

Installing Debian GNU/Linux 2.1 For Motorola 680x0

Bruce Perens
Sven Rudolph
Igor Grobman
James Treacy
Adam Di Carlo

version 2.1.9.1, 07 March, 1999

Abstract

This document contains installation instructions for the Debian GNU/Linux 2.1 system, for the Motorola 680x0 ("m68k") architecture. It also contains pointers to more information and information on how to make the most of your new Debian system. The procedures in this document are *not* to be used for users upgrading existing systems; if you are upgrading, see the Debian 2.1 Release Notes (<http://www.debian.org/releases/2.1/m68k/release-notes/>).

Copyright Notice

This document may be distributed and modified under the terms of the GNU General Public License.

© 1996 Bruce Perens

© 1996, 1997 Sven Rudolph

© 1998 Igor Grobman, James Treacy

© 1998, 1999 Adam Di Carlo

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

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

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

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

Contents

- 1 Welcome to Debian** **1**
- 1.1 Getting the Newest Version of This Document 2
- 1.2 Organization of This Document 2
- 1.3 About Copyrights and Software Licenses 3

- 2 System Requirements** **5**
- 2.1 Supported Hardware 5
 - 2.1.1 Supported Architectures 5
 - 2.1.2 CPU, Mainboards, and Video Support 6
- 2.2 Installation Media 6
 - 2.2.1 Supported Storage Systems 7
- 2.3 Memory and Disk Space Requirements 7
- 2.4 Peripherals and Other Hardware 8
- 2.5 Purchasing Hardware Specifically for GNU/Linux 8
 - 2.5.1 Avoid Proprietary or Closed Hardware 8

- 3 Before You Start** **9**
- 3.1 Backups 9
- 3.2 Information You Will Need 9
- 3.3 Pre-installation Hardware and Operating System Setup 10
 - 3.3.1 Firmware Revisions and Existing OS Setup 10
 - 3.3.2 Over-Clocking your CPU 11
 - 3.3.3 Bad Memory Modules 11

4	Partitioning Your Hard Drive	12
4.1	Background	12
4.2	Planning Use of the System	13
4.3	Device Names in Linux	14
4.4	Recommended Partitioning Scheme	15
4.5	Example Partitioning	15
4.6	Partitioning Prior to Installation	16
4.6.1	Partitioning in AmigaOS	16
4.6.2	Partitioning in Atari TOS	17
4.6.3	Partitioning in MacOS	18
5	Methods for Installing Debian	19
5.1	Choosing Your Installation Media	20
5.1.1	Choosing Initial Boot Media	20
5.1.2	Choosing Media for Installing Base	20
5.2	Description of Installation System Files	21
5.3	Installing from a Hard Disk	22
5.3.1	Installing from AmigaOS	22
5.3.2	Installing from Atari TOS	23
5.3.3	Installing from MacOS	24
5.3.4	Installing from a Linux Partition	24
5.4	Installing from a CD-ROM	25
5.5	Booting from TFTP	25
5.6	Installing from NFS	26
5.7	Booting from Floppies	26
5.8	Installing Base from Floppies	27
5.9	Creating Floppies from Disk Images	27
5.9.1	Writing Disk Images From a Linux or Unix System	28
5.9.2	Writing Disk Images on Atari Systems	28

5.9.3	Writing Disk Images on Macintosh Systems	28
5.9.4	Writing Disk Images From DOS, Windows, or OS/2	29
5.9.5	Floppy Disk Reliability	29
6	Booting the Installation System	30
6.1	Boot Parameter Arguments	30
6.2	Booting With the Rescue Floppy	31
6.3	Interpreting the Kernel Startup Messages	31
6.4	Troubleshooting the Boot Process	32
7	Using <code>dbootstrap</code> for Initial System Configuration	33
7.1	Introduction to <code>dbootstrap</code>	33
7.2	"Select Color or Monochrome display"	34
7.3	"Debian GNU/Linux Installation Main Menu"	34
7.4	"Configure the Keyboard"	35
7.5	Last Chance!	35
7.6	"Partition a Hard Disk"	35
7.7	"Initialize and Activate a Swap Partition"	36
7.8	"Initialize a Linux Partition"	36
7.9	"Mount a Previously-Initialized Partition"	37
7.10	"Install Operating System Kernel and Modules"	37
7.11	"Configure Device Driver Modules"	38
7.12	"Configure the Network"	39
7.13	"Install the Base System"	39
7.14	"Configure the Base System"	40
7.15	"Make Linux Bootable Directly From Hard Disk"	40
7.16	The Moment of Truth	40
7.17	Set the Root Password	41
7.18	Create an Ordinary User	41

7.19	Shadow Password Support	41
7.20	Select and Install Profiles	42
7.21	Log In	43
7.22	Setting up PPP	43
7.23	Installing the Rest of Your System	44
8	Next Steps and Where to Go From Here	45
8.1	If You Are New to Unix	45
8.2	Orienting Yourself to Debian	45
8.3	Further Reading and Information	46
8.4	Compiling a New Kernel	46
8.5	Using the Linux 2.2 Kernel with Debian 2.1	48
9	Technical Information on the Boot Floppies	49
9.1	Source Code	49
9.2	Rescue Floppy	49
9.3	Replacing the Rescue Floppy Kernel	49
9.4	The Base Floppies	50
10	Administrivia	51
10.1	About This Document	51
10.2	Contributing to This Document	51
10.3	Major Contributions	52
10.4	Trademark Acknowledgement	52

Chapter 1

Welcome to Debian

We're delighted that you have decided to try Debian. We are sure that you will find that Debian is unique among operating system distributions. Debian brings together quality free software from around the world, integrating it into a coherent whole. The sum is truly more than the parts.

The Debian GNU/Linux distribution is made up of a number of software *packages*. Each package consists of executables, scripts, documentation, and configuration information. Each package has a *maintainer* who is responsible for that package. In this way, Debian grows *scalably*. Anyone who agrees to abide by the Debian Social Contract (http://www.debian.org/social_contract) may become a new maintainer. Any maintainer can introduce new software into Debian – provided it meets our criteria of being free, and the package follows our quality standards.

The Debian Free Software Guidelines (http://www.debian.org/social_contract\#guidelines) is a clear and concise statement of Debian's criteria for free software. It is a very influential document in the Free Software Movement, and provided the basis of the Open Source Free Software Guidelines (<http://opensource.org/osd.html>).

Only Debian has an extensive specification of our standards of quality, the Debian Policy (<http://www.debian.org/doc/debian-policy/>). This document defines the qualities and standards to which we hold Debian packages.

To protect your system against trojan horses and other malevolent software, Debian verifies that packages have come from their real Debian maintainers. Debian packagers also take great care to configure the packages in a secure manner. If security problems do arise with shipped packages, fixes are generally quickly available. Simply by updating your systems periodically, you will download and install security fixes.

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

1.1 Getting the Newest Version of This Document

This document is continually changing. Make sure to check Debian 2.1 pages (<http://www.debian.org/releases/2.1/>) for last minute information about the 2.1 release. Updated versions of this installation manual area also available on at Official Install Manual pages (<http://www.debian.org/releases/2.1/m68k/install>).

1.2 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 the level of expertise of the reader. However, general knowledge of how your hardware works is assumed.

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

In general, the document is arranged in linear fashion, walking the user through the installation process. Here are the steps, and the sections of this document which correlate with the steps.

1. Determine whether your hardware meets the requirements for using the installation system, in ‘System Requirements’ on page 5.
2. Backup your system, and perform any planning and hardware configuration prior to installing Debian, in ‘Before You Start’ on page 9.
3. Partition your hard disk as described in ‘Partitioning Your Hard Drive’ on page 12. Partitioning is very important, since you may have to live with it for a while.
4. In ‘Methods for Installing Debian’ on page 19, the different ways to install Debian are presented. Select and prepare your installation media accordingly.
5. Next, you shall boot the installation system. Information on this step is covered in ‘Booting the Installation System’ on page 30; this chapter also contains troubleshooting procedures in case you have a hard time booting.
6. Perform initial system configuration, which is discussed in ‘Using `dbootstrap` for Initial System Configuration’ on page 33, Sections ‘Introduction to `dbootstrap`’ on page 33 to “‘Configure the Network’” on page 39.
7. Install the base system, from “‘Install the Base System’” on page 39.
8. Boot into the newly installed base system and run through some post-base-installation tasks, from ‘The Moment of Truth’ on page 40.

9. Install the rest of the system, using `dselect`, in ‘Installing the Rest of Your System’ on page 44.

Once you’ve got your system installed, you can read ‘Next Steps and Where to Go From Here’ on page 45. This chapter explains where to look to find more information about Unix, Debian, and how to replace your kernel. In case you want to build your own install system from sources, take a look at ‘Technical Information on the Boot Floppies’ on page 49.

Finally, information about this document, and how to contribute to it, may be found in ‘Administrivia’ on page 51.

1.3 About Copyrights and Software Licenses

I’m sure you’ve read the licenses that come with most commercial software – they say you can only use one copy of the software on one computer. The Debian GNU/Linux system isn’t like that. We encourage you to put a copy on every computer in your school or place of business. Lend it to your friends, and help them install it on their computers. You can even make thousands of copies and *sell* them – with a few restrictions. That’s because Debian is based on *free software*.

Free software doesn’t mean that it doesn’t have a copyright, and it doesn’t mean that the CD you buy containing this software is 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 the programs. It also means that anyone may extend, adapt, and modify the software, and distribute the results of their work as well.¹

Many of the programs in the system are licensed under the *GNU General Public License*, or *GPL*. The GPL requires that you make the *source code* of the programs available whenever you distribute a copy of the program; that ensures that you, the user, are able to modify the software. Thus, we’ve included the source code for all of those programs in the Debian system.² There are several other forms of copyright and software license used on the programs in Debian. You can find the copyrights and licenses of every program by looking in the file `/usr/doc/{package--name}/copyright` once you’ve installed your system.

For more information on licenses and how Debian decides what 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

¹Note that we do make available many packages which do not meet our criteria of being free. These are distributed in the `contrib` area or the `non-free` area; see the Debian FAQ (<http://www.debian.org/doc/FAQ/>), under “The Debian FTP archives”.

²For information on how to locate and unpack Debian source packages, see the Debian FAQ (<http://www.debian.org/doc/FAQ/>).

to modify software to suit your needs as needed – and enjoy the benefits of others who have extended the software in this way.

Chapter 2

System Requirements

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

2.1 Supported Hardware

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

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

Rather than attempting to describe all the different hardware configurations which are supported for Motorola 680x0, this section contains general information and pointers to where additional information can be found.

2.1.1 Supported Architectures

Debian 2.1 supports four architectures: Intel x86-based architectures; Motorola 680x0 machines such as Atari, Amiga, and Macintoshes; DEC Alpha machines; and Sun SPARC machines. These are referred to as *i386*, *m68k*, *alpha*, and *sparc*, respectively.

This document covers installation for the *m68k* architecture. Separate versions of this document exist for other architectures.

2.1.2 CPU, Mainboards, and Video Support

Complete information concerning supported M68000 based (*m68k*) systems can be found at the Linux/m68k FAQ (<http://www.linux-m68k.org/faq/faq.html>). This section merely outlines the basics.

The m68k port of Linux runs on any 680x0 with a PMMU (Paged Memory Management Unit) and a FPU (floating-point unit). This includes the 68020 with an external 68851 PMMU, the 68030, and better, and excludes the "EC" line of 680x0 processors. See the Linux/m68k FAQ (<http://www.linux-m68k.org/faq/faq.html>) for complete details.

There are four major flavors of supported *m68k* flavors: Amiga, Atari, Macintosh and VME machines. Amiga and Atari were the first two systems to which Linux was ported; in keeping, they are also the two most well-supported Debian ports. The Macintosh line is supported incompletely, both by Debian and by the Linux kernel; see Linux m68k for Macintosh (<http://www.mac.linux-m68k.org/>) for project status and supported hardware. The BVM and Motorola single board VMEbus computers are the most recent addition to the list of machines supported by Debian. Ports to other m68k architectures, such as the Sun3 architecture and NeXT black box, are underway but not yet supported by Debian.

2.2 Installation Media

There are four different media which can be used to install Debian: floppies, CD-ROMs, local disk partitions, or the network. Different parts of the same Debian installation can mix and match these options; we'll go into that in 'Methods for Installing Debian' on page 19.

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

Low-density installation floppies (720 k) are also provided for Ataris.

CD-ROM based installation is also supported for some architectures. On machines which support bootable CD-ROMs, you should be able to do a completely floppy-less installation. Even if your system doesn't support booting from a CD-ROM, you can use the CD-ROM in conjunction with the other techniques to install your system, once you've booted up by other means; see 'Installing from a CD-ROM' on page 25.

Installation from local disk is another option. If you have free space on partitions other than the partitions you're installing to, this is definitely a good option. Some platforms even have local installers, i.e., for booting from AmigaOS, TOS, or MacOS. In fact, installation from your local disk is the preferred installation technique for m68k.

The last option is network installation. You can install your system via NFS. Diskless installation, using NFS-mounting of all local filesystems, is another option. You also *boot* your system over the network. After your base system is installed, you can install the rest of your system via any sort of network connection (including PPP), via FTP, HTTP, or NFS.

More complete descriptions of these methods, and helpful hints for picking which method is best for you, can be found in ‘Methods for Installing Debian’ on page 19. Please be sure to continue reading to make sure the device you intend to boot and install from is supported by the Debian installation system.

2.2.1 Supported Storage Systems

The Debian boot disks contain a kernel which is built to maximize the number of systems it runs on. Unfortunately, this makes for a larger kernel, with a lot of drivers which will never be used (see ‘Compiling a New Kernel’ on page 46 to learn how to build your own). However, support for the widest possible range of devices is desirable in order to ensure that Debian can be installed on the widest array of hardware.

Pretty much all storage systems supported by the Linux kernel are supported by the Debian installation system. Note that the current Linux kernel does not support floppies on the Macintosh at all, and the Debian installation system doesn’t support floppies for Amigas. Also supported on the Atari is the Macintosh HFS system, and AFFS as a module. Macs support the Atari (FAT) filesystem. Amiga support the FAT filesystem, and HFS as a module.

2.3 Memory and Disk Space Requirements

You must have at least 5MB of memory and 35MB of hard disk. If you want to install a reasonable amount of software, including the X Window System, and some development programs and libraries, you’ll need at least 300MB. For a more or less complete installation, you’ll need around 800MB. To install *everything* available in Debian, you’ll probably need around 2 GB. Actually, installing everything doesn’t even make sense, since some packages conflict with others.

On the Amiga the size of FastRAM is relevant towards the total memory requirements. Also, using a GVP (or “Zorro”) card with 16-bit RAM is not supported; you’ll need 32-bit RAM. The `amiboot` program can be used to disable 16-bit RAM; see the Linux/m68k FAQ (<http://www.linux-m68k.org/faq/faq.html>).

On the Atari, both ST-RAM and Fast RAM (TT-RAM) are used by Linux. Many users have reported problems running the kernel itself in Fast RAM, so the Atari bootstrap will place the kernel in ST-RAM. The minimum requirement for ST-RAM is 2 MB.

On the Macintosh, care should be taken on machines with RAM-based video (RBV). The RAM segment at physical address 0 is used as screen memory, making the default load position for the kernel unavailable. The alternate RAM segment used for kernel and ramdisk must be at least 4 MB.

2.4 Peripherals and Other Hardware

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

Any network interface card (NIC) supported by the Linux kernel should also be supported by the boot disks. You may need to load your network driver as a module. Again, see Linux/m68k FAQ (<http://www.linux-m68k.org/faq/faq.html>) for complete details.

2.5 Purchasing Hardware Specifically for GNU/Linux

There are several vendors, now, who ship systems with Debian or other distributions of GNU/Linux pre-installed. You might pay more for the privilege, but it does buy a level of peace of mind, since you can be sure that the hardware is well-supported by GNU/Linux. Unfortunately, it's quite rare to find any vendor shipping new Motorola 680x0 machines at all.

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

2.5.1 Avoid Proprietary or Closed Hardware

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

In fact, no specifications or documentation have ever been released for any Macintosh hardware, most notably the ADB controller (used by the mouse and keyboard), the floppy controller, and all acceleration and CLUT manipulation of the video hardware. In a nutshell, this explains why the Macintosh Linux port lags behind other Linux ports.

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

Chapter 3

Before You Start

3.1 Backups

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

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

Since the only supported installation method for m68k systems is booting from a local disk or floppy using an AmigaOS/TOS/MacOS-based bootstrap, you will need the original operating system in order to boot Linux.

3.2 Information You Will Need

Besides this document, you'll need

the `atari-fdisk` (`atari-fdisk.txt`) manual page,

the `amiga-fdisk` (`amiga-fdisk.txt`) manual page,

the `mac-fdisk` (`mac-fdisk.txt`) manual page,

the `pmac-fdisk` (`pmac-fdisk.txt`) manual page,

the `dselect` Tutorial (`dselect-beginner.html`), and the

Linux/m68k FAQ (<http://www.linux-m68k.org/faq/faq.html>).

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

- Your host name (you may be able to decide this on your own).
- Your domain name.
- Your computer’s IP address.
- The IP address of your network.
- The netmask to use with your network.
- The broadcast address to use on your network.
- The IP address of the default gateway system you should route to, if your network *has* a gateway.
- The system on your network that you should use as a DNS (Domain Name Service) server.
- Whether you connect to the network using Ethernet.

If your computer’s only network connection is via a serial line, using PPP or an equivalent dialup connection, you are probably not installing the base system over a network. You don’t need to worry about getting your network setup until your system is already installed. See ‘Setting up PPP’ on page 43 below for information on setting up PPP under Debian.

3.3 Pre-installation Hardware and Operating System Setup

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

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

3.3.1 Firmware Revisions and Existing OS Setup

Motorola 680x0 machine are generally self-configuring and do not require firmware configuration. However, you should make sure that you have the appropriate ROM and system patches. On the Macintosh, MacOS version ≥ 7.1 is recommended because version 7.0.1 contains a bug in the video drivers preventing the booter from deactivating the video interrupts, resulting in a boot hang. The Amiga bootstrap requires `ixemul.library`, a version of which is distributed on the CD-ROM.

3.3.2 Over-Clocking your CPU

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

3.3.3 Bad Memory Modules

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

Atari TT RAM boards are notorious for RAM problems under Linux; if you encounter any strange problems, try running at least the kernel in ST-RAM. Amiga users may need to exclude RAM using a booter memfile.

Chapter 4

Partitioning Your Hard Drive

4.1 Background

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

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

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

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

The second reason is generally more important in a business setting, but it really depends on your use of the machine. Suppose something runs out of control and starts eating disk space. If the process causing the

problem happens to have root privileges (the system keeps a percentage of the disk away from users), you could suddenly find yourself out of disk space. This is not good as the OS needs to use real files (besides swap space) for many things. It may not even be a problem of local origin. For example, getting spammed with e-mail can easily fill a partition. By using more partitions, you protect the system from many of these problems. Using mail as an example again, by putting `/var/spool/mail` on its own partition, the bulk of the system will work even if you get spammed.

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

4.2 Planning Use of the System

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

There are a number of default "Profiles" which Debian offers for your convenience (see 'Select and Install Profiles' on page 42). Profiles are simply sets of package selections which make it easier for you, in that a number of packages are automatically marked for installation.

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

The following are some of the available profiles and their sizes:

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

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

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

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

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

4.3 Device Names in Linux

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

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

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

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

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

4.4 Recommended Partitioning Scheme

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

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

For very complex systems, you should see the Multi Disk HOWTO (<http://metalab.unc.edu/LDP/HOWTO/Multi-Disk-HOWTO.html>). This contains in-depth information, mostly of interest to ISPs and people setting up servers.

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

Note that Linux for your architecture will not use more than 128 megabytes of swap on a single swap partition. However, you can make multiple swap partitions by hand and edit `/etc/fstab` after you've installed to get more than 128 megabytes of swap. If your swap requirements are this high, however, you should probably try to spread the swap across different disks (also called "spindles"). Or you can try the more recent Linux kernels (2.2 and higher) where this limitation was relaxed (be careful, it may require other changes in your system).

4.5 Example Partitioning

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

4.6 Partitioning Prior to Installation

There are two different times that you can partition: prior to the installation of Debian, or during installation of Debian. If your computer will be solely dedicated to Debian, you should partition as part of the boot process ("Partition a Hard Disk" on page 35). If you have a machine with more than one operating system on it, you generally should let the native operating system create its own partitions.

The following sections contain information regarding partitioning in your native operating system prior to installation. Note that you'll have to map between how the other operating system names partitions, and how Linux names partitions; see 'Device Names in Linux' on page 14.

4.6.1 Partitioning in AmigaOS

If you are running AmigaOS, you can use the `HDToolBox` program to partition your disk prior to installation. Here's how:

1. Start `HDToolBox`, select the disk you want to use, click on the "Partition Drive" button and select or create the partition you want to use as the Debian root filesystem.
2. Next, you need to enable the "Advanced options" and change the following items under "Change":
 - set the filesystem to "Custom Filesystem" or "Reserved Filesystem" (the label which is shown depends on version of `HDToolBox` you have installed)
 - set the identifier to `0x4c4e5800` (this is the hexadecimal equivalent of "LNX\0")
 - disable the "Auto-mount this partition" checkbox
 - disable "Custom Bootcode"
 - set the "Reserved blocks at" settings to 2 for start and 0 for end
3. If you are making more than one Linux partition, go ahead and create the additional partitions, just as above.
4. After having done this, select a partition that is to be used as a swap partition, and repeat the same steps as above, but set the identifier to `0x53575000` instead (this represents "SWP\0" in ASCII).
5. Write down the *Linux* partition names for the root and swap filesystems you just created. See 'Device Names in Linux' on page 14 for more information on Linux partition naming.
6. Go back to the main window of `HDToolBox` and select "Save changes to drive". Think twice before actually clicking on "Yes" – have you chosen the correct partitions? No important data could get lost now if you made a mistake? Then click "OK". If required, the Amiga will reboot after this.

4.6.2 Partitioning in Atari TOS

Atari partition IDs are three ASCII characters, use "LNK" for data and "SWP" for swap partitions. If using the low memory installation method, a small Minix partition is also needed (about 2 MB), for which the partition ID is "MNX". Failure to set the appropriate partition IDs not only prevents the Debian installation process from recognizing the partitions, but also results in TOS attempting to use the Linux partitions, which confuses the harddisk driver and renders the whole disk inaccessible.

There are a multitude of third party partitioning tools available (the Atari `harddisk` utility doesn't permit changing the partition ID); this manual cannot give detailed descriptions for all of them. The following description covers `SCSITool` (from Hard+Soft GmBH).

1. Start `SCSITool` and select the disk you want to partition ("Disk" menu, item "select").
2. From the "Partition" menu, select either "New" to add new partitions or change the existing partition sizes, or "Change" to change one specific partition. Unless you have already created partitions with the right sizes and only want to change the partition ID, "New" is probably the best choice.
3. For the "New" choice, select "existing" in the dialog box prompting the initial settings. The next window shows a list of existing partitions which you can adjust using the scroll buttons, or by clicking in the bar graphs. The first column in the partition list is the partition type; just click on the text field to edit it. When you are finished changing partition settings, save the changes by leaving the window with the "Ok" button.

For the "Change" option, select the partition to change in the selection list, and select "other systems" in the dialog box. The next window lists detailed information about the location of this partition, and lets you change the partition ID. Save changes by leaving the window with the "Ok" button.

4. Write down the Linux names for each of the partitions you created or changed for use with Linux – see 'Device Names in Linux' on page 14.
5. Quit `SCSITool` using the "Quit" item from the "File" menu. The computer will reboot to make sure the changed partition table is used by TOS. If you changed any TOS/GEM partitions, they will be invalidated and have to be reinitialized (we told you to back up everything on the disk, didn't we?).

There is a partitioning tool for Linux/m68k called `atari-fdisk` in the installation system, but for now we recommend you partition your disk using a TOS partition editor or some disk tool. If your partition editor doesn't have an option to edit the partition type, you can do this crucial step at a later stage (from the booted temporary install ramdisk). `SCSITool` is only one of the partition editors we know of which supports selection of arbitrary partition types. There may be others; select the tool that suits your needs.

4.6.3 Partitioning in MacOS

Partitioning tools for Macintosh tested include HD SC Setup 7.3.5 (Apple), HDT 1.8 (FWB), SilverLining (LaCie), and DiskTool (Tim Endres, GPL). Full versions are required for HDT and SilverLining. The Apple tool requires a patch in order to recognize third-party disks (a description on how to patch HD SC Setup using ResEdit can be found at <http://www.euronet.nl/users/ernstoud/patch.html>).

The following recipe is for partition with Apple's HD SC Setup.

Whatever tool you use, the partition type has to be set to "Apple_Unix_SVR2". The partition names need to be "A/UX Root", "A/UX Root&Usr" or "A/UX Usr" for data partitions; and "A/UX swap" for swap partitions. HD SC Setup will use the right names and type when creating A/UX partitions in a "Custom" partition scheme. Partitions are selected for deletion, creation or resizing using the mouse, the partition name and type can be selected from a list of predefined types. DiskTool can create A/UX type partitions but requires that the user type in the partition names manually. Descriptions for other tools are welcome.

Chapter 5

Methods for Installing Debian

As you initially install Debian, there are several steps that you shall undergo, in order:

1. booting the installation system
2. initial system configuration
3. installing the base system
4. booting the newly installed base system
5. installing the rest of the system

Booting the Debian installation system, the first step, is generally done with the Rescue Floppy or from the CD-ROM.

The first boot is sometimes the hardest, depending on your hardware. Therefore, it is described in 'Booting the Installation System' on page 30.

Once you've booted into Linux, the `dbootstrap` program will launch and guide you through the second step, the initial system configuration. This step is described in detail in 'Using `dbootstrap` for Initial System Configuration' on page 33.

The "Debian base system" is a core set of packages which are required to run Debian in a minimal, stand-alone fashion. Once you have configured and installed the base system, your machine can "stand on its own". The Debian base system can be installed from the following media: floppies, hard disk, CD-ROM, or from an NFS server. `dbootstrap` will perform this installation; it is described in "Install the Base System" on page 39.

The final step is the installation of the remainder of the Debian system. This would include the applications and documents that you actually use on your computer, such as the X Window System, editors, shells, and

development environments. The rest of the Debian system can be installed from CD-ROM or any mirror of the Debian archive (on or off the Internet, via HTTP, FTP, or NFS). At this point, you'll be using the standard Debian package management tools, such as `dselect` or `apt-get`. This step is described in 'Installing the Rest of Your System' on page 44.

Note that the media you use for one step and the media used for another step do *not* need to be the same. That is, you can boot from the Rescue Floppy, install the base system from NFS, and then install the remainder of the system from CD-ROM. If you're downloading the system from the archive, you'll generally boot and install the base system from floppies, installing the complete Debian system from the Internet.

The installation system, which is required for the first three installation steps, are divided into three parts: the "Rescue Floppy", the "Drivers Floppy", and the "Base System". Below you will find a description of the different installation methods, and a description of files which might be required for installation. Which files you use, and what steps you have to take to prepare your installation media, will vary with the method that you select to install Debian.

5.1 Choosing Your Installation Media

First, choose the media to use to boot the installation system. Next, choose the method you will use to install the base system.

5.1.1 Choosing Initial Boot Media

To boot the installation system, you have the following choices: floppies, network boot (TFTP), or a non-Linux boot loader.

Booting from floppies is supported for most platforms. Amigas and Macs are an exception to this rule, unfortunately. Floppy booting is described in 'Booting from Floppies' on page 26. For most m68k architectures, booting from a local filesystem is the recommended method.

Booting from the network requires that you have a TFTP server, a RARP server, and a network connection supported by the boot floppies. This installation method is described in 'Booting from TFTP' on page 25.

Booting from an existing operating system is often a convenient option; for some systems it is the only supported method of installation. This method is described in 'Installing from a Hard Disk' on page 22.

5.1.2 Choosing Media for Installing Base

The base system can be installed in the following ways: from floppies ('Installing Base from Floppies' on page 27), from a CD-ROM ('Installing from a CD-ROM' on page 25), from an NFS server ('Installing from NFS' on page 26), or from a local hard disk ('Installing from a Hard Disk' on page 22). You should choose whatever method matches the media you have, and whatever is the most convenient.

5.2 Description of Installation System Files

This section contains an annotated list of files you will find in the `disks--m68k` directory. You may not need to download these at all; it all depends on the booting and base system installation media you have chosen.

Most files are floppy disk images; that is, a single file which can be written to a disk to create the necessary floppy disk. These images are, obviously, dependent on the size of the target floppy, such as 1.4MB, 1.2MB, or 720KB. Which sizes are available depends on your platform (i.e., 720KB drives are Atari-specific). The images for 1.4MB drives have '14' embedded in their filenames, 1.2MB images have '12' somewhere in their filename, 720KB drives have '72' in their filename.

If you are using a web browser on a networked computer to read this document, you can probably retrieve the files by selecting their names in your web browser. Depending on your browser you may need to take special action to download directly to a file, in raw binary mode. For example, in Netscape you need to hold the shift key when clicking on the URL to retrieve the file. Files can be downloaded from the URLs in this document, or you can retrieve them from `ftp://ftp.debian.org/debian/dists/slink/main/disks-m68k/current/`, or the corresponding directory on any of the Debian mirror sites (`http://www.debian.org/distrib/ftplist`).

`amiga/resc1440.bin, atari/resc1440.bin, atari/resc720.bin, mac/resc1440.bin bvme6000/resc`

These are the Rescue Floppy disk images. The Rescue Floppy is used for initial setup and for emergencies, such as when your system doesn't boot for some reason. Therefore it is recommended you write the disk image to the floppy even if you are not using floppies for installation.

If you have a low-density drive on an Atari, you can use `atari/resc720.bin`.

`amiga/drv1440.bin, atari/drv1440.bin, atari/drv720.bin, mac/drv1440.bin bvme6000/drv1440`

These are the Drivers Floppy disk images. They contain the kernel modules, or drivers, for all kinds of hardware that are not necessary for initial booting. You will be prompted to choose the drivers you need during the installation process.

If you used a special Rescue Floppy image, you need to use the corresponding Drivers Floppy image.

`common/base2_1.tgz (recommended), or common/base14-1.bin, common/base14-2.bin, common/base`

These files contain the base system which will be installed on your Linux partition during the installation process. This is the bare minimum necessary for you to be able to install the rest of the packages. The `common/base2_1.tgz` file is for installation from non-floppy media, i.e., CD-ROM, hard-disk, or NFS.

`amiga/amigainstall.lha (Amiga), atari/install.lzh (Atari), or mac/Install.sit.hqx (Mac) – Open`

Files you uncompress on your local disk in your pre-existing operating system. They contain parts of the Debian installation process.

amiga/rootamiga.bin, atari/root.bin, mac/root.bin, bvme6000/root.bin, mvme162/root.bin, r

This file contains an image of a temporary filesystem that gets loaded into memory when you boot.

This is used for installations from hard disk and from CD-ROM.

tftpboot.img – TFTP boot image Boot image used for network booting, see ‘Booting from TFTP’ on page 25.

install.txt, install.html – Installation Manual This file you are now reading, in plain ASCII or HTML format.

amiga/install.txt, atari/install.txt, mac/install.txt – Install Guide Quick reference describing the installation on the corresponding systems step by step, like a condensed version of sections 5 - 7 of this manual.

atari-fdisk.txt amiga-fdisk.txt mac-fdisk.txt pmac-fdisk.txt Instructions for using your available partitioning programs.

basecont.txt Listing of the contents of the base system.

md5sum.txt List of MD5 checksums for the binary files. If you have the md5sum program, you can ensure that your files are not corrupt by running `md5sum -v -c md5sum.txt`.

5.3 Installing from a Hard Disk

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

5.3.1 Installing from AmigaOS

Use the following steps to install Debian from your pre-existing AmigaOS setup.

1. Get the files `amiga/amigainstall.lha` and `common/base2_1.tgz`.
2. Unpack `amigainstall.lha` into a partition with at least 10MB free. We recommend you unpack it into the main directory.
3. After unpacking, you should have a `debian` directory. Move `common/base2_1.tgz` into that same `debian` directory. Do not rename any files in this directory.
4. Write down the Linux partition name for the location where your new `debian` directory is. See ‘Device Names in Linux’ on page 14 for more information on Linux partition naming.
5. Prepare your partitions for Linux. See ‘Partitioning Prior to Installation’ on page 16.

6. In the `Workbench`, start the Linux installation process by double-clicking on the "`StartInstall`" icon in the `debian` directory.

You may have to press the *Return* key twice after the Amiga installer program has output some debugging information into a window. After this, the screen will go grey, there will be a few seconds' delay. Next, a black screen with white text should come up, displaying all kinds of kernel debugging information. These messages may scroll by too fast for you to read, but that's OK. After a couple of seconds, the installation program should start automatically, so you can continue down at 'Using `dbootstrap` for Initial System Configuration' on page 33.

If, on the other hand, you have problems booting, see 'Troubleshooting the Boot Process' on page 32.

5.3.2 Installing from Atari TOS

Use the following steps to install Debian from your pre-existing Atari TOS setup.

1. Get the files `atari/install.lzh` and `common/base2_1.tgz`.
2. Unpack `install.lzh` into a partition with at least 10 MB free. We recommend you unpack it into the "`main`" directory.
3. After unpacking, you should have a `debian` directory. Move `common/base2_1.tgz` into that same `debian` directory. Do not rename any files in this directory.
4. Write down the Linux partition name for the location where your new `debian` directory is. See 'Device Names in Linux' on page 14 for more information on Linux partition naming.
5. Prepare your partitions for Linux, if you haven't already done so. See 'Partitioning Prior to Installation' on page 16.
6. At the GEM desktop, start the Linux installation process by double-clicking on the "`bootstra.ttp`" icon in the `debian` directory and clicking "`Ok`" at the program options dialog box.

You may have to press the *Return* key after the Atari bootstrap program has output some debugging information into a window. After this, the screen will go grey, there will be a few seconds' delay. Next, a black screen with white text should come up, displaying all kinds of kernel debugging information. These messages may scroll by too fast for you to read, but that's OK. After a couple of seconds, the installation program should start automatically, so you can continue below at 'Using `dbootstrap` for Initial System Configuration' on page 33.

If, on the other hand, you have problems booting, see 'Troubleshooting the Boot Process' on page 32.

5.3.3 Installing from MacOS

Use the following steps to install Debian from your pre-existing MacOS setup.

1. Get the files `mac/Install.sit.hqx` and `common/base2_1.tgz`.
2. Unpack `Install.sit.hqx` into a partition with at least 10 MB free. We recommend you unpack it into the top-level directory of a volume with sufficient space.
3. After unpacking, you should have a `debian` directory. Move `common/base2_1.tgz` into that same `debian` directory. Do not rename any files in this directory.
4. Write down the Linux partition name for the location where your new `debian` directory is. See ‘Device Names in Linux’ on page 14 for more information on Linux partition naming.
5. Prepare your partitions for Linux, if you haven’t already done so. See ‘Partitioning Prior to Installation’ on page 16.
6. At the MacOS desktop, start the Linux installation process by double-clicking on the “Penguin Prefs” icon in the `debian` directory. The Linux booter will start up. Go to the “Settings” item in the “File” menu and select the kernel and ramdisk images in the `debian` directory by clicking on the corresponding buttons in the upper right corner, and navigating the file select dialogs to locate the files. Close the “Settings” dialog, save the settings and start the bootstrap using the “Boot Now” item in the “File” menu.

The `Penguin booter` will output some debugging information into a window. After this, the screen will go grey, there will be a few seconds’ delay. Next, a black screen with white text should come up, displaying all kinds of kernel debugging information. These messages may scroll by too fast for you to read, but that’s OK. After a couple of seconds, the installation program should start automatically, so you can continue below at ‘Using `dbootstrap` for Initial System Configuration’ on page 33.

If, on the other hand, you have problems booting, see ‘Troubleshooting the Boot Process’ on page 32.

5.3.4 Installing from a Linux Partition

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

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

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

1. Get the following files and place them in a directory on your Linux partition. Use the largest possible files for your architecture:

- a Rescue Floppy image
 - a Drivers Floppy image
 - `common/base2_1.tgz`
2. You can use any other functional boot method when installing from a partition. The following assumes you are booting with floppies; however, any boot installation can be used.
 3. Create the Rescue Floppy as discussed in ‘Creating Floppies from Disk Images’ on page 27. Note that you won’t need the Drivers Floppy.
 4. Insert the Rescue Floppy into your floppy drive, and reboot the computer.
 5. Skip down to ‘Booting the Installation System’ on page 30.

5.4 Installing from a CD-ROM

However you decide to boot, you can install the base Debian system from the CD-ROM. Simply boot using one of the other installation techniques; when it is time to install the base system and any additional packages, just point your installation system at the CD-ROM drive as described in “‘Install the Base System’” on page 39.

5.5 Booting from TFTP

You need to setup two servers: a RARP server and a TFTP server. The Reverse Address Resolution Protocol (RARP) is how your client will figure out what IP address to use; 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.

To setup RARP, you need to know the ethernet address of the client (a.k.a. the MAC address). If you don’t know this information, you can boot into “‘Rescue’” mode (e.g., from the Rescue Floppy) and use the command `/sbin/ifconfig eth0`.

In GNU/Linux you need to populate the kernel’s RARP table. To do this execute

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

Under SunOS, you need to ensure that the ethernet hardware address for the client is listed in the “‘ethers’” database (either in the `/etc/ethers` file, or via NIS/NIS+) and in the “‘hosts’” database. Then you need

to start the RARP daemon. In SunOS 4, issue the command (as root): `/usr/etc/rarpd -a`; in SunOS 5, use `/usr/sbin/rarpd -a`.

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

```
tftp dgram udp wait root /usr/etc/in.tftpd in.tftpd -l /boot
```

Look in that file and remember the directory which is used as the argument of `in.tftpd`; you'll need that below. The `-l` argument enables some versions of `in.tftpd` to log all requests to the system logs; this is useful for diagnosing boot errors. If you've had to change `/etc/inetd.conf`, you'll have to notify the running `inetd` process that the file has changed. On a Debian machine, run `/etc/init.d/netbase reload`; on other machines, find out the process ID for `inetd`, and run `kill -1 inetd-pid`.

Next, place the TFTP boot image, `tftpboot.img`, in the `tftpd` boot image directory. Generally, this directory will be `/boot` in Debian, and `/tftpboot` in other operating systems. Then, you'll have to make a link from that file to the file which `tftpd` will use for booting a particular client. The form of the file that `tftpd` will look for is `\textit{client--ip--in--hex}.\textit{client--architecture}`. To compute *client-ip-in-hex*, take each byte of the client IP address and translate it into hexadecimal notation. If you have a machine handy with the `bc` program, you can use the program. First issue the `obase=16` command to set the output to hex, then enter the individual components of the client IP one at a time. As for *client-architecture*, try out some values. Once you've determined the name, make the link like this: `ln /boot/tftpboot.img /boot/file-name`.

Now you should be ready to actually boot your system.

5.6 Installing from NFS

Due to the nature of this method of installation, only the base system can be installed via NFS. You will need to have the rescue disk and the driver disk available locally using one of the above methods. To install the base system via NFS, you'll have to go through the regular installation as explained in 'Using `dbootstrap` for Initial System Configuration' on page 33. Do not forget to insert the module (driver) for your ethernet card, and the file system module for NFS.

When `dbootstrap` asks you where the base system is located ("Install the Base System" on page 39), you should choose NFS, and follow the instructions.

5.7 Booting from Floppies

Booting from floppies is a simple process. Simply download the Rescue Floppy image and the Drivers Floppy image. Copy these to floppies as described in 'Creating Floppies from Disk Images' on the current

page. If you need to, you can also modify the Rescue Floppy; see ‘Replacing the Rescue Floppy Kernel’ on page 49.

Booting from the Rescue Floppy is supported only for Atari and VME (with a SCSI floppy drive on VME) at this time. On Macintosh, you can boot from the HFS floppy image supplied as a DiskCopy format image, which is a raw disk image containing the Rescue Floppy image.

5.8 Installing Base from Floppies

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

Installing the base system from floppies is not supported on Amiga and Macintosh systems.

Complete these steps:

1. Obtain these disk images (these files are described in greater detail in ‘Description of Installation System Files’ on page 21):
 - a Rescue Floppy image
 - a Drivers Floppy image
 - the base system disk images, i.e., `base14-1.bin`, `base14-2.bin`, etc.
2. Locate sufficient floppies for all the images you need to write.
3. Create the floppies, as discussed in ‘Creating Floppies from Disk Images’ on this page.
4. Insert the Rescue Floppy into your floppy drive, and reboot the computer.
5. Skip down to ‘Booting the Installation System’ on page 30.

5.9 Creating Floppies from Disk Images

Disk images are files containing the complete contents of a floppy disk in *raw* form. Disk images, such as `resc1440.bin`, cannot simply be copied to floppy drives. A special program is used to write the image files to floppy disk in *raw* mode. This is required because these images are raw representations of the disk; it is required to do a *sector copy* of the data from the file onto the floppy.

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

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

5.9.1 Writing Disk Images From a Linux or Unix System

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

```
dd if=file of=/dev/fd0 bs=512 conv=sync ; sync
```

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

Some systems attempt to automatically mount a floppy disk when you place it in the drive. You might have to disable this feature before the workstation will allow you to write a floppy in *raw mode*. Unfortunately, how to accomplish this will vary based on your operating system. On Solaris, make sure `vold` isn't running. On other systems, ask your system administrator.

5.9.2 Writing Disk Images on Atari Systems

You'll find the `atari/rawwrite.ttp` program in the same directory as the floppy disk images. Start the program by double clicking on the program icon, and type in the name of the floppy image file you want written to the floppy at the TOS program command line dialog box.

5.9.3 Writing Disk Images on Macintosh Systems

Using `DiskCopy` (version 4.2 or later), you can create a MacOS floppy from the `mac/Debian-m68k-2.1-Mac.img` file in the same directory as the Macintosh installer files. Start `DiskCopy` and select the "Make a Floppy" option in the "Utilities" menu. Select the disk image file in the file select dialog.

There is no MacOS application to write the `mac/rescl440.bin` and `mac/drv1440.bin` images to floppy disks (and there would be no point in doing this as you can't use these floppies to boot the installation system or install kernel and modules from on Macintosh). However, these files are needed for the installation of the operating system and modules, later in the process.

Be careful whenever transferring files on the Macintosh. Files with the suffix `.bin` or `.tgz` always need to be transferred using binary mode.

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

If you have access to a PC running one of these systems – we might never like to admit it, but these do exist – you can use it to write the disk images.

You'll find the `rawrite2.exe` program in the `i386` section of a Debian archive, in the same directory as the floppy disk images. There's also a `rawrite2.txt` file containing instructions for using `rawrite2`.

5.9.5 Floppy Disk Reliability

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

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

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

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

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

Chapter 6

Booting the Installation System

You have already chosen your boot system in the previous chapter. This could be booting off the Rescue Floppy, booting from the network, or booting from a pre-installed operating system. This chapter describes some of the ways booting can be controlled, common problems which occur during booting, and some ways to work around them, or at least to help us diagnose the problems.

6.1 Boot Parameter Arguments

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

If you are booting from the Rescue Floppy you will be presented with the boot prompt, `boot:`. Details about how to use boot parameters with the Rescue Floppy can be found in 'Booting With the Rescue Floppy' on the following page. If you are booting from an existing operating system, you'll have to use other means to set boot parameters. Full information on boot parameters can be found in the Linux BootPrompt HOWTO (<http://metalab.unc.edu/LDP/HOWTO/BootPrompt-HOWTO.html>); this section contains only a sketch of the most salient parameters.

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

When the kernel boots, a message `Memory: availk/totalk available` should be emitted early in the process. *total* should match the total amount of RAM, in kilobytes, which is available. If this doesn't match the actual of RAM you have installed, you need to use the `mem=ram` parameter, where *ram* is set to the amount of memory, suffixed with "k" for kilobytes, or "m" for megabytes. For example, both `mem=8192k` or `mem=8m` mean 8MB 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"`.

Again, full details on boot parameters can be found in the Linux BootPrompt HOWTO (<http://metalab.unc.edu/LDP/HOWTO/BootPrompt-HOWTO.html>), including tips for obscure hardware.

6.2 Booting With the Rescue Floppy

Booting from the Rescue Floppy is easy: place the Rescue Floppy in the primary floppy drive, and reset the system by pressing *reset*, or by turning the system off and on.

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

If you are using an alternative way to boot the system, follow the instructions, and wait for the `boot :` prompt to come up. If you boot from floppies smaller than 1.4MB floppy drive,

you have to use a ram-disk boot method, and you will need the Root Disk.

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

Information on boot parameters which might be useful can be found by pressing *F4* and *F5*. If you add any parameters to the boot command line, be sure to type the boot method (the default is `linux`) and a space before the first parameter (e.g., `linux floppy=thinkpad`). If you simply press *Enter*, that's the same as typing `linux` without any special parameters.

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

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

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

6.3 Interpreting the Kernel Startup Messages

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

depends on something. Most of these messages are harmless. You see them because the kernel for the installation system is built to run on computers with many different peripheral devices. Obviously, no one computer will have every possible peripheral device, so the operating system may emit a few complaints while it looks for peripherals you don't own. You may also see the system pause for a while. This happens when it is waiting for a device to respond, and that device is not present on your system. If you find the time it takes to boot the system unacceptably long, you can create a custom kernel later (see 'Compiling a New Kernel' on page 46).

6.4 Troubleshooting the Boot Process

If you have problems and the kernel hangs during the boot process, doesn't recognize peripherals you actually have, or drives are not recognized properly, the first thing to check is the boot parameters, as discussed in 'Boot Parameter Arguments' on page 30.

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

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

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

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

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

```
architecture: m68k
model: your general hardware vendor and model
memory: amount of RAM
scsi: SCSI host adapter, if any
cd-rom: CD-ROM model and interface type, i.e., ATAPI
network card: network interface card, if any
pcmcia: details of any PCMCIA devices
```

Depending on the nature of the bug, it also might be useful to report the disk model, disk capacity, and the model of video card.

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

Chapter 7

Using `dbootstrap` for Initial System Configuration

7.1 Introduction to `dbootstrap`

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

The main job of `dbootstrap`, and the main purpose of your initial system configuration, is to configure certain core elements of your system. For instance, this includes your IP address, host name, and other aspects of your networking setup, if any. This also includes the configuration of "kernel modules", which are drivers which are linked into the kernel. These modules include storage hardware drivers, network drivers, special language support, and support for other peripherals.

Configuring these fundamentals is done first, because it is often necessary for the proper functioning of your system or for the next steps of installation.

`dbootstrap` is a simple, character-based application (some systems do not have graphics capability). It is very easy to use; generally, it will guide you through each step of the installation process in a linear fashion. You can also go back and repeat steps if you found you made a mistake.

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

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

detect that you've done this from the shell. Press *Left Alt-F1* to get back to menus. Linux provides up to 64 virtual consoles, although the Rescue Floppy only uses a few of them.

Error messages are usually redirected the third virtual terminal (known as `ttty3`). You can access this terminal by pressing *Left Alt-F3* (hold the *Alt* key while pressing the *F3* function key); get back to `dbootstrap` with *Left Alt-F1*.

7.2 "Select Color or Monochrome display"

Once the system has finished booting, you should see the "Select Color or Monochrome display" dialog box.

That is, unless you are booting from a serial console. In that case, this dialog will be skipped; continue below at "'Debian GNU/Linux Installation Main Menu'" on the current page.

If your monitor is capable of displaying color, press *Enter*. The display should change from black-and-white to color. Then press *Enter* again, on the "Next" item, to continue with the installation.

If your monitor is only capable of black-and-white, use the arrow keys to move the cursor to the "Next" menu item and then *Enter* to continue with the installation. If you have a A2024 monitor, you might need to choose the "Monochrome" option.

7.3 "'Debian GNU/Linux Installation Main Menu'"

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

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

7.4 "Configure the Keyboard"

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

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

Due to a bug in `dbootstrap`, if you are running from a serial console, you will see a message such as "Cannot open /dev/tty0" when the system tries to load the keymap. Just ignore this problem and continue.

If you are installing a diskless workstation, the next few steps will be skipped, since there are no local disks to partition. In that case, your next step will be "Configure the Network" on page 39. After that, you will be prompted to mount your NFS root partition in "Mount a Previously-Initialized Partition" on page 37.

7.5 Last Chance!

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

7.6 "Partition a Hard Disk"

If you have not already partitioned your disks for Linux native and Linux swap filesystems, i.e., as described in 'Partitioning Prior to Installation' on page 16, the menu item "Next" will be "Partition a Hard Disk". If you have already created at least one Linux native and one Linux swap disk partition, the "Next" menu selection will be "Initialize and Activate a Swap Partition", or you may even skip that step if your system had low memory and you were asked to activate the swap partition as soon as the system started. Whatever the "Next" menu selection is, you can use the down-arrow key to select "Partition a Hard Disk".

The "Partition a Hard Disk" menu item presents you with a list of disk drives you can partition, and runs a partitioning application. You must create at least one "Linux native" (type 83) disk partition, and you probably want at least one "Linux swap" (type 82) partition, as explained in 'Partitioning Your Hard Drive' on page 12. If you are unsure how to partition your system, go back and read that chapter.

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

atari-fdisk Atari-aware version of `fdisk`; read the `atari-fdisk` manual page (`atari-fdisk.txt`).

amiga-fdisk Amiga-aware version of `fdisk`; read the `amiga-fdisk` manual page (`amiga-fdisk.txt`).

mac-fdisk Mac-aware version of `fdisk`; read the `mac-fdisk` manual page (`mac-fdisk.txt`).

mac-fdisk PowerMac-aware version of `fdisk`; read the `pmac-fdisk` manual page (`pmac-fdisk.txt`).

If you're unsure how to decide which partitions to make, and how large to make them, re-read 'Partitioning Your Hard Drive' on page 12.

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

7.7 "Initialize and Activate a Swap Partition"

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

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

Next you have the option to scan the entire partition for unreadable disk blocks caused by defects on the surface of the hard disk platters. This is useful if you have ACSI or older SCSI disks, and never hurts (although it can be time-consuming). Properly-working disks in most modern systems don't need this choice, as they have their own internal mechanism for mapping out bad disk blocks.

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

7.8 "Initialize a Linux Partition"

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

You can initialize a Linux partition, or alternately you can mount a previously-initialized one. Note that `dbootstrap` will *not* upgrade an old system without destroying it. If you're upgrading, Debian can usually upgrade itself, and you won't need to use `dbootstrap`. For upgrading instructions for Debian 2.1, see the upgrade instructions (<http://www.debian.org/releases/2.1/m68k/release-notes/>).

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

Select the "Next" menu item to initialize and mount the / disk partition. The first partition that you mount or initialize will be the one mounted as / (pronounced "root"). You will be offered the choice to scan the disk partition for bad blocks, as you were when you initialized the swap partition. It never hurts to scan for bad blocks, but it could take 10 minutes or more to do so if you have a large disk.

Once you've mounted the / partition, the "Next" menu item will be "Install Operating System Kernel and Modules" unless you've already performed some of the installation steps. You can use the arrow keys to select the menu items to initialize and/or mount disk partitions if you have any more partitions to set up. If you have created separate partitions for /var, /usr, or other filesystems, you should initialize and/or mount them now.

7.9 "Mount a Previously-Initialized Partition"

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

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

7.10 "Install Operating System Kernel and Modules"

This should be the next menu step after you've mounted your root partition, unless you've already performed this step in a previous run of `dbootstrap`. First, you will be asked to confirm that the device you have mounted on root is the proper one. Next, you will be offered a menu of devices from which you can install the kernel. Choose the appropriate device from which to install the kernel and modules (as you planned in 'Choosing Initial Boot Media' on page 20).

If you are installing from a local filesystem, select the "`harddisk`" device if the device is not yet mounted, or the "`mounted`" device if it is. Next, select the partition where the Debian installation software was installed back in 'Installing from a Hard Disk' on page 22. Next you'll be asked to specify the location on the filesystem where you put files; make sure you put a leading "/" on the location. After that, you should probably let `dbootstrap` try to find the actual files on its own; but it will let you pick if you need to.

On Macintosh systems, you will be offered three choices due to a quirk in the Linux HFS filesystem code:

- `/instmnt/debian/.finderinfo`
- `/instmnt/debian/.resource`
- `/instmnt/debian`

Only the last directory actually contains the data portion of the files. Either type in the right path, or skip the `.finderinfo` and `.resource` entries.

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

If you wish to install the kernel and modules over the network, you can do this using the "`nfs`" option. Your networking interfaces must be supported by the standard kernel (see 'Peripherals and Other Hardware' on page 8). If the "`nfs`" option doesn't appear, you need to select "`Cancel`", then go back and select the "`Configure the Network`" step (see "`Configure the Network`" on the next page). Then re-run this step. Select the "`nfs`" option, and then tell `dbootstrap` your NFS server name and path. Assuming you've put the Rescue Floppy and Drivers Floppy images on the NFS server in the proper location, these file should be available to you for installing the kernel and modules.

If you are installing a diskless workstation, you should have already configured your networking as described in "`Configure the Network`" on the following page. You should be given the option to install the kernel and modules from NFS. Proceed just as in the above paragraph.

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

7.11 "`Configure Device Driver Modules`"

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

You don't have to configure all your devices at this point; what is crucial is that any device configuration required for the installation of the base system is done here (see 'Choosing Media for Installing Base' on page 20). This includes ethernet drivers.

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

7.12 "Configure the Network"

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

If you are connected to a network, you'll need the information you collected from 'Information You Will Need' on page 9. However, if your primary connection to the network will be PPP, you should choose *NOT* to configure the network.

`dbootstrap` will ask you a number of questions about your network; fill in the answers from 'Information You Will Need' on page 9. The system will also summarize your network information and ask you for confirmation. Next, you need to specify the network device that your primary network connection uses. Usually, this will be "eth0" (the first ethernet device).

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

7.13 "Install the Base System"

During the "Install the Base System" step, you'll be offered a menu of devices from which you may install the base system. You should select the appropriate device, depending on the choice you made in 'Choosing Media for Installing Base' on page 20.

If you choose to install from a filesystem on the harddisk or from CD-ROM, you will be prompted to specify the path to the `common/base2_1.tgz` file. As with the "Install Operating System Kernel and Modules" step, you can either let `dbootstrap` find the file itself or type in the path at the prompt.

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

If you are installing the base system from NFS, then choose NFS and continue. You'll be prompted to specify the server, the share on the server, and the subdirectory within that share where the `common/base2_1.tgz` file can be found. If you have problems mounting NFS, make sure that the system time on the NFS server more or less agrees with the system time on the client. You can set your date on `tty2` using the `date` command; you'll have to set it by hand. See the `date(1)` manual page.

7.14 "Configure the Base System"

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

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

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

7.15 "Make Linux Bootable Directly From Hard Disk"

If you elect to make the hard disk boot directly to Linux, and you are *not* installing a diskless workstation, you will be asked to install a master boot record. If you aren't using a boot manager (and this is probably the case if you don't know what a boot manager is) and you don't have another different operating system on the same machine, answer "Yes" to this question.

If you answer "Yes", the next question will be whether you want to boot Linux automatically from the hard disk when you turn on your system. This sets Linux to be the *bootable partition* – the one that will be loaded from the hard disk.

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

FIXME: about the boot manager

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

7.16 The Moment of Truth

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

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

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

7.17 Set the Root Password

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

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

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

7.18 Create an Ordinary User

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

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

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

7.19 Shadow Password Support

Next, the system will ask whether you want to enable shadow passwords. This is a system in which your Linux system is made to be a bit more secure. In a system without shadow passwords, passwords are stored

(encrypted) in a world-readable file, `/etc/passwd`. This file has to be readable to anyone who can log in because it contains vital user information, for instance, how to map between numeric user identifiers and login names. Therefore, someone could conceivably grab your `/etc/passwd` file and run a brute force attack against it to try to determine passwords.

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

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

7.20 Select and Install Profiles

The system will now ask you if you want to use the pre-rolled software configurations offered by Debian. You can always choose, package by package, what do you want to install on your new machine. This is the purpose of the `dselect` program, described below. But this can be a long task with around 2050 packages available in Debian!

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

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

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

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

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

7.21 Log In

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

7.22 Setting up PPP

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

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

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

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

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

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

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 `pooff`, again, as root.

7.23 Installing the Rest of Your System

Information about the installation of the rest of your Debian system is contained in a separate document, the `dselect` Tutorial (`dselect-beginner.html`). Remember to skip the "Select" step in `dselect` if you are using the profiles and tasks from 'Select and Install Profiles' on page 42.

Chapter 8

Next Steps and Where to Go From Here

8.1 If You Are New to Unix

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

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

Information specific to Debian can be found below.

8.2 Orienting Yourself to Debian

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

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

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

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

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

8.3 Further Reading and Information

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

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

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

8.4 Compiling a New Kernel

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

- handle hardware or options not included in the stock kernel, such as APM or SMP
- optimize the kernel by removing useless drivers, which speeds up boot time and makes the kernel size smaller (kernel memory cannot be swapped to disk)
- use options of the kernel which are not supported by the default kernel (such as network firewalling)
- run a development kernel
- impress your friends, try new things

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

To compile a kernel the Debian way, you need some packages: `kernel-package`, `kernel-source-2.0.35` (the most recent version at the time of this writing), `fakeroot` and a few others which are probably already installed (see `/usr/doc/kernel--package/README.gz` for the complete list). Note that you don't *have* to compile your kernel the "Debian way"; but we find that using the packaging system to manage your kernel is actually safer and easier. In fact, you can get your kernel sources right from Linus instead of `kernel-source-2.0.35`, yet still use the `kernel-package` compilation method.

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

Hereafter, we'll assume your kernel source will be located in `/usr/local/src` and that your kernel version is 2.0.35. As root, create a directory under `/usr/local/src` and change the owner of that file to your normal non-root account. As your normal non-root account, change your directory to where you want to unpack the kernel sources (`cd /usr/local/src`), extract the kernel sources (`tar xzf /usr/src/kernel-source-2.0.35.tar.gz`), change your directory to it (`cd kernel-source-2.0.35/`). Now, you can configure your kernel (make `xconfig` if X11 is installed and configured, make `menuconfig` otherwise). 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 daemon support (e.g. autoload of modules)" in "Loadable module support" (it is not selected by default) or your Debian installation will experience problems.¹

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

Now, compile the kernel: `fakeroot /usr/sbin/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" (i.e., a host name). Kernel compilation may take quite a while, depending on the power of your machine.

Once the compilation is complete, you can install your custom kernel like any package. As root, do `dpkg -i ../kernel-image-2.0.35-subarch_custom.1.0_m68k.deb`. The *subarch* part is an optional sub-architecture, depending on what kernel options you set. `dpkg -i kernel_image...` will install the kernel, along with some other nice supporting files. For instance, the `System.map` will be properly installed (helpful for debugging kernel problems), and `/boot/config--2.0.35` will be installed, containing your current configuration set. Your new `kernel-image-2.0.35` package is also clever enough to automatically use `lilo` to update the kernel image information allowing you to boot, so there's no need to re-run `lilo`. If you have created a modules package, 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`.

¹Note that `kernelld` is replaced by `kmod` and you have to select "Kernel module loader" instead. The Linux 2.2 kernel is not fully supported by Debian 2.1; see "Using the Linux 2.2 Kernel with Debian 2.1" on the following page for details and workarounds.

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

8.5 Using the Linux 2.2 Kernel with Debian 2.1

Debian 2.1 is not certified for use with the Linux 2.2 kernel. However, if you are willing to download some packages from `ftp://ftp.debian.org/debian/dists/unstable/`, you should be able to have a functioning system. We do expect to add 2.2 compatibility soon; see the Debian 2.1 pages (<http://www.debian.org/releases/2.1/>) for updates.

There are a number of packages which are known to be incompatible with the 2.2 kernel. The Unofficial Debian GNU/Linux 2.2 Checklist (<http://www.debian.org/~rcw/2.2/warnings.html>) might be helpful in isolating these.

Chapter 9

Technical Information on the Boot Floppies

9.1 Source Code

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

9.2 Rescue Floppy

The Rescue Floppy has an Ext2 filesystem (or a FAT filesystem, depending on your architecture), and you should be able to access it from anything else that can mount EXT2 or FAT disks. The Linux kernel is in the file `linux`. The file `root.bin` is a gzip-compressed disk image of a 1.4MB Minix or EXT2 filesystem, and will be loaded into the RAM disk and used as the root filesystem.

9.3 Replacing the Rescue Floppy Kernel

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

- Initial RAM disk
- FAT, Minix, and EXT2 filesystems (some architectures don't need FAT and/or Minix filesystems – see the source)
- ELF executables

Copy your new kernel to the file `linux` on the Rescue Floppy, and then run the shell script `rdev.sh` that you'll find on the floppy.

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

9.4 The Base Floppies

The base floppies contain a 512-byte header followed by a portion of a `gzip`-compressed `tar` archive. If you strip off the headers and then concatenate the contents of the base floppies, the result should be the compressed tar archive. The archive contains the base system that will be installed on your hard disk. Once this archive is installed, you must go through the "Configure the Base System" menu item in the installation system and other menu items to configure the network and install the operating system kernel and modules before the system will be usable.

Chapter 10

Administrivia

10.1 About This Document

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

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

10.2 Contributing to This Document

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

Better yet, get a copy of the SGML source for this document, and produce patches against it. The SGML source can be found in the `boot-floppies`; try to find the newest revision in the unstable (<ftp://ftp.debian.org/debian/dists/unstable/>) distribution. CVS source access is also coming soon.

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

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

10.3 Major Contributions

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

Extremely helpful text and information was found in Jim Mintha's HOWTO for network booting (http://www.geog.ubc.ca/s_linux/howto/netboot.html), 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.geog.ubc.ca/s_linux/faq.html), the Linux/Alpha FAQ (<http://www.alphalinux.org/faq/FAQ.html>), amongst others. The maintainers of these freely available and rich sources of information must be recognized.

10.4 Trademark Acknowledgement

All trademarks are property of their respective trademark owners.