Release Notes for Debian 7.0 (wheezy), Mipsel

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

May 22, 2013

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/licenses/gpl-2.0.html and /usr/share/common-licenses/GPL-2 on Debian.

Contents

1 Introduction **1**
1.1 Reporting bugs on this document . . . 1
1.2 Contributing upgrade reports . . . 1
1.3 Sources for this document . . . 2

2 What's new in Debian 7.0 **3**
2.1 Supported architectures . . . 3
2.2 What's new in the distribution? . . . 4
2.2.1 CDs, DVDs and BDs . . . 4
2.2.2 Multiarch . . . 5
2.2.3 Dependency booting . . . 5
2.2.4 systemd . . . 5
2.2.5 Multimedia . . . 5
2.2.6 Hardened security . . . 5
2.2.7 AppArmor . . . 6
2.2.8 The stable-backports section . . . 6
2.2.9 The stable-updates section . . . 6
2.2.10 GNOME 3 . . . 6
2.2.10.1 New and removed applications . . . 7
2.2.10.2 Settings . . . 7
2.2.10.3 Display manager . . . 7
2.2.10.4 Network management . . . 7
2.2.11 Cloud . . . 7
2.2.12 Temporary filesystems . . . 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 Debian 6.0 (squeeze) **11**
4.1 Preparing for the upgrade . . . 11
4.1.1 Back up any data or configuration information . . . 11
4.1.2 Inform users in advance . . . 11
4.1.3 Prepare for downtime on services . . . 11
4.1.4 Prepare for recovery . . . 12
4.1.4.1 Debug shell during boot using initrd . . . 12
4.1.5 Prepare a safe environment for the upgrade . . . 12
4.2 Checking system status . . . 13
4.2.1 Review actions pending in package manager . . . 13
4.2.2 Disabling APT pinning . . . 13
4.2.3 Checking packages status . . . 14
4.2.4 The proposed-updates section . . . 14
4.2.5 Unofficial sources and backports . . . 14
4.3 Preparing sources for APT . . . 15
4.3.1 Adding APT Internet sources . . . 15
4.3.2 Adding APT sources for a local mirror . . . 15
4.3.3 Adding APT sources from optical media . . . 15
4.4 Upgrading packages . . . 16
4.4.1 Recording the session . . . 16
4.4.2 Updating the package list . . . 17
4.4.3 Make sure you have sufficient space for the upgrade . . . 17
4.4.4 Minimal system upgrade . . . 18
4.4.5 Upgrading the system . . . 19

4.5 Possible issues during upgrade . . . 19
4.5.1 Dist-upgrade fails with "Could not perform immediate configuration" . . . 19
4.5.2 Expected removals . . . 19
4.5.3 Conflicts or Pre-Depends loops . . . 20
4.5.4 File conflicts . . . 20
4.5.5 Configuration changes . . . 20
4.5.6 Change of session to console . . . 20
4.5.7 Special care for specific packages . . . 21
4.5.7.1 Sudo . . . 21
4.5.7.2 Screen . . . 21
4.5.7.3 Suhosin PHP module . . . 21
4.6 Upgrading your kernel and related packages . . . 21
4.6.1 Installing the kernel metapackage . . . 22
4.6.2 Boot timing issues (waiting for root device) . . . 22
4.7 Preparing for the next release . . . 22
4.8 Obsolete packages . . . 23
4.8.1 Dummy packages . . . 24

5 Issues to be aware of for wheezy **25**
5.1 LDAP support . . . 25
5.2 Security status of web browsers . . . 25
5.3 ConsoleKit and alternative display managers . . . 25
5.4 GNOME desktop changes and support . . . 26
5.5 KDE desktop changes . . . 26
5.6 NetworkManager . . . 26
5.7 `perl-suid` removed . . . 26
5.8 Request Tracker versions . . . 26
5.9 Bootlogd changes . . . 27
5.10 `/etc/mtab` and `_netdev` . . . 27
5.11 The pdksh to mksh transition . . . 27
5.12 Puppet 2.6 / 2.7 compatibility . . . 28
5.13 Multiarch implications for the toolchain . . . 28
5.14 Cyrus SASL SQL backends . . . 28
5.15 Firmware for network and graphics drivers . . . 29

6 More information on Debian **31**
6.1 Further reading . . . 31
6.2 Getting help . . . 31
6.2.1 Mailing lists . . . 31
6.2.2 Internet Relay Chat . . . 31
6.3 Reporting bugs . . . 31
6.4 Contributing to Debian . . . 32

7 Glossary **33**

A Managing your squeeze system before the upgrade **35**
A.1 Upgrading your squeeze system . . . 35
A.2 Checking your sources list . . . 35
A.3 Removing obsolete configuration files . . . 36
A.4 Upgrade legacy locales to UTF-8 . . . 36

B Contributors to the Release Notes **37**

Index **39**

Chapter 1

Introduction

This document informs users of the Debian distribution about major changes in version 7.0 (codenamed wheezy).

The release notes provide information on how to upgrade safely from release 6.0 (codenamed squeeze) 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/wheezy/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 6.0). If you need to upgrade from older releases, we suggest you read previous editions of the release notes and upgrade to 6.0 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/`) against the `release-notes` package. You might want to review first the existing bug reports (`http://bugs.debian.org/release-notes`) in case the issue you've found has already been reported. Feel free to add additional information to existing bug reports if you can contribute content for this document.

We appreciate, and encourage, reports providing patches to the document's sources. You will find more information describing how to obtain the sources of this document in Section 1.3.

1.2 Contributing upgrade reports

We welcome any information from users related to upgrades from squeeze to wheezy. If you are willing to share information please file a bug in the bug tracking system (`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 `apt`'s package state information, available at `/var/lib/`

`apt/extended_states`. You should have made a backup before the upgrade as described at Section 4.1.1, but you can also find backups of `/var/lib/dpkg/status` in `/var/backups`.

- Session logs created using **script**, as described in Section 4.4.1.
- 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://anonscm.debian.org/viewvc/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 7.0

The Wiki (`http://wiki.debian.org/NewInWheezy`) has more information about this topic.

2.1 Supported architectures

Debian 7.0 introduces two new architectures:

- s390x, 64-bit port for IBM System z machines intended to replace s390.
- armhf, an alternative to armel for ARMv7 machines with hard-float. A lot of modern ARM boards and devices ship with a floating-point unit (FPU), but the older Debian armel port doesn't take much advantage of it. The armhf port was started to improve this situation and also take advantage of other features of newer ARM CPUs. The Debian armhf port requires at least an ARMv7 CPU with Thumb-2 and VFP3D16 coprocessor.

The following are the officially supported architectures for Debian wheezy:

- 32-bit PC ('i386')
- SPARC ('sparc')
- PowerPC ('powerpc')
- MIPS ('mips' (big-endian) and 'mipsel' (little-endian))
- Intel Itanium ('ia64')
- S/390 ('s390')
- 64-bit PC ('amd64')
- ARM EABI ('armel')
- ARMv7 (EABI hard-float ABI, 'armhf')
- IBM System z ('s390x')

In addition to the officially supported architectures, Debian wheezy contains the GNU/kFreeBSD ports ('kfreebsd-amd64' and 'kfreebsd-i386') introduced in Debian squeeze, as a technology preview. These ports are the first ones included in a Debian release which aren't based on the Linux kernel, but instead use the kernel of FreeBSD with a GNU userland. Users of these versions however should be warned that the quality of these ports is still catching up with the outstanding high quality of our Linux ports, and that some advanced desktop features are not supported yet. However, the support of common server software is strong and extends the features of Linux-based Debian versions by the unique features known from the BSD world.

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/`).

2.2 What's new in the distribution?

This new release of Debian again comes with a lot more software than its predecessor squeeze; the distribution includes over 12800 new packages, for a total of over 37493 packages. Most of the software in the distribution has been updated: over 20160 software packages (this is 70% of all packages in squeeze). Also, a significant number of packages (over 4125, 14% of the packages in squeeze) 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 updates from X.Org 7.5 to X.Org 7.7.

Debian again ships with several desktop applications and environments. Among others it now includes the desktop environments GNOME 3.4, KDE 4.8.4, Xfce 4.8, and LXDE.

Productivity applications have also been upgraded, including the office suites:

- LibreOffice 3.5 replaces OpenOffice.org, which is now only a transitional package that can be removed;
- Calligra 2.4 replaces KOffice, which is now only a transitional package that can be removed;
- GNUcash is upgraded to 2.4;
- GNUmeric is upgraded to 1.10;
- Abiword is upgraded to 2.9.

Updates of other desktop applications include the upgrade to Evolution 3.4 and Pidgin 2.10. The Mozilla suite has also been updated: `iceweasel` (version 10 ESR) is the unbranded Firefox web browser and `icedove` (version 10) is the unbranded Thunderbird mail client.

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

Package	**Version in 6.0 (squeeze)**	**Version in 7.0 (wheezy)**
Apache	2.2.16	2.2.22
BIND DNS Server	9.7	9.8
Courier MTA	0.65	0.68
Dia	0.97.1	0.97.2
Exim default email server	4.72	4.80
GNU Compiler Collection as default compiler	4.4	4.7 on PCs, 4.6 elsewhere
GIMP	2.6	2.8
the GNU C library	2.11	2.13
lighttpd	1.4.28	1.4.31
Linux kernel image	2.6 series	3.2 series
maradns	1.4.03	1.4.12
MySQL	5.1	5.5
OpenLDAP	2.4.23	2.4.31
OpenSSH	5.5p1	6.0p1
Perl	5.10	5.14
PHP	5.3	5.4
Postfix MTA	2.7	2.9
PostgreSQL	8.4	9.1
Python	2.6	2.7
Python 3	3.1	3.2
Samba	3.5	3.6

Debian supports Linux Standard Base (LSB) version 4.1, with one explicit and Debian-specific derogation from the LSB 4.1 specification: Qt3 is not included.

2.2.1 CDs, DVDs and BDs

The official Debian distribution now ships on 9 to 10 binary DVDs or 61 to 69 binary CDs (depending on the architecture) and 8 source DVDs or 46 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. Debian is also released as Blu-ray (BD) images, 2 each for the `amd64` and `i386` architectures, or 2 for the source code. For size reasons, some very large packages are omitted from the CD builds; these packages fit better in the DVD and BD builds, so are still included there.

2.2.2 Multiarch

New in Debian 7.0 is *multiarch*. Multiarch lets you install packages from multiple architectures on the same machine. This is useful in various ways, but the most common is installing both 64 and 32-bit software on the same machine and having dependencies correctly resolved automatically. The Debian wiki has an extensive manual (`http://wiki.debian.org/Multiarch/HOWTO`) on how to make use of this functionality if you need it.

2.2.3 Dependency booting

The dependency-based boot sequencing introduced with Debian 6.0 is now always enabled, including for users of `file-rc`.

For optimal sequencing, all init.d scripts should declare their dependencies in an LSB header. This is already the case for scripts shipped in Debian, but users should check their local scripts and consider adding that information.

For more information on this feature refer to the information available in `/usr/share/doc/insserv/README.Debian`.

2.2.4 systemd

Debian 7.0 introduces preliminary support for systemd, an init system with advanced monitoring, logging and service management capabilities.

While it is designed as a drop-in `sysvinit` replacement and as such makes use of existing SysV init scripts, the `systemd` package can be installed safely alongside `sysvinit` and started via the `init=/bin/systemd` kernel option. To utilize the features provided by systemd, about 50 packages already provide native support, among them core packages like udev, dbus and rsyslog.

systemd is shipped as a technology preview in Debian 7.0. For more information on this topic, see the Debian wiki (`http://wiki.debian.org/systemd`).

2.2.5 Multimedia

Debian wheezy comes with improved multimedia support: `ffmpeg` has been replaced by the libav fork (`libav-tools`), which is considered to feature a more conservative release process and thus fit better to Debian's needs. It provides all libraries and prepares an upgrade path for existing application packages. The full-featured libav libraries and frontends include e.g. **mplayer**, **mencoder**, **vlc** and **transcode**. Additional codec support is provided e.g. through **lame** for MP3 audio encoding, **xvidcore** for MPEG-4 ASP video encoding, **x264** for H.264/MPEG-4 AVC video encoding, **vo-aacenc** for AAC audio encoding and **opencore-amr** and **vo-amrwbenc** for Adaptive Multi-Rate Narrowband and Wideband encoding and decoding, respectively. For most use cases, installation of packages from third-party repositories should not be necessary anymore. The times of crippled multimedia support in Debian are finally over!

2.2.6 Hardened security

Many Debian packages have now been built with `gcc` compiler hardening flags enabled. These flags enable various protections against security issues such as stack smashing, predictable locations of values in memory, etc. An effort has been made to ensure that as many packages as possible include these flags, especially focusing on those in the 'base'-installation, network-accessible daemons and packages which have had security issues in recent years.

Note that the hardened build flags are not enabled by default in `gcc`, so are not used automatically when locally building software. The package `hardening-wrapper` can provide a `gcc` with these flags enabled.

2.2.7 AppArmor

Debian 7.0 supports the AppArmor Mandatory Access Control system. When enabled, AppArmor confines programs according to a set of rules that specify what files a given program can access. This proactive approach helps protecting the system against both known and unknown vulnerabilities.

AppArmor is disabled by default in Debian 7.0. The Debian wiki has instructions (`http://wiki.debian.org/AppArmor`) on how to use this functionality.

2.2.8 The stable-backports section

Note that this replaces the functionality previously provided by the backports.debian.org archive (`http://backports.debian.org/`).

In order to use packages from `wheezy-backports`, you can add an entry to your `sources.list`:

```
deb     http://mirrors.kernel.org/debian wheezy-backports main contrib
deb-src http://mirrors.kernel.org/debian wheezy-backports main contrib
```

The next time you run **apt-get update**, the system will become aware of the packages in the `wheezy-backports` section and they will be available for installation in the same way as the old backports.debian.org archive.

When a new package is made available via `wheezy-backports` to fix a security issue, this will be announced on the debian-backports-announce (`http://lists.debian.org/debian-backports-announce/`) mailing list.

2.2.9 The stable-updates section

Some packages from `proposed-updates` may also be made available via the `wheezy-updates` mechanism. This path will be used for updates which many users may wish to install on their systems before the next point release is made, such as updates to virus scanners and timezone data. All packages from `wheezy-updates` will be included in point releases.

In order to use packages from `wheezy-updates`, you can add an entry to your `sources.list`:

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

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

Note that if `APT::Default-Release` is set in your `/etc/apt/apt.conf` (or in any of `/etc/apt/apt.conf.d/*`), then, in order for automatic upgrades to work, it is necessary to add the following configuration block into `/etc/apt/preferences` (see apt_preferences(5) for more information):

```
Package: *
Pin: release o=Debian,n=wheezy-updates
Pin-Priority: 990
```

When a new package is made available via `wheezy-updates`, this will be announced on the debian-stable-announce (`http://lists.debian.org/debian-stable-announce/`) mailing list.

2.2.10 GNOME 3

GNOME has undergone a major interface rewrite in the upgrade to version 3.4. The traditional GNOME panel has been replaced by the "shell", an innovative interface with major usability improvements.

Among other things, it features dynamic workspaces, an on-screen keyboard (Caribou), instant messaging built into the interface, and integration with the GNOME keyring and PolicyKit.

If you want to keep an interface closer to the GNOME 2.30 version in wheezy, you can select the "GNOME Classic" session at the login prompt. It will bring you an improved version of the traditional panel. You can still edit the panel to add more applets, by using the hidden alt+right click combination.

If your hardware is not compatible with the GNOME shell's requirements, you will also be redirected to the "classic" interface.

2.2.10.1 New and removed applications

Sushi is a new previewing application. Just press the space key on a file in the file manager, and enjoy.

The Tracker indexing tool is now part of the GNOME desktop. After your first login, it will index your desktop, and is now available as the default search tool. It is also the key to the new GNOME documents tool to manage your recently used documents.

Audio and mixing applications now require the PulseAudio sound daemon, which provides per-application mixing.

The help system has been entirely redesigned, with a new documentation format.

GNOME boxes is a tool to handle your virtual machines, integrated to the shell and using QEMU/KVM.

Some other new applications: GNOME contacts, GNOME online accounts, GNOME PackageKit, GNOME color manager, Rygel.

Ekiga is no longer part of GNOME. Many of its features are now available in Empathy.

2.2.10.2 Settings

Most technologies underlying GNOME are still here: the D-Bus messaging system, the PolicyKit permissions manager, the GStreamer multimedia system, the gvfs virtual file system, the MIME system, the ConsoleKit, udisks and upower interfaces to hardware management; all are kept without major changes.

However, the underlying configuration system to GNOME has undergone a major evolution, from GConf to a new system named GSettings, which is much faster and more versatile. The settings can be browsed or edited using the (recommended) gsettings command-line tool, or the dconf-editor graphical tool. The GConf system is still available for third-party applications that use it.

Most settings are migrated upon upgrade, but for technical and conceptual reasons, a selected number of settings are not:

- default session and language (now managed by the accountsservice daemon);
- desktop wallpaper;
- default GTK+ theme (none of the previous themes exist anymore);
- panel and applets configuration (applets now use relative positioning);
- default browser and mailer (the settings are now part of the MIME system through `x-scheme-handler/*` types).

2.2.10.3 Display manager

The GNOME display manager (`gdm3`) has undergone a major evolution together with the desktop. The primary change is that settings for the login prompt have been migrated to GSettings as well. The configuration file has changed to greeter.gsettings and settings are not preserved. This only affects interface settings; daemon settings are still in the same place.

The legacy GDM 2.20 package is no longer available; most of its former features are now available in GDM 3.x.

2.2.10.4 Network management

GNOME now features online connectivity awareness, with several applications and the GNOME shell using **NetworkManager**. This enables support for IPv6 and a wide range of other networking technologies, such as VPNs, wireless and 3G.

GNOME users are strongly advised to use **NetworkManager** for network connectivity; the GNOME components work best with **NetworkManager**. If you are planning on using another network management daemon instead (such as **wicd-daemon**), please see Section Section 5.6.

2.2.11 Cloud

Debian 7.0 includes the OpenStack suite as well as the Xen Cloud Platform (XCP), allowing users to deploy their own cloud infrastructure.

Debian images are also provided on the major public cloud platforms, including Amazon EC2, Windows Azure and Google Compute Engine.

2.2.12 Temporary filesystems

In previous releases, temporary (`tmpfs`) filesystems were mounted on `/lib/init/rw`, `/dev/shm/` and optionally on `/var/lock` and `/var/run`. `/lib/init/rw` has been removed, and the others have been moved under `/run`. `/var/run` and `/var/lock` were configured using `RAMRUN` and `RAMLOCK` in `/etc/default/rcS`. All these `tmpfs` filesystems are now configurable using `/etc/default/tmpfs`; the old settings are not migrated automatically.

Old location	New location	Old setting	New setting
		`/etc/default/rcS`	`/etc/default/tmpfs`
`/lib/init/rw`	`/run`	N/A	N/A
`/var/run`	`/run`	`RAMRUN`	N/A
`/var/lock`	`/run/lock`	`RAMLOCK`	`RAMLOCK`
`/dev/shm`	`/run/shm`	N/A	`RAMSHM`
N/A	`/tmp`	N/A	`RAMTMP`

The migration of data to the new locations will occur automatically during the upgrade and will continue to be available at the old and new locations, with the exception of `/lib/init/rw`. No action is required on your part, though you may wish to customize which `tmpfs` filesystems are mounted, and their size limits, in `/etc/default/tmpfs` after the upgrade is complete. Please see the tmpfs(5) manual page for further details.

If you have written any custom scripts which make use of `/lib/init/rw`, these must be updated to use `/run` instead.

`/tmp` is not a `tmpfs` by default. If you chose to use this feature, please note that:

- the contents of `/tmp` are not preserved across reboots; `/var/tmp` exists for this purpose;
- the maximum size of `/tmp` may (depending upon your specific system) be smaller than before. If you find that there is insufficient free space, it is possible to increase the size limits; see tmpfs(5).
- Applications which create excessively large temporary files may cause `/tmp` to run out of free space. It should be possible to configure a different location for those files by setting the `TMPDIR` environment variable.
- If desired, the defaults may also be overridden with an entry in `/etc/fstab`, for example:

```
tmpfs     /tmp tmpfs     nodev,nosuid,size=20%,mode=1777    0    0
```

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 wheezy can be found together with the Installation Guide on the Debian website (`http://www.debian.org/releases/wheezy/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/wheezy/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 previous official release with Debian 6.0, 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 squeeze, please check the release announcements for the wheezy beta and RC releases available from the Debian Installer's news history (`http://www.debian.org/devel/debian-installer/News/`).

3.1.1 Major changes

New ports Support for the 'armhf' and 's390x' architectures has been added to the installer.

New languages Thanks to the huge efforts of translators, Debian can now be installed in 74 languages, including English. This is three more languages than in squeeze. Most languages are available in both the text-based installation user interface and the graphical user interface, while some are only available in the graphical user interface.

Languages added in this release include:

- Welsh has been re-added to the graphical and text-based installer (it had been removed in squeeze).
- Tibetan and Uyghur have been added to the graphical installer.

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, Kannada, Khmer, Malayalam, Marathi, Nepali, Punjabi, Tamil, Telugu, Tibetan and Uyghur.

Network configuration The installer now supports installation on IPv6-only networks.

It is now possible to install over a WPA-encrypted wireless network.

Default filesystem `ext4` is the default filesystem for new installations, replacing `ext3`.

The `btrfs` filesystem is provided as a technology preview.

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 squeeze installer, you cannot expect these to work with the new installer without modification.

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

Chapter 4

Upgrades from Debian 6.0 (squeeze)

4.1 Preparing for the upgrade

We suggest that before upgrading you also read the information in Chapter 5. That chapter covers potential issues which are not directly related to the upgrade process but 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/apt/extended_states` and the output of `dpkg --get-selections "*"` (the quotes are important). If you use **aptitude** to manage packages on your system, you will also want to back up `/var/lib/aptitude/pkgstates`.

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.

4.1.2 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 have to do a kernel upgrade when upgrading to wheezy, so a reboot will be necessary. Typically, this will be done after the upgrade is finished.

4.1.3 Prepare for downtime on services

There might be services that are offered by the system which are associated with packages that will be included in the upgrade. If this is the case, please note that, during the upgrade, these services will be stopped while their associated packages are being replaced and configured. During this time, these services will not be available.

The precise downtime for these services will vary depending on the number of packages being upgraded in the system, and it also includes the time the system administrator spends answering any configuration questions from package upgrades. Notice that if the upgrade process is left unattended and

the system requests input during the upgrade there is a high possibility of services being unavailable[1] for a significant period of time.

If the system being upgraded provides critical services for your users or the network[2], you can reduce the downtime if you do a minimal system upgrade, as described in Section 4.4.4, followed by a kernel upgrade and reboot, and then upgrade the packages associated with your critical services. Upgrade these packages prior to doing the full upgrade described in Section 4.4.5. This way you can ensure that these critical services are running and available through the full upgrade process, and their downtime is reduced.

4.1.4 Prepare for recovery

Although Debian tries to ensure that your system stays bootable at all times, there is always a chance that you may experience problems rebooting your system after the upgrade. Known potential issues are documented in this and the next chapters of these Release Notes.

For this 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 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, 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, 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 wheezy Debian Installer. The advantage of using the installer is that you can choose between its many installation methods to find 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/wheezy/installmanual`) and the Debian Installer FAQ (`http://wiki.debian.org/DebianInstaller/FAQ`).

4.1.4.1 Debug shell during boot using initrd

The `initramfs-tools` package includes a debug shell[3] 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.

4.1.5 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.

[1] If the debconf priority is set to a very high level you might prevent configuration prompts, but services that rely on default answers that are not applicable to your system will fail to start.

[2] For example: DNS or DHCP services, especially when there is no redundancy or failover. In the DHCP case end-users might be disconnected from the network if the lease time is lower than the time it takes for the upgrade process to complete.

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

Important

If you are using some VPN services (such as `tinc`) they might not be available throughout the upgrade process. Please see Section 4.1.3.

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. Use of the GNOME application **update-manager** is *strongly discouraged* for upgrades to new releases, as this tool relies on the desktop session remaining active.

4.2 Checking system status

The upgrade process described in this chapter has been designed for upgrades from "pure" squeeze 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.

Direct upgrades from Debian releases older than 6.0 (squeeze) are not supported. Please follow the instructions in the Release Notes for Debian 6.0 (`http://www.debian.org/releases/squeeze/releasenotes`) to upgrade to 6.0 first.

This procedure also assumes your system has been updated to the latest point release of squeeze. 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 *squeeze* and not to *stable* or *wheezy*; 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` and `/etc/apt/preferences.d/`) 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 **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"
```

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 **apt-get** can be changed using:

```
# echo package_name hold | dpkg --set-selections
```

Replace `hold` with `install` 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 squeeze 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 wheezy 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 squeeze system. Such packages are most likely to cause problems during an upgrade as they may result in file conflicts[4]. Section 4.5 has some information on how to deal with file conflicts if they should occur.

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

4.3 Preparing sources for APT

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

`apt` will consider all packages that can be found via any "`deb`" line, and install the package with the highest version number, giving priority to the first 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).

A release can often be referred to both by its codename (e.g. `squeeze`, `wheezy`) 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.

4.3.1 Adding APT Internet sources

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

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/wheezy/main/binary-mipsel/...
http://mirrors.kernel.org/debian/dists/wheezy/contrib/binary-mipsel/...
```

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

```
deb http://mirrors.kernel.org/debian wheezy 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.

4.3.2 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/wheezy/main/binary-mipsel/...
/var/ftp/debian/dists/wheezy/contrib/binary-mipsel/...
```

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

```
deb file:/var/ftp/debian wheezy 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.

4.3.3 Adding APT sources from optical media

If you want to use *only* CDs (or DVDs or Blu-ray Discs), 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/scd0` is your CD-ROM drive, `/etc/fstab` should contain a line like:

```
/dev/scd0 /cdrom auto noauto,ro 0 0
```

Note that there must be *no spaces* between the words `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.

4.4 Upgrading packages

The recommended way to upgrade from previous Debian releases is to use the package management tool **apt-get**. In previous releases, **aptitude** was recommended for this purpose, but recent versions of **apt-get** provide equivalent functionality and also have shown to more consistently give the desired upgrade results.

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 "`wheezy`" or to "`stable`". There should not be any sources entries pointing to squeeze.

> **Note**
>
> Source lines for a CD-ROM might sometimes refer to "`unstable`"; although this may be confusing, you should *not* change it.

4.4.1 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-wheezystep.time -a ~/upgrade-wheezystep.script
```

or similar. If you have to rerun the typescript (e.g. if you have to reboot the system) use different *`step`* values to indicate which step of the upgrade you are logging. 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. If you are at the system's console, just switch to VT2 (using Alt+F2) and, after logging in, use `less -R ~root/upgrade-wheezy.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-wheezy.time ~/upgrade-wheezy.script
```

4.4.2 Updating the package list

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

```
# apt-get update
```

4.4.3 Make sure you have sufficient space for the upgrade

You have to make sure before upgrading your system that you will have sufficient hard disk space when you start the full system upgrade described in Section 4.4.5. 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 is difficult to recover from.

apt-get can show you detailed information about the disk space needed for the installation. Before executing the upgrade, you can see this estimate by running:

```
# apt-get -o APT::Get::Trivial-Only=true dist-upgrade
[ ... ]
XXX upgraded, XXX newly installed, XXX to remove and XXX not upgraded.
Need to get xx.xMB of archives.
After this operation, AAAMB of additional disk space will be used.
```

> **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 4.4.4 before running this command to estimate the disk space.

If you do not have enough space for the upgrade, **apt-get** will warn you with a message like this:

```
E: You don't have enough free space in /var/cache/apt/archives/.
```

In this situation, 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** will remove all previously downloaded package files.
- Remove forgotten packages. If you have used **aptitude** or **apt-get** to manually install packages in squeeze it will have kept track of those packages you manually installed, and will be able to mark as redundant those packages pulled in by dependencies alone which are no longer needed due to a package being removed. They will not mark for removal packages that you manually installed. To remove automatically installed packages that are no longer used, run:

  ```
  # apt-get autoremove
  ```

  You can also use **deborphan**, **debfoster**, or **cruft** to find redundant packages. Do not blindly remove the packages these tools present, especially if you are using aggressive non-default options that are prone to false positives. It is highly recommended that you manually review the packages suggested for removal (i.e. their contents, sizes, and descriptions) before you remove them.
- Remove packages that take up too much space and are not currently needed (you can always reinstall them after the upgrade). 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 find the

packages that just take up the most disk space with **dpigs** (available in the `debian-goodies` package) or with **wajig** (running `wajig size`). They can also be found with `aptitude`. Start **aptitude** in "visual mode", select Views → New Flat Package List, press **l** and enter `~i`, then press **S** and enter `~installsize`. This will give you a handy list to work with.

- 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.

- Do a minimal upgrade of the system (see Section 4.4.4) or partial upgrades of the system followed by a full upgrade. This will make it possible to upgrade the system partially, and allow you to clean the package cache before the full upgrade.

Note that in order to safely remove packages, it is advisable to switch your `sources.list` back to squeeze as described in Section A.2.

4.4.4 Minimal system upgrade

In some cases, doing the full upgrade (as described below) directly might 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 upgrade as described in Section 4.4.5.

To do this, first run:

```
# apt-get upgrade
```

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

The minimal system upgrade can also be useful when the system is tight on space and a full upgrade cannot be run due to space constraints.

If the `apt-listchanges` package is installed, it will (in its default configuration) show important information about upgraded packages in a pager. Press **q** after reading to exit the pager and continue the upgrade.

4.4.5 Upgrading the system

Once you have taken the previous steps, you are now ready to continue with the main part of the upgrade. Execute:

```
# apt-get dist-upgrade
```

> **Note**
>
> The upgrade process for some previous releases recommended the use of **aptitude** for the upgrade. This tool is not recommended for upgrades from squeeze to wheezy.

This will perform a complete upgrade of the system, installing the newest available versions of all packages, and resolving 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 interrelated 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 `apt-get -f install` *`package`*.

4.5 Possible issues during upgrade

The following sections describe known issues that might appear during an upgrade to wheezy.

4.5.1 Dist-upgrade fails with "Could not perform immediate configuration"

In some cases the **apt-get dist-upgrade** step can fail after downloading packages with:

```
E: Could not perform immediate configuration on 'package'.  Please see man 5 apt. ↩
    conf under APT::Immediate-Configure for details.
```

If that happens, running **apt-get dist-upgrade -o APT::Immediate-Configure=0** instead should allow the upgrade to proceed.

Another possible workaround for this problem is to temporarily add both squeeze and wheezy sources to your `sources.list` and run **apt-get update**.

4.5.2 Expected removals

The upgrade process to wheezy might ask for the removal of packages on the system. The precise list of packages will vary depending on the set of packages that you have installed. These release notes give general advice on these removals, but if in doubt, it is recommended that you examine the package removals proposed by each method before proceeding.

4.5.3 Conflicts or Pre-Depends loops

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. **apt-get** 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 **apt-get** command line.

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

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

to eliminate some of the offending packages, or

```
# apt-get -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
```

4.5.4 File conflicts

File conflicts should not occur if you upgrade from a "pure" squeeze 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 **apt-get** commands.

4.5.5 Configuration changes

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` directory, 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.

4.5.6 Change of session to console

If you are running the upgrade using the system's local console you might find that at some points during the upgrade the console is shifted over to a different view and you lose visibility of the upgrade process. For example, this may happen in desktop systems when the display manager is restarted.

To recover the console where the upgrade was running you will have to use Ctrl+Alt+F1 (if in the graphical startup screen) or Alt+F1 (if in the local text-mode console) to switch back to the virtual terminal 1. Replace F1 with the function key with the same number as the virtual terminal the upgrade was running in. You can also use Alt+Left Arrow or Alt+Right Arrow to switch between the different text-mode terminals.

4.5.7 Special care for specific packages

In most cases, packages should upgrade smoothly between squeeze and wheezy. There are a small number of cases where some intervention may be required, either before or during the upgrade; these are detailed below on a per-package basis.

4.5.7.1 Sudo

If you have modified `/etc/sudoers` then you should be aware of changes made to how `sudo` configuration is handled. The default `/etc/sudoers` now includes the following two directives:

```
Defaults        secure_path="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/ ↩
    sbin:/bin"
```

```
#includedir /etc/sudoers.d
```

Neither of these entries are added to your `/etc/sudoers` automatically during the upgrade. (Although you will still be able to run **sudo** commands by specifying their fully-qualified path.) So you might wish to consider migrating your changes to the new `/etc/sudoers.d` directory and using the default `/etc/sudoers` file. For example:

```
# mv /etc/sudoers /etc/sudoers.d/mychanges
# mv /etc/sudoers.dpkg-new /etc/sudoers
```

You may also need to edit your `/etc/sudoers.d/mychanges` to remove unwanted `Defaults` and `#includedir` entries. You should use **visudo** for this:

```
# visudo -f /etc/sudoers.d/mychanges
```

4.5.7.2 Screen

The GNU Screen versions in squeeze and wheezy don't use the same communication protocol between the **screen** client and the **SCREEN** server. Wheezy's `screen` package has been patched so that the most important functionality is present even if the versions of **screen** client and server don't match.

The most prominent functionality which does not work properly when connecting to a Screen session started with squeeze's version of `screen` with wheezy's version of `screen` as client is terminal resizing (`WINCH` signal). The workaround is to detach and reattach again to get the size of the terminals inside the screen session adjusted properly.

Some ncurses-based applications, e.g. **aptitude** in visual mode, may leave traces of previous contents on the screen. Pressing Ctrl+L solves the issue.

Another (harmless) symptom of such an inter-version connection is **screen** issuing messages like "Message 40 of 12376 bytes too small".

All these issues vanish as soon as the Screen sessions started with squeeze's version of `screen` are exited.

See also `/usr/share/doc/screen/NEWS.Debian.gz` in wheezy's `screen` package.

4.5.7.3 Suhosin PHP module

The `php5-suhosin` package has been removed. If your PHP configuration included the suhosin module, it will fail to load after the PHP upgrade. Run **dpkg --purge php5-suhosin** to remove the leftover configuration in `/etc/php5/conf.d/suhosin.ini`.

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 squeeze to wheezy, it is strongly recommended that you install a linux-image-* metapackage, if you haven't done so before. 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 metapackages, run:

```
# apt-cache search linux-image- | 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 '`2.6.32-5-amd64`', it is recommended that you install `linux-image-amd64`. 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-amd64
```

You should then use `apt-get 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. Install the kernel sources, provided in the `linux-source` package. You can make use of the `deb-pkg` target available in the sources' makefile for building a binary package. More information can be found in the Debian Linux Kernel Handbook (`http://kernel-handbook.alioth.debian.org/`), which can also be found as the `debian-kernel-handbook` 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 4.4.4.

4.6.2 Boot timing issues (waiting for root device)

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

The usual symptoms are that the boot will fail because the root file system cannot be mounted and you are dropped into a debug shell:

```
Gave up waiting for root device.  Common problems:
 - Boot args (cat /proc/cmdline)
   - Check rootdelay= (did the system wait long enough?)
   - Check root= (did the system wait for the right device?)
 - Missing modules (cat /proc/modules; ls /dev)
ALERT!  /dev/something does not exist.  Dropping to a shell!
(initramfs)
```

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 Preparing for the next release

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

- Remove newly redundant or obsolete packages as described in Section 4.4.3 and Section 4.8. You should review which configuration files they use and consider purging the packages to remove their configuration files.

4.8 Obsolete packages

Introducing several thousand new packages, wheezy also retires and omits more than four thousand old packages that were in squeeze. 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 wheezy's release[5], 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 wheezy 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.

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:

- `mysql-5.1`, successor is `mysql-5.5`.
- `postgresql-8.4`, successor is `postgresql-9.1`. Wheezy provides only an updated `postgresql-plperl-8.4` package that is linked against the new version of libperl in order to enable upgrading to the new Perl version in wheezy without making existing postgresql-8.4 installations unusable. Once the operating system upgrade is finished, you should plan to also upgrade your PostgreSQL 8.4 database clusters to the new PostgreSQL version 9.1 using the **pg_upgradecluster** tool.
- `python2.5`, successor is `python2.7`.
- `portmap`, successor is `rpcbind`.
- `sun-java6`, successor is `openjdk-7`.
- `gdm`, successor is `gdm3`. Users of lightweight desktop environments such as Xfce or LXDE may wish to consider `lightdm` as a lighter weight alternative.
- `mpich`, successors are `openmpi` and `mpich2`.
- The `compiz` OpenGL window and compositing manager, see bugreports #677864 (`http://bugs.debian.org/677864`) (and #698815 (`http://bugs.debian.org/698815`)).
- Some of Xorg's video drivers are no longer available in wheezy and are obsolete. This includes `xserver-xorg-video-nv` and `xserver-xorg-video-radeonhd`. They may be removed during the upgrade. Users should install `xserver-xorg-video-all` instead.
- All Horde 3 packages, providing web collaborative software, have been removed and are obsolete. This includes `ansel1`, `chora2`, `dimp1`, `gollem`, `horde-sam`, `horde3`, `imp4`, `ingo1`, `kronolith2`, `mnemo2`, `nag2`, `sork-forwards-h3`, `sork-passwd-h3`, `sork-vacation-h3` and `turba2`. As the Horde 4 packages have not reached sufficient quality before the wheezy release, they are also not available. They may be available in testing as `php-horde-*` packages.
- Most Kolab packages, providing groupware server, have been removed. This includes `kolab-cyrus-imapd`, `kolab-webadmin`, `kolabd`, `libkolab-perl`, `php-kolab-filter` and `php-kolab-freebusy`. As of 2012, Kolab was in a major rewrite and may get shipped with a later Debian release as the `kolab` package. NB: The SOGo server (formerly named Scalable OpenGroupware.org) is shipped with wheezy as `sogo`.

[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.

- All OpenERP 5 packages have been removed and are obsolete. This includes `openerp-client`, `openerp-server`, `openerp-web`.
- The `pootle` 2.0.5 package has been removed.
- The `uw-imapd` and `ipopd` packages have been removed. Better alternatives exist, for example `dovecot-imapd` and `courier-imap` for IMAP, or `dovecot-pop3d` and `courier-pop` for POP3.
- The `drupal6` package is no longer available; it is replaced by `drupal7`. However, no automatic upgrade path exists, and users should read the instructions on the Debian Wiki (`http://wiki.debian.org/Drupal/Upgrade/From6To7`).

4.8.1 Dummy packages

Some packages from squeeze have been split into several packages in wheezy, often to improve system maintainability. To ease the upgrade path in such cases, wheezy often provides "dummy" packages: empty packages that have the same name as the old package in squeeze with dependencies that cause the new packages to be installed. These "dummy" packages are considered redundant 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 (e.g. `--guess-dummy`) useful to detect them in your system. Note that some dummy packages are not intended to be removed after an upgrade but are, instead, used to keep track of the current available version of a program over time.

Chapter 5

Issues to be aware of for wheezy

Sometimes, changes introduced in a new release have side-effects we cannot reasonably avoid, or they expose bugs somewhere else. This section documents 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 LDAP support

A feature in the cryptography libraries used in the LDAP libraries causes programs that use LDAP and attempt to change their effective privileges to fail when connecting to an LDAP server using TLS or SSL. This can cause problems for setuid programs on systems using `libnss-ldap` like **sudo**, **su** or **schroot** and for setuid programs that perform LDAP searches like `sudo-ldap`.

It is recommended to replace the `libnss-ldap` package with `libnss-ldapd`, a newer library which uses a separate daemon (**nslcd**) for all LDAP lookups. The replacement for `libpam-ldap` is `libpam-ldapd`.

Note that `libnss-ldapd` recommends the NSS caching daemon (`nscd`) which you should evaluate for suitability in your environment before installing. As an alternative to `nscd` you can consider `unscd`.

Further information is available in bugs #566351 (`http://bugs.debian.org/566351`) and #545414 (`http://bugs.debian.org/545414`).

5.2 Security status of web browsers

Debian 7.0 includes several browser engines which are affected by a steady stream of security vulnerabilities. The high rate of vulnerabilities and partial lack of upstream support in the form of long term branches make it very difficult to support these browsers with backported security fixes. Additionally, library interdependencies make it impossible to update to newer upstream releases. Therefore, browsers built upon the webkit, qtwebkit and khtml engines are included in Wheezy, but not covered by security support. These browsers should not be used against untrusted websites.

For general web browser use we recommend browsers building on the Mozilla xulrunner engine (Iceweasel and Iceape) or Chromium.

Xulrunner has had a history of good backportability for older releases over the previous release cycles. Chromium - while built upon the Webkit codebase - is a leaf package, which will be kept up-to-date by rebuilding the current Chromium releases for stable.

5.3 ConsoleKit and alternative display managers

ConsoleKit in Debian 7.0 does not consider sessions started using **startx** or display managers lacking `consolekit` integration (e.g. `xdm` or `slim`) as local, which might prevent access to some devices.

We recommend using one of `gdm3`, `kdm` or `lightdm` instead.

5.4 GNOME desktop changes and support

By default, some accessibility tools are not enabled in the GNOME display manager (`gdm3`). The simplest way to enable zooming or a visual keyboard is to activate the "shell" greeter.

To do that, edit the `/etc/gdm3/greeter.gsettings` file, and uncomment the following:

```
session-name='gdm-shell'
```

while commenting

```
session-name='gdm-fallback'
```

Note that it requires a compatible 3D graphics card — which is the reason why it is not enabled by default.

5.5 KDE desktop changes

The `knetworkmanager` package has been deprecated, and replaced by `plasma-widget-networkmanagement` in the new KDE Plasma Workspace.

If you are using the deprecated **knetworkmanager** standalone application, you should be prepared to do some manual configuration after the upgrade. You might need to manually add plasma-widget-networkmanagement to your panel or desktop.

Also, if the network connection shouldn't depend on having a network-manager widget running, you might want to set it as a "system connection".

5.6 NetworkManager

NetworkManager can detect if a network interface is managed by **ifupdown** in order to avoid conflicts, but is not able to do so with other network management programs such as **wicd-daemon**. Problems and unexpected behavior can result if two such daemons are managing the same interface when attempting to make a network connection.

For instance, if **wicd-daemon** and **NetworkManager** are both running, attempting to use a **wicd** client to make a connection will fail with the error message:

```
Connection Failed: bad password
```

Attempting to use a **NetworkManager** client may likewise fail with the message:

```
NetworkManager is not running.  Please start it.
```

It is recommended that users of GNOME consider installing and trying **NetworkManager**, but the **NetworkManager** daemon may be permanently disabled if desired using the following command:

```
# update-rc.d network-manager disable
```

After disabling the daemon, it is recommended to examine the contents of `/etc/resolv.conf`. This file is used to specify DNS servers for name resolution and the contents of this file may have been replaced by **NetworkManager**.

5.7 `perl-suid` removed

suidperl was removed upstream with 5.12, so the `perl-suid` package which used to be distributed in Debian has been removed too. Possible alternatives include using a simple setuid C wrapper to execute a Perl script from a hard-coded location, or using a more general tool like **sudo**.

5.8 Request Tracker versions

If you have `request-tracker3.8` installed on your squeeze system, note that this package has been removed from wheezy, to be replaced by `request-tracker4`. Some manual steps are required to upgrade between `request-tracker3.8` and `request-tracker4`: please install `request-tracker4` alongside your existing `request-tracker3.8` installation and consult the installation/upgrade notes

in `/usr/share/doc/request-tracker4/README.Debian.gz` (section: "Upgrading from request-tracker3.8 to request-tracker4").

The same advice applies if you have `request-tracker3.6` or older packages from previous Debian releases still in use; if this is the case it is recommended to upgrade step by step, following the appropriate upgrade documents.

5.9 Bootlogd changes

bootlogd has moved from `sysvinit-utils` to a separate `bootlogd` package. If you wish to continue using bootlogd, you need to install the `bootlogd` package. Note that the configuration file `/etc/default/bootlogd` and its option `BOOTLOGD_ENABLE` no longer exist; if you do not wish to run bootlogd, remove the `bootlogd` package.

5.10 `/etc/mtab` and `_netdev`

The file `/etc/mtab`, used to store the list of currently mounted filesystems, has been changed to be a symbolic link to `/proc/mounts`. For almost every case, this change will result in a more robust system since the list can never become inconsistent with reality. However, if you use the `_netdev` option in `/etc/fstab` to indicate that a filesystem is a network filesystem requiring special handling, this will no longer be set in `/proc/mounts` after rebooting. This will *not* cause problems for standard network filesystems such as NFS, which do not rely on the `_netdev` option. Filesystems which are *unaffected* by this issue are `ceph`, `cifs`, `coda`, `gfs`, `ncp`, `ncpfs`, `nfs`, `nfs4`, `ocfs2` and `smbfs`. For filesystems which *do* rely on `_netdev` for correct unmounting at shutdown, for example when using an NBD, a static mtab will be the only way to use `_netdev` in wheezy. If you have such a setup, then after completing the upgrade to wheezy restore a static `/etc/mtab` by doing the following:

- Edit `/etc/init.d/checkroot.sh`, and comment out these lines:

```
        if [ "$rootmode" != "ro" ]; then
                mtab_migrate
        fi
```

- If you have rebooted the system, and `/etc/mtab` is now a symbolic link:

```
# rm /etc/mtab
# cp /proc/mounts /etc/mtab
```

  Re-add the `_netdev` option by remounting the affected filesystems:

```
# mount -o remount filesystem
```

  `/etc/mtab` will be recreated fully next time you reboot the system.

5.11 The pdksh to mksh transition

The Public Domain Korn Shell (`pdksh`) package is being retired for the release after wheezy, since **pdksh** is no longer maintained (it has not been actively developed since 1999).

The MirBSD Korn Shell (`mksh`) package contains its successor; it has evolved from the Public Domain Korn Shell and has been kept up to date with the POSIX standard on the shell. In Debian wheezy, `pdksh` is a transitional package using **lksh**, a variant of `mksh` built with special compatibility options to provide a **pdksh** binary symlink. This compatibility binary behaves more like the traditional Public Domain Korn Shell than the current **mksh**. However as it contains behavior-changing bugfixes it is not a pure drop-in replacement. So, you're advised to change your

```
#!/bin/pdksh
```

scripts to

```
#!/bin/mksh
```

and test them. If the test fails, you're advised to fix your scripts. If, for some reason, this is not possible, you can change them to

```
#!/bin/lksh
```

scripts, and test them again. This test has more chances of succeeding without changing a lot of your code. However, be aware at some point in the future the transitional package will get dropped from Debian.

The compatibility binary is not suitable for interactive use, so as system administrator, adjust the login shell of your Korn Shell users. For minimal service interruption, do this before the upgrade of the O.S.: manually install the `mksh` package and change the login and/or interactive shells of users that use **pdksh** to **mksh**. Furthermore, you're encouraged to copy `/etc/skel/.mkshrc` into their home directories: this provides some shell functions like **pushd**, **popd** and **dirs** and a nice

```
PS1
```

(shell prompt).

5.12 Puppet 2.6 / 2.7 compatibility

When upgrading a Puppet managed system from squeeze to wheezy, you must ensure that the corresponding puppetmaster runs at least Puppet version 2.7. If the master is running squeeze's `puppetmaster`, the managed wheezy system will not be able to connect to it.

Such a combination will lead to the following error message during a **puppet agent** run:

```
Could not retrieve catalog from remote server: Error 400 on SERVER: No support ↩
    for http method POST
```

In order to resolve this issue the puppetmaster must be upgraded. A 2.7 master is able to manage a 2.6 client system.

5.13 Multiarch implications for the toolchain

The introduction of multiarch (as described in Section 2.2.2) changes the paths for some files, which may break assumptions made by toolchain components. Debian's toolchain has been updated, but users trying to build or use external compilers might need to be aware of this.

Some hints to work around these issues can be found in `/usr/share/doc/libc6/NEWS.Debian.gz` and in bugreport #637232 (`http://bugs.debian.org/637232`).

5.14 Cyrus SASL SQL backends

Configuration of SQL engine backends for Cyrus SASL, as provided in the `libsasl2-modules-sql` package, has changed from database specific configuration (e.g. `mysql`) to the generic `sql` auxprop plugin.

Configuration files for applications using SASL have to be updated, for example:

```
auxprop_plugin: mysql
```

should be replaced by:

```
auxprop_plugin: sql
sql_engine: mysql
```

In addition, the SQL query (if used) needs to have `%u` replaced with `%u@%r`, because user and realm are now provided separately.

5.15 Firmware for network and graphics drivers

Some hardware drivers, including drivers for (wired or wireless) network cards, as well as the driver for ATI/AMD graphics chipsets, require loadable firmware in order to operate properly.

That firmware is often not free software, and as such only available from the non-free archive, in the `firmware-linux` and other (`http://packages.debian.org/search?keywords=firmware&searchon=names&suite=wheezy&section=all`) packages.

Chapter 6

More information on Debian

6.1 Further reading

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

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 to these sources which may be helpful for new Debian users.

6.2.1 Mailing lists

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

6.2.2 Internet Relay Chat

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

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

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

6.3 Reporting bugs

We strive to make Debian 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/cvs`) 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 (`http://www.debian.org/devel/#projects`) which include ports to particular architectures and Debian Pure Blends (`http://wiki.debian.org/DebianPureBlends`) for specific user groups, among many others.

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.

Chapter 7

Glossary

ACPI
: Advanced Configuration and Power Interface

ALSA
: Advanced Linux Sound Architecture

APM
: Advanced Power Management

BD
: Blu-ray Disc

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 Standard Base

LVM
Logical Volume Manager

MTA
Mail Transport Agent

NBD
Network Block Device

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

SSL
Secure Sockets Layer

TLS
Transport Layer Security

USB
Universal Serial Bus

UUID
Universally Unique Identifier

VGA
Video Graphics Array

WPA
Wi-Fi Protected Access

Appendix A

Managing your squeeze system before the upgrade

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

A.1 Upgrading your squeeze system

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

If you upgrade your system using a Debian mirror, it will automatically be upgraded to the latest squeeze 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" wheezy. This might not be what you want if you are not ready yet for the upgrade. If you have already run `apt-get update`, you can still get back without problems by following the procedure below.

If you have also already installed packages from wheezy, there probably is not much point in installing packages from squeeze 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 `squeeze`.

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

A.3 Removing obsolete configuration files

Before upgrading your system to wheezy, it is recommended to remove old configuration files (such as `*.dpkg-{new,old}` files under `/etc`, as well as the file `/etc/X11/XF86Config-4`[1]) from the system.

A.4 Upgrade legacy locales to UTF-8

If your system is localized and is using a locale that is not based on UTF-8 you should strongly consider switching your system over to using UTF-8 locales. In the past, there have been bugs[2] identified that manifest themselves only when using a non-UTF-8 locale. On the desktop, such legacy locales are supported through ugly hacks in the library internals, and we cannot decently provide support for users who still use them.

To configure your system's locale you can run **dpkg-reconfigure locales**. Ensure you select a UTF-8 locale when you are presented with the question asking which locale to use as a default in the system. In addition, you should review the locale settings of your users and ensure that they do not have legacy locale definitions in their configuration environment.

[1] Since release 2:1.7.7-12, xorg-server no longer reads the file XF86Config-4. See also #619177 (`http://bugs.debian.org/619177`).

[2] In the GNOME screensaver, using passwords with non-ASCII characters, pam_ldap support, or even the ability to unlock the screen may be unreliable when not using UTF-8. The GNOME screenreader is affected by bug #599197 (`http://bugs.debian.org/599197`). The Nautilus file manager (and all glib-based programs, and likely all Qt-based programs too) assume that filenames are in UTF-8, while the shell assumes they are in the current locale's encoding. In daily use, non-ASCII filenames are just unusable in such setups. Furthermore, the gnome-orca screen reader (which grants sight-impaired users access to the GNOME desktop environment) requires a UTF-8 locale since Squeeze; under a legacy characterset, it will be unable to read out window information for desktop elements such as Nautilus/GNOME Panel or the Alt-F1 menu.

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, David Prévot, Eddy Petrișor, Emmanuel Kasper, Esko Arajärvi, Frans Pop, Giovanni Rapagnani, Gordon Farquharson, Javier Fernández-Sanguino Peña, Jens Seidel, Jonas Meurer, Jonathan Nieder, Josip Rodin, Julien Cristau, 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 Bienlein, Simon Paillard, Stefan Fritsch, Steve Langasek, Steve McIntyre, Tobias Scherer, Vincent McIntyre, and W. Martin Borgert.

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

Index

A
Abiword, 4
Apache, 4

B
BIND, 4
Blu-ray, 5

C
Calligra, 4
CD, 4
Courier, 4

D
Dia, 4
DocBook XML, 2
DVD, 4

E
Evolution, 4
Exim, 4

F
Firefox, 4

G
GCC, 4
GIMP, 4
GNOME, 4
GNUcash, 4
GNUmeric, 4

K
KDE, 4

L
LDAP, 25
LibreOffice, 4
LILO, 22
Linux Standard Base, 4
LXDE, 4

M
Mozilla, 4
MySQL, 4

O
OpenSSH, 4

P
packages
- ansel1, 23
- apt, 1, 2, 15
- apt-listchanges, 19
- aptitude, 18
- bootlogd, 27
- chora2, 23
- compiz, 23
- consolekit, 25
- courier-imap, 24
- courier-pop, 24
- dblatex, 2
- debian-goodies, 18
- debian-kernel-handbook, 22
- dimp1, 23
- doc-debian, 32
- docbook-xsl, 2
- dovecot-imapd, 24
- dovecot-pop3d, 24
- drupal6, 24
- drupal7, 24
- ffmpeg, 5
- file-rc, 5
- firmware-linux, 29
- gcc, 5
- gdm, 23
- gdm3, 7, 23, 25, 26
- gollem, 23
- hardening-wrapper, 5
- horde-sam, 23
- horde3, 23
- icedove, 4
- iceweasel, 4
- imp4, 23
- ingo1, 23
- initramfs-tools, 12, 21, 22
- ipopd, 24
- kdm, 25
- knetworkmanager, 26
- kolab, 23
- kolab-cyrus-imapd, 23
- kolab-webadmin, 23
- kolabd, 23
- kronolith2, 23
- libav-tools, 5
- libkolab-perl, 23
- libnss-ldap, 25
- libnss-ldapd, 25
- libpam-ldap, 25
- libpam-ldapd, 25
- libsasl2-modules-sql, 28
- lightdm, 23, 25
- linux-image-*, 21
- linux-image-amd64, 22
- linux-source, 22
- localepurge, 18
- mksh, 27, 28
- mnemo2, 23
- mpich, 23
- mpich2, 23
- mysql-5.1, 23
- mysql-5.5, 23
- nag2, 23
- nscd, 25
- openerp-client, 24

openerp-server, 24
openerp-web, 24
openjdk-7, 23
openmpi, 23
pdksh, 27
perl-suid, 26
php-horde-*, 23
php-kolab-filter, 23
php-kolab-freebusy, 23
php5-suhosin, 21
plasma-widget-networkmanagement, 26
pootle, 24
popularity-contest, 17
portmap, 23
postgresql-8.4, 23
postgresql-9.1, 23
postgresql-plperl-8.4, 23
puppetmaster, 28
python2.5, 23
python2.7, 23
release-notes, 1
request-tracker3.6, 27
request-tracker3.8, 26
request-tracker4, 26
rpcbind, 23
screen, 21
slim, 25
sogo, 23
sork-forwards-h3, 23
sork-passwd-h3, 23
sork-vacation-h3, 23
sudo, 21
sudo-ldap, 25
sun-java6, 23
systemd, 5
sysvinit, 5
sysvinit-utils, 27
tinc, 13
turba2, 23
udev, 21, 22
unscd, 25
upgrade-reports, 1
uw-imapd, 24
xdm, 25
xmlroff, 2
xserver-xorg-video-all, 23
xserver-xorg-video-nv, 23
xserver-xorg-video-radeonhd, 23
xsltproc, 2
Perl, 4
PHP, 4
Pidgin, 4
Postfix, 4
PostgreSQL, 4

T
Thunderbird, 4

X
Xfce, 4