

Release Notes for Debian GNU/Linux 5.0 (lenny), ARM EABI

The Debian Documentation Project (<http://www.debian.org/doc/>)

September 18, 2009

Release Notes for Debian GNU/Linux 5.0 (lenny), ARM EABI

Published 2009-02-14

This document is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License, version 2, as published by the Free Software Foundation.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

The license text can also be found at <http://www.gnu.org/copyleft/gpl.html> and `/usr/share/common-licenses/GPL-2` on Debian GNU/Linux.

Contents

1	Introduction	3
1.1	Reporting bugs on this document	3
1.2	Contributing upgrade reports	3
1.3	Sources for this document	4
2	What's new in Debian GNU/Linux 5.0	5
2.1	What's new for ARM?	5
2.2	What's new in the distribution?	6
2.2.1	Package management	6
2.2.2	The proposed-updates section	7
2.3	System improvements	7
2.4	Major kernel-related changes	8
2.4.1	Changes in kernel packaging	8
2.5	Emdebian 1.0 (based on Debian GNU/Linux lenny 5.0)	8
2.6	Netbook support	8
2.7	Java now in Debian	8
3	Installation System	9
3.1	What's new in the installation system?	9
3.1.1	Major changes	9
3.1.2	Automated installation	10
4	Upgrades from previous releases	11
4.1	Preparing for the upgrade	11
4.1.1	Back up any data or configuration information	11
4.1.1.1	Make sure you are on a suitable kernel	11
4.1.2	Inform users in advance	11
4.1.3	Prepare for recovery	11
4.1.3.1	Debug shell during boot using initrd	12
4.1.4	Prepare a safe environment for the upgrade	12
4.2	Checking system status	12
4.2.1	Review actions pending in package manager	13
4.2.2	Disabling APT pinning	13
4.2.3	Checking packages status	13
4.2.4	The proposed-updates section	14
4.2.5	Unofficial sources and backports	14
4.2.5.1	Using backports.org packages	14
4.3	Manually unmarking packages	14
4.4	Preparing sources for APT	14
4.4.1	Adding APT Internet sources	15
4.4.2	Adding APT sources for a local mirror	15
4.4.3	Adding APT source from CD-ROM or DVD	16
4.5	Upgrading packages	16
4.5.1	Recording the session	16
4.5.2	Updating the package list	17
4.5.3	Make sure you have sufficient space for the upgrade	17
4.5.4	Upgrade apt and/or aptitude first	18
4.5.5	Using aptitude's list of automatically-installed packages with apt	19
4.5.6	Minimal system upgrade	19
4.5.7	Upgrading the rest of the system	19
4.5.8	Possible issues during upgrade	19
4.6	Upgrading your kernel and related packages	20
4.6.1	Installing the kernel metapackage	20
4.6.2	Device enumeration reordering	21

4.6.3	Boot timing issues	21
4.7	Things to do before rebooting	22
4.7.1	Rerun lilo	22
4.8	System boot hangs on Waiting for root file system	22
4.8.1	How to avoid the problem before upgrading	22
4.8.2	How to recover from the problem after the upgrade	24
4.8.2.1	Solution 1	24
4.8.2.2	Solution 2	24
4.8.2.3	Solution 3	25
4.9	Preparing for the next release	26
4.10	Obsolete packages	26
4.10.1	Dummy packages	27
4.11	Plans for the next Debian release	27
4.11.1	Drop of the ARM ABI port, in favor of the ARM EABI port	27
5	Issues to be aware of for lenny	29
5.1	Potential problems	29
5.1.1	Problems with devices related to udev	29
5.1.2	Some applications may no longer work with a 2.4 kernel	29
5.1.3	Certain network sites cannot be reached by TCP	29
5.1.4	Automatic poweroff stops working	29
5.1.5	Asynchronous network initialization may cause unpredictable behavior	29
5.1.6	Trouble when using WPA secured wireless networks	30
5.1.7	Problems with non-ASCII characters in filenames	30
5.1.8	Sound stops working	30
5.2	NFS mounts now handled by nfs-common	30
5.3	Change of Romanian (ro) keyboard layout	31
5.4	Upgrading apache2	31
5.5	NIS and Network Manager	31
5.6	Security status of Mozilla products	31
5.7	KDE desktop	31
5.8	GNOME desktop changes and support	32
5.9	No default support for Unicode in emacs21*	32
5.10	slurpd/replica will no longer work	32
5.11	Desktop not using full screen	32
5.12	DHCP failover issue	32
5.13	VServer Disk Limit	32
6	More information on Debian GNU/Linux	33
6.1	Further reading	33
6.2	Getting help	33
6.2.1	Mailing lists	33
6.2.2	Internet Relay Chat	33
6.3	Reporting bugs	33
6.4	Contributing to Debian	34
A	Managing your etch system	35
A.1	Upgrading your etch system	35
A.2	Checking your sources list	35
B	Contributors to the Release Notes	37
C	Lenny dedicated to Thiemo Seufer	39
D	Glossary	41
	Index	43

[The Debian Documentation Project](http://www.debian.org/doc/) (<http://www.debian.org/doc/>)

Chapter 1

Introduction

This document informs users of the Debian GNU/Linux distribution about major changes in version 5.0 (codenamed "lenny").

The release notes provide information on how to upgrade safely from release 4.0 (codenamed etch) to the current release and inform users of known potential issues they could encounter in that process.

You can get the most recent version of this document from <http://www.debian.org/releases/lenny/releasenotes>. If in doubt, check the date on the first page to make sure you are reading a current version.

CAUTION



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.

Please note that we only support and document upgrading from the previous release of Debian (in this case, the upgrade from 4.0). If you need to upgrade from older releases, we suggest you read previous editions of the release notes and upgrade to 4.0 first.

1.1 Reporting bugs on this document

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

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

1.2 Contributing upgrade reports

We welcome any information from users related to upgrades from etch to lenny. If you are willing to share information please file a bug in the [bug tracking system](http://bugs.debian.org/) (<http://bugs.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 Section 4.1.1, but you can also find backups of this information in `/var/backups`.
- Session logs created using `script`, as described in Section 4.5.1.

- Your apt logs, available at `/var/log/apt/term.log` or 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

The source of this document is in DocBook XML format. The HTML version is generated using `docbook-xsl` and `xsltproc`. The PDF version is generated using `dblatex` or `xmlroff`. Sources for the Release Notes are available in the SVN repository of the *Debian Documentation Project*. You can use the **web interface** (<http://svn.debian.org/viewsvn/ddp/manuals/trunk/release-notes/>) to access its files individually through the web and see their changes. For more information on how to access the SVN please consult the **Debian Documentation Project SVN information pages** (<http://www.debian.org/doc/cvs>).

Chapter 2

What's new in Debian GNU/Linux 5.0

The [Wiki](http://wiki.debian.org/NewInLenny) (<http://wiki.debian.org/NewInLenny>) has more information about this topic.

This release adds official support for the ARM EABI (armel).

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

- 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')
- ARM EABI ('armel')

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/) (<http://www.debian.org/ports/>).

2.1 What's new for ARM?

Support has also been added for Marvell's Orion platform. Specifically, Debian GNU/Linux 5.0 supports the following devices based on the Orion platform: QNAP Turbo Station ([TS-109](http://www.cyrius.com/debian/orion/qnap/ts-109/) (<http://www.cyrius.com/debian/orion/qnap/ts-109/>)), [TS-209](http://www.cyrius.com/debian/orion/qnap/ts-209/) (<http://www.cyrius.com/debian/orion/qnap/ts-209/>), [TS-409](http://www.cyrius.com/debian/orion/qnap/ts-409/) (<http://www.cyrius.com/debian/orion/qnap/ts-409/>)), [HP mv2120](http://www.cyrius.com/debian/orion/hp/mv2120/) (<http://www.cyrius.com/debian/orion/hp/mv2120/>), and [Buffalo Kurobox Pro](http://www.cyrius.com/debian/orion/buffalo/kuroboxpro/) (<http://www.cyrius.com/debian/orion/buffalo/kuroboxpro/>).

Support has been added for the Versatile platform which is emulated by QEMU.

An Ethernet driver for the IXP4xx platform (e.g. Linksys NSLU2) has recently been integrated into the mainline kernel, so the Debian kernel in lenny uses this driver rather than the unofficial driver which the previous release of Debian used.

The proprietary IXP4xx microcode needed to use the in-built Ethernet is now available in the package `ixp4xx-microcode` in non-free. Installer images for Debian which include this microcode will continue to be made available from slug-firmware.net (slug-firmware.net).

2.2 What's new in the distribution?

This new release of Debian again comes with a lot more software than its predecessor etch; the distribution includes over 7700 new packages, for a total of over 23200 packages. Most of the software in the distribution has been updated: over 13400 software packages (this is 72% of all packages in etch). Also, a significant number of packages (over 3100, 17% of the packages in etch) 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 updates from X.Org 7.1 to X.Org 7.3.

Debian GNU/Linux again ships with several desktop applications and environments. Among others it now includes the desktop environments GNOME 2.22¹, KDE 3.5.10, Xfce 4.4.2, and LXDE 0.3.2.1+svn20080509. Productivity applications have also been upgraded, including the office suites OpenOffice.org 2.4.1 and KOffice 1.6.3 as well as GNUMcash 2.2.6, GNUMeric 1.8.3 and Abiword 2.6.4.

Updates of other desktop applications include the upgrade to Evolution 2.22.3 and Pidgin 2.4.3 (formerly known as Gaim). The Mozilla suite has also been updated: *iceweasel* (version 3.0.6) is the unbranded Firefox web browser and *icedove* (version 2.0.0.19) is the unbranded Thunderbird mail client.

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

Package	Version in 4.0 (etch)	Version in 5.0 (lenny)
Apache	2.2.3	2.2.9
BIND DNS Server	9.3.4	9.5.0
Cherokee web server	0.5.5	0.7.2
Courier MTA	0.53.3	0.60.0
Dia	0.95.0	0.96.1
Ekiga VoIP Client	2.0.3	2.0.12
Exim default email server	4.63	4.69
GNU Compiler Collection as default compiler	4.1.1	4.3.2
GIMP	2.2.13	2.4.7
the GNU C library	2.3.6	2.7
lighttpd	1.4.13	1.4.19
maradns	1.2.12.04	1.3.07.09
MySQL	5.0.32	5.0.51a
OpenLDAP	2.3.30	2.4.11
OpenSSH	4.3	5.1p1
PHP	5.2.0	5.2.6
Postfix MTA	2.3.8	2.5.5
PostgreSQL	8.1.15	8.3.5
Python	2.4.4	2.5.2
Tomcat	5.5.20	5.5.26

The official Debian GNU/Linux distribution now ships on 4 to 5 binary DVDs or 28 to 32 binary CDs (depending on the architecture) and 4 source DVDs or 28 source CDs. Additionally, there is a *multi-arch* DVD, with a subset of the release for the amd64 and i386 architectures, along with the source code. For the first time, Debian GNU/Linux is also released as Blu-ray images, also for the amd64 and i386 architectures, along with the source code.

Debian now supports Linux Standards Base (LSB) version 3.2. Debian 4.0 did support version 3.1.

2.2.1 Package management

The preferred program for package management from the command line is **aptitude**, which can perform the same package management functions as **apt-get** and has proven to be better at dependency resolution. If you are still using **dselect**, you should switch to **aptitude** as the official front-end for package management.

¹ With some modules from GNOME 2.20.

For lenny 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.

2.2.2 The proposed-updates section

All changes to the released stable distribution (and to oldstable) go through an extended testing period before they are accepted into the archives. Each such update of the stable (or oldstable) release is called a point release. Preparation of point releases is done through the `proposed-updates` mechanism.

Packages can enter `proposed-updates` in two ways. Firstly, security-patched packages added to security.debian.org are automatically added to `proposed-updates` as well. Secondly, Debian GNU/Linux developers may upload new packages directly to `proposed-updates`. The current list of packages can be seen at <http://ftp-master.debian.org/proposed-updates.html> (<http://ftp-master.debian.org/proposed-updates.html>).

If you wish to help test updates to packages before they are formally added to a point release, you can do this by adding the `proposed-updates` section to your `sources.list`:

```
deb      http://mirrors.kernel.org/debian lenny-proposed-updates main contrib
deb-src  http://mirrors.kernel.org/debian lenny-proposed-updates main contrib
```

The next time you run **aptitude update**, the system will become aware of the packages in the `proposed-updates` section and will consider them when looking for packages to upgrade.

This is not strictly a new feature of Debian, but one that has not been given much exposure before.

2.3 System improvements

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

SELinux priority standard, but not enabled by default The packages needed for SELinux (Security-Enhanced Linux) 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) (<http://wiki.debian.org/SELinux>).

New default syslog daemon The package `rsyslog` takes over as default system and kernel logging daemon for Debian 5.0, replacing `syslogd` and `klogd`. With stock logging rules, it can be used as a drop-in replacement; if you have custom rules, you should migrate them to the new configuration file, `/etc/rsyslog.conf`.

Users upgrading from etch need to install `rsyslog` and remove `sysklogd` manually. The default syslog daemon is not replaced automatically at the upgrade to lenny.

Better support for UTF-8 A number of additional applications will be set up to use UTF-8 by default or have better UTF-8 support than before. See at <http://wiki.debian.org/UTF8BrokenApps> (<http://wiki.debian.org/UTF8BrokenApps>) about applications that still have difficulties in handling UTF-8.

Identification of the release's revision Starting from Lenny, `/etc/debian_version` will indicate the revision number of the debian release (5.0, then 5.0.1, etc.)

This also means that you should not expect this file to be constant through the release lifetime.

The [Debian Wiki](http://wiki.debian.org/Etch2LennyUpgrade) (<http://wiki.debian.org/Etch2LennyUpgrade>) has some additional information about changes between etch and lenny.

2.4 Major kernel-related changes

Debian GNU/Linux 5.0 ships with kernel version 2.6.26 for all architectures.

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 lenny. This section gives an overview of the most important changes; information on how to work around potential issues is included in later chapters.

2.4.1 Changes in kernel packaging

Binary firmware for some drivers moved to non-free Some drivers load binary firmware into the device they are supporting at run time. While this firmware was included in the stock kernel in previous releases, it has now be separately packaged in the non-free section. If you want to continue to use these devices after reboot, make sure the required firmware is present on the installed system. See section 6.4 of the [Installation Manual](http://www.debian.org/releases/stable/installmanual) (<http://www.debian.org/releases/stable/installmanual>) for details.

New OpenVZ kernel flavor Debian GNU/Linux 5.0 provides pre-built kernel images for OpenVZ, a second virtualization solution to go alongside the Linux-VServer support included in etch. Advantages of OpenVZ include support for live migration, at the expense of a slightly higher overhead.

Kernel x86 packages unified In previous releases there was a special `-k7` kernel flavor for 32-bit AMD Athlon/Duron/Sempron processors. This variant has been dropped; the single single flavor `-686` now handles all AMD/Intel/VIA 686 class processors.

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

2.5 Emdebian 1.0 (based on Debian GNU/Linux lenny 5.0)

Lenny now contains the build tools for Emdebian which allow Debian source packages to be cross-built and shrunk to suit embedded ARM systems.

The Emdebian 1.0 distribution itself contains prebuilt ARM packages sufficient to create root filesystems that can be customised for specific machines and machine variants. Kernels and kernel modules need to be provided separately. Support for armel and i386 is under development. See the [Emdebian webpage](http://www.emdebian.org/) (<http://www.emdebian.org/>) for further information.

2.6 Netbook support

Netbooks, such as the Eee PC by Asus, are now supported by Debian. For the Eee PC, have a look at the `eeepc-acpi-scripts`. Also, Debian features a new Lightweight X11 Desktop Environment, `lxde`, which is beneficial for netbooks or other computers with relatively low performance.

2.7 Java now in Debian

The OpenJDK Java Runtime Environment `openjdk-6-jre` and Development Kit `openjdk-6-jdk`, needed for executing Java GUI and Webstart programs or building such programs, are now in Debian. The packages are built using the IcedTea build support and patches from the IcedTea project.

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 lenny can be found together with the Installation Guide on the [Debian website](http://www.debian.org/releases/stable/debian-installer/) (<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) (<http://www.debian.org/releases/stable/debian-installer/index#errata>) for debian-installer for a list of known issues.

3.1 What's new in the installation system?

There has been a lot of development on the Debian Installer since its first official release with Debian GNU/Linux 3.1 (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 etch, please check the release announcements for the lenny beta and RC releases available from the Debian Installer's [news history](http://www.debian.org/devel/debian-installer/News/) (<http://www.debian.org/devel/debian-installer/News/>).

3.1.1 Major changes

Support for loading firmware during installation It is now possible to load firmware binary files from removable media when they're provided externally to Debian installation media.

Support for installation from Microsoft Windows The installation media are now provided with an application that allows preparing the system to install Debian from Microsoft Windows environments.

SATA RAID support

Early upgrade of packages with security fixes When used with functional network access, the installer will upgrade all packages that have been updated since the initial release of lenny. This upgrade happens during the installation step, before the installed system is booted.

As a consequence, the installed system is less likely to be vulnerable to security issues that were discovered and fixed between the release time of lenny and the installation time.

Support for *volatile* The installer can now optionally set up the installed system to use updated packages from volatile.debian.org. This archive hosts packages providing data that needs to be regularly updated over time, such as timezones definitions, anti-virus signature files, etc.

New ports The armel architecture is now supported. Images for i386 Xen guests are also provided.

Support for hardware speech synthesis devices Several devices designed to provide hardware speech synthesis are now supported by the installer, therefore improving its accessibility for visually-impaired users.

Support for `relatime` mount options The installer can now set up partitions with the `relatime` mount option, so that access time on files and directories is updated only if the previous access time was earlier than the current modify or change time.

NTP clock synchronization at installation time The computer clock is now synchronized with NTP servers over the network during installation so that the installed system immediately has an accurate clock.

New languages Thanks to the huge efforts of translators, Debian can now be installed in 63 languages (50 using the text-based installation user interface and 13 supported only with the graphical user interface). This is five languages more than in `etch`. Languages added in this release include Amharic, Marathi, Irish, Northern Sami, and Serbian. Due to lack of translation updates, one language has been dropped in this release: Estonian. Another language that was disabled in `etch` has been reactivated: Welsh.

The languages that can only be selected using the graphical installer as their character sets cannot be presented in a non-graphical environment are: Amharic, Bengali, Dzongkha, Gujarati, Hindi, Georgian, Khmer, Malayalam, Marathi, Nepali, Punjabi, Tamil, and Thai.

Simplified country choice The country choice list is now grouped by continents, allowing an easier selection of country, when users don't want to pick the ones associated with the chosen language.

3.1.2 Automated installation

Some 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 `etch` installer, you cannot expect these to work with the new installer without modification.

The **Installation Guide** (<http://www.debian.org/releases/stable/installmanual>) has an updated separate appendix with extensive documentation on using preconfiguration.

Chapter 4

Upgrades from previous releases

4.1 Preparing for the upgrade

We suggest that before upgrading you also read the information in Chapter 5. 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 log in 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.

4.1.1.1 Make sure you are on a suitable kernel

lenny's version of `glibc` will not work with kernels older than `2.6.8` on any architecture and some architectures have higher requirements. We strongly recommend that you upgrade to and test an `etch 2.6.18` or `2.6.24` kernel or a custom kernel of at least version `2.6.18` before beginning the upgrade process.

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 the `/home` partition before upgrading.

You will probably have to do a kernel upgrade when upgrading to lenny, 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 `etch` and `lenny` 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 Section 4.6.2) 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 lenny 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) (<http://www.debian.org/releases/stable/installmanual>) and the [Debian Installer FAQ](http://wiki.debian.org/DebianInstaller/FAQ) (<http://wiki.debian.org/DebianInstaller/FAQ>).

4.1.3.1 Debug shell during boot using `initrd`

The `initramfs-tools` includes a debug shell¹ 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.

4.2 Checking system status

The upgrade process described in this chapter has been designed for upgrades from 'pure' etch systems without third-party packages. For the greatest reliability of the upgrade process, you may wish to remove third-party packages from your system before you begin upgrading.

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

This procedure also assumes your system has been updated to the latest point release of etch. If you have not done this or are unsure, follow the instructions in Section [A.1](#).

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 *etch* and not to *stable* or *lenny*; see Section [A.2](#).

To perform this review, launch **aptitude** in ‘visual mode’ 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 etch as explained in Section [A.2](#).

4.2.4 The proposed-updates section

If you have listed the `proposed-updates` section in your `/etc/apt/sources.list` file, you should remove it from that file before attempting to upgrade your system. This is a precaution to reduce the likelihood of conflicts.

4.2.5 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 lenny 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 etch system. Such packages are most likely to cause problems during an upgrade as they may result in file conflicts². Section 4.5.8 has some information on how to deal with file conflicts if they should occur.

4.2.5.1 Using backports.org packages

`backports.org` is a semi-official repository provided by Debian GNU/Linux developers, which provides newer packages for the stable release, based on a rebuild from the packages from the ‘testing’ archive.

The `backports.org` repository mainly contains packages from ‘testing’, with reduced version numbers so that the upgrade path from etch backports to lenny still works. However, there are a few backports which are made from unstable: security updates, plus the following exceptions: Firefox, the Linux kernel, OpenOffice.org, and X.Org.

If you do not use one of these exceptions, you can safely upgrade to lenny. If you use one of these exceptions, set the `Pin-Priority` (see `apt_preferences(5)`) temporarily to 1001 for all packages from lenny, and you should be able to do a safe dist-upgrade too.

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 'linux-image-2.6.*' | cut -f1)
```

NOTE



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

```
# aptitude search '~i~M'
```

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`.

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.

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 line in the file (thus where you have multiple mirror locations, you'd typically first name a local hard disk, then CD-ROMs, and then HTTP/FTP mirrors).

TIP



You might need to add an GPG checking exception for DVDs and CD-ROMs. Add the following line to `/etc/apt/apt.conf`, if it's not already in `/etc/apt/apt.conf.d/00trustcdrom`:

```
APT::Authentication::TrustCDROM "true";
```

This does not work with DVD/CD-ROM image files, however.

A release can often be referred to both by its codename (e.g. `etch`, `lenny`) 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 'list of Debian mirrors' section). HTTP mirrors are generally speedier than FTP mirrors.

For example, suppose your closest Debian mirror is `http://mirrors.kernel.org`. 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/lenny/main/binary-armel/...
http://mirrors.kernel.org/debian/dists/lenny/contrib/binary-armel/...
```

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

```
deb http://mirrors.kernel.org/debian lenny 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 package 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 package mirror may be under `/var/ftp/debian/`, and have main directories like this:

```
/var/ftp/debian/dists/lenny/main/binary-armel/...
/var/ftp/debian/dists/lenny/contrib/binary-armel/...
```

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

```
deb file:/var/ftp/debian lenny 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 `'lenny'` or to `'stable'`. There should not be any sources entries pointing to `etch`.

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-lenny.time -a ~/upgrade-lenny.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-lenny.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-lenny.time ~/upgrade-lenny.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 Section 4.5.7. 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.
```

NOTE



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 Section 4.5.6 and upgraded your kernel before running this command to estimate the disk space.

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/archives`). Cleaning up the package cache by running **apt-get clean** or **aptitude clean** will remove all previously downloaded package files.
- Remove forgotten packages. If you have `popularity-contest` installed, you can use **popcon-largest-unused** to list the packages you do not use that occupy the most space. You can also use **deborphan** or **debfooster** to find obsolete packages (see Section 4.10). Alternatively you can start **aptitude** in 'visual mode' and find obsolete packages under 'Obsolete and Locally Created Packages'.
- Remove packages that take up too much space and are not currently needed (you can always reinstall them after the upgrade). You can list the packages that take up the most disk space with **dpigs** (available in the `debian-goodies` package) or with **wajig** (running `wajig size`).

You can list packages that take up most of the disk space with `aptitude`. Start **aptitude** into ‘visual mode’, select Views → New Flat Package List (this menu entry is available only after `etch` version), press **I** and enter `~i`, press **S** and enter `~installsize`, then it will give you nice list to work with. Doing this after upgrading `aptitude` should give you access to this new feature.

- Remove translations and localization files from the system if they are not needed. You can install the `localepurge` package and configure it so that only a few selected locales are kept in the system. This will reduce the disk space consumed at `/usr/share/locale`.
- Temporarily move to another system, or permanently remove, system logs residing under `/var/log/`.
- Use a temporary `/var/cache/apt/archives`: You can use a temporary cache directory from another filesystem (USB storage device, temporary hard disk, filesystem already in use, ...)

NOTE



Do not use an NFS mount as the network connection could be interrupted during the upgrade.

For example, if you have a USB drive mounted on `/media/usbkey`:

1. remove the packages that have been previously downloaded for installation:

```
# apt-get clean
```

2. copy the directory `/var/cache/apt/archives` to the USB drive:

```
# cp -ax /var/cache/apt/archives /media/usbkey/
```

3. mount the temporary cache directory on the current one:

```
# mount --bind /media/usbkey/archives /var/cache/apt/archives
```

4. after the upgrade, restore the original `/var/cache/apt/archives` directory:

```
# umount /media/usbkey/archives
```

5. remove the remaining `/media/usbkey/archives`.

You can create the temporary cache directory on whatever filesystem is mounted on your system.

Note that in order to safely remove packages, it is advisable to switch your `sources.list` back to `etch` as described in Section [A.2](#).

4.5.4 Upgrade apt and/or aptitude first

Several bug reports have shown that the versions of the `aptitude` and `apt` packages in `etch` are often unable to handle the upgrade to `lenny`. In `lenny`, `apt` is better at dealing with complex chains of packages requiring immediate configuration and `aptitude` is smarter at searching for solutions to satisfy the dependencies. These two features are heavily involved during the `dist-upgrade` to `lenny`, so it is necessary to upgrade these two packages before upgrading anything else. For `apt`, run:

```
# apt-get install apt
```

and for `aptitude` (if you have it installed) run:

```
# aptitude install aptitude
```

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 might be disconnected.

4.5.5 Using aptitude's list of automatically-installed packages with apt

`aptitude` maintains a list of packages that were installed automatically (for instance, as dependencies of another package). In `lenny`, `apt` now has this feature as well.

The first time the `lenny` version of `aptitude` is run, it will read in its list of automatically installed packages and convert it for use with the `lenny` version of `apt`. If you have `aptitude` installed, you should at least issue one `aptitude` command to do the conversion. One way to do this is by searching for a non-existent package:

```
# aptitude search "?false"
```

4.5.6 Minimal system upgrade

Because of certain necessary package conflicts between `etch` and `lenny`, 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 safe-upgrade
```

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

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 more information about packages obsoleted in `lenny`, see Section 4.10.

4.5.7 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 (or DVDs), 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`.

4.5.8 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 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 this by specifying the option `-o APT::Force-LoopBreak=1` on the **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' etch 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.

4.6.1 Installing the kernel metapackage

When you dist-upgrade from etch to lenny, 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 `'2.6.18-6-686'`, it is recommended that you install `linux-image-2.6-686`. (Note that the `k7` flavor no longer exists; if you are currently using the `k7` kernel flavor, you should install the `686` flavor instead.) 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`.

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. Note that this should only be done after the minimal upgrade process described in Section 4.5.6.

4.6.2 Device enumeration reordering

`lenny` 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 `lenny` 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/70-persistent-net.rules`³. 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 `etch` and `lenny`. 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 `lenny` 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.

4.6.3 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.

³ The rules there are automatically generated by the script `/etc/udev/rules.d/75-persistent-net-generator.rules` to have persistent names for network interfaces. Delete this symlink to disable persistent device naming for NICs by `udev`.

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 if 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 Rerun lilo

If you are using `lilo` as your bootloader (it is the default bootloader for some installations of `etch`) it is strongly recommended that you rerun `lilo` after the upgrade:

```
# /sbin/lilo
```

Notice this is needed even if you did not upgrade your system’s kernel, as `lilo`’s second stage will change due to the package upgrade.

Also, review the contents of your `/etc/kernel-img.conf` and make sure that you have `do_bootloader = Yes` in it. That way the bootloader will always be rerun after a kernel upgrade.

If you encounter any issues when running `lilo`, review the symbolic links in `/` to `vmlinuz` and `initrd` and the contents of your `/etc/lilo.conf` for discrepancies.

If you forgot to rerun `lilo` before the reboot or the system is accidentally rebooted before you could do this manually, your system might fail to boot. Instead of the `lilo` prompt, you will only see `LI` when booting the system⁴. See Section 4.1.3 for information on how to recover from this.

4.8 System boot hangs on Waiting for root file system

Procedure to recover from `/dev/hda` that became `/dev/sda` Some users have reported that an upgrade could cause the kernel not to find the system root partition after a system reboot.

In such case, the system boot will hang on the following message:

```
Waiting for root file system ...
```

and after a few seconds a bare `busybox` prompt will appear.

This problem can occur when the upgrade of the kernel introduces the use of the new generation of IDE drivers. The IDE disk naming convention for the old drivers was `hda`, `hdb`, `hdc`, `hdd`. The new drivers will name the same disks respectively `sda`, `sdb`, `sdc`, `sdd`. The problem appears when the upgrade does not generate a new `/boot/grub/menu.lst` file to take the new naming convention into account. During the boot, Grub will pass a system root partition to the kernel that the kernel doesn’t find.

If you have encountered this problem after upgrading, jump to Section 4.8.2. To avoid the problem before upgrading, read ahead.

4.8.1 How to avoid the problem before upgrading

One can avoid this problem entirely by using an identifier for the root filesystem that does not change from one boot to the next. There are two possible methods for doing this - labeling the filesystem, or using the filesystem’s universally unique identifier (UUID). These methods are supported in Debian since the ‘etch’ release.

The two approaches have advantages and disadvantages. The labeling approach is more readable, but there may be problems if another filesystem on your machine has the same label. The UUID approach is uglier, but having two clashing UUIDs is highly unlikely.

⁴ For more information on `lilo`’s boot error codes please see [The Linux Bootdisk HOWTO](http://tldp.org/HOWTO/Bootdisk-HOWTO/a1483.html) (<http://tldp.org/HOWTO/Bootdisk-HOWTO/a1483.html>).

For the examples below we assume the root filesystem is on `/dev/hda6`. We also assume your system has a working `udev` installation and `ext2` or `ext3` filesystems.

To implement the labeling approach:

1. Label the filesystem (the name must be < 16 characters) by running the command: **`e2label /dev/hda6 rootfilesys`**
2. Edit `/boot/grub/menu.lst` and change the line:

```
# kopt=root=/dev/hda6 ro
```

to

```
# kopt=root=LABEL=rootfilesys ro
```

NOTE



Do not remove the # at the start of the line, it needs to be there.

3. Update the kernel lines in `menu.lst` by running the command **`update-grub`**.
4. Edit `/etc/fstab` and change the line that mounts the `/` partition, e.g.:

```
/dev/hda6 / ext3 defaults,errors=remount-ro 0 1
```

to

```
LABEL=rootfilesys / ext3 defaults,errors=remount-ro 0 1
```

The change that matters here is the first column, you don't need to modify the other columns of this line.

To implement the UUID approach:

1. Find out the universally unique identifier of your filesystem by issuing: **`ls -l /dev/disk/by-uuid | grep hda6`**

You should get a line similar to this one:

```
lrwxrwxrwx 1 root root 24 2008-09-25 08:16 d0dfcc8a-417a-41e3-ad2e-9736317f2d8a ↔  
f2d8a -> ../../hda6
```

The **UUID** is the name of the symbolic link pointing to `/dev/hda6` i.e.: `d0dfcc8a-417a-41e3-ad2e-9736317f2d8a`.

NOTE



Your filesystem UUID will be a different string.

2. Edit `/boot/grub/menu.lst` and change the line:

```
# kopt=root=/dev/hda6 ro
```

to

```
# kopt=root=UUID=d0dfcc8a-417a-41e3-ad2e-9736317f2d8 ro
```

NOTE



Do not remove the # at the start of the line, it needs to be there.

3. Update the `kernel` lines in `menu.lst` by running the command **update-grub**.
4. Edit `/etc/fstab` and change the line that mounts the `/` partition, e.g.:

```
/dev/hda6      /          ext3  defaults,errors=remount-ro 0 1
```

to

```
UUID=d0dfcc8a-417a-41e3-ad2e-9736317f2d8 /  ext3  defaults,errors=remount- ↔  
ro 0 1
```

The change that matters here is the first column, you don't need to modify the other columns of this line.

4.8.2 How to recover from the problem after the upgrade

4.8.2.1 Solution 1

This is applicable when Grub shows you the menu interface for selecting the entry you want to boot from. If such a menu does not appear, try pressing the **Esc** key before the kernel boots in order to make it appear. If you can't get into this menu, try Section 4.8.2.2 or Section 4.8.2.3.

1. In the Grub menu, highlight the entry you want to boot from. Press the **e** key to edit the options related to this entry. You will see something like:

```
root (hd0,0)  
kernel /vmlinuz-2.6.26-1-686 root=/dev/hda6 ro  
initrd /initrd.img-2.6.26-1-686
```

2. Highlight the line

```
kernel /vmlinuz-2.6.26-1-686 root=/dev/hda6 ro
```

press the **e** key and replace `hdX` with `sdX` (`X` being the letter `a`, `b`, `c` or `d` depending of your system). In my example the line becomes:

```
kernel /vmlinuz-2.6.26-1-686 root=/dev/sda6 ro
```

Then press **Enter** to save the modification. If other lines show `hdX`, change these line too. Don't modify the entry similar to `root (hd0,0)`. Once all modifications are done, press the **b** key. And your system should now boot as usual.

3. Now that your system has booted, you need to fix this issue permanently. Jump to Section 4.8.1 and apply one of the two proposed procedures.

4.8.2.2 Solution 2

Boot from Debian GNU/Linux installation media (CD/DVD) and when prompted, pick `rescue` to launch rescue mode. Select your language, location, and keyboard mapping; then let it configure the network (no matter whether it succeeds or not). After a while, you should be asked to select the partition you want to use as root file system. The proposed choices will look something like:

```
/dev/ide/host0/bus0/target0/lun0/part1
/dev/ide/host0/bus0/target0/lun0/part2
/dev/ide/host0/bus0/target0/lun0/part5
/dev/ide/host0/bus0/target0/lun0/part6
```

If you know which partition is your root file system, choose the appropriate one. If you don't, just try with the first. If it complains about an invalid root file system partition, try the next one, and so on. Trying one after the other shouldn't harm your partitions and if you have only one operating system installed on your disks, you should easily find the right root file system partition. If you have many operating systems installed on your disks, it would be better to know exactly which is the right partition.

Once you have chosen a partition, you will be offered a range of options. Pick the option of executing a shell in the selected partition. If it complains that it cannot do that then try with another partition.

Now you should have shell access as user `root` on your root file system mounted on `/target`. You need access to the contents of the `/boot`, `/sbin` and `/usr` directories on your hard disk, which should now be available under `/target/boot`, `/target/sbin` and `/target/usr`. If these directories need to be mounted from other partitions, do so (see `/etc/fstab` if you have no idea of which partition to mount).

Jump to Section 4.8.1 and apply one of the two proposed procedures to fix the problem permanently. Then type `exit` to leave the rescue shell and select `reboot` for rebooting the system as usual (don't forget to remove the bootable media).

4.8.2.3 Solution 3

1. Boot from your favorite LiveCD distribution, such as Debian Live, Knoppix, or Ubuntu Live.
2. Mount the partition where your `/boot` directory is. If you don't know which one it is, use the output of the command `dmeg` to find whether your disk is known as `hda`, `hdb`, `hdc`, `hdd` or `sda`, `sdb`, `sdc`, `sdd`. Once you know which disk to work on, for example `sdb`, issue the following command to see the partition table of the disk and to find the right partition: **`fdisk -l /dev/sdb`**
3. Assuming that you have mounted the right partition under `/mnt` and that this partition contains the `/boot` directory and its content, edit the `/mnt/boot/grub/menu.lst` file.

Find the section similar to:

```
## ## End Default Options ##

title          Debian GNU/Linux, kernel 2.6.26-1-686
root           (hd0,0)
kernel         /vmlinuz-2.6.26-1-686 root=/dev/hda6 ro
initrd        /initrd.img-2.6.26-1-686

title          Debian GNU/Linux, kernel 2.6.26-1-686 (single-user mode)
root           (hd0,0)
kernel         /vmlinuz-2.6.26-1-686 root=/dev/hda6 ro single
initrd        /initrd.img-2.6.26-1-686

### END DEBIAN AUTOMAGIC KERNELS LIST
```

and replace every `hda`, `hdb`, `hdc`, `hdd` with `sda`, `sdb`, `sdc`, `sdd`, as appropriate. Don't modify the line similar to:

```
root           (hd0,0)
```

4. Reboot the system, remove the LiveCD and your system should boot correctly.
5. When it has booted, apply one of the two proposed procedures under Section 4.8.1 to fix the problem permanently.

4.9 Preparing for the next release

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

- 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 obsolete and unused packages as described in Section 4.10. You should review which configuration files they use and consider purging the packages to remove their configuration files.

4.10 Obsolete packages

Introducing several thousand new packages, lenny also retires and omits more than two thousand old packages that were in etch. 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 lenny's release⁵, 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 lenny 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 etch 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**, **debfoister** or **craft**. **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 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>).

The list of obsolete packages includes:

- apache (1.x), successor is apache2
- bind (8), successor is bind9
- php4, successor is php5
- postgresql-7.4, successor is postgresql-8.1
- exim (3), successor is exim4

⁵ 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.

4.10.1 Dummy packages

Some packages from etch have been split into several packages in lenny, often to improve system maintainability. To ease the upgrade path in such cases, lenny often provides ‘dummy’ packages: empty packages that have the same name as the old package in etch 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.

4.11 Plans for the next Debian release

4.11.1 Drop of the ARM ABI port, in favor of the ARM EABI port

Debian lenny has two different and incompatible ARM ports: the old ABI port (`arm`) and the new EABI port (`armel`). Debian lenny is the last release with support for the ARM ABI port and future releases will only support the ARM EABI or `armel` port. It’s therefore recommended to use `armel` for new installations of lenny.

With the exception of Netwinder, installer images for supported ARM machines are available for both `arm` and `armel` in lenny. Netwinder support is only available for `arm` and it will be dropped after lenny along with the `arm` port.

Please visit [this page](http://wiki.debian.org/ArmEabiPort) (<http://wiki.debian.org/ArmEabiPort>) to learn more about the ARM EABI (`armel`) port.

Chapter 5

Issues to be aware of for lenny

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 Section 6.1.

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 lenny 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) (<http://bugs.debian.org/381262>), [#395066](http://bugs.debian.org/395066) (<http://bugs.debian.org/395066>), [#401435](http://bugs.debian.org/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) (<http://www.debian.org/devel/debian-installer/errata>).

5.1.4 Automatic poweroff stops working

On some older systems, `shutdown -h` may not power off the system anymore (but just stop it). This happens because APM needs to be used there. Adding `acpi=off apm=power_off` to the kernel's command line, e.g. in `grub` or `lilo` configuration files should fix this issue. Please see bug [#390547](http://bugs.debian.org/390547) (<http://bugs.debian.org/390547>) for additional information.

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 `etch`, the `wpa_supplicant` package was set up as a system service, configured via `/etc/default/wpa_supplicant` and a user-provided `/etc/wpa_supplicant.conf`.

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

For information on configuring `wpa_supplicant` please refer to `/usr/share/doc/wpa_supplicant/README.modes.gz`, which gives examples for `/etc/network/interfaces` files. Updated information about the usage of the `wpa_supplicant` package in Debian can be found in the [Debian Wiki](http://wiki.debian.org/WPA) (<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, 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,
- make sure the sound channel levels are up and unmuted (using `alsamixer`),
- make sure `arts` and `esound` are not running,
- make sure no OSS modules are loaded,
- make sure the speakers are actually switched on, and
- check whether the command

```
cat /dev/urandom > /dev/audio
```

or the command

```
speaker-test
```

works for `root`.

5.2 NFS mounts now handled by `nfs-common`

Since `util-linux 2.13` NFS mounts are no longer handled by `util-linux` itself, but by `nfs-common`. Since not all systems mount NFS shares and to avoid a standard `portmapper` installation `util-linux` only suggests `nfs-common`. If you need to mount NFS shares, make sure `nfs-common` is installed on your system. The preinstallation script of the `mount` package checks whether NFS mounts exist and aborts if `/usr/sbin/mount.nfs` from `nfs-common` is not present or if `nfs-common` is out-of-date. Either upgrade `nfs-common` or unmount any NFS mounts prior to upgrading `mount`.

5.3 Change of Romanian (ro) keyboard layout

Because of the upgrade to `xkb-data` version 1.3 in lenny the default variant for Romanian (ro) layout is now producing the correct `șț` characters (comma below) instead of `șţ` (cedilla below). Also some of the variants have been renamed. The old variant names still work, but users are encouraged to update their `/etc/X11/xorg.conf`. More info as well as possible side effects due to this change are available in the [wiki \(Romanian language only\)](http://wiki.debian.org/L10N/Romanian/Lenny/Notes) (<http://wiki.debian.org/L10N/Romanian/Lenny/Notes>).

5.4 Upgrading apache2

The apache2 default configuration has changed in some ways that may require manual changes to your configuration. The most important changes are:

`NameVirtualHost *` has been changed to `NameVirtualHost *:80`. If you have added more name based virtual hosts, you need to change `<VirtualHost *>` to `<VirtualHost *:80>` for each of them.

The Apache User and Group and the `PidFile` path are now configured in `/etc/apache2/envvars`. If you have changed these settings from their default values, you need to change that file. This also means that starting apache2 with `apache2 -k start` is no longer possible, you have to use `/etc/init.d/apache2` or `apache2ctl`.

The `suexec` helper program needed for `mod_suexec` is now shipped in a separate package, `apache2-suexec`, which is not installed by default.

More module specific configuration has been moved from `/etc/apache2/apache2.conf` to `/etc/apache2/mods-available/*.conf`.

For more detailed information, see `/usr/share/doc/apache2.2-common/NEWS.Debian.gz` and `/usr/share/doc/apache2.2-common/README.Debian.gz`.

5.5 NIS and Network Manager

The version of `ybind` included with `nis` for lenny contains support for Network Manager. This support causes `ybind` 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.

5.6 Security status of Mozilla products

The Mozilla programs `firefox`, `thunderbird`, and `sunbird` (rebranded in Debian to `iceweasel`, `icedove`, and `iceowl`, 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 lenny 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.

`iceape`, the unbranded version of the `seamonkey` internet suite has been removed from lenny (with the exception of a few internal library packages).

5.7 KDE desktop

There are no huge changes in the KDE Desktop Environment from the version shipped in etch. Lenny ships an updated translation and service release of KDE 3.5 that is a mixture of 3.5.9 and 3.5.10. Some modules are labeled as version 3.5.9, but have been updated and include most of the same changes

found in 3.5.10. Overall, lenny ships 3.5.10 without the kicker improvements shipped in kdebase and some bug fixes in kdepim.

Lenny will be the last stable release including a KDE 3 series environment.

5.8 GNOME desktop changes and support

There have been many changes in the GNOME desktop environment from the version shipped in etch to the version in lenny, you can find more information in the [GNOME 2.22 Release Notes](http://library.gnome.org/misc/release-notes/2.22/) (<http://library.gnome.org/misc/release-notes/2.22/>).

5.9 No 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) (<http://bugs.debian.org/419490>). Consider switching to emacs22, emacs22-gtk, or emacs22-nox.

5.10 slurpd/replica will no longer work

OpenLDAP has dropped support for LDAP replication via the slurpd service in release 2.4.7. Existing configurations need to be reconfigured for the LDAP Sync Replication engine (syncrepl). More verbose documentation can be found at <http://www.openldap.org/doc/admin24/replication.html> (<http://www.openldap.org/doc/admin24/replication.html>).

5.11 Desktop not using full screen

The driver for Intel Mobile GM965 may wrongly detect a VGA output and set the size of the screen to a lower value to accomodate it. The symptom of this bug is that the desktop manager will only use a fraction of the screen. Correct behaviour can be forced by adding the following lines to the `/etc/X11/xorg.conf` configuration file.

```
Section "Monitor"
    Identifier "VGA"
    Option "Ignore" "true"
EndSection
```

Please refer to the bug [#496169](http://bugs.debian.org/496169) (<http://bugs.debian.org/496169>) for more informations.

5.12 DHCP failover issue

When running a failover pair of DHCP servers, the peer names need to be consistent, otherwise DHCP will crash.

Please see bug [#513506](http://bugs.debian.org/513506) (<http://bugs.debian.org/513506>) and <https://lists.isc.org/pipermail/dhcp-users/2007-September/004538.html> for more information.

5.13 VServer Disk Limit

To use the disk limit feature of vservers in lenny, you should use the `mount` option tag (instead of `tagid` in etch).

You should manually update `/etc/fstab` and/or any script which uses `tagid`. Otherwise, the partition will not be mounted and thus the vservers will not start.

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) (<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) (<http://wiki.debian.org/DebianIRC>).

For more information on OFTC please visit the [website](http://www.oftc.net/) (<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 <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 documentation (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 etch system

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

A.1 Upgrading your etch system

Basically this is no different than any other upgrade of etch you've been doing. The only difference is that you first need to make sure your package list still contains references to etch as explained in Section [A.2](#).

If you upgrade your system using a Debian mirror, it will automatically be upgraded to the latest etch 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' lenny. 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 lenny, there probably is not much point in installing packages from etch 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 `etch`.

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

Appendix B

Contributors to the Release Notes

Many people helped with the release notes, including, but not limited to

Adam Di Carlo, Andreas Barth, Andrei Popescu, Anne Bezemer, Bob Hilliard, Charles Plessy, Christian Perrier, Daniel Baumann, Eddy Petrișor, Emmanuel Kasper, Esko Arajärvi, Frans Pop, Giovanni Ragnani, Gordon Farquharson, Javier Fernández-Sanguino Peña, Jens Seidel, Jonas Meurer, Josip Rodin, Justin B Rye, LaMont Jones, Luk Claes, Martin Michlmayr, Michael Biebl, Moritz Mühlenhoff, Noah Meyerhans, Noritada Kobayashi, Osamu Aoki, Peter Green, Rob Bradford, Samuel Thibault, Simon Bilenlein, Simon Paillard, Stefan Fritsch, Steve Langasek, Tobias Scherer, Vincent McIntyre, and W. Martin Borgert.

This document has been translated into many languages. Many thanks to the translators!

Appendix C

Lenny dedicated to Thiemo Seufer

The Debian Project has lost an active member of its community. Thiemo Seufer died on December 26th, 2008 in a tragic car accident.

Thiemo was involved in Debian in many ways. He maintained several packages and was the main supporter of the Debian ports to the MIPS architecture. He was also a member of our kernel team, as well as a member of the Debian Installer team. His contributions reached far beyond the Debian project: he also worked on the MIPS port of the Linux kernel, the MIPS emulation of qemu, and far too many smaller projects to be named here.

Thiemo's work, dedication, broad technical knowledge and ability to share this with others will be missed. His contributions will not be forgotten. The high standards of Thiemo's work make it hard to pick up.

To honour his contributions to Debian, the project dedicates the release of Debian GNU/Linux 5.0 'Lenny' to Thiemo.

Appendix D

Glossary

ACPI

Advanced Configuration and Power Interface

ALSA

Advanced Linux Sound Architecture

APM

Advanced Power Management

CD

Compact Disc

CD-ROM

Compact Disc Read Only Memory

DHCP

Dynamic Host Configuration Protocol

DNS

Domain Name System

DVD

Digital Versatile Disc

GIMP

GNU Image Manipulation Program

GNU

GNU's Not Unix

GPG

GNU Privacy Guard

IDE

Integrated Drive Electronics

LDAP

Lightweight Directory Access Protocol

LILO

Linux LOader

LSB

Linux Standards Base

LVM

Logical Volume Manager

MTA

Mail Transport Agent

NFS

Network File System

NIC

Network Interface Card

NIS

Network Information Service

OSS

Open Sound System

RAID

Redundant Array of Independent Disks

RPC

Remote Procedure Call

SATA

Serial Advanced Technology Attachment

USB

Universal Serial Bus

UUID

Universally Unique Identifier

VGA

Video Graphics Array

WPA

Wi-Fi Protected Access

Index

A

Abiword, 6
Apache, 6

B

BIND, 6
Blu-ray, 6

C

CD, 6
Cherokee, 6
Courier, 6

D

Dia, 6
DocBook XML, 4
DVD, 6

E

Ekiga, 6
Emdebian, 8
Evolution, 6
Exim, 6

F

Firefox, 6

G

Gaim, 6
GCC, 6
GIMP, 6
GNOME, 6
GNUCash, 6
GNUmeric, 6

I

IcedTea, 8

J

Java, 8

K

KDE, 6
KOffice, 6

L

LILLO, 22
Linux Standards Base, 6
LXDE, 6

M

Microsoft Windows, 9
Mozilla, 6, 31
MySQL, 6

N

Netwinder, 27
Network Manager, 31
NIS, 31

O

OpenJDK, 8
OpenOffice.org, 6
OpenSSH, 6
OpenVZ, 8

P

packages
 apache, 26
 apache2, 26
 apache2-suexec, 31
 apt, 4, 14, 15, 17–19
 aptitude, 6, 18, 19
 base-config, 19
 bind, 26
 bind9, 26
 dbrlatex, 4
 debian-goodies, 17
 doc-debian, 34
 docbook-xsl, 4
 eeepc-acpi-scripts, 8
 emacs22, 32
 emacs22-gtk, 32
 emacs22-nox, 32
 exim, 26
 exim4, 26
 firefox, 31
 glibc, 11
 grub, 29
 hotplug, 19
 iceape, 31
 icedove, 6, 31
 iceowl, 31
 iceweasel, 6, 31
 initramfs-tools, 12, 20, 21
 ixp4xx-microcode, 5
 kernel-package, 21
 libc6, 18
 libselinux1, 18
 lilo, 22, 29
 linux-image-*, 20
 linux-image-2.6-686, 21
 localepurge, 18
 locales, 18
 lxde, 8
 mount, 30
 netkit-inetd, 19
 network-manager, 31
 nfs-common, 30
 nis, 31
 php4, 26
 php5, 26
 popularity-contest, 17
 postgresql-7.4, 26
 postgresql-8.1, 26
 python2.3, 19
 release-notes, 3

rsyslog, 7
seamonkey, 31
squid, 29
sunbird, 31
sysklogd, 7
thunderbird, 31
udev, 20, 21, 29
upgrade-reports, 3
util-linux, 30
wireless-tools, 30
wpa_supplicant, 30
xfree86-common, 19
xkb-data, 31
xlibs, 19
xmlroff, 4
xserver-common, 19
xsltproc, 4

PHP, 6
Pidgin, 6
Postfix, 6
PostgreSQL, 6

S
SELinux, 7, 18

T
Thunderbird, 6
Tomcat, 6

U
Unicode, 32

V
virtualization, 8
visually-impaired users, 9
VServer, 8

W
WPA, 30

X
Xfce, 6