Release Notes for Debian 8 (jessie), 64-bit PC

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

May 18, 2015

Contents

Chapter 1

Introduction

This document informs users of the Debian distribution about major changes in version 8 (codenamed jessie).

The release notes provide information on how to upgrade safely from release 7 (codenamed wheezy) 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 https://www.debian.org/releases/jessie/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 wheezy). If you need to upgrade from older releases, we suggest you read previous editions of the release notes and upgrade to wheezy 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 (https://bugs.debian.org/) against the release-notes package. You might want to review first the existing bug reports (https://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 wheezy to jessie. If you are willing to share information please file a bug in the bug tracking system (https://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 (`https://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 (`https://www.debian.org/doc/cvs`).

Chapter 2

What's new in Debian 8

The Wiki (https://wiki.debian.org/NewInJessie) has more information about this topic.

2.1 Supported architectures

Debian 8 introduces two new architectures:

- arm64, 64-bit port for ARM machines.
- ppc64el, 64-bit little-endian port for POWER machines.

The following are the officially supported architectures for Debian 8:

- 32-bit PC ('i386') and 64-bit PC ('amd64')
- 64-bit ARM ('arm64')
- ARM EABI ('armel')
- ARMv7 (EABI hard-float ABI, 'armhf')
- MIPS ('mips' (big-endian) and 'mipsel' (little-endian))
- PowerPC ('powerpc')
- 64-bit little-endian PowerPC ('ppc64el')
- IBM System z ('s390x')

Three architectures which were part of Debian 7 are not released with jessie.

- As announced when Debian 7 was released, the 32-bit `s390` port is discontinued and replaced with `s390x`.
- In addition, the ports to IA-64 and Sparc had to be removed from this release due to insufficient developer support. Sparc had been a supported architecture in Debian since 2.1 (1999), while `ia64` was introduced in Debian 3.0 (2002).

Finally, the Debian ports to the FreeBSD kernel, kfreebsd-amd64 and kfreebsd-i386, included as technology previews in Debian 6.0 and Debian 7, are not part of this release.

You can read more about port status, and port-specific information for your architecture at the Debian port web pages (https://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 wheezy; the distribution includes over 12253 new packages, for a total of over 43512 packages. Most of the software in the distribution has been updated: over 24573 software packages (this is 66% of all packages in wheezy). Also, a significant number of packages (over 5441, 14% of the packages in wheezy) 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; see Section 4.10.

Debian again ships with several desktop applications and environments. Among others it now includes the desktop environments GNOME 3.14, KDE 4.11, Xfce 4.10, and LXDE.

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

- LibreOffice is upgraded to version 4.3;
- Calligra is upgraded to 2.8;
- GNUcash is upgraded to 2.6;
- GNUmeric is upgraded to 1.12;
- Abiword is upgraded to 3.0.

Updates of other desktop applications include the upgrade to Evolution 3.12.

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

Package	Version in 7 (wheezy)	Version in 8 (jessie)
Apache	2.2.22	2.4.10
BIND DNS Server	9.8	9.9
Courier MTA	0.68	0.73
Dia	0.97.2	0.97.3
Exim default e-mail server	4.80	4.84
GNU Compiler Collection as default compiler	4.7 on PCs, 4.6 elsewhere	4.9
the GNU C library	2.13	2.19
lighttpd	1.4.31	1.4.35
Linux kernel image	3.2 series	3.16 series
OpenLDAP	2.4.31	2.4.40
OpenSSH	6.0p1	6.7p1
Perl	5.14	5.20
PHP	5.4	5.6
Postfix MTA	2.9	2.11
PostgreSQL	9.1	9.4
Python 3	3.2	3.4
Samba	3.6	4.1

2.2.1 CDs, DVDs, and BDs

The official Debian distribution now ships on 9 to 10 binary DVDs or 75 to 85 binary CDs (depending on the architecture) and 10 source DVDs or 59 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 Changes in the GNOME desktop

Being upgraded to version 3.14, the new GNOME desktop brings many new features and usability improvements.

The design of the GNOME shell has been updated. The bottom message tray is larger, easier to use and less prone to appear accidentally. A new system status area in the upper right corner puts all useful

settings in the same place.

The screensaver has been replaced by a "lock screen" that still brings minimal functionality when the user is away. For example, you can pause the music, be informed of a new e-mail, or change the screen brightness, all of that without entering a password. Pressing the Escape key or starting to type the password brings back the login prompt. The GNOME display manager uses the exact same design for consistency.

Several applications, including the Nautilus file manager, the gedit text editor, and the evince document viewer have seen their design made much more compact, merging the window titlebar with the toolbar. This leaves much more room for the documents the user is working on. The standard dialog boxes have undergone similar changes.

Support for touch screens has been fully integrated, including intuitive gestures based on multiple finger movements. GNOME now also supports high resolution (HiDPI) screens, taking full advantage of fine pixeling for the clearest rendering.

GNOME supports geolocation, and includes a smooth mapping and navigation application.

The user documentation is much more complete, and includes video tutorials for new users.

More information can be found in the GNOME 3.14 release notes (`https://help.gnome.org/misc/release-notes/3.14/`).

2.2.3 New default init system (systemd)

Introduced in Debian 7, `systemd` is now the default init system. It provides 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. The `systemd-sysv` package provides the `/sbin/init` replacement.

For more information on this topic, see the Debian wiki (`http://wiki.debian.org/systemd`).

2.2.4 Security

The legacy secure sockets layer protocol SSLv3 has been disabled in this release. Many system cryptography libraries as well as servers and client applications have been compiled or configured without support for this protocol.

The Linux kernel features a security mechanism which nullifies many symlink attacks. It is enabled in the Debian Linux kernel by default. /tmp-related bugs which are rendered non-exploitable by this mechanism are not treated as security vulnerabilities. If you use a custom Linux kernel you should enable it using a sysctl setting:

```
echo 1 > /proc/sys/fs/protected_symlinks
```

In some rare cases the security support for a package shipped in a Debian release needs to be terminated prior to the end of support for the full distribution. Jessie provides a new package (`debian-security-support`) which emits a warning if support for a package needs to be terminated in advance. It also documents packages where the scope of security support is limited. As such, it is recommended to install debian-security-support on all security-relevant systems.

Continuing on the path set by Wheezy, more packages have been built with hardened compiler flags. Also, the stack protector flag has been switched to stack-protector-strong for extra hardening. Note that the hardened build flags are not enabled by default in `gcc`, so they are not used automatically when locally building software.

New in this release is the `needrestart` package. When installed, it will perform a check after each APT upgrade session. If any services running on the system require a restart to take advantage of changes in the upgraded packages then it offers to perform these restarts. It is recommended to install `needrestart` to ensure that security updates in libraries are propagated to running services.

2.2.5 MariaDB next to MySQL

Along with the older MySQL 5.5, Jessie ships the new MariaDB 10.0. See the Debian MySQL Team wiki page (`http://wiki.debian.org/Teams/MySQL`) for more information. Only one of them is likely to be included in Debian 9.

Note that upstream support for MySQL 5.5 will cease in December 2018 (and Debian security support will have to follow that), while MariaDB 10.0 will receive upstream security support until March 2019.

2.2.6 PHP applications

The Horde Application Framework is available in Jessie, via the `php-horde` package.

2.2.7 Debian Games Blend

The Debian Games Team proudly presents the Debian Games Blend (`http://blends.debian.org/blends/`) consisting of 33 metapackages (`https://tracker.debian.org/pkg/debian-games`) which simplify the installation of games per category. The selection includes among many others strategy, simulation, card, and programming games. Debian Games also offers developers a quick way to install recommended software for developing games in the C++, Java, Perl, or Python 3 programming languages. Content developers will find useful tools for creating game art in `games-content-dev`. The project homepage provides screenshots and further information and offers a compact overview about all games including Debian's finest games (`http://blends.debian.org/games/tasks/finest`).

2.2.8 News from Debian Med Blend

The Debian Med team has again considerably increased not only the number of packages in the fields of biology and medicine but also their quality in terms of testing (at package build time as well as autopkgtest). These enhancements in version 2.0 of the Debian Med Blend metapackages reflect the demand from scientists for reliable software to provide reproducible results. Visit the Debian Med tasks pages (`http://blends.debian.org/med/tasks`) to see the full range of biological and medical software in Debian.

2.2.9 News from Debian Science Blend

Due to the continuous work of the Debian Science team not only new scientific applications were added to the Debian package pool but also new fields of science are covered by certain applications. Visit Debian Science tasks pages (`http://blends.debian.org/science/tasks`) to see the full range of scientific software inside Debian.

2.2.10 News from Debian Geographical Information Systems (GIS) Blend

During the jessie development cycle many changes from UbuntuGIS were merged back into Debian GIS. The collaboration with UbuntuGIS and OSGeo-Live projects was improved, resulting in new packages and contributors. Visit Debian GIS tasks pages (`http://blends.debian.org/gis/tasks`) to see the full range of GIS software inside Debian and the Debian GIS homepage (`https://wiki.debian.org/DebianGis`) for more information.

2.2.11 News from the Debian Java Team

Jessie ships with 799 source packages (442 updated, +130 new ones since Wheezy) which will be maintained by the Java Team (`https://qa.debian.org/developer.php?login=pkg-java-maintainers@lists.alioth.debian.org`). Notable changes:

- OpenJDK 7 is the new supported default Java runtime.
- OpenJDK 8 will be available from jessie-backports.
- Tomcat 7 and Tomcat 8 are supported and Tomcat 6 was removed.
- New developer tools including VisualVM, the Dynamic Code Evolution VM (openjdk-7-jre-dcevm), Gradle, eclipse-wtp-webtools, closure-compiler and more.
- Inclusion of androidsdk-tools (`https://packages.debian.org/source/jessie/androidsdk-tools`)

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 jessie can be found together with the Installation Guide on the Debian website (`https://www.debian.org/releases/jessie/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 (`https://www.debian.org/releases/jessie/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 7, 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 wheezy, please check the release announcements for the jessie beta and RC releases available from the Debian Installer's news history (`https://www.debian.org/devel/debian-installer/News/`).

3.1.1 Major changes

Removed ports Support for the 'ia64' and 'sparc' architectures has been dropped from the installer since they have been removed from the archive.

New ports Support for the 'arm64' and 'ppc64el' architectures has been added to the installer.

New default init system The installation system now installs systemd as the default init system.

Desktop selection The desktop can now be chosen within tasksel during installation. Note that several desktops can be selected at the same time, but some combinations of desktops may not be co-installable.

Replacing "--" by "---" for boot parameters Due to a change on the Linux kernel side, the "---" separator is now used instead of the historical "--" to separate kernel parameters from userland parameters.

New languages Thanks to the huge efforts of translators, Debian can now be installed in 75 languages, including English. This is one more language than in wheezy. 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:

- Tajik has been added to the graphical and text-based 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.

UEFI boot The Jessie installer improves support for a lot of UEFI firmware and also supports installing on 32-bit UEFI firmware with a 64-bit kernel.

Note that this does not include support for UEFI Secure Boot.

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

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

Chapter 4

Upgrades from Debian 7 (wheezy)

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 jessie, 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.

Generally we recommend using the *rescue mode* of the jessie Debian Installer. The advantage of using the installer is that you can choose between its many 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 (`https://www.debian.org/releases/jessie/installmanual`) and the Debian Installer FAQ (`https://wiki.debian.org/DebianInstaller/FAQ`).

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.

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.

4.1.4.2 Debug shell during boot using systemd

If the boot fails under systemd, it is possible to obtain a debug root shell by changing the kernel command line. If the basic boot succeeds, but some services fail to start, it may be useful to add `systemd.unit=rescue.target` to the kernel parameters.

Otherwise, the kernel parameter `systemd.unit=emergency.target` will provide you with a root shell at the earliest possible point. However, this is done before mounting the root file system with read-write permissions. You will have to do that manually with:

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

[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.

More information on debugging a broken boot under systemd can be found in the Diagnosing Boot Problems (`http://freedesktop.org/wiki/Software/systemd/Debugging/`) article.

If everything else fails, you might be able to boot via the old sysvinit system. This requires that `sysvinit` is still installed and the binary `/lib/sysvinit/init` is included in your initramfs. If these requirements are met, add `init=/lib/sysvinit/init` on the kernel command-line and it will boot with the sysvinit binary.

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.

> 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" wheezy 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 7 (wheezy) are not supported. Please follow the instructions in the Release Notes for Debian 7 (`https://www.debian.org/releases/wheezy/releasenotes`) to upgrade to 7 first.

This procedure also assumes your system has been updated to the latest point release of wheezy. 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 *wheezy* and not to *stable* or *jessie*; 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 wheezy 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

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 jessie 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 wheezy 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.

4.3 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. `wheezy`, `jessie`) 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 `https://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/jessie/main/binary-amd64/...
http://mirrors.kernel.org/debian/dists/jessie/contrib/binary-amd64/...
```

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

```
deb http://mirrors.kernel.org/debian jessie 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/jessie/main/binary-amd64/...
/var/ftp/debian/dists/jessie/contrib/binary-amd64/...
```

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

```
deb file:/var/ftp/debian jessie 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] 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.

4.3.3 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 /media/cdrom mount point. For example, if /dev/sr0 is your CD-ROM drive, /etc/fstab should contain a line like:

```
/dev/sr0 /media/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 /media/cdrom    # this will mount the CD to the mount point
# ls -alF /media/cdrom  # this should show the CD's root directory
# umount /media/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 "jessie" or to "stable". There should not be any sources entries pointing to wheezy.

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-jessie-step.time -a ~/upgrade-jessie-step.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-jessie.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-jessie.time ~/upgrade-jessie.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 wheezy 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 wheezy 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
```

Note

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

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 wheezy to jessie.

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 install package`.

4.5 Possible issues during upgrade

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

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 wheezy and jessie sources to your `sources.list` and run **apt-get update**.

4.5.2 Expected removals

The upgrade process to jessie 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. For more information about packages obsoleted in jessie, see Section 4.10.

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" wheezy 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 wheezy and jessie. 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 systemd

The Debian upgrade from wheezy to jessie will by default migrate your init system from the SysV to systemd. Depending on your system and setup, you may need to do some manual changes. We have detailed known issues in Section 5.6.

4.5.7.2 LXC

If you have LXC installed, you may need special care when upgrading your system and your containers. Please have a look at Section 5.8 for known issues and solutions.

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 a kernel metapackage

When you dist-upgrade from wheezy to jessie, it is strongly recommended that you install a linux-image-* metapackage, if you have not done so before. These metapackages will automatically pull in a newer version of the kernel during upgrades. You can verify whether you have one installed by running:

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

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

```
# apt-cache search linux-image- | grep -i meta | 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. However, please have a look at Section 4.7 before performing the first reboot after the upgrade.

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 Changes to root and `/usr` filesystem mounting and checking

`initramfs-tools` will now also run `fsck` on the root filesystem before mounting it. If the chosen init program is `systemd` and there is a separate `/usr` filesystem, it will also fsck and mount `/usr`.

- If `/usr` is a separate filesystem on a RAID device and the `INITRDSTART` setting in `/etc/default/mdadm` is not '`all`', you will need to change it to include that device.
- If `/usr` is a separate filesystem on an LVM logical volume, and the line for `/usr` in `/etc/fstab` specifies the device by `UUID` or `LABEL`, you must change this line to specify the device using the format `/dev/mapper/`*`VG-LV`* or `/dev/`*`VG/LV`*.
- It is no longer possible to bind-mount the `/usr` filesystem.
- If the RTC (real time clock) is set to local time and the local time is ahead of UTC, `e2fsck` will print a warning during boot about the time changing backward (bug #767040 (`https://bugs.debian.org/767040`)). You can disable this by putting the following lines in `/etc/e2fsck.conf`:

```
[options]
broken_system_clock=1
```

4.7 Things to do before rebooting

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

- When upgrading from Wheezy to Jessie, it can be a good idea to purge old packages *before* the first reboot. In particular, obsolete init scripts may cause issues. Please see Section 4.8.1 for details on finding and purging removed packages.

4.8 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 4.10. You should review which configuration files they use and consider purging the packages to remove their configuration files. See also Section 4.8.1.

4.8.1 Purging removed packages

It is generally advisable to purge removed packages. This is especially true if these have been removed in an earlier release upgrade (e.g. from the upgrade to wheezy) or they were provided by third-party vendors. In particular, old init.d scripts have been known to cause issues.

Caution

Purging a package will generally also purge its log files, so you might want to back them up first.

The following command displays a list of all removed packages that may have configuration files left on the system (if any):

```
# dpkg -l | awk '/^rc/ { print $2 }'
```

The packages can be removed by using **apt-get purge**. Assuming you want to purge all of them in one go, you can use the following command:

```
# apt-get purge $(dpkg -l | awk '/^rc/ { print $2 }')
```

If you use `aptitude`, you can also use the following alternative to the commands above:

```
$ aptitude search '~c'
$ aptitude purge '~c'
```

4.9 Deprecated components

With the next release of Debian 9 (codenamed stretch) some features will be deprecated. Users will need to migrate to other alternatives to prevent trouble when updating to 9.

This includes the following features:

- The `hardening-wrapper` package is deprecated and is expected to be removed in Stretch.

4.10 Obsolete packages

Introducing lot of new packages, jessie also retires and omits quite some old packages that were in wheezy. 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 jessie'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 jessie 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 (`https://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 (`https://bugs.debian.org/cgi-bin/pkgreport.cgi?pkg=ftp.debian.org&archive=yes`).

The list of obsolete packages includes:

- `postgresql-9.1`, successor is `postgresql-9.4`. Once the operating system upgrade is finished, you should plan to also upgrade your PostgreSQL 9.1 database clusters to the new PostgreSQL version 9.4 using the **pg_upgradecluster** tool. For users of the PL/perl procedural language, jessie provides an updated `postgresql-plperl-9.1` package linked against jessie's version of libperl in order to enable upgrading to the new perl version in jessie while keeping the old PL/perl database functions usable until the database is upgraded as well.

- `python3.2`, successor is `python3.4`. (Version 2.7 is supported in both wheezy and jessie.)

- `ruby1.8` and `ruby1.9.1`; successor is `ruby2.1`. Please install the package `ruby` to automatically track the current ruby version.

[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.

- `mplayer`; alternatives are `mplayer2`, and `mpv` (new in jessie). Whilst the former is mostly compatible with mplayer in terms of command-line arguments and configuration (and adds a few new features too), the latter adds a lot of new features and improvements, and it is actively maintained upstream.
- `openoffice.org`; please use `libreoffice`.
- `squid`, successor is `squid3`.
- `libjpeg-progs`, successor is `libjpeg-turbo-progs`.
- `openjdk-6-*`, successor is `openjdk-7-*`.

4.10.1 Dummy packages

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

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 Limitations in security support

There are some packages where Debian cannot promise to provide minimal backports for security issues. These are covered in the following subsections.

Note that the package `debian-security-support`, introduced in Jessie, helps to track security support status of installed packages.

5.1.1 Security status of web browsers

Debian 8 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 Jessie, but not covered by security support. These browsers should not be used against untrusted websites.

For general web browser use we recommend Iceweasel or Chromium.

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. Iceweasel and Icedove will also be kept up-to-date by rebuilding the current ESR releases for stable.

5.1.2 Lack of security support for the ecosystem around libv8 and Node.js

The Node.js platform is built on top of `libv8-3.14`, which experiences a high volume of security issues, but there are currently no volunteers within the project or the security team sufficiently interested and willing to spend the large amount of time required to stem those incoming issues.

Unfortunately, this means that `libv8-3.14`, `nodejs`, and the associated node-* package ecosystem should not currently be used with untrusted content, such as unsanitized data from the Internet.

In addition, these packages will not receive any security updates during the lifetime of the Jessie release.

5.1.3 Early termination of MediaWiki security support

Upstream security support for the 1.19 series of `mediawiki` ends during the expected lifecycle of Jessie. The `mediawiki` package is included in Jessie to satisfy dependencies in other packages.

Security support for `mediawiki` will end in conjunction with support for Wheezy in April 2016.

5.2 OpenSSH server defaults to "PermitRootLogin without-password"

In an attempt to harden the default setup, the `openssh-server` configuration will now default to "PermitRootLogin without-password". If you rely on password authentication for the `root` user, you may be affected by this change.

The `openssh-server` will attempt to detect such cases and increase the priority of its debconf prompt.

If you want to keep password authentication for the `root` user, you can also preseed this question by using:

```
$ echo 'openssh-server openssh-server/permit-root-login boolean true' | debconf- ↩
    set-selections
```

5.3 Puppet 2.7 / 3.7 compatibility

If you are using Puppet, please be aware that Puppet 3.7 is not backwards compatible with Puppet 2.7. Among other things, the scoping rules have changed and many deprecated constructs have been removed. See the Puppet 3.x release notes (`https://docs.puppetlabs.com/puppet/3/reference/release_notes.html#puppet-300`) for some of the changes, although be aware that there are further changes in 3.7.

Checking the log files of your current puppetmaster for deprecation warnings and resolving all of those warnings before proceeding with the upgrade will make it much easier to complete the upgrade. Alternatively, or additionally, testing the manifests with a tool like Puppet catalog test (`https://github.com/duritong/puppet_catalog_test`) may also find potential issues prior to the upgrade.

When upgrading a Puppet managed system from Wheezy to Jessie, you must ensure that the corresponding puppetmaster runs at least Puppet version 3.7. If the master is running Wheezy's `puppetmaster`, the managed Jessie system will not be able to connect to it.

For more information on incompatability changes, please have a look at Telly upgrade issues (`https://projects.puppetlabs.com/projects/puppet/wiki/Telly_Upgrade_Issues`) and "The Angry Guide to Puppet 3" (`http://somethingsinistral.net/blog/the-angry-guide-to-puppet-3/`).

5.4 PHP 5.6 upgrade has behavioral changes

The upgrade to Jessie includes an upgrade of PHP from 5.4 to 5.6. This may affect any local PHP scripts and you are advised to check those scripts before upgrading. Below are a selected subset of these issues:

- To prevent man-in-the-middle attacks against encrypted transfers, client streams now verify peer certificates by default.

  As a result of this change, existing code using ssl:// or tls:// stream wrappers (e.g. file_get_contents(), fsockopen(), stream_socket_client()) may no longer connect successfully without manually disabling peer verification via the stream context's "verify_peer" setting.

  For more information about this particular issue, please read this document (`https://wiki.php.net/rfc/tls-peer-verification`).

- PHP changes the handling of case-insensitivity in many cases:

  - All internal case insensitivity handling for class, function, and constant names is done according to ASCII rules. Current locale settings are ignored.
  - The keywords "self", "parent", and "static" are now always case insensitive.
  - The json_decode() function no longer accepts non-lowercase variants of "boolean" values.

- The logo GUID functions (e.g. php_logo_guid()) have been removed.

- It is no longer possible to overwrite keys in static scalar arrays. Please see PHP bug 66015 (`https://bugs.php.net/bug.php?id=66015`) for an example and more information about this particular issue.

- The mcrypt_encrypt(), mcrypt_decrypt() and mcrypt_{MODE}() functions no longer accept keys or IVs with incorrect sizes. Furthermore an IV is now required if the used block cipher mode requires it.
- For legal reasons, the JSON implementation bundled with PHP has been replaced with the version provided by the "jsonc" PECL module. Code that makes assumptions about the finer implementation details of the PHP JSON parser may need to be reviewed.
- The "short_open_tag" setting is now disabled by default. Short tags ("<?" and "?>") are scheduled for removal in PHP7.

For more information or the full list of potential issues, please have a look at upstream's list of backwards incompatible changes for PHP 5.5 (`https://php.net/manual/en/migration55.incompatible.php`) and 5.6 (`https://php.net/manual/en/migration56.incompatible.php`).

5.5 Incompatible changes in Apache HTTPD 2.4

Note

This section only applies to systems which have installed an Apache HTTPD server and configured it manually.

There have been a number of changes to the configuration of the Apache HTTPD server in version 2.4. On the upstream side, the syntax has changed. Notably, the access control directives have changed considerably and will need manual migration to the new directives.

The `mod_access_compat` module is mentioned in the upstream upgrade guide as a possible alternative to immediate migration. However, the reports suggest it may not always work.

The managing of configuration files has also been changed in the Debian packaging. In particular, all configuration files and sites must now end with ".conf" to be parsed by default. This change also replaces the existing use of `/etc/apache2/conf.d/`.

Note

During the upgrade, you may also see warnings about configuration files placed in `/etc/apache2/conf.d/`, which are provided by packages from Debian. This warning is unavoidable but harmless as the affected packages will move their configuration once their upgrade completes (which will generally happen after the Apache HTTPD emits its warning).

For more information and the full list of changes, please refer to:

- Upgrading to 2.4 from 2.2 (`http://httpd.apache.org/docs/2.4/upgrading.html`) document provided by Apache for the upstream side.
- The `/usr/share/doc/apache2/NEWS.Debian.gz` file provided by the `apache2` package.

5.6 Upgrading installs the new default init system for Jessie

Jessie ships with `systemd-sysv` as *default* init system. This package is installed automatically on upgrades.

If you have a preference for another init such as `sysvinit-core` or `upstart`, it is recommended to set up APT pinning prior to the upgrade. This may also be required if you are upgrading LXC containers before the host. In this case, please refer to Section 5.8.1.

As an example, to prevent `systemd-sysv` from being installed during the upgrade, you can create a file called `/etc/apt/preferences.d/local-pin-init` with the following contents:

```
Package: systemd-sysv
Pin: release o=Debian
Pin-Priority: -1
```

Caution

Be advised that some packages may have degraded behavior or may be lacking features under a non-default init system.

Please note that the upgrade may install packages containing "systemd" in their name even with APT pinning. These alone do *not* change your init system. To use systemd as your init system, the `systemd-sysv` package must be installed first.

If APT or aptitude has issues computing an upgrade path with the pin in place, you may be able to help it by manually installing both `sysvinit-core` and `systemd-shim`.

5.6.1 Stricter handling of failing mounts during boot under systemd

The new default init system, `systemd-sysv`, has a stricter handling of failing "auto" mounts during boot compared to sysvinit. If it fails to mount an "auto" mount (without the "nofail" option), systemd will drop to an emergency shell rather than continuing the boot.

We recommend that all removable or "optional" mount points (e.g. non-critical network drives) listed in `/etc/fstab` either have the "noauto" or the "nofail" option.

5.6.2 Obsolete init-scripts should be purged

If you are upgrading from previous releases, your system may contain obsolete init-scripts provided by (now) removed packages. These scripts may have inaccurate or no dependency metadata, which can lead to dependency cycles in your init configuration.

To avoid this, we recommend that you go and review the list of packages that are in the "rc" ("Removed, but Config-files remain") state, and purge at least all those containing init-scripts.

Please see Section 4.8.1 for details on finding and purging removed packages.

5.6.3 Locally modified init-scripts may need to be ported to systemd

Note

This section only applies to systems where Debian-provided init scripts have been modified locally.

If you have modified some of the init scripts provided by Debian, please be aware that these may now have been superseded by a systemd unit file or by systemd itself. If you have debsums installed, you can check for locally modified init scripts by using the following shell command.

```
debsums -c -e | grep ^/etc/init.d
```

Alternatively, the following can be used in the absence of debsums.

```
dpkg-query --show -f'${Conffiles}' | sed 's, /,\n/,g' | \
  grep /etc/init.d | awk 'NF,OFS="  " {print $2, $1}' | \
  md5sum --quiet -c
```

If either command flags any files and their corresponding packages *or* the `systemd` now provides an systemd unit file for that service, the systemd unit file will take precedence to your locally modified init script. Depending on the nature of the change, there are different way to perform the migration.

If necessary, it is possible to override the systemd unit file to have it start the sysvinit script. For more information on systemd unit files, please have a look at the following resources.

- How Do I Convert A SysV Init Script Into A systemd Service File? (`http://0pointer.de/blog/projects/systemd-for-admins-3.html`)
- systemd.special — Special systemd units (`http://0pointer.de/public/systemd-man/systemd.special.html`)
- My Service Can't Get Realtime! (`http://www.freedesktop.org/wiki/Software/systemd/MyServiceCantGetRealtime/`) (also contains a very short mention on invoking init scripts from unit files)

5.6.4 Plymouth needed for boot-prompts under systemd boots

If your boot is interactive (e.g. needs a password for an encrypted disk), please ensure that you have `plymouth` installed *and configured*. Please refer to `/usr/share/doc/plymouth/README.Debian` for information on how to configure plymouth.

Without `plymouth`, you may find that your boot prompt disappears. Reports suggest that the crypt-setup prompt still accepts input despite not being visible. Should you experience this issue, typing the correct password may still work.

5.6.5 Interaction between logind and acpid

ACPI events can be handled by logind or acpid. In case both services are configured to handle events in different ways, this can lead to undesired results.

We recommend to migrate any non-default settings to logind and uninstall acpid. Alternatively it is also possible to configure logind to ignore ACPI events by adding:

```
HandlePowerKey=ignore
HandleSuspendKey=ignore
HandleHibernateKey=ignore
HandleLidSwitch=ignore
```

to `/etc/systemd/logind.conf`. Note that this might change behaviour of desktop environments relying on logind.

5.6.6 Unsupported crypttab features under systemd (e.g. "keyscript=...")

There are some cryptsetup features that are unfortunately not supported when running with systemd as the init system. These are:

- precheck
- check
- checkargs
- noearly
- loud
- keyscript

If your system relies on any of these for successful booting, you will have to use sysvinit (`sysvinit-core`) as init system. Please refer to Section 5.6 for how to avoid a particular init system.

You can check if any of these options are in use on your system by running the following command:

```
grep -e precheck -e check -e checkargs -e noearly -e loud -e keyscript /etc/ ↩
    crypttab
```

If there is no output from the above, your system does not use any of the affected options.

5.6.7 systemd: issues SIGKILL too early

A regression was reported in systemd after the Jessie release. The bug occurs during shutdown or reboot, where systemd does not give any reasonable delay before issuing SIGKILL to processes. This can lead to data loss in processes that have not saved all data at the time of the reboot (e.g. running databases).

This issue is tracked in the Debian bug #784720 (`https://bugs.debian.org/784720`)

5.7 Required kernel config options for Jessie

Note

This section is only for people who compile their own kernel. If you use the kernels compiled by Debian, you can disregard this section.

The following kernel configuration options are now either required or recommended for Jessie (in addition to existing ones from previous releases):

```
# Required for udev
CONFIG_DEVTMPFS=y
# Required for *some* systemd services
CONFIG_DEVPTS_MULTIPLE_INSTANCES=y
# Required by "bluez" (GNOME)
CONFIG_BT=y
# Required for cups + systemd.
CONFIG_PPDEV=y
```

The systemd services which require CONFIG_DEVPTS_MULTIPLE_INSTANCES=y will typically contain at least one of the following directives:

```
PrivateTmp=yes
PrivateDevices=yes
PrivateNetwork=yes
ProtectSystem=yes
```

If you do not use systemd, or can assert that none of the systemd services will use the above directives, the config option might not be required for your particular system.

For more information about the requirements, please refer to the section called "REQUIREMENTS" in the README (`https://sources.debian.net/src/systemd/jessie/README/`) file for the package `systemd`.

5.8 Upgrade considerations for LXC hosts and containers

Note

This section only applies to systems that have LXC containers and hosts. Normal end user systems usually do not have these.

The upgrade from Wheezy to Jessie will migrate your system to the systemd init system by default (see Section 5.6).

When upgrading an LXC container or an LXC virtual machine, this will have different consequences depending on whether the *host system* has already been upgraded to Jessie or not.

5.8.1 Upgrading LXC guests running on Wheezy hosts

If you are upgrading an LXC guest container that is running on a *Wheezy host* system, then you will need to prevent the guest from being automatically migrated to systemd. You prevent the migration via pinning, as described in Section 5.6.

This is required as the Wheezy host lacks functionality to boot a system running systemd.

You should be able to switch over to systemd inside the LXC guest once you have upgraded the *host* system to Jessie. See the next paragraph for things that need to be adapted on Jessie hosts.

5.8.2 Upgrading LXC guests running on Jessie hosts

In order to be able to boot LXC guests with systemd, you need to adapt your LXC container configuration. The container configuration can usually be found in `/var/lib/lxc/CONTAINER_NAME/config` You need to add the following two settings to the configuration:

```
lxc.autodev = 1
lxc.kmsg = 0
```

5.8.3 Further information

You can find further information on LXC in Debian in the Debian wiki (`https://wiki.debian.org/LXC`).

5.9 Manual migration of disks encrypted with LUKS whirlpool (non-standard setups)

Note

This section is only for people who have set up LUKS encrypted disks themselves using the whirlpool hash. The debian-installer has *never* supported creating such disks.

If you have *manually* set up an encrypted disk with LUKS whirlpool, you will need to migrate it manually to a stronger hash. You can check if your disk is using whirlpool by using the following command:

```
# /sbin/cryptsetup luksDump <disk-device> | grep -i whirlpool
```

For more information on migrating, please see item "8.3 Gcrypt 1.6.x and later break Whirlpool" of the cryptsetup FAQ (`https://code.google.com/p/cryptsetup/wiki/FrequentlyAskedQuestions`).

Caution

If you have such a disk, `cryptsetup` will refuse to decrypt it by default. If your rootdisk or other system disks (e.g. /usr) are encrypted with whirlpool, you should migrate them prior to the first reboot after upgrading `cryptsetup`.

5.10 The GNOME desktop requires basic 3D graphics

The GNOME 3.14 desktop in Jessie no longer has fallback support for machines without basic 3D graphics. To run properly, it needs either a recent enough PC (any PC built in the last 10 years should have the required SSE2 support) or, for architectures other than i386 and amd64, a 3D-accelerated graphics adapter with EGL drivers.

5.11 The GNOME desktop does not work with the AMD proprietary FGLRX driver

Unlike other OpenGL drivers, the AMD FGLRX driver for Radeon adapters does not support the EGL interface. As such, several GNOME applications, including the core of the GNOME desktop, will not start at all when this driver is in use.

It is recommended to use the free `radeon` driver, which is the default in jessie, instead.

5.12 Changes in the GNOME default keyboard shortcuts

The default keyboard shortcuts in the GNOME desktop have changed in order to match more closely those of some other operating systems.

Shortcut settings previously modified by the user will be preserved upon upgrade. These settings can still be configured from the GNOME control center, accessible from the top right menu by clicking on the "settings" icon.

5.13 Changes to default shell of system users provided by `base-passwd`

The upgrade of the `base-passwd` package will reset the shell of some system users to the "nologin" shell. This includes the following users:

- daemon
- bin
- sys
- sync
- games
- man
- lp
- mail
- news
- uucp
- proxy
- www-data
- backup
- list
- irc
- gnats
- nobody

If your local setup requires that any of these users have a shell, you should say no to migrating, or migrate and then change the shell of the corresponding users. Notable examples include local backups done via the "backup" user with "ssh-key" authentication.

Caution

The migration will happen automatically if your debconf question priority is "high" or above.

If you know you want to keep the current shell of a given user, you can preseed the questions by using the following:

```
echo 'base-passwd base-passwd/system/username/shell/current-shell-mangled/ ↩
    _usr_sbin_nologin boolean false' | debconf-set-selections
```

Where *username* is the name of the user in question and *current-shell-mangled* is the mangled name of the shell. The mangling is done by replacing all characters other than alphanumerics, dashes, and underscores with underscores. E.g. /bin/bash becomes _bin_bash.

5.14 Migration to new KDE E-mail, Calendar, and Contacts (Kontact)

The Kontact Personal Information Management system has received a major upgrade. The new version makes much greater use of metadata indexing and each user's data must be migrated into these new indices.

E-mail, calendar events, and addressbook contacts are automatically migrated when the user logs in and the relevant component is started. Some advanced settings such as e-mail filters and custom templates require manual intervention. Further details and troubleshooting suggestions are collected on the Debian Wiki (`https://wiki.debian.org/KDE/Jessie/kontact`).

5.15 Missing virtual consoles ("getty"s) with multiple desktop environments

Note

This issue is currently reported as fixed in Jessie. Should you stil be able to reproduce it, then please follow up to Debian Bug#766462 (`https://bugs.debian.org/766462`). Note that you may have to unarchive the issue first (please refer to the Debian BTS control server (`https://www.debian.org/Bugs/server-control`) documentation on how to unarchive bugs).

If you have multiple desktop environments installed, you may experience that none of the "virtual consoles" show a login prompt.

This issue seems to occur when `plymouth`, `systemd`, and GNOME are all installed. This issue is reported as Debian Bug#766462 (`https://bugs.debian.org/766462`).

It has been reported that removing the "splash" argument from the kernel command-line may work around the issue. Please see `/etc/default/grub` and remember to run `update-grub` after updating the file.

5.16 "VGA signal out of range" / blank screen during boot with grub-pc

There is a compatibility issue in `grub-pc` with older graphics cards (e.g. the "ATI Rage 128 Pro Ultra TR") that can cause it to show a blank screen during boot. The display may issue a "VGA signal out of range" message (or something similar).

A simple work around is to set `GRUB_TERMINAL=console` in `/etc/default/grub`.

5.17 Stricter validation of cron files in crontab

The `crontab` program is now more strict and may refuse to save a changed cron file if it is invalid. If you experience issues with `crontab -e`, please review your crontab for existing mistakes.

5.18 Change in handling of unreadable module paths by `perl`

From version 5.18 (and 5.20, which is included in Jessie), Perl will exit with a fatal error if it encounters unreadable module paths in `@INC`. The previous behavior was to skip such entries. It is recommended to check the contents of `@INC` in your environment for directories which are not world-readable, and take appropriate action.

You can see the default `@INC` for Perl by running **perl -V**.

5.19 Upgrade considerations for Ganeti clusters

5.19.1 Problem upgrading Ganeti clusters with DRBD-backed instances

The version of `ganeti` (2.12.0-3) released with Jessie does not support migrations from installations running 2.5 or earlier (including Wheezy) in cases where there are instances with DRBD disks. It is hoped that this issue will be fixed in a point release, and recommended that you do not upgrade affected Ganeti clusters in the meantime. You can find more information about this issue at Debian Bug#783186 (`https://bugs.debian.org/783186`).

5.19.2 General notes on upgrading Ganeti clusters

The recommended procedure to upgrade a Ganeti cluster from Wheezy's `ganeti` version (2.5.2-1) to Jessie's (2.12.0-3) is to stop all instances and then upgrade and reboot all nodes at once. This will ensure that all instances run with Jessie's hypervisor version and that all nodes run the same versions of Ganeti and DRBD.

Note that running a cluster with mixed 2.5 and 2.12 nodes is not supported. Also note that, depending on the hypervisor, instance live migrations may not work between Wheezy and Jessie hypervisor versions.

5.20 New requirements for file execution in Samba4

If a client requests that a file should be "opened for execution", Samba4 will require the executable bit to be set on the file in addition to the regular read permissions. This also causes "netlogon" scripts to be silently ignored if they lack this executable bit.

5.21 Cryptsetup can break boot with BUSYBOX=n

Note

This section only applies to people that have manually changed their `/etc/initramfs-tools/initramfs.conf` to not use busybox.

If you have *both* `busybox` and `cryptsetup` installed plus configured initramfs to *not* use busybox, then it may render your system unbootable.

Please check the value of your BUSYBOX setting in `/etc/initramfs-tools/initramfs.conf` if you have both of these packages installed. At this time, known work arounds are uninstalling `busybox` or setting `BUSYBOX=y` in `/etc/initramfs-tools/initramfs.conf`.

Warning

If you had to make any changes, please remember to run `update-initramfs -u` to update your initramfs. Otherwise, you may still end up with a broken boot.

Please see Debian Bug#783297 (`https://bugs.debian.org/783297`) for more information.

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 (`https://www.debian.org/doc/`) and the Debian Wiki website (`https://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 `https://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 (`https://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 `https://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 e-mail 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 e-mail. 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 (`https://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 (`https://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 (`https://lists.debian.org/`) is also extremely helpful. To maintain Debian's high quality distribution, submit bugs (`https://bugs.debian.org/`) and help developers track them down and fix them. The tool `how-can-i-help` helps you to find suitable reported bugs to work on. If you have a way with words then you may want to contribute more actively by helping to write documentation (`https://www.debian.org/doc/cvs`) or translate (`https://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 (`https://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 (`https://www.debian.org/devel/#projects`) which include ports to particular architectures and Debian Pure Blends (`https://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

UEFI
: Unified Extensible Firmware Interface

USB
: Universal Serial Bus

UUID
: Universally Unique Identifier

VGA
: Video Graphics Array

WPA
: Wi-Fi Protected Access

Appendix A

Managing your wheezy system before the upgrade

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

A.1 Upgrading your wheezy system

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

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

If you have any lines starting with `deb file:`, you will have to check for yourself if the location they refer to contains an wheezy or a jessie 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 jessie, it is recommended to remove old configuration files (such as `*.dpkg-{new,old}` files under `/etc`) 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[1] 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] 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, Joost van Baal-Ilić, Josip Rodin, Julien Cristau, Justin B Rye, LaMont Jones, Luk Claes, Martin Michlmayr, Michael Biebl, Moritz Mühlenhoff, Niels Thykier, 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