

Some Intel x86 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.simtel.net/pub/simtelnet/msdos/.

3.6.2. 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 enable booting from a CD-ROM in the SCSI-BIOS of your controller.

Other popular option is to boot from a USB storage (also called USB memory stick or USB key). Some BIOSes can boot USB storage directly, and some cannot. You may need to configure your BIOS to boot from a "Removable drive" or even a "USB-ZIP" to get it to boot from the USB device.

Here are some details about how to set the boot order. Remember to reset the boot order after Linux is installed, so that you restart your machine from the hard drive.

3.6.2.1. Changing the Boot Order on IDE Computers

1. As your computer starts, press the keys to enter the BIOS utility. Often, it is the **Delete** key. However, consult the hardware documentation for the exact keystrokes.
2. Find the boot sequence in the setup utility. Its location depends on your BIOS, but you are looking for a field that lists drives.

   Common entries on IDE machines are C, A, cdrom or A, C, cdrom.

   C is the hard drive, and A is the floppy drive.
3. Change the boot sequence setting so that the CD-ROM or the floppy is first. Usually, the **Page Up** or **Page Down** keys cycle through the possible choices.
4. Save your changes. Instructions on the screen tell you how to save the changes on your computer.

3.6.2.2. Changing the Boot Order on SCSI Computers

1. As your computer starts, press the keys to enter the SCSI setup utility.

   You can start the SCSI setup utility after the memory check and the message about how to start the BIOS utility displays when you start your computer.

   The keystrokes you need depend on the utility. Often, it is **Ctrl-F2**. However, consult your hardware documentation for the exact keystrokes.
2. Find the utility for changing the boot order.
3. Set the utility so that the SCSI ID of the CD drive is first on the list.
4. Save your changes. Instructions on the screen tell you how to save the changes on your computer. Often, you must press **F10**.

3.6.3. Miscellaneous BIOS Settings

3.6.3.1. CD-ROM Settings

Some BIOS systems (such as Award BIOS) allow 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.

3.6.3.2. 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.

3.6.3.3. 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].

3.6.3.4. 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.

3.6.3.5. Memory Hole

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.

3.6.3.6. 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.

3.6.4. Hardware Issues to Watch Out For

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

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

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

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 Section 2.4.3.

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.

3.6.4.1. 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.

3.6.4.2. 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.

3.6.4.3. Peripheral Hardware Settings

You may have to change some settings or jumpers on your computer's peripheral 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.

3.6.4.4. USB BIOS support and keyboards

If you have no AT-style keyboard and only a USB model, you may need to enable legacy AT keyboard emulation in your BIOS setup. Only do this if the installation system fails to use your keyboard in USB mode. Conversely, for some systems (especially laptops) you may need to disable legacy USB support if your keyboard does not respond. Consult your main board manual and look in the BIOS for "Legacy keyboard emulation" or "USB keyboard support" options.

3.6.4.5. More than 64 MB RAM

The Linux Kernel cannot always detect what amount of RAM you have. If this is the case please look at Section 5.2.

Chapter 4. Obtaining System Installation Media

4.1. Official Debian GNU/Linux CD-ROM Sets

By far the easiest way to install Debian GNU/Linux is from an Official Debian CD-ROM Set. You can buy a set from a vendor (see the CD vendors page (http://www.debian.org/CD/vendors/)). You may also download the CD-ROM images from a Debian mirror and make your own set, if you have a fast network connection and a CD burner (see the Debian CD page (http://www.debian.org/CD/) for detailed instructions). If you have a Debian CD set and CDs are bootable on your machine, you can skip right to Chapter 5; much effort has been expended to ensure the files most people need are there on the CD. Although a full set of binary packages requires several CDs, it is unlikely you will need packages on the third CD and above. You may also consider using the DVD version, which saves a lot of space on your shelf and you avoid the CD shuffling marathon.

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

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

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

4.2. Downloading Files from Debian Mirrors

To find the nearest (and thus probably the fastest) mirror, see the list of Debian mirrors (http://www.debian.org/distrib/ftplist).

When downloading files from a Debian mirror, be sure to download the files in *binary* mode, not text or automatic mode.

4.2.1. Where to Find Installation Images

The installation images are located on each Debian mirror in the directory debian/dists/sarge/main/installer-i386/current/images/ (http://http.us.debian.org/debian/dists/sarge/main/installer-i386/current//images)
— the MANIFEST (http://http.us.debian.org/debian/dists/sarge/main/installer-i386/current//images/MANIFEST) lists each image and its purpose.

4.3. Creating Floppies from Disk Images

Bootable floppy disks are generally used as a last resort to boot the installer on hardware that cannot boot from CD or by other means.

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

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

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

4.3.1. Writing Disk Images From a Linux or Unix System

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

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

where *`filename`* is one of the floppy disk image files (see Section 4.2 for what *`filename`* should be). `/dev/fd0` is a commonly used name of the floppy disk device, it may be different on your workstation . 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 .

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.

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

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

The **rawrite1** and **rawrite2** programs can be used under MS-DOS. To use these programs, first make sure that you are booted into DOS. Trying to use these programs from within a DOS box in Windows, or double-clicking on these programs from the Windows Explorer is *not* expected to work.

The **rwwrtwin** program runs on Windows 95, NT, 98, 2000, ME, XP and probably later versions. To use it you will need to unpack diskio.dll in the same directory.

These tools can be found on the Official Debian CD-ROMs under the `/tools` directory.

4.4. Preparing Files for USB Memory Stick Booting

For preparing the USB stick you will need a system where GNU/Linux is already running and where USB is supported. You should ensure that the usb-storage kernel module is loaded (**`modprobe`**

usb-storage) and try to find out which SCSI device the USB stick has been mapped to (in this example `/dev/sda` is used). To write to your stick, you will probably have to turn off its write protection switch.

Note, that the USB stick should be at least 128 MB in size (smaller setups are possible if you follow Section 4.4.2).

4.4.1. Copying the files — the easy way

There is an all-in-one file `hd-media/boot.img.gz` which contains all the installer files (including the kernel) as well as **SYSLINUX** and its configuration file. You only have to extract it directly to your USB stick:

```
# zcat boot.img.gz > /dev/sda
```

Warning

Using this method will destroy anything already on the device. Make sure that you use the correct device name for your USB stick.

After that, mount the USB memory stick (**mount /dev/sda /mnt**), which will now have a FAT filesystem on it, and copy a Debian netinst or businesscard ISO image to it. Please note that the file name must end in `.iso`. Unmount the stick (**umount /mnt**) and you are done.

4.4.2. Copying the files — the flexible way

If you like more flexibility or just want to know what's going on, you should use the following method to put the files on your stick.

4.4.2.1. USB stick partitioning on Intel x86

We will show how to setup the memory stick to use the first partition, instead of the entire device.

> **Note:** Since most USB sticks come pre-configured with a single FAT16 partition, you probably won't have to repartition or reformat the stick. If you have to do that anyway, use **cfdisk** or any other partitioning tool for creating a FAT16 partition and then create the filesystem using:
>
> ```
> # mkdosfs /dev/sda1
> ```
>
> Take care that you use the correct device name for your USB stick. The **mkdosfs** command is contained in the `dosfstools` Debian package.

In order to start the kernel after booting from the USB stick, we will put a boot loader on the stick. Although any boot loader (e.g. **LILO**) should work, it's convenient to use **SYSLINUX**, since it uses a FAT16 partition and can be reconfigured by just editing a text file. Any operating system which supports the FAT file system can be used to make changes to the configuration of the boot loader.

To put **SYSLINUX** on the FAT16 partition on your USB stick, install the `syslinux` and `mtools` packages on your system, and do:

```
# syslinux /dev/sda1
```

Again, take care that you use the correct device name. The partition must not be mounted when starting **SYSLINUX**. This procedure writes a boot sector to the partition and creates the file `ldlinux.sys` which contains the boot loader code.

Mount the partition (**`mount /dev/sda1 /mnt`**) and copy the following files from the Debian archives to the stick:

- `vmlinuz` (kernel binary)
- `initrd.gz` (initial ramdisk image)
- `syslinux.cfg` (SYSLINUX configuration file)
- Optional kernel modules

If you want to rename the files, please note that **SYSLINUX** can only process DOS (8.3) file names.

The `syslinux.cfg` configuration file should contain the following two lines:

```
default vmlinuz
append initrd=initrd.gz ramdisk_size=12000 root=/dev/ram rw
```

Please note that the **`ramdisk_size`** parameter may need to be increased, depending on the image you are booting. If the boot fails, you can try adding **`devfs=mount,dall`** to the "append" line.

4.4.2.2. Adding an ISO image

Now you should put any Debian ISO image (businesscard, netinst or even a full one) onto your stick (if it fits). The file name of such an image must end in `.iso`.

If you want to install over the network, without using an ISO image, you will of course skip the previous step. Moreover you will have to use the initial ramdisk from the `netboot` directory instead of the one from `hd-media`, because `hd-media/initrd.gz` does not have network support.

When you are done, unmount the USB memory stick (**`umount /mnt`**) and activate its write protection switch.

4.4.2.3. Booting the USB stick

Warning

If your system refuses to boot from the memory stick, the stick may contain an invalid master boot record (MBR). To fix this, use the **install-mbr** command from the package `mbr`:

```
# install-mbr /dev/sda
```

4.5. Preparing Files for Hard Disk Booting

The installer may be booted using boot files placed on an existing hard drive partition, either launched from another operating system or by invoking a boot loader directly from the BIOS.

A full, "pure network" installation can be achieved using this technique. This avoids all hassles of removable media, like finding and burning CD images or struggling with too numerous and unreliable floppy disks.

The installer cannot boot from files on an NTFS file system.

4.5.1. Hard disk installer booting using LILO or GRUB

This section explains how to add to or even replace an existing linux installation using either **LILO** or **GRUB**.

At boot time, both bootloaders support loading in memory not only the kernel, but also a disk image. This RAM disk can be used as the root file-system by the kernel.

Copy the following files from the Debian archives to a convenient location on your hard drive, for instance to `/boot/newinstall/`.

- `vmlinuz` (kernel binary)
- `initrd.gz` (ramdisk image)

Finally, to configure the bootloader proceed to Section 5.1.2.

4.6. Preparing Files for TFTP Net Booting

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

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

BOOTP is an IP protocol that informs a computer of its IP address and where on the network to obtain a boot image. The DHCP (Dynamic Host Configuration Protocol) is a more flexible, backwards-compatible extension of BOOTP. Some systems can only be configured via DHCP.

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

> **Note:** To use the Pre-boot Execution Environment (PXE) method of TFTP booting, you will need a TFTP server with **`tsize`** support. On a Debian GNU/Linux server, the `atftpd` and `tftpd-hpa` packages qualify; we recommend `tftpd-hpa`.

4.6.1. Setting up BOOTP server

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

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

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

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

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

You will need to change at least the "ha" option, which specifies the hardware address of the client. The "bf" option specifies the file a client should retrieve via TFTP; see Section 4.6.4 for more details.

By contrast, setting up BOOTP with ISC **dhcpd** is really easy, because it treats BOOTP clients as a moderately special case of DHCP clients. Some architectures require a complex configuration for booting clients via BOOTP. If yours is one of those, read the section Section 4.6.2. Otherwise, you will probably be able to get away with simply adding the **`allow bootp`** directive to the configuration block for the subnet containing the client, and restart **dhcpd** with **`/etc/init.d/dhcpd restart`**.

4.6.2. Setting up a DHCP server

One free software DHCP server is ISC **dhcpd**. In Debian GNU/Linux, this is available in the `dhcp` package. Here is a sample configuration file for it (usually `/etc/dhcpd.conf`):

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

subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.200 192.168.1.253;
  option routers 192.168.1.1;
}

host clientname {
  filename "/tftpboot/tftpboot.img";
  server-name "servername";
```

```
  next-server servername;
  hardware ethernet 01:23:45:67:89:AB;
  fixed-address 192.168.1.90;
}
```

Note: the new (and preferred) `dhcp3` package uses `/etc/dhcp3/dhcpd.conf`.

In this example, there is one server *`servername`* which performs all of the work of DHCP server, TFTP server, and network gateway. You will almost certainly need to change the domain-name options, as well as the server name and client hardware address. The *`filename`* option should be the name of the file which will be retrieved via TFTP.

After you have edited the **dhcpd** configuration file, restart it with **`/etc/init.d/dhcpd restart`**.

4.6.2.1. Enabling PXE Booting in the DHCP configuration

Here is another example for a `dhcp.conf` using the Pre-boot Execution Environment (PXE) method of TFTP.

```
option domain-name "example.com";

default-lease-time 600;
max-lease-time 7200;

allow booting;
allow bootp;

# The next paragraph needs to be modified to fit your case
subnet 192.168.1.0 netmask 255.255.255.0 {
  range 192.168.1.200 192.168.1.253;
  option broadcast-address 192.168.1.255;
# the gateway address which can be different
# (access to the internet for instance)
  option routers 192.168.1.1;
# indicate the dns you want to use
  option domain-name-servers 192.168.1.3;
}

group {
 next-server 192.168.1.3;
 host tftpclient {
# tftp client hardware address
  hardware ethernet  00:10:DC:27:6C:15;
  filename "/tftpboot/pxelinux.0";
 }
}
```

Note that for PXE booting, the client filename `pxelinux.0` is a boot loader, not a kernel image (see Section 4.6.4 below).

4.6.3. Enabling the TFTP Server

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

```
tftp dgram udp wait nobody /usr/sbin/tcpd in.tftpd /tftpboot
```

Debian packages will in general set this up correctly by default when they are installed.

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

4.6.4. Move TFTP Images Into Place

Next, place the TFTP boot image you need, as found in Section 4.2.1, in the **tftpd** boot image directory. Generally, this directory will be `/tftpboot`. You'll have to make a link from that file to the file which **tftpd** will use for booting a particular client. Unfortunately, the file name is determined by the TFTP client, and there are no strong standards.

For PXE booting, everything you should need is set up in the `netboot/netboot.tar.gz` tarball. Simply extract this tarball into the **tftpd** boot image directory. Make sure your dhcp server is configured to pass `/pxelinux.0` to **tftpd** as the filename to boot.

4.7. Automatic Installation

For installing on multiple computers it's possible to do fully automatic installations. Debian packages intended for this include `fai` (which uses an install server), `replicator`, `systemimager`, `autoinstall`, and the Debian Installer itself.

4.7.1. Automatic Installation Using the Debian Installer

The Debian Installer supports automating installs via preconfiguration files. A preconfiguration file can be loaded from the network or from removable media, and used to fill in answers to questions asked during the installation process.

Although most dialogs used by `debian-installer` can be preseeded using this method, there are some notable exceptions. You can (re)partition an entire disk or use available free space on a disk; it is not possible to use existing partitions. You currently cannot use preseeding to set up RAID and LVM. Also, with the exception of network driver modules, it is not possible to preconfigure kernel module parameters.

The preconfiguration file is in the format used by the debconf-set-selections command. A well documented and working example that you can edit is in Section C.1.

Alternatively, one way to get a complete file listing all the values that can be preseeded is to do a manual install, and then use `debconf-get-selections`, from the `debconf-utils` package, to dump both the debconf database and the cdebconf database in /var/log/debian-installer/cdebconf to a single file:

```
$ debconf-get-selections --installer > file
$ debconf-get-selections >> file
```

However, a file generated in this manner will have some items that should not be preseeded, and the file in Section C.1 is a better starting place for most users.

Once you have a preconfiguration file, you can edit it if necessary, and place it on a web server, or copy it onto the installer's boot media. Wherever you place the file, you need to pass a parameter to the installer at boot time to tell it to use the file.

To make the installer use a preconfiguration file downloaded from the network, add preseed/url=http://url/to/preseed.cfg to the kernel boot parameters. Of course the preconfiguration will not take effect until the installer manages to set up the network to download the file, so this is most useful if the installer can set up the network via DHCP without asking any questions. You may want to set the installation priority to critical to avoid any questions while the network is being configured. See Section 5.2.1.

To place a preconfiguration file on a CD, you would need to remaster the ISO image to include your preconfiguration file. See the manual page for mkisofs for details. Alternatively, put the preseed file on a floppy, and use preseed/file=/floppy/preseed.cfg

If you'll be booting from a USB memory stick, then you can simply copy your preconfiguration file onto the memory stick's filesystem, and edit the syslinux.cfg file to add preseed/file=/hd-media/preseed.cfg to the kernel boot parameters.

Chapter 5. Booting the Installation System

5.1. Booting the Installer on Intel x86

5.1.1. Booting from a CD-ROM

The easiest route for most people will be to use a set of Debian CDs. If you have a CD set, and if your machine supports booting directly off the CD, great! Simply configure your system for booting off a CD as described in Section 3.6.2, insert your CD, reboot, and proceed to the next chapter.

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

Even if you cannot boot from CD-ROM, you can probably install the Debian system components and any packages you want from CD-ROM. Simply boot using a different media, such as floppies. When it's time to install the operating system, base system, and any additional packages, point the installation system at the CD-ROM drive.

If you have problems booting, see Section 5.3.

5.1.2. Booting from Linux Using LILO or GRUB

To boot the installer from hard disk, you must first download and place the needed files as described in Section 4.5.

If you intend to use the hard drive only for booting and then download everything over the network, you should download the `netboot/debian-installer/i386/initrd.gz` file and its corresponding kernel. This will allow you to repartition the hard disk from which you boot the installer, although you should do so with care.

Alternatively, if you intend to keep an existing partition on the hard drive unchanged during the install, you can download the `hd-media/initrd.gz` file and its kernel, as well as copy a CD iso to the drive (make sure the file is named ending in `.iso`). The installer can then boot from the drive and install from the CD image, without needing the network.

For **LILO**, you will need to configure two essential things in `/etc/lilo.conf`:

- to load the `initrd.gz` installer at boot time;
- have the `vmlinuz` kernel use a RAM disk as its root partition.

Here is a `/etc/lilo.conf` example:

```
image=/boot/newinstall/vmlinuz
       label=newinstall
       initrd=/boot/newinstall/initrd.gz
       root=/dev/ram0
       append="devfs=mount,dall ramdisk_size=12000"
```

For more details, refer to the initrd(4) and lilo.conf(5) man pages. Now run **`lilo`** and reboot.

The procedure for **GRUB** is quite similar. Locate your `menu.lst` in the `/boot/grub/` directory (sometimes in the `/boot/boot/grub/`), add the following lines:

```
title  New Install
kernel (hd0,0)/boot/newinstall/vmlinuz root=/dev/ram0 ramdisk_size=12000
initrd (hd0,0)/boot/newinstall/initrd.gz
```

and reboot. If the boot fails, you can try adding **`devfs=mount,dall`** to the "kernel" line.

Note that the value of the **`ramdisk_size`** may need to be adjusted for the size of the initrd image. From here on, there should be no difference between **GRUB** or **LILO**.

5.1.3. Booting from USB Memory Stick

Let's assume you have prepared everything from Section 3.6.2 and Section 4.4. Now just plug your USB stick into some free USB connector and reboot the computer. The system should boot up, and you should be presented with the `boot:` prompt. Here you can enter optional boot arguments, or just hit **Enter**.

In case your computer doesn't support booting from USB memory devices, you can still use a single floppy to do the initial boot and then switch to USB. Boot your system as described in Section 5.1.4; the kernel on the boot floppy should detect your USB stick automatically. When it asks for the root floppy, simply press **Enter**. You should see `debian-installer` starting.

5.1.4. Booting from Floppies

You will have already downloaded the floppy images you needed and created floppies from the images in Section 4.3.

To boot from the installer boot floppy, place it in the primary floppy drive, shut down the system as you normally would, then turn it back on.

For installing from an LS-120 drive (ATAPI version) with a set of floppies, you need to specify the virtual location for the floppy device. This is done with the *root*= boot argument, giving the device that the ide-floppy driver maps the device to. For example, if your LS-120 drive is connected as the first IDE device (master) on the second cable, you enter **`linux root=/dev/hdc`** at the boot prompt. Installation from LS-120 is only supported by 2.4 and later kernels.

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 reboot when booting.

The floppy disk will be accessed, and you should then see a screen that introduces the boot floppy and ends with the `boot:` prompt.

Once you press **Enter**, you should see the message `Loading...`, followed by `Uncompressing Linux...`, and then a screenfull or so of information about the hardware in your system. More information on this phase of the boot process can be found below in Section 5.3.4.

After booting from the boot floppy, the root floppy is requested. Insert the root floppy and press **Enter**, and the contents are loaded into memory. The installer program **debian-installer** is automatically launched.

5.1.5. Booting with TFTP

Booting from the network requires that you have a network connection and a TFTP network boot server (DHCP, RARP, or BOOTP).

The installation method to support network booting is described in Section 4.6.

There are various ways to do a TFTP boot on i386.

5.1.5.1. NIC or Motherboard that support PXE

It could be that your Network Interface Card or Motherboard provides PXE boot functionality. This is a Intel™ re-implementation of TFTP boot. If so you may be able to configure your BIOS to boot from the network.

5.1.5.2. NIC with Network BootROM

It could be that your Network Interface Card provides TFTP boot functionality.

5.1.5.3. Etherboot

The etherboot project (http://www.etherboot.org) provides bootdiskettes and even bootroms that do a TFTPboot.

5.1.6. The Boot Prompt

When the installer boots, you should be presented with a friendly graphical screen showing the Debian logo and the boot prompt:

```
Press F1 for help, or ENTER to boot:
```

At the boot prompt you can either just press **Enter** to boot the installer with default options or enter a specific boot method and, optionally, boot parameters.

Information on available boot methods and on boot parameters which might be useful can be found by pressing **F2** through **F7**. 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 debconf/priority=medium`**).

> **Note:** If you are installing the system via a remote management device that provides a text interface to the VGA console, you may not be able to see the initial graphical splash screen upon booting the installer; you may even not see the boot prompt. Examples of these devices include the text console of Compaq's "integrated Lights Out" (iLO) and HP's "Integrated Remote Assistant" (IRA). You can blindly press F1[1] to bypass this screen and view the help text. Once you are past the splash screen and at the help text your keystrokes will be echoed at the prompt as expected. To prevent the installer from using the framebuffer for the rest of the installation, you will also want to add **`debian-installer/framebuffer=false`** to the boot prompt, as described in the help text.

1. In some cases these devices will require special escape sequences to enact this keypress, for example the IRA uses **Ctrl-F**, **1**.

5.2. Boot Parameters

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 this is the first time you're booting the system, try the default boot parameters (i.e., don't try setting parameters) 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.

Information on many boot parameters can be found in the Linux BootPrompt HOWTO (http://www.tldp.org/HOWTO/BootPrompt-HOWTO.html), including tips for obscure hardware. This section contains only a sketch of the most salient parameters. Some common gotchas are included below in Section 5.3.

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 amount of RAM you have installed, you need to use the **mem=*ram*** parameter, where *ram* is set to the amount of memory, suffixed with "k" for kilobytes, or "m" for megabytes. For example, both **mem=65536k** and **mem=64m** mean 64MB of RAM.

If 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`.

5.2.1. Debian Installer Parameters

The installation system recognizes a few additional boot parameters[2] which may be useful.

debconf/priority

This parameter sets the lowest priority of messages to be displayed.

The default installation uses **debconf/priority=high**. This means that both high and critical priority messages are shown, but medium and low priority messages are skipped. If problems are encountered, the installer adjusts the priority as needed.

If you add **debconf/priority=medium** as boot parameter, you will be shown the installation menu and gain more control over the installation. When **debconf/priority=low** is used, all messages are shown (this is equivalent to the *expert* boot method). With **debconf/priority=critical**, the installation system will display only critical messages and try to do the right thing without fuss.

2. Note that the kernel accepts a maximum of 8 command line options and 8 environment options (including any options added by default for the installer). If these numbers are exceeded, 2.4 kernels will drop any excess options and 2.6 kernels will panic.

DEBIAN_FRONTEND

This boot parameter controls the type of user interface used for the installer. The current possible parameter settings are:

- **DEBIAN_FRONTEND=noninteractive**
- **DEBIAN_FRONTEND=text**
- **DEBIAN_FRONTEND=newt**
- **DEBIAN_FRONTEND=slang**
- **DEBIAN_FRONTEND=ncurses**
- **DEBIAN_FRONTEND=bogl**
- **DEBIAN_FRONTEND=gtk**
- **DEBIAN_FRONTEND=corba**

The default front end is **DEBIAN_FRONTEND=newt**. **DEBIAN_FRONTEND=text** may be preferable for serial console installs. Generally only the **newt** frontend is available on default install media, so this is not very useful right now.

BOOT_DEBUG

Setting this boot parameter to 2 will cause the installer's boot process to be verbosely logged. Setting it to 3 makes debug shells available at strategic points in the boot process. (Exit the shells to continue the boot process.)

BOOT_DEBUG=0

This is the default.

BOOT_DEBUG=1

More verbose than usual.

BOOT_DEBUG=2

Lots of debugging information.

BOOT_DEBUG=3

Shells are run at various points in the boot process to allow detailed debugging. Exit the shell to continue the boot.

INSTALL_MEDIA_DEV

The value of the parameter is the path to the device to load the Debian installer from. For example, **INSTALL_MEDIA_DEV=/dev/floppy/0**

The boot floppy, which normally scans all floppies and USB storage devices it can to find the root floppy, can be overridden by this parameter to only look at the one device.

debian-installer/framebuffer

Some architectures use the kernel framebuffer to offer installation in a number of languages. If framebuffer causes a problem on your system you can disable the feature by the parameter **debian-installer/framebuffer=false**. Problem symptoms are error messages about bterm or bogl, a blank screen, or a freeze within a few minutes after starting the install.

The **video=vga16:off** argument may also be used to disable the framebuffer. Such problems have been reported on a Dell Inspiron with Mobile Radeon card.

debian-installer/probe/usb

Set to **false** to prevent probing for USB on boot, if that causes problems.

netcfg/disable_dhcp

By default, the `debian-installer` automatically probes for network configuration via DHCP. If the probe succeeds, you won't have a chance to review and change the obtained settings. You can get to the manual network setup only in case the DHCP probe fails.

If you have a DHCP server on your local network, but want to avoid it because e.g. it gives wrong answers, you can use the parameter **netcfg/disable_dhcp=true** to prevent configuring the network with DHCP and to enter the information manually.

hw-detect/start_pcmcia

Set to **false** to prevent starting PCMCIA services, if that causes problems. Some laptops are well known for this misbehavior.

preseed/url

Specify the url to a preconfiguration file to download and use in automating the install. See Section 4.7.

preseed/file

Specify the path to a preconfiguration file to load to automating the install. See Section 4.7.

ramdisk_size

If you are using a 2.2.x kernel, you may need to set **ramdisk_size=13000** .

5.3. Troubleshooting the Installation Process

5.3.1. Floppy Disk Reliability

The biggest problem for people using floppy disks to install Debian seems to be floppy disk reliability.

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

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

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

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

5.3.2. Boot Configuration

If you have problems and the kernel hangs during the boot process, doesn't recognize peripherals you actually have, or drives are not recognized properly, the first thing to check is the boot parameters, as discussed in Section 5.2.

If you are booting with your own kernel instead of the one supplied with the installer, be sure that **CONFIG_DEVFS** is set in your kernel. The installer requires **CONFIG_DEVFS**.

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

5.3.3. Common Intel x86 Installation Problems

There are some common installation problems that can be solved or avoided by passing certain boot parameters to the installer.

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 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 your screen begins to show a weird picture while the kernel boots, eg. pure white, pure black or colored pixel garbage, your system may contain a problematic video card which does not switch to the framebuffer mode properly. Then you can use the boot parameter **debian-installer/framebuffer=false** or **video=vga16:off** to disable the framebuffer console. Only the English language will be available during the installation due to limited console features. See Section 5.2 for details.

5.3.3.1. System Freeze During the PCMCIA Configuration Phase

Some laptop models produced by Dell are known to crash when PCMCIA device detection tries to access some hardware addresses. Other laptops may display similar problems. If you experience such a problem and you don't need PCMCIA support during the installation, you can disable PCMCIA using the **hw-detect/start_pcmcia=false** boot parameter. You can then configure PCMCIA after the installation is completed and exclude the resource range causing the problems.

Alternatively, you can boot the installer in expert mode. You will then be asked to enter the resource range options your hardware needs. For example, if you have one of the Dell laptops mentioned above, you should enter **exclude port 0x800-0x8ff** here. There is also a list of some common resource range options in the System resource settings section of the PCMCIA HOWTO (http://pcmcia-cs.sourceforge.net/ftp/doc/PCMCIA-HOWTO-1.html#ss1.12). Note that you have to omit the commas, if any, when you enter this value in the installer.

5.3.3.2. System Freeze while Loading the USB Modules

The kernel normally tries to install USB modules and the USB keyboard driver in order to support some non-standard USB keyboards. However, there are some broken USB systems where the driver hangs on loading. A possible workaround may be disabling the USB controller in your mainboard BIOS setup. Another option is passing the **debian-installer/probe/usb=false** parameter at the boot prompt, which will prevent the modules from being loaded.

5.3.4. 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 Section 8.5).

5.3.5. Bug Reporter

If you get through the initial boot phase but cannot complete the install, the bug reporter menu choice may be helpful. It copies system error logs and configuration information to a user-supplied floppy. This information may provide clues as to what went wrong and how to fix it. If you are submitting a bug report you may want to attach this information to the bug report.

Other pertinent installation messages may be found in `/var/log/` during the installation, and `/var/log/debian-installer/` after the computer has been booted into the installed system.

5.3.6. Submitting Installation Reports

If you still have problems, please submit an installation report. We also encourage installation reports to be sent even if the installation is successful, so that we can get as much information as possible on the largest number of hardware configurations.

Please use this template when filling out installation reports, and file the report as a bug report against the `installation-reports` pseudo package, by sending it to <`submit@bugs.debian.org`>.

```
Package: installation-reports

Boot method: <How did you boot the installer? CD? floppy? network?>
Image version: <Fill in date and from where you got the image>
Date: <Date and time of the install>

Machine: <Description of machine (eg, IBM Thinkpad R32)>
Processor:
Memory:
Partitions: <df -Tl will do; the raw partition table is preferred>

Output of lspci and lspci -n:
```

```
Base System Installation Checklist:
[O] = OK, [E] = Error (please elaborate below), [ ] = didn't try it

Initial boot worked:    [ ]
Configure network HW:   [ ]
Config network:         [ ]
Detect CD:              [ ]
Load installer modules: [ ]
Detect hard drives:     [ ]
Partition hard drives:  [ ]
Create file systems:    [ ]
Mount partitions:       [ ]
Install base system:    [ ]
Install boot loader:    [ ]
Reboot:                 [ ]

Comments/Problems:

<Description of the install, in prose, and any thoughts, comments
      and ideas you had during the initial install.>
```

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 6. Using the Debian Installer

6.1. How the Installer Works

The Debian Installer consists of a number of special-purpose components to perform each installation task. Each component performs its task, asking the user questions as necessary to do its job. The questions themselves are given priorities, and the priority of questions to be asked is set when the installer is started.

When a default installation is performed, only essential (high priority) questions will be asked. This results in a highly automated installation process with little user interaction. Components are automatically run in sequence; which components are run depends mainly on the installation method you use and on your hardware. The installer will use default values for questions that are not asked.

If there is a problem, the user will see an error screen, and the installer menu may be shown in order to select some alternative action. If there are no problems, the user will never see the installer menu, but will simply answer questions for each component in turn. Serious error notifications are set to priority "critical" so the user will always be notified.

Some of the defaults that the installer uses can be influenced by passing boot arguments when `debian-installer` is started. If, for example, you wish to force static network configuration (DHCP is used by default if available), you could add the boot parameter **`netcfg/disable_dhcp=true`**. See Section 5.2.1 for available options.

Power users may be more comfortable with a menu-driven interface, where each step is controlled by the user rather than the installer performing each step automatically in sequence. To use the installer in a manual, menu-driven way, add the boot argument **`debconf/priority=medium`**.

If your hardware requires you to pass options to kernel modules as they are installed, you will need to start the installer in "expert" mode. This can be done by either using the **expert** command to start the installer or by adding the boot argument **`debconf/priority=low`**. Expert mode gives you full control over `debian-installer`.

The normal installer display is character-based (as opposed to the now more familiar graphical interface). The mouse is not operational in this environment. Here are the keys you can use to navigate within the various dialogs. The **Tab** or **right** arrow keys move "forward", and the **Shift**-**Tab** or **left** arrow keys move "backward" between displayed buttons and selections. The **up** and **down** arrow select different items within a scrollable list, and also scroll the list itself. In addition, in long lists, you can type a letter to cause the list to scroll directly to the section with items starting with the letter you typed and use **Pg-Up** and **Pg-Down** to scroll the list in sections. The **space bar** selects an item such as a checkbox. Use **Enter** to activate choices.

Error messages are redirected to the third console. You can access this console by pressing **Left Alt**-**F3** (hold the left **Alt** key while pressing the **F3** function key); get back to the main installer process with **Left Alt**-**F1**.

These messages can also be found in `/var/log/messages`. After installation, this log is copied to `/var/log/debian-installer/messages` on your new system. Other installation messages may be found in `/var/log/` during the installation, and `/var/log/debian-installer/` after the computer has been booted into the installed system.

6.2. Components Introduction

Here is a list of installer components with a brief description of each component's purpose. Details you might need to know about using a particular component are in Section 6.3.

main-menu

Shows the list of components to the user during installer operation, and starts a component when it is selected. Main-menu's questions are set to priority medium, so if your priority is set to high or critical (high is the default), you will not see the menu. On the other hand, if there is an error which requires your intervention, the question priority may be downgraded temporarily to allow you to resolve the problem, and in that case the menu may appear.

You can get to the main menu by selecting the "Back" button repeatedly to back all the way out of the currently running component.

languagechooser

Shows a list of languages and language variants. The installer will display messages in the chosen language, unless the translation for that language is not complete. When a translation is not complete, English messages are shown.

countrychooser

Shows a list of countries. The user may choose the country he lives in.

kbd-chooser

Shows a list of keyboards, from which the user chooses the model which matches his own.

hw-detect

Automatically detects most of the system's hardware, including network cards, disk drives, and PCMCIA.

cdrom-detect

Looks for and mounts a Debian installation CD.

netcfg

Configures the computer's network connections so it can communicate over the internet.

iso-scan

Looks for ISO file systems, which may be on a CD-ROM or on the hard drive.

choose-mirror

Presents a list of Debian archive mirrors. The user may choose the source of his installation packages.

cdrom-checker

Checks integrity of a CD-ROM. This way the user may assure him/herself that the installation CD-ROM was not corrupted.

lowmem

Lowmem tries to detect systems with low memory and then does various tricks to remove unnecessary parts of `debian-installer` from the memory (at the cost of some features).

anna

Anna's Not Nearly APT. Installs packages which have been retrieved from the chosen mirror or CD.

partman

Allows the user to partition disks attached to the system, create file systems on the selected partitions, and attach them to the mountpoints. Included are also interesting features like a fully automatic mode or LVM support. This is the preferred partitioning tool in Debian.

autopartkit

Automatically partitions an entire disk according to preset user preferences.

partitioner

Allows the user to partition disks attached to the system. A partitioning program appropriate to your computer's architecture is chosen.

partconf

Displays a list of partitions, and creates file systems on the selected partitions according to user instructions.

lvmcfg

Helps the user with the configuration of the *LVM* (Logical Volume Manager).

mdcfg

Allows the user to setup Software *RAID* (Redundant Array of Inexpensive Disks). This Software RAID is usually superior to the cheap IDE (pseudo hardware) RAID controllers found on newer motherboards.

base-installer

Installs the most basic set of packages which would allow the computer to operate under Linux when rebooted.

os-prober

Detects currently installed operating systems on the computer and passes this information to the bootloader-installer, which may offer you an ability to add discovered operating systems to the bootloader's start menu. This way the user could easily choose at the boot time which operating system to start.

bootloader-installer

Installs a boot loader program on the hard disk, which is necessary for the computer to start up using Linux without using a floppy or CD-ROM. Many boot loaders allow the user to choose an alternate operating system each time the computer boots.

base-config

Provides dialogs for setting up the base system packages according to user preferences. This is normally done after rebooting the computer; it is the "first run" of the new Debian system.

shell

Allows the user to execute a shell from the menu, or in the second console.

bugreporter

Provides a way for the user to record information on a floppy disk when trouble is encountered, in order to accurately report installer software problems to Debian developers later.

6.3. Using Individual Components

In this section we will describe each installer component in detail. The components have been grouped into stages that should be recognizable for users. They are presented in the order they appear during the install. Note that not all modules will be used for every installation; which modules are actually used depends on the installation method you use and on your hardware.

6.3.1. Setting up Debian Installer and Hardware Configuration

Let's assume the Debian Installer has booted and you are facing its first screen. At this time, the capabilities of `debian-installer` are still quite limited. It doesn't know much about your hardware, preferred language, or even the task it should perform. Don't worry. Because `debian-installer` is quite clever, it can automatically probe your hardware, locate the rest of its components and upgrade itself to a capable installation system. However, you still need to help `debian-installer` with some information it can't determine automatically (like selecting your preferred language, keyboard layout or desired network mirror).

You will notice that `debian-installer` performs *hardware detection* several times during this stage. The first time is targeted specifically at the hardware needed to load installer components (e.g. your CD-ROM or network card). As not all drivers may be available during this first run, hardware detection needs to be repeated later in the process.

6.3.1.1. Check available memory

One of the first things `debian-installer` does, is to check available memory. If the available memory is limited, this component will make some changes in the installation process which hopefully will allow you to install Debian GNU/Linux on your system.

During a low memory install, not all components will be available. One of the limitations is that you won't be able to choose a language for the installation.

6.3.1.2. Language selection

As the first step of the installation, select the language in which you want the installation process to proceed. The language names are listed in both English (left side) and in the language itself (right side); the names on the right side are also shown in the proper script for the language. The list is sorted on the English names.

The language you choose will be used for the rest of the installation process, provided a translation of the different dialogs is available. If no valid translation is available for the selected language, the installer will default to English. The selected language will also be used to help select a suitable keyboard layout.

6.3.1.3. Country selection

If you selected a language in Section 6.3.1.2 which has more than one country associated with it (true for Chinese, English, French, and many other languages), you can specify the country here. If you choose Other at the bottom of the list, you will be presented with a list of all countries, grouped by continent.

This selection will be used later in the installation process to pick the default timezone and a Debian mirror appropriate for your geographic location. If the defaults proposed by the installer are not suitable, you can make a different choice. The selected country, together with the selected language, may also affect locale settings for your new Debian system.

6.3.1.4. Choosing a Keyboard

Keyboards are often tailored to the characters used in a language. Select a layout that conforms to the keyboard you are using, 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 after you have completed the installation).

Move the highlight to the keyboard selection you desire and press **Enter**. Use the arrow keys to move the highlight — they are in the same place in all national language keyboard layouts, so they are independent of the keyboard configuration. An 'extended' keyboard is one with **F1** through **F10** keys along the top row.

6.3.1.5. Looking for the Debian Installer ISO Image

When installing via the *hd-media* method, there will be a moment where you need to find and mount the Debian Installer iso image in order to get the rest of the installation files. The component **iso-scan** does exactly this.

At first, **iso-scan** automatically mounts all block devices (e.g. partitions) which have some known filesystem on them and sequentially searches for filenames ending with `.iso` (or `.ISO` for that matter). Beware that the first attempt scans only files in the root directory and in the first level of subdirectories (i.e. it finds `/`*`whatever`*`.iso`, `/data/`*`whatever`*`.iso`, but not `/data/tmp/`*`whatever`*`.iso`). After an iso image has been found, **iso-scan** checks its content to determine if the image is a valid Debian iso image or not. In the former case we are done, in the latter **iso-scan** seeks for another image.

In case the previous attempt to find an installer iso image fails, **iso-scan** will ask you whether you would like to perform a more thorough search. This pass doesn't just look into the topmost directories, but really traverses whole filesystem.

If **iso-scan** does not discover your installer iso image, reboot back to your original operating system and check if the image is named correctly (ending in `.iso`), if it is placed on a filesystem recognizable by `debian-installer`, and if it is not corrupted (verify the checksum). Experienced Unix users could do this without rebooting on the second console.

6.3.1.6. Configuring Network

As you enter this step, if the system detects that you have more than one network device, you'll be asked to choose which device will be your *primary* network interface, i.e. the one which you want to use for installation. The other interfaces won't be configured at this time. You may configure additional interfaces after installation is complete; see the interfaces(5) man page.

By default, debian-installer tries to configure your computer's network automatically via DHCP. If the DHCP probe succeeds, you are done. If the probe fails, it may be caused by many factors ranging from unplugged network cable, to a misconfigured DHCP setup. Or maybe you don't have a DHCP server in your local network at all. For further explanation check the error messages on the third console. In any case, you will be asked if you want to retry, or if you want to perform manual setup. DHCP servers are sometimes really slow in their responses, so if you are sure everything is in place, try again.

The manual network setup in turn asks you a number of questions about your network, notably IP address, Netmask, Gateway, Name server addresses, and a Hostname. Moreover, if you have a wireless network interface, you will be asked to provide your Wireless ESSID and a WEP key. Fill in the answers from Section 3.3.

> **Note:** 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 also guess your gateway. If you can't find any of these answers, use the system's guesses — you can change them once the system has been installed, if necessary, by editing /etc/network/interfaces. Alternatively, you can install etherconf, which will step you through your network setup.

6.3.2. Partitioning and Mount Point Selection

At this time, after hardware detection has been executed a final time, debian-installer should be at its full strength, customized for the user's needs and ready to do some real work. As the title of this section indicates, the main task of the next few components lies in partitioning your disks, creating filesystems, assigning mountpoints and optionally configuring closely related issues like LVM or RAID devices.

6.3.2.1. Partitioning Your Disks

Now it is time to partition your disks. If you are uncomfortable with partitioning, or just want to know more details, see Appendix B.

First you will be given the opportunity to automatically partition either an entire drive, or free space on a drive. This is also called "guided" partitioning. If you do not want to autopartition, choose Manually edit partition table from the menu.

If you choose guided partitioning, you will be able to choose from the schemes listed in the table below. All schemes have their pros and cons, some of which are discussed in Appendix B. If you are unsure, choose the first one. Bear in mind, that guided partitioning needs certain minimal amount of free space to operate with. If you don't give it at least about 1GB of space (depends on chosen scheme), guided partitioning will fail.

Partitioning scheme	Minimum space	Created partitions
All files in one partition	600MB	/, swap
Desktop machine	500MB	/, /home, swap
Multi-user workstation	1GB	/, /home, /usr, /var, /tmp, swap

After selecting a scheme, the next screen will show your new partition table, including information on whether and how partitions will be formatted and where they will be mounted.

The list of partitions might look like this:

```
IDE1 master (hda) - 6.4 GB WDC AC36400L
      #1 primary   16.4 MB     ext2       /boot
      #2 primary  551.0 MB     swap       swap
      #3 primary    5.8 GB     ntfs
         pri/log    8.2 MB     FREE SPACE

IDE1 slave (hdb) - 80.0 GB ST380021A
      #1 primary   15.9 MB     ext3
      #2 primary  996.0 MB     fat16
      #3 primary    3.9 GB     xfs        /home
      #5 logical    6.0 GB     ext3       /
      #6 logical    1.0 GB     ext3       /var
      #7 logical  498.8 MB     ext3
      #8 logical  551.5 MB     swap       swap
      #9 logical   65.8 GB     ext2
```

This example shows two IDE harddrives divided into several partitions; the first disk has some free space. Each partition line consists of the partition number, its type, size, optional flags, file system, and mountpoint (if any).

This concludes the guided partitioning. If you are satisfied with the generated partition table, you can choose Finish partitioning and write changes to disk from the menu to implement the new partition table (as described at the end of this section). If you are not happy, you can choose to Undo changes to partitions, to run guided partitioning again or modify the proposed changes as described below for manual partitioning.

A similar screen to the one shown just above will be displayed if you choose manual partitioning except that your existing partition table will be shown and without the mount points. How to manually setup your partition table and the usage of partitions by your new Debian system will be covered in the remainder of this section.

If you select a pristine disk which doesn't have neither partitions nor free space on it, you will be offered to create a new partition table (this is needed so you can create new partitions). After this a new line entitled "FREE SPACE" should appear under the selected disk.

If you select some free space, you will be offered to create new partition. You will have to answer a quick series of questions about its size, type (primary or logical), and location (beginning or end of the free space). After this, you will be presented with detailed overview of your new partition. There are options like mountpoint, mount options, bootable flag, or way of usage. If you don't like the preselected defaults, feel free to change them to your liking. E.g. by selecting the option Use as:, you can choose different filesystem for this partition including the possibility to use the partition for swap, software RAID, LVM, or not use it at all. Other nice feature is the possibility to copy data from existing partition onto this one. When you are satisfied with your new partition, select Done setting up the partition and you will be thrown back to the **partman**'s main screen.

If you decide you want to change something about your partition, simply select the partition, which will bring you to the partition configuration menu. Because this is the same screen like when creating a new partition, you can change the same set of options. One thing which might not be very obvious at a first glance is that you can resize the partition by selecting the item displaying the size of the partition. Filesystems known to work are at least fat16, fat32, ext2, ext3 and swap. This menu also allows you to delete a partition.

Be sure to create at least two partitions: one for the *root* filesystem (which must be mounted as `/`) and one for *swap*. If you forget to mount the root filesystem, **partman** won't let you continue until you correct this issue.

Capabilities of **partman** can be extended with installer modules, but are dependent on your system's architecture. So if you can't see all promised goodies, check if you have loaded all required modules (e.g. `partman-ext3`, `partman-xfs`, or `partman-lvm`).

After you are satisfied with partitioning, select Finish partitioning and write changes to disk from the partitioning menu. You will be presented with a summary of changes made to the disks and asked to confirm that the filesystems should be created as requested.

6.3.2.2. Configuring Logical Volume Manager (LVM)

If you are working with computers at the level of system administrator or "advanced" user, you have surely seen the situation where some disk partition (usually the most important one) was short on space, while some other partition was grossly underused and you had to manage this situation with moving stuff around, symlinking, etc.

To avoid the described situation you can use Logical Volume Manager (LVM). Simply said, with LVM you can combine your partitions (*physical volumes* in LVM lingo) to form a virtual disc (so called *volume group*), which can then be divided into virtual partitions (*logical volumes*). The point is that logical volumes (and of course underlying volume groups) can span across several physical discs.

Now when you realize you need more space for your old 160GB `/home` partition, you can simply add a new 300GB disc to the computer, join it with your existing volume group and then resize the logical volume which holds your `/home` filesystem and voila — your users have some room again on their renewed 460GB partition. This example is of course a bit oversimplified. If you haven't read it yet, you should consult the LVM HOWTO (http://www.tldp.org/HOWTO/LVM-HOWTO.html).

LVM setup in `debian-installer` is quite simple. At first, you have to mark your partitions to be used as physical volumes for LVM. (This is done in **partman** in the Partition settings menu where you should select Use as:⟶physical volume for LVM.) Then start the **lvmcfg** module (either directly from **partman** or from the `debian-installer`'s main menu) and combine physical volumes to volume group(s) under the Modify volume groups (VG) menu. After that, you should create logical volumes on the top of volume groups from the menu Modify logical volumes (LV).

After returning from **lvmcfg** back to **partman**, you will see any created logical volumes in the same way as ordinary partitions (and you should treat them like that).

6.3.2.3. Configuring Multidisk Device (Software RAID)

If you have more than one harddrive[1] in your computer, you can use **mdcfg** to setup your drives for increased performance and/or better reliability of your data. The result is called *Multidisk Device* (or after its most famous variant *software RAID*).

MD is basically a bunch of partitions located on different disks and combined together to form a *logical* device. This device can then be used like an ordinary partition (i.e. in **partman** you can format it, assign a mountpoint, etc.).

The benefit you gain depends on a type of a MD device you are creating. Currently supported are:

1. To be honest, you can construct MD device even from partitions residing on single physical drive, but that won't bring you anything useful.

RAID0

Is mainly aimed at performance. RAID0 splits all incoming data into *stripes* and distributes them equally over each disk in the array. This can increase the speed of read/write operations, but when one of the disks fails, you will loose *everything* (part of the information is still on the healthy disk(s), the other part *was* on the failed disk).

The typical use for RAID0 is a partition for video editing.

RAID1

Is suitable for setups where reliability is the first concern. It consists of several (usually two) equally sized partitions where every partition contains exactly the same data. This essentially means three things. First, if one of your disks fails, you still have the data mirrored on the remaining disks. Second, you can use only a fraction of the available capacity (more precisely, it is the size of the smallest partition in the RAID). Third, file reads are load balanced among the disks, which can improve performance on a server, such as a file server, that tends to be loaded with more disk reads than writes.

Optionally you can have a spare disk in the array which will take the place of the failed disk in the case of failure.

RAID5

Is a good compromise between speed, reliability and data redundancy. RAID5 splits all incomming data into stripes and distributes them equally on all but one disks (similar to RAID0). Unlike RAID0, RAID5 also computes *parity* information, which gets written on the remaining disk. The parity disk is not static (that would be called RAID4), but is changing periodically, so the parity information is distributed equally on all disks. When one of the disks fails, the missing part of information can be computed from remaining data and its parity. RAID5 must consist of at least three active partitions. Optionally you can have a spare disk in the array which will take the place of the failed disk in the case of failure.

As you can see, RAID5 has similar degree of reliability like RAID1 while achieving less redundancy. On the other hand it might be a bit slower on write operation than RAID0 due to computation of parity information.

To sum it up:

Type	Minimum Devices	Spare Device	Survives disk failure?	Available Space
RAID0	2	no	no	Size of the smallest partition multiplied by number of devices in RAID
RAID1	2	optional	yes	Size of the smallest partition in RAID
RAID5	3	optional	yes	Size of the smallest partition multiplied by (number of devices in RAID minus one)

If you want to know the whole truth about Software RAID, have a look at Software RAID HOWTO (http://www.tldp.org/HOWTO/Software-RAID-HOWTO.html).

To create a MD device, you need to have the desired partitions it should consist of marked for use in a RAID. (This is done in **partman** in the Partition settings menu where you should select Use as:⟶physical volume for RAID.)

Warning

Support for MD is a relatively new addition to the installer. You may experience problems for some RAID levels and in combination with some bootloaders if you try to use MD for the root (`/`) filesystem. For experienced users, it may be possible to work around some of these problems by executing some configuration or installation steps manually from a shell.

Next, you should choose Configure software RAID from the main **partman** menu. On the first screen of **mdcfg** simply select Create MD device. You will be presented with a list of supported types of MD devices, from which you should choose one (e.g. RAID1). What follows depends on the type of MD you selected.

- RAID0 is simple — you will be issued with the list of available RAID partitions and your only task is to select the partitions which will form the MD.
- RAID1 is a bit more tricky. First, you will be asked to enter the number of active devices and the number of spare devices which will form the MD. Next, you need to select from the list of available RAID partitions those that will be active and then those that will be spare. The count of selected partitions must be equal to the number provided few seconds ago. Don't worry. If you make a mistake and select different number of partitions, the `debian-installer` won't let you continue until you correct the issue.
- RAID5 has similar setup procedure as RAID1 with the exception that you need to use at least *three* active partitions.

It is perfectly possible to have several types of MD at once. For example if you have three 200 GB hard drives dedicated to MD, each containing two 100 GB partitions, you can combine first partitions on all three disk into the RAID0 (fast 300 GB video editing partition) and use the other three partitions (2 active and 1 spare) for RAID1 (quite reliable 100 GB partition for `/home`).

After you setup MD devices to your liking, you can Finish **mdcfg** to return back to the **partman** to create filesystems on your new MD devices and assign them the usual attributes like mountpoints.

6.3.3. Installing the Base System

Although this stage is the least problematic, it consumes most time of the install because it downloads, verifies and unpacks the whole base system. If you have a slow computer or network connection, this could take some time.

6.3.3.1. Base System Installation

During the Base installation, package unpacking and setup messages are redirected to **`tty3`**. You can access this terminal by pressing **Left Alt**-**F3**; get back to the main installer process with **Left Alt**-**F1**.

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

As part of the installation, a Linux kernel will be installed. At the default priority, the installer will choose one for you that best matches your hardware. In lower priority modes, you will be able to choose from a list of available kernels.

6.3.4. Making Your System Bootable

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

Note that multiple operating systems 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.

6.3.4.1. Detecting other operating systems

Before a boot loader is installed, the installer will attempt to probe for other operating systems which are installed on the machine. If it finds a supported operating system, you will be informed of this during the boot loader installation step, and the computer will be configured to boot this other operating system in addition to Debian.

Note that multiple operating systems booting on a single machine is still something of a black art. The automatic support for detecting and setting up boot loaders to boot other operating systems varies by architecture and even by subarchitecture. If it does not work you should consult your boot manager's documentation for more information.

> **Note:** The installer may fail to detect other operating systems if the partitions on which they reside are mounted when the detection takes place. This may occur if you select a mountpoint (e.g. /win) for a partition containing another operating system in **partman**, or if you have mounted partitions manually from a console.

6.3.4.2. Install the Grub Boot Loader on a Hard Disk

The main i386 boot loader is called "grub". Grub is a flexible and robust boot loader and a good default choice for newbies and old hands alike.

By default, grub will be installed into the Master Boot Record (MBR), where it will take over complete control of the boot process. If you prefer, you can install it elsewhere. See the grub manual for complete information.

If you do not want to install grub at all, use the Back button to get to the main menu, and from there select whatever bootloader you would like to use.

6.3.4.3. Install the LILO Boot Loader on a Hard Disk

The second i386 boot loader is called "LILO". It is an old complex program which offers lots of functionality, including DOS, Windows, and OS/2 boot management. Please carefully read the in-

structions in the directory `/usr/share/doc/lilo/` if you have special needs; also see the LILO mini-HOWTO (http://www.tldp.org/HOWTO/LILO.html).

> **Note:** Currently the LILO installation will only create menu entries for other operating systems if these can be *chainloaded*. This means you may have to manually add a menu entry for operating systems like GNU/Linux and GNU/Hurd after the installation.

`debian-installer` presents you three choices where to install the **LILO** boot loader:

Master Boot Record (MBR)

This way the **LILO** will take complete control of the boot process.

new Debian partition

Choose this if you want to use another boot manager. **LILO** will install itself at the beginning of the new Debian partition and it will serve as a secondary boot loader.

Other choice

Useful for advanced users who want to install **LILO** somewhere else. In this case you will be asked for desired location. You can use devfs style names, such as those that start with `/dev/ide`, `/dev/scsi`, and `/dev/discs`, as well as traditional names, such as `/dev/hda` or `/dev/sda`.

If you can no longer boot into Windows 9x (or DOS) after this step, you'll need to use a Windows 9x (MS-DOS) boot disk and use the **`fdisk /mbr`** command to reinstall the MS-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 Section 8.3.

6.3.4.4. Continue Without Boot Loader

This option can be used to complete the installation even when no boot loader is to be installed, either because the arch/subarch doesn't provide one, or because none is desired (e.g. you will use existing boot loader).

If you plan to manually configure your bootloader, you should check the name of the installed kernel in `/target/boot`. You should also check that directory for the presence of an *initrd*; if one is present, you will probably have to instruct your bootloader to use it. Other information you will need are the disk and partition you selected for your `/` filesystem and, if you chose to install `/boot` on a separate partition, also your `/boot` filesystem.

6.3.5. Finishing the First Stage

These are the last bits to do before rebooting to your new Debian. It mostly consists of tidying up after the `debian-installer`.

6.3.5.1. Finish the Installation and Reboot

This is the last step in the initial Debian installation process. You will be prompted to remove the boot media (CD, floppy, etc) that you used to boot the installer. The installer will do any last minute tasks, and then reboot into your new Debian system.

6.3.6. Miscellaneous

The components listed in this section are usually not involved in the installation process, but are waiting in the background to help the user in case something goes wrong.

6.3.6.1. Saving the installation logs

If the installation is successful, the logfiles created during the installation process will be automatically saved to `/var/log/debian-installer/` on your new Debian system.

Choosing Save debug logs from the main menu allows you to save the log files to a floppy disk. This can be useful if you encounter fatal problems during the installation and wish to study the logs on another system or attach them to an installation report.

6.3.6.2. Using the Shell and Viewing the Logs

There is an Execute a Shell item on the menu. If the menu is not available when you need to use the shell, press **Left Alt**-**F2** (on a Mac keyboard, **Option**-**F2**) to switch 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** and by typing **help**. The text editor is **nano**. The shell has some nice features like autocompletion and history.

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, or type **exit** if you used a menu item to open the shell.

6.3.6.3. Installation Over the Network

One of the more interesting components is *network-console*. It allows you to do a large part of the installation over the network via SSH. The use of the network implies you will have to perform the first steps of the installation from the console, at least to the point of setting up the networking. (Although you can automate that part with Section 4.7.)

This component is not loaded into the main installation menu by default, so you have to explicitly ask for it. If you are installing from CD, you need to boot with medium priority or otherwise invoke the main installation menu and choose Load installer components from CD and from the list of additional components select network-console: Continue installation remotely using SSH. Successful load is indicated by a new menu entry called Continue installation remotely using SSH.

After selecting this new entry, you will be asked for a new password to be used for connecting to the installation system and for its confirmation. That's all. Now you should see a screen which instructs

you to login remotely as the user *installer* with the password you just provided. Another important detail to notice on this screen is the fingerprint of this system. You need to transfer the fingerprint securely to the "person who will continue the installation remotely".

Should you decide to continue with the installation locally, you can always press **Enter**, which will bring you back to the main menu, where you can select another component.

Now let's switch to the other side of the wire. As a prerequisite, you need to configure your terminal for UTF-8 encoding, because that is what the installation system uses. If you do not, remote installation will be still possible, but you may encounter strange display artefacts like destroyed dialog borders or unreadable non-ascii characters. Establishing a connection with the installation system is as simple as typing:

```
$ ssh -l installer install_host
```

Where *install_host* is either the name or IP address of the computer being installed. Before the actual login the fingerprint of the remote system will be displayed and you will have to confirm that it is correct.

> **Note:** If you install several computers in turn and they happen to have the same IP address or hostname, **ssh** will refuse to connect to such host. The reason is that it will have different fingerprint, which is usually a sign of a spoofing attack. If you are sure this is not the case, you will need to delete the relevant line from `~/.ssh/known_hosts` and try again.

After the login you will be presented with an initial screen where you have two possibilities called Start menu and Start shell. The former brings you to the main installer menu, where you can continue with the installation as usual. The latter starts a shell from which you can examine and possibly fix the remote system. You should only start one SSH session for the installation menu, but may start multiple sessions for shells.

Warning

After you have started the installation remotely over SSH, you should not go back to the installation session running on the local console. Doing so may corrupt the database that holds the configuration of the new system. This in turn may result in a failed installation or problems with the installed system.

Also, if you are running the SSH session from an X terminal, you should not resize the window as that will result in the connection being terminated.

6.3.6.4. Running base-config From Within `debian-installer`

It is possible to configure the base system within the first stage installer (before rebooting from the hard drive), by running **base-config** in a *chroot* environment. This is mainly useful for testing the installer and should normally be avoided.

Chapter 7. Booting Into Your New Debian System

7.1. The Moment of Truth

Your system's first boot on its own power is what electrical engineers call the "smoke test".

If you are booting directly into Debian, and the system doesn't start up, either use your original installation boot media, or insert the custom boot floppy if you have one, and reset your system. This way, you will probably need to add some boot arguments like **root=*root***, where *root* is your root partition, such as `/dev/sda1`.

7.2. 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`. Its concept is very similar to the `debian-installer` from the first stage. Indeed, `base-config` consists of a number of specialized components, where each component handles one configuration task, contains "hidden menu in the background" and also uses the same navigation system.

If you wish to re-run the `base-config` at any point after installation is complete, as root run **base-config**.

7.2.1. Configuring Your Time Zone

After a welcome screen, you will be prompted to configure your time zone. First select whether the hardware clock of your system is set to local time or Greenwich Mean Time (GMT or UTC). The time displayed in the dialog may help you decide on the correct option. Systems that (also) run Dos or Windows are normally set to local time. If you want to dual-boot, select local time instead of GMT.

Depending on the location selected at the beginning of the installation process, you will next be shown either a single timezone or a list of timezones relevant for that location. If a single timezone is shown, choose Yes to confirm or choose No to select from the full list of timezones. If a list is shown, select your timezone from the list, or select Other for the full list.

7.2.2. Setting Up Users And Passwords

7.2.2.1. 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 at least 6 characters, and should contain both upper- and lower-case characters, as well as punctuation characters. Take extra care when setting your root pass-

word, 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 password out, unless you are administering a machine with more than one system administrator.

7.2.2.2. 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.

You will first be prompted for the user's full name. Then you'll be asked for a name for the user account; generally your first name or something similar will suffice and indeed will be the default. Finally, you will be prompted for a password for this account.

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

7.2.3. Setting Up PPP

If no network was configured during the first stage of the installation, you will next be asked whether you wish to install the rest of the system using PPP. PPP is a protocol used to establish dialup connections with modems. If you configure the modem at this point, the installation system will be able to download additional packages or security updates from the Internet during the next steps of the installation. If you don't have a modem in your computer or if you prefer to configure your modem after the installation, you can skip this step.

In order to configure your PPP connection, you will need some information from your Internet Service Provider (ISP), including phone number, username, password and DNS servers (optional). Some ISPs provide installation guidelines for Linux distributions. You can use that information even if they don't specifically target Debian since most of the configuration parameters (and software) is similar amongst Linux distributions.

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 trouble-free PPP connection setup. However, if it does not work for you, see below for detailed instructions.

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

Edit `/etc/ppp/peers/provider` and replace **/dev/modem** with **/dev/ttyS#** where `#` stands for the number of your serial port. In Linux, serial ports are counted from 0; your first serial port (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.

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

7.2.3.1. Setting Up PPP over Ethernet (PPPOE)

PPPOE is a protocol related to PPP used for some broadband connections. There is currently no support in base configuration to help you set this up. However, the necessary software has been installed, which means you can configure PPPOE manually at this stage of the installation by switching to VT2 and running **pppoeconf**.

7.2.4. Configuring APT

The main means that people use to install packages on their system is via a program called **apt-get**, from the apt package.[1] Other front-ends for package management, like **aptitude**, **synaptic** and the older **dselect** also use and depend on **apt-get**. These front-ends are recommended for new users, since they integrate some additional features (package searching and status checks) in a nice user interface.

APT must be configured 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 an official CD-ROM is in the drive at this point, then that CD-ROM should automatically be configured as an apt source without prompting. You will notice this because you will see the CD-ROM being scanned.

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

For users without an official CD-ROM, you will be offered an array of choices for how Debian packages are accessed: FTP, HTTP, CD-ROM, or a local file system.

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.

7.2.4.1. 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 somewhat slower making connections.

The next step during the configuration of network package sources is to tell **apt-setup** which country you live in. This configures which of the official Debian Internet mirrors you will connect to. Depending on which country you select, you will be presented with a list of possible servers. It's generally fine to pick the one at the top of the list, but any of them should work. Note however that the mirror list provided by the installation was generated when this version of Debian was released and some mirrors may no longer be available.

After you have selected a mirror, you will be asked if a proxy server should be used. A proxy server is a server that will forward all your HTTP and/or FTP requests to the Internet and is most often used to regulate and optimize access to the Internet on corporate networks. In some networks only the proxy server is allowed access to the Internet, in which case you will have to enter the name of the proxy server. You may also have to include an user name and password. Most home users will not need to specify a proxy server, although some ISPs may provide proxy servers for their users.

After you select a mirror, your new network package source will be tested. If all goes well, you will be prompted whether you want to add another package source. If you have any problems using the package source you selected, try using a different mirror (either from your country list or from the global list), or try using a different network package source.

7.2.5. Package Installation

Next you will be offered a number of pre-rolled software configurations offered by Debian. You could always choose, package by package, what you want to install on your new machine. This is the purpose of the **aptitude** program, described below. But this can be a long task with around 15250 packages available in Debian!

So, you have the ability to choose *tasks* first, and then add on more individual packages later. These tasks loosely represent a number of different jobs or things you want to do with your computer, such as "desktop environment", "web server", or "print server"[2]. Section C.3 lists the space requirements for the available tasks.

Once you've selected your tasks, select Ok. At this point, **aptitude** will install the packages you've selected.

2. You should know that to present this list, **base-config** is merely invoking the **tasksel** program. For manual package selection, the **aptitude** program is being run. Any of these can be run at any time after installation to install (or remove) more packages. If you are looking for a specific single package, after installation is complete, simply run **`aptitude install`** ***`package`***, where *package* is the name of the package you are looking for.

Note: Even if you did not select any tasks at all, any standard, important, or required priority packages that are not yet present on your system will be installed. This functionality is the same as running **`tasksel -ris`** at the command line, and currently involves a download of about 37M of archives. You will be shown the number of packages to be installed, and how many kilobytes of packages, if any, need to be downloaded.

If you do want to choose what to install on a package by package basis, select the "manual package selection" option in **tasksel**. If you select one or more tasks alongside this option, **aptitude** will be called with the **--visual-preview** option. This means you will be able to review[3] the packages that are to be installed. If you do not select any tasks, the normal **aptitude** screen will be displayed. After making your selections you should press "**`g`**" to start the download and installation of packages.

Note: If you choose "manual package selection" *without* selecting any tasks, no packages will be installed by default. This means you can use this option if you want to install a minimal system, but also that the responsibility for selecting any packages not installed as part of the base system (before the reboot) that might be required for your system lies with you.

Of the 15250 packages available in Debian, only a small minority are covered by tasks offered in the Task Installer. To see information on more packages, either use **`apt-cache search`** ***`search-string`*** for some given search string (see the apt-cache(8) man page), or run **aptitude** as described below.

7.2.5.1. Advanced Package Selection with aptitude

Aptitude is a modern program for managing packages. **aptitude** allows you to select individual packages, set of packages matching given criteria (for advanced users), or whole tasks.

The most basic keybindings are:

Key	Action
Up, **Down**	Move selection up or down.
Enter	Open/collapse/activate item.
+	Mark package for installation.
-	Mark package for removal.
d	Show package dependencies.
g	Actually download/install/remove packages.
q	Quit current view.
F10	Activate menu.

For more commands see the online help under the **?** key.

3. You can also change the default selections. If you would like to select any additional package, use View⟶New Package View.

7.2.6. Prompts During Software Installation

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

7.2.7. Configuring Your Mail Transport Agent

Today, email is a very important part of many people's life, so it's no surprise Debian lets you configure your mail system right as a part of the installation process. The standard mail transport agent in Debian is **exim4**, which is relatively small, flexible, and easy to learn.

You may ask if this is needed even if your computer is not connected to any network. The short answer is: Yes. The longer explanation: Some system utilities (like **cron**, **quota**, **aide**, ...) may send you important notices via email.

So on the first screen you will be presented with several common mail scenarios. Choose the one that most closely resembles your needs:

internet site

> Your system is connected to a network and your mail is sent and received directly using SMTP. On the following screens you will be asked a few basic questions, like your machine's mail name, or a list of domains for which you accept or relay mail.

mail sent by smarthost

> In this scenario is your outgoing mail forwarded to another machine, called a "smarthost", which does the actual job for you. Smarthost also usually stores incoming mail addressed to your computer, so you don't need to be permanently online. That also means you have to download your mail from the smarthost via programs like fetchmail. This option is suitable for dial-up users.

local delivery only

> Your system is not on a network and mail is sent or received only between local users. Even if you don't plan to send any messages, this option is highly recommended, because some system utilities may send you various alerts from time to time (e.g. beloved "Disk quota exceeded"). This option is also convenient for new users, because it doesn't ask any further questions.

no configuration at this time

> Choose this if you are absolutely convinced you know what you are doing. This will leave you with an unconfigured mail system — until you configure it, you won't be able to send or receive any mail and you may miss some important messages from your system utilities.

If none of these scenarios suits your needs, or if you need a finer setup, you will need to edit configuration files under the `/etc/exim4` directory after the installation is complete. More information about **exim4** may be found under `/usr/share/doc/exim4`.

7.3. Log In

After you've installed packages, you'll be presented with the login prompt. Log in using the personal login and password you selected. Your system is now ready to use.

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

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

In addition, there are some special folders within the `/usr/share/doc/` hierarchy. Linux HOWTOs are installed in *.gz* format, in `/usr/share/doc/HOWTO/en-txt/`. After installing **dhelp** you will find a browse-able index of documentation in `/usr/share/doc/HTML/index.html`.

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

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

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

Chapter 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. A lot of valuable information can also be found in the Debian Reference (http://www.debian.org/doc/user-manuals#quick-reference). This list of Unix FAQs (http://www.faqs.org/faqs/unix-faq/) contains a number of UseNet documents which provide a nice historical reference.

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

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.

8.2.1. Debian Packaging System

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

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

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

One of the best installation methods is apt. You can use the command line version **apt-get** or full-screen text version aptitude. Note apt will also let you merge main, contrib, and non-free so you can have export-restricted packages as well as standard versions.

8.2.2. Application Version Management

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

8.2.3. Cron Job Management

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

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

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

8.3. Reactivating DOS and Windows

After installing the base system and writing to the *Master Boot Record*, you will be able to 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** man pages `lilo.conf` for an overview of configuration keywords and `lilo` 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/share/doc` as well. In particular, `/usr/share/doc/HOWTO` and `/usr/share/doc/FAQ` contain lots of interesting information. To submit bugs, look at `/usr/share/doc/debian/bug*`. To read about Debian-specific issues for particular programs, look at `/usr/share/doc/(package name)/README.Debian`.

The Debian web site (http://www.debian.org/) contains a large quantity of documentation about Debian. In particular, see the Debian GNU/Linux FAQ (http://www.debian.org/doc/FAQ/) and the Debian Reference (http://www.debian.org/doc/user-manuals#quick-reference). An index of more Debian documentation is available from the Debian Documentation Project (http://www.debian.org/doc/ddp). 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. Last, but not least, the Debian Mailing List Archives (http://lists.debian.org/) contain a wealth of information on Debian.

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

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. Also, Debian often offers several alternative kernels. So you may want to check first if there is an alternative kernel image package that better corresponds to your hardware. However, it can be useful to compile a new kernel in order to:

- handle special hardware needs, or hardware conflicts with the pre-supplied kernels
- use options of the kernel which are not supported in the pre-supplied kernels (such as high memory support)
- optimize the kernel by removing useless drivers to speed up boot time
- create a monolithic instead of a modularized kernel
- run an updated or development kernel
- learn more about linux kernels

8.5.1. Kernel Image Management

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

To compile a kernel the Debian way, you need some packages: `fakeroot`, `kernel-package`, `kernel-source-2.6.8` (the most recent version at the time of this writing) and a few others which are probably already installed (see `/usr/share/doc/kernel-package/README.gz` for the complete list).

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

Note that you don't *have* to compile your kernel the "Debian way"; but we find that using the packaging system to manage your kernel is actually safer and easier. In fact, you can get your kernel sources right from Linus instead of `kernel-source-2.6.8`, 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 you have free rein over your machine and will extract your kernel source to somewhere in your home directory[1]. We'll also assume that your kernel version is 2.6.8. Make sure you are in the directory to where you want to unpack the kernel sources, extract them using **`tar xjf /usr/src/kernel-source-2.6.8.tar.bz2`** and change to the directory `kernel-source-2.6.8` that will have been created.

Now, you can configure your kernel. Run **`make xconfig`** if X11 is installed, configured and being run; run **`make menuconfig`** otherwise (you'll need `libncurses5-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-2.6.8-`*`subarchitecture`*`_custom.1.0_i386.deb`**. The *`subarchitecture`* 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-2.6.8` will be installed, containing your current configuration set. Your new `kernel-image-2.6.8` package is also clever enough to automatically use your platform's boot-loader to run an update on the booting, allowing you to boot without

1. There are other locations where you can extract kernel sources and build your custom kernel, but this is easiest as it does not require special permissions.

re-running the boot loader. If you have created a modules package, e.g., if you have PCMCIA, you'll need to install that package as well.

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

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

Appendix A. Installation Howto

This document describes how to install Debian GNU/Linux sarge for the Intel x86 (“i386”) with the new `debian-installer`. It is a quick walkthrough of the installation process which should contain all the information you will need for most installs. When more information can be useful, we will link to more detailed explanations in the Debian GNU/Linux Installation Guide.

A.1. Preliminaries

If you encounter bugs during your install, please refer to Section 5.3.6 for instructions on how to report them. If you have questions which cannot be answered by this document, please direct them to the debian-boot mailing list (debian-boot@lists.debian.org) or ask on IRC (#debian-boot on the freenode network).

A.2. Booting the installer

The debian-cd team provides builds of CD images using `debian-installer` on the Debian CD page (http://www.debian.org/CD/). For more information on where to get CDs, see Section 4.1.

Some installation methods require other images than CD images. Section 4.2.1 explains how to find images on Debian mirrors.

The subsections below will give the details about which images you should get for each possible means of installation.

A.2.1. CDROM

There are two different netinst CD images which can be used to install sarge with the `debian-installer`. These images are intended to boot from CD and install additional packages over a network, hence the name ’netinst’. The difference between the two images is that on the full netinst image the base packages are included, whereas you have to download these from the web if you are using the business card image. If you’d rather, you can get a full size CD image which will not need the network to install. You only need the first CD of the set.

Download whichever type you prefer and burn it to a CD. To boot the CD, you may need to change your BIOS configuration, as explained in Section 3.6.1.

A.2.2. Floppy

If you can’t boot from CD, you can download floppy images to install Debian. You need the `floppy/boot.img`, the `floppy/root.img` and possibly one of the driver disks.

The boot floppy is the one with `boot.img` on it. This floppy, when booted, will prompt you to insert a second floppy — use the one with `root.img` on it.

If you’re planning to install over the network, you will usually need the `floppy/net-drivers.img`, which contains additional drivers for many ethernet cards, and support for PCMCIA.

If you have a CD, but cannot boot from it, then boot from floppies and use `floppy/cd-drivers.img` on a driver disk to complete the install using the CD.

Floppy disks are one of the least reliable media around, so be prepared for lots of bad disks (see Section 5.3.1). Each `.img` file you downloaded goes on a single floppy; you can use the dd command to write it to /dev/fd0 or some other means (see Section 4.3 for details). Since you'll have more than one floppy, it's a good idea to label them.

A.2.3. USB memory stick

It's also possible to install from removable USB storage devices. For example a USB keychain can make a handy Debian install medium that you can take with you anywhere.

The easiest way to prepare your USB memory stick is to download `hd-media/boot.img.gz`, and use gunzip to extract the 128 MB image from that file. Write this image directly to your memory stick, which must be at least 128 mb in size. Of course this will destroy anything already on the memory stick. Then mount the memory stick, which will now have a FAT filesystem on it. Next, download a Debian netinst CD image, and copy that file to the memory stick; any filename is ok as long as it ends in `.iso`.

There are other, more flexible ways to set up a memory stick to use the debian-installer, and it's possible to get it to work with smaller memory sticks. For details, see Section 4.4.

Some BIOSes can boot USB storage directly, and some cannot. You may need to configure your BIOS to boot from a "removable drive" or even a "USB-ZIP" to get it to boot from the USB device. If it doesn't, you can boot from one floppy and use the USB stick for the rest of the install. For helpful hints and details, see Section 5.1.3.

A.2.4. Booting from network

It's also possible to boot `debian-installer` completely from the net. The various methods to netboot depend on your architecture and netboot setup. The files in `netboot/` can be used to netboot `debian-installer`.

The easiest thing to set up is probably PXE netbooting. Untar the file `netboot/pxeboot.tar.gz` into `/var/lib/tftpboot` or wherever is appropriate for your tftp server. Set up your DHCP server to pass filename `/pxelinux.0` to clients, and with luck everything will just work. For detailed instructions, see Section 4.6.

A.2.5. Booting from hard disk

It's possible to boot the installer using no removable media, but just an existing hard disk, which can have a different OS on it. Download `hd-media/initrd.gz`, `hd-media/vmlinuz`, and a Debian CD image to the top-level directory of the hard disk. Make sure that the CD image has a filename ending in `.iso`. Now it's just a matter of booting linux with the initrd. Section 5.1.2 explains one way to do it.

A.3. Installation

Once the installer starts, you will be greeted with an initial screen. Press **Enter** to boot, or read the instructions for other boot methods and parameters (see Section 5.2). If you want a 2.6 kernel, type

linux26 at the `boot:` prompt. [1]

After a while you will be asked to select your language. Use the arrow keys to pick a language and press **Enter** to continue. Next you'll be asked to select your country, with the choices including countries where your language is spoken. If it's not on the short list, a list of all the countries in the world is available.

You may be asked to confirm your keyboard layout. Choose the default unless you know better.

Now sit back while debian-installer detects some of your hardware, and loads the rest of itself from CD, floppy, USB, etc.

Next the installer will try to detect your network hardware and set up networking by DHCP. If you are not on a network or do not have DHCP, you will be given the opportunity to configure the network manually.

Now it is time to partition your disks. First you will be given the opportunity to automatically partition either an entire drive, or free space on a drive. This is recommended for new users or anyone in a hurry, but if you do not want to autopartition, choose manual from the menu.

If you have an existing DOS or Windows partition that you want to preserve, be very careful with automatic partitioning. If you choose manual partitioning, you can use the installer to resize existing FAT or NTFS partitions to create room for the Debian install: simply select the partition and specify its new size.

On the next screen you will see your partition table, how the partitions will be formatted, and where they will be mounted. Select a partition to modify or delete it. If you did automatic partitioning, you should just be able to choose Finished partitioning from the menu to use what it set up. Remember to assign at least one partition for swap space and to mount a partition on `/`. Appendix B has more information about partitioning.

Now `debian-installer` formats your partitions and starts to install the base system, which can take a while. That is followed by installing a kernel.

The last step is to install a boot loader. If the installer detects other operating systems on your computer, it will add them to the boot menu and let you know. By default GRUB will be installed to the master boot record of the first harddrive, which is generally a good choice. You'll be given the opportunity to override that choice and install it elsewhere.

`debian-installer` will now tell you that the installation has finished. Remove the cdrom or other boot media and hit **Enter** to reboot your machine. It should boot up into the next stage of the install process, which is explained in Chapter 7.

If you need more information on the install process, see Chapter 6.

A.4. Send us an installation report

If you successfully managed an installation with `debian-installer`, please take time to provide us with a report. There is a template named `install-report.template` in the `/root` directory of a freshly installed system. Please fill it out and file it as a bug against the package `installation-reports`, as explained in Section 5.3.6.

If you did not reach base-config or ran into other trouble, you probably found a bug in debian-installer. To improve the installer it is necessary that we know about them, so please take the time to report them. You can use an installation report to report problems; if the install completely fails, see Section 5.3.5.

1. The 2.6 kernel is available for most boot methods, but not when booting from a floppy.

A.5. And finally..

We hope that your Debian installation is pleasant and that you find Debian useful. You might want to read Chapter 8.

Appendix B. Partitioning for Debian

B.1. Deciding on Debian Partitions and Sizes

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

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

The second reason is generally more important in a business setting, but it really depends on your use of the machine. For example, a mail server getting spammed with e-mail can easily fill a partition. If you made `/var/mail` a separate partition on the mail server, most of the system will remain working even if you get spammed.

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

B.2. The Directory Tree

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

Directory	Content
`bin`	Essential command binaries
`boot`	Static files of the boot loader
`dev`	Device files
`etc`	Host-specific system configuration
`home`	User home directories
`lib`	Essential shared libraries and kernel modules
`media`	Contains mount points for replaceable media

Directory	Content
`mnt`	Mount point for mounting a file system temporarily
`proc`	Virtual directory for system information (2.4 and 2.6 kernels)
`root`	Home directory for the root user
`sbin`	Essential system binaries
`sys`	Virtual directory for system information (2.6 kernels)
`tmp`	Temporary files
`usr`	Secondary hierarchy
`var`	Variable data
`opt`	Add-on application software packages

The following is a list of important considerations regarding directories and partitions. Note that disk usage varies widely given system configuration and specific usage patterns. The recommendations here are general guidelines and provide a starting point for partitioning.

- The root partition `/` must always physically contain `/etc`, `/bin`, `/sbin`, `/lib` and `/dev`, otherwise you won't be able to boot. Typically 150–250 MB is needed for the root partition.
- `/usr`: contains all user programs (`/usr/bin`), libraries (`/usr/lib`), documentation (`/usr/share/doc`), etc. This is the part of the file system that generally takes up most space. You should provide at least 500 MB of disk space. This amount should be increased depending on the number and type of packages you plan to install. A generous workstation or server installation should allow 4-6 GB.
- `/var`: variable data like news articles, e-mails, web sites, databases, the packaging system cache, etc. will be placed under this directory. The size of this directory depends greatly on the usage of your system, 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 gigabyte 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 MB. If hard drive space is at a premium and you don't plan on doing major system updates, you can get by with as little as 30 or 40 MB.
- `/tmp`: temporary data created by programs will most likely go in this directory. 40–100 MB should usually be enough. Some applications — including archive manipulators, CD/DVD authoring tools, and multimedia software — may use `/tmp` to temporarily store image files. If you plan to use such applications, you should adjust the space available in `/tmp` accordingly.
- `/home`: every user will put his personal data into a subdirectory of this directory. Its size 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. Reserve a lot more space if you plan to save a lot of multimedia files (MP3, movies) in your home directory.

B.3. Recommended Partitioning Scheme

For new users, personal Debian boxes, home systems, and other single-user setups, a single `/` partition (plus swap) is probably the easiest, simplest way to go. However, if your partition is larger than around 6GB, choose ext3 as your partition type. Ext2 partitions need periodic file system integrity checking, and this can cause delays during booting when the partition is large.

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

You might need a separate `/usr/local` partition if you plan to install many programs that are not part of the Debian distribution. If your machine will be a mail server, you might need to make `/var/mail` a separate partition. Often, putting `/tmp` on its own partition, for instance 20 to 50MB, is a good idea. If you are setting up a server with lots of user accounts, it's generally good to have a separate, large `/home` partition. In general, the partitioning situation varies from computer to computer depending on its uses.

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

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

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

As an example, an older home machine might have 32MB of RAM and a 1.7GB IDE drive on `/dev/hda`. There might be a 500MB partition for another operating system on `/dev/hda1`, a 32MB swap partition on `/dev/hda3` and about 1.2GB on `/dev/hda2` as the Linux partition.

For an idea of the space taken by tasks you might be interested in adding after your system installation is complete, check Section C.3.

B.4. Device Names in Linux

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

- The first floppy drive is named `/dev/fd0`.
- The second floppy drive is named `/dev/fd1`.
- The first SCSI disk (SCSI ID address-wise) is named `/dev/sda`.
- The second SCSI disk (address-wise) is named `/dev/sdb`, and so on.
- The first SCSI CD-ROM is named `/dev/scd0`, also known as `/dev/sr0`.
- The master disk on IDE primary controller is named `/dev/hda`.
- The slave disk on IDE primary controller is named `/dev/hdb`.

- The master and slave disks of the secondary controller can be called `/dev/hdc` and `/dev/hdd`, respectively. Newer IDE controllers can actually have two channels, effectively acting like two controllers.
- The 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 the drive models and/or capacities.

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.

B.5. Debian Partitioning Programs

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

partman

Recommended partitioning tool in Debian. This Swiss army knife can also resize partitions, create filesystems ("format" in Windows speak) and assign them to the mountpoints.

fdisk

The original Linux disk partitioner, good for gurus.

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

cfdisk

A simple-to-use, full-screen disk partitioner for the rest of us.

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**) by pressing **Alt** and **F2** keys together, and manually type in the name of the program you want to use (and arguments, if any). Then skip the Partition a Hard Disk step in **debian-installer** and continue to the next step.

If you will be working with more than 20 partitions on your ide disk, you will need to create devices for partitions 21 and beyond. The next step of initializing the partition will fail unless a proper device is present. As an example, here are commands you can use in **tty2** or under Execute A Shell to add a device so the 21st partition can be initialized:

```
# cd /dev
# mknod hda21 b 3 21
# chgrp disk hda21
# chmod 660 hda21
```

Booting into the new system will fail unless proper devices are present on the target system. After installing the kernel and modules, execute:

```
# cd /target/dev
# mknod hda21 b 3 21
# chgrp disk hda21
# chmod 660 hda21
```

Remember to mark your boot partition as "Bootable".

B.5.1. Partitioning for Intel x86

If you have an existing other operating system such as DOS or Windows and you want to preserve that operating system while installing Debian, you may need to resize its partition to free up space for the Debian installation. The installer supports resizing of both FAT and NTFS filesystems; when you get to the installer's partitioning step, select the option to partition manually and then simply select an existing partition and change its size.

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 BIOSes, 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.tldp.org/HOWTO/Partition/) and the Phoenix BIOS FAQ (http://www.phoenix.com/en/Customer+Services/BIOS/BIOS+FAQ/default.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). However the normal Debian GNU/Linux system provides only 20 devices for partitions, so you may not install on partitions higher than 20 unless you first manually create devices for those partitions.

If you have a large IDE disk, and are using neither LBA addressing, nor overlay drivers (sometimes provided by hard disk manufacturers), then the boot partition (the partition containing your kernel image) must be placed within the first 1024 cylinders of your hard drive (usually around 524 megabytes, without BIOS translation).

This restriction doesn't apply if 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 fall-back, 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.tldp.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 (5–10MB 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.

Appendix C. Random Bits

C.1. Preconfiguration File Example

This is a complete working example of a preconfiguration file for an automated install. Its use is explained in Section 4.7. You may want to uncomment some of the lines before using the file.

> **Note:** In order to be able to properly present this example in the manual, we've had to split some lines. This is indicated by the use of the line-continuation-character "\" and extra indentation in the next line. In a real preconfiguration file, these split lines have to be joined into *one single line*. If you do not, preconfiguration will fail with unpredictable results.
>
> A "clean" example file is available from ../example-preseed.txt.

```
#### Startup.

# To use a preseed file, you'll first need to boot the installer,
# and tell it what preseed file to use. This is done by passing the
# kernel a boot parameter, either manually at boot or by editing the
# syslinux.cfg (or similar) file and adding the parameter to the end
# of the append line(s) for the kernel.
#
# If you're netbooting, use this:
#   preseed/url=http://host/path/to/preseed
# If you're remastering a CD, you could use this:
#   preseed/file=/cdrom/preseed
# If you're installing from USB media, use this, and put the preseed file
# in the toplevel directory of the USB stick.
#   preseed/file=/hd-media/preseed
# Be sure to copy this file to the location you specify.
#
# Some parts of the installation process cannot be automated using
# some forms of preseeding, because the questions are asked before
# the preseed file is loaded. For example, if the preseed file is
# downloaded over the network, the network setup must be done first.
# One reason to use initrd preseeding is that it allows preseeding
# of even these early steps of the installation process.
#
# If a preseed file cannot be used to preseed some steps, the install can
# still be fully automated, since you can pass preseed values to the kernel
# on the command line. Just pass path/to/var=value for any of the preseed
# variables listed below.
#
# While you're at it, you may want to throw a debconf/priority=critical in
# there, to avoid most questions even if the preseeding below misses some.
# And you might set the timeout to 1 in syslinux.cfg to avoid needing to hit
# enter to boot the installer.
#
# Note that the kernel accepts a maximum of 8 command line options and
# 8 environment options (including any options added by default for the
# installer). If these numbers are exceeded, 2.4 kernels will drop any
# excess options and 2.6 kernels will panic. With kernel 2.6.9 or newer,
```

```
# you can use 32 command line options and 32 environment options.
#
# Some of the default options, like 'vga=normal' may be safely removed
# for most installations, which may allow you to add more options for
# preseeding.

# It is not possible to use preseeding to set language, country, and
# keyboard. Instead you should use kernel parameters. Example:
# languagechooser/language-name=English
# countrychooser/shortlist=US
# console-keymaps-at/keymap=us

#### Network configuration.

# Of course, this won't work if you're loading your preseed file from the
# network! But it's great if you're booting from CD or USB stick. You can
# also pass network config parameters in on the kernel params if you are
# loading preseed files from the network.

# netcfg will choose an interface that has link if possible. This makes it
# skip displaying a list if there is more than one interface.
d-i netcfg/choose_interface select auto

# If you have a slow dhcp server and the installer times out waiting for
# it, this might be useful.
#d-i netcfg/dhcp_timeout string 60

# If you prefer to configure the network manually, here's how:
#d-i netcfg/disable_dhcp boolean true
#d-i netcfg/get_nameservers string 192.168.1.1
#d-i netcfg/get_ipaddress string 192.168.1.42
#d-i netcfg/get_netmask string 255.255.255.0
#d-i netcfg/get_gateway string 192.168.1.1
#d-i netcfg/confirm_static boolean true

# Note that any hostname and domain names assigned from dhcp take
# precedence over values set here. However, setting the values still
# prevents the questions from being shown even if values come from dhcp.
d-i netcfg/get_hostname string unassigned-hostname
d-i netcfg/get_domain string unassigned-domain

# Disable that annoying WEP key dialog.
d-i netcfg/wireless_wep string
# The wacky dhcp hostname that some ISPs use as a password of sorts.
#d-i netcfg/dhcp_hostname string radish

#### Mirror settings.

d-i mirror/country string enter information manually
d-i mirror/http/hostname string http.us.debian.org
d-i mirror/http/directory string /debian
d-i mirror/suite string testing
d-i mirror/http/proxy string

#### Partitioning.

# If the system has free space you can choose to only partition that space.
```

```
#d-i partman-auto/init_automatically_partition \
#    select Use the largest continuous free space

# Alternatively, you can specify a disk to partition. The device name can
# be given in either devfs or traditional non-devfs format.
# For example, to use the first disk devfs knows of:
d-i partman-auto/disk string /dev/discs/disc0/disc

# You can choose from any of the predefined partitioning recipes:
d-i partman-auto/choose_recipe select \
     All files in one partition (recommended for new users)
#d-i partman-auto/choose_recipe select Desktop machine
#d-i partman-auto/choose_recipe select Multi-user workstation

# Or provide a recipe of your own...
# The recipe format is documented in the file devel/partman-auto-recipe.txt.
# If you have a way to get a recipe file into the d-i environment, you can
# just point at it.
#d-i partman-auto/expert_recipe_file string /hd-media/recipe

# If not, you can put an entire recipe in one line. This example creates
# a small /boot partition, suitable swap, and uses the rest of the space
# for the root partition:
#d-i partman-auto/expert_recipe string boot-root :: \
#    20 50 100 ext3 $primary{ } $bootable{ } method{ format } format{ } \
#    use_filesystem{ } filesystem{ ext3 } mountpoint{ /boot } . \
#    500 10000 1000000000 ext3 method{ format } format{ } \
#    use_filesystem{ } filesystem{ ext3 } mountpoint{ / } . \
#    64 512 300% linux-swap method{ swap } format{ } .
# For reference, here is that same recipe in a more readable form:
#    boot-root ::
#       40 50 100 ext3
#          $primary{ } $bootable{ }
#          method{ format } format{ }
#          use_filesystem{ } filesystem{ ext3 }
#          mountpoint{ /boot }
#       .
#       500 10000 1000000000 ext3
#          method{ format } format{ }
#          use_filesystem{ } filesystem{ ext3 }
#          mountpoint{ / }
#       .
#       64 512 300% linux-swap
#          method{ swap } format{ }
#       .

# This makes partman automatically partition without confirmation.
d-i partman/confirm_write_new_label boolean true
d-i partman/choose_partition select \
    Finish partitioning and write changes to disk
d-i partman/confirm boolean true

#### Boot loader installation.

# Grub is the default boot loader (for x86). If you want lilo installed
# instead, uncomment this:
#d-i grub-installer/skip boolean true
```

```
# This is fairly safe to set, it makes grub install automatically to the MBR
# if no other operating system is detected on the machine.
d-i grub-installer/only_debian boolean true

# This one makes grub-installer install to the MBR if if finds some other OS
# too, which is less safe as it might not be able to boot that other OS.
d-i grub-installer/with_other_os boolean true

# Alternatively, if you want to install to a location other than the mbr,
# uncomment and edit these lines:
#d-i grub-installer/bootdev  string (hd0,0)
#d-i grub-installer/only_debian boolean false
#d-i grub-installer/with_other_os boolean false

#### Finishing up the first stage install.

# Avoid that last message about the install being complete.
d-i prebaseconfig/reboot_in_progress note

#### Shell commands.

# d-i preseeding is inherently not secure. Nothing in the installer checks
# for attempts at buffer overflows or other exploits of the values of a
# preseed file like this one. Only use preseed files from trusted
# locations! To drive that home, and because it's generally useful, here's
# a way to run any shell command you'd like inside the installer,
# automatically.

# This first command is run as early as possible, just after
# preseeding is read.
#d-i preseed/early_command string anna-install some-udeb

# This command is run just before the install finishes, but when there is
# still a usable /target directory.
#d-i preseed/late_command string echo foo > /target/etc/bar

# This command is run just as base-config is starting up.
#base-config base-config/early_command string echo hi mom

# This command is run after base-config is done, just before the login:
# prompt. This is a good way to install a set of packages you want, or to
# tweak the configuration of the system.
#base-config base-config/late_command string \
#    apt-get install zsh; chsh -s /bin/zsh

###### Preseeding the 2nd stage of the installation.

#### Preseeding base-config.

# Avoid the introductory message.
base-config base-config/intro note

# Avoid the final message.
base-config base-config/login note

# If you installed a display manager, but don't want to start it immediately
```

```
# after base-config finishes.
#base-config base-config/start-display-manager boolean false

# Some versions of the installer can report back on what you've installed.
# The default is not to report back, but sending reports helps the project
# determine what software is most popular and include it on CDs.
#popularity-contest popularity-contest/participate boolean false

#### Clock and time zone setup.

# Controls whether or not the hardware clock is set to UTC.
#base-config tzconfig/gmt boolean true
# If you told the installer that you're in the United States, then you
# can set the time zone using this variable.
# (Choices are: Eastern, Central, Mountain, Pacific, Alaska, Hawaii,
# Aleutian, Arizona East-Indiana, Indiana-Starke, Michigan, Samoa, other)
#base-config tzconfig/choose_country_zone/US select Eastern
# If you told it you're in Canada.
# (Choices are: Newfoundland, Atlantic, Eastern, Central,
# East-Saskatchewan, Saskatchewan, Mountain, Pacific, Yukon, other)
#base-config tzconfig/choose_country_zone/CA select Eastern
# If you told it you're in Brazil. (Choices are: East, West, Acre,
# DeNoronha, other)
#base-config tzconfig/choose_country_zone/BR select East
# Many countries have only one time zone. If you told the installer you're
# in one of those countries, you can choose its standard time zone via this
# question.
#base-config tzconfig/choose_country_zone_single boolean true
# This question is asked as a fallback for countries other than those
# listed above, which have more than one time zone. You can preseed one of
# the time zones, or "other".
#base-config tzconfig/choose_country_zone_multiple select

#### Account setup.

# To preseed the root password, you have to put it in the clear in this
# file. That is not a very good idea, use caution!
#passwd passwd/root-password password r00tme
#passwd passwd/root-password-again password r00tme

# If you want to skip creation of a normal user account.
#passwd passwd/make-user boolean false

# Alternatively, you can preseed the user's name and login.
#passwd passwd/user-fullname string Debian User
#passwd passwd/username string debian
# And their password, but use caution!
#passwd passwd/user-password password insecure
#passwd passwd/user-password-again password insecure

#### Apt setup.

# This question controls what source the second stage installation uses
# for packages. Choices are cdrom, http, ftp, filesystem, edit sources list
# by hand
base-config apt-setup/uri_type select http
```

```
# If you choose ftp or http, you'll be asked for a country and a mirror.
base-config apt-setup/country select enter information manually
base-config apt-setup/hostname string http.us.debian.org
base-config apt-setup/directory string /debian
# Stop after choosing one mirror.
base-config apt-setup/another boolean false

# You can choose to install non-free and contrib software.
#base-config apt-setup/non-free boolean true
#base-config apt-setup/contrib boolean true

# Do enable security updates.
base-config apt-setup/security-updates boolean true

#### Package selection.

# You can choose to install any combination of tasks that are available.
# Available tasks as of this writing include: Desktop environment,
# Web server, Print server, DNS server, File server, Mail server,
# SQL database, Laptop, Standard system, manual package selection. The
# last of those will run aptitude. You can also choose to install no
# tasks, and force the installation of a set of packages in some other
# way. We recommend always including the Standard system task.
tasksel tasksel/first multiselect Desktop environment, Standard system
#tasksel tasksel/first multiselect Web server, Standard system

#### Mailer configuration.

# During a normal install, exim asks only a few questions. Here's how to
# avoid even those. More complicated preseeding is possible.
exim4-config exim4/dc_eximconfig_configtype \
    select no configuration at this time
exim4-config exim4/no_config boolean true
exim4-config exim4/no_config boolean true

# It's a good idea to set this to whatever user account you choose to
# create. Leaving the value blank results in postmaster mail going to
# /var/mail/mail.
exim4-config exim4/dc_postmaster string

#### X Configuration.

# Preseeding Debian's X config is possible, but you probably need to know
# some details about the video hardware of the machine, since Debian's X
# configurator does not do fully automatic configuration of everything.

# X can detect the right driver for some cards, but if you're preseeding,
# you override whatever it chooses. Still, vesa will work most places.
#xserver-xfree86 xserver-xfree86/config/device/driver select vesa

# A caveat with mouse autodetection is that if it fails, X will retry it
# over and over. So if it's preseeded to be done, there is a possibility of
# an infinite loop if the mouse is not autodetected.
#xserver-xfree86 xserver-xfree86/autodetect_mouse boolean true

# Monitor autodetection is recommended.
xserver-xfree86 xserver-xfree86/autodetect_monitor boolean true
```

```
# Uncomment if you have an LCD display.
#xserver-xfree86 xserver-xfree86/config/monitor/lcd boolean true
# X has three configuration paths for the monitor. Here's how to preseed
# the "medium" path, which is always available. The "simple" path may not
# be available, and the "advanced" path asks too many questions.
xserver-xfree86 xserver-xfree86/config/monitor/selection-method \
    select medium
xserver-xfree86 xserver-xfree86/config/monitor/mode-list \
    select 1024x768 @ 60 Hz

#### Everything else.

# Depending on what software you choose to install, or if things go wrong
# during the installation process, it's possible that other questions may
# be asked. You can preseed those too, of course. To get a list of every
# possible question that could be asked during an install, do an
# installation, and then run these commands:
#   debconf-get-selections --installer > file
#   debconf-get-selections >> file

# If you like, you can include other preseed files into this one.
# Any settings in those files will override pre-existing settings from this
# file. More that one file can be listed, separated by spaces; all will be
# loaded. The included files can have preseed/include directives of their
# own as well. Note that if the filenames are relative, they are taken from
# the same directory as the preseed file that includes them.
#d-i preseed/include string x.cfg

# More flexibly, this runs a shell command and if it outputs the names of
# preseed files, includes those files. For example, to switch configs based
# on a particular usb storage device (in this case, a built-in card reader):
#d-i preseed/include_command string \
#    if $(grep -q "GUID: 0aec3050aec305000001a003" /proc/scsi/usb-storage-*/*); \
#    then echo kraken.cfg; else echo otherusb.cfg; fi

# To check the format of your preseed file before performing an install,
# you can use debconf-set-selections:
#   debconf-set-selections -c preseed.cfg
```

C.2. Linux Devices

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

`fd0`	First Floppy Drive
`fd1`	Second Floppy Drive

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

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

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

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

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

null	Everything pointed to this device will disappear
zero	One can endlessly read zeros out of this device

C.2.1. Setting Up Your Mouse

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

```
mouse => /dev/psaux  => gpm => /dev/gpmdata -> /dev/mouse => X
```

```
/dev/ttyS0              (repeater)        (symlink)
/dev/ttyS1
```

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

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

```
# /etc/init.d/gpm restart
```

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

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

C.3. Disk Space Needed for Tasks

The base installation for i386 using the default 2.4 kernel, including all standard packages, requires 573MB of disk space.

The following table lists sizes reported by aptitude for the tasks listed in tasksel. Note that some tasks have overlapping constituents, so the total installed size for two tasks together may be less than the total obtained by adding up the numbers.

Note that you will need to add the sizes listed in the table to the size of the base installation when determining the size of partitions. Most of the size listed as "Installed size" will end up in `/usr`; the size listed as "Download size" is (temporarily) required in `/var`.

Task	Installed size (MB)	Download size (MB)	Space needed to install (MB)
Desktop	1392	460	1852
Web server	36	12	48
Print server	168	58	226
DNS server	2	1	3
File server	47	24	71
Mail server	10	3	13
SQL database	66	21	87

Note: The *Desktop* task will install both the Gnome and KDE desktop environments.

If you install in a language other than English, **tasksel** may automatically install a *localization task*, if one is available for your language. Space requirements differ per language; you should allow up to 200MB in total for download and installation.

C.4. Installing Debian GNU/Linux from a Unix/Linux System

This section explains how to install Debian GNU/Linux from an existing Unix or Linux system, without using the menu-driven installer as explained in the rest of the manual. This "cross-install" HOWTO has been requested by users switching to Debian GNU/Linux from Red Hat, Mandrake, and SUSE. In this section some familiarity with entering *nix commands and navigating the file system is assumed. In this section, `$` symbolizes a command to be entered in the user's current system, while `#` refers to a command entered in the Debian chroot.

Once you've got the new Debian system configured to your preference, you can migrate your existing user data (if any) to it, and keep on rolling. This is therefore a "zero downtime" Debian GNU/Linux install. It's also a clever way for dealing with hardware that otherwise doesn't play friendly with various boot or installation media.

C.4.1. Getting Started

With your current *nix partitioning tools, repartition the hard drive as needed, creating at least one filesystem plus swap. You need at least 150MB of space available for a console only install, or at least 300MB if you plan to install X.

To create file systems on your partitions. For example, to create an ext3 file system on partition `/dev/hda6` (that's our example root partition):

```
# mke2fs -j /dev/hda6
```

To create an ext2 file system instead, omit **`-j`**.

Initialize and activate swap (substitute the partition number for your intended Debian swap partition):

```
# mkswap /dev/hda5
# sync; sync; sync
# swapon /dev/hda5
```

Mount one partition as `/mnt/debinst` (the installation point, to be the root (`/`) filesystem on your new system). The mount point name is strictly arbitrary, it is referenced later below.

```
# mkdir /mnt/debinst
# mount /dev/hda6 /mnt/debinst
```

Note: If you want to have parts of the filesystem (e.g. /usr) mounted on separate partitions, you will need to create and mount these directories manually before proceding with the next stage.

C.4.2. Install debootstrap

The tool that the Debian installer uses, which is recognized as the official way to install a Debian base system, is **debootstrap**. It uses **wget** and **ar**, but otherwise depends only on `/bin/sh`. Install **wget** and **ar** if they aren't already on your current system, then download and install **debootstrap**.

If you have an rpm-based system, you can use alien to convert the .deb into .rpm, or download an rpm-ized version at http://people.debian.org/~blade/install/debootstrap

Or, you can use the following procedure to install it manually. Make a work folder for extracting the .deb into:

```
# mkdir work
# cd work
```

The **debootstrap** binary is located in the Debian archive (be sure to select the proper file for your architecture). Download the **debootstrap** .deb from the pool (http://ftp.debian.org/debian/pool/main/d/debootstrap/), copy the package to the work folder, and extract the binary files from it. You will need to have root privileges to install the binaries.

```
# ar -x debootstrap_0.X.X_arch.deb
# cd /
# zcat /full-path-to-work/work/data.tar.gz | tar xv
```

Note that running **debootstrap** may require you to have a minimal version of `glibc` installed (currently GLIBC_2.3). **debootstrap** itself is a shell script, but it calls various utilities that require `glibc`.

C.4.3. Run debootstrap

debootstrap can download the needed files directly from the archive when you run it. You can substitute any Debian archive mirror for **`http.us.debian.org/debian`** in the command example below, preferably a mirror close to you network-wise. Mirrors are listed at http://www.debian.org/misc/README.mirrors.

If you have a sarge Debian GNU/Linux CD mounted at `/cdrom`, you could substitute a file URL instead of the http URL: **`file:/cdrom/debian/`**

Substitute one of the following for *`ARCH`* in the **debootstrap** command: **`alpha`**, **`arm`**, **`hppa`**, **`i386`**, **`ia64`**, **`m68k`**, **`mips`**, **`mipsel`**, **`powerpc`**, **`s390`**, or **`sparc`**.

```
# /usr/sbin/debootstrap --arch ARCH sarge \
     /mnt/debinst http://http.us.debian.org/debian
```

C.4.4. Configure The Base System

Now you've got a real Debian system, though rather lean, on disk. **Chroot** into it:

```
# LANG= chroot /mnt/debinst /bin/bash
```

C.4.4.1. Mount Partitions

You need to create `/etc/fstab`.

```
# editor /etc/fstab
```

Here is a sample you can modify to suit:

```
# /etc/fstab: static file system information.
#
# file system    mount point   type    options                  dump pass
/dev/XXX         /             ext3    defaults                 0    1
/dev/XXX         /boot         ext3    ro,nosuid,nodev          0    2

/dev/XXX         none          swap    sw                       0    0
proc             /proc         proc    defaults                 0    0

/dev/fd0         /mnt/floppy   auto    noauto,rw,sync,user,exec 0    0
/dev/cdrom       /mnt/cdrom    iso9660 noauto,ro,user,exec      0    0

/dev/XXX         /tmp          ext3    rw,nosuid,nodev          0    2
/dev/XXX         /var          ext3    rw,nosuid,nodev          0    2
/dev/XXX         /usr          ext3    rw,nodev                 0    2
/dev/XXX         /home         ext3    rw,nosuid,nodev          0    2
```

Use **`mount -a`** to mount all the file systems you have specified in your `/etc/fstab`, or to mount file systems individually use:

```
# mount /path   # e.g.: mount /usr
```

You can mount the proc file system multiple times and to arbitrary locations, though `/proc` is customary. If you didn't use **`mount -a`**, be sure to mount proc before continuing:

```
# mount -t proc proc /proc
```

The command **`ls /proc`** should now show a non-empty directory. Should this fail, you may be able to mount proc from outside the chroot:

```
# mount -t proc proc /mnt/debinst/proc
```

C.4.4.2. Configure Keyboard

To configure your keyboard:

```
# dpkg-reconfigure console-data
```

Note that the keyboard cannot be set while in the chroot, but will be configured for the next reboot.

C.4.4.3. Configure Networking

To configure networking, edit `/etc/network/interfaces`, `/etc/resolv.conf`, and `/etc/hostname`.

```
# editor /etc/network/interfaces
```

Here are some simple examples from `/usr/share/doc/ifupdown/examples`:

```
######################################################################
# /etc/network/interfaces -- configuration file for ifup(8), ifdown(8)
# See the interfaces(5) manpage for information on what options are
# available.
######################################################################

# We always want the loopback interface.
#
auto lo
iface lo inet loopback

# To use dhcp:
#
# auto eth0
# iface eth0 inet dhcp

# An example static IP setup: (broadcast and gateway are optional)
#
# auto eth0
# iface eth0 inet static
#     address 192.168.0.42
#     network 192.168.0.0
#     netmask 255.255.255.0
#     broadcast 192.168.0.255
#     gateway 192.168.0.1
```

Enter your nameserver(s) and search directives in `/etc/resolv.conf`:

```
# editor /etc/resolv.conf
```

A simple `/etc/resolv.conf`:

```
search hqdom.local\000
nameserver 10.1.1.36
nameserver 192.168.9.100
```

Enter your system's host name (2 to 63 characters):

```
# echo DebianHostName > /etc/hostname
```

If you have multiple network cards, you should arrange the names of driver modules in the `/etc/modules` file into the desired order. Then during boot, each card will be associated with the interface name (eth0, eth1, etc.) that you expect.

C.4.4.4. Configure Timezone, Users, and APT

Set your timezone, add a normal user, and choose your **apt** sources by running

```
# /usr/sbin/base-config new
```

C.4.4.5. Configure Locales

To configure your locale settings to use a language other than English, install the locales support package and configure it:

```
# apt-get install locales
# dpkg-reconfigure locales
```

NOTE: Apt must be configured before, ie. during the base-config phase. Before using locales with character sets other than ASCII or latin1, please consult the appropriate localization HOWTO.

C.4.5. Install a Kernel

If you intend to boot this system, you probably want a Linux kernel and a boot loader. Identify available pre-packaged kernels with

```
# apt-cache search kernel-image
```

Then install your choice using its package name.

```
# apt-get install kernel-image-2.X.X-arch-etc
```

C.4.6. Set up the Boot Loader

To make your Debian GNU/Linux system bootable, set up your boot loader to load the installed kernel with your new root partition. Note that debootstrap does not install a boot loader, though you can use apt-get inside your Debian chroot to do so.

Check **`info grub`** or **`man lilo.conf`** for instructions on setting up the bootloader. If you are keeping the system you used to install Debian, just add an entry for the Debian install to your existing grub `menu.lst` or `lilo.conf`. For `lilo.conf`, you could also copy it to the new system and edit it there. After you are done editing, call lilo (remember it will use `lilo.conf` relative to the system you call it from).

Here is a basic `/etc/lilo.conf` as an example:

```
boot=/dev/hda6
root=/dev/hda6
install=menu
delay=20
```

```
lba32
image=/vmlinuz
label=Debian
```

C.5. Installing Debian GNU/Linux over Parallel Line IP (PLIP)

This section explains how to install Debian GNU/Linux on a computer without Ethernet card, but with just a remote gateway computer attached via a Null-Modem cable (also called Null-Printer cable). The gateway computer should be connected to a network that has a Debian mirror on it (e.g. to the Internet).

In the example in this appendix we will set up a PLIP connection using a gateway connected to the Internet over a dial-up connection (ppp0). We will use IP addresses 192.168.0.1 and 192.168.0.2 for the PLIP interfaces on the target system and the source system respectively (these addresses should be unused within your network address space).

The PLIP connection set up during the installation will also be available after the reboot into the installed system (see Chapter 7).

Before you start, you will need to check the BIOS configuration (IO base address and IRQ) for the parallel ports of both the source and target systems. The most common values are `io=0x378`, `irq=7`.

C.5.1. Requirements

- A target computer, called *target*, where Debian will be installed.
- System installation media; see Section 2.2.
- Another computer connected to the Internet, called *source*, that will function as the gateway.
- A DB-25 Null-Modem cable. See the PLIP-Install-HOWTO (http://www.tldp.org/HOWTO/PLIP-Install-HOWTO.html) for more information on this cable and instructions how to make your own.

C.5.2. Setting up source

The following shell script is a simple example of how to configure the source computer as a gateway to the Internet using ppp0.

```
#!/bin/sh

# We remove running modules from kernel to avoid conflicts and to
# reconfigure them manually.
modprobe -r lp parport_pc
modprobe parport_pc io=0x378 irq=7
modprobe plip

# Configure the plip interface (plip0 for me, see dmesg | grep plip)
ifconfig plip0 192.168.0.2 pointopoint 192.168.0.1 netmask 255.255.255.255 up
```

```
# Configure gateway
modprobe iptable_nat
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward
```

C.5.3. Installing target

Boot the installation media. The installation needs to be run in expert mode; enter **expert** at the boot prompt. Below are the answers that should be given during various stages of the installation.

1. Load installer components

   Select the **plip-modules** option from the list; this will make the PLIP drivers available to the installation system.

2. Detect network hardware

   - If target *does* have a network card, a list of driver modules for detected cards will be shown. If you want to force `debian-installer` to use plip instead, you have to deselect all listed driver modules. Obviously, if target doesn't have a network card, the installer will not show this list.
   - Prompt for module parameters: Yes
   - Because no network card was detected/selected earlier, the installer will ask you to select a network driver module from a list. Select the **plip** module.
   - Additional parameters for module parport_pc: ***io=0x378 irq=7***
   - Additional parameters for module plip: leave empty

3. Configure the network

   - Auto-configure network with DHCP: No
   - IP address: ***192.168.0.1***
   - Point-to-point address: ***192.168.0.2***
   - Name server addresses: you can enter the same addresses used on source (see `/etc/resolv.conf`)

Appendix D. Administrivia

D.1. About This Document

This manual was created for Sarge's debian-installer, based on the Woody installation manual for boot-floppies, which was based on earlier Debian installation manuals, and on the Progeny distribution manual which was released under GPL in 2003.

This document is written in DocBook XML. Output formats are generated by various programs using information from the `docbook-xml` and `docbook-xsl` packages.

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

D.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 `debian-installer-manual`. See the `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 debian-installer-manual (http://bugs.debian.org/debian-installer-manual) to see whether your problem has already been reported. If so, you can supply additional 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 DocBook source for this document, and produce patches against it. The DocBook source can be found at the debian-installer WebSVN (http://svn.debian.org/wsvn/d-i/). If you're not familiar with DocBook, don't worry: there is a simple cheatsheet in the manuals directory that will get you started. It's like html, but oriented towards the meaning of the text rather than the presentation. Patches submitted to the debian-boot mailing list (see below) are welcomed. For instructions on how to check out the sources via SVN, see README (http://svn.debian.org/wsvn/d-i/README?op=file) from the source root directory.

Please do *not* contact the authors of this document directly. There is also a discussion list for `debian-installer`, 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; or you can browse the Debian Mailing List Archives (http://lists.debian.org/) online.

D.3. Major Contributions

This document was originally written by Bruce Perens, Sven Rudolph, Igor Grobman, James Treacy, and Adam Di Carlo. Sebastian Ley wrote the Installation Howto. Many, many Debian users and developers contributed to this document. Particular note must be made of Michael Schmitz (m68k support), Frank Neumann (original author of the Amiga install manual (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. We have to thank Pascal

Le Bail for useful information about booting from USB memory sticks. Miroslav Kuře has documented a lot of the new functionality in Sarge's debian-installer.

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

The section on chrooted installations in this manual (Section C.4) was derived in part from documents copyright Karsten M. Self.

The section on installations over plip in this manual (Section C.5) was based on the PLIP-Install-HOWTO (http://www.tldp.org/HOWTO/PLIP-Install-HOWTO.html) by Gilles Lamiral.

D.4. Trademark Acknowledgement

All trademarks are property of their respective trademark owners.

Appendix E. GNU General Public License

Version 2, June 1991

E.1. Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the gnu General Public License is intended to guarantee your freedom to share and change free software — to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the gnu Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

E.2. GNU GENERAL PUBLIC LICENSE

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

- This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program",

below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

- You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

  You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

- You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

  a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

  b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

  c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

  These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

  Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

  In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

- You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the

following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

- You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
- You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
- Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
- If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

  If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other

circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

- If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
- The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
- If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

- because the program is licensed free of charge, there is no warranty for the program, to the extent permitted by applicable law. except when otherwise stated in writing the copyright holders and/or other parties provide the program "as is" without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. the entire risk as to the quality and performance of the program is with you. should the program prove defective, you assume the cost of all necessary servicing, repair or correction.
- in no event unless required by applicable law or agreed to in writing will any copyright holder, or any other party who may modify and/or redistribute the program as permitted above, be liable to you for damages, including any general, special, incidental or consequential damages arising out of the use or inability to use the program (including but not limited to loss of data or data being rendered inaccurate or losses sustained by you or third parties or a failure of the program to operate with any other programs), even if such holder or other party has been advised of the possibility of such damages.

END OF TERMS AND CONDITIONS

E.3. How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the

best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

one line to give the program's name and a brief idea of what it does.

Copyright (C) year name of author

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

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

You should have received a copy of the gnu General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author

Gnomovision comes with absolutely no warranty; for details type 'show w'.

This is free software, and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items — whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

signature of Ty Coon, 1 April 1989

Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the gnu Library General Public License instead of this License.