Release Notes for Debian GNU/Linux 3.1 (‘sarge’), SPARC

Josip Rodin, Bob Hilliard, Adam Di Carlo, Anne Bezemer, Rob Bradford (current), Frans Pop (current)
<debian-doc@lists.debian.org>

$Id: release-notes.en.sgml,v 1.71 2006/09/18 13:21:10 fjp Exp $

Contents

Chapter 1

What's new in the Release Notes

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

Please note that starting with Debian GNU/Linux 3.1, we only support and document upgrading from the previous release of Debian (in this case, the upgrade from woody). If you need to upgrade from older releases, we suggest you read previous editions of the release notes.

1.1 Changes in the Release Notes

This section lists changes in the Release Notes since the original version that was published with Debian GNU/Linux 3.1r0. Minor textual corrections are omitted.

- Improved description on module loading for ALSA in 'Sound configuration' on page 22.
- Document upgrading from raidtools2 to mdadm in 'Upgrading from raidtools2 to mdadm' on page 18. This section may also be relevant while upgrading the kernel as part of the upgrade.
- `aptitude` uses a different method for registering packages that are on hold than `apt-get` and `dselect`. Properly document how hold status can be checked and set in 'Upgrades from previous releases' on page 9.

Chapter 2

What's new in Debian GNU/Linux 3.1

The list of supported architectures has not changed since the previous release, Debian GNU/Linux 3.0 ('woody'). Here is the full list of architectures for this release.

- Intel x86 ('i386')
- Motorola 680x0 ('m68k')
- Alpha ('alpha')
- SPARC ('sparc')
- PowerPC ('powerpc')
- ARM ('arm')
- MIPS ('mips' (Big endian) and 'mipsel' (Little endian))
- Intel Itanium ('ia64')
- HP PA-RISC ('hppa')
- S/390 ('s390')

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

Debian GNU/Linux 3.1 for the SPARC architecture ships with kernel version 2.4.27.

On the SPARC architecture a 2.6 kernel is also available; this has kernel version 2.6.8. Note that Debian's 2.6.8 kernel packages include the 2.6.8.1 kernel release and selected other patches.

2.1 What's new in the distribution?

This new release of Debian again comes with a lot more software than its predecessor woody; the distribution includes over 9000 new packages. Most of the software in the distribution has been updated: almost 6500 software packages (that is 73% of the number of packages in woody). Also, a significant number of packages has 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.

This release of Debian GNU/Linux contains the much improved XFree86 4.3 release, which includes support for a greater range of hardware, better autodetection support, and improved support for advanced technologies such as Xinerama and 3D acceleration.

Debian GNU/Linux is more desktop orientated than ever in this new release, it now includes GNOME 2.8 and KDE 3.3. Also included for the first time is a complete office suite in the form of OpenOffice.org 1.1, other productivity tools included in the release are the Evolution groupware software and GAIM instant messaging client.

The sarge version of `aptitude` is the preferred program for package management from console. It has proven to be better at dependency resolution than `apt-get`. `aptitude` supports most command line operations of `apt-get`. If you are still using `dselect`, you should also give `aptitude` a try as frontend for package management.

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

2.1.1 New service debian-volatile

There is a new service *debian-volatile* allowing users to easily update stable packages that contain information that quickly goes out of date. Examples are a virus scanner's signatures list or a spam filter's pattern set. An administrator can use the "volatile.debian.net" archive with similar ease to the "security.debian.org" archive, and enjoy the use of packages with up-to-date information without the hassle and risks of maintaining an entire (or partial) system based on bleeding-edge packages. For more information and a list of mirrors, please see the archive's web page (`http://volatile.debian.net/`).

Note that debian-volatile is *not* an official Debian service. Use it at your own discretion.

2.1.2 non-US obsoleted

For the sarge release, packages that were formerly in the non-US part of the archive have been moved into the regular archive. If you have any lines referring to "non-us" in your `/etc/apt/sources.list`, you should remove them.

2.2 What's new in the installation system?

The old Debian GNU/Linux installation system has been replaced by a completely new installation system called `debian-installer`. The new installation system is modular in design and so has been developed with extensibility in mind. It has been fully translated into almost forty languages; additional translations are in progress and may be added in point releases for sarge.

Some of the new features in the installation system include improved hardware detection, support for booting off USB flash devices, the use of `aptitude` to install packages during configuration of the base system, and support for the `XFS` file system, `RAID` and `LVM` (logical volume management).

For full details on the new Debian installation system, users are advised to read the Debian Installation Guide included on the first CD or available from the release pages (`http://www.debian.org/releases/stable/installmanual`). The Installation Guide has been fully translated into eight languages and more are being worked on. Additional translations will be made available from the website when completed.

Chapter 3

New installations

The old Debian GNU/Linux installation system called `boot-floppies` has been replaced by a new componentized and more powerful installation system called `debian-installer`.

The installer offers a variety of installation methods. Which methods are available to install your system depends on your architecture.

If you are making a new installation of Debian, you should read the Installation Guide, which is available on the Official CD at:

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

or on the Internet from the sarge release pages (http://www.debian.org/releases/stable/installmanual). You may also want to check the errata (http://www.debian.org/releases/stable/debian-installer/index#errata) for debian-installer.

The installation system uses a 2.4 series kernel by default. Installation using a 2.6 based kernel is also possible for SPARC. For more details on how to use this please consult the Installation Guide.

3.1 Issues with keyboards on SPARC

There are several issues with keyboard selection during installation.

The first issue is with USB keyboards by Sun as used on for example SunBlade systems. When installing using the default 2.4 kernel, these are incorrectly "recognized" by the installer as regular Sun keyboards. A workaround is documented in the Installation Guide (see link above, chapter "Using the Debian Installer").

The second issue is kernel related. Kernels in the 2.6 series use a different input layer that makes all keyboards look like "normal" PC keyboards. This means that if you boot the installer with a 2.4 kernel and configure it for a Sun or USB keyboard and later (in expert mode) select a 2.6 kernel for the new system, you will very likely end up with a non-working keyboard after reboot.

3.2 Issues with framebuffer on SPARC

Because of display problems on some systems, framebuffer support is disabled by default for SPARC. This can result in ugly display on systems that do properly support the framebuffer, like those with ATI graphical cards. If you see display problems in the installer, you can try booting the installer with parameter `debian-installer/framebuffer=true`.

3.3 Popularity contest

For technical reasons the `popularity-contest` package is no longer installed by default for new sarge installations. This will probably be corrected in future releases.

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

Information from `popularity-contest` is processed anonymously. We would appreciate it if you install the package and allow it to participate in the official survey; you will thereby help improve Debian.

Chapter 4

Upgrades from previous releases

4.1 Preparing for the upgrade

Before upgrading your system, it is strongly recommended that you make a full backup, or at least backup 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` and the output of `dpkg --get-selections "*"` (the quotes are important).

The upgrade process in itself does not modify anything in the `/home` directory. However, some applications (e.g. Mozilla, some KDE applications) 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 issue.

It's wise to inform all users in advance of any upgrades you're planning, although users accessing your system via SSH (at least) shouldn't notice much during the upgrade, and may want to continue working. If you wish to take extra precautions, back up or unmount user's partitions (`/home`) before upgrading. A reboot will not normally be necessary, unless you plan to also upgrade your kernel.

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! 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.

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

4.2 Checking system status

The upgrade process described in this chapter has been designed for upgrades from "pure" woody systems. It assumes your system has been updated to the latest point release of woody. If you have not or are unsure, follow the instructions in 'Upgrading your woody system' on page 31.

It also assumes you have the woody version of `aptitude` installed. You can check if it is installed using

```
$ dpkg -l aptitude
```

If the line of output does *not* begin with "i", you should install it before you start the upgrade using the instructions in 'Installing woody version of aptitude' on page 31.

4.2.1 Disabling APT pinning

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

4.2.2 Checking packages status

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

```
# dpkg --audit
```

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

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

or

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

It is desirable to remove any holds before upgrading. If any package that is essential for the upgrade is on hold, the upgrade will fail. Note that `aptitude` uses a different method for registering packages that are on hold than `apt-get` and `dselect`. You can identify packages on hold for `aptitude` with

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

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

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

If you changed and recompiled a package locally, and didn't rename it or put an epoch in the version, you must put it on hold to prevent it from being upgraded. The "hold" package state for `aptitude` can be changed using (replace `hold` with `unhold` to unset the "hold" state):

```
# aptitude hold package_name
```

If there is anything you need to fix, it is best to make sure your `sources.list` still refers to woody as explained in 'Checking your sources list' on page 31.

4.2.3 Unofficial sources and backports

If you have any non-Debian packages on your system, you should be aware that these may be removed during the upgrade because of conflicting dependencies. If these packages were installed by adding an extra package archive in your `/etc/apt/sources.list`, you should check if that archive also offers packages compiled for sarge 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 woody system. Such packages are most likely to cause problems during an upgrade as they may result in file conflicts[1]. Section 'Possible issues during upgrade' on page 16 has some information on how to deal with file conflicts if they should occur.

4.3 Checking kernel support

All machines with a 64bit SPARC CPU (sun4u) should be upgradable without any special considerations about kernel support.

sun4c CPUs are *no longer supported* in sarge. The support for sun4d CPUs is in a rather unknown state since they are very rare. It is possible that sun4d CPUs with an MMU work.

sun4m CPUs are still supported but you need to install a newer kernel version first before upgrading the system. This is necessary because newer versions of glibc use assembler instructions not available on certain machines, so you need a updated kernel first that emulates the missing instructions.

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

Technically only *some* sun4m chips are affected, but as glibc can't reliably detect whether a system is affected it will refuse to be upgraded on any 32bit SPARC system before a fixed kernel is installed.

For those interested in the gory details: some of the sun4m chips, produced by Cypress/ROSS, don't implement the `umul` instruction (RT601/CY7C601, same chip, only different names). They were used in the early SPARCserver 6xxMP models. Later models used chips manufactured by TI. Currently we don't know if these are also affected.

4.3.1 Upgrading the kernel

If (and only if) the previous section indicates that you should upgrade your kernel *before* the upgrade of the system, you should do so now.

Backports are available of all tools needed to install the current kernel from sarge. Detailed instructions on how to install the new kernel can be found in 'Upgrading the kernel' on page 29.

4.4 Preparing sources for APT

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

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

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

4.4.1 Adding APT Internet sources

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

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

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

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

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

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

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 before attempting to start the installation. With a reasonably extended Debian installation, you can expect at least 300 MB of downloaded data.

4.4.2 Adding APT sources for a local mirror

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

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

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

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

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

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

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

4.4.3 Adding APT source from CD-ROM or DVD

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

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

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

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

To verify it works, insert a CD and try running

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

Next, run:

```
# apt-cdrom add
```

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

4.5 Upgrading packages

The recommended tool for upgrading between Debian GNU/Linux releases is to use the package management tool `aptitude`. This tool makes safer decisions about package installations than running `apt-get` directly.

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

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

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

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 -a ~/upgrade-to-sarge.typescript
```

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

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

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

4.5.1 Updating the package list

First the list of available packages for the new release needs to be fetched. This is done by executing[2]:

```
# apt-get update
```

4.5.2 Upgrading aptitude

Upgrade tests have shown that sarge's version of `aptitude` is better at solving the complex dependencies during an upgrade than either `apt-get` or woody's `aptitude`. It should therefore be upgraded first using:

```
# aptitude install aptitude
```

You will be shown a list of the changes that will be made and asked you to confirm them. You should take a careful look at the proposed changes, especially packages that will be removed by the upgrade, before you confirm.

In some cases if a large number of packages is listed for removal, you may be able to reduce this list by "pre-upgrading" selected other packages alongside `aptitude`. An example may clarify this. During upgrade tests for systems having KDE installed, we have seen that this step would cause removal of a large number of KDE packages and/or perl. The solution proved to be to `install aptitude perl` instead of `install aptitude`.

4.5.3 Upgrading doc-base

If you have `doc-base` installed, it must be upgraded before the rest of the system too. Reason is that it may fail if `perl` is upgraded at the same time. You can find out if it is installed using:

```
# dpkg -l doc-base
```

If the line of output begins with "i" then it is installed and must be upgraded before continuing.

```
# aptitude install doc-base
```

[2] We use `apt-get` for this because the woody version `aptitude` may fail when new sources have been added to `sources.list`.

4.5.4 Upgrading the rest of the system

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

```
# aptitude -f --with-recommends dist-upgrade
```

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

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

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

The `--fix-broken` (or just `-f`) option causes `apt` to attempt to correct a system with broken dependencies in place. `apt` does not allow broken package dependencies to exist on a system.

4.5.5 Possible issues during upgrade

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

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

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

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

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

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

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

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

to eliminate some of the offending packages, or

```
# aptitude --fix-broken install
# dpkg --configure --pending
```

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

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

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

```
Unpacking replacement <package-foo> ...
dpkg: error processing <package-name-for-foo> (--unpack):
 trying to overwrite '<some-file-name>',
 which is also in package <package-bar>
```

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

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

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

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

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

4.6 Things to do before rebooting

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

Read `/usr/share/doc/xfree86-common/README.Debian-upgrade.gz` for more info on the upgrade of the X window system packages. This is relevant for users of all previous Debian releases. In short, you need to read it.

4.6.1 Upgrading your kernel

Note that the Linux kernel was *not* upgraded by these procedures. You may wish to do so yourself, either by installing one of the `kernel-image-*` packages or by compiling a customized kernel from sources.

If you are currently using a kernel from the 2.4 series, the older stable Linux kernel series, you may wish to upgrade to a 2.6 series kernel for better hardware support or improved performance.

However, you are strongly advised **not** to upgrade to a 2.6 kernel as part of the upgrade from woody to sarge. Some issues associated with an upgrade to 2.6 are documented in 'Upgrading to a 2.6 kernel' on page 21.

To upgrade your kernel you must first choose the kernel most appropriate for your subarchitecture. A list of kernels available for you to install can be found with:

```
# apt-cache search ^kernel-image
```

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

Please note that the installation system of woody (and previous releases) did *not* install the kernel as a package in your system. This has changed in sarge and you can install virtual packages to keep track of kernel changes. These packages are named kernel-image-`VERSION-ARCH`, with `VERSION` corresponding to the kernel version number (2.4 or 2.6) and `ARCH` corresponding to any of the supported architectures. If you want to have security support for the kernel integrated in package management please install the kernel package most suitable for your hardware after the upgrade.

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

4.6.2 Upgrading from raidtools2 to mdadm

`raidtools2` is no longer maintained by its upstream developers and has been replaced by the `mdadm` package. `mdadm` is a single program that can perform almost any RAID management task without a configuration file; by default it does not use one.

The remainder of this section gives some upgrade hints for users of `raidtools2`.

If your RAID array was created on a 2.2 Linux kernel patched with RAID support, the superblock was created incorrectly, or at least in a way that is incompatible with 2.4 and later kernels. In order to fix this problem, you have to execute the following two commands:

```
# mdadm --examine --sparc2.2
# mdadm --assemble --update=sparc2.2
```

As mentioned above, in many cases `mdadm` can work without configuration file. If you use a kernel that automatically configures the RAID array for you, can skip this paragraph — you merely have to install the package `mdadm` and the RAID will be detected during the boot process. The standard kernels in Debian have support for the configuration of RAID arrays on boot. You also need to make sure that the partitions are set to type "Linux raid autodetect" (id `fd`). The following command will list the current type of partitions:

```
# fdisk -l disk_device
```

If you have a mixed setup with some RAID arrays that are auto-configured and some that are not, you have to create a configuration file.

To migrate from the configuration file `/etc/raidtab` (`raidtools2`) to `/etc/mdadm/mdadm.conf` (`mdadm`), please execute:

```
# echo 'DEVICE /dev/hd*[0-9] /dev/sd*[0-9]' > /etc/mdadm/mdadm.conf
# mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

These commands will generate a configuration file with the existing arrays on the system.

You should also make sure that the RAID arrays are started automatically on boot. Check the file `/etc/default/mdadm` to see if the variable AUTOSTART is set to `true`.

4.7 Obsolete packages

Introducing several thousand new packages, sarge also retires and omits more than two thousand old packages that were in woody. 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 sarge's release[3], 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 sarge due to bugs in them. In the later case, packages might still be present in the "unstable" distribution.

Detecting which packages in an updated system are "obsolete" is easy since the package management front-ends will mark them as such. If you are using `aptitude`, you will see a listing of these packages in the "Obsolete and Locally Created Packages" entry. `dselect` provides a similar section but the listing it presents might differ. Also, if you have used `aptitude`

[3] 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.

to manually install packages in woody it will have kept track of those packages you manually installed and will be able to mark as obsolete those packages pulled in by dependencies alone which are no longer needed if a package has been removed. Also, `aptitude`, unlike `deborphan` will not mark as obsolete packages that you manually installed, as opposed to those that were automatically installed through dependencies.

There are additional tools you can use to find obsolete packages such as `deborphan`, `debfoster` or `cruft`. `deborphan` is highly recommended, although it will (in default mode) only report obsolete libraries: packages in the "libs" or "oldlibs" sections that are not used by any other packages. Do not blindly remove the packages these tools present, especially if you are using aggressive non-default options that are prone to produce false positives. It is highly recommended that you manually review the packages suggested for removal (i.e. their contents, size and description) before you remove them.

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

4.7.1 Dummy packages

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

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

Chapter 5

Issues to be aware of for sarge

5.1 Changes to Python packages

None of the python2.X packages that are included with sarge include the standard modules 'profile' and 'pstats', because they are licensed under a license that does not conform to the DFSG (see bug #293932 for details). These two modules can be found in the python-profiler and python2.X-profiler packages that are included in the non-free section of the Debian archive.

5.2 Upgrading to a 2.6 kernel

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

You are therefore strongly advised not to upgrade to a 2.6 kernel as part of the upgrade from woody to sarge. Instead, you should first make sure your system works correctly with either the old kernel or with a 2.4 kernel from sarge and do the upgrade to a 2.6 kernel later as a separate project.

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

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

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

Once you have installed your 2.6 kernel, but before you reboot, make sure you have a recovery method. First, make sure that the bootloader configuration has entries for both the new kernel and the old, working 2.4 kernel. You should also ensure you have a "rescue" floppy or cdrom to hand, in case misconfiguration of the bootloader prevents you booting the old kernel.

5.2.1 Keyboard configuration

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

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

If your console keyboard is affected, you will probably also need to reconfigure your keyboard for the X Window System. You can do this either by running `dpkg-reconfigure xserver-xfree86` or by editing `/etc/X11/XF86Config-4` directly. Don't forget to read the documentation referred to in 'Things to do before rebooting' on page 17.

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

5.2.2 Mouse configuration

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

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

5.2.3 Sound configuration

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

5.2.4 Switching to 2.6 may activate udev

`udev` is a userspace implementation of devfs. It is mounted over the `/dev` directory and will populate that directory with devices supported by the kernel. It will also dynamically add and remove devices as kernel modules are loaded or unloaded respectively, working together with `hotplug` to detect new devices. `udev` works only with 2.6 kernels.

As `udev` is automatically installed as a dependency of e.g. `gnome`, there is a chance that upgrading to a 2.6 kernel will result in `udev` being activated.

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

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

Chapter 6

More information on Debian GNU/Linux

6.1 Further reading

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

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

6.2 Getting help

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

6.2.1 Mailing lists

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

6.2.2 Internet Relay Chat

Debian has an IRC channel dedicated to the support and aid of Debian users located on the Freenode IRC network which exists to provide interactive services to peer-directed project communities. To access the channel point your favourite IRC client at irc.debian.org and join #debian.

Please follow the channel guidelines, respecting other users fully. For more information on Freenode please visit the website (`http://freenode.net/`).

6.3 Reporting bugs

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

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

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

6.4 Contributing to Debian

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

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

include ports to particular architectures, Debian Jr. (http://www.debian.org/devel/debian-jr/) and Debian Med (http://www.debian.org/devel/debian-med/).

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

Appendix A

Upgrading the kernel

The information in this appendix is relevant only if, for a successful upgrade of the system, you need to upgrade the kernel *before* upgrading the system. Please read 'Checking kernel support' on page 11 to find out if that is required for your system.

The following instructions explain step by step how to use the available backported tools to install the newer kernel.

Because packages may need to be installed from woody, you should first check that entries in your `sources.list` still refer to woody as explained in 'Checking your sources list' on page 31.

Download and install the needed packages *with apt*: to install the packages with `apt` or one of its frontends, add the following line in your `/etc/apt/sources.list`:

```
deb http://ftp.debian.org/debian/dists/sarge/main/upgrade-kernel ./
# sources are also available if you need them
# deb-src http://ftp.debian.org/debian/dists/sarge/main/upgrade-kernel
```

Then install the packages `modutils` and `initrd-tools`. (Afterwards you can safely drop the additional entry again.)

After that change your `sources.list` file to point to sarge as described in 'Preparing sources for APT' on page 12, update your packages lists and install the `kernel-image-2.4.27-2-sparc32` package.

with dpkg: to install the packages directly with `dpkg` you need to download the necessary files first.

- `http://ftp.debian.org/debian/pool/main/k/kernel-image-2.4.27-sparc/kernel-image-2.4.27-2-sparc32_2.4.27-2_sparc.deb`
- `http://ftp.debian.org/debian/dists/sarge/main/upgrade-kernel/modutils_2.4.26-1.2woody1_sparc.deb`

- http://ftp.debian.org/debian/dists/sarge/main/upgrade-kernel/initrd-tools_0.1.79-0.woody1_all.deb
- http://ftp.debian.org/debian/dists/sarge/main/upgrade-kernel/cramfsprogs_1.1-6.woody1_sparc.deb

The kernel package depends on `modutils`; `initrd-tools` depends on `cramfsprogs`. All other dependencies (which are `stat`, `cpio` and `ash`) can be satisfied with packages from woody in the usual way.

Don't delete your old kernel yet You should first verify that the new one boots and all hardware needed for the upgrade works (e.g. network adaptors).

Make your system bootable You will probably have to adapt your boot loader configuration `/etc/silo.conf`. Note that the kernel now uses an initrd while the Debian kernels in woody did not.

If you currently use `raidtools2`, you should read 'Upgrading from raidtools2 to mdadm' on page 18 before you reboot.

Reboot to the new kernel

Check your system Check especially input devices, display devices, devices needed to access the sarge packages (i.e. network adaptors, CD drives, etc). Some driver modules may have been renamed, some drivers which have been compiled into the old kernel might now be compiled as modules, ...

Appendix B

Managing your woody system

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

B.1 Upgrading your woody system

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

B.2 Installing woody version of aptitude

First you need to make sure you will install woody's version of `aptitude` and not sarge's by following the instructions in 'Checking your sources list' on the current page.

After that, just execute

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

to install `aptitude`.

B.3 Checking your sources list

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

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

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