Release Notes for Debian GNU/Linux 4.0 ("etch"), SPARC

Josip Rodin, Bob Hilliard, Adam Di Carlo, Anne Bezemer, Rob Bradford, Frans Pop (current), Andreas Barth (current), Javier Fernández-Sanguino Peña (current), Steve Langasek (current)

`<debian-doc@lists.debian.org>`

Contents

Chapter 1

Introduction

The primary goals of these Release Notes are to inform users of major changes in this release of the Debian GNU/Linux distribution, to provide information on how to upgrade safely from the previous release to the current release and finally to inform users of known potential issues they could encounter when upgrading to or using the etch release.

Note that it is impossible to list every known issue and that therefore a selection has been made based on a combination of the expected prevalence and impact of issues.

The most recent version of this document is always available at http://www.debian.org/releases/stable/releasenotes. If the version you are reading is more than a month old[1], you might wish to obtain the latest version.

Please note that we only support and document upgrading from the previous release of Debian (in this case, the upgrade from sarge). If you need to upgrade from older releases, we suggest you read previous editions of the release notes and upgrade to sarge first.

1.1   Reporting bugs on this document

We have attempted to test all the different upgrade steps described in this document and we have also tried to anticipate all the possible issues our users might encounter.

Nevertheless, if you think you have found any bug in this documentation (incorrect information or information that is missing), please file a bug in the bug tracking system (http://bugs.debian.org/) against the release-notes package.

1.2   Contributing upgrade reports

We welcome any information from users related to upgrades from sarge to etch. If you are willing to share information please file a bug in the bug tracking system (http://bugs.

---

[1]As listed on the front page of the PDF version and in the footer of the HTML version.

debian.org/) against the upgrade-reports package with your results. We request that you compress any attachments that are included (using gzip).

Please include the following information when submitting your upgrade report:

- The status of your package database before and after the upgrade: dpkg's status database available at /var/lib/dpkg/status and aptitude's package state information, available at /var/lib/aptitude/pkgstates. You should have made a backup before the upgrade as described at 'Back up any data or configuration information' on page 15, but you can also find backups of this information in /var/backups.

- Session logs using script, as described in 'Recording the session' on page 22.

- Your aptitude logs, available at /var/log/aptitude.

Note: you should take some time to review and remove any sensitive and/or confidential information from the logs before including them in a bug report as the information will be published in a public database.

1.3   Sources for this document

This document is generated using debiandoc-sgml.   Sources for the Release Notes are available in the CVS repository of the *Debian Documentation Project*.   You can use the web interface (http://cvs.debian.org/ddp/manuals.sgml/release-notes/?root=debian-doc) to access its files individually through the web and see their changes. For more information on how to access the CVS please consult the Debian Documentation Project CVS pages (http://www.debian.org/doc/cvs).

Chapter 2

What's new in Debian GNU/Linux 4.0

This release adds official support for the AMD64 architecture which supports 64-bit processors from both Intel (EM64T) and AMD (AMD64). During the previous release, Debian GNU/Linux 3.1 ('sarge'), an unofficial version of this port was available.

Official support for the Motorola 680x0 ('m68k') architecture has been dropped because it did not meet the criteria set by the Debian Release Managers. The most important underlying reasons are performance and limited upstream support for essential toolchain components. However, the m68k port is expected to remain active and available for installation even if not a part of this official stable release.

The following are the officially supported architectures for Debian GNU/Linux etch:

- Intel x86 ('i386')

- Alpha ('alpha')

- SPARC ('sparc')

- PowerPC ('powerpc')

- ARM ('arm')

- MIPS ('mips' (big-endian) and 'mipsel' (little-endian))

- Intel Itanium ('ia64')

- HP PA-RISC ('hppa')

- S/390 ('s390')

- AMD64 ('amd64')

You can read more about port status, and port-specific information for your architecture at the Debian port web pages (http://www.debian.org/ports/sparc/).

2.1   What's new in the distribution?

This new release of Debian again comes with a lot more software than its predecessor sarge; the distribution includes over 6500 new packages, for a total of over 18200 packages. Most of the software in the distribution has been updated: over 10700 software packages (this is 68% of all packages in sarge). Also, a significant number of packages (over 3500, 23% of the packages in sarge) have for various reasons been removed from the distribution. You will not see any updates for these packages and they will be marked as 'obsolete' in package management front-ends.

With this release, Debian GNU/Linux switches from XFree86 to the 7.1 release of X.Org, which includes support for a greater range of hardware and better autodetection. This allows the use of Compiz, which is one of the first compositing window managers for the X Window System, taking full advantage of hardware OpenGL acceleration for supported devices.

Debian GNU/Linux again ships with several desktop applications and environments. Among others it now includes the desktop environments GNOME 2.14[1], KDE 3.5.5a, and Xfce 4.4. Productivity applications have also been upgraded, including the office suites OpenOffice.org 2.0.4a and KOffice 1.6 as well as GNUcash 2.0.5, GNUmeric 1.6.3 and Abiword 2.4.6.

Updates of other desktop applications include the upgrade to Evolution 2.6.3 and Gaim 2.0. The Mozilla suite has also been updated, with a rename of the main programs: `iceweasel` (version 2.0.0.2) is the unbranded `Firefox` web browser and `icedove` (version 1.5) is the unbranded `Thunderbird` mail client.

Among many others, this release also includes the following software updates:

- the GNU C library, version 2.3.6

- the GNU Compiler Collection 4.1 as default compiler

- language interpreters: Python 2.4, PHP 5.2

- server software:

    - e-mail servers: Exim 4.63 (default email server for new installations), Postfix 2.3, Courier 0.53, Cyrus 2.2
    - web servers: Apache 2.2, fnord 1.10
    - database servers: MySQL 5.0.32, PostgreSQL 8.1
    - the OpenSSH server, version 4.3
    - name servers: Bind 9.3, maradns 1.2
    - directory server: OpenLDAP 2.3

The official Debian GNU/Linux distribution now ships on 19 to 23 binary CDs (depending on the architecture) and a similar number of source CDs. A DVD version of the distribution is also available.

---

[1]With some modules from GNOME 2.16.

2.1.1   Package management

`aptitude` is the preferred program for package management from console. `aptitude` supports most command line operations of `apt-get` and has proven to be better at dependency resolution than `apt-get`. If you are still using `dselect`, you should switch to `aptitude` as the official frontend for package management.

For etch an advanced conflict resolving mechanism has been implemented in `aptitude` that will try to find the best solution if conflicts are detected because of changes in dependencies between packages.

Secure APT is now available in etch. This feature adds extra security to Debian GNU/Linux systems by easily supporting strong cryptography and digital signatures to validate downloaded packages. This release includes the `apt-key` tool for adding new keys to apt's keyring, which by default includes only the current Debian archive signing key, provided in the `debian-archive-keyring` package.

In its default configuration, `apt` will now warn if packages are downloaded from sources that are not authenticated. Future releases might force all packages to be verified before downloading them. Administrators of unofficial apt repositories are encouraged to generate a cryptographic key and sign their Release files, as well as providing a secure way to distribute their public keys.

For more information please read `apt(8)`, the Package signing in Debian (http://www.debian.org/doc/manuals/securing-debian-howto/ch7#s-deb-pack-sign) chapter of the *Securing Debian Manual* and the Debian Wiki (http://wiki.debian.org/SecureApt).

Another feature that was added in `apt` is the ability to download only the changes in `Packages` files since your last update. More about this feature in 'Slower updates of APT package index files' on page 38.

2.1.2   debian-volatile now an official service

The *debian-volatile* service that was introduced as an unofficial service with the release of sarge has now become an official Debian GNU/Linux service.

This means that it now uses a `.debian.org` address[2]. Please make sure to update your `/etc/apt/sources.list` accordingly if you were already using this service.

debian-volatile allows users to easily update stable packages that contain information that quickly goes out of date. Examples are a virus scanner's signatures list or a spam filter's pattern set. For more information and a list of mirrors, please see the archive's web page (http://volatile.debian.org/).

---

[2]The old `volatile.debian.net` address will also remain valid for the time being.

2.2   System improvements

There have been a number of changes in the distribution that will benefit new installations of etch, but may not be automatically applied on upgrades from sarge. This section gives an overview of the most relevant changes.

Priority for basic development packages lowered A number of development packages that used to be priority *standard* are now priority *optional*, which means they will no longer be installed by default. This includes the standard C/C++-compiler, `gcc`, as well as some other software (`dpkg-dev`, `flex`, `make`) and development headers (`libc6-dev`, `linux-kernel-headers`).

   If you do wish to have these packages on your system, the easiest way to install them is by installing `build-essential`, which will pull in most of them.

SELinux priority standard, but not enabled by default The packages needed for SELinux support have been promoted to priority *standard*. This means that they will be installed by default during new installations. For existing systems you can install SELinux using:

```
# aptitude install selinux-basics
```

   Note that SELinux support is *not* enabled by default. Information on setting up and enabling SELinux can be found on the Debian Wiki (http://wiki.debian.org/SELinux).

New default inet superdaemon The default inet superdaemon for etch is `openbsd-inetd` instead of `netkit-inetd`. It will not be started if no services are configured, which is true by default. The new default daemon will be installed automatically on upgrade.

Default `vi` clone changed The `vi` clone installed by default is now a compact version of `vim` (`vim-tiny`) instead of `nvi`.

Changes in default features for `ext2/ext3` New ext2 and ext3 file systems will be created with features *dir_index* and *resize_inode* enabled by default. The first feature speeds up operations on directories with many files; the second makes it possible to resize a file system on-line (i.e. while it is mounted).

   Users upgrading from sarge could consider adding the *dir_index* flag manually using `tune2fs`[3]; the *resize_inode* flag cannot be added to an existing file system. It is possible to check which flags are set for a file system using `dumpe2fs -h`.

Default encoding for etch is UTF-8 The default encoding for new Debian GNU/Linux installations is UTF-8. A number of applications will also be set up to use UTF-8 by default.

   Users upgrading to etch that wish to switch to UTF-8 will need to reconfigure their environment and locale definitions. The system-wide default can be changed using

---

[3]The flag *filetype* should already be set on most file systems, except possibly on systems installed before sarge.

dpkg-reconfigure locales; first select a UTF-8 locale for your language and country and then set that as default. Note that switching to UTF-8 means that you will probably also need to convert existing files from your previous (legacy) encoding to UTF-8.

The package utf8-migration-tool contains a tool that may help the migration, however that package is only available in unstable as it was not ready in time for etch. Making a backup of your data and configuration before using the tool is strongly recommended.

Note that some applications may not yet work correctly in a UTF-8 environment, mostly due to display issues.

The Debian Wiki (http://wiki.debian.org/Sarge2EtchUpgrade) has some additional information about changes between sarge and etch.

2.3   Major kernel-related changes

Debian GNU/Linux 4.0 ships with kernel version 2.6.18 for all architectures; the release is still mostly [4] compatible with 2.4 kernels, but Debian no longer provides or supports 2.4 kernel packages.

There have been major changes both in the kernel itself and in the packaging of the kernel for Debian. Some of these changes complicate the upgrade procedure and can potentially result in problems while rebooting the system after the upgrade to etch. This section gives an overview of the most important changes; potential issues and information on how to work around them is included in later chapters.

If you are currently using a 2.4 kernel, you should read 'Upgrading to a 2.6 kernel' on page 39 carefully.

2.3.1   Changes in kernel packaging

Kernel packages renamed All Linux kernel packages have been renamed from kernel-* to linux-* to clean up the namespace. This will make it easier to include non-Linux kernels in Debian in the future.

Where possible, dummy transition packages that depend on the new packages have been provided for the dropped packages.

2.3.2   New utilities to generate initrds

The Debian kernel image packages for SPARC require an initrd for booting the system. Because of changes in the kernel, the utility used to generate initrds in sarge, initrd-tools

---

[4]Some individual packages may no longer work correctly with a 2.4 kernel; see 'Some applications may no longer work with a 2.4 kernel' on page 37.

can no longer be used and has been deprecated. Two new utilities have been developed that replace it: `initramfs-tools` and `yaird`. The concepts behind the new utilities are very different; an overview is available on the Debian Wiki (`http://wiki.debian.org/InitrdReplacementOptions`). Both will generate an initrd using the *initramfs* file system, which is a compressed `cpio` archive. The default and recommended utility is `initramfs-tools`.

Upgrading to an etch kernel will cause `initramfs-tools` to be installed by default. If you are upgrading from a 2.4 kernel to a 2.6 Debian kernel, you must use `initramfs-tools`. Using `yaird` will cause linux-image-2.6 installations to fail if you are running a 2.2 or 2.4 kernel.

The package `initrd-tools` is still included in etch because it is needed for upgrades from sarge. It will be dropped for the next release.

2.3.3 Dynamic `/dev` management and hardware discovery

etch kernels no longer provide support for `devfs`.

The replacement for `devfs` is `udev`, a userspace implementation of devfs.

`udev` is mounted over the `/dev` directory and will populate that directory with devices supported by the kernel. It will also dynamically add and remove devices as kernel modules are loaded or unloaded respectively, based on events generated by the kernel. `udev` is a lot more versatile than `devfs` and offers services that are used by other packages like `hal` (hardware abstraction layer).

In combination with the kernel, `udev` also takes care of hardware discovery and module loading for detected devices. Because of this it conflicts with `hotplug`. In sarge, `discover` could also be used for loading modules during the boot process, but its new version in etch no longer provides that function. `discover` is still used by X.Org to detect what graphics controller is present in the system.

If you install a Debian kernel image, `udev` will be installed by default as `initramfs-tools` depends on it.

You can avoid installing `udev` by compiling a custom non-modular kernel or by using an alternative initrd generator, such as `yaird`. However, `initramfs-tools` is the recommended initrd generator.

Chapter 3

Installation System

The Debian Installer is the official installation system for Debian. It offers a variety of installation methods. Which methods are available to install your system depends on your architecture.

Images of the installer for etch can be found together with the Installation Guide on the Debian website (http://www.debian.org/releases/stable/debian-installer/).

The Installation Guide is also included on the first CD/DVD of the official Debian CD/DVD sets, at:

```
/doc/install/manual/language/index.html
```

You may also want to check the errata (http://www.debian.org/releases/stable/debian-installer/index#errata) for debian-installer for a list of known issues.

3.1   Issues with framebuffer on SPARC

Because of display problems on some systems, framebuffer support is disabled by default for SPARC for most graphics cards. This can result in ugly display on systems that do properly support the framebuffer. If you see display problems in the installer, you can try booting the installer with the parameter framebuffer=true. Please let us know if the framebuffer is not used by default, but works for your hardware.

3.2   Issues with booting on SPARC

It has been reported by several users that the installation CD fails to boot successfully upon the 'boot cdrom' PROM command, displaying the error 'Illegal Instruction'.

The apparent explanation for this problem is that it doesn't work because the machine had previously been rebooted from Solaris. The workaround is to power the machine off fully, and then boot it directly into the installation CD.

The problem was reported by users of various systems (namely, Enterprise 450, Blade 2000, Fire V240, Enterprise 250, Blade 100, Enterprise 220R, and Sun Ultra 60 at the time of writing), so it is believed to be generic. Please let us know if you observe similar issues with your hardware.

3.3   Issues with booting from qla2xxx on SPARC

It has been reported by several users that the installation system fails to recognize hard disks on machines which have the hard disks connected to a QLogic fibre-channel SCSI controller. This includes the Sun Fire 280R servers. The qla2xxx driver loads, but it cannot load firmware, which makes it useless.

The explanation for this problem is that the QLogic controller firmware is not free, and it had to be moved to a separate non-free package (firmware-qlogic (`http://packages.debian.org/firmware-qlogic`)) which is not used by the installation system.

There is no straightforward solution, unfortunately; one has to first provide the firmware image to the installation system, and then later do the same in the installed system. To get the installer to load the firmware, one has to have network connectivity while the machine is being installed in order to download the `firmware-qlogic` udeb package with wget, install it with udpkg, and then reload the qla2xxx module. After the installation is complete, mount the new root partition, chroot to it, fetch the `firmware-qlogic` deb package, install it with dpkg, and then run `update-initramfs` in order to include it in the initial ramdisk image used by the kernel.

Alternatively, install from an older installation CD (where that non-free firmware was still integrated) and then upgrade.

3.4   Size of harddisk not detected correctly

If a hard disk has previously used under Solaris, the partitioner may not detect the size of the drive correctly. Creating a new partition table does not fix this issue. What does help, is to "zero" the first few sectors of the drive:

```
# dd if=/dev/zero of=/dev/hdX bs=512 count=2; sync
```

Note that this will make any existing data on that disk inaccessible.

3.5   What's new in the installation system?

There has been a lot of development on the Debian Installer since its first official release with sarge resulting in both improved hardware support and some exciting new features.

In these Release Notes we'll only list the major changes in the installer. If you are interested in an overview of the detailed changes since sarge, please check the release announcements for the etch beta and RC releases available from the Debian Installer's news history (`http://www.debian.org/devel/debian-installer/News/`).

3.5.1   Major changes

No reboot during the installation Previously, the installation was split into two parts: setting up the base system and making it bootable, followed by a reboot and after that the execution of `base-config` which would take care of things like user setup, setup of the package management system and installation of additional packages (using tasksel).

For etch the second stage has been integrated into Debian Installer itself. This has a number of advantages, including increased security and the fact that after the reboot at the end of the installation the new system should already have the correct timezone and, if you installed the Desktop environment, will at once start the graphical user interface.

UTF-8 encoding default for new systems The installer will set up systems to use UTF-8 encoding rather than the old language-specific encodings (like ISO-8859-1, EUC-JP or KOI-8).

More flexible partitioning It is now possible to set up file systems on an LVM volume using guided partitioning.

The installer is also able to set up encrypted file systems. Using manual partitioning you have the choice between `dm-crypt` and `loop-aes`, using a passphrase or a random key, and you can tune various other options. Using guided partitioning, the installer will create an encrypted LVM partition that contains any other file systems (except `/boot`) as logical volumes.

Rescue mode You can use the installer to solve problems with your system, for example when it refuses to boot. The first steps will be just like a regular installation, but the installer will not start the partitioner. Instead it will offer you a menu of rescue options.

Activate the rescue mode by booting the installer with `rescue`, or by adding a boot parameter `rescue/enable=true`.

Using sudo instead of root account During expert installations you can choose to not set up the root account (it will be locked), but instead set up `sudo` so that the first user can use that for system administration.

Cryptographic verification of downloaded packages Packages downloaded with the installer are now cryptographically checked using `apt`, making it more difficult to compromise a system being installed over the network.

Simplified mail configuration If the "standard system" is installed, the installer sets up a basic configuration for the system's mail server which will only provide for local e-mail delivery. The mail server will be unavailable to other systems connected to the same network. If you want to configure your system to handle e-mail not local to the system

(either to send e-mail or to receive it), you will have to reconfigure the mail system after installation.

Desktop selection The installation system will install a GNOME desktop as the default desktop if the user asks for one.

However, users wishing to install alternate desktop environments can easily do so by adding boot parameters: `tasks="standard, kde-desktop"` for KDE and `tasks="standard, xfce-desktop"` for Xfce. Note that this will not work when installing from a full CD image without using a network mirror as an additional package source; it will work when using a DVD image or any other installation method.

There are also separate CD images available that install the KDE or Xfce desktop environment by default.

New languages Thanks to the huge efforts of translators, Debian can now be installed in 47 languages using the text-based installation user interface. This is six languages more than in sarge. Languages added in this release include Belarusian, Esperanto, Estonian, Kurdish, Macedonian, Tagalog, Vietnamese and Wolof. Due to lack of translation updates, two languages have been dropped in this release: Persian and Welsh.

Users that do not wish to use any locale can now select *C* as their preferred locale in the installer's language selection. More information on language coverage is available at the d-i languages list (http://d-i.alioth.debian.org/i18n-doc/languages.html).

Simplified localization and timezone selection Configuration of language, countries and timezones has been simplified to reduce the amount of information needed from the user. The installer will now guess what the system's country and timezone is based on the language selected, or will provide a limited selection if it cannot. Users can still introduce obscure combinations if need be.

Improved system-wide localization Most of the internationalization and localization tasks that were previously handled by the `localization-config` tool are now included in the stock Debian installer or in packages themselves. This means that selection of a language will automatically install packages necessary for that language (dictionaries, documentation, fonts...) in both standard and desktop environments. Configuration that is no longer handled automatically includes the papersize configuration and some advanced X Windows keyboard settings for some languages.

Note that language-specific packages will only be installed automatically if they are available during the installation.

3.5.2 Automated installation

A lot of the changes mentioned in the previous section also imply changes in the support in the installer for automated installation using preconfiguration files. This means that if you have existing preconfiguration files that worked with the sarge installer, you cannot expect these to work with the new installer without modification.

The good news is that the Installation Guide (http://www.debian.org/releases/stable/installmanual) now has a separate appendix with extensive documentation on using preconfiguration.

The etch installer introduces some exciting new features that allow further and easier automation of installs. It also adds support for advanced partitioning using RAID, LVM and encrypted LVM. See the documentation for details.

3.6 Popularity contest

The installation system will again offer to install the `popularity-contest` package. This package was not installed by default in sarge but it was installed in older releases.

`popularity-contest` provides the Debian project with valuable information on which packages in the distribution are actually used. This information is used mainly to decide the order in which packages are included on installation CD-ROMs, but is also often consulted by Debian developers in deciding whether or not to adopt a package that no longer has a maintainer.

Information from `popularity-contest` is processed anonymously. We would appreciate it if you would participate in this official survey, helping to improve Debian.

Chapter 4

Upgrades from previous releases

4.1 Preparing for the upgrade

We suggest that before upgrading you also read the information in 'Issues to be aware of for etch' on page 37. That chapter covers potential issues not directly related to the upgrade process but which could still be important to know about before you begin.

4.1.1 Back up any data or configuration information

Before upgrading your system, it is strongly recommended that you make a full backup, or at least back up any data or configuration information you can't afford to lose. The upgrade tools and process are quite reliable, but a hardware failure in the middle of an upgrade could result in a severely damaged system.

The main things you'll want to back up are the contents of `/etc`, `/var/lib/dpkg`, `/var/lib /aptitude/pkgstates` and the output of `dpkg --get-selections` "`*`" (the quotes are important).

The upgrade process itself does not modify anything in the `/home` directory. However, some applications (e.g. parts of the Mozilla suite, and the GNOME and KDE desktop environments) are known to overwrite existing user settings with new defaults when a new version of the application is first started by a user. As a precaution, you may want to make a backup of the hidden files and directories ("dotfiles") in users' home directories. This backup may help to restore or recreate the old settings. You may also want to inform users about this.

Any package installation operation must be run with superuser privileges, so either login as root or use `su` or `sudo` to gain the necessary access rights.

The upgrade has a few preconditions; you should check them before actually executing the upgrade.

4.1.2 Inform users in advance

It's wise to inform all users in advance of any upgrades you're planning, although users accessing your system via an `ssh` connection should notice little during the upgrade, and should be able to continue working.

If you wish to take extra precautions, back up or unmount users' partitions (`/home`) before upgrading.

You will probably have to do a kernel upgrade when upgrading to etch, so a reboot will normally be necessary. Typically, this will be done after the upgrade is finished.

4.1.3 Prepare for recovery

Because of the many changes in the kernel between sarge and etch regarding drivers, hardware discovery and the naming and ordering of device files, there is a real risk that you may experience problems rebooting your system after the upgrade. A lot of known potential issues are documented in this and the next chapters of these Release Notes.

For that reason it makes sense to ensure that you will be able to recover if your system should fail to reboot or, for remotely managed systems, fail to bring up networking.

If you are upgrading remotely via an `ssh` link it is highly recommended that you take the necessary precautions to be able to access the server through a remote serial terminal. There is a chance that, after upgrading the kernel and rebooting, some devices will be renamed (as described in 'Device enumeration reordering' on page 31) and you will have to fix the system configuration through a local console. Also, if the system is rebooted accidentally in the middle of an upgrade there is a chance you will need to recover using a local console.

The most obvious thing to try first is to reboot with your old kernel. However, for various reasons documented elsewhere in this document, this is not guaranteed to work.

If that fails, you will need an alternative way to boot your system so you can access and repair it. One option is to use a special rescue image or a Linux live CD. After booting from that, you should be able to mount your root file system and `chroot` into it to investigate and fix the problem.

Another option we'd like to recommend is to use the *rescue mode* of the etch Debian Installer. The advantage of using the installer is that you can choose between its many installation methods for one that best suits your situation. For more information, please consult the section "Recovering a Broken System" in chapter 8 of the Installation Guide (`http://www.debian.org/releases/stable/installmanual`) and the Debian Installer FAQ (`http://wiki.debian.org/DebianInstaller/FAQ`).

Debug shell during boot using initrd

The `initramfs-tools` includes a debug shell[1] in the initrds it generates. If for example the initrd is unable to mount your root file system, you will be dropped into this debug shell which has basic commands available to help trace the problem and possibly fix it.

Basic things to check are: presence of correct device files in `/dev`; what modules are loaded (`cat /proc/modules`); output of `dmesg` for errors loading drivers. The output of `dmesg` will also show what device files have been assigned to which disks; you should check that against the output of `echo $ROOT` to make sure that the root file system is on the expected device.

If you do manage to fix the problem, typing `exit` will quit the debug shell and continue the boot process at the point it failed. Of course you will also need to fix the underlying problem and regenerate the initrd so the next boot won't fail again.

4.1.4 Prepare a safe environment for the upgrade

The distribution upgrade should be done either locally from a textmode virtual console (or a directly connected serial terminal), or remotely via an `ssh` link.

In order to gain extra safety margin when upgrading remotely, we suggest that you run upgrade processes in the virtual console provided by the `screen` program, which enables safe reconnection and ensures the upgrade process is not interrupted even if the remote connection process fails.

Important! You should *not* upgrade using `telnet`, `rlogin`, `rsh`, or from an X session managed by `xdm`, `gdm` or `kdm` etc on the machine you are upgrading. That is because each of those services may well be terminated during the upgrade, which can result in an *inaccessible* system that is only half-upgraded.

4.1.5 Support for 2.2-kernels has been dropped

In case you run a kernel prior to 2.4.1, you need to upgrade to (at least) the 2.4-series before upgrading `glibc`. This should be done before starting the upgrade. It is recommended that you directly upgrade to the 2.6.8 kernel available in sarge, instead of upgrading to a 2.4 kernel.

4.2 Checking system status

The upgrade process described in this chapter has been designed for upgrades from "pure" sarge systems without third-party packages. In particular, there are known problems with third-party packages which install programs under `/usr/X11R6/bin/` causing problems with upgrades due to the X.Org transition ('XFree86 to X.Org transition' on page 41). For

---

[1]This feature can be disabled by adding the parameter `panic=0` to your boot parameters.

greatest reliability of the upgrade process, you may wish to remove third-party packages from your system before you begin upgrading.

This procedure also assumes your system has been updated to the latest point release of sarge. If you have not done this or are unsure, follow the instructions in 'Upgrading your sarge system' on page .

4.2.1   Review actions pending in package manager

In some cases, the use of `apt-get` for installing packages instead of `aptitude` might make `aptitude` consider a package as "unused" and schedule it for removal. In general, you should make sure the system is fully up-to-date and "clean" before proceeding with the upgrade.

Because of this you should review if there are any pending actions in the package manager `aptitude`. If a package is scheduled for removal or update in the package manager, it might negatively impact the upgrade procedure. Note that correcting this is only possible if your `sources.list` still points to *sarge*; and not to *stable* or *etch*; see 'Checking your sources list' on page .

To do this, you have to run `aptitude`'s user interface and press 'g' ("Go"). If it shows any actions, you should review them and either fix them or implement the suggested actions. If no actions are suggested you will be presented with a message saying "No packages are scheduled to be installed, removed, or upgraded".

4.2.2   Disabling APT pinning

If you have configured APT to install certain packages from a distribution other than stable (e.g. from testing), you may have to change your APT pinning configuration (stored in `/etc/apt/preferences`) to allow the upgrade of packages to the versions in the new stable release. Further information on APT pinning can be found in `apt_preferences(5)`.

4.2.3   Checking packages status

Regardless of the method used for upgrading, it is recommended that you check the status of all packages first, and verify that all packages are in an upgradable state. The following command will show any packages which have a status of Half-Installed or Failed-Config, and those with any error status.

```
# dpkg --audit
```

You could also inspect the state of all packages on your system using `dselect`, `aptitude`, or with commands such as

```
# dpkg -l | pager
```

or

```
# dpkg --get-selections "*" > ~/curr-pkgs.txt
```

It is desirable to remove any holds before upgrading. If any package that is essential for the upgrade is on hold, the upgrade will fail.

Note that `aptitude` uses a different method for registering packages that are on hold than `apt-get` and `dselect`. You can identify packages on hold for `aptitude` with

```
# aptitude search "~ahold" | grep "^.h"
```

If you want to check which packages you had on hold for `apt-get`, you should use

```
# dpkg --get-selections | grep hold
```

If you changed and recompiled a package locally, and didn't rename it or put an epoch in the version, you must put it on hold to prevent it from being upgraded.

The "hold" package state for `aptitude` can be changed using:

```
# aptitude hold package_name
```

Replace `hold` with `unhold` to unset the "hold" state.

If there is anything you need to fix, it is best to make sure your `sources.list` still refers to sarge as explained in 'Checking your sources list' on page 51.

4.2.4 Unofficial sources and backports

If you have any non-Debian packages on your system, you should be aware that these may be removed during the upgrade because of conflicting dependencies. If these packages were installed by adding an extra package archive in your `/etc/apt/sources.list`, you should check if that archive also offers packages compiled for etch and change the source line accordingly at the same time as your source lines for Debian packages.

Some users may have unofficial backported "newer" versions of packages that *are* in Debian installed on their sarge system. Such packages are most likely to cause problems during an upgrade as they may result in file conflicts[2]. Section 'Possible issues during upgrade' on page 28 has some information on how to deal with file conflicts if they should occur.

---

[2]Debian's package management system normally does not allow a package to remove or replace a file owned by another package unless it has been defined to replace that package.

4.3   Manually unmarking packages

To prevent `aptitude` from removing some packages that were pulled in through dependencies, you need to manually unmark them as *auto* packages. This includes OpenOffice and Vim for desktop installs:

```
# aptitude unmarkauto openoffice.org vim
```

And 2.6 kernel images if you have installed them using a kernel metapackage:

```
# aptitude unmarkauto $(dpkg-query -W 'kernel-image-2.6.*' | cut -f1)
```

Note: You can review which packages are marked as *auto* in aptitude by running:

```
# aptitude search 'i~M <package name>'
```

4.4   Preparing sources for APT

Before starting the upgrade you must set up `apt`'s configuration file for package lists, `/etc/apt/sources.list`.

`apt` will consider all packages that can be found via any "`deb`" line, and install the package with the highest version number, giving priority to the first mentioned lines (that way, in case of multiple mirror locations, you'd typically first name a local harddisk, then CD-ROMs, and then HTTP/FTP mirrors).

A release can often be referred to by both its codename (e.g. sarge, etch) and by its status name (i.e. oldstable, stable, testing, unstable). Referring to a release by its codename has the advantage that you will never be surprised by a new release and for this reason is the approach taken here. It does of course mean that you will have to watch out for release announcements yourself. If you use the status name instead, you will just see loads of updates for packages available as soon as a release has happened.

4.4.1   Adding APT Internet sources

The default configuration is set up for installation from main Debian Internet servers, but you may wish to modify `/etc/apt/sources.list` to use other mirrors, preferably a mirror that is network-wise closest to you.

Debian HTTP or FTP mirror addresses can be found at http://www.debian.org/distrib/ftplist (look at the "Full list of mirrors" section). HTTP mirrors are generally speedier than FTP mirrors.

For example, suppose your closest Debian mirror is `http://mirrors.kernel.org/debian/`. When inspecting that mirror with a web browser or FTP program, you will notice that the main directories are organized like this:

```
http://mirrors.kernel.org/debian/dists/etch/main/binary-sparc/...
http://mirrors.kernel.org/debian/dists/etch/contrib/binary-sparc/...
```

To use this mirror with `apt`, you add this line to your `sources.list` file:

```
deb http://mirrors.kernel.org/debian etch main contrib
```

Note that the '`dists`' is added implicitly, and the arguments after the release name are used to expand the path into multiple directories.

After adding your new sources, disable the previously existing "`deb`" lines in `sources.list` by placing a hash sign (#) in front of them.

4.4.2 Adding APT sources for a local mirror

Instead of using HTTP or FTP packages mirrors, you may wish to modify `/etc/apt/sources.list` to use a mirror on a local disk (possibly mounted over NFS).

For example, your packages mirror may be under `/var/ftp/debian/`, and have main directories like this:

```
/var/ftp/debian/dists/etch/main/binary-sparc/...
/var/ftp/debian/dists/etch/contrib/binary-sparc/...
```

To use this with `apt`, add this line to your `sources.list` file:

```
deb file:/var/ftp/debian etch main contrib
```

Note that the '`dists`' is added implicitly, and the arguments after the release name are used to expand the path into multiple directories.

After adding your new sources, disable the previously existing "`deb`" lines in `sources.list` by placing a hash sign (#) in front of them.

4.4.3 Adding APT source from CD-ROM or DVD

If you want to use CDs *only*, comment out the existing "`deb`" lines in `/etc/apt/sources.list` by placing a hash sign (#) in front of them.

Make sure there is a line in `/etc/fstab` that enables mounting your CD-ROM drive at the `/cdrom` mount point (the exact `/cdrom` mount point is required for `apt-cdrom`). For example, if `/dev/hdc` is your CD-ROM drive, `/etc/fstab` should contain a line like:

```
/dev/hdc /cdrom auto defaults,noauto,ro 0 0
```

Note that there must be *no spaces* between the words `defaults,noauto,ro` in the fourth field.

To verify it works, insert a CD and try running

```
# mount /cdrom     # this will mount the CD to the mount point
# ls -alF /cdrom   # this should show the CD's root directory
# umount /cdrom    # this will unmount the CD
```

Next, run:

```
# apt-cdrom add
```

for each Debian Binary CD-ROM you have, to add the data about each CD to APT's database.

4.5   Upgrading packages

The recommended way to upgrade from previous Debian GNU/Linux releases is to use the package management tool `aptitude`. This program makes safer decisions about package installations than running `apt-get` directly.

Don't forget to mount all needed partitions (notably the root and `/usr` partitions) read-write, with a command like:

```
# mount -o remount,rw /mountpoint
```

Next you should double-check that the APT source entries (in `/etc/apt/sources.list`) refer either to "`etch`" or to "`stable`". There should not be any sources entries pointing to sarge. Note: source lines for a CD-ROM will often refer to "`unstable`"; although this may be confusing, you should *not* change it.

4.5.1   Recording the session

It is strongly recommended that you use the `/usr/bin/script` program to record a transcript of the upgrade session. Then if a problem occurs, you will have a log of what happened, and if needed, can provide exact information in a bug report. To start the recording, type:

```
# script -t 2>~/upgrade-etch.time -a ~/upgrade-etch.script
```

or similar. Do not put the typescript file in a temporary directory such as `/tmp` or `/var/tmp` (files in those directories may be deleted during the upgrade or during any restart).

The typescript will also allow you to review information that has scrolled off-screen.   Just switch to VT2 (using `Alt-F2`) and, after logging in, use `less -R` `~root/upgrade-etch.script` to view the file.

After you have completed the upgrade, you can stop `script` by typing `exit` at the prompt.

If you have used the *-t* switch for `script` you can use the `scriptreplay` program to replay the whole session:

```
# scriptreplay ~/upgrade-etch.time ~/upgrade-etch.script
```

4.5.2   Updating the package list

First the list of available packages for the new release needs to be fetched. This is done by executing:

```
# aptitude update
```

Running this the first time new sources are updated will print out some warnings related to the availability of the sources. These warnings are harmless and will not appear if you rerun the command again.

4.5.3   Make sure you have sufficient space for the upgrade

You have to make sure before upgrading your system that you have sufficient hard disk space when you start the full system upgrade described in 'Upgrading the rest of the system' on page 27. First, any package needed for installation that is fetched from the network is stored in `/var/cache/apt/archives` (and the `partial/` subdirectory, during download), so you must make sure you have enough space on the file system partition that holds `/var/` to temporarily download the packages that will be installed in your system. After the download, you will probably need more space in other file system partitions in order to both install upgraded packages (which might contain bigger binaries or more data) and new packages that will be pulled in for the upgrade. If your system does not have sufficient space you might end up with an incomplete upgrade that might be difficult to recover from.

Both `aptitude` and `apt` will show you detailed information of the disk space needed for the installation. Before executing the upgrade, you can see this estimate by running:

```
# aptitude -y -s -f --with-recommends dist-upgrade
[ ... ]
XXX upgraded, XXX newly installed, XXX to remove and XXX not upgraded.
Need to get xx.xMB/yyyMB of archives. After unpacking AAAMB will be used.
Would download/install/remove packages.
```

3

If you do not have enough space for the upgrade, make sure you free up space beforehand. You can:

- Remove packages that have been previously downloaded for installation (at `/var/cache/apt/archive`). Cleaning up the package cache by running `apt-get clean` or `aptitude clean` will remove all previously downloaded package files.

- Remove old packages you no longer use. If you have `popularity-contest` installed, you can use `popcon-largest-unused` to list the packages you do not use in the system that occupy the most space. You can also use `deborphan` or `debfoster` to find obsolete packages (see 'Obsolete packages' on page 34). Alternatively you can start `aptitude` in "visual mode" and find obsolete packages under "Obsolete and Locally Created Packages".

- Remove packages taking up too much space, which are not currently needed (you can always reinstall them after the upgrade). You can list the packages that take up most of the disk space with `dpigs` (available in the `debian-goodies` package) or with `wajig` (running `wajig size`).

- Temporarily move to another system, or permanently remove, system logs residing under `/var/log/`.

Note that in order to safely remove packages, it is advisable to switch your `sources.list` back to sarge as described in 'Checking your sources list' on page 51.

4.5.4   Minimal system upgrade

Because of certain necessary package conflicts between sarge and etch, running `aptitude dist-upgrade` directly will often remove large numbers of packages that you will want to keep. We therefore recommend a two-part upgrade process, first a minimal upgrade to overcome these conflicts, then a full `dist-upgrade`.

First, run:

```
# aptitude upgrade
```

This has the effect of upgrading those packages which can be upgraded without requiring any other packages to be removed or installed.

Follow the minimal upgrade with:

---

[3]Running this command at the beginning of the upgrade process may give an error, for the reasons described in the next sections. In that case you will need to wait until you've done the minimal system upgrade as in 'Minimal system upgrade' on the current page and upgraded your kernel as in 'Upgrading the kernel' on page 26 before running this command to estimate the disk space.

```
# aptitude install initrd-tools
```

This step will automatically upgrade `libc6` and `locales` and will pull in SELinux support libraries (`libselinux1`). At this point, some running services will be restarted, including `xdm`, `gdm` and `kdm`. As a consequence, local X11 sessions will be disconnected.

The next step will vary depending on the set of packages that you have installed. These release notes give general advice about which method should be used, but if in doubt, it is recommended that you examine the package removals proposed by each method before proceeding.

Some common packages that are expected to be removed include `base-config`, `hotplug`, `xlibs`, `netkit-inetd`, `python2.3`, `xfree86-common`, and `xserver-common`. For a more complete list of packages obsoleted in etch, see 'Obsolete packages' on page 34.

Upgrading a desktop system

This upgrade path has been verified to work on systems with the sarge `desktop` task installed. It is probably the method that will give the best results on systems with the `desktop` task installed, or with the `gnome` or `kde` packages installed.

It is probably *not* the correct method to use if you do not already have the `libfam0c102` and `xlibmesa-glu` packages installed:

```
# dpkg -l libfam0c102 | grep ^ii
# dpkg -l xlibmesa-glu | grep ^ii
```

If you do have a full desktop system installed, run:

```
# aptitude install libfam0 xlibmesa-glu
```

Upgrading a system with some X packages installed

Systems with some X packages installed, but not the full `desktop` task, require a different method. This method applies in general to systems with `xfree86-common` installed, including some server systems which have `tasksel` server tasks installed as some of these tasks include graphical management tools. It is likely the correct method to use on systems which run X, but do not have the full `desktop` task installed.

```
# dpkg -l xfree86-common | grep ^ii
```

First, check whether you have the `libfam0c102` and `xlibmesa-glu` packages installed.

```
# dpkg -l libfam0c102 | grep ^ii
# dpkg -l xlibmesa-glu | grep ^ii
```

If you do not have `libfam0c102` installed, do not include `libfam0` in the following commandline. If you do not have `xlibmesa-glu` installed, do not include it in the following commandline. [4]

```
# aptitude install x11-common libfam0 xlibmesa-glu
```

Note that installing `libfam0` will also install the File Alteration Monitor (`fam`) as well as the RPC portmapper (`portmap`) if not already available in your system. Both packages will enable a new network service in the system although they can both be configured to be bound to the (internal) loopback network device.

Upgrading a system with no X support installed

On a system with no X, no additional `aptitude install` command should be required, and you can move on to the next step.

4.5.5   Upgrading the kernel

The `udev` version in etch does not support kernel versions earlier than 2.6.15 (which includes sarge 2.6.8 kernels), and the `udev` version in sarge will not work properly with the latest kernels. In addition, installing the etch version of `udev` will force the removal of `hotplug`, used by Linux 2.4 kernels.

As a consequence, the previous kernel package will probably not boot properly after this upgrade. Similarly, there is a time window during the upgrade in which `udev` has been upgraded but the latest kernel has not been installed. If the system were to be rebooted at this point, in the middle of the upgrade, it might not be bootable because of drivers not being properly detected and loaded. (See 'Prepare a safe environment for the upgrade' on page 17 for recommendations on preparing for this possibility if you are upgrading remotely.)

Unless your system has the `desktop` task installed, or other packages that would cause an unacceptable number of package removals, it is therefore recommended that you upgrade the kernel on its own at this point.

To proceed with this kernel upgrade, run:

```
# aptitude install linux-image-2.6-flavor
```

---

[4]This command will determine whether you need libfam0 and xlibmesa-glu installed, and auto-select them for you:

```
# aptitude install x11-common \ $(dpkg-query -showformat '${Package} ${Status}\n' -W
libfam0c102 xlibmesa-glu \ | grep 'ok installed$' | sed -e's/ .*//; s/c102//')
```

See 'Installing the kernel metapackage' on page 29 for help in determining which flavor of kernel package you should install.

In the desktop case, it is unfortunately not possible to ensure the new kernel package is installed immediately after the new `udev` is installed, so there is a window of unknown length when your system will have no kernel installed with full hotplug support. See 'Upgrading your kernel and related packages' on page 29 for information on configuring your system to not depend on hotplug for booting.

4.5.6 Upgrading the rest of the system

You are now ready to continue with the main part of the upgrade. Execute:

```
# aptitude dist-upgrade
```

This will perform a complete upgrade of the system, i.e. install the newest available versions of all packages, and resolve all possible dependency changes between packages in different releases. If necessary, it will install some new packages (usually new library versions, or renamed packages), and remove any conflicting obsoleted packages.

When upgrading from a set of CD-ROMs, you will be asked to insert specific CDs at several points during the upgrade. You might have to insert the same CD multiple times; this is due to inter-related packages that have been spread out over the CDs.

New versions of currently installed packages that cannot be upgraded without changing the install status of another package will be left at their current version (displayed as "held back"). This can be resolved by either using `aptitude` to choose these packages for installation or by trying `aptitude -f install` *package*.

4.5.7 Getting package signatures

After the upgrade, with the new version of `apt` you can now update your package information, which will include the new package signature checking mechanism:

```
# aptitude update
```

The upgrade will have already retrieved and enabled the signing keys for Debian's package archives. If you add other (unofficial) package sources, `apt` will print warnings related to its inability to confirm that packages downloaded from them are legitimate and have not been tampered with. For more information please see 'Package management' on page 5.

You will notice that, since you are using the new version of `apt`, it will download package differences files (`pdiff`) instead of the full package index list. For more information on this feature please read 'Slower updates of APT package index files' on page 38.

4.5.8   Possible issues during upgrade

If an operation using `aptitude`, `apt-get`, or `dpkg` fails with the error

```
E: Dynamic MMap ran out of room
```

the default cache space is insufficient. You can solve this by either removing or commenting lines you don't need in `/etc/apt/sources.list` or by increasing the cache size. The cache size can be increased by setting `APT::Cache-Limit` in `/etc/apt/apt.conf`. The following command will set it to a value that should be sufficient for the upgrade:

```
# echo 'APT::Cache-Limit "12500000";' >> /etc/apt/apt.conf
```

This assumes that you do not yet have this variable set in that file.

Sometimes it's necessary to enable the `APT::Force-LoopBreak` option in APT to be able to temporarily remove an essential package due to a Conflicts/Pre-Depends loop. `aptitude` will alert you of this and abort the upgrade. You can work around that by specifying `-o APT::Force-LoopBreak=1` option on `aptitude` command line.

It is possible that a system's dependency structure can be so corrupt as to require manual intervention. Usually this means using `aptitude` or

```
# dpkg --remove package_name
```

to eliminate some of the offending packages, or

```
# aptitude -f install
# dpkg --configure --pending
```

In extreme cases you might have to force re-installation with a command like

```
# dpkg --install /path/to/package_name.deb
```

File conflicts should not occur if you upgrade from a "pure" sarge system, but can occur if you have unofficial backports installed. A file conflict will result in an error like:

```
Unpacking <package-foo> (from <package-foo-file>) ...
dpkg: error processing <package-foo> (--install):
 trying to overwrite '<some-file-name>',
 which is also in package <package-bar>
dpkg-deb: subprocess paste killed by signal (Broken pipe)
 Errors were encountered while processing:
 <package-foo>
```

You can try to solve a file conflict by forcibly removing the package mentioned on the *last* line of the error message:

```
# dpkg -r --force-depends package_name
```

After fixing things up, you should be able to resume the upgrade by repeating the previously described `aptitude` commands.

During the upgrade, you will be asked questions regarding the configuration or re-configuration of several packages. When you are asked if any file in the `/etc/init.d` or `/etc/terminfo` directories, or the `/etc/manpath.config` file should be replaced by the package maintainer's version, it's usually necessary to answer 'yes' to ensure system consistency. You can always revert to the old versions, since they will be saved with a `.dpkg-old` extension.

If you're not sure what to do, write down the name of the package or file and sort things out at a later time. You can search in the typescript file to review the information that was on the screen during the upgrade.

4.6   Upgrading your kernel and related packages

This section explains how to upgrade your kernel and identifies potential issues related to this upgrade. You can either install one of the `linux-image-*` packages provided by Debian, or compile a customized kernel from source.

Note that a lot of information in this section is based on the assumption that you will be using one of the modular Debian kernels, together with `initramfs-tools` and `udev`. If you choose to use a custom kernel that does not require an initrd or if you use a different initrd generator, some of the information may not be relevant for you.

Note also that if `udev` is *not* installed on your system, it is still possible to use `hotplug` for hardware discovery.

If you are currently using a 2.4 kernel, you should also read 'Upgrading to a 2.6 kernel' on page 39 carefully.

4.6.1   Installing the kernel metapackage

When you dist-upgrade from sarge to etch, it is strongly recommended that you install a new linux-image-2.6-* metapackage. This package may be installed automatically by the dist-upgrade process. You can verify this by running:

```
# dpkg -l "linux-image*" | grep ^ii
```

If you do not see any output, then you will need to install a new linux-image package by hand. To see a list of available linux-image-2.6 metapackages, run:

```
# apt-cache search linux-image-2.6- | grep -v transition
```

If you are unsure about which package to select, run `uname -r` and look for a package with a similar name. For example, if you see '2.4.27-3-686', it is recommended that you install `linux-image-2.6-686`. You may also use `apt-cache` to see a long description of each package in order to help choose the best one available. For example:

```
# apt-cache show linux-image-2.6-686
```

You should then use `aptitude install` to install it. Once this new kernel is installed you should reboot at the next available opportunity to get the benefits provided by the new kernel version.

For the more adventurous there is an easy way to compile your own custom kernel on Debian GNU/Linux. Install the `kernel-package` tool and read the documentation in `/usr/share/doc/kernel-package`.

4.6.2   Upgrading from a 2.6 kernel

If you are currently running a 2.6 series kernel from sarge this upgrade will take place automatically after you do a full upgrade of the system packages (as described in 'Upgrading packages' on page 22).

If possible, it is to your advantage to upgrade the kernel package separately from the main `dist-upgrade` to reduce the chances of a temporarily non-bootable system. See 'Upgrading the kernel' on page 26 for a description of this process. Note that this should only be done after the minimal upgrade process described in 'Minimal system upgrade' on page 24.

You can also take this step if you are using your own custom kernel and want to use the kernel available in etch. If your kernel version is not supported by `udev` then it is recommended that you upgrade after the minimal upgrade. If your version is supported by `udev` you can safely wait until after the full system upgrade.

4.6.3   Upgrading from a 2.4 kernel

If you have a 2.4 kernel installed, and your system relies on `hotplug` for its hardware detection you should first upgrade to a 2.6 series kernel from sarge before attempting the upgrade. Make sure that the 2.6 series kernel boots your system and all your hardware is properly detected before you perform the upgrade. The `hotplug` package is removed from the system (in favor of `udev`) when you do a full system upgrade. If you do not do the kernel upgrade before this your system might not boot up properly from this point on. Once you have done an upgrade to a 2.6 series kernel in sarge you can do a kernel upgrade as described in 'Upgrading from a 2.6 kernel' on the current page.

If your system does not rely on `hotplug`[5] you can delay the kernel upgrade to after you have done a full system upgrade, as described in 'Upgrading the rest of the system' on page 27. Once your system has been upgraded you can then do the following (changing the kernel package name to the one most suited to your system by substituting *<flavor>*):

```
# aptitude install linux-image-2.6-<flavor>
```

4.6.4   Device enumeration reordering

etch features a more robust mechanism for hardware discovery than previous releases. However, this may cause changes in the order devices are discovered on your system, affecting the order in which device names are assigned. For example, if you have two network adapters that are associated with two different drivers, the devices eth0 and eth1 refer to may be swapped. Please note that the new mechanism means that if you e.g. exchange ethernet adapters in a running etch system, the new adapter will also get a new interface name.

For network devices, you can avoid this reordering by using `udev` rules, more specifically, through the definitions at `/etc/udev/rules.d/z25_persistent-net.rules`[6]. Alternatively you can use the `ifrename` utility to bind physical devices to specific names at boot time. See `ifrename(8)` and `iftab(5)` for more information. The two alternatives (`udev` and `ifrename`) should not be used at the same time.

For storage devices, you can avoid this reordering by using `initramfs-tools` and configuring it to load storage device driver modules in the same order they are currently loaded. To do this, identify the order the storage modules on your system were loaded by looking at the output of `lsmod`. `lsmod` lists modules in the reverse order that they were loaded in, i.e., the first module in the list was the last one loaded. Note that this will only work for devices which the kernel enumerates in a stable order (like PCI devices).

However, removing and reloading modules after initial boot will affect this order. Also, your kernel may have some drivers linked statically, and these names will not appear in the output of `lsmod`. You may be able to decipher these driver names and load order from looking at `/var/log/kern.log`, or the output of `dmesg`.

Add these module names to `/etc/initramfs-tools/modules` in the order they should be loaded at boot time. Some module names may have changed between sarge and etch. For example, sym53c8xx_2 has become sym53c8xx.

You will then need to regenerate your initramfs image(s) by executing `update-initramfs -u -k all`.

Once you are running a etch kernel and `udev`, you may reconfigure your system to access disks by an alias that is not dependent upon driver load order. These aliases reside in the `/dev /disk/` hierarchy.

---

[5]You can have the kernel modules needed by your system loaded statically through proper configuration of `/etc/modules`

[6]The   rules   there   are   automatically   generated   by   the   script   `/etc/udev/rules.d /z45_persistent-net-generator.rules` to have persistent names for network interfaces. Delete this symlink to disable persistent device naming for NICs by `udev`.

4.6.5   Boot timing issues

If an initrd created with `initramfs-tools` is used to boot the system, in some cases the creation of device files by `udev` can happen too late for the boot scripts to act on.

The usual symptoms are that the boot will fail because the root file system cannot be mounted and you are dropped into a debug shell, but that when you check afterwards, all devices that are needed are present in `/dev`. This has been observed in cases where the root file system is on a USB disk or on RAID, especially if lilo is used.

A workaround for this issue is to use the boot parameter `rootdelay=9`. The value for the timeout (in seconds) may need to be adjusted.

4.7   Things to do before rebooting

When `aptitude dist-upgrade` has finished, the "formal" upgrade is complete, but there are some other things that should be taken care of *before* the next reboot.

4.7.1   Converting from devfs

Debian kernels no longer include support for `devfs`, so `devfs` users will need to convert their systems manually before booting an etch kernel.

If you see the string 'devfs' in `/proc/mounts`, you are most likely using `devfs`. Any configuration files that reference `devfs`-style names will need to be adjusted to use `udev`-style names. Files that are likely to refer to `devfs`-style device names include `/etc/fstab`, `/etc/lilo.conf`, `/boot/grub/menu.lst`, and `/etc/inittab`.

More information about potential issues is available in bug report #341152 (`http://bugs.debian.org/341152`).

4.7.2   Possible missing drivers in initrd

The etch kernels do not yet have full sysfs support for the native sparc sbus. `initramfs-tools` relies on this to include drivers for disk controllers in the initrd. If a driver is not included in the initrd, your system may fail to boot.

If your system uses the `esp` or `qlogicpti` module to access your hard disks, you will need to include that module in `/etc/initramfs-tools/modules` and regenerate the initrd before you reboot your system. The initrd can be regenerated using:

```
# update-initramfs -u -k all
```

4.7.3   Upgrading mdadm

mdadm now needs a configuration file to assemble MD arrays (RAID) from the initial ramdisk and during the system initialisation sequence. Please make sure to read and act upon the instructions in `/usr/share/doc/mdadm/README.upgrading-2.5.3.gz` after the package has been upgraded **and before you reboot**. The latest version of this file is available at `http://svn.debian.org/wsvn/pkg-mdadm/mdadm/trunk/debian/README.upgrading-2.5.3?op=file`; please consult it in case of problems.

4.7.4   Conflict between tulip and dmfe drivers

On some Netras (e.g. on Netra X1) udev loads both tulip and dmfe drivers, which claim support for the same PCI IDs (#334104 (`http://bugs.debian.org/334104`)). The tulip driver is correct one for Netra. If this happens, you should blacklist the dmfe driver and reboot, and/or remove both dmfe and tulip (modprobe -r) and then modprobe tulip only.

4.8   Preparing for the next release

After the upgrade there are several things you can do to prepare for the next release.

- If using `grub`, edit `/etc/kernel-img.conf` and adjust the location of the `update-grub` program changing `/sbin/update-grub` to `/usr/sbin/update-grub`.

- If the new kernel image metapackage was pulled in as a dependency of the old one, it will be marked as automatically installed, which should be corrected:

    ```
    # aptitude unmarkauto $(dpkg-query -W 'linux-image-2.6-*' | cut -f1)
    ```

- Remove sarge's kernel metapackages by running:

    ```
    # aptitude purge kernel-image-2.6-<flavor>
    ```

- Move any configuration options from `/etc/network/options` to `/etc/sysctl.conf`. Please see `/usr/share/doc/netbase/README.Debian` for details.

- Remove obsolete and unused packages as described in 'Obsolete packages' on the next page. You should review which configuration files they use and consider purging the packages to remove their configuration files

4.9 Deprecated packages

With the release of `Lenny` a bigger number of server packages will be deprecated, thus updating to newer versions of those now will save you from trouble when updating to `Lenny`.

This includes the following packages:

- apache (1.x), successor is apache2

- bind8, successor is bind9

- php4, successor is php5

- postgresql-7.4, successor is postgresql-8.1

- exim 3, successor is exim4

4.10 Obsolete packages

Introducing several thousand new packages, etch also retires and omits more than two thousand old packages that were in sarge. It provides no upgrade path for these obsolete packages. While nothing prevents you from continuing to use an obsolete package where desired, the Debian project will usually discontinue security support for it a year after etch's release[7], and will not normally provide other support in the meantime. Replacing them with available alternatives, if any, is recommended.

There are many reasons why packages might have been removed from the distribution: they are no longer maintained upstream; there is no longer a Debian Developer interested in maintaining the packages; the functionality they provide has been superseded by different software (or a new version); or they are no longer considered suitable for etch due to bugs in them. In the latter case, packages might still be present in the "unstable" distribution.

Detecting which packages in an updated system are "obsolete" is easy since the package management front-ends will mark them as such. If you are using `aptitude`, you will see a listing of these packages in the "Obsolete and Locally Created Packages" entry. `dselect` provides a similar section but the listing it presents might differ. Also, if you have used `aptitude` to manually install packages in sarge it will have kept track of those packages you manually installed and will be able to mark as obsolete those packages pulled in by dependencies alone which are no longer needed if a package has been removed. Also, `aptitude`, unlike `deborphan` will not mark as obsolete packages that you manually installed, as opposed to those that were automatically installed through dependencies.

There are additional tools you can use to find obsolete packages such as `deborphan`, `debfoster` or `cruft`. `deborphan` is highly recommended, although it will (in default mode) only report obsolete libraries: packages in the "libs" or "oldlibs" sections that are not used

---

[7]Or for as long as there is not another release in that time frame. Typically only two stable releases are supported at any given time.

by any other packages. Do not blindly remove the packages these tools present, especially if you are using aggressive non-default options that are prone to produce false positives. It is highly recommended that you manually review the packages suggested for removal (i.e. their contents, size and description) before you remove them.

The Debian Bug Tracking System (`http://bugs.debian.org/`) often provides additional information on why the package was removed. You should review both the archived bug reports for the package itself and the archived bug reports for the ftp.debian.org pseudo-package (`http://bugs.debian.org/cgi-bin/pkgreport.cgi?pkg=ftp.debian.org\&archive=yes`).

4.10.1   Dummy packages

Some packages from sarge have been split into several packages in etch, often to improve system maintainability. To ease the upgrade path in such cases, etch often provides "dummy" packages: empty packages that have the same name as the old package in sarge with dependencies that cause the new packages to be installed. These "dummy" packages are considered obsolete packages after the upgrade and can be safely removed.

Most (but not all) dummy packages' descriptions indicate their purpose. Package descriptions for dummy packages are not uniform, however, so you might also find `deborphan` with the `--guess` options useful to detect them in your system. Note that some dummy packages are not intended to be removed after an upgrade but are, instead, used to keep track of the current available version of a program over time.

Chapter 5

Issues to be aware of for etch

5.1 Potential problems

Sometimes, changes have side-effects we cannot reasonably avoid, or we expose bugs somewhere else. We document here the issues we are aware of. Please also read the errata, the relevant packages' documentation, bug reports and other information mentioned in 'Further reading' on page .

5.1.1 Problems with devices related to udev

Although `udev` has been tested extensively, you may experience minor problems with some devices that will need to be fixed. The most common problems are changed permission and/or ownership of a device. In some cases a device may not be created by default (e.g. `/dev /video` and `/dev/radio`).

`udev` provides configuration mechanisms to deal with these issues. See `udev(8)` and `/etc /udev` for further information.

5.1.2 Some applications may no longer work with a 2.4 kernel

Some applications in etch may no longer work with a 2.4 kernel, for example because they require `epoll()` support, which is not available in 2.4 kernels. Such applications may either not work at all or not work correctly until the system has been rebooted with a 2.6 kernel.

One example is the HTTP proxy `squid`.

5.1.3 Certain network sites cannot be reached by TCP

Since 2.6.17, Linux aggressively uses TCP window scaling which is specified in RFC 1323. Some servers have a broken behavior, and announce wrong window sizes for themselves.

For more details, please see the bug reports #381262 (http://bugs.debian.org/381262), #395066 (http://bugs.debian.org/395066), #401435 (http://bugs.debian.org/401435).

There are usually two workarounds to these problems: either revert the maximum allowed TCP window sizes to a smaller value (preferable) or turn off TCP window scaling altogether (deprecated). See the example commands in the debian-installer errata page (http://www.debian.org/devel/debian-installer/errata).

5.1.4 Slower updates of APT package index files

By default, the etch version of apt uses a new way to update APT package index files (when you run aptitude update) which downloads differences files (instead of the full package index file) called pdiff. This new feature should use less bandwidth and be faster for most systems. Unfortunately, it can also have the opposite effect of making the updates slower on systems with fast network connections (or a very nearby mirror) which are infrequently updated, as it might take more time for the system to merge the differences files than to download a full package index. It is possible to disable this feature by adding Acquire::Pdiffs "false"; to the /etc/apt/apt.conf configuration file.

This change mostly affects users of the *unstable* and *testing* branch of Debian GNU/Linux, due to the changing nature of these archives. Users of etch will notice this feature mainly when updating their package status for the security archive.

5.1.5 Asynchronous network initialization may cause unpredictable behavior

On systems which use udev to load drivers for network interfaces, it is possible due to the asynchronous nature of udev that the network driver will not be loaded before /etc/init.d/networking runs on system boot. Although including allow-hotplug to /etc/network/interfaces (in addition to auto) will ensure that the network interface is enabled once it becomes available, there is no guarantee that this will finish before the boot sequence begins to start network services, some of which may not behave correctly in the absence of the network interface.

5.1.6 Trouble when using WPA secured wireless networks

In sarge, the wpasupplicant package was set up as a system service, configured via /etc/default/wpasupplicant and a user-provided /etc/wpasupplicant.conf.

In etch, /etc/init.d/wpasupplicant has been dropped and the Debian package now integrates with /etc/network/interfaces, similar to other packages such as wireless-tools. This means wpasupplicant no longer provides a system service directly.

For information on configuring wpasupplicant please refer to /usr/share/doc/wpasupplicant/README.modes.gz, which gives examples for /etc/network

/interfaces files. Updated information about the usage of the wpasupplicant package in Debian can be found in the Debian Wiki (http://wiki.debian.org/WPA).

5.1.7 Problems with non-ASCII characters in filenames

Mounting vfat, ntfs or iso9660 file systems with files that include non-ASCII characters in their filenames will give failures when one tries to use the filenames unless mounting is done with the utf8 option. An indication might be the following failure: 'Invalid or incomplete multibyte or wide character'. A possible solution is to use defaults,utf8 as mount options for vfat, ntfs and iso9660 file systems when they contain filenames with non-ASCII characters.

Note that the Linux kernel does not support case-insensitive filename handling for vfat when the utf8 option is used.

5.1.8 Sound stops working

In rare cases the sound might stop working after the upgrade. If this happens, go through the alsa checklist: run alsaconf as root user, add your user to the audio group, use alsamixer and make sure levels are up and it is unmuted, make sure arts or esound stopped, make sure OSS modules unloaded, make sure speakers are on, check whether the command cat /dev/urandom > /dev/dsp works for root.

5.2 Upgrading to a 2.6 kernel

The 2.6 kernel series contains major changes from the 2.4 series. Modules have been renamed and a lot of drivers have been partially or sometimes almost completely rewritten. Upgrading to a 2.6 kernel from an earlier version is therefore not a process to be undertaken lightly. This section aims to make you aware of some of the issues you may face.

If you compile your own kernel from source, make sure you install module-init-tools before you reboot with the 2.6 kernel. This package replaces modutils for 2.6 kernels. If you install one of the Debian linux-image packages, this package will be installed automatically because of dependencies.

If you use *LVM*, you should also install lvm2 before you reboot as the 2.6 kernel does not directly support LVM1. To access LVM1 volumes, the compatibility layer of lvm2 (the dm-mod module) is used. You can leave lvm10 installed; the init scripts will detect which kernel is used and execute the appropriate version.

If you have entries in the /etc/modules file (the list of modules to be loaded during system boot), be aware that some module names may have changed. If this happens you will have to update this file with the new module names.

Once you have installed your 2.6 kernel, but before you reboot, make sure you have a recovery method. First, make sure that the bootloader configuration has entries for both the new kernel

and the old, working 2.4 kernel. You should also ensure you have a "rescue" floppy or CD-ROM to hand, in case misconfiguration of the bootloader prevents you from booting the old kernel.

5.2.1 Keyboard configuration

The most invasive change in the 2.6 kernels is a fundamental change of the input layer. This change makes all keyboards look like "normal" PC keyboards. This means that if you currently have a different type of keyboard selected (e.g. a USB-MAC or Sun keyboard), you will very likely end up with a non-working keyboard after rebooting with the new 2.6 kernel.

If you can SSH into the box from another system, you can resolve this issue by running `dpkg-reconfigure console-data`, choosing the option "Select keymap from full list" and selecting a "pc" keyboard.

If your console keyboard is affected, you will probably also need to reconfigure your keyboard for the X Window System. You can do this either by running `dpkg-reconfigure xserver-xorg` or by editing `/etc/X11/xorg.conf` directly. Don't forget to read the documentation referred to in 'Things to do before rebooting' on page 32.

Note that if you are using a USB keyboard, this may be configured as either a "normal" PC keyboard or as a USB-MAC keyboard. In the first case you will not be affected by this issue.

5.2.2 Mouse configuration

Again because of the changes in the input layer, you may have to reconfigure the X Window System and `gpm` if your mouse is not working after upgrading to a 2.6 kernel. The most likely cause is that the device which gets the data from the mouse has changed. You may also need to load different modules.

If you currently have X configured for `/dev/sunmouse`, you probably need to change this to `/dev/psaux`.

5.2.3 Sound configuration

For the 2.6 kernel series the ALSA sound drivers are recommended over the older OSS sound drivers. ALSA sound drivers are provided as modules by default. In order for sound to work, the ALSA modules appropriate for your sound hardware need to be loaded. In general this will happen automatically if you have, in addition to the `alsa-base` package, either the `hotplug` package or the `discover` package installed. The `alsa-base` package also "blacklists" OSS modules to prevent `hotplug` and `discover` from loading them. If you have OSS modules listed in `/etc/modules`, you should remove them.

5.3   XFree86 to X.Org transition

The transition to X.Org involves some structural changes. In case all installed packages are from Debian and also included in etch, the upgrade should work without problems. However, experience has shown that there are a few changes to be aware of, as they can potentially cause issues during the upgrade.

The most important change is that `/usr/X11R6/bin` has been dropped and only remains as a symlink to `/usr/bin`. This means the directory has to be empty at the time the new packages are installed. The new packages conflict with most packages that used `/usr/X11R6/bin`, but in some cases manual intervention may be needed. Please remember to not run the distribution upgrade from within an X session.

In case the upgrade aborts during X.Org installation, you should check if any files are still left in `/usr/X11R6/bin`. You can then use `dpkg -S` to find out which Debian package installed that file (if any), and remove such packages with `dpkg --remove`. Please make a note which packages you remove, so that you can install substitute packages later on. Before continuing with the upgrade, all files in `/usr/X11R6/bin` need to be removed.

Please read http://wiki.debian.org/Xorg69To7 for more details and other issues.

If you experience problems with X.Org after restarting, it might be also worth to restart the font server by running `/etc/init.d/xfs restart`. This happens due to `/etc/X11/fs /xfs.options` containing a line with `no-restart-on-upgrade`, but the font paths have changed.

5.4   No support for 8-bit displays in many applications

After the upgrade to the X.Org and the latest libraries, X terminals which can only represent colors 8 bits depth will not work. This is because the Cairo 2D vector graphics library (`libcairo2`) doesn't have 8-bit pseudocolor support. This library is used by the GNOME and Xfce desktops as well as by many desktop applications compiled with the Gtk2+ toolkit, such as `abiword`.

Known systems that are affected by this include some Sun machines and X terminals from Tektronix, NCD, IBM and SGI, as well as some other remote X windowing systems. You should configure these terminals to use 16-bit colour, if possible.

More information is available in Freedesktop's bug #4945 (https://bugs.freedesktop.org/show_bug.cgi?id=4945).

5.5   Upgrading from exim to exim4

One of the packages that has been obsoleted by the etch release is the Mail Transfer Agent (MTA) `exim`, which has been replaced by the completely new package `exim4`.

exim (version 3.xx) has been unmaintained upstream for years, and Debian has dropped support for that version as well. If you are still using exim 3.xx, please upgrade your exim installation to exim4 manually. Since exim4 is already part of sarge, you can choose to do the upgrade on your sarge system before the upgrade to etch, or after the etch upgrade at your convenience. Just remember that your old exim package is not going to be upgraded and that it won't get security support after support for sarge has been discontinued.

Note that, depending on your configuration of debconf, you may not be asked any configuration question during installation of exim4. If no questions are asked, the system will default to a 'local delivery' setup. Configuration is possible using the command dpkg-reconfigure exim4-config.

The exim4 packages in Debian are extensively documented. The package's home page is http://wiki.debian.org/PkgExim4 on the Debian Wiki, and the README file can be found at http://pkg-exim4.alioth.debian.org/README/README.Debian.html and inside the packages as well.

The README file has a chapter about Packaging, which explains the different package variations we offer, and it has a chapter about Updating from Exim 3, which will help you in doing the actual transition.

5.6   Upgrading apache2

Apache has been upgraded to the new version 2.2. Although this shouldn't impact the average user, there are some potential issues to be aware of.

http://httpd.apache.org/docs/2.2/upgrading.html contains the upstream changes. Please read this page, and remember that especially:

- all modules need to be recompiled

- authorization modules have been resorted and renamed

- some configuration options have been renamed

Debian-specific changes include that the string SSL is no longer defined, as ssl is now supported by the default package.

If you are using the experimental ITK MPM (from the apache2-mpm-itk package), the cgi module will not be correctly enabled by default. To properly enable it, you will need to manually disable mod_cgid and enable mod_cgi:

```
# cd /etc/apache2/mods-enabled
# rm cgid.conf cgid.load
# ln -s ../mods-available/cgi.load .
# /etc/init.d/apache2 force-reload
```

5.7 Upgrading Zope and Plone

Zope and all related products have been updated. Many products were also dropped from the distribution (either because they were obsoleted, or because they are incompatible with the newer Zope, CMF or Plone).

Unfortunately there is no easy and guaranteed way to upgrade a complex `zope` or `plone` server. Even though Plone includes a migration tool, experience has shown that automatic migrations can easily fail.

For this reason, users are recommended to set up their system so they can continue to run the sarge installation of Zope/Plone alongside the new etch versions while testing the migration.

The easiest and safest way to achieve this, is to make a copy of your sarge system to another hard disk or partition, and then upgrade only one of the two copies. You can then use `chroot` to run the sarge version in parallel to the etch version.

It is not possible to have the old and new versions of Zope/Plone installed together on an etch system, partly because the old packages depend on `python2.3` which cannot be installed together with `python2.4`.

5.8 Wildcard expansion (globbing) with GNU tar

Previous versions of GNU `tar` assumed shell-style globbing when extracting files from or listing an archive. For example:

```
tar xf foo.tar '*.c'
```

would extract all files whose names end in '.c'. This behavior was not documented and was incompatible with traditional `tar` implementations. Therefore, starting from version 1.15.91, GNU `tar` no longer uses globbing by default. For example, the above invocation is now interpreted as a request to extract from the archive the file named '*.c'.

See `/usr/share/doc/tar/NEWS.gz` for further information.

5.9 NIS and Network Manager

The version of `ypbind` included with `nis` for etch contains support for Network Manager. This support causes `ypbind` to disable NIS client functionality when Network Manager reports that the computer is disconnected from the network. Since Network Manager will usually report that the computer is disconnected when it is not in use, NIS users with NIS client systems should ensure that Network Manager support is disabled on those systems.

This can be done by either uninstalling the `network-manager` package, or editing `/etc/default/nis` to add `-no-dbus` to `YPBINDARGS`.

The use of `-no-dbus` is the default for new installs of Debian, but was not the default in previous releases.

5.10 Deprecated insecure php configurations

For many years, turning on the `register_globals` settings in PHP has been known to be insecure and dangerous, and this option has defaulted to off for some time now. This configuration is now finally deprecated on Debian systems as too dangerous. The same applies to flaws in `safe_mode` and `open_basedir`, which have also been unmaintained for some time.

Starting with this release, the Debian security team does not provide security support for a number of PHP configurations which are known to be insecure. Most importantly, issues resulting from `register_globals` being turned on will no longer be addressed.

If you run legacy applications that require `register_globals`, enable it for the respective paths only, e.g. through the Apache configuration file. More information is available in the `README.Debian.security` file in the PHP documentation directory (`/usr/share/doc/php4`, `/usr/share/doc/php5`).

5.11 Security status of Mozilla products

The Mozilla programs `firefox` and `thunderbird` (rebranded in Debian to `iceweasel` and `icedove`, respectively), are important tools for many users. Unfortunately the upstream security policy is to urge users to update to new upstream versions, which conflicts with Debian's policy of not shipping large functional changes in security updates. We cannot predict it today, but during the lifetime of etch the Debian Security Team may come to a point where supporting Mozilla products is no longer feasible and announce the end of security support for Mozilla products. You should take this into account when deploying Mozilla and consider alternatives available in Debian if the absence of security support would pose a problem for you.

5.12 KDE desktop

KDE media handling has changed in the version available in etch from using `device: /` to `media: /`. Some user configuration files might have stored `device: /` links in them which should be adapted. Notably, `~/.kde/share/apps/konqsidebartng/virtual_folders/services` contains this reference and can be safely deleted as it will not be created when setting up new users.

There have been many changes in the KDE desktop environment from the version shipped in sarge to the version in etch, you can find more information in the KDE 3.5 Release Notes (http://www.kde.org/announcements/announce-3.5.php).

5.13   GNOME desktop changes and support

If you used the GNOME desktop in sarge you will not benefit from some of the changes introduced in the default configuration in Debian for etch. In some extreme cases the GNOME desktop might not properly handle your old configuration and might not behave properly.

If you have not heavily invested in configuring your GNOME desktop you might want to move the `.gconf` directory in user's home directories to a different name (such as `.gconf.old`) so that it gets recreated, with the default configuration for etch, upon starting a new session.

With the release of etch, Debian no longer contains packages for most of the obsolete version 1 release of GNOME, although some packages remain in order to support some Debian packages which have not yet been updated to GNOME 2. Packages for GTK1.2 remain fully maintained.

There have been many changes in the GNOME desktop environment from the version shipped in sarge to the version in etch, you can find more information in the GNOME 2.14 Release Notes (http://www.gnome.org/start/2.14/notes/en/).

5.14   Default editor

If you were using `vim` as your default editor, this may be changed to `nano` during the upgrade.

Administrators who wish to change the default editor for all users will have to update the alternatives system using:

```
# update-alternatives --config editor
```

Users wishing to change the default editor can define the environment variable *EDITOR* by introducing the following lines in their own profiles:

```
EDITOR=vi
export EDITOR
alias editor=$EDITOR
```

5.15   Message of the day

`/etc/motd` is now a symlink to `/var/run/motd` which is rebuilt by `/etc/init.d/bootmisc.sh` from a template, `/etc/motd.tail`, at each reboot. It means that changes made to `/etc/motd` will be lost. Changes made into `/etc/motd.tail` are not automatically applied to `/etc/motd` other than at reboot.

Also, the EDITMOTD variable at `/etc/default/rcS` no longer has any effect. If you wish to disable updating of the motd, or want to maintain your own content for the message of the day you just have to point the `/etc/motd` symlink to a different file such as `/etc/motd.static` and make your changes there.

5.16   Not default support for unicode in emacs21*

Emacs21 and emacs21-nox are not configured to use Unicode by default. For more information and a workaround please see Bug #419490 (`http://bugs.debian.org/419490`).

Chapter 6

More information on Debian GNU/Linux

6.1 Further reading

Beyond these release notes and the installation guide, further documentation on Debian GNU/Linux is available from the Debian Documentation Project (DDP), whose goal is to create high-quality documentation for Debian users and developers. Documentation, including the Debian Reference, Debian New Maintainers Guide, and Debian FAQ are available, and many more. For full details of the existing resources see the DDP website (`http://www.debian.org/doc/ddp`).

Documentation for individual packages is installed into `/usr/share/doc/package`. This may include copyright information, Debian specific details and any upstream documentation.

6.2 Getting help

There are many sources of help, advice and support for Debian users, but these should only be considered if research into documentation of the issue has exhausted all sources. This section provides a short introduction into these which may be helpful for new Debian users.

6.2.1 Mailing lists

The mailing lists of most interest to Debian users are the debian-user list (English) and other debian-user-*language* lists (for other languages). For information on these lists and details of how to subscribe see `http://lists.debian.org/`. Please check the archives for answers to your question prior to posting and also adhere to standard list etiquette.

6.2.2   Internet Relay Chat

Debian has an IRC channel dedicated to the support and aid of Debian users located on the OFTC IRC network. To access the channel, point your favorite IRC client at irc.debian.org and join #debian.

Please follow the channel guidelines, respecting other users fully. The guidelines are available at the Debian Wiki (http://wiki.debian.org/DebianIRC).

For more information on OFTC please visit the website (http://www.oftc.net/).

6.3   Reporting bugs

We strive to make Debian GNU/Linux a high quality operating system, however that does not mean that the packages we provide are totally free of bugs. Consistent with Debian's "open development" philosophy and as a service to our users, we provide all the information on reported bugs at our own Bug Tracking System (BTS). The BTS is browseable at bugs.debian.org (http://bugs.debian.org/).

If you find a bug in the distribution or in packaged software that is part of it, please report it so that it can be properly fixed for future releases. Reporting bugs requires a valid email address. We ask for this so that we can trace bugs and developers can get in contact with submitters should additional information be needed.

You can submit a bug report using the program reportbug or manually using email. You can read more about the Bug Tracking System and how to use it by reading the reference cards (available at /usr/share/doc/debian if you have doc-debian installed) or online at the Bug Tracking System (http://bugs.debian.org/).

6.4   Contributing to Debian

You do not need to be an expert to contribute to Debian. By assisting users with problems on the various user support lists (http://lists.debian.org/) you are contributing to the community. Identifying (and also solving) problems related to the development of the distribution by participating on the development lists (http://lists.debian.org/) is also extremely helpful. To maintain Debian's high quality distribution, submit bugs (http://bugs.debian.org/) and help developers track them down and fix them. If you have a way with words then you may want to contribute more actively by helping to write documentation (http://www.debian.org/doc/ddp) or translate (http://www.debian.org/international/) existing documentation into your own language.

If you can dedicate more time, you could manage a piece of the Free Software collection within Debian. Especially helpful is if people adopt or maintain items that people have requested for inclusion within Debian. The Work Needing and Prospective Packages database (http://www.debian.org/devel/wnpp/) details this information. If you have an interest in specific

groups then you may find enjoyment in contributing to some of Debian's subprojects which include ports to particular architectures, Debian Jr. (http://www.debian.org/devel/debian-jr/) and Debian Med (http://www.debian.org/devel/debian-med/).

In any case, if you are working in the free software community in any way, as a user, programmer, writer or translator you are already helping the free software effort. Contributing is rewarding and fun, and as well as allowing you to meet new people it gives you that warm fuzzy feeling inside.

Appendix A

Managing your sarge system

This appendix contains information on how to make sure you can install or upgrade sarge packages before you upgrade to etch. This should only be necessary in specific situations.

A.1 Upgrading your sarge system

Basically this is no different than any other upgrade of sarge you've been doing. The only difference is that you first need to make sure your package list still contains sarge packages as explained in 'Checking your sources list' on this page.

If you upgrade your system using a Debian mirror, it will automatically be upgraded to the latest sarge point release.

A.2 Checking your sources list

If any of the lines in your `/etc/apt/sources.list` refer to 'stable', you are effectively already "using" etch. If you have already run `apt-get update`, you can still get back without problems following the procedure below.

If you have also already installed packages from etch, there probably is not much point in installing packages from sarge anymore. In that case you will have to decide for yourself whether you want to continue or not. It is possible to downgrade packages, but that is not covered here.

Open the file `/etc/apt/sources.list` with your favorite editor (as root) and check all lines beginning with `deb http:` or `deb ftp:` for a reference to "`stable`". If you find any, change `stable` to `sarge`.

If you have any lines starting with `deb file:`, you will have to check for yourself if the location they refer to contains a sarge or an etch archive.

Important! Do not change any lines that begin with `deb cdrom:`. Doing so would invalidate the line and you would have to run `apt-cdrom` again. Do not be alarmed if a 'cdrom' source line refers to "`unstable`". Although confusing, this is normal.

If you've made any changes, save the file and execute

```
# apt-get update
```

to refresh the package list.