debian

Debian Reference

This Debian Reference (version 2.72+9.0trisquel2) (2019-11-15 20:38:45 UTC) is intended to provide a broad overview of the De-
bian system as a post-installation user's guide. It covers many aspects of system administration through shell-command examples
for non-developers.

	COLLABORATORS		
	TITLE : Debian Reference		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Osamu Aoki	November 15, 2019	

	REVISION HISTORY		
NUMBER	DATE	DESCRIPTION	NAME

Contents

List of Tables

Abstract

This book is free; you may redistribute it and/or modify it under the terms of the GNU General Public License of any version compliant to the Debian Free Software Guidelines (DFSG).

Preface

This Debian Reference (version 2.72+9.0trisquel2) (2019-11-15 20:38:45 UTC) is intended to provide a broad overview of the Debian system administration as a post-installation user guide.

The target reader is someone who is willing to learn shell scripts but who is not ready to read all the C sources to figure out how the GNU/Linux system works.

For installation instructions, see:

- Debian GNU/Linux Installation Guide for current stable system

- Debian GNU/Linux Installation Guide for current testing system

Disclaimer

All warranties are disclaimed. All trademarks are property of their respective trademark owners.

The Debian system itself is a moving target. This makes its documentation difficult to be current and correct. Although the current unstable version of the Debian system was used as the basis for writing this, some contents may be already outdated by the time you read this.

Please treat this document as the secondary reference. This document does not replace any authoritative guides. The author and contributors do not take responsibility for consequences of errors, omissions or ambiguity in this document.

What is Debian

The Debian Project is an association of individuals who have made common cause to create a free operating system. It's distribution is characterized by the following.

- Commitment to the software freedom: Debian Social Contract and Debian Free Software Guidelines (DFSG)

- Internet based distributed unpaid volunteer effort: https://www.debian.org

- Large number of pre-compiled high quality software packages

- Focus on stability and security with easy access to the security updates

- Focus on smooth upgrade to the latest software packages in the `unstable` and `testing` archives

- Large number of supported hardware architectures

Free Software pieces in Debian come from GNU, Linux, BSD, X, ISC, Apache, Ghostscript, Common Unix Printing System , Samba, GNOME, KDE, Mozilla, LibreOffice, Vim, TeX, LaTeX, DocBook, Perl, Python, Tcl, Java, Ruby, PHP, Berkeley DB, MariaDB, PostgreSQL, SQLite, Exim, Postfix, Mutt, FreeBSD, OpenBSD, Plan 9 and many more independent free software projects. Debian integrates this diversity of Free Software into one system.

About this document

Guiding rules

Following guiding rules are followed while compiling this document.

- Provide overview and skip corner cases. (**Big Picture**)

- Keep It Short and Simple. (**KISS**)

- Do not reinvent the wheel. (Use pointers to **the existing references**)

- Focus on non-GUI tools and consoles. (Use **shell examples**)

- Be objective. (Use popcon etc.)

Tip
I tried to elucidate hierarchical aspects and lower levels of the system.

Prerequisites

⚠ **Warning**
You are expected to make good efforts to seek answers by yourself beyond this documentation. This document only gives efficient starting points.

You must seek solution by yourself from primary sources.

- The Debian Administrator's Handbook

- The Debian site at https://www.debian.org for the general information

- The documentation under the "`/usr/share/doc/<package_name>`" directory

- The Unix style **manpage**: "`dpkg -L <package_name> |grep '/man/man.*/'`"

- The GNU style **info page**: "`dpkg -L <package_name> |grep '/info/'`"

- The bug report: http://bugs.debian.org/<package_name>

- The Debian Wiki at https://wiki.debian.org/ for the moving and specific topics

- The HOWTOs from The Linux Documentation Project (TLDP) at http://tldp.org/

- The Single UNIX Specification from the Open Group's The UNIX System Home Page at http://www.unix.org/

- The free encyclopedia from Wikipedia at https://www.wikipedia.org/

Note
For detailed documentation, you may need to install the corresponding documentation package named with "`-doc`" as its suffix.

Conventions

This document provides information through the following simplified presentation style with bash(1) shell command examples.

```
# <command in root account>
$ <command in user account>
```

These shell prompts distinguish account used and correspond to set environment variables as: "PS1='\$'" and "PS2=' '". These values are chosen for the sake of readability of this document and are not typical on actual installed system.

Note
See the meaning of the "$PS1" and "$PS2" environment variables in bash(1).

Action required by the system administrator is written in the imperative sentence, e.g. "Type Enter-key after typing each command string to the shell."

The **description** column and similar ones in the table may contain a **noun phrase** following the package short description convention which drops leading articles such as "a" and "the". They may alternatively contain an infinitive phrase as a **noun phrase** without leading "to" following the short command description convention in manpages. These may look funny to some people but are my intentional choices of style to keep this documentation as simple as possible. These **Noun phrases** do not capitalize their starting nor end with periods following these short description convention.

Note
Proper nouns including command names keeps their case irrespective of their location.

A **command snippet** quoted in a text paragraph is referred by the typewriter font between double quotation marks, such as "aptitude safe-upgrade".

A **text data** from a configuration file quoted in a text paragraph is referred by the typewriter font between double quotation marks, such as "deb-src".

A **command** is referred by its name in the typewriter font optionally followed by its manpage section number in parenthesis, such as bash(1). You are encouraged to obtain information by typing the following.

```
$ man 1 bash
```

A **manpage** is referred by its name in the typewriter font followed by its manpage section number in parenthesis, such as sources.list(5). You are encouraged to obtain information by typing the following.

```
$ man 5 sources.list
```

An **info page** is referred by its command snippet in the typewriter font between double quotation marks, such as "info make". You are encouraged to obtain information by typing the following.

```
$ info make
```

A **filename** is referred by the typewriter font between double quotation marks, such as "/etc/passwd". For configuration files, you are encouraged to obtain information by typing the following.

```
$ sensible-pager "/etc/passwd"
```

A **directory name** is referred by the typewriter font between double quotation marks, such as "/etc/apt/". You are encouraged to explore its contents by typing the following.

```
$ mc "/etc/apt/"
```

A **package name** is referred by its name in the typewriter font, such as vim. You are encouraged to obtain information by typing the following.

```
$ dpkg -L vim
$ apt-cache show vim
$ aptitude show vim
```

A **documentation** may indicate its location by the filename in the typewriter font between double quotation marks, such as "/usr/share/doc/base-passwd/users-and-groups.txt.gz" and "/usr/share/doc/base-passwd/users-an or by its URL, such as https://www.debian.org. You are encouraged to read the documentation by typing the following.

```
$ zcat "/usr/share/doc/base-passwd/users-and-groups.txt.gz" | sensible-pager
$ sensible-browser "/usr/share/doc/base-passwd/users-and-groups.html"
$ sensible-browser "https://www.debian.org"
```

An **environment variable** is referred by its name with leading "$" in the typewriter font between double quotation marks, such as "$TERM". You are encouraged to obtain its current value by typing the following.

```
$ echo "$TERM"
```

The popcon

The popcon data is presented as the objective measure for the popularity of each package. It was downloaded on 2018-03-11 00:00:33 UTC and contains the total submission of 204944 reports over 170151 binary packages and 27 architectures.

Note
Please note that the amd64 unstable archive contains only 58227 packages currently. The popcon data contains reports from many old system installations.

The popcon number preceded with "V:" for "votes" is calculated by "1000 * (the popcon submissions for the package executed recently on the PC)/(the total popcon submissions)".

The popcon number preceded with "I:" for "installs" is calculated by "1000 * (the popcon submissions for the package installed on the PC)/(the total popcon submissions)".

Note
The popcon figures should not be considered as absolute measures of the importance of packages. There are many factors which can skew statistics. For example, some system participating popcon may have mounted directories such as "/bin" with "noatime" option for system performance improvement and effectively disabled "vote" from such system.

The package size

The package size data is also presented as the objective measure for each package. It is based on the "Installed-Size:" reported by "apt-cache show" or "aptitude show" command (currently on amd64 architecture for the unstable release). The reported size is in KiB (Kibibyte = unit for 1024 bytes).

Note
A package with a small numerical package size may indicate that the package in the unstable release is a dummy package which installs other packages with significant contents by the dependency. The dummy package enables a smooth transition or split of the package.

Note
A package size followed by "(*)" indicates that the package in the unstable release is missing and the package size for the experimental release is used instead.

Bug reports on this document

Please file bug reports on the `debian-reference` package using reportbug(1) if you find any issues on this document. Please include correction suggestion by "`diff -u`" to the plain text version or to the source.

Reminders for new users

Here are some reminders for new users:

- Backup your data

- Secure your password and security keys

- KISS (keep it simple stupid)

  – Don't over-engineer your system

- Read your log files

  – The **FIRST** error is the one that counts

- RTFM (read the fine manual)

- Search the Internet before asking questions

- Don't be root when you don't have to be

- Don't mess with the package management system

- Don't type anything you don't understand

- Don't change the file permissions (before the full security review)

- Don't leave your root shell until you **TEST** your changes

- Always have an alternative boot media (USB memory stick, CD, ⋯)

Some quotes for new users

Here are some interesting quotes from the Debian mailing list which may help enlighten new users.

- "This is Unix. It gives you enough rope to hang yourself." --- Miquel van Smoorenburg `<miquels at cistron.nl>`

- "Unix IS user friendly⋯It's just selective about who its friends are." --- Tollef Fog Heen `<tollef at add.no>`

Wikipedia has article "Unix philosophy" which lists interesting quotes.

Chapter 1

GNU/Linux tutorials

I think learning a computer system is like learning a new foreign language. Although tutorial books and documentation are helpful, you have to practice it yourself. In order to help you get started smoothly, I elaborate a few basic points.

The powerful design of Debian GNU/Linux comes from the Unix operating system, i.e., a multiuser, multitasking operating system. You must learn to take advantage of the power of these features and similarities between Unix and GNU/Linux.

Don't shy away from Unix oriented texts and don't rely solely on GNU/Linux texts, as this robs you of much useful information.

Note
If you have been using any Unix-like system for a while with command line tools, you probably know everything I explain here. Please use this as a reality check and refresher.

1.1   Console basics

1.1.1   The shell prompt

Upon starting the system, you are presented with the character based login screen if you did not install X Window System with the display manager such as gdm3. Suppose your hostname is foo, the login prompt looks as follows.

```
foo login:
```

If you did install a GUI environment such as GNOME or KDE, then you can get to a login prompt by Ctrl-Alt-F1, and you can return to the GUI environment via Alt-F7 (see Section 1.1.6 below for more).

At the login prompt, you type your username, e.g. penguin, and press the Enter-key, then type your password and press the Enter-key again.

Note
Following the Unix tradition, the username and password of the Debian system are case sensitive. The username is usually chosen only from the lowercase. The first user account is usually created during the installation. Additional user accounts can be created with adduser(8) by root.

The system starts with the greeting message stored in "/etc/motd" (Message Of The Day) and presents a command prompt.

```
Debian GNU/Linux jessie/sid foo tty1
foo login: penguin
Password:
Last login: Mon Sep 23 19:36:44 JST 2013 on tty3
```

```
Linux snoopy 3.11-1-amd64 #1 SMP Debian 3.11.6-2 (2013-11-01) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
foo:~$
```

Now you are in the shell. The shell interprets your commands.

1.1.2   The shell prompt under X

If you installed X Window System with a display manager such as GNOME's `gdm3` by selecting "Desktop environment" task during the installation, you are presented with the graphical login screen upon starting your system. You type your username and your password to login to the non-privileged user account. Use tab to navigate between username and password, or use the mouse and primary click.

You can gain the shell prompt under X by starting a `x-terminal-emulator` program such as gnome-terminal(1), rxvt(1) or xterm(1). Under the GNOME Desktop environment, clicking "Applications" → "Accessories" → "Terminal" does the trick.

You can also see the section below Section 1.1.6.

Under some other Desktop systems (like `fluxbox`), there may be no obvious starting point for the menu. If this happens, just try (right) clicking the background of the desktop screen and hope for a menu to pop-up.

1.1.3   The root account

The root account is also called superuser or privileged user. From this account, you can perform the following system administration tasks.

• Read, write, and remove any files on the system irrespective of their file permissions

• Set file ownership and permissions of any files on the system

• Set the password of any non-privileged users on the system

• Login to any accounts without their passwords

This unlimited power of root account requires you to be considerate and responsible when using it.

⚠ **Warning**
Never share the root password with others.

Note
File permissions of a file (including hardware devices such as CD-ROM etc. which are just another file for the Debian system) may render it unusable or inaccessible by non-root users. Although the use of root account is a quick way to test this kind of situation, its resolution should be done through proper setting of file permissions and user's group membership (see Section 1.2.3).

1.1.4   The root shell prompt

Here are a few basic methods to gain the root shell prompt by using the root password.

- Type `root` at the character based login prompt.

- Click "Applications" → "Accessories" → "Root Terminal", under the GNOME Desktop environment.

- Type "`su -l`" from any user shell prompt.

  – This does not preserve the environment of the current user.

- Type "`su`" from any user shell prompt.

  – This preserves some of the environment of the current user.

1.1.5   GUI system administration tools

When your desktop menu does not start GUI system administration tools automatically with the appropriate privilege, you can start them from the root shell prompt of the X terminal emulator, such as gnome-terminal(1), rxvt(1), or xterm(1). See Section 1.1.4 and Section 7.8.5.

> ⚠️ **Warning**
> Never start the X display/session manager under the root account by typing in `root` to the prompt of the display manager such as gdm3(1).

> ⚠️ **Warning**
> Never run untrusted remote GUI program under X Window when critical information is displayed since it may eavesdrop your X screen.

1.1.6   Virtual consoles

In the default Debian system, there are six switchable VT100-like character consoles available to start the command shell directly on the Linux host. Unless you are in a GUI environment, you can switch between the virtual consoles by pressing the `Left-Alt-key` and one of the `F1` —`F6` keys simultaneously. Each character console allows independent login to the account and offers the multiuser environment. This multiuser environment is a great Unix feature, and very addictive.

If you are under the X Window System, you gain access to the character console 1 by pressing `Ctrl-Alt-F1` key, i.e., the `left-Ctrl-key`, the `left-Alt-key`, and the `F1-key` are pressed together. You can get back to the X Window System, normally running on the virtual console 7, by pressing `Alt-F7`.

You can alternatively change to another virtual console, e.g. to the console 1, from the commandline.

```
# chvt 1
```

1.1.7   How to leave the command prompt

You type `Ctrl-D`, i.e., the `left-Ctrl-key` and the `d-key` pressed together, at the command prompt to close the shell activity. If you are at the character console, you return to the login prompt with this. Even though these control characters are referred as "control D" with the upper case, you do not need to press the Shift-key. The short hand expression, ˆD, is also used for `Ctrl-D`. Alternately, you can type "exit".

If you are at x-terminal-emulator(1), you can close `x-terminal-emulator` window with this.

1.1.8   How to shutdown the system

Just like any other modern OS where the file operation involves caching data in memory for improved performance, the Debian system needs the proper shutdown procedure before power can safely be turned off. This is to maintain the integrity of files, by forcing all changes in memory to be written to disk. If the software power control is available, the shutdown procedure automatically turns off power of the system. (Otherwise, you may have to press power button for few seconds after the shutdown procedure.)

You can shutdown the system under the normal multiuser mode from the commandline.

```
# shutdown -h now
```

You can shutdown the system under the single-user mode from the commandline.

```
# poweroff -i -f
```

Alternatively, you may type `Ctrl-Alt-Delete` (The `left-Ctrl-key`, the `left-Alt-Key`, and the `Delete` are pressed together) to shutdown if "`/etc/inittab`" contains "`ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -h now`" in it. See inittab(5) for details.

See Section 6.9.6.

1.1.9   Recovering a sane console

When the screen goes berserk after doing some funny things such as "`cat <some-binary-file>`", type "`reset`" at the command prompt. You may not be able to see the command echoed as you type. You may also issue "`clear`" to clean up the screen.

1.1.10   Additional package suggestions for the newbie

Although even the minimal installation of the Debian system without any desktop environment tasks provides the basic Unix functionality, it is a good idea to install few additional commandline and curses based character terminal packages such as `mc` and `vim` with apt-get(8) for beginners to get started by the following.

```
# apt-get update
 ...
# apt-get install mc vim sudo
 ...
```

If you already had these packages installed, no new packages are installed.

package	popcon	size	description
mc	V:67, I:234	1428	A text-mode full-screen file manager
sudo	V:447, I:719	3178	A program to allow limited root privileges to users
vim	V:121, I:390	2660	Unix text editor Vi IMproved, a programmers text editor (standard version)
vim-tiny	V:68, I:967	1241	Unix text editor Vi IMproved, a programmers text editor (compact version)
emacs25	V:10, I:19	19216	GNU project Emacs, the Lisp based extensible text editor
w3m	V:126, I:564	2302	Text-mode WWW browsers
gpm	V:12, I:20	497	The Unix style cut-and-paste on the text console (daemon)

Table 1.1: List of interesting text-mode program packages

It may be a good idea to read some informative documentations.

You can install some of these packages by the following.

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

package	popcon	size	description
doc-debian	I:859	166	Debian Project documentation, (Debian FAQ) and other documents
debian-policy	I:78	5237	Debian Policy Manual and related documents
developers-reference	I:6	1304	Guidelines and information for Debian developers
maint-guide	I:4	987	Debian New Maintainers' Guide
debian-history	I:1	5034	History of the Debian Project
debian-faq	I:849	1277	Debian FAQ

Table 1.2: List of informative documentation packages

1.1.11 An extra user account

If you do not want to use your main user account for the following training activities, you can create a training user account, e.g. `fish` by the following.

```
# adduser fish
```

Answer all questions.

This creates a new account named as `fish`. After your practice, you can remove this user account and its home directory by the following.

```
# deluser --remove-home fish
```

1.1.12 sudo configuration

For the typical single user workstation such as the desktop Debian system on the laptop PC, it is common to deploy simple configuration of sudo(8) as follows to let the non-privileged user, e.g. `penguin`, to gain administrative privilege just with his user password but without the root password.

```
# echo "penguin  ALL=(ALL) ALL" >> /etc/sudoers
```

Alternatively, it is also common to do as follows to let the non-privileged user, e.g. `penguin`, to gain administrative privilege without any password.

```
# echo "penguin  ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers
```

This trick should only be used for the single user workstation which you administer and where you are the only user.

⚠ Warning
Do not set up accounts of regular users on multiuser workstation like this because it would be very bad for system security.

⚠ Caution
The password and the account of the `penguin` in the above example requires as much protection as the root password and the root account.

⚠ Caution
Administrative privilege in this context belongs to someone authorized to perform the system administration task on the workstation. Never give some manager in the Admin department of your company or your boss such privilege unless they are authorized and capable.

Note

For providing access privilege to limited devices and limited files, you should consider to use **group** to provide limited access instead of using the `root` privilege via sudo(8).

Note

With more thoughtful and careful configuration, sudo(8) can grant limited administrative privileges to other users on a shared system without sharing the root password. This can help with accountability with hosts with multiple administrators so you can tell who did what. On the other hand, you might not want anyone else to have such privileges.

1.1.13 Play time

Now you are ready to play with the Debian system without risks as long as you use the non-privileged user account.

This is because the Debian system is, even after the default installation, configured with proper file permissions which prevent non-privileged users from damaging the system. Of course, there may still be some holes which can be exploited but those who worry about these issues should not be reading this section but should be reading Securing Debian Manual.

We learn the Debian system as a Unix-like system with the following.

- Section 1.2 (basic concept)

- Section 1.3 (survival method)

- Section 1.4 (basic method)

- Section 1.5 (shell mechanism)

- Section 1.6 (text processing method)

1.2   Unix-like filesystem

In GNU/Linux and other Unix-like operating systems, files are organized into directories. All files and directories are arranged in one big tree rooted at "/". It's called a tree because if you draw the filesystem, it looks like a tree but it is upside down.

These files and directories can be spread out over several devices. mount(8) serves to attach the filesystem found on some device to the big file tree. Conversely, umount(8) detaches it again. On recent Linux kernels, mount(8) with some options can bind part of a file tree somewhere else or can mount filesystem as shared, private, slave, or unbindable. Supported mount options for each filesystem are available in "/usr/share/doc/linux-doc-*/Documentation/filesystems/".

Directories on Unix systems are called **folders** on some other systems. Please also note that there is no concept for **drive** such as "A:" on any Unix system. There is one filesystem, and everything is included. This is a huge advantage compared to Windows.

1.2.1   Unix file basics

Here are some Unix file basics.

- Filenames are **case sensitive**. That is, "MYFILE" and "MyFile" are different files.

- The **root directory** means root of the filesystem referred as simply "/". Don't confuse this with the home directory for the root user: "/root".

- Every directory has a name which can contain any letters or symbols **except "/"**. The root directory is an exception; its name is "/" (pronounced "slash" or "the root directory") and it cannot be renamed.

- Each file or directory is designated by a **fully-qualified filename**, **absolute filename**, or **path**, giving the sequence of directories which must be passed through to reach it. The three terms are synonymous.

- All **fully-qualified filenames** begin with the "/" directory, and there's a "/" between each directory or file in the filename. The first "/" is the top level directory, and the other "/"'s separate successive subdirectories, until we reach the last entry which is the name of the actual file. The words used here can be confusing. Take the following **fully-qualified filename** as an example: "/usr/share/keytables/us.map.gz". However, people also refers to its basename "us.map.gz" alone as a filename.

- The root directory has a number of branches, such as "/etc/" and "/usr/". These subdirectories in turn branch into still more subdirectories, such as "/etc/init.d/" and "/usr/local/". The whole thing viewed collectively is called the **directory tree**. You can think of an absolute filename as a route from the base of the tree ("/") to the end of some branch (a file). You also hear people talk about the directory tree as if it were a **family** tree encompassing all direct descendants of a single figure called the root directory ("/"): thus subdirectories have **parents**, and a path shows the complete ancestry of a file. There are also relative paths that begin somewhere other than the root directory. You should remember that the directory "../" refers to the parent directory. This terminology also applies to other directory like structures, such as hierarchical data structures.

- There's no special directory path name component that corresponds to a physical device, such as your hard disk. This differs from RT-11, CP/M, OpenVMS, MS-DOS, AmigaOS, and Microsoft Windows, where the path contains a device name such as "C:\". (However, directory entries do exist that refer to physical devices as a part of the normal filesystem. See Section 1.2.2.)

Note
While you **can** use almost any letters or symbols in a file name, in practice it is a bad idea to do so. It is better to avoid any characters that often have special meanings on the command line, including spaces, tabs, newlines, and other special characters: { } ( ) [ ] ' ` " \ / > < | ; ! # & ^ * % @ $. If you want to separate words in a name, good choices are the period, hyphen, and underscore. You could also capitalize each word, "LikeThis". Experienced Linux users tend to avoid spaces in filenames.

Note
The word "root" can mean either "root user" or "root directory". The context of their usage should make it clear.

Note
The word **path** is used not only for **fully-qualified filename** as above but also for the **command search path**. The intended meaning is usually clear from the context.

The detailed best practices for the file hierarchy are described in the Filesystem Hierarchy Standard ("/usr/share/doc/debian-po and hier(7)). You should remember the following facts as the starter.

directory	usage of the directory
/	the root directory
/etc/	system wide configuration files
/var/log/	system log files
/home/	all the home directories for all non-privileged users

Table 1.3: List of usage of key directories

1.2.2 Filesystem internals

Following the **Unix tradition**, the Debian GNU/Linux system provides the filesystem under which physical data on hard disks and other storage devices reside, and the interaction with the hardware devices such as console screens and remote serial consoles are represented in an unified manner under "/dev/".

Each file, directory, named pipe (a way two programs can share data), or physical device on a Debian GNU/Linux system has a data structure called an inode which describes its associated attributes such as the user who owns it (owner), the group that it belongs to, the time last accessed, etc. The idea of representing just about everything in the filesystem was a Unix innovation, and modern Linux kernels have developed this idea ever further. Now, even information about processes running in the computer can be found in the filesystem.

This abstract and unified representation of physical entities and internal processes is very powerful since this allows us to use the same command for the same kind of operation on many totally different devices. It is even possible to change the way the kernel works by writing data to special files that are linked to running processes.

Tip
If you need to identify the correspondence between the file tree and the physical entity, execute mount(8) with no arguments.

1.2.3 Filesystem permissions

Filesystem permissions of Unix-like system are defined for three categories of affected users.

- The **user** who owns the file (**u**)
- Other users in the **group** which the file belongs to (**g**)
- All **other** users (**o**) also referred to as "world" and "everyone"

For the file, each corresponding permission allows following actions.

- The **read** (**r**) permission allows owner to examine contents of the file.
- The **write** (**w**) permission allows owner to modify the file.
- The **execute** (**x**) permission allows owner to run the file as a command.

For the directory, each corresponding permission allows following actions.

- The **read** (**r**) permission allows owner to list contents of the directory.
- The **write** (**w**) permission allows owner to add or remove files in the directory.
- The **execute** (**x**) permission allows owner to access files in the directory.

Here, the **execute** permission on a directory means not only to allow reading of files in that directory but also to allow viewing their attributes, such as the size and the modification time.

ls(1) is used to display permission information (and more) for files and directories. When it is invoked with the "-l" option, it displays the following information in the order given.

- **Type of file** (first character)
- Access **permission** of the file (nine characters, consisting of three characters each for user, group, and other in this order)
- **Number of hard links** to the file
- Name of the **user** who owns the file
- Name of the **group** which the file belongs to
- **Size** of the file in characters (bytes)
- **Date and time** of the file (mtime)

character	meaning
-	normal file
d	directory
l	symlink
c	character device node
b	block device node
p	named pipe
s	socket

Table 1.4: List of the first character of "ls -l" output

- **Name** of the file

chown(1) is used from the root account to change the owner of the file. chgrp(1) is used from the file's owner or root account to change the group of the file. chmod(1) is used from the file's owner or root account to change file and directory access permissions. Basic syntax to manipulate a foo file is the following.

```
# chown <newowner> foo
# chgrp <newgroup> foo
# chmod  [ugoa][+-=][rwxXst][,...] foo
```

For example, you can make a directory tree to be owned by a user foo and shared by a group bar by the following.

```
# cd /some/location/
# chown -R foo:bar .
# chmod -R ug+rwX,o=rX .
```

There are three more special permission bits.

- The **set user ID** bit (**s** or **S** instead of user's **x**)

- The **set group ID** bit (**s** or **S** instead of group's **x**)

- The **sticky** bit (**t** or **T** instead of other's **x**)

Here the output of "ls -l" for these bits is **capitalized** if execution bits hidden by these outputs are **unset**.

Setting **set user ID** on an executable file allows a user to execute the executable file with the owner ID of the file (for example **root**). Similarly, setting **set group ID** on an executable file allows a user to execute the executable file with the group ID of the file (for example **root**). Because these settings can cause security risks, enabling them requires extra caution.

Setting **set group ID** on a directory enables the BSD-like file creation scheme where all files created in the directory belong to the **group** of the directory.

Setting the **sticky bit** on a directory prevents a file in the directory from being removed by a user who is not the owner of the file. In order to secure contents of a file in world-writable directories such as "/tmp" or in group-writable directories, one must not only reset the **write** permission for the file but also set the **sticky bit** on the directory. Otherwise, the file can be removed and a new file can be created with the same name by any user who has write access to the directory.

Here are a few interesting examples of file permissions.

```
$ ls -l /etc/passwd /etc/shadow /dev/ppp /usr/sbin/exim4
crw------T 1 root root    108, 0 Oct 16 20:57 /dev/ppp
-rw-r--r-- 1 root root      2761 Aug 30 10:38 /etc/passwd
-rw-r----- 1 root shadow   1695 Aug 30 10:38 /etc/shadow
-rwsr-xr-x 1 root root    973824 Sep 23 20:04 /usr/sbin/exim4
$ ls -ld /tmp /var/tmp /usr/local /var/mail /usr/src
drwxrwxrwt 14 root root  20480 Oct 16 21:25 /tmp
drwxrwsr-x 10 root staff  4096 Sep 29 22:50 /usr/local
drwxr-xr-x 10 root root   4096 Oct 11 00:28 /usr/src
drwxrwsr-x  2 root mail   4096 Oct 15 21:40 /var/mail
drwxrwxrwt  3 root root   4096 Oct 16 21:20 /var/tmp
```

There is an alternative numeric mode to describe file permissions with chmod(1). This numeric mode uses 3 to 4 digit wide octal (radix=8) numbers.

digit	meaning
1st optional digit	sum of **set user ID** (=4), **set group ID** (=2), and **sticky bit** (=1)
2nd digit	sum of **read** (=4), **write** (=2), and **execute** (=1) permissions for **user**
3rd digit	ditto for **group**
4th digit	ditto for **other**

Table 1.5: The numeric mode for file permissions in chmod(1) commands

This sounds complicated but it is actually quite simple. If you look at the first few (2-10) columns from "ls -l" command output and read it as a binary (radix=2) representation of file permissions ("-" being "0" and "rwx" being "1"), the last 3 digit of the numeric mode value should make sense as an octal (radix=8) representation of file permissions to you.

For example, try the following

```
$ touch foo bar
$ chmod u=rw,go=r foo
$ chmod 644 bar
$ ls -l foo bar
-rw-r--r-- 1 penguin penguin 0 Oct 16 21:39 bar
-rw-r--r-- 1 penguin penguin 0 Oct 16 21:35 foo
```

Tip
If you need to access information displayed by "ls -l" in shell script, you should use pertinent commands such as test(1), stat(1) and readlink(1). The shell builtin such as "[" or "test" may be used too.

1.2.4 Control of permissions for newly created files: umask

What permissions are applied to a newly created file or directory is restricted by the umask shell builtin command. See dash(1), bash(1), and builtins(7).

```
(file permissions) = (requested file permissions) & ~(umask value)
```

umask	file permissions created	directory permissions created	usage
0022	-rw-r--r--	-rwxr-xr-x	writable only by the user
0002	-rw-rw-r--	-rwxrwxr-x	writable by the group

Table 1.6: The **umask** value examples

The Debian system uses a user private group (UPG) scheme as its default. A UPG is created whenever a new user is added to the system. A UPG has the same name as the user for which it was created and that user is the only member of the UPG. UPG scheme makes it safe to set umask to 0002 since every user has their own private group. (In some Unix variants, it is quite common to setup all normal users belonging to a single **users** group and is a good idea to set umask to 0022 for security in such cases.)

Tip
Enable UPG by putting "umask 002" in the ~/.bashrc file.

1.2.5 Permissions for groups of users (group)

In order to make group permissions to be applied to a particular user, that user needs to be made a member of the group using "sudo vigr" for /etc/group and "sudo vigr -s" for /etc/gshadow. You need to login after logout (or run "exec newgrp") to enable the new group configuration.

Note
Alternatively, you may dynamically add users to groups during the authentication process by
adding   "auth optional pam_group.so"   line   to   "/etc/pam.d/common-auth"   and   setting
"/etc/security/group.conf". (See Chapter 4.)

The hardware devices are just another kind of file on the Debian system. If you have problems accessing devices such as CD-ROM and USB memory stick from a user account, you should make that user a member of the relevant group.

Some notable system-provided groups allow their members to access particular files and devices without root privilege.

group	description for accessible files and devices
dialout	full and direct access to serial ports ("/dev/ttyS[0-3]")
dip	limited access to serial ports for **Dialup IP** connection to trusted peers
cdrom	CD-ROM, DVD+/-RW drives
audio	audio device
video	video device
scanner	scanner(s)
adm	system monitoring logs
staff	some directories for junior administrative work: "/usr/local", "/home"

Table 1.7: List of notable system-provided groups for file access

---

Tip
You need to belong to the dialout group to reconfigure modem, dial anywhere, etc. But if root creates pre-
defined configuration files for trusted peers in "/etc/ppp/peers/", you only need to belong to the dip group to
create **Dialup IP** connection to those trusted peers using pppd(8), pon(1), and poff(1) commands.

Some notable system-provided groups allow their members to execute particular commands without root privilege.

group	accessible commands
sudo	execute sudo without their password
lpadmin	execute commands to add, modify, and remove printers from printer databases

Table 1.8: List of notable system provided groups for particular command executions

For the full listing of the system provided users and groups, see the recent version of the "Users and Groups" document in "/usr/share/doc/base-passwd/users-and-groups.html" provided by the base-passwd package.

See passwd(5), group(5), shadow(5), newgrp(1), vipw(8), vigr(8), and pam_group(8) for management commands of the user and group system.

1.2.6 Timestamps

There are three types of timestamps for a GNU/Linux file.

type	meaning (historic Unix definition)
mtime	the file modification time (`ls -l`)
ctime	the file status change time (`ls -lc`)
atime	the last file access time (`ls -lu`)

Table 1.9: List of types of timestamps

Note
ctime is not file creation time.

Note
The actual value of **atime** on GNU/Linux system may be different from that of the historic Unix definition.

- Overwriting a file changes all of the **mtime**, **ctime**, and **atime** attributes of the file.

- Changing ownership or permission of a file changes the **ctime** and **atime** attributes of the file.

- Reading a file changes the **atime** attribute of the file on the historic Unix system.

- Reading a file changes the **atime** attribute of the file on the GNU/Linux system if its filesystem is mounted with "`strictatime`".

- Reading a file for the first time or after one day changes the **atime** attribute of the file on the GNU/Linux system if its filesystem is mounted with "`relatime`". (default behavior since Linux 2.6.30)

- Reading a file doesn't change the **atime** attribute of the file on the GNU/Linux system if its filesystem is mounted with "`noatime`".

Note
The "`noatime`" and "`relatime`" mount options are introduced to improve the filesystem read performance under the normal use cases. Simple file read operation under the "`strictatime`" option accompanies the time-consuming write operation to update the **atime** attribute. But the **atime** attribute is rarely used except for the mbox(5) file. See mount(8).

Use touch(1) command to change timestamps of existing files.

For timestamps, the `ls` command outputs different strings under non-English locale ("`fr_FR.UTF-8`") from under the old one ("`C`").

```
$ LANG=fr_FR.UTF-8  ls -l foo
-rw-rw-r-- 1 penguin penguin 0 oct. 16 21:35 foo
$ LANG=C  ls -l foo
-rw-rw-r-- 1 penguin penguin 0 Oct 16 21:35 foo
```

Tip
See Section 9.2.5 to customize "`ls -l`" output.

1.2.7 Links

There are two methods of associating a file "`foo`" with a different filename "`bar`".

- Hard link

– Duplicate name for an existing file

– ”ln foo bar”

- Symbolic link or symlink

    – Special file that points to another file by name

    – ”ln -s foo bar”

See the following example for changes in link counts and the subtle differences in the result of the `rm` command.

```
$ umask 002
$ echo "Original Content" > foo
$ ls -li foo
1449840 -rw-rw-r-- 1 penguin penguin 17 Oct 16 21:42 foo
$ ln foo bar     # hard link
$ ln -s foo baz  # symlink
$ ls -li foo bar baz
1449840 -rw-rw-r-- 2 penguin penguin 17 Oct 16 21:42 bar
1450180 lrwxrwxrwx 1 penguin penguin  3 Oct 16 21:47 baz -> foo
1449840 -rw-rw-r-- 2 penguin penguin 17 Oct 16 21:42 foo
$ rm foo
$ echo "New Content" > foo
$ ls -li foo bar baz
1449840 -rw-rw-r-- 1 penguin penguin 17 Oct 16 21:42 bar
1450180 lrwxrwxrwx 1 penguin penguin  3 Oct 16 21:47 baz -> foo
1450183 -rw-rw-r-- 1 penguin penguin 12 Oct 16 21:48 foo
$ cat bar
Original Content
$ cat baz
New Content
```

The hardlink can be made within the same filesystem and shares the same inode number which the ”-i” option with ls(1) reveals.

The symlink always has nominal file access permissions of ”rwxrwxrwx”, as shown in the above example, with the effective access permissions dictated by permissions of the file that it points to.

⚠ Caution

It is generally a good idea not to create complicated symbolic links or hardlinks at all unless you have a very good reason. It may cause nightmares where the logical combination of the symbolic links results in loops in the filesystem.

Note

It is generally preferable to use symbolic links rather than hardlinks unless you have a good reason for using a hardlink.

The ”.” directory links to the directory that it appears in, thus the link count of any new directory starts at 2. The ”..” directory links to the parent directory, thus the link count of the directory increases with the addition of new subdirectories.

If you are just moving to Linux from Windows, it soon becomes clear how well-designed the filename linking of Unix is, compared with the nearest Windows equivalent of ”shortcuts”. Because it is implemented in the filesystem, applications can't see any difference between a linked file and the original. In the case of hardlinks, there really is no difference.

1.2.8 Named pipes (FIFOs)

A named pipe is a file that acts like a pipe. You put something into the file, and it comes out the other end. Thus it's called a FIFO, or First-In-First-Out: the first thing you put in the pipe is the first thing to come out the other end.

If you write to a named pipe, the process which is writing to the pipe doesn't terminate until the information being written is read from the pipe. If you read from a named pipe, the reading process waits until there is nothing to read before terminating. The size of the pipe is always zero --- it does not store data, it just links two processes like the functionality offered by the shell "|" syntax. However, since this pipe has a name, the two processes don't have to be on the same command line or even be run by the same user. Pipes were a very influential innovation of Unix.

For example, try the following

```
$ cd; mkfifo mypipe
$ echo "hello" >mypipe & # put into background
[1] 8022
$ ls -l mypipe
prw-rw-r-- 1 penguin penguin 0 Oct 16 21:49 mypipe
$ cat mypipe
hello
[1]+  Done                    echo "hello" >mypipe
$ ls mypipe
mypipe
$ rm mypipe
```

1.2.9 Sockets

Sockets are used extensively by all the Internet communication, databases, and the operating system itself. It is similar to the named pipe (FIFO) and allows processes to exchange information even between different computers. For the socket, those processes do not need to be running at the same time nor to be running as the children of the same ancestor process. This is the endpoint for the inter process communication (IPC). The exchange of information may occur over the network between different hosts. The two most common ones are the Internet socket and the Unix domain socket.

Tip
"netstat -an" provides a very useful overview of sockets that are open on a given system.

1.2.10 Device files

Device files refer to physical or virtual devices on your system, such as your hard disk, video card, screen, or keyboard. An example of a virtual device is the console, represented by "/dev/console".

There are 2 types of device files.

- **Character device**

  - Accessed one character at a time
  - 1 character = 1 byte
  - E.g. keyboard device, serial port, ⋯

- **Block device**

  - accessed in larger units called blocks
  - 1 block > 1 byte
  - E.g. hard disk, ⋯

You can read and write device files, though the file may well contain binary data which may be an incomprehensible-to-humans gibberish. Writing data directly to these files is sometimes useful for the troubleshooting of hardware connections. For example, you can dump a text file to the printer device "/dev/lp0" or send modem commands to the appropriate serial port "/dev/ttyS0". But, unless this is done carefully, it may cause a major disaster. So be cautious.

Note
For the normal access to a printer, use lp(1).

The device node number are displayed by executing ls(1) as the following.

```
$ ls -l /dev/sda /dev/sr0 /dev/ttyS0 /dev/zero
brw-rw---T  1 root disk     8,  0 Oct 16 20:57 /dev/sda
brw-rw---T+ 1 root cdrom   11,  0 Oct 16 21:53 /dev/sr0
crw-rw---T  1 root dialout  4, 64 Oct 16 20:57 /dev/ttyS0
crw-rw-rw-  1 root root     1,  5 Oct 16 20:57 /dev/zero
```

• "`/dev/sda`" has the major device number 8 and the minor device number 0. This is read/write accessible by users belonging to the `disk` group.

• "`/dev/sr0`" has the major device number 11 and the minor device number 0. This is read/write accessible by users belonging to the `cdrom` group.

• "`/dev/ttyS0`" has the major device number 4 and the minor device number 64. This is read/write accessible by users belonging to the `dialout` group.

• "`/dev/zero`" has the major device number 1 and the minor device number 5. This is read/write accessible by anyone.

On the modern Linux system, the filesystem under "`/dev/`" is automatically populated by the udev(7) mechanism.

1.2.11   Special device files

There are some special device files.

device file	action	description of response
/dev/null	read	return "end-of-file (EOF) character"
/dev/null	write	return nothing (a bottomless data dump pit)
/dev/zero	read	return "the \0 (NUL) character" (not the same as the number zero ASCII)
/dev/random	read	return random characters from a true random number generator, delivering real entropy (slow)
/dev/urandom	read	return random characters from a cryptographically secure pseudorandom number generator
/dev/full	write	return the disk-full (ENOSPC) error

Table 1.10: List of special device files

These are frequently used in conjunction with the shell redirection (see Section 1.5.8).

1.2.12   procfs and sysfs

The procfs and sysfs mounted on "`/proc`" and "`/sys`" are the pseudo-filesystem and expose internal data structures of the kernel to the userspace. In other word, these entries are virtual, meaning that they act as a convenient window into the operation of the operating system.

The directory "`/proc`" contains (among other things) one subdirectory for each process running on the system, which is named after the process ID (PID). System utilities that access process information, such as ps(1), get their information from this directory structure.

The directories under "`/proc/sys/`" contain interfaces to change certain kernel parameters at run time. (You may do the same through the specialized sysctl(8) command or its preload/configuration file "`/etc/sysctl.conf`".)

People frequently panic when they notice one file in particular - "/proc/kcore" - which is generally huge. This is (more or less) a copy of the content of your computer's memory. It's used to debug the kernel. It is a virtual file that points to computer memory, so don't worry about its size.

The directory under "/sys" contains exported kernel data structures, their attributes, and their linkages between them. It also contains interfaces to change certain kernel parameters at run time.

See "proc.txt(.gz)", "sysfs.txt(.gz)" and other related documents in the Linux kernel documentation ("/usr/share/doc/ provided by the linux-doc-* package.

1.2.13 tmpfs

The tmpfs is a temporary filesystem which keeps all files in the virtual memory. The data of the tmpfs in the page cache on memory may be swapped out to the swap space on disk as needed.

The directory "/run" is mounted as the tmpfs in the early boot process. This enables writing to it even when the directory "/" is mounted as read-only. This is the new location for the storage of transient state files and replaces several locations described in the Filesystem Hierarchy Standard version 2.3:

- "/var/run" → "/run"

- "/var/lock" → "/run/lock"

- "/dev/shm" → "/run/shm"

See "tmpfs.txt(.gz)" in the Linux kernel documentation ("/usr/share/doc/linux-doc-*/Documentation/filesys provided by the linux-doc-* package.

1.3 Midnight Commander (MC)

Midnight Commander (MC) is a GNU "Swiss army knife" for the Linux console and other terminal environments. This gives newbie a menu driven console experience which is much easier to learn than standard Unix commands.

You may need to install the Midnight Commander package which is titled "mc" by the following.

```
$ sudo apt-get install mc
```

Use the mc(1) command to explore the Debian system. This is the best way to learn. Please explore few interesting locations just using the cursor keys and Enter key.

- "/etc" and its subdirectories

- "/var/log" and its subdirectories

- "/usr/share/doc" and its subdirectories

- "/sbin" and "/bin"

1.3.1 Customization of MC

In order to make MC to change working directory upon exit and cd to the directory, I suggest to modify "~/.bashrc" to include a script provided by the mc package.

```
. /usr/lib/mc/mc.sh
```

See mc(1) (under the "-P" option) for the reason. (If you do not understand what exactly I am talking here, you can do this later.)

1.3.2   Starting MC

MC can be started by the following.

```
$ mc
```

MC takes care of all file operations through its menu, requiring minimal user effort. Just press F1 to get the help screen. You can play with MC just by pressing cursor-keys and function-keys.

Note
In some consoles such as gnome-terminal(1), key strokes of function-keys may be stolen by the console program. You can disable these features by "Edit" → "Keyboard Shortcuts" for `gnome-terminal`.

If you encounter character encoding problem which displays garbage characters, adding "-a" to MC's command line may help prevent problems.

If this doesn't clear up your display problems with MC, see Section 9.4.6.

1.3.3   File manager in MC

The default is two directory panels containing file lists. Another useful mode is to set the right window to "information" to see file access privilege information, etc. Following are some essential keystrokes. With the gpm(8) daemon running, one can use a mouse on Linux character consoles, too. (Make sure to press the shift-key to obtain the normal behavior of cut and paste in MC.)

key	key binding
F1	help menu
F3	internal file viewer
F4	internal editor
F9	activate pull down menu
F10	exit Midnight Commander
Tab	move between two windows
Insert or Ctrl-T	mark file for a multiple-file operation such as copy
Del	delete file (be careful---set MC to safe delete mode)
Cursor keys	self-explanatory

Table 1.11: The key bindings of MC

1.3.4   Command-line tricks in MC

- `cd` command changes the directory shown on the selected screen.

- `Ctrl-Enter` or `Alt-Enter` copies a filename to the command line. Use this with cp(1) and mv(1) commands together with command-line editing.

- `Alt-Tab` shows shell filename expansion choices.

- One can specify the starting directory for both windows as arguments to MC; for example, "mc /etc /root".

- `Esc` + `n-key` → `Fn` (i.e., `Esc` + `1` → `F1`, etc.; `Esc` + `0` → `F10`)

- Pressing `Esc` before the key has the same effect as pressing the `Alt` and the key together.; i.e., type `Esc` + `c` for `Alt-C`. `Esc` is called meta-key and sometimes noted as "M-".

1.3.5   The internal editor in MC

The internal editor has an interesting cut-and-paste scheme. Pressing `F3` marks the start of a selection, a second `F3` marks the end of selection and highlights the selection. Then you can move your cursor. If you press F6, the selected area is moved to the cursor location. If you press F5, the selected area is copied and inserted at the cursor location. `F2` saves the file. `F10` gets you out. Most cursor keys work intuitively.

This editor can be directly started on a file using one of the following commands.

```
$ mc -e filename_to_edit
```

```
$ mcedit filename_to_edit
```

This is not a multi-window editor, but one can use multiple Linux consoles to achieve the same effect. To copy between windows, use Alt-F<n> keys to switch virtual consoles and use "File→Insert file" or "File→Copy to file" to move a portion of a file to another file.

This internal editor can be replaced with any external editor of choice.

Also, many programs use the environment variables "`$EDITOR`" or "`$VISUAL`" to decide which editor to use. If you are uncomfortable with vim(1) or nano(1) initially, you may set these to "`mcedit`" by adding the following lines to "`~/.bashrc`".

```
export EDITOR=mcedit
export VISUAL=mcedit
```

I do recommend setting these to "`vim`" if possible.

If you are uncomfortable with vim(1), you can keep using mcedit(1) for most system maintenance tasks.

1.3.6   The internal viewer in MC

MC is a very smart viewer. This is a great tool for searching words in documents. I always use this for files in the "`/usr/share/doc`" directory. This is the fastest way to browse through masses of Linux information. This viewer can be directly started using one of the following commands.

```
$ mc -v path/to/filename_to_view
```

```
$ mcview path/to/filename_to_view
```

1.3.7   Auto-start features of MC

Press Enter on a file, and the appropriate program handles the content of the file (see Section 9.3.11). This is a very convenient MC feature.

file type	reaction to enter key
executable file	execute command
man file	pipe content to viewer software
html file	pipe content to web browser
"`*.tar.gz`" and "`*.deb`" file	browse its contents as if subdirectory

Table 1.12: The reaction to the enter key in MC

In order to allow these viewer and virtual file features to function, viewable files should not be set as executable. Change their status using chmod(1) or via the MC file menu.

1.3.8   FTP virtual filesystem of MC

MC can be used to access files over the Internet using FTP. Go to the menu by pressing F9, then type "p" to activate the FTP virtual filesystem. Enter a URL in the form "username:passwd@hostname.domainname", which retrieves a remote directory that appears like a local one.

Try "[deb.debian.org/debian]" as the URL and browse the Debian archive.

1.4   The basic Unix-like work environment

Although MC enables you to do almost everything, it is very important for you to learn how to use the command line tools invoked from the shell prompt and become familiar with the Unix-like work environment.

1.4.1   The login shell

You can select your login shell with chsh(1).

package	popcon	size	POSIX shell	description
bash	V:830, I:999	5790	Yes	Bash: the GNU Bourne Again SHell (de facto standard)
tcsh	V:10, I:37	1306	No	TENEX C Shell: an enhanced version of Berkeley csh
dash	V:917, I:985	218	Yes	Debian Almquist Shell, good for shell script
zsh	V:35, I:68	2183	Yes	Z shell: the standard shell with many enhancements
mksh	V:6, I:12	1339	Yes	A version of the Korn shell
csh	V:2, I:11	339	No	OpenBSD C Shell, a version of Berkeley csh
sash	V:0, I:4	969	Yes	Stand-alone shell with builtin commands (Not meant for standard "/bin/sh")
ksh	V:4, I:21	3276	Yes	the real, AT&T version of the Korn shell
rc	V:0, I:3	154	No	implementation of the AT&T Plan 9 rc shell
posh	V:0, I:0	186	Yes	Policy-compliant Ordinary SHell (pdksh derivative)

Table 1.13: List of shell programs

Tip
Although POSIX-like shells share the basic syntax, they can differ in behavior for things as basic as shell variables and glob expansions. Please check their documentation for details.

In this tutorial chapter, the interactive shell always means bash.

1.4.2   Customizing bash

You can customize bash(1) behavior by "~/.bashrc".

For example, try the following.

```
# enable bash-completion
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
```

```
fi

# CD upon exiting MC
. /usr/lib/mc/mc.sh

# set CDPATH to a good one
CDPATH=.:/usr/share/doc:~:~/Desktop:~
export CDPATH

PATH="${PATH+$PATH:}/usr/sbin:/sbin"
# set PATH so it includes user's private bin if it exists
if [ -d ~/bin ] ; then
  PATH="~/bin${PATH+:$PATH}"
fi
export PATH

EDITOR=vim
export EDITOR
```

Tip
You can find more `bash` customization tips, such as Section 9.2.7, in Chapter 9.

Tip
The `bash-completion` package enables programmable completion for `bash`.

1.4.3   Special key strokes

In the Unix-like environment, there are few key strokes which have special meanings. Please note that on a normal Linux character console, only the left-hand `Ctrl` and `Alt` keys work as expected. Here are few notable key strokes to remember.

key	description of key binding
`Ctrl-U`	erase line before cursor
`Ctrl-H`	erase a character before cursor
`Ctrl-D`	terminate input (exit shell if you are using shell)
`Ctrl-C`	terminate a running program
`Ctrl-Z`	temporarily stop program by moving it to the background job
`Ctrl-S`	halt output to screen
`Ctrl-Q`	reactivate output to screen
`Ctrl-Alt-Del`	reboot/halt the system, see inittab(5)
`Left-Alt-key` (optionally, `Windows-key`)	meta-key for Emacs and the similar UI
`Up-arrow`	start command history search under `bash`
`Ctrl-R`	start incremental command history search under `bash`
`Tab`	complete input of the filename to the command line under `bash`
`Ctrl-V Tab`	input `Tab` without expansion to the command line under `bash`

Table 1.14: List of key bindings for bash

Tip
The terminal feature of `Ctrl-S` can be disabled using stty(1).

1.4.4   Unix style mouse operations

Unix style mouse operations are based on the 3 button mouse system.

action	response
Left-click-and-drag mouse	select and copy to the clipboard
Left-click	select the start of selection
Right-click	select the end of selection and copy to the clipboard
Middle-click	paste clipboard at the cursor

Table 1.15: List of Unix style mouse operations

The center wheel on the modern wheel mouse is considered middle mouse button and can be used for middle-click. Clicking left and right mouse buttons together serves as the middle-click under the 2 button mouse system situation. In order to use a mouse in Linux character consoles, you need to have gpm(8) running as daemon.

1.4.5   The pager

The less(1) command is the enhanced pager (file content browser). It reads the file specified by its command argument or its standard input. Hit "h" if you need help while browsing with the `less` command. It can do much more than more(1) and can be supercharged by executing "`eval $(lesspipe)`" or "`eval $(lessfile)`" in the shell startup script. See more in "`/usr/share/doc/lessf/LESSOPEN`". The "`-R`" option allows raw character output and enables ANSI color escape sequences. See less(1).

1.4.6   The text editor

You should become proficient in one of variants of Vim or Emacs programs which are popular in the Unix-like system.

I think getting used to Vim commands is the right thing to do, since Vi-editor is always there in the Linux/Unix world. (Actually, original `vi` or new `nvi` are programs you find everywhere. I chose Vim instead for newbie since it offers you help through `F1` key while it is similar enough and more powerful.)

If you chose either Emacs or XEmacs instead as your choice of the editor, that is another good choice indeed, particularly for programming. Emacs has a plethora of other features as well, including functioning as a newsreader, directory editor, mail program, etc. When used for programming or editing shell scripts, it intelligently recognizes the format of what you are working on, and tries to provide assistance. Some people maintain that the only program they need on Linux is Emacs. Ten minutes learning Emacs now can save hours later. Having the GNU Emacs manual for reference when learning Emacs is highly recommended.

All these programs usually come with tutoring program for you to learn them by practice. Start Vim by typing "`vim`" and press F1-key. You should at least read the first 35 lines. Then do the online training course by moving cursor to "`|tutor|`" and pressing `Ctrl-]`.

Note
Good editors, such as Vim and Emacs, can handle UTF-8 and other exotic encoding texts correctly. It is a good idea to use the X environment in the UTF-8 locale and to install required programs and fonts to it. Editors have options to set the file encoding independent of the X environment. Please refer to their documentation on multibyte text.

1.4.7   Setting a default text editor

Debian comes with a number of different editors. We recommend to install the `vim` package, as mentioned above.

Debian provides unified access to the system default editor via command "`/usr/bin/editor`" so other programs (e.g., reportbug(1)) can invoke it. You can change it by the following.

```
$ sudo update-alternatives --config editor
```

The choice "/usr/bin/vim.basic" over "/usr/bin/vim.tiny" is my recommendation for newbies since it supports syntax highlighting.

Tip
Many programs use the environment variables "$EDITOR" or "$VISUAL" to decide which editor to use (see Section 1.3.5 and Section 9.3.11). For the consistency on the Debian system, set these to "/usr/bin/editor". (Historically, "$EDITOR" was "ed" and "$VISUAL" was "vi".)

1.4.8 Customizing vim

You can customize vim(1) behavior by "~/.vimrc".

For example, try the following

```
" -------------------------------
" Local configuration
"
set nocompatible
set nopaste
set pastetoggle=<f2>
syn on
if $USER == "root"
 set nomodeline
 set noswapfile
else
 set modeline
 set swapfile
endif
" filler to avoid the line above being recognized as a modeline
" filler
" filler
```

1.4.9 Recording the shell activities

The output of the shell command may roll off your screen and may be lost forever. It is a good practice to log shell activities into the file for you to review them later. This kind of record is essential when you perform any system administration tasks.

The basic method of recording the shell activity is to run it under script(1).

For example, try the following

```
$ script
Script started, file is typescript
```

Do whatever shell commands under script.

Press Ctrl-D to exit script.

```
$ vim typescript
```

See Section 9.2.3 .

1.4.10   Basic Unix commands

Let's learn basic Unix commands. Here I use "Unix" in its generic sense. Any Unix clone OSs usually offer equivalent commands. The Debian system is no exception. Do not worry if some commands do not work as you wish now. If `alias` is used in the shell, its corresponding command outputs are different. These examples are not meant to be executed in this order.

Try all following commands from the non-privileged user account.

Note
Unix has a tradition to hide filenames which start with ".". They are traditionally files that contain configuration information and user preferences.

Note
For `cd` command, see builtins(7).

Note
The default pager of the bare bone Debian system is more(1) which cannot scroll back. By installing the `less` package using command line "`apt-get install less`", less(1) becomes default pager and you can scroll back with cursor keys.

Note
The "[" and "]" in the regular expression of the "`ps aux | grep -e "[e]xim4*"`" command above enable `grep` to avoid matching itself. The "4*" in the regular expression means 0 or more repeats of character "4" thus enables `grep` to match both "exim" and "exim4". Although "*" is used in the shell filename glob and the regular expression, their meanings are different. Learn the regular expression from grep(1).

Please traverse directories and peek into the system using the above commands as training. If you have questions on any of console commands, please make sure to read the manual page.

For example, try the following

```
$ man man
$ man bash
$ man builtins
$ man grep
$ man ls
```

The style of man pages may be a little hard to get used to, because they are rather terse, particularly the older, very traditional ones. But once you get used to it, you come to appreciate their succinctness.

Please note that many Unix-like commands including ones from GNU and BSD display brief help information if you invoke them in one of the following ways (or without any arguments in some cases).

```
$ <commandname> --help
$ <commandname> -h
```

1.5   The simple shell command

Now you have some feel on how to use the Debian system. Let's look deep into the mechanism of the command execution in the Debian system. Here, I have simplified reality for the newbie. See bash(1) for the exact explanation.

A simple command is a sequence of components.

command	description
pwd	display name of current/working directory
whoami	display current user name
id	display current user identity (name, uid, gid, and associated groups)
file \<foo\>	display a type of file for the file "\<foo\>"
type -p \<commandname\>	display a file location of command "\<commandname\>"
which \<commandname\>	, ,
type \<commandname\>	display information on command "\<commandname\>"
apropos \<key-word\>	find commands related to "\<key-word\>"
man -k \<key-word\>	, ,
whatis \<commandname\>	display one line explanation on command "\<commandname\>"
man -a \<commandname\>	display explanation on command "\<commandname\>" (Unix style)
info \<commandname\>	display rather long explanation on command "\<commandname\>" (GNU style)
ls	list contents of directory (non-dot files and directories)
ls -a	list contents of directory (all files and directories)
ls -A	list contents of directory (almost all files and directories, i.e., skip ".." and ".")
ls -la	list all contents of directory with detail information
ls -lai	list all contents of directory with inode number and detail information
ls -d	list all directories under the current directory
tree	display file tree contents
lsof \<foo\>	list open status of file "\<foo\>"
lsof -p \<pid\>	list files opened by the process ID: "\<pid\>"
mkdir \<foo\>	make a new directory "\<foo\>" in the current directory
rmdir \<foo\>	remove a directory "\<foo\>" in the current directory
cd \<foo\>	change directory to the directory "\<foo\>" in the current directory or in the directory listed in the variable "$CDPATH"
cd /	change directory to the root directory
cd	change directory to the current user's home directory
cd /\<foo\>	change directory to the absolute path directory "/\<foo\>"
cd ..	change directory to the parent directory
cd ~\<foo\>	change directory to the home directory of the user "\<foo\>"
cd -	change directory to the previous directory
\</etc/motd pager	display contents of "/etc/motd" using the default pager
touch \<junkfile\>	create a empty file "\<junkfile\>"
cp \<foo\> \<bar\>	copy a existing file "\<foo\>" to a new file "\<bar\>"
rm \<junkfile\>	remove a file "\<junkfile\>"
mv \<foo\> \<bar\>	rename an existing file "\<foo\>" to a new name "\<bar\>" ("\<bar\>" must not exist)
mv \<foo\> \<bar\>	move an existing file "\<foo\>" to a new location "\<bar\>/\<foo\>" (the directory "\<bar\>" must exist)
mv \<foo\> \<bar\>/\<baz\>	move an existing file "\<foo\>" to a new location with a new name "\<bar\>/\<baz\>" (the directory "\<bar\>" must exist but the directory "\<bar\>/\<baz\>" must not exist)
chmod 600 \<foo\>	make an existing file "\<foo\>" to be non-readable and non-writable by the other people (non-executable for all)
chmod 644 \<foo\>	make an existing file "\<foo\>" to be readable but non-writable by the other people (non-executable for all)
chmod 755 \<foo\>	make an existing file "\<foo\>" to be readable but non-writable by the other people (executable for all)
find . -name \<pattern\>	find matching filenames using shell "\<pattern\>" (slower)
locate -d . \<pattern\>	find matching filenames using shell "\<pattern\>" (quicker using regularly generated database)
grep -e "\<pattern\>" *.html	find a "\<pattern\>" in all files ending with ".html" in current directory and display them all
top	display process information using full screen, type "q" to quit
ps aux | pager	display information on all the running processes using BSD style output
ps -ef | pager	display information on all the running processes using Unix system-V style output
ps aux | grep -e "[e]xim4*"	display all processes running "exim" and "exim4"
ps axf | pager	display information on all the running processes with ASCII art output
kill \<1234\>	kill a process identified by the process ID: "\<1234\>"

1. Variable assignments (optional)

2. Command name

3. Arguments (optional)

4. Redirections (optional: `>` , `>>` , `<` , `<<` , etc.)

5. Control operator (optional: `&&` , `||` , <newline> , `;` , `&` , `(` , `)`)

1.5.1 Command execution and environment variable

The values of some environment variables change the behavior of some Unix commands.

Default values of environment variables are initially set by the PAM system and then some of them may be reset by some application programs.

- The display manager such as `gdm3` resets environment variables.

- The shell in its start up codes resets environment variables in "`~/.bash_profile`" and "`~/.bashrc`".

1.5.2 The "$LANG" variable

The full locale value given to "`$LANG`" variable consists of 3 parts: "`xx_YY.ZZZZ`".

locale value	meaning
xx	ISO 639 language codes (lower case) such as "en"
YY	ISO 3166 country codes (upper case) such as "US"
ZZZZ	codeset, always set to "UTF-8"

Table 1.17: The 3 parts of locale value

For language codes and country codes, see pertinent description in the "`info gettext`".

For the codeset on the modern Debian system, you should always set it to **UTF-8** unless you specifically want to use the historic one with good reason and background knowledge.

For fine details of the locale configuration, see Section 8.4.

Note
The "LANG=en_US" is not "LANG=C" nor "LANG=en_US.UTF-8". It is "LANG=en_US.ISO-8859-1" (see Section 8.4.1).

Typical command execution uses a shell line sequence as the following.

```
$ date
Sun Jun  3 10:27:39 JST 2007
$ LANG=fr_FR.UTF-8 date
dimanche 3 juin 2007, 10:27:33 (UTC+0900)
```

Here, the program date(1) is executed with different values of the environment variable "$LANG".

- For the first command, "$LANG" is set to the system default locale value "`en_US.UTF-8`".

- For the second command, "$LANG" is set to the French UTF-8 locale value "`fr_FR.UTF-8`".

locale recommendation	Language (area)
en_US.UTF-8	English (USA)
en_GB.UTF-8	English (Great Britain)
fr_FR.UTF-8	French (France)
de_DE.UTF-8	German (Germany)
it_IT.UTF-8	Italian (Italy)
es_ES.UTF-8	Spanish (Spain)
ca_ES.UTF-8	Catalan (Spain)
sv_SE.UTF-8	Swedish (Sweden)
pt_BR.UTF-8	Portuguese (Brazil)
ru_RU.UTF-8	Russian (Russia)
zh_CN.UTF-8	Chinese (P.R. of China)
zh_TW.UTF-8	Chinese (Taiwan R.O.C.)
ja_JP.UTF-8	Japanese (Japan)
ko_KR.UTF-8	Korean (Republic of Korea)
vi_VN.UTF-8	Vietnamese (Vietnam)

Table 1.18: List of locale recommendations

Most command executions usually do not have preceding environment variable definition. For the above example, you can alternatively execute as the following.

```
$ LANG=fr_FR.UTF-8
$ date
dimanche 3 juin 2007, 10:27:33 (UTC+0900)
```

As you can see here, the output of command is affected by the environment variable to produce French output. If you want the environment variable to be inherited to subprocesses (e.g., when calling shell script), you need to **export** it instead by the following.

```
$ export LANG
```

Note
When you use a typical console terminal, the "$LANG" environment variable is usually set to be **exported** by the desktop environment. So the above is not really a good example to test the effect of export.

Tip
When filing a bug report, running and checking the command under "LANG=en_US.UTF-8" is a good idea if you use non-English environment.

See locale(5) and locale(7) for "$LANG" and related environment variables.

Note
I recommend you to configure the system environment just by the "$LANG" variable and to stay away from "$LC_*" variables unless it is absolutely needed.

1.5.3 The "$PATH" variable

When you type a command into the shell, the shell searches the command in the list of directories contained in the "$PATH" environment variable. The value of the "$PATH" environment variable is also called the shell's search path.

In the default Debian installation, the "$PATH" environment variable of user accounts may not include "/sbin" and "/usr/sbin". For example, the ifconfig command needs to be issued with full path as "/sbin/ifconfig". (Similar ip command is located in "/bin".)

You can change the "$PATH" environment variable of Bash shell by "~/.bash_profile" or "~/.bashrc" files.

1.5.4 The "$HOME" variable

Many commands stores user specific configuration in the home directory and changes their behavior by their contents. The home directory is identified by the environment variable "$HOME".

value of "$HOME"	program execution situation
/	program run by the init process (daemon)
/root	program run from the normal root shell
/home/<normal_user>	program run from the normal user shell
/home/<normal_user>	program run from the normal user GUI desktop menu
/home/<normal_user>	program run as root with "sudo program"
/root	program run as root with "sudo -H program"

Table 1.19: List of "$HOME" values

Tip
Shell expands "~/" to current user's home directory, i.e., "$HOME/". Shell expands "~foo/" to foo's home directory, i.e., "/home/foo/".

1.5.5 Command line options

Some commands take arguments. Arguments starting with "-" or "--" are called options and control the behavior of the command.

```
$ date
Mon Oct 27 23:02:09 CET 2003
$ date -R
Mon, 27 Oct 2003 23:02:40 +0100
```

Here the command-line argument "-R" changes date(1) behavior to output RFC2822 compliant date string.

1.5.6 Shell glob

Often you want a command to work with a group of files without typing all of them. The filename expansion pattern using the shell **glob**, (sometimes referred as **wildcards**), facilitate this need.

For example, try the following

```
$ mkdir junk; cd junk; touch 1.txt 2.txt 3.c 4.h .5.txt ..6.txt
$ echo *.txt
1.txt 2.txt
$ echo *
1.txt 2.txt 3.c 4.h
$ echo *.[hc]
3.c 4.h
$ echo .*
. .. .5.txt ..6.txt
$ echo .*[^.]*
```

shell glob pattern	description of match rule
*	filename (segment) not started with ".".
.*	filename (segment) started with ".".
?	exactly one character
[···]	exactly one character with any character enclosed in brackets
[a-z]	exactly one character with any character between "a" and "z"
[^···]	exactly one character other than any character enclosed in brackets (excluding "^")

Table 1.20: Shell glob patterns

```
.5.txt ..6.txt
$ echo [^1-3]*
4.h
$ cd ..; rm -rf junk
```

See glob(7).

> **Note**
> Unlike normal filename expansion by the shell, the shell pattern "*" tested in find(1) with "-name" test etc., matches the initial "." of the filename. (New POSIX feature)

> **Note**
> BASH can be tweaked to change its glob behavior with its shopt builtin options such as "dotglob", "noglob", "nocaseglob", "nullglob", "extglob", etc. See bash(1).

1.5.7 Return value of the command

Each command returns its exit status (variable: "$?") as the return value.

command exit status	numeric return value	logical return value
success	zero, 0	**TRUE**
error	non-zero, -1	**FALSE**

Table 1.21: Command exit codes

For example, try the following.

```
$ [ 1 = 1 ] ; echo $?
0
$ [ 1 = 2 ] ; echo $?
1
```

> **Note**
> Please note that, in the logical context for the shell, **success** is treated as the logical **TRUE** which has 0 (zero) as its value. This is somewhat non-intuitive and needs to be reminded here.

command idiom	description
`command &`	**background** execution of `command` in the subshell
`command1	command2`	**pipe** the standard output of `command1` to the standard input of `command2` (**concurrent** execution)
`command1 2>&1	command2`	**pipe** both standard output and standard error of `command1` to the standard input of `command2` (**concurrent** execution)
`command1 ; command2`	execute `command1` and `command2` **sequentially**
`command1 && command2`	execute `command1`; if successful, execute `command2` **sequentially** (return success if both `command1` **and** `command2` are successful)
`command1 || command2`	execute `command1`; if not successful, execute `command2` **sequentially** (return success if `command1` **or** `command2` are successful)
`command > foo`	redirect standard output of `command` to a file `foo` (overwrite)
`command 2> foo`	redirect standard error of `command` to a file `foo` (overwrite)
`command >> foo`	redirect standard output of `command` to a file `foo` (append)
`command 2>> foo`	redirect standard error of `command` to a file `foo` (append)
`command > foo 2>&1`	redirect both standard output and standard error of `command` to a file `foo`
`command < foo`	redirect standard input of `command` to a file `foo`
`command << delimiter`	redirect standard input of `command` to the following lines until "`delimiter`" is met (here document)
`command <<- delimiter`	redirect standard input of `command` to the following lines until "`delimiter`" is met (here document, the leading tab characters are stripped from input lines)

Table 1.22: Shell command idioms

1.5.8   Typical command sequences and shell redirection

Let's try to remember following shell command idioms typed in one line as a part of shell command.

The Debian system is a multi-tasking system. Background jobs allow users to run multiple programs in a single shell. The management of the background process involves the shell builtins: `jobs`, `fg`, `bg`, and `kill`. Please read sections of bash(1) under "SIGNALS", and "JOB CONTROL", and builtins(1).

For example, try the following

```
$ </etc/motd pager
```

```
$ pager </etc/motd
```

```
$ pager /etc/motd
```

```
$ cat /etc/motd | pager
```

Although all 4 examples of shell redirections display the same thing, the last example runs an extra `cat` command and wastes resources with no reason.

The shell allows you to open files using the `exec` builtin with an arbitrary file descriptor.

```
$ echo Hello >foo
$ exec 3<foo 4>bar  # open files
$ cat <&3 >&4       # redirect stdin to 3, stdout to 4
$ exec 3<&- 4>&-    # close files
$ cat bar
Hello
```

The file descriptor 0-2 are predefined.

device	description	file descriptor
stdin	standard input	0
stdout	standard output	1
stderr	standard error	2

Table 1.23: Predefined file descriptors

1.5.9 Command alias

You can set an alias for the frequently used command.

For example, try the following

```
$ alias la='ls -la'
```

Now, "la" works as a short hand for "ls -la" which lists all files in the long listing format.

You can list any existing aliases by alias (see bash(1) under "SHELL BUILTIN COMMANDS").

```
$ alias
...
alias la='ls -la'
```

You can identity exact path or identity of the command by type (see bash(1) under "SHELL BUILTIN COMMANDS").

For example, try the following

```
$ type ls
ls is hashed (/bin/ls)
$ type la
la is aliased to ls -la
$ type echo
echo is a shell builtin
$ type file
file is /usr/bin/file
```

Here ls was recently searched while "file" was not, thus "ls" is "hashed", i.e., the shell has an internal record for the quick access to the location of the "ls" command.

Tip
See Section 9.2.7.

1.6 Unix-like text processing

In Unix-like work environment, text processing is done by piping text through chains of standard text processing tools. This was another crucial Unix innovation.

1.6.1 Unix text tools

There are few standard text processing tools which are used very often on the Unix-like system.

- No regular expression is used:
  - cat(1) concatenates files and outputs the whole content.
  - tac(1) concatenates files and outputs in reverse.

- – cut(1) selects parts of lines and outputs.
    - – head(1) outputs the first part of files.
    - – tail(1) outputs the last part of files.
    - – sort(1) sorts lines of text files.
    - – uniq(1) removes duplicate lines from a sorted file.
    - – tr(1) translates or deletes characters.
    - – diff(1) compares files line by line.

- Basic regular expression (**BRE**) is used:

    - – grep(1) matches text with patterns.
    - – ed(1) is a primitive line editor.
    - – sed(1) is a stream editor.
    - – vim(1) is a screen editor.
    - – emacs(1) is a screen editor. (somewhat extended **BRE**)

- Extended regular expression (**ERE**) is used:

    - – egrep(1) matches text with patterns.
    - – awk(1) does simple text processing.
    - – tcl(3tcl) can do every conceivable text processing: See re_syntax(3). Often used with tk(3tk).
    - – perl(1) can do every conceivable text processing. See perlre(1).
    - – pcregrep(1) from the `pcregrep` package matches text with Perl Compatible Regular Expressions (PCRE) pattern.
    - – python(1) with the `re` module can do every conceivable text processing. See ”`/usr/share/doc/python/html/index.htm`

If you are not sure what exactly these commands do, please use ”`man command`” to figure it out by yourself.

Note
Sort order and range expression are locale dependent. If you wish to obtain traditional behavior for a command, use **C** locale instead of **UTF-8** ones by prepending command with ”`LANG=C`” (see Section 1.5.2 and Section 8.4).

Note
Perl regular expressions (perlre(1)), Perl Compatible Regular Expressions (PCRE), and Python regular expressions offered by the `re` module have many common extensions to the normal **ERE**.

1.6.2   Regular expressions

Regular expressions are used in many text processing tools. They are analogous to the shell globs, but they are more complicated and powerful.

The regular expression describes the matching pattern and is made up of text characters and **metacharacters**.

A **metacharacter** is just a character with a special meaning. There are 2 major styles, **BRE** and **ERE**, depending on the text tools as described above.

The regular expression of **emacs** is basically **BRE** but has been extended to treat ”**+**”and ”**?**” as the **metacharacters** as in **ERE**. Thus, there are no needs to escape them with ”\” in the regular expression of emacs.

grep(1) can be used to perform the text search using the regular expression.

For example, try the following

BRE	ERE	description of the regular expression
`\ . [ ] ^ $ *`	`\ . [ ] ^ $ *`	common **metacharacters**
`\+ \? \( \) \{ \} |`		BRE only "\" escaped **metacharacters**
	`+ ? ( ) { } |`	ERE only non-"\" escaped **metacharacters**
`c`	`c`	match **non-metacharacter** "`c`"
`\c`	`\c`	match a literal character "`c`" even if "`c`" is **metacharacter** by itself
`.`	`.`	match any character including newline
`^`	`^`	position at the beginning of a string
`$`	`$`	position at the end of a string
`\<`	`\<`	position at the beginning of a word
`\>`	`\>`	position at the end of a word
`[abc⋯]`	`[abc⋯]`	match any characters in "`abc⋯`"
`[^abc⋯]`	`[^abc⋯]`	match any characters except in "`abc⋯`"
`r*`	`r*`	match zero or more regular expressions identified by "`r`"
`r\+`	`r+`	match one or more regular expressions identified by "`r`"
`r\?`	`r?`	match zero or one regular expressions identified by "`r`"
`r1|r2`	`r1|r2`	match one of the regular expressions identified by "`r1`" or "`r2`"
`\(r1|r2\)`	`(r1|r2)`	match one of the regular expressions identified by "`r1`" or "`r2`" and treat it as a **bracketed** regular expression

Table 1.24: Metacharacters for BRE and ERE

```
$ egrep 'GNU.*LICENSE|Yoyodyne' /usr/share/common-licenses/GPL
GNU GENERAL PUBLIC LICENSE
GNU GENERAL PUBLIC LICENSE
Yoyodyne, Inc., hereby disclaims all copyright interest in the program
```

Tip
See Section 9.2.7.

1.6.3 Replacement expressions

For the replacement expression, some characters have special meanings.

replacement expression	description of the text to replace the replacement expression
`&`	what the regular expression matched (use `\&` in emacs)
`\n`	what the n-th **bracketed** regular expression matched ("n" being number)

Table 1.25: The replacement expression

For Perl replacement string, "$&" is used instead of "&" and "$n" is used instead of "\n".

For example, try the following

```
$ echo zzz1abc2efg3hij4 | \
sed -e 's/\(1[a-z]*\)[0-9]*\(.*\)$/=&=/'
```

```
zzz=1abc2efg3hij4=
$ echo zzz1abc2efg3hij4 | \
sed -e 's/\(1[a-z]*\)[0-9]*\(.*\)$/\2===\1/'
zzzefg3hij4===1abc
$ echo zzz1abc2efg3hij4 | \
perl -pe 's/(1[a-z]*)[0-9]*(.*)$/$2===$1/'
zzzefg3hij4===1abc
$ echo zzz1abc2efg3hij4 | \
perl -pe 's/(1[a-z]*)[0-9]*(.*)$/=$&=/'
zzz=1abc2efg3hij4=
```

Here please pay extra attention to the style of the **bracketed** regular expression and how the matched strings are used in the text replacement process on different tools.

These regular expressions can be used for cursor movements and text replacement actions in some editors too.

The back slash "\" at the end of line in the shell commandline escapes newline as a white space character and continues shell command line input to the next line.

Please read all the related manual pages to learn these commands.

1.6.4 Global substitution with regular expressions

The ed(1) command can replace all instances of "FROM_REGEX" with "TO_TEXT" in "file".

```
$ ed file <<EOF
,s/FROM_REGEX/TO_TEXT/g
w
q
EOF
```

The sed(1) command can replace all instances of "FROM_REGEX" with "TO_TEXT" in "file".

```
$ sed -i -e 's/FROM_REGEX/TO_TEXT/g' file
```

The vim(1) command can replace all instances of "FROM_REGEX" with "TO_TEXT" in "file" by using ex(1) commands.

```
$ vim '+%s/FROM_REGEX/TO_TEXT/gc' '+w' '+q' file
```

Tip
The "c" flag in the above ensures interactive confirmation for each substitution.

Multiple files ("file1", "file2", and "file3") can be processed with regular expressions similarly with vim(1) or perl(1).

```
$ vim '+argdo %s/FROM_REGEX/TO_TEXT/ge|update' '+q' file1 file2 file3
```

Tip
The "e" flag in the above prevents the "No match" error from breaking a mapping.

```
$ perl -i -p -e 's/FROM_REGEX/TO_TEXT/g;' file1 file2 file3
```

In the perl(1) example, "-i" is for the in-place editing of each target file, and "-p" is for the implicit loop over all given files.

Tip

Use of argument "`-i.bak`" instead of "`-i`" keeps each original file by adding "`.bak`" to its filename. This makes recovery from errors easier for complex substitutions.

Note

ed(1) and vim(1) are **BRE**; perl(1) is **ERE**.

1.6.5 Extracting data from text file table

Let's consider a text file called "DPL" in which some pre-2004 Debian project leader's names and their initiation date are listed in a space-separated format.

```
Ian     Murdock    August  1993
Bruce   Perens     April   1996
Ian     Jackson    January 1998
Wichert Akkerman   January 1999
Ben     Collins    April   2001
Bdale   Garbee     April   2002
Martin  Michlmayr  March   2003
```

Tip

See "A Brief History of Debian" for the latest Debian leadership history.

Awk is frequently used to extract data from these types of files.

For example, try the following

```
$ awk '{ print $3 }' <DPL                 # month started
August
April
January
January
April
April
March
$ awk '($1=="Ian") { print }' <DPL        # DPL called Ian
Ian     Murdock    August  1993
Ian     Jackson    January 1998
$ awk '($2=="Perens") { print $3,$4 }' <DPL # When Perens started
April 1996
```

Shells such as Bash can be also used to parse this kind of file.

For example, try the following

```
$ while read first last month year; do
    echo $month
  done <DPL
... same output as the first Awk example
```

Here, the `read` builtin command uses characters in "`$IFS`" (internal field separators) to split lines into words.

If you change "`$IFS`" to "`:`", you can parse "`/etc/passwd`" with shell nicely.

```
$ oldIFS="$IFS"   # save old value
$ IFS=':'
$ while read user password uid gid rest_of_line; do
    if [ "$user" = "bozo" ]; then
      echo "$user's ID is $uid"
    fi
  done < /etc/passwd
bozo's ID is 1000
$ IFS="$oldIFS"   # restore old value
```

(If Awk is used to do the equivalent, use "FS=':'" to set the field separator.)

IFS is also used by the shell to split results of parameter expansion, command substitution, and arithmetic expansion. These do not occur within double or single quoted words. The default value of IFS is <space>, <tab>, and <newline> combined.

Be careful about using this shell IFS tricks. Strange things may happen, when shell interprets some parts of the script as its **input**.

```
$ IFS=":,"                         # use ":" and "," as IFS
$ echo IFS=$IFS,   IFS="$IFS"      # echo is a Bash builtin
IFS=  , IFS=:,
$ date -R                          # just a command output
Sat, 23 Aug 2003 08:30:15 +0200
$ echo $(date -R)                  # sub shell --> input to main shell
Sat  23 Aug 2003 08 30 36 +0200
$ unset IFS                        # reset IFS to the default
$ echo $(date -R)
Sat, 23 Aug 2003 08:30:50 +0200
```

1.6.6 Script snippets for piping commands

The following scripts do nice things as a part of a pipe.

script snippet (type in one line)	effect of command
`find /usr -print`	find all files under "/usr"
`seq 1 100`	print 1 to 100
`| xargs -n 1 <command>`	run command repeatedly with each item from pipe as its argument
`| xargs -n 1 echo`	split white-space-separated items from pipe into lines
`| xargs echo`	merge all lines from pipe into a line
`| grep -e <regex_pattern>`	extract lines from pipe containing <regex_pattern>
`| grep -v -e <regex_pattern>`	extract lines from pipe not containing <regex_pattern>
`| cut -d: -f3 -`	extract third field from pipe separated by ":" (passwd file etc.)
`| awk '{ print $3 }'`	extract third field from pipe separated by whitespaces
`| awk -F'\t' '{ print $3 }'`	extract third field from pipe separated by tab
`| col -bx`	remove backspace and expand tabs to spaces
`| expand -`	expand tabs
`| sort| uniq`	sort and remove duplicates
`| tr 'A-Z' 'a-z'`	convert uppercase to lowercase
`| tr -d '\n'`	concatenate lines into one line
`| tr -d '\r'`	remove CR
`| sed 's/^/# /'`	add "#" to the start of each line
`| sed 's/\.ext//g'`	remove ".ext"
`| sed -n -e 2p`	print the second line
`| head -n 2 -`	print the first 2 lines
`| tail -n 2 -`	print the last 2 lines

Table 1.26: List of script snippets for piping commands

A one-line shell script can loop over many files using find(1) and xargs(1) to perform quite complicated tasks. See Section 10.1.5 and Section 9.3.9.

When using the shell interactive mode becomes too complicated, please consider to write a shell script (see Section 12.1).

Chapter 2

Debian package management

Note
This chapter is written assuming the latest stable release is codename: `stretch`.

Debian is a volunteer organization which builds **consistent** distributions of pre-compiled binary packages of free software and distributes them from its archive.

The Debian archive is offered by many remote mirror sites for access through HTTP and FTP methods. It is also available as CD-ROM/DVD.

The Debian package management system, **when used properly**, offers the user to install **consistent sets of binary packages** to the system from the archive. Currently, there are 58227 packages available for the amd64 architecture.

The Debian package management system has a rich history and many choices for the front end user program and back end archive access method to be used. Currently, we recommend the following.

- apt(8) for all interactive command line operations, including package installation, removal and dist-upgrades. Available since Debian Jessie (Debian 8).

- apt-get(8) for calling Debian package management system from scripts. It is also a fallback option when `apt` is not available (often with older Debian systems).

- aptitude(8) for an interactive text interface to manage the installed packages and to search the available packages.

2.1 Debian package management prerequisites

2.1.1 Package configuration

Here are some key points for package configuration on the Debian system.

- The manual configuration by the system administrator is respected. In other words, the package configuration system makes no intrusive configuration for the sake of convenience.

- Each package comes with its own configuration script with standardized user interface called debconf(7) to help initial installation process of the package.

- Debian Developers try their best to make your upgrade experience flawless with package configuration scripts.

- Full functionalities of packaged software are available to the system administrator. But ones with security risks are disabled in the default installation.

package	popcon	size	description
apt	V:870, I:999	3755	Advanced Packaging Tool (APT), front-end for dpkg providing "http", "ftp", and "file" archive access methods (apt/apt-get/apt-cache commands included)
aptitude	V:126, I:685	4217	interactive terminal-based package manager with aptitude(8)
tasksel	V:38, I:973	375	tool for selecting tasks for installation on the Debian system (front-end for APT)
unattended-upgrades	V:244, I:399	252	enhancement package for APT to enable automatic installation of security upgrades
dselect	V:4, I:52	2506	terminal-based package manager (previous standard, front-end for APT and other old access methods)
dpkg	V:931, I:999	6743	package management system for Debian
synaptic	V:71, I:454	7793	graphical package manager (GNOME front-end for APT)
apt-utils	V:405, I:998	1113	APT utility programs: apt-extracttemplates(1), apt-ftparchive(1), and apt-sortpkgs(1)
apt-listchanges	V:379, I:843	385	package change history notification tool
apt-listbugs	V:8, I:12	449	lists critical bugs before each APT installation
apt-file	V:12, I:78	82	APT package searching utility —command-line interface
apt-rdepends	V:0, I:6	40	recursively lists package dependencies

Table 2.1: List of Debian package management tools

- If you manually activate a service with some security risks, you are responsible for the risk containment.

- Esoteric configuration may be manually enabled by the system administrator. This may create interference with popular generic helper programs for the system configuration.

2.1.2 Basic precautions

⚠ **Warning**
Do not install packages from random mixture of suites. It probably breaks the package consistency which requires deep system management knowledge, such as compiler ABI, library version, interpreter features, etc.

The newbie Debian system administrator should stay with the **stable** release of Debian while applying only security updates. I mean that some of the following valid actions are better avoided, as a precaution, until you understand the Debian system very well. Here are some reminders.

- Do not include **testing** or **unstable** in "/etc/apt/sources.list".

- Do not mix standard Debian with other non-Debian archives such as Ubuntu in "/etc/apt/sources.list".

- Do not create "/etc/apt/preferences".

- Do not change default behavior of package management tools through configuration files without knowing their full impacts.

- Do not install random packages by "dpkg -i <random_package>".

- Do not ever install random packages by "dpkg --force-all -i <random_package>".

- Do not erase or alter files in "/var/lib/dpkg/".

- Do not overwrite system files by installing software programs directly compiled from source.

  – Install them into "/usr/local" or "/opt", if needed.

The non-compatible effects caused by above actions to the Debian package management system may leave your system unusable.

The serious Debian system administrator who runs mission critical servers, should use extra precautions.

- Do not install any packages including security updates from Debian without thoroughly testing them with your particular configuration under safe conditions.

  - You as the system administrator are responsible for your system in the end.
  - The long stability history of the Debian system is no guarantee by itself.

2.1.3 Life with eternal upgrades

Despite my warnings above, I know many readers of this document wish to run the `testing` or `unstable` suites of Debian as their main system for **self-administered Desktop environments**. This is because they work very well, are updated frequently, and offer the latest features.

> ⚠️ **Caution**
> For your **production server**, the `stable` suite with the security updates is recommended. The same can be said for desktop PCs on which you can spend limited administration efforts, e.g. for your mother's PC.

It takes no more than simply setting the distribution string in the "`/etc/apt/sources.list`" to the suite name: "`testing`" or "`unstable`"; or the codename: "`buster`" or "`sid`". This makes you live **the life of eternal upgrades**.

The use of `testing` or `unstable` is **a lot of fun** but comes with some risks. Even though the `unstable` suite of the Debian system looks very stable for most of the times, there have been some package problems on the `testing` and `unstable` suite of the Debian system and a few of them were not so trivial to resolve. It may be **quite painful** for you. Sometimes, you may have a broken package or missing functionality for a few weeks.

Here are some ideas to ensure quick and easy recovery from bugs in Debian packages.

- Make the system **dual bootable** by installing the `stable` suite of the Debian system to another partition

- Make the installation CD handy for the **rescue boot**

- Consider installing `apt-listbugs` to check the Debian Bug Tracking System (BTS) information before the upgrade

- Learn the package system infrastructure enough to work around the problem

- Create a chroot or similar environment and run the latest system in it in advance (see Section 9.10)

(If you can not do any one of these precautionary actions, you are probably not ready for the `testing` and `unstable` suites.)

Enlightenment with the following saves a person from the eternal karmic struggle of upgrade hell and let him reach Debian nirvana.

2.1.4 Debian archive basics

Let's look into the Debian archive from a system user's perspective.

> 💡 **Tip**
> Official policy of the Debian archive is defined at Debian Policy Manual, Chapter 2 - The Debian Archive.

For the typical HTTP access, the archive is specified in the "`/etc/apt/sources.list`" file as the following, e.g. for the current `stable` = `stretch` system.

```
deb http://deb.debian.org/debian/ stretch main
deb-src http://deb.debian.org/debian/ stretch main

deb http://security.debian.org/ stretch/updates main
deb-src http://security.debian.org/ stretch/updates main
```

Here, I tend to use codename "stretch" instead of suite name "stable" to avoid surprises when the next stable is released.

The meaning of "/etc/apt/sources.list" is described in sources.list(5) and key points are followings.

• The "deb" line defines for the binary packages.

• The "deb-src" line defines for the source packages.

• The 1st argument is the root URL of the Debian archive.

• The 2nd argument is the distribution name: either the suite name or the codename.

• The 3rd and following arguments are the list of valid archive area names of the Debian archive.

The "deb-src" lines can safely be omitted (or commented out by placing "#" at the start of the line) if it is just for aptitude which does not access source related meta data. It speeds up the updates of the archive meta data. The URL can be "http://", "ftp://", "file://", ···.

Tip
If "sid" is used in the above example instead of "stretch", the "deb: http://security.debian.org/ ···" line for security updates in the "/etc/apt/sources.list" is not required. This is because there is no security update archive for "sid" (unstable).

Here is the list of URL of the Debian archive sites and suite name or codename used in the configuration file.

archive URL	suite name (codename)	purpose
http://deb.debian.org/debian/	stable (stretch)	stable (stretch) release
http://deb.debian.org/debian/	testing (buster)	testing (buster) release
http://deb.debian.org/debian/	unstable (sid)	unstable (sid) release
http://deb.debian.org/debian/	experimental	experimental pre-release (optional, only for developer)
http://deb.debian.org/debian/	stable-proposed-updates	Updates for the next stable point release (optional)
http://security.debian.org/	stable/updates	security updates for stable release (important)
http://security.debian.org/	testing/updates	security updates for testing release (important)
http://deb.debian.org/debian/	stretch-updates	compatible updates for spam filter, IM clients, etc. for stretch
http://deb.debian.org/debian/	stretch-backports	newer backported packages for stretch (optional)

Table 2.2: List of Debian archive sites

Caution
Only pure **stable** release with security updates provides the best stability. Running mostly **stable** release mixed with some packages from **testing** or **unstable** release is riskier than running pure **unstable** release for library version mismatch etc. If you really need the latest version of some programs under **stable** release, please use packages from stretch-updates and http://backports.debian.org (see Section 2.7.4) services. These services must be used with extra care.

Caution

You should basically list only one of `stable`, `testing`, or `unstable` suites in the "deb" line. If you list any combination of `stable`, `testing`, and `unstable` suites in the "deb" line, APT programs slow down while only the latest archive is effective. Multiple listing makes sense for these when the "/etc/apt/preferences" file is used with clear objectives (see Section 2.7.3).

Tip

For the Debian system with the `stable` and `testing` suites, it is a good idea to include lines with "http://security.debian.org/" in the "/etc/apt/sources.list" to enable security updates as in the example above.

Note

The security bugs for the `stable` archive are fixed by the Debian security team. This activity has been quite rigorous and reliable. Those for the `testing` archive may be fixed by the Debian testing security team. For several reasons, this activity is not as rigorous as that for `stable` and you may need to wait for the migration of fixed `unstable` packages. Those for the `unstable` archive are fixed by the individual maintainer. Actively maintained `unstable` packages are usually in a fairly good shape by leveraging latest upstream security fixes. See Debian security FAQ for how Debian handles security bugs.

---

area	number of packages	criteria of package component
`main`	57346	DFSG compliant

Table 2.3: List of Debian archive area

Here the number of packages in the above is for the amd64 architecture. The `main` area provides the Debian system (see Section 2.1.5).

The Debian archive organization can be studied best by pointing your browser to the each archive URL appended with `dists` or `pool`.

The distribution is referred by two ways, the suite or codename. The word distribution is alternatively used as the synonym to the suite in many documentations. The relationship between the suite and the codename can be summarized as the following.

Timing	suite = `stable`	suite = `testing`	suite = `unstable`
after the `stretch` release	codename = `stretch`	codename = `buster`	codename = `sid`
after the `buster` release	codename = `buster`	codename = `bullseye`	codename = `sid`

Table 2.4: The relationship between suite and codename

The history of codenames are described in Debian FAQ: 6.2.1 Which other codenames have been used in the past?

In the stricter Debian archive terminology, the word "section" is specifically used for the categorization of packages by the application area. (Although, the word "main section" may sometimes be used to describe the Debian archive area named as "main".)

Every time a new upload is done by a Debian developer (DD) to the `unstable` archive (via incoming processing), the DD is required to ensure uploaded packages to be compatible with the latest set of packages in the latest `unstable` archive.

If DD breaks this compatibility intentionally for important library upgrade etc, there is usually announcement to the debian-devel mailing list etc.

Before a set of packages are moved by the Debian archive maintenance script from the `unstable` archive to the `testing` archive, the archive maintenance script not only checks the maturity (about 10 days old) and the status of the RC bug reports for

the packages but also tries to ensure them to be compatible with the latest set of packages in the `testing` archive. This process makes the `testing` archive very current and usable.

Through the gradual archive freeze process led by the release team, the `testing` archive is matured to make it completely consistent and bug free with some manual interventions. Then the new `stable` release is created by assigning the codename for the old `testing` archive to the new `stable` archive and creating the new codename for the new `testing` archive. The initial contents of the new `testing` archive is exactly the same as that of the newly released `stable` archive.

Both the `unstable` and the `testing` archives may suffer temporary glitches due to several factors.

- Broken package upload to the archive (mostly for `unstable`)

- Delay of accepting the new packages to the archive (mostly for `unstable`)

- Archive synchronization timing issue (both for `testing` and `unstable`)

- Manual intervention to the archive such as package removal (more for `testing`) etc.

So if you ever decide to use these archives, you should be able to fix or work around these kinds of glitches.

Caution

For about few months after a new `stable` release, most desktop users should use the `stable` archive with its security updates even if they usually use `unstable` or `testing` archives. For this transition period, both `unstable` and `testing` archives are not good for most people. Your system is difficult to keep in good working condition with the `unstable` archive since it suffers surges of major upgrades for core packages. The `testing` archive is not useful either since it contains mostly the same content as the `stable` archive without its security support (Debian testing-security-announce 2008-12). After a month or so, the `unstable` archive may be usable if you are careful.

Tip

When tracking the `testing` archive, a problem caused by a removed package is usually worked around by installing corresponding package from the `unstable` archive which is uploaded for bug fix.

See Debian Policy Manual for archive definitions.

- ”Sections”

- ”Priorities”

- ”Base system”

- ”Essential packages”

2.1.5 Debian is 100% free software

Debian is 100% free software because of the followings:

- Debian installs only free software by default to respect user's freedoms.

- Debian provides only free software in `main`.

- Debian recommends running only free software from `main`.

Some people wonder if the following 2 facts contradict or not.

- ”Debian will remain 100% free”. (First term of Debian Social Contract)

These do not contradict, because of the followings.

- The Debian system is 100% free and its packages are hosted by Debian servers in the `main` area.

These are precisely explained in the 4th and 5th terms of Debian Social Contract:

- Our priorities are our users and free software
- Works that do not meet our free software standards
- lack of freedom for such software packages
- lack of support from Debian on such software packages (Debian can't support software properly without having access to its source code.)
- contamination of your 100% free Debian system

The Debian Free Software Guidelines are the free software standards for Debian. Debian interprets "software" in the widest scope including document, firmware, logo, and artwork data in the package. This makes Debian's free software standards very strict ones.

In order to meet this strict free software standards required for `main`, Debian once unbranded Mozilla software packages such as Firefox, Thunderbird, and Seamonkey by removing their logo and some artwork data; and ships them as Iceweasel, Icedove, and Iceape, respectively. Those packages have restored their original names with the release of Debian Stretch (Debian 9) after those problems get solved.

- Game and font packages with restriction on commercial use and/or content modification.

2.1.6 Package dependencies

The Debian system offers a consistent set of binary packages through its versioned binary dependency declaration mechanism in the control file fields. Here is a bit over simplified definition for them.

- "Depends"

  - This declares an absolute dependency and all of the packages listed in this field must be installed at the same time or in advance.

- "Pre-Depends"

  - This is like Depends, except that it requires completed installation of the listed packages in advance.

- "Recommends"

  - This declares a strong, but not absolute, dependency. Most users would not want the package unless all of the packages listed in this field are installed.

- "Suggests"

  - This declares a weak dependency. Many users of this package may benefit from installing packages listed in this field but can have reasonable functions without them.

- "Enhances"

  - This declares a week dependency like Suggests but works in the opposite direction.

- "Breaks"

  - This declares a package incompatibility usually with some version specification. Generally the resolution is to upgrade all of the packages listed in this field.

- "Conflicts"

  - This declares an absolute incompatibility. All of the packages listed in this field must be removed to install this package.

- "Replaces"

  - This is declared when files installed by this package replace files in the listed packages.

- "Provides"

  - This is declared when this package provide all of the files and functionality in the listed packages.

Note
Please note that defining "Provides", "Conflicts" and "Replaces" simultaneously to an virtual package is the sane configuration. This ensures that only one real package providing this virtual package can be installed at any one time.

The official definition including source dependency can be found in the Policy Manual: Chapter 7 - Declaring relationships between packages.

2.1.7 The event flow of the package management

Here is a summary of the simplified event flow of the package management by APT.

- **Update** ("`apt update`", "`aptitude update`" or "`apt-get update`"):

  1. Fetch archive metadata from remote archive
  2. Reconstruct and update local metadata for use by APT

- **Upgrade** ("`apt upgrade`" and "`apt full-upgrade`", or "`aptitude safe-upgrade`" and "`aptitude full-upgrade`" or "`apt-get upgrade`" and "`apt-get dist-upgrade`"):

  1. Chose candidate version which is usually the latest available version for all installed packages (see Section 2.7.3 for exception)
  2. Make package dependency resolution
  3. Fetch selected binary packages from remote archive if candidate version is different from installed version
  4. Unpack fetched binary packages
  5. Run **preinst** script
  6. Install binary files
  7. Run **postinst** script

- **Install** ("`apt install ⋯`", `aptitude install ⋯`" or "`apt-get install ⋯`"):

  1. Chose packages listed on the command line
  2. Make package dependency resolution
  3. Fetch selected binary packages from remote archive
  4. Unpack fetched binary packages
  5. Run **preinst** script
  6. Install binary files
  7. Run **postinst** script

- **Remove** ("`apt remove ⋯`", "`aptitude remove ⋯`" or "`apt-get remove ⋯`"):

1. Chose packages listed on the command line
2. Make package dependency resolution
3. Run **prerm** script
4. Remove installed files **except** configuration files
5. Run **postrm** script

- **Purge** ("apt purge", "aptitude purge ⋯" or "apt-get purge ⋯"):

  1. Chose packages listed on the command line
  2. Make package dependency resolution
  3. Run **prerm** script
  4. Remove installed files **including** configuration files
  5. Run **postrm** script

Here, I intentionally skipped technical details for the sake of big picture.

2.1.8   First response to package management troubles

You should read the fine official documentation. The first document to read is the Debian specific "/usr/share/doc/<package_na
Other documentation in "/usr/share/doc/<package_name>/" should be consulted too. If you set shell as Section 1.4.2,
type the following.

```
$ cd <package_name>
$ pager README.Debian
$ mc
```

You may need to install the corresponding documentation package named with "-doc" suffix for detailed information.

If you are experiencing problems with a specific package, make sure to check out the Debian bug tracking system (BTS) sites,
first.

web site	command
Home page of the Debian bug tracking system (BTS)	sensible-browser "http://bugs.debian.org/"
The bug report of a known package name	sensible-browser "http://bugs.debian.org/<package_name>"
The bug report of known bug number	sensible-browser "http://bugs.debian.org/<bug_number>"

Table 2.5: List of key web site to resolving problems with a specific package

Search Google with search words including "site:debian.org", "site:wiki.debian.org", "site:lists.debian.org"
etc.

When you file a bug report, please use reportbug(1) command.

2.2   Basic package management operations

Repository based package management operations on the Debian system can be performed by many APT-based package manage-
ment tools available on the Debian system. Here, we explain 3 basic package management tools: apt, apt-get / apt-cache
and aptitude.

For the package management operation which involves package installation or updates package metadata, you need to have root
privilege.

2.2.1 `apt` vs. `apt-get` / `apt-cache` vs. `aptitude`

Although `aptitude` is a very nice interactive tool which the author mainly uses, you should know some cautionary facts:

- The `aptitude` command is not recommended for the release-to-release system upgrade on the `stable` Debian system after the new release.

  - The use of "`apt full-upgrade`" or "`apt-get dist-upgrade`" is recommended for it. See Bug #411280.

- The `aptitude` command sometimes suggests mass package removals for the system upgrade on the `testing` or `unstable` Debian system.

  - This situation has frightened many system administrators. Don't panic.
  - This seems to be caused mostly by the version skew among packages depended or recommended by a meta-package such as `gnome-core`.
  - This can be resolved by selecting "Cancel pending actions" in the `aptitude` command menu, exiting `aptitude`, and using "`apt full-upgrade`".

The `apt-get` and `apt-cache` commands are the most **basic** APT-based package management tools.

- `apt-get` and `apt-cache` offer only the commandline user interface.
- `apt-get` is most suitable for the **major system upgrade** between releases, etc.
- `apt-get` offers a **robust** package dependency resolver.
- `apt-get` is less demanding on hardware resources. It consumes less memory and runs faster.
- `apt-cache` offers a **standard** regex based search on the package name and description.
- `apt-get` and `apt-cache` can manage multiple versions of packages using `/etc/apt/preferences` but it is quite cumbersome.

The `apt` command is a high-level commandline interface for package management. It is basically a wrapper of `apt-get`, `apt-cache` and similar commands, originally intended as an end-user interface and enables some options better suited for interactive usage by default.

- `apt` provides a friendly progress bar when installing packages using `apt install`.
- `apt` will **remove** cached `.deb` packages by default after sucessful installation of downloaded packages.

Tip
Users are recommended to use the new apt(8) command for **interactive** usage and use the apt-get(8) and apt-cache(8) commands in the shell script.

The `aptitude` command is the most **versatile** APT-based package management tool.

- `aptitude` offers the fullscreen interactive text user interface.
- `aptitude` offers the commandline user interface, too.
- `aptitude` is most suitable for the **daily interactive package management** such as inspecting installed packages and searching available packages.
- `aptitude` is more demanding on hardware resources. It consumes more memory and runs slower.
- `aptitude` offers an **enhanced** regex based search on all of the package metadata.
- `aptitude` can manage multiple versions of packages without using `/etc/apt/preferences` and it is quite intuitive.

apt syntax	**aptitude** syntax	**apt-get**/**apt-cache** syntax	description
apt update	aptitude update	apt-get update	update package archive metadata
apt install foo	aptitude install foo	apt-get install foo	install candidate version of "foo" package with its dependencies
apt upgrade	aptitude safe-upgrade	apt-get upgrade	install candidate version of installed packages without removing any other packages
apt full-upgrade	aptitude full-upgrade	apt-get dist-upgrade	install candidate version of installed packages while removing other packages if needed
apt remove foo	aptitude remove foo	apt-get remove foo	remove "foo" package while leaving its configuration files
apt autoremove	N/A	apt-get autoremove	remove auto-installed packages which are no longer required
apt purge foo	aptitude purge foo	apt-get purge foo	purge "foo" package with its configuration files
apt clean	aptitude clean	apt-get clean	clear out the local repository of retrieved package files completely
apt autoclean	aptitude autoclean	apt-get autoclean	clear out the local repository of retrieved package files for outdated packages
apt show foo	aptitude show foo	apt-cache show foo	display detailed information about "foo" package
apt search <regex>	aptitude search <regex>	apt-cache search <regex>	search packages which match <regex>
N/A	aptitude why <regex>	N/A	explain the reason why <regex> matching packages should be installed
N/A	aptitude why-not <regex>	N/A	explain the reason why <regex> matching packages can not be installed
N/A	aptitude search '~i!~M'	apt-mark showmanual	list manually installed packages

Table 2.6: Basic package management operations with the commandline using apt(8), aptitude(8) and apt-get(8) /apt-cache(8)

2.2.2   Basic package management operations with the commandline

Here are basic package management operations with the commandline using apt(8), aptitude(8) and apt-get(8) /apt-cache(8).

Note
Although the `aptitude` command comes with rich features such as its enhanced package resolver, this complexity has caused (or may still causes) some regressions such as Bug #411123, Bug #514930, and Bug #570377. In case of doubt, please use the `apt`, `apt-get` and `apt-cache` commands over the `aptitude` command.

Note
Since `apt` / `apt-get` and `aptitude` share auto-installed package status (see Section 2.5.5) after `lenny`, you can mix these tools without major troubles (see Bug #594490).

The "`aptitude why <regex>`" can list more information by "`aptitude -v why <regex>`". Similar information can be obtained by "`apt rdepends <package>`" or "`apt-cache rdepends <package>`".

When `aptitude` command is started in the commandline mode and faces some issues such as package conflicts, you can switch to the full screen interactive mode by pressing "`e`"-key later at the prompt.

You may provide command options right after "`aptitude`".

command option	description
-s	simulate the result of the command
-d	download only but no install/upgrade
-D	show brief explanations before the automatic installations and removals

Table 2.7: Notable command options for aptitude(8)

See aptitude(8) and "aptitude user's manual" at "`/usr/share/doc/aptitude/README`" for more.

Tip
The `dselect` package is still available and was the preferred full screen interactive package management tool in previous releases.

2.2.3   Interactive use of aptitude

For the interactive package management, you start `aptitude` in interactive mode from the console shell prompt as follows.

```
$ sudo aptitude -u
Password:
```

This updates the local copy of the archive information and display the package list in the full screen with menu. Aptitude places its configuration at "`~/.aptitude/config`".

Tip
If you want to use root's configuration instead of user's one, use "`sudo -H aptitude ⋯`" instead of "`sudo aptitude ⋯`" in the above expression.

Tip
Aptitude automatically sets **pending actions** as it is started interactively. If you do not like it, you can reset it from menu: "Action" → "Cancel pending actions".

2.2.4 Key bindings of aptitude

Notable key strokes to browse status of packages and to set "planned action" on them in this full screen mode are the following.

key	key binding
F10 or Ctrl-t	menu
?	display **help** for keystroke (more complete listing)
F10 → Help → User's Manual	display User's Manual
u	update package archive information
+	mark the package for the **upgrade** or the **install**
-	mark the package for the **remove** (keep configuration files)
_	mark the package for the **purge** (remove configuration files)
=	place the package on **hold**
U	mark all upgradable packages (function as **full-upgrade**)
g	start **downloading** and **installing** selected packages
q	quit current screen and save changes
x	quit current screen and discard changes
Enter	view information about a package
C	view a package's changelog
l	change the limit for the displayed packages
/	search for the first match
\	repeat the last search

Table 2.8: List of key bindings for aptitude

The file name specification of the command line and the menu prompt after pressing "l" and "//" take the aptitude regex as described below. Aptitude regex can explicitly match a package name using a string started by "~n and followed by the package name.

Tip
You need to press "U" to get all the installed packages upgraded to the **candidate version** in the visual interface. Otherwise only the selected packages and certain packages with versioned dependency to them are upgraded to the **candidate version**.

2.2.5 Package views under aptitude

In the interactive full screen mode of aptitude(8), packages in the package list are displayed as the next example.

```
idA   libsmbclient                          -2220kB 3.0.25a-1  3.0.25a-2
```

Here, this line means from the left as the following.

- The "current state" flag (the first letter)

- The "planned action" flag (the second letter)

- The "automatic" flag (the third letter)

- The Package name

- The change in disk space usage attributed to "planned action"

- The current version of the package

- The candidate version of the package

> **Tip**
> The full list of flags are given at the bottom of **Help** screen shown by pressing "?".

The **candidate version** is chosen according to the current local preferences (see apt_preferences(5) and Section 2.7.3).

Several types of package views are available under the menu "Views".

view	status	description of view
Package View	Good	see Table 2.10 (default)
Audit Recommendations	Good	list packages which are recommended by some installed packages but not yet installed
Flat Package List	Good	list packages without categorization (for use with regex)
Debtags Browser	Very usable	list packages categorized according to their debtags entries
Categorical Browser	Deprecated	list packages categorized according to their category (use Debtags Browser, instead)

Table 2.9: List of views for aptitude

> **Note**
> Please help us improving tagging packages with debtags!

The standard "Package View" categorizes packages somewhat like dselect with few extra features.

category	description of view
Upgradable Packages	list packages organized as section → area → package
New Packages	,,
Installed Packages	,,
Not Installed Packages	,,
Obsolete and Locally Created Packages	,,
Virtual Packages	list packages with the same function
Tasks	list packages with different functions generally needed for a task

Table 2.10: The categorization of standard package views

> **Tip**
> Tasks view can be used to cherry pick packages for your task.

2.2.6   Search method options with aptitude

Aptitude offers several options for you to search packages using its regex formula.

- Shell commandline:

  - "aptitude search '<aptitude_regex>'" to list installation status, package name and short description of matching packages
  - "aptitude show '<package_name>'" to list detailed description of the package

- Interactive full screen mode:

  - "l" to limit package view to matching packages
  - "/" for search to a matching package
  - "\" for backward search to a matching package
  - "n" for find-next
  - "N" for find-next (backward)

Tip
The string for <package_name> is treated as the exact string match to the package name unless it is started explicitly with "~" to be the regex formula.

2.2.7   The aptitude regex formula

The aptitude regex formula is mutt-like extended **ERE** (see Section 1.6.2) and the meanings of the `aptitude` specific special match rule extensions are as follows.

- The regex part is the same **ERE** as the one used in typical Unix-like text tools using "ˆ", ".*", "$" etc. as in egrep(1), awk(1) and perl(1).

- The dependency <type> is one of (depends, predepends, recommends, suggests, conflicts, replaces, provides) specifying the package interrelationship.

- The default dependency <type> is "depends".

Tip
When <regex_pattern> is a null string, place "~T" immediately after the command.

Here are some short cuts.

- "~P<term>" == "~Dprovides:<term>"

- "~C<term>" == "~Dconflicts:<term>"

- "···~W term" == "(···|term)"

Users familiar with `mutt` pick up quickly, as mutt was the inspiration for the expression syntax. See "SEARCHING, LIMITING, AND EXPRESSIONS" in the "User's Manual" "/usr/share/doc/aptitude/README".

Note
With the `lenny` version of aptitude(8), the new **long form** syntax such as "?broken" may be used for regex matching in place for its old **short form** equivalent "~b". Now space character " " is considered as one of the regex terminating character in addition to tilde character "~". See "User's Manual" for the new **long form** syntax.

2.2.8   Dependency resolution of aptitude

The selection of a package in `aptitude` not only pulls in packages which are defined in its "`Depends:`" list but also defined in the "`Recommends:`" list if the menu "F10 → Options → Preferences → Dependency handling" is set accordingly. These auto installed packages are removed automatically if they are no longer needed under `aptitude`.

The flag controlling the "auto install" behavior of the `aptitude` command can also be manipulated using the apt-mark(8) command from the `apt` package.

description of the extended match rule	regex formula
match on package name	`~n<regex_name>`
match on description	`~d<regex_description>`
match on task name	`~t<regex_task>`
match on debtag	`~G<regex_debtag>`
match on maintainer	`~m<regex_maintainer>`
match on package section	`~s<regex_section>`
match on package version	`~V<regex_version>`
match archive	`~A{stretch,buster,sid}`
match origin	`~O{debian,···}`
match priority	`~p{extra,important,optional,required,standard}`
match essential packages	`~E`
match virtual packages	`~v`
match new packages	`~N`
match with pending action	`~a{install,upgrade,downgrade,remove,purge,hold,keep}`
match installed packages	`~i`
match installed packages with **A**-mark (auto installed packages)	`~M`
match installed packages without **A**-mark (administrator selected packages)	`~i!~M`
match installed and upgradable packages	`~U`
match removed but not purged packages	`~c`
match removed, purged or can-be-removed packages	`~g`
match packages declaring a broken dependency	`~b`
match packages declaring broken dependency of <type>	`~B<type>`
match <pattern> packages declaring dependency of <type>	`~D[<type>:]<pattern>`
match <pattern> packages declaring broken dependency of <type>	`~DB[<type>:]<pattern>`
match packages to which the <pattern> matching package declares dependency <type>	`~R[<type>:]<pattern>`
match packages to which the <pattern> matching package declares broken dependency <type>	`~RB[<type>:]<pattern>`
match packages to which some other installed packages depend on	`~R~i`
match packages to which no other installed packages depend on	`!~R~i`
match packages to which some other installed packages depend or recommend on	`~R~i|~Rrecommends:~i`
match <pattern> package with filtered version	`~S filter <pattern>`
match all packages (true)	`~T`
match no packages (false)	`~F`

Table 2.11: List of the aptitude regex formula

2.2.9   Package activity logs

You can check package activity history in the log files.

file	content
/var/log/dpkg.log	Log of dpkg level activity for all package activities
/var/log/apt/term.log	Log of generic APT activity
/var/log/aptitude	Log of aptitude command activity

Table 2.12: The log files for package activities

In reality, it is not so easy to get meaningful understanding quickly out from these logs. See Section 9.2.10 for easier way.

2.3   Examples of aptitude operations

Here are few examples of aptitude(8) operations.

2.3.1   Listing packages with regex matching on package names

The following command lists packages with regex matching on package names.

```
$ aptitude search '~n(pam|nss).*ldap'
p libnss-ldap - NSS module for using LDAP as a naming service
p libpam-ldap - Pluggable Authentication Module allowing LDAP interfaces
```

This is quite handy for you to find the exact name of a package.

2.3.2   Browsing with the regex matching

The regex "~dipv6" in the "New Flat Package List" view with "l" prompt, limits view to packages with the matching description and let you browse their information interactively.

2.3.3   Purging removed packages for good

You can purge all remaining configuration files of removed packages.

Check results of the following command.

```
# aptitude search '~c'
```

If you think listed packages are OK to be purged, execute the following command.

```
# aptitude purge '~c'
```

You may want to do the similar in the interactive mode for fine grained control.

You provide the regex "~c" in the "New Package View" view with "l" prompt. This limits the package view only to regex matched packages, i.e., "removed but not purged". All these regex matched packages can be shown by pressing "[" at top level headings.

Then you press "_" at top level headings such as "Not Installed Packages". Only regex matched packages under the heading are marked to be purged by this. You can exclude some packages to be purged by pressing "=" interactively for each of them.

This technique is quite handy and works for many other command keys.

2.3.4   Tidying auto/manual install status

Here is how I tidy auto/manual install status for packages (after using non-aptitude package installer etc.).

1. Start `aptitude` in interactive mode as root.

2. Type ”u”, ”U”, ”f” and ”g” to update and upgrade package list and packages.

3. Type ”l” to enter the package display limit as ”~i(~R~i|~Rrecommends:~i)” and type ”M” over ”`Installed Packages`” as auto installed.

4. Type ”l” to enter the package display limit as ”~prequired|~pimportant|~pstandard|~E” and type ”m” over ”`Installed Packages`” as manual installed.

5. Type ”l” to enter the package display limit as ”~i!~M” and remove unused package by typing ”-” over each of them after exposing them by typing ”[” over ”`Installed Packages`”.

6. Type ”l”, to enter the package display limit as ”~i”; then type ”m” over ”`Tasks`”, to mark that packages as manual installed.

7. Exit `aptitude`.

8. Start ”`apt-get -s autoremove|less`” as root to check what are not used.

9. Restart `aptitude` in interactive mode and mark needed packages as ”m”.

10. Restart ”`apt-get -s autoremove|less`” as root to recheck REMOVED contain only expected packages.

11. Start ”`apt-get autoremove|less`” as root to autoremove unused packages.

The ”m” action over ”`Tasks`” is an optional one to prevent mass package removal situation in future.

2.3.5   System wide upgrade

Note
When moving to a new release etc, you should consider to perform a clean installation of new system even though Debian is upgradable as described below. This provides you a chance to remove garbages collected and exposes you to the best combination of latest packages. Of course, you should make a full backup of system to a safe place (see Section 10.2) before doing this. I recommend to make a dual boot configuration using different partition to have the smoothest transition.

You can perform system wide upgrade to a newer release by changing contents of the ”`/etc/apt/sources.list`” file pointing to a new release and running the ”`apt update; apt dist-upgrade`” command.

To upgrade from `stable` to `testing` or `unstable`, you replace ”`stretch`” in the ”`/etc/apt/sources.list`” example of Section 2.1.4 with ”`buster`” or ”`sid`”.

In reality, you may face some complications due to some package transition issues, mostly due to package dependencies. The larger the difference of the upgrade, the more likely you face larger troubles. For the transition from the old `stable` to the new `stable` after its release, you can read its new Release Notes and follow the exact procedure described in it to minimize troubles.

When you decide to move from `stable` to `testing` before its formal release, there are no Release Notes to help you. The difference between `stable` and `testing` could have grown quite large after the previous `stable` release and makes upgrade situation complicated.

You should make precautionary moves for the full upgrade while gathering latest information from mailing list and using common senses.

1. Read previous ”Release Notes”.

2. Backup entire system (especially data and configuration information).

3. Have bootable media handy for broken bootloader.

4. Inform users on the system well in advance.

5. Record upgrade activity with script(1).

6. Apply "unmarkauto" to required packages, e.g., "`aptitude unmarkauto vim`", to prevent removal.

7. Minimize installed packages to reduce chance of package conflicts, e.g., remove desktop task packages.

8. Remove the "`/etc/apt/preferences`" file (disable apt-pinning).

9. Try to upgrade step wise: `oldstable` → `stable` → `testing` → `unstable`.

10. Update the "`/etc/apt/sources.list`" file to point to new archive only and run "`aptitude update`".

11. Install, optionally, new **core packages** first, e.g., "`aptitude install perl`".

12. Run the "`apt-get -s dist-upgrade`" command to assess impact.

13. Run the "`apt-get dist-upgrade`" command at last.

⚠ **Caution**
It is not wise to skip major Debian release when upgrading between `stable` releases.

⚠ **Caution**
In previous "Release Notes", GCC, Linux Kernel, initrd-tools, Glibc, Perl, APT tool chain, etc. have required some special attention for system wide upgrade.

For daily upgrade in `unstable`, see Section 2.4.3.

2.4 Advanced package management operations

2.4.1 Advanced package management operations with commandline

Here are list of other package management operations for which `aptitude` is too high-level or lacks required functionalities.

Note
For a package with the multi-arch feature, you may need to specify the architecture name for some commands. For example, use "`dpkg -L libglib2.0-0:amd64`" to list contents of the `libglib2.0-0` package for the `amd64` architecture.

⚠ **Caution**
Lower level package tools such as "`dpkg -i ···`" and "`debi ···`" should be carefully used by the system administrator. It does not automatically take care required package dependencies. Dpkg's commandline options "`--force-all`" and similar (see dpkg(1)) are intended to be used by experts only. Using them without fully understanding their effects may break your whole system.

---

command	action
`COLUMNS=120 dpkg -l <package_name_pattern>`	list status of an installed package for the bug report
`dpkg -L <package_name>`	list contents of an installed package
`dpkg -L <package_name>	egrep '/usr/share/man/man.*/.+'`	list manpages for an installed package
`dpkg -S <file_name_pattern>`	list installed packages which have matching file name
`apt-file search <file_name_pattern>`	list packages in archive which have matching file name
`apt-file list <package_name_pattern>`	list contents of matching packages in archive
`dpkg-reconfigure <package_name>`	reconfigure the exact package
`dpkg-reconfigure -p=low <package_name>`	reconfigure the exact package with the most detailed question
`configure-debian`	reconfigure packages from the full screen menu
`dpkg --audit`	audit system for partially installed packages
`dpkg --configure -a`	configure all partially installed packages
`apt-cache policy <binary_package_name>`	show available version, priority, and archive information of a binary package
`apt-cache madison <package_name>`	show available version, archive information of a package
`apt-cache showsrc <binary_package_name>`	show source package information of a binary package
`apt-get build-dep <package_name>`	install required packages to build package
`aptitude build-dep <package_name>`	install required packages to build package
`apt-get source <package_name>`	download a source (from standard archive)
`dget <URL for dsc file>`	download a source packages (from other archive)
`dpkg-source -x <package_name>_<version>-<debian_version>.dsc`	build a source tree from a set of source packages (”`*.orig.tar.gz`” and ”`*.debian.tar.gz`”/”`*.diff.gz`”)
`debuild binary`	build package(s) from a local source tree
`make-kpkg kernel_image`	build a kernel package from a kernel source tree
`make-kpkg --initrd kernel_image`	build a kernel package from a kernel source tree with initramfs enabled
`dpkg -i <package_name>_<version>-<debian_version>_<arch>.deb`	install a local package to the system
`apt install /path/to/<package_filename>.deb`	install a local package to the system, meanwhile try to resolve dependency automatically
`debi <package_name>_<version>-<debian_version>_<arch>.dsc`	install local package(s) to the system
`dpkg --get-selections '*' >selection.txt`	save `dpkg` level package selection state information
`dpkg --set-selections <selection.txt`	set `dpkg` level package selection state information
`echo <package_name> hold	dpkg --set-selections`	set `dpkg` level package selection state for a package to **hold** (equivalent to ”`aptitude hold <package_name>`”)

Table 2.13: List of advanced package management operations

Please note the following.

- All system configuration and installation commands require to be run from root.

- Unlike `aptitude` which uses regex (see Section 1.6.2), other package management commands use pattern like shell glob (see Section 1.5.6).

- apt-file(1) provided by the `apt-file` package must run "`apt-file update`" in advance.

- configure-debian(8) provided by the `configure-debian` package runs dpkg-reconfigure(8) as its backend.

- dpkg-reconfigure(8) runs package scripts using debconf(1) as its backend.

- "`apt-get build-dep`", "`apt-get source`" and "`apt-cache showsrc`" commands require "`deb-src`" entry in "`/etc/apt/sources.list`".

- dget(1), debuild(1), and debi(1) require `devscripts` package.

- See (re)packaging procedure using "`apt-get source`" in Section 2.7.13.

- `make-kpkg` command requires the `kernel-package` package (see Section 9.9).

- See Section 12.11 for general packaging.

2.4.2 Verification of installed package files

The installation of `debsums` enables verification of installed package files against MD5sum values in the "`/var/lib/dpkg/info/*`" file with debsums(1). See Section 10.3.5 for how MD5sum works.

Note
Because MD5sum database may be tampered by the intruder, debsums(1) is of limited use as a security tool. It is only good for checking local modifications by the administrator or damage due to media errors.

2.4.3 Safeguarding for package problems

Many users prefer to follow the **unstable** release of the Debian system for its new features and packages. This makes the system more prone to be hit by the critical package bugs.

The installation of the `apt-listbugs` package safeguards your system against critical bugs by checking Debian BTS automatically for critical bugs when upgrading with APT system.

The installation of the `apt-listchanges` package provides important news in "`NEWS.Debian`" when upgrading with APT system.

2.4.4 Searching on the package meta data

Although visiting Debian site https://packages.debian.org/ facilitates easy ways to search on the package meta data these days, let's look into more traditional ways.

The grep-dctrl(1), grep-status(1), and grep-available(1) commands can be used to search any file which has the general format of a Debian package control file.

The "`dpkg -S <file_name_pattern>`" can be used search package names which contain files with the matching name installed by `dpkg`. But this overlooks files created by the maintainer scripts.

If you need to make more elaborate search on the dpkg meta data, you need to run "`grep -e regex_pattern *`" command in the "`/var/lib/dpkg/info/`" directory. This makes you search words mentioned in package scripts and installation query texts.

If you wish to look up package dependency recursively, you should use apt-rdepends(8).

2.5   Debian package management internals

Let's learn how the Debian package management system works internally. This should help you to create your own solution to some package problems.

2.5.1   Archive meta data

Meta data files for each distribution are stored under "dist/<codename>" on each Debian mirror sites, e.g., "http://deb.debian
Its archive structure can be browsed by the web browser. There are 6 types of key meta data.

file	location	content
Release	top of distribution	archive description and integrity information
Release.gpg	top of distribution	signature file for the "Release" file signed with the archive key
Contents-<architecture>	top of distribution	list of all files for all the packages in the pertinent archive
Release	top of each distribution/area/architecture combination	archive description used for the rule of apt_preferences(5)
Packages	top of each distribution/area/binary-architecture combination	concatenated debian/control for binary packages
Sources	top of each distribution/area/source combination	concatenated debian/control for source packages

Table 2.14: The content of the Debian archive meta data

In the recent archive, these meta data are stored as the compressed and differential files to reduce network traffic.

2.5.2   Top level "Release" file and authenticity

> **Tip**
> The top level "Release" file is used for signing the archive under the **secure APT** system.

Each suite of the Debian archive has a top level "Release" file, e.g., "http://deb.debian.org/debian/dists/unstable/
as follows.

```
Origin: Debian
Label: Debian
Suite: unstable
Codename: sid
Date: Sat, 14 May 2011 08:20:50 UTC
Valid-Until: Sat, 21 May 2011 08:20:50 UTC
Architectures: alpha amd64 armel hppa hurd-i386 i386 ia64 kfreebsd-amd64 kfreebsd-i386 mips ←
     mipsel powerpc s390 sparc
Components: main
Description: Debian x.y Unstable - Not Released
MD5Sum:
 bdc8fa4b3f5e4a715dd0d56d176fc789 18876880 Contents-alpha.gz
 9469a03c94b85e010d116aeeab9614c0 19441880 Contents-amd64.gz
 3d68e206d7faa3aded660dc0996054fe 19203165 Contents-armel.gz
...
```

> **Note**
> Here, you can find my rationale to use the "suite", and "codename" in Section 2.1.4. The "distribution" is used
> when referring to both "suite" and "codename". All archive "area" names offered by the archive are listed under
> "Components".

The integrity of the top level "`Release`" file is verified by cryptographic infrastructure called the secure apt.

- The cryptographic signature file "`Release.gpg`" is created from the authentic top level "`Release`" file and the secret
  Debian archive key.

- The public Debian archive key can be seeded into "`/etc/apt/trusted.gpg`";

  - automatically by installing the keyring with the latest `base-files` package, or

  - manually by `gpg` or `apt-key` tool with the latest public archive key posted on the ftp-master.debian.org .

- The **secure APT** system verifies the integrity of the downloaded top level "`Release`" file cryptographically by this "`Release.gpg`"
  file and the public Debian archive key in "`/etc/apt/trusted.gpg`".

The integrity of all the "`Packages`" and "`Sources`" files are verified by using MD5sum values in its top level "`Release`"
file. The integrity of all package files are verified by using MD5sum values in the "`Packages`" and "`Sources`" files. See
debsums(1) and Section 2.4.2.

Since the cryptographic signature verification is a much more CPU intensive process than the MD5sum value calculation, use
of MD5sum value for each package while using cryptographic signature for the top level "`Release`" file provides the good
security with the performance (see Section 10.3).

2.5.3   Archive level "Release" files

> **Tip**
> The archive level "`Release`" files are used for the rule of apt_preferences(5).

There are archive level "`Release`" files for all archive locations specified by "`deb`" line in "`/etc/apt/sources.list`",
such as "`http://deb.debian.org/debian/dists/unstable/main/binary-amd64/Release`" or "`http://deb.d`
as follows.

```
Archive: unstable
Origin: Debian
Label: Debian
Component: main
Architecture: amd64
```

> ⚠ **Caution**
> For "`Archive:`" stanza, suite names ("`stable`", "`testing`", "`unstable`", ⋯) are used in the Debian
> archive while codenames ("`trusty`", "`xenial`", "`artful`", ⋯) are used in the Ubuntu archive.

For some archives, such as `experimental`, and `stretch-backports`, which contain packages which should not be in-
stalled automatically, there is an extra line, e.g., "`http://deb.debian.org/debian/dists/experimental/main/binary`
as follows.

```
Archive: experimental
Origin: Debian
Label: Debian
NotAutomatic: yes
Component: main
Architecture: amd64
```

Please note that for normal archives without "`NotAutomatic: yes`", the default Pin-Priority value is 500, while for special archives with "`NotAutomatic: yes`", the default Pin-Priority value is 1 (see apt_preferences(5) and Section 2.7.3).

2.5.4 Fetching of the meta data for the package

When APT tools, such as `aptitude`, `apt-get`, `synaptic`, `apt-file`, `auto-apt`, ⋯are used, we need to update the local copies of the meta data containing the Debian archive information. These local copies have following file names corresponding to the specified `distribution`, `area`, and `architecture` names in the "`/etc/apt/sources.list`" (see Section 2.1.4).

- "`/var/lib/apt/lists/deb.debian.org_debian_dists_<distribution>_Release`"

- "`/var/lib/apt/lists/deb.debian.org_debian_dists_<distribution>_Release.gpg`"

- "`/var/lib/apt/lists/deb.debian.org_debian_dists_<distribution>_<area>_binary-<architectur`

- "`/var/lib/apt/lists/deb.debian.org_debian_dists_<distribution>_<area>_source_Sources`"

- "`/var/cache/apt/apt-file/deb.debian.org_debian_dists_<distribution>_Contents-<architectur`
  (for `apt-file`)

First 4 types of files are shared by all the pertinent APT commands and updated from command line by "`apt-get update`" or "`aptitude update`". The "`Packages`" meta data are updated if there is the "`deb`" line in "`/etc/apt/sources.list`". The "`Sources`" meta data are updated if there is the "`deb-src`" line in "`/etc/apt/sources.list`".

The "`Packages`" and "`Sources`" meta data contain "`Filename:`" stanza pointing to the file location of the binary and source packages. Currently, these packages are located under the "`pool/`" directory tree for the improved transition over the releases.

Local copies of "`Packages`" meta data can be interactively searched with the help of `aptitude`. The specialized search command grep-dctrl(1) can search local copies of "`Packages`" and "`Sources`" meta data.

Local copy of "`Contents-<architecture>`" meta data can be updated by "`apt-file update`" and its location is different from other 4 ones. See apt-file(1). (The `auto-apt` uses different location for local copy of "`Contents-<architecture>.gz`" as default.)

2.5.5 The package state for APT

In addition to the remotely fetched meta data, the APT tool after `lenny` stores its locally generated installation state information in the "`/var/lib/apt/extended_states`" which is used by all APT tools to track all auto installed packages.

2.5.6 The package state for aptitude

In addition to the remotely fetched meta data, the `aptitude` command stores its locally generated installation state information in the "`/var/lib/aptitude/pkgstates`" which is used only by it.

2.5.7 Local copies of the fetched packages

All the remotely fetched packages via APT mechanism are stored in the "`/var/cache/apt/archives`" until they are cleaned.

This cache file cleaning policy for `aptitude` can be set under "`Options`" → "`Preferences`" and it may be forced by its menu "`Clean package cache`" or "`Clean obsolete files`" under "`Actions`".

2.5.8   Debian package file names

Debian package files have particular name structures.

package type	name structure
The binary package (a.k.a `deb`)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`
The binary package for debian-installer (a.k.a `udeb`)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`
The source package (upstream source)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`
The `1.0` source package (Debian changes)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`
The `3.0 (quilt)` source package (Debian changes)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`
The source package (description)	`<package-name>_<epoch>:<upstream-version>-<debian.vers`

Table 2.15: The name structure of Debian packages

Tip
Here only the basic source package formats are described. See more on dpkg-source(1).

---

name component	usable characters (regex)	existence
`<package-name>`	`[a-z,A-Z,0-9,.,+,-]+`	required
`<epoch>:`	`[0-9]+:`	optional
`<upstream-version>`	`[a-z,A-Z,0-9,.,+,-,:]+`	required
`<debian.version>`	`[a-z,A-Z,0-9,.,+,~]+`	optional

Table 2.16: The usable characters for each component in the Debian package names

---

Note
You can check package version order by dpkg(1), e.g., "dpkg --compare-versions 7.0 gt 7.~pre1 ; echo $?".

Note
The debian-installer (d-i) uses `udeb` as the file extension for its binary package instead of normal deb. An udeb package is a stripped down `deb` package which removes few non-essential contents such as documentation to save space while relaxing the package policy requirements. Both `deb` and `udeb` packages share the same package structure. The "u" stands for micro.

2.5.9   The dpkg command

dpkg(1) is the lowest level tool for the Debian package management. This is very powerful and needs to be used with care.

While installing package called "`<package_name>`", `dpkg` process it in the following order.

1. Unpack the deb file ("`ar -x`" equivalent)

2. Execute "`<package_name>.preinst`" using debconf(1)

3. Install the package content to the system ("`tar -x`" equivalent)

4. Execute "`<package_name>.postinst`" using debconf(1)

file	description of contents
/var/lib/dpkg/info/<package_name>.conffiles	configuration files. (user modifiable)
/var/lib/dpkg/info/<package_name>.list	list of files and directories installed by the package
/var/lib/dpkg/info/<package_name>.md5sums	MD5 hash values for files installed by the package
/var/lib/dpkg/info/<package_name>.preinst	package script to be run before the package installation
/var/lib/dpkg/info/<package_name>.postinst	package script to be run after the package installation
/var/lib/dpkg/info/<package_name>.prerm	package script to be run before the package removal
/var/lib/dpkg/info/<package_name>.postrm	package script to be run after the package removal
/var/lib/dpkg/info/<package_name>.config	package script for debconf system
/var/lib/dpkg/alternatives/<package_name>	the alternative information used by the update-alternatives command
/var/lib/dpkg/available	the availability information for all the package
/var/lib/dpkg/diversions	the diversions information used by dpkg(1) and set by dpkg-divert(8)
/var/lib/dpkg/statoverride	the stat override information used by dpkg(1) and set by dpkg-statoverride(8)
/var/lib/dpkg/status	the status information for all the packages
/var/lib/dpkg/status-old	the first-generation backup of the "var/lib/dpkg/status" file
/var/backups/dpkg.status*	the second-generation backup and older ones of the "var/lib/dpkg/status" file

Table 2.17: The notable files created by dpkg

The debconf system provides standardized user interaction with I18N and L10N (Chapter 8) supports.

The "status" file is also used by the tools such as dpkg(1), "dselect update" and "apt-get -u dselect-upgrade".

The specialized search command grep-dctrl(1) can search the local copies of "status" and "available" meta data.

Tip
In the debian-installer environment, the udpkg command is used to open udeb packages. The udpkg command
is a stripped down version of the dpkg command.

2.5.10 The update-alternatives command

The Debian system has mechanism to install somewhat overlapping programs peacefully using update-alternatives(1). For example, you can make the vi command select to run vim while installing both vim and nvi packages.

```
$ ls -l $(type -p vi)
lrwxrwxrwx 1 root root 20 2007-03-24 19:05 /usr/bin/vi -> /etc/alternatives/vi
$ sudo update-alternatives --display vi
...
$ sudo update-alternatives --config vi
  Selection    Command
 ---------------------------------------------
      1        /usr/bin/vim
*+    2        /usr/bin/nvi

Enter to keep the default[*], or type selection number: 1
```

The Debian alternatives system keeps its selection as symlinks in "/etc/alternatives/". The selection process uses corresponding file in "/var/lib/dpkg/alternatives/".

2.5.11 The dpkg-statoverride command

Stat overrides provided by the dpkg-statoverride(8) command are a way to tell dpkg(1) to use a different owner or mode for a **file** when a package is installed. If "--update" is specified and file exists, it is immediately set to the new owner and mode.

> **⚠ Caution**
> The direct alteration of owner or mode for a **file** owned by the package using `chmod` or `chown` commands
> by the system administrator is reset by the next upgrade of the package.

Note
I use the word **file** here, but in reality this can be any filesystem object that `dpkg` handles, including directories,
devices, etc.

2.5.12 The dpkg-divert command

File **diversions** provided by the dpkg-divert(8) command are a way of forcing dpkg(1) not to install a file into its default location,
but to a **diverted** location. The use of `dpkg-divert` is meant for the package maintenance scripts. Its casual use by the system
administrator is deprecated.

2.6 Recovery from a broken system

When running `unstable` system, the administrator is expected to recover from broken package management situation.

> **⚠ Caution**
> Some methods described here are high risk actions. You have been warned!

2.6.1 Incompatibility with old user configuration

If a desktop GUI program experienced instability after significant upstream version upgrade, you should suspect interferences
with old local configuration files created by it. If it is stable under a newly created user account, this hypothesis is confirmed.
(This is a bug of packaging and usually avoided by the packager.)

To recover stability, you should move corresponding local configuration files and restart the GUI program. You may need to read
old configuration file contents to recover configuration information later. (Do not erase them too quickly.)

2.6.2 Different packages with overlapped files

Archive level package management systems, such as aptitude(8) or apt-get(1), do not even try to install packages with overlapped
files using package dependencies (see Section 2.1.6).

Errors by the package maintainer or deployment of inconsistently mixed source of archives (see Section 2.7.2) by the system
administrator may create a situation with incorrectly defined package dependencies. When you install a package with overlapped
files using aptitude(8) or apt-get(1) under such a situation, dpkg(1) which unpacks package ensures to return error to the calling
program without overwriting existing files.

> **⚠ Caution**
> The use of third party packages introduces significant system risks via maintainer scripts which are run with
> root privilege and can do anything to your system. The dpkg(1) command only protects against overwriting
> by the unpacking.

You can work around such broken installation by removing the old offending package, `<old-package>`, first.

```
$ sudo dpkg -P <old-package>
```

2.6.3   Fixing broken package script

When a command in the package script returns error for some reason and the script exits with error, the package management system aborts their action and ends up with partially installed packages. When a package contains bugs in its removal scripts, the package may become impossible to remove and quite nasty.

For the package script problem of "`<package_name>`", you should look into following package scripts.

- "/var/lib/dpkg/info/<package_name>.preinst"

- "/var/lib/dpkg/info/<package_name>.postinst"

- "/var/lib/dpkg/info/<package_name>.prerm"

- "/var/lib/dpkg/info/<package_name>.postrm"

Edit the offending package script from the root using following techniques.

- disable the offending line by preceding "#"

- force to return success by appending the offending line with "|| true"

Configure all partially installed packages with the following command.

```
# dpkg --configure -a
```

2.6.4   Rescue with the dpkg command

Since `dpkg` is very low level package tool, it can function under the very bad situation such as unbootable system without network connection. Let's assume `foo` package was broken and needs to be replaced.

You may still find cached copies of older bug free version of `foo` package in the package cache directory: "/var/cache/apt/archi (If not, you can download it from archive of http://snapshot.debian.org/ or copy it from package cache of a functioning machine.)

If you can boot the system, you may install it by the following command.

```
# dpkg -i /path/to/foo_<old_version>_<arch>.deb
```

Tip
If system breakage is minor, you may alternatively downgrade the whole system as in Section 2.7.10 using the higher level APT system.

If your system is unbootable from hard disk, you should seek other ways to boot it.

1. Boot the system using the debian-installer CD in rescue mode.

2. Mount the unbootable system on the hard disk to "/target".

3. Install older version of `foo` package by the following.

```
# dpkg --root /target -i /path/to/foo_<old_version>_<arch>.deb
```

This example works even if the `dpkg` command on the hard disk is broken.

Tip
Any GNU/Linux system started by another system on hard disk, live GNU/Linux CD, bootable USB-key drive, or netboot can be used similarly to rescue broken system.

If attempting to install a package this way fails due to some dependency violations and you really need to do this as the last resort, you can override dependency using dpkg's "--ignore-depends", "--force-depends" and other options. If you do this, you need to make serious effort to restore proper dependency later. See dpkg(8) for details.

Note
If your system is seriously broken, you should make a full backup of system to a safe place (see Section 10.2) and should perform a clean installation. This is less time consuming and produces better results in the end.

2.6.5 Recovering package selection data

If "/var/lib/dpkg/status" becomes corrupt for any reason, the Debian system loses package selection data and suffers severely. Look for the old "/var/lib/dpkg/status" file at "/var/lib/dpkg/status-old" or "/var/backups/dpkg.s

Keeping "/var/backups/" in a separate partition may be a good idea since this directory contains lots of important system data.

For serious breakage, I recommend to make fresh re-install after making backup of the system. Even if everything in "/var/" is gone, you can still recover some information from directories in "/usr/share/doc/" to guide your new installation.

Reinstall minimal (desktop) system.

```
# mkdir -p /path/to/old/system
```

Mount old system at "/path/to/old/system/".

```
# cd /path/to/old/system/usr/share/doc
# ls -1 >~/ls1.txt
# cd /usr/share/doc
# ls -1 >>~/ls1.txt
# cd
# sort ls1.txt | uniq | less
```

Then you are presented with package names to install. (There may be some non-package names such as "texmf".)

2.7 Tips for the package management

2.7.1 How to pick Debian packages

You can seek packages which satisfy your needs with aptitude from the package description or from the list under "Tasks".

When you encounter more than 2 similar packages and wonder which one to install without "trial and error" efforts, you should use some **common sense**. I consider following points are good indications of preferred packages.

- Essential: yes > no

- Priority: required > important > standard > optional > extra

- Tasks: packages listed in tasks such as "Desktop environment"

- Packages selected by the dependency package (e.g., python2.4 by python)

- Popcon: higher in the vote and install number

- Changelog: regular updates by the maintainer

- BTS: No RC bugs (no critical, no grave, and no serious bugs)

- BTS: responsive maintainer to bug reports

- BTS: higher number of the recently fixed bugs

- BTS: lower number of remaining non-wishlist bugs

Debian being a volunteer project with distributed development model, its archive contains many packages with different focus and quality. You must make your own decision what to do with them.

2.7.2   Packages from mixed source of archives

> **Caution**
> Installing packages from mixed source of archives is not supported by the official Debian distribution except for officially supported particular combinations of archives such as `stable` with security updates and stretch-updates.

Here is an example of operations to include specific newer upstream version packages found in `unstable` while tracking `testing` for single occasion.

1. Change the "`/etc/apt/sources.list`" file temporarily to single "`unstable`" entry.

2. Run "`aptitude update`".

3. Run "`aptitude install <package-name>`".

4. Recover the original "`/etc/apt/sources.list`" file for `testing`.

5. Run "`aptitude update`".

You do not create the "`/etc/apt/preferences`" file nor need to worry about apt-pinning with this manual approach. But this is very cumbersome.

> **Caution**
> When using mixed source of archives, you must ensure compatibility of packages by yourself since the Debian does not guarantee it. If package incompatibility exists, you may break system. You must be able to judge these technical requirements. The use of mixed source of random archives is completely optional operation and its use is not something I encourage you to use.

General rules for installing packages from different archives are the following.

- Non-binary packages ("`Architecture: all`") are **safer** to install.

  - documentation packages: no special requirements
  - interpreter program packages: compatible interpreter must be available

- Binary packages (non "`Architecture: all`") usually face many road blocks and are **unsafe** to install.

  - library version compatibility (including "`libc`")
  - related utility program version compatibility
  - Kernel ABI compatibility
  - C++ ABI compatibility
  - ⋯

> **Note**
> Except to avoid broken package for a short term, installing binary packages from officially unsupported archives is generally bad idea. This is true even if you use apt-pinning (see Section 2.7.3). You should consider chroot or similar techniques (see Section 9.10) to run programs from different archives.

2.7.3 Tweaking candidate version

Without the "/etc/apt/preferences" file, APT system choses the latest available version as the **candidate version** using the version string. This is the normal state and most recommended usage of APT system. All officially supported combinations of archives do not require the "/etc/apt/preferences" file since some archives which should not be used as the automatic source of upgrades are marked as **NotAutomatic** and dealt properly.

Tip
The version string comparison rule can be verified with, e.g., "dpkg --compare-versions ver1.1 gt ver1.1~1; echo $?" (see dpkg(1)).

When you install packages from mixed source of archives (see Section 2.7.2) regularly, you can automate these complicated operations by creating the "/etc/apt/preferences" file with proper entries and tweaking the package selection rule for **candidate version** as described in apt_preferences(5). This is called **apt-pinning**.

⚠ **Warning**
Use of apt-pinning by a novice user is sure call for major troubles. You must avoid using apt-pinning except when you absolutely need it.

⚠ **Caution**
When using apt-pinning, you must ensure compatibility of packages by yourself since the Debian does not guarantee it. The apt-pinning is completely optional operation and its use is not something I encourage you to use.

⚠ **Caution**
Archive level Release files (see Section 2.5.3) are used for the rule of apt_preferences(5). Thus apt-pinning works only with "suite" name for normal Debian archives and security Debian archives. (This is different from Ubuntu archives.) For example, you can do "Pin: release a=unstable" but can not do "Pin: release a=sid" in the "/etc/apt/preferences" file.

⚠ **Caution**
When you use non-Debian archive as a part of apt-pinning, you should check what they are intended for and also check their credibility. For example, Ubuntu and Debian are not meant to be mixed.

Note
Even if you do not create the "/etc/apt/preferences" file, you can do fairly complex system operations (see Section 2.6.4 and Section 2.7.2) without apt-pinning.

Here is a simplified explanation of **apt-pinning** technique.

The APT system choses the highest Pin-Priority **upgrading** package from available package sources defined in the "/etc/apt/source
file as the **candidate version** package. If the Pin-Priority of the package is larger than 1000, this version restriction for **upgrading** is dropped to enable downgrading (see Section 2.7.10).

Pin-Priority value of each package is defined by "Pin-Priority" entries in the "/etc/apt/preferences" file or uses its default value.

The **target release** archive can be set by several methods.

Pin-Priority	apt-pinning effects to the package
1001	install the package even if this constitutes a downgrade of the package
990	used as the default for the **target release** archive
500	used as the default for the **normal** archive
100	used as the default for the **NotAutomatic** and **ButAutomaticUpgrades** archive
100	used for the **installed** package
1	used as the default for the **NotAutomatic** archive
-1	**never install** the package even if recommended

Table 2.18: List of notable Pin-Priority values for **apt-pinning** technique.

- "/etc/apt/apt.conf" configuration file with "APT::Default-Release "stable";" line

- command line option, e.g., "apt-get install -t testing some-package"

The **NotAutomatic** and **ButAutomaticUpgrades** archive is set by archive server having its archive level Release file (see Section 2.5.3) containing both "NotAutomatic: yes" and "ButAutomaticUpgrades: yes". The **NotAutomatic** archive is set by archive server having its archive level Release file containing only "NotAutomatic: yes".

The **apt-pinning situation** of <package> from multiple archive sources is displayed by "apt-cache policy <package>".

- A line started with "Package pin:" lists the package version of **pin** if association just with <package> is defined, e.g., "Package pin: 0.190".

- No line with "Package pin:" exists if no association just with <package> is defined.

- The Pin-Priority value associated just with <package> is listed right side of all version strings, e.g., "0.181 700".

- "0" is listed right side of all version strings if no association just with <package> is defined, e.g., "0.181 0".

- The Pin-Priority values of archives (defined as "Package: *" in the "/etc/apt/preferences" file) are listed left side of all archive paths, e.g., "100 http://deb.debian.org/debian/ stretch-backports/main Packages".

2.7.4 Updates and Backports

There are stretch-updates and backports.debian.org archives which provide updgrade packages for stable (stretch).

In order to use these archives, you list all required archives in the "/etc/apt/sources.list" file as the following.

```
deb http://deb.debian.org/debian/ stretch main
deb http://security.debian.org/ stretch/updates main
deb http://deb.debian.org/debian/ stretch-updates main
deb http://deb.debian.org/debian/ stretch-backports main
```

There is no need to set Pin-Priority value explicitly in the "/etc/apt/preferences" file. When newer packages become available, the default configuration provides most reasonable upgrades (see Section 2.5.3).

- All installed older packages are upgraded to newer ones from stretch-updates.

- Only manually installed older packages from stretch-backports are upgraded to newer ones from stretch-backports.

Whenever you wish to install a package named "<package-name>" with its dependency from stretch-backports archive manually, you use following command while switching target release with "-t" option.

```
$ sudo apt-get install -t stretch-backports <package-name>
```

2.7.5   Blocking packages installed by "Recommends"

If you wish not to pull in particular packages automatically by "Recommends", you must create the "`/etc/apt/preferences`" file and explicitly list all those packages at the top of it as the following.

```
Package: <package-1>
Pin: version *
Pin-Priority: -1

Package: <package-2>
Pin: version *
Pin-Priority: -1
```

2.7.6   Tracking `testing` with some packages from `unstable`

Here is an example of **apt-pinning** technique to include specific newer upstream version packages found in `unstable` regularly upgraded while tracking `testing`. You list all required archives in the "`/etc/apt/sources.list`" file as the following.

```
deb http://deb.debian.org/debian/ testing main
deb http://deb.debian.org/debian/ unstable main
deb http://security.debian.org/ testing/updates main
```

Set the "`/etc/apt/preferences`" file as the following.

```
Package: *
Pin: release a=unstable
Pin-Priority: 100
```

When you wish to install a package named "`<package-name>`" with its dependencies from `unstable` archive under this configuration, you issue the following command which switches target release with "`-t`" option (Pin-Priority of `unstable` becomes 990).

```
$ sudo apt-get install -t unstable <package-name>
```

With this configuration, usual execution of "`apt-get upgrade`" and "`apt-get dist-upgrade`" (or "`aptitude safe-upgr` and "`aptitude full-upgrade`") upgrades packages which were installed from `testing` archive using current `testing` archive and packages which were installed from `unstable` archive using current `unstable` archive.

> ⚠ **Caution**
> Be careful not to remove "`testing`" entry from the "`/etc/apt/sources.list`" file. Without "`testing`" entry in it, APT system upgrades packages using newer `unstable` archive.

Tip
I usually edit the "`/etc/apt/sources.list`" file to comment out "`unstable`" archive entry right after above operation. This avoids slow update process of having too many entries in the "`/etc/apt/sources.list`" file although this prevents upgrading packages which were installed from `unstable` archive using current `unstable` archive.

Tip
If "`Pin-Priority: 1`" is used instead of "`Pin-Priority: 100`" in the "`/etc/apt/preferences`" file, already installed packages having Pin-Priority value of 100 are not upgraded by `unstable` archive even if "`testing`" entry in the "`/etc/apt/sources.list`" file is removed.

If you wish to track particular packages in `unstable` automatically without initial "`-t unstable`" installation, you must create the "`/etc/apt/preferences`" file and explicitly list all those packages at the top of it as the following.

```
Package: <package-1>
Pin: release a=unstable
Pin-Priority: 700

Package: <package-2>
Pin: release a=unstable
Pin-Priority: 700
```

These set Pin-Priority value for each specific package. For example, in order to track the latest `unstable` version of this "Debian Reference" in English, you should have following entries in the "`/etc/apt/preferences`" file.

```
Package: debian-reference-en
Pin: release a=unstable
Pin-Priority: 700

Package: debian-reference-common
Pin: release a=unstable
Pin-Priority: 700
```

Tip
This apt-pinning technique is valid even when you are tracking `stable` archive. Documentation packages have been always safe to install from `unstable` archive in my experience, so far.

2.7.7 Tracking `unstable` with some packages from `experimental`

Here is another example of **apt-pinning** technique to include specific newer upstream version packages found in `experimental` while tracking `unstable`. You list all required archives in the "`/etc/apt/sources.list`" file as the following.

```
deb http://deb.debian.org/debian/ unstable main
deb http://deb.debian.org/debian/ experimental main
deb http://security.debian.org/ testing/updates main
```

The default Pin-Priority value for `experimental` archive is always 1 (<<100) since it is **NotAutomatic** archive (see Section 2.5.3). There is no need to set Pin-Priority value explicitly in the "`/etc/apt/preferences`" file just to use `experimental` archive unless you wish to track particular packages in it automatically for next upgrading.

2.7.8 Automatic download and upgrade of packages

The `apt` package comes with its own cron script "`/etc/cron.daily/apt`" to support the automatic download of packages. This script can be enhanced to perform the automatic upgrade of packages by installing the `unattended-upgrades` package. These can be customized by parameters in "`/etc/apt/apt.conf.d/02backup`" and "`/etc/apt/apt.conf.d/50unattended`" as described in "`/usr/share/doc/unattended-upgrades/README`".

The `unattended-upgrades` package is mainly intended for the security upgrade for the `stable` system. If the risk of breaking an existing `stable` system by the automatic upgrade is smaller than that of the system broken by the intruder using its security hole which has been closed by the security update, you should consider using this automatic upgrade with configuration parameters as the following.

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::Unattended-Upgrade "1";
```

If you are running an `unstable` system, you do not want to use the automatic upgrade since it certainly breaks system some day. Even for such `unstable` case, you may still want to download packages in advance to save time for the interactive upgrade with configuration parameters as the following.

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::Unattended-Upgrade "0";
```

2.7.9   Limiting download bandwidth for APT

If you want to limit the download bandwidth for APT to e.g. 800Kib/sec (=100kiB/sec), you should configure APT with its configuration parameter as the following.

```
APT::Acquire::http::Dl-Limit "800";
```

2.7.10   Emergency downgrading

> **Caution**
> Downgrading is not officially supported by the Debian by design. It should be done only as a part of emergency recovery process. Despite of this situation, it is known to work well in many incidents. For critical systems, you should backup all important data on the system after the recovery operation and re-install the new system from the scratch.

You may be lucky to downgrade from newer archive to older archive to recover from broken system upgrade by manipulating **candidate version** (see Section 2.7.3). This is lazy alternative to tedious actions of many "`dpkg -i <broken-package>_<old-vers` commands (see Section 2.6.4).

Search lines in the "`/etc/apt/sources.list`" file tracking `unstable` as the following.

```
deb http://deb.debian.org/debian/ sid main
```

Replace it with the following to track `testing`.

```
deb http://deb.debian.org/debian/ buster main
```

Set the "`/etc/apt/preferences`" file as the following.

```
Package: *
Pin: release a=testing
Pin-Priority: 1010
```

Run "`apt-get update; apt-get dist-upgrade`" to force downgrading of packages across the system.

Remove this special "`/etc/apt/preferences`" file after this emergency downgrading.

> **Tip**
> It is a good idea to remove (not purge!) as much packages to minimize dependency problems. You may need to manually remove and install some packages to get system downgraded. Linux kernel, bootloader, udev, PAM, APT, and networking related packages and their configuration files require special attention.

2.7.11   Who uploaded the package?

Although the maintainer name listed in "`/var/lib/dpkg/available`" and "`/usr/share/doc/package_name/changelog`" provide some information on "who is behind the packaging activity", the actual uploader of the package is somewhat obscure. who-uploads(1) in the `devscripts` package identifies the actual uploader of Debian source packages.

2.7.12   The equivs package

If you are to compile a program from source to replace the Debian package, it is best to make it into a real local debianized package (`*.deb`) and use private archive.

If you chose to compile a program from source and to install them under "`/usr/local`" instead, you may need to use `equivs` as a last resort to satisfy the missing package dependency.

```
Package: equivs
Priority: optional
Section: admin
Description: Circumventing Debian package dependencies
 This package provides a tool to create trivial Debian packages.
 Typically these packages contain only dependency information, but they
 can also include normal installed files like other packages do.
 .
 One use for this is to create a metapackage: a package whose sole
 purpose is to declare dependencies and conflicts on other packages so
 that these will be automatically installed, upgraded, or removed.
 .
 Another use is to circumvent dependency checking: by letting dpkg
 think a particular package name and version is installed when it
 isn't, you can work around bugs in other packages' dependencies.
 (Please do still file such bugs, though.)
```

2.7.13   Porting a package to the stable system

For partial upgrades of the `stable` system, rebuilding a package within its environment using the source package is desirable. This avoids massive package upgrades due to their dependencies.

Add the following entries to the "`/etc/apt/sources.list`" of a `stable` system.

```
deb-src http://deb.debian.org/debian unstable  main
```

Install required packages for the compilation and download the source package as the following.

```
# apt-get update
# apt-get dist-upgrade
# apt-get install fakeroot devscripts build-essential
$ apt-get build-dep foo
$ apt-get source foo
$ cd foo*
```

Update some tool chain packages such as `dpkg`, and `debhelper` from the backport packages if they are required for the backporting.

Execute the following.

```
$ dch -i
```

Bump package version, e.g. one appended with "`+bp1`" in "`debian/changelog`"

Build packages and install them to the system as the following.

```
$ debuild
$ cd ..
# debi foo*.changes
```

2.7.14 Proxy server for APT

Since mirroring whole subsection of Debian archive wastes disk space and network bandwidth, deployment of a local proxy server for APT is desirable consideration when you administer many systems on LAN. APT can be configure to use generic web (http) proxy servers such as squid (see Section 6.10) as described in apt.conf(5) and in "/usr/share/doc/apt/examples/configur The "$http_proxy" environment variable can be used to override proxy server setting in the "/etc/apt/apt.conf" file.

There are proxy tools specially for Debian archive. You should check BTS before using them.

package	popcon	size	description
approx	V:0, I:0	4729	caching proxy server for Debian archive files (compiled OCaml program)
apt-cacher	V:1, I:1	289	Caching proxy for Debian package and source files (Perl program)
apt-cacher-ng	V:4, I:5	1401	Caching proxy for distribution of software packages (compiled C++ program)

Table 2.19: List of the proxy tools specially for Debian archive

> **Caution**
> When Debian reorganizes its archive structure, these specialized proxy tools tend to require code rewrites by the package maintainer and may not be functional for a while. On the other hand, generic web (http) proxy servers are more robust and easier to cope with such changes.

2.7.15 Small public package archive

> **Tip**
> Setting up package archive manually is complicated. There are several repository management tools available. A comprehensive list is available online.

Here is a demo for manually creating a small public package archive compatible with the **secure APT** system (see Section 2.5.2). Let's assume few things.

- Account name: "foo"

- Host name: "www.example.com"

- Required packages: apt-utils, gnupg, and other packages

- URL: "http://www.example.com/~foo/" (→ "/home/foo/public_html/index.html")

- Architecture of packages: "amd64"

Create an APT archive key of Foo on your server system as the following.

```
$ ssh foo@www.example.com
$ gpg --gen-key
...
$ gpg -K
...
sec   1024D/3A3CB5A6 2008-08-14
uid                  Foo (ARCHIVE KEY) <foo@www.example.com>
ssb   2048g/6856F4A7 2008-08-14
$ gpg --export -a 3A3CB5A6 >foo.public.key
```

Publish the archive key file "`foo.public.key`" with the key ID "`3A3CB5A6`" for Foo

Create an archive tree called "Origin: Foo" as the following.

```
$ umask 022
$ mkdir -p ~/public_html/debian/pool/main
$ mkdir -p ~/public_html/debian/dists/unstable/main/binary-amd64
$ mkdir -p ~/public_html/debian/dists/unstable/main/source
$ cd ~/public_html/debian
$ cat > dists/unstable/main/binary-amd64/Release << EOF
Archive: unstable
Version: 4.0
Component: main
Origin: Foo
Label: Foo
Architecture: amd64
EOF
$ cat > dists/unstable/main/source/Release << EOF
Archive: unstable
Version: 4.0
Component: main
Origin: Foo
Label: Foo
Architecture: source
EOF
$ cat >aptftp.conf <<EOF
APT::FTPArchive::Release {
  Origin "Foo";
  Label "Foo";
  Suite "unstable";
  Codename "sid";
  Architectures "amd64";
  Components "main";
  Description "Public archive for Foo";
};
EOF
$ cat >aptgenerate.conf <<EOF
Dir::ArchiveDir ".";
Dir::CacheDir ".";
TreeDefault::Directory "pool/";
TreeDefault::SrcDirectory "pool/";
Default::Packages::Extensions ".deb";
Default::Packages::Compress ". gzip bzip2";
Default::Sources::Compress "gzip bzip2";
Default::Contents::Compress "gzip bzip2";

BinDirectory "dists/unstable/main/binary-amd64" {
  Packages "dists/unstable/main/binary-amd64/Packages";
  Contents "dists/unstable/Contents-amd64";
  SrcPackages "dists/unstable/main/source/Sources";
};

Tree "dists/unstable" {
  Sections "main";
  Architectures "amd64 source";
};
EOF
```

You can automate repetitive updates of APT archive contents on your server system by configuring `dupload`.

Place all package files into "`~foo/public_html/debian/pool/main/`" by executing "`dupload -t foo changes_file`" in client while having "`~/.dupload.conf`" containing the following.

```
$cfg{'foo'} = {
  fqdn => "www.example.com",
  method => "scpb",
  incoming => "/home/foo/public_html/debian/pool/main",
  # The dinstall on ftp-master sends emails itself
  dinstall_runs => 1,
};

$cfg{'foo'}{postupload}{'changes'} = "
  echo 'cd public_html/debian ;
  apt-ftparchive generate -c=aptftp.conf aptgenerate.conf;
  apt-ftparchive release -c=aptftp.conf dists/unstable >dists/unstable/Release ;
  rm -f dists/unstable/Release.gpg ;
  gpg -u 3A3CB5A6 -bao dists/unstable/Release.gpg dists/unstable/Release'|
  ssh foo@www.example.com  2>/dev/null ;
  echo 'Package archive created!'";
```

The **postupload** hook script initiated by dupload(1) creates updated archive files for each upload.

You can add this small public archive to the apt-line of your client system by the following.

```
$ sudo bash
# echo "deb http://www.example.com/~foo/debian/ unstable main" \
   >> /etc/apt/sources.list
# apt-key add foo.public.key
```

Tip
If the archive is located on the local filesystem, you can use "deb file:///home/foo/debian/ ⋯" instead.

2.7.16   Recording and copying system configuration

You can make a local copy of the package and debconf selection states by the following.

```
# dpkg --get-selections '*' > selection.dpkg
# debconf-get-selections    > selection.debconf
```

Here, "*" makes "selection.dpkg" to include package entries for "purge" too.

You can transfer these 2 files to another computer, and install there with the following.

```
# dselect update
# debconf-set-selections < myselection.debconf
# dpkg --set-selections  < myselection.dpkg
# apt-get -u dselect-upgrade    # or dselect install
```

If you are thinking about managing many servers in a cluster with practically the same configuration, you should consider to use specialized package such as fai to manage the whole system.

2.7.17   Converting or installing an alien binary package

alien(1) enables the conversion of binary packages provided in Red Hat rpm, Stampede slp, Slackware tgz, and Solaris pkg file formats into a Debian deb package. If you want to use a package from another Linux distribution than the one you have installed on your system, you can use alien to convert it from your preferred package format and install it. alien also supports LSB packages.

2.7.18   Extracting package without dpkg

The "`dpkg*.deb`" package contents can be extracted without using dpkg(1) on any Unix-like environment using standard ar(1) and tar(1).

```
# ar x /path/to/dpkg_<version>_<arch>.deb
# ls
total 24
-rw-r--r-- 1 bozo bozo  1320 2007-05-07 00:11 control.tar.gz
-rw-r--r-- 1 bozo bozo 12837 2007-05-07 00:11 data.tar.gz
-rw-r--r-- 1 bozo bozo     4 2007-05-07 00:11 debian-binary
# mkdir control
# mkdir data
# tar xvzf control.tar.gz -C control
# tar xvzf data.tar.gz -C data
```

The other "`*.deb`" package contents can be extracted by the dpkg-deb(1) command obtained from the "`dpkg*.deb`" package as above; or using standard ar(1) and newer GNU tar(1) with the xz(1) decompression support similarly as above.

You can also browse package content using the `mc` command.

2.7.19   More readings for the package management

You can learn more on the package management from following documentations.

- Primary documentations on the package management:

  - aptitude(8), dpkg(1), tasksel(8), apt(8), apt-get(8), apt-config(8), apt-key(8), sources.list(5), apt.conf(5), and apt_preferences(5);

  - "`/usr/share/doc/apt-doc/guide.html/index.html`" and "`/usr/share/doc/apt-doc/offline.html/in`" from the `apt-doc` package; and

  - "`/usr/share/doc/aptitude/html/en/index.html`" from the `aptitude-doc-en` package.

- Official and detailed documentations on the Debian archive:

  - "Debian Policy Manual Chapter 2 - The Debian Archive",

  - "Debian Developer's Reference, Chapter 4 - Resources for Debian Developers 4.6 The Debian archive", and

  - "The Debian GNU/Linux FAQ, Chapter 6 - The Debian FTP archives".

- Tutorial for building of a Debian package for Debian users:

  - "Debian New Maintainers' Guide" (deprecated).

  - "Guide for Debian Maintainers".

Chapter 3

The system initialization

It is wise for you as the system administrator to know roughly how the Debian system is started and configured. Although the exact details are in the source files of the packages installed and their documentations, it is a bit overwhelming for most of us.

I did my best to provide a quick overview of the key points of the Debian system and their configuration for your reference, based on the current and previous knowledge of mine and others. Since the Debian system is a moving target, the situation over the system may have been changed. Before making any changes to the system, you should refer to the latest documentation for each package.

Tip
bootup(7) describes the system bootup process based on `systemd` . (Recent Debian)

Tip
boot(7) describes the system bootup process based on UNIX System V Release 4. (Older Debian)

3.1   An overview of the boot strap process

The computer system undergoes several phases of boot strap processes from the power-on event until it offers the fully functional operating system (OS) to the user.

For simplicity, I limit discussion to the typical PC platform with the default installation.

The typical boot strap process is like a four-stage rocket. Each stage rocket hands over the system control to the next stage one.

- Section 3.1.1

- Section 3.1.2

- Section 3.1.3

- Section 3.1.4

Of course, these can be configured differently. For example, if you compiled your own kernel, you may be skipping the step with the mini-Debian system. So please do not assume this is the case for your system until you check it yourself.

Note
For non-legacy PC platform such as the SUN or the Macintosh system, the BIOS on ROM and the partition on the disk may be quite different (Section 9.5.2). Please seek the platform specific documentations elsewhere for such a case.

3.1.1   Stage 1: the BIOS

The BIOS is the 1st stage of the boot process which is started by the power-on event. The BIOS residing on the read only memory (ROM) is executed from the particular memory address to which the program counter of CPU is initialized by the power-on event.

This BIOS performs the basic initialization of the hardware (POST: power on self test) and hands the system control to the next step which you provide. The BIOS is usually provided with the hardware.

The BIOS startup screen usually indicates what key(s) to press to enter the BIOS setup screen to configure the BIOS behavior. Popular keys used are F1, F2, F10, Esc, Ins, and Del. If your BIOS startup screen is hidden by a nice graphics screen, you may press some keys such as Esc to disable this. These keys are highly dependent on the hardware.

The hardware location and the priority of the code started by the BIOS can be selected from the BIOS setup screen. Typically, the first few sectors of the first found selected device (hard disk, floppy disk, CD-ROM, ⋯) are loaded to the memory and this initial code is executed. This initial code can be any one of the following.

- The boot loader code

- The kernel code of the stepping stone OS such as FreeDOS

- The kernel code of the target OS if it fits in this small space

Typically, the system is booted from the specified partition of the primary hard disk partition. First 2 sectors of the hard disk on legacy PC contain the master boot record (MBR). The disk partition information including the boot selection is recorded at the end of this MBR. The first boot loader code executed from the BIOS occupies the rest of this MBR.

3.1.2   Stage 2: the boot loader

The boot loader is the 2nd stage of the boot process which is started by the BIOS. It loads the system kernel image and the initrd image to the memory and hands control over to them. This initrd image is the root filesystem image and its support depends on the bootloader used.

The Debian system normally uses the Linux kernel as the default system kernel. The initrd image for the current 2.6/3.x Linux kernel is technically the initramfs (initial RAM filesystem) image. The initramfs image is a gzipped cpio archive of files in the root filesystem.

> ⚠ **Warning**
> The above is no longer true with the new multi-segment initramfs. See Bug #790100.

The default install of the Debian system places first-stage GRUB boot loader code into the MBR for the PC platform. There are many boot loaders and configuration options available.

> ⚠ **Warning**
> Do not play with boot loaders without having bootable rescue media (USB memory stick, CD or floppy) created from images in the `grub-rescue-pc` package. It makes you boot your system even without functioning bootloader on the hard disk.

For GRUB Legacy, the menu configuration file is located at "`/boot/grub/menu.lst`". For example, it has entries as the following.

```
title           Debian GNU/Linux
root            (hd0,2)
kernel          /vmlinuz root=/dev/hda3 ro
initrd          /initrd.img
```

package	popcon	size	initrd	bootloader	description
grub-legacy	V:0, I:2	710	Supported	GRUB Legacy	This is smart enough to understand disk partitions and filesystems such as vfat, ext3, ···.
grub-pc	V:29, I:849	584	Supported	GRUB 2	This is smart enough to understand disk partitions and filesystems such as vfat, ext4, ···. (default)
grub-rescue-pc	V:0, I:1	6294	Supported	GRUB 2	This is GRUB 2 bootable rescue images (CD and floppy) (PC/BIOS version)
lilo	V:0, I:3	693	Supported	Lilo	This relies on the sector locations of data on the hard disk. (Old)
syslinux	V:5, I:59	299	Supported	Isolinux	This understands the ISO9660 filesystem. This is used by the boot CD.
syslinux	V:5, I:59	299	Supported	Syslinux	This understands the MSDOS filesystem (FAT). This is used by the boot floppy.
loadlin	V:0, I:1	84	Supported	Loadlin	New system is started from the FreeDOS/MSDOS system.
mbr	V:0, I:10	50	Not supported	MBR by Neil Turton	This is free software which substitutes MSDOS MBR. This only understands disk partitions.

Table 3.1: List of boot loaders

For GRUB 2, the menu configuration file is located at "/boot/grub/grub.cfg". It is automatically generated by "/usr/sbin/up
using templates from "/etc/grub.d/*" and settings from "/etc/default/grub". For example, it has entries as the fol-
lowing.

```
menuentry "Debian GNU/Linux" {
        set root=(hd0,3)
        linux /vmlinuz root=/dev/hda3
        initrd /initrd.img
}
```

For these examples, these GRUB parameters mean the following.

GRUB parameter	meaning
root	use 3rd partition on the primary disk by setting it as "(hd0,2)" in GRUB legacy or as "(hd0,3)" in GRUB 2
kernel	use kernel located at "/vmlinuz" with kernel parameter: "root=/dev/hda3 ro"
initrd	use initrd/initramfs image located at "/initrd.img"

Table 3.2: The meaning of GRUB parameters

Note
The value of the partition number used by GRUB legacy program is one less than normal one used by Linux kernel
and utility tools. GRUB 2 program fixes this problem.

Tip
UUID (see Section 9.5.3) may be used to identify a block special device instead of its file name such as
"/dev/hda3", e.g."root=UUID=81b289d5-4341-4003-9602-e254a17ac232 ro".

Tip
If GRUB is used, the kernel boot parameter is set in /boot/grub/grub.cfg. On Debian system, you should not edit /boot/grub/grub.cfg directly. You should edit the GRUB_CMDLINE_LINUX_DEFAULT value in /etc/default/grub and run update-grub(8) to update /boot/grub/grub.cfg.

Tip
You can start a boot loader from another boot loader using techniques called chain loading.

See "info grub" and grub-install(8).

3.1.3   Stage 3: the mini-Debian system

The mini-Debian system is the 3rd stage of the boot process which is started by the boot loader. It runs the system kernel with its root filesystem on the memory. This is an optional preparatory stage of the boot process.

Note
The term "the mini-Debian system" is coined by the author to describe this 3rd stage boot process for this document. This system is commonly referred as the initrd or initramfs system. Similar system on the memory is used by the Debian Installer.

The "/init" program is executed as the first program in this root filesystem on the memory. It is a program which initializes the kernel in user space and hands control over to the next stage. This mini-Debian system offers flexibility to the boot process such as adding kernel modules before the main boot process or mounting the root filesystem as an encrypted one.

• The "/init" program is a shell script program if initramfs was created by initramfs-tools. **You can interrupt this part of the boot process to gain root shell by providing "break=init" etc. to the kernel boot parameter. See the "/init" script for more break conditions. This shell environment is sophisticated enough to make a good inspection of your machine's hardware.** Commands available in this mini-Debian system are stripped down ones and mainly provided by a GNU tool called busybox(1).

• The "/init" program is a binary systemd program if initramfs was created by dracut. ** Commands available in this mini-Debian system are stripped down systemd(1) environment.

⚠ **Caution**
You need to use "-n" option for mount command when you are on the readonly root filesystem.

3.1.4   Stage 4: the normal Debian system

The normal Debian system is the 4th stage of the boot process which is started by the mini-Debian system. The system kernel for the mini-Debian system continues to run in this environment. The root filesystem is switched from the one on the memory to the one on the real hard disk filesystem.

The init program is executed as the first program with PID=1 to perform the main boot process of starting many programs. The default file path for the init program is "/sbin/init" but it can be changed by the kernel boot parameter as "init=/path/to/init_p

The default init program has been changing:

• Debian before squeeze uses the simple SysV-style init.

- Debian `wheezy` improves the SysV-style init by ordering the boot sequence with LSB header and starting boot scripts in parallel.

- Debian `jessie` switches its default init to the systemd for the event-driven and parallel initialization.

> **Tip**
> The actual init command on your system can be verified by the "`ps --pid 1 -f`" command.

> **Tip**
> "`/sbin/init`" is symlinked to "`/lib/systemd/systemd`" after Debian `jessie`.

package	popcon	size	description
systemd	V:703, I:805	11933	event-based init(8) daemon for concurrency (alternative to `sysvinit`)
systemd-sysv	V:684, I:798	112	the manual pages and links needed for `systemd` to replace `sysvinit`
systemd-cron	V:0, I:1	133	`systemd` units to provide `cron` daemon and `anacron` functionality
init-system-helpers	V:704, I:826	129	helper tools for switching between `sysvinit` and `systemd`
initscripts	V:284, I:667	205	scripts for initializing and shutting down the system
sysvinit-core	V:13, I:17	225	System-V-like init(8) utilities
sysv-rc	V:477, I:673	123	System-V-like runlevel change mechanism
sysvinit-utils	V:791, I:999	110	System-V-like utilities (startpar(8), bootlogd(8), ⋯)
lsb-base	V:877, I:999	49	Linux Standard Base 3.2 init script functionality
insserv	V:539, I:660	140	tool to organize boot sequence using LSB init.d script dependencies
uswsusp	V:4, I:11	699	tools to use userspace software suspend provided by Linux
kexec-tools	V:1, I:8	270	kexec tool for kexec(8) reboots (warm reboot)
systemd-bootchart	V:0, I:0	122	boot process performance analyser
bootchart2	V:0, I:1	94	boot process performance analyser
pybootchartgui	V:0, I:1	177	boot process performance analyser (visualisation)
mingetty	V:0, I:3	35	console-only getty(8)
mgetty	V:0, I:1	301	smart modem getty(8) replacement

Table 3.3: List of boot utilities for the Debian system

> **Tip**
> See Debian wiki: BootProcessSpeedup for the latest tips to speed up the boot process.

3.2 Systemd init

This section describes how system is started by the systemd(1) program with `PID=1` (i.e., init process).

The `systemd` init process spawns processes in parallel based on the unit configuration files (see systemd.unit(5)) which are written in declarative style instead of SysV-like procedural style. These are loaded from a set of paths (see systemd-system.conf(5)) as follows:

- "`/lib/systemd/system`": OS default configuration files

- "/etc/systemd/system": system administrator configuration files which override the OS default configuration files

- "/run/systemd/system": run-time generated configuration files which override the installed configuration files

Their inter-dependencies are specified by the directives "Wants=", "Requires=", "Before=", "After=", ⋯(see "MAP-PING OF UNIT PROPERTIES TO THEIR INVERSES" in systemd.unit(5)). The resource controls are also defined (see systemd.resource-control(5)).

The suffix of the unit configuration file encodes their types as:

- ***.service** describes the process controlled and supervised by systemd. See systemd.service(5).

- ***.device** describes the device exposed in the sysfs(5) as udev(7) device tree. See systemd.device(5).

- ***.mount** describes the file system mount point controlled and supervised by systemd. See systemd.mount(5).

- ***.automount** describes the file system auto mount point controlled and supervised by systemd. See systemd.automount(5).

- ***.swap** describes the swap device or file controlled and supervised by systemd. See systemd.swap(5).

- ***.path** describes the path monitored by systemd for path-based activation. See systemd.path(5).

- ***.socket** describes the socket controlled and supervised by systemd for socket-based activation. See systemd.socket(5).

- ***.timer** describes the timer controlled and supervised by systemd for timer-based activation. See systemd.timer(5).

- ***.slice** manages resources with the cgroups(7). See systemd.slice(5).

- ***.scope** is created programmatically using the bus interfaces of systemd to manages a set of system processes. See systemd.scope(5).

- ***.target** groups other unit configuration files to create the synchronization point during start-up. See systemd.target(5).

Upon system start up (i.e., init), the systemd process tries to start the "/lib/systemd/system/default.target (normally symlinked to "graphical.target"). First, some special target units (see systemd.special(7)) such as "local-fs.target", "swap.target" and "cryptsetup.target" are pulled in to mount the filesystems. Then, other target units are also pulled in by the target unit dependencies. For details, read bootup(7).

systemd offers backward compatibility features. SysV-style boot scripts in "/etc/init.d/rc[0123456S].d/[KS]<name>" are still parsed and telinit(8) is translated into systemd unit activation requests.

⚠ **Caution**
Emulated runlevel 2 to 4 are all symlinked to the same "multi-user.target".

3.2.1 The hostname

The kernel maintains the system **hostname**. The system unit started by systemd-hostnamed.service sets the system hostname at boot time to the name stored in "/etc/hostname". This file should contain **only** the system hostname, not a fully qualified domain name.

To print out the current hostname run hostname(1) without an argument.

3.2.2   The filesystem

The mount options of normal disk and network filesystems are set in "`/etc/fstab`". See fstab(5) and Section 9.5.7.

The configuration of the encrypted filesystem is set in "`/etc/crypttab`". See crypttab(5)

The configuration of software RAID with mdadm(8) is set in "`/etc/mdadm/mdadm.conf`". See mdadm.conf(5).

⚠ **Warning**

After mounting all the filesystems, temporary files in "`/tmp`", "`/var/lock`", and "`/var/run`" are cleaned for each boot up.

3.2.3   Network interface initialization

Network interfaces are typically initialized in "`networking.service`" for the `lo` interface and "`NetworkManager.service`" for other interfaces on modern Debian desktop system under `systemd`.

See Chapter 5 for how to configure them.

3.2.4   The kernel message

The kernel error message displayed to the console can be configured by setting its threshold level.

```
# dmesg -n3
```

error level value	error level name	meaning
0	KERN_EMERG	system is unusable
1	KERN_ALERT	action must be taken immediately
2	KERN_CRIT	critical conditions
3	KERN_ERR	error conditions
4	KERN_WARNING	warning conditions
5	KERN_NOTICE	normal but significant condition
6	KERN_INFO	informational
7	KERN_DEBUG	debug-level messages

Table 3.4: List of kernel error levels

3.2.5   The system message

Under `systemd`, both kernel and system messages are logged by the journal service `systemd-journald.service` (a.k.a `journald`) either into a persistent binary data below "`/var/log/journal`" or into a volatile binary data below "`/run/log/journ`". These binary log data are accessed by the journalctl(1) command.

Under `systemd`, the system logging utility rsyslogd(8) changes its behavior to read the volatile binary log data (instead of pre-systemd default "`/dev/log`") and to create traditional permanent ASCII system log data.

The system message can be customized by "`/etc/default/rsyslog`" and "`/etc/rsyslog.conf`" for both the log file and on-screen display. See rsyslogd(8) and rsyslog.conf(5). See also Section 9.2.2.

3.2.6 System management under systemd

The `systemd` offers not only init system but also generic system management functionalities such as journal logging, login management, time management, network management. etc..

The systemd(1) is managed by several commands:

- the systemctl(1) command controls the `systemd` system and service manager (CLI),

- the systemsdm(1) command controls the `systemd` system and service manager (GUI),

- the journalctl(1) command queries the `systemd` journal,

- the loginctl(1) command controls the `systemd` login manager, and

- the systemd-analyze(1) analyzes system boot-up performance.

Here are a list of typical `systemd` management command snippets. For the exact meanings, please read the pertinent manpages.

Here, "`$unit`" in the above examples may be a single unit name (suffix such as `.service` and `.target` are optional) or, in many cases, multiple unit specifications (shell-style globs "*", "?", "[]" using fnmatch(3) which will be matched against the primary names of all units currently in memory).

System state changing commands in the above examples are typically preceded by the "`sudo`" to attain the required administrative privilege.

The output of the "`systemctl status $unit|$PID|$device`" uses color of the dot ("●") to summarize the unit state at a glance.

- White "●" indicates an "inactive" or "deactivating" state.

- Red "●" indicates a "failed" or "error" state.

- Green "●" indicates an "active", "reloading" or "activating" state.

3.2.7 Customizing systemd

With default installation, many network services (see Chapter 6) are started as daemon processes after `network.target` at boot time by `systemd`. The "sshd" is no exception. Let's change this to on-demand start of "sshd" as a customization example.

First, disable system installed service unit.

```
$ sudo systemctl stop sshd.service
$ sudo systemctl mask sshd.service
```

The on-demand socket activation system of the classic Unix services was through the `indetd` superserver. Under `systemd`, the equivalent can be enabled by adding ***.socket** and ***.service** unit configuration files.

`sshd.socket` for specifying a socket to listen on

```
[Unit]
Description=SSH Socket for Per-Connection Servers

[Socket]
ListenStream=22
Accept=yes

[Install]
WantedBy=sockets.target
```

Operation	Type	Command snippets
GUI for service manager	GUI	"systemadm" (systemd-ui package)
List all target unit configuration	Unit	"systemctl list-units --type=target"
List all service unit configuration	Unit	"systemctl list-units --type=service"
List all unit configuration types	Unit	"systemctl list-units --type=help"
List all socket units in memory	Unit	"systemctl list-sockets"
List all timer units in memory	Unit	"systemctl list-timers"
Start "$unit"	Unit	"systemctl start $unit"
Stop "$unit"	Unit	"systemctl stop $unit"
Reload service-specific configuration	Unit	"systemctl reload $unit"
Stop and start all "$unit"	Unit	"systemctl restart $unit"
Start "$unit" and stop all others	Unit	"systemctl isolate $unit"
Switch to "graphical" (GUI system)	Unit	"systemctl isolate graphical"
Switch to "multi-user" (CLI system)	Unit	"systemctl isolate multi-user"
Switch to "rescue" (single user CLI system)	Unit	"systemctl isolate rescue"
Send kill signal to "$unit"	Unit	"systemctl kill $unit"
Send kill signal to "$unit"	Unit	"systemctl kill $unit"
Check if "$unit" service is active	Unit	"systemctl is-active $unit"
Check if "$unit" service is failed	Unit	"systemctl is-failed $unit"
Check status of "$unit|$PID|device"	Unit	"systemctl status $unit|$PID|$device"
Show properties of "$unit|$job"	Unit	"systemctl show $unit|$job"
Reset failed "$unit"	Unit	"systemctl reset-failed $unit"
List dependency of all unit services	Unit	"systemctl list-dependencies --all"
List unit files installed on the system	Unit file	"systemctl list-unit-files"
Enable "$unit" (add symlink)	Unit file	"systemctl enable $unit"
Disable "$unit" (remove symlink)	Unit file	"systemctl disable $unit"
Unmask "$unit" (remove symlink to "/dev/null")	Unit file	"systemctl unmask $unit"
Mask "$unit" (add symlink to "/dev/null")	Unit file	"systemctl mask $unit"
Get default-target setting	Unit file	"systemctl get-default"
Set default-target to "graphical" (GUI system)	Unit file	"systemctl set-default graphical"
Set default-target to "multi-user" (CLI system)	Unit file	"systemctl set-default multi-user"
Show job environment	Environment	"systemctl show-environment"
Set job environment "variable" to "value"	Environment	"systemctl set-environment variable=value"
Unset job environment "variable"	Environment	"systemctl unset-environment variable"
Reload all unit files and daemons	Lifecycle	"systemctl daemon-reload"
Shut down the system	System	"systemctl poweroff"
Shut down and reboot the system	System	"systemctl reboot"
Suspend the system	System	"systemctl suspend"
Hibernate the system	System	"systemctl hibernate"
View job log of "$unit"	Journal	"journalctl -u $unit"
View job log of "$unit" ("tail -f" style)	Journal	"journalctl -u $unit -f"
Show time spent for each initialization steps	Analyze	"systemd-analyze time"

`sshd@.service` as the matching service file of `sshd.socket`

```
[Unit]
Description=SSH Per-Connection Server

[Service]
ExecStart=-/usr/sbin/sshd -i
StandardInput=socket
```

Then reload.

```
 $ sudo systemctl daemon-reload
```

3.3 The udev system

For Linux kernel 2.6 and newer, the udev system provides mechanism for the automatic hardware discovery and initialization (see udev(7)). Upon discovery of each device by the kernel, the udev system starts a user process which uses information from the sysfs filesystem (see Section 1.2.12), loads required kernel modules supporting it using the modprobe(8) program (see Section 3.3.1), and creates corresponding device nodes.

Tip
If "/lib/modules/<kernel-version>/modules.dep" was not generated properly by depmod(8) for some reason, modules may not be loaded as expected by the udev system. Execute "depmod -a" to fix it.

The name of device nodes can be configured by udev rule files in "/etc/udev/rules.d/". Current default rules tend to create dynamically generated names resulting non-static device names except for cd and network devices. By adding your custom rules similar to what cd and network devices do, you can generate static device names for other devices such as USB memory sticks, too. See "Writing udev rules" or "/usr/share/doc/udev/writing_udev_rules/index.html".

Since the udev system is somewhat a moving target, I leave details to other documentations and describe the minimum information here.

Tip
For mounting rules in "/etc/fstab", device nodes do not need to be static ones. You can use UUID to mount devices instead of device names such as "/dev/sda". See Section 9.5.3.

3.3.1 The kernel module initialization

The modprobe(8) program enables us to configure running Linux kernel from user process by adding and removing kernel modules. The udev system (see Section 3.3) automates its invocation to help the kernel module initialization.

There are non-hardware modules and special hardware driver modules as the following which need to be pre-loaded by listing them in the "/etc/modules" file (see modules(5)).

- TUN/TAP modules providing virtual Point-to-Point network device (TUN) and virtual Ethernet network device (TAP),

- netfilter modules providing netfilter firewall capabilities (iptables(8), Section 5.10), and

- watchdog timer driver modules.

The configuration files for the modprobe(8) program are located under the "/etc/modprobes.d/" directory as explained in modprobe.conf(5). (If you want to avoid some kernel modules to be auto-loaded, consider to blacklist them in the "/etc/modprobes.d" file.)

The "/lib/modules/<version>/modules.dep" file generated by the depmod(8) program describes module dependencies used by the modprobe(8) program.

Note
If you experience module loading issues with boot time module loading or with modprobe(8), "depmod -a" may resolve these issues by reconstructing "modules.dep".

The modinfo(8) program shows information about a Linux kernel module.

The lsmod(8) program nicely formats the contents of the "/proc/modules", showing what kernel modules are currently loaded.

Tip
You can identify exact hardware on your system. See Section 9.4.3.

Tip
You may configure hardware at boot time to activate expected hardware features. See Section 9.4.4.

Tip
You can probably add support for your special device by recompiling the kernel. See Section 9.9.

Chapter 4

Authentication

When a person (or a program) requests access to the system, authentication confirms the identity to be a trusted one.

⚠ **Warning**
Configuration errors of PAM may lock you out of your own system. You must have a rescue CD handy or setup an alternative boot partition. To recover, boot the system with them and correct things from there.

⚠ **Warning**
This chapter is getting outdated since this is based on Debian 7.0 (`wheezy`) released in 2013.

4.1 Normal Unix authentication

Normal Unix authentication is provided by the pam_unix(8) module under the PAM (Pluggable Authentication Modules). Its 3 important configuration files, with ":" separated entries, are the following.

file	permission	user	group	description
/etc/passwd	-rw-r--r--	root	root	(sanitized) user account information
/etc/shadow	-rw-r-----	root	shadow	secure user account information
/etc/group	-rw-r--r--	root	root	group information

Table 4.1: 3 important configuration files for pam_unix(8)

"/etc/passwd" contains the following.

```
 ...
user1:x:1000:1000:User1 Name,,,:/home/user1:/bin/bash
user2:x:1001:1001:User2 Name,,,:/home/user2:/bin/bash
 ...
```

As explained in passwd(5), each ":" separated entry of this file means the following.

• Login name

• Password specification entry

- Numerical user ID

- Numerical group ID

- User name or comment field

- User home directory

- Optional user command interpreter

The second entry of "`/etc/passwd`" was used for the encrypted password entry. After the introduction of "`/etc/shadow`", this entry is used for the password specification entry.

content	meaning
(empty)	passwordless account
x	the encrypted password is in "`/etc/shadow`"
*	no login for this account
!	no login for this account

Table 4.2: The second entry content of "`/etc/passwd`"

"`/etc/shadow`" contains the following.

```
...
user1:$1$Xop0FYH9$IfxyQwBe9b8tiyIkt2P4F/:13262:0:99999:7:::
user2:$1$vXGZLVbS$ElyErNf/agUDsm1DehJMS/:13261:0:99999:7:::
...
```

As explained in shadow(5), each "`:`" separated entry of this file means the following.

- Login name

- Encrypted password (The initial "`$1$`" indicates use of the MD5 encryption. The "*" indicates no login.)

- Date of the last password change, expressed as the number of days since Jan 1, 1970

- Number of days the user will have to wait before she will be allowed to change her password again

- Number of days after which the user will have to change her password

- Number of days before a password is going to expire during which the user should be warned

- Number of days after a password has expired during which the password should still be accepted

- Date of expiration of the account, expressed as the number of days since Jan 1, 1970

- …

"`/etc/group`" contains the following.

```
group1:x:20:user1,user2
```

As explained in group(5), each "`:`" separated entry of this file means the following.

- Group name

- Encrypted password (not really used)

- Numerical group ID

- "," separated list of user names

> **Note**
> "/etc/gshadow" provides the similar function as "/etc/shadow" for "/etc/group" but is not really used.

> **Note**
> The actual group membership of a user may be dynamically added if "auth optional pam_group.so" line is added to "/etc/pam.d/common-auth" and set it in "/etc/security/group.conf". See pam_group(8).

> **Note**
> The   base-passwd   package   contains   an   authoritative   list   of   the   user   and   the   group: "/usr/share/doc/base-passwd/users-and-groups.html".

4.2 Managing account and password information

Here are few notable commands to manage account information.

command	function
getent passwd <user_name>	browse account information of "<user_name>"
getent shadow <user_name>	browse shadowed account information of "<user_name>"
getent group <group_name>	browse group information of "<group_name>"
passwd	manage password for the account
passwd -e	set one-time password for the account activation
chage	manage password aging information

Table 4.3: List of commands to manage account information

You may need to have the root privilege for some functions to work. See crypt(3) for the password and data encryption.

> **Note**
> On the system set up with PAM and NSS as the Debian salsa machine, the content of local "/etc/passwd", "/etc/group" and "/etc/shadow" may not be actively used by the system. Above commands are valid even under such environment.

4.3 Good password

When creating an account during your system installation or with the passwd(1) command, you should choose a good password which consists of at least 6 to 8 characters including one or more characters from each of the following sets according to passwd(1).

- Lower case alphabetics
- Digits 0 through 9
- Punctuation marks

> **Warning**
> Do not choose guessable words for the password. Account name, social security number, phone number, address, birthday, name of your family members or pets, dictionary words, simple sequence of characters such as "12345" or "qwerty", ···are all bad choice for the password.

4.4 Creating encrypted password

There are independent tools to generate encrypted passwords with salt.

package	popcon	size	command	function
whois	V:48, I:630	338	mkpasswd	over-featured front end to the crypt(3) library
openssl	V:794, I:991	1303	openssl passwd	compute password hashes (OpenSSL). passwd(1ssl)

Table 4.4: List of tools to generate password

4.5 PAM and NSS

Modern Unix-like systems such as the Debian system provide PAM (Pluggable Authentication Modules) and NSS (Name Service Switch) mechanism to the local system administrator to configure his system. The role of these can be summarizes as the following.

- PAM offers a flexible authentication mechanism used by the application software thus involves password data exchange.

- NSS offers a flexible name service mechanism which is frequently used by the C standard library to obtain the user and group name for programs such as ls(1) and id(1).

These PAM and NSS systems need to be configured consistently.

The notable packages of PAM and NSS systems are the following.

package	popcon	size	description
libpam-modules	V:803, I:999	870	Pluggable Authentication Modules (basic service)
libpam-ldap	V:1, I:15	244	Pluggable Authentication Module allowing LDAP interfaces
libpam-cracklib	I:16	114	Pluggable Authentication Module to enable cracklib support
libpam-systemd	V:408, I:685	353	Pluggable Authentication Module to register user sessions for logind
libpam-doc	I:2	1020	Pluggable Authentication Modules (documentation in html and text)
libc6	V:930, I:999	11401	GNU C Library: Shared libraries which also provides "Name Service Switch" service
glibc-doc	I:13	2918	GNU C Library: Manpages
glibc-doc-reference	I:5	13132	GNU C Library: Reference manual in info, pdf and html format
libnss-mdns	I:568	107	NSS module for Multicast DNS name resolution
libnss-ldap	I:14	255	NSS module for using LDAP as a naming service
libnss-ldapd	I:19	146	NSS module for using LDAP as a naming service (new fork of libnss-ldap)

Table 4.5: List of notable PAM and NSS systems

- "The Linux-PAM System Administrators' Guide" in `libpam-doc` is essential for learning PAM configuration.

- "System Databases and Name Service Switch" section in `glibc-doc-reference` is essential for learning NSS configuration.

> **Note**
> You can see more extensive and current list by "`aptitude search 'libpam-|libnss-'`" command. The acronym NSS may also mean "Network Security Service" which is different from "Name Service Switch".

> **Note**
> PAM is the most basic way to initialize environment variables for each program with the system wide default value.

Under systemd, `libpam-systemd` package is installed to manage user logins by registering user sessions in the systemd control group hierarchy for logind. See systemd-logind(8), logind.conf(5), and pam_systemd(8).

4.5.1 Configuration files accessed by PAM and NSS

Here are a few notable configuration files accessed by PAM and NSS.

configuration file	function
`/etc/pam.d/<program_name>`	set up PAM configuration for the "`<program_name>`" program; see pam(7) and pam.d(5)
`/etc/nsswitch.conf`	set up NSS configuration with the entry for each service. See nsswitch.conf(5)
`/etc/nologin`	limit the user login by the pam_nologin(8) module
`/etc/securetty`	limit the tty for the root access by the pam_securetty(8) module
`/etc/security/access.conf`	set access limit by the pam_access(8) module
`/etc/security/group.conf`	set group based restraint by the pam_group(8) module
`/etc/security/pam_env.conf`	set environment variables by the pam_env(8) module
`/etc/environment`	set additional environment variables by the pam_env(8) module with the "`readenv=1`" argument
`/etc/default/locale`	set locale by pam_env(8) module with the "`readenv=1 envfile=/etc/default/locale`" argument (Debian)
`/etc/security/limits.conf`	set resource restraint (ulimit, core, ···) by the pam_linits(8) module
`/etc/security/time.conf`	set time restraint by the pam_time(8) module
`/etc/systemd/logind.conf`	set systemd login manager configuration (see logind.conf(5) and systemd-logind.service(8))

Table 4.6: List of configuration files accessed by PAM and NSS

The limitation of the password selection is implemented by the PAM modules, pam_unix(8) and pam_cracklib(8). They can be configured by their arguments.

> **Tip**
> PAM modules use suffix ".so" for their filenames.

4.5.2 The modern centralized system management

The modern centralized system management can be deployed using the centralized Lightweight Directory Access Protocol (LDAP) server to administer many Unix-like and non-Unix-like systems on the network. The open source implementation of the Lightweight Directory Access Protocol is OpenLDAP Software.

The LDAP server provides the account information through the use of PAM and NSS with `libpam-ldap` and `libnss-ldap` packages for the Debian system. Several actions are required to enable this (I have not used this setup and the following is purely secondary information. Please read this in this context.).

- You set up a centralized LDAP server by running a program such as the stand-alone LDAP daemon, slapd(8).

- You change the PAM configuration files in the ”/etc/pam.d/” directory to use ”pam_ldap.so” instead of the default ”pam_unix.so”.

    – Debian uses ”/etc/pam_ldap.conf” as the configuration file for libpam-ldap and ”/etc/pam_ldap.secret” as the file to store the password of the root.

- You change the NSS configuration in the ”/etc/nsswitch.conf” file to use ”ldap” instead of the default (”compat” or ”file”).

    – Debian uses ”/etc/libnss-ldap.conf” as the configuration file for libnss-ldap.

- You must make libpam-ldap to use SSL (or TLS) connection for the security of password.

- You may make libnss-ldap to use SSL (or TLS) connection to ensure integrity of data at the cost of the LDAP network overhead.

- You should run nscd(8) locally to cache any LDAP search results in order to reduce the LDAP network traffic.

See documentations in pam_ldap.conf(5) and ”/usr/share/doc/libpam-doc/html/” offered by the libpam-doc package and ”info libc 'Name Service Switch'” offered by the glibc-doc package.

Similarly, you can set up alternative centralized systems with other methods.

- Integration of user and group with the Windows system.

    – Access Windows domain services by the winbind and libpam_winbind packages.
    – See winbindd(8) and Integrating MS Windows Networks with Samba.

- Integration of user and group with the legacy Unix-like system.

    – Access NIS (originally called YP) or NIS+ by the nis package.
    – See The Linux NIS(YP)/NYS/NIS+ HOWTO.

4.5.3   ”Why GNU su does not support the wheel group”

This is the famous phrase at the bottom of the old ”info su” page by Richard M. Stallman. Not to worry: the current su command in Debian uses PAM, so that one can restrict the ability to use su to the root group by enabling the line with ”pam_wheel.so” in ”/etc/pam.d/su”.

4.5.4   Stricter password rule

Installing the libpam-cracklib package enables you to force stricter password rule, for example, by having active lines in ”/etc/pam.d/common-password” as the following.

For squeeze:

```
password required pam_cracklib.so retry=3 minlen=9 difok=3
password [success=1 default=ignore] pam_unix.so use_authtok nullok md5
password requisite pam_deny.so
password required pam_permit.so
```

4.6   Other access controls

Note
See Section 9.3.15 for restricting the kernel secure attention key (SAK) feature.

4.6.1 **sudo**

sudo(8) is a program designed to allow a sysadmin to give limited root privileges to users and log root activity. `sudo` requires only an ordinary user's password. Install `sudo` package and activate it by setting options in "`/etc/sudoers`". See configuration example at "`/usr/share/doc/sudo/examples/sudoers`" and Section 1.1.12.

My usage of `sudo` for the single user system (see Section 1.1.12) is aimed to protect myself from my own stupidity. Personally, I consider using `sudo` a better alternative than using the system from the root account all the time. For example, the following changes the owner of "`<some_file>`" to "`<my_name>`".

```
$ sudo chown <my_name> <some_file>
```

Of course if you know the root password (as self-installed Debian users do), any command can be run under root from any user's account using "`su -c`".

4.6.2 **PolicyKit**

PolicyKit is an operating system component for controlling system-wide privileges in Unix-like operating systems.

Newer GUI applications are not designed to run as privileged processes. They talk to privileged processes via PolicyKit to perform administrative operations.

PolicyKit limits such operations to user accounts belonging to the `sudo` group on the Debian system.

See polkit(8).

4.6.3 **SELinux**

Security-Enhanced Linux (SELinux) is a framework to tighten privilege model tighter than the ordinary Unix-like security model with the mandatory access control (MAC) policies. The root power may be restricted under some conditions.

4.6.4 **Restricting access to some server services**

For system security, it is a good idea to disable as much server programs as possible. This becomes critical for network servers. Having unused servers, activated either directly as daemon or via super-server program, are considered security risks.

Many programs, such as sshd(8), use PAM based access control. There are many ways to restrict access to some server services.

- configuration files: "`/etc/default/<program_name>`"

- service unit configuration for daemon

- PAM (Pluggable Authentication Modules)

- "`/etc/inetd.conf`" for super-server

- "`/etc/hosts.deny`" and "`/etc/hosts.allow`" for TCP wrapper, tcpd(8)

- "`/etc/rpc.conf`" for Sun RPC

- "`/etc/at.allow`" and "`/etc/at.deny`" for atd(8)

- "`/etc/cron.allow`" and "`/etc/cron.deny`" for crontab(1)

- Network firewall of netfilter infrastructure

See Section 3.2.6, Section 4.5.1, and Section 5.10.

Tip
Sun RPC services need to be active for NFS and other RPC based programs.

Tip
If you have problems with remote access in a recent Debian system, comment out offending configuration such as "ALL: PARANOID" in "`/etc/hosts.deny`" if it exists. (But you must be careful on security risks involved with this kind of action.)

4.7 Security of authentication

Note
The information here **may not be sufficient** for your security needs but it should be a **good start**.

4.7.1 Secure password on the Internet

Many popular transportation layer services communicate messages including password authentication in the plain text. It is very bad idea to transmit password in the plain text over the wild Internet where it can be intercepted. You can run these services over "Transport Layer Security" (TLS) or its predecessor, "Secure Sockets Layer" (SSL) to secure entire communication including password by the encryption.

insecure service name	port	secure service name	port
www (http)	80	https	443
smtp (mail)	25	ssmtp (smtps)	465
ftp-data	20	ftps-data	989
ftp	21	ftps	990
telnet	23	telnets	992
imap2	143	imaps	993
pop3	110	pop3s	995
ldap	389	ldaps	636

Table 4.7: List of insecure and secure services and ports

The encryption costs CPU time. As a CPU friendly alternative, you can keep communication in plain text while securing just the password with the secure authentication protocol such as "Authenticated Post Office Protocol" (APOP) for POP and "Challenge-Response Authentication Mechanism MD5" (CRAM-MD5) for SMTP and IMAP. (For sending mail messages over the Internet to your mail server from your mail client, it is recently popular to use new message submission port 587 instead of traditional SMTP port 25 to avoid port 25 blocking by the network provider while authenticating yourself with CRAM-MD5.)

4.7.2 Secure Shell

The Secure Shell (SSH) program provides secure encrypted communications between two untrusted hosts over an insecure network with the secure authentication. It consists of the OpenSSH client, ssh(1), and the OpenSSH daemon, sshd(8). This SSH can be used to tunnel an insecure protocol communication such as POP and X securely over the Internet with the port forwarding feature.

The client tries to authenticate itself using host-based authentication, public key authentication, challenge-response authentication, or password authentication. The use of public key authentication enables the remote password-less login. See Section 6.9.

4.7.3   Extra security measures for the Internet

Even when you run secure services such as Secure Shell (SSH) and Point-to-point tunneling protocol (PPTP) servers, there are still chances for the break-ins using brute force password guessing attack etc. from the Internet. Use of the firewall policy (see Section 5.10) together with the following secure tools may improve the security situation.

package	popcon	size	description
knockd	V:0, I:2	89	small port-knock daemon knockd(1) and client konck(1)
fail2ban	V:97, I:108	1731	ban IPs that cause multiple authentication errors
libpam-shield	V:0, I:0	107	lock out remote attackers trying password guessing

Table 4.8: List of tools to provide extra security measures

4.7.4   Securing the root password

To prevent people to access your machine with root privilege, you need to make following actions.

- Prevent physical access to the hard disk

- Lock BIOS and prevent booting from the removable media

- Set password for GRUB interactive session

- Lock GRUB menu from editing

With physical access to hard disk, resetting the password is relatively easy with following steps.

1. Move the hard disk to a PC with CD bootable BIOS.

2. Boot system with a rescue media (Debian boot disk, Knoppix CD, GRUB CD, ⋯).

3. Mount root partition with read/write access.

4. Edit "/etc/passwd" in the root partition and make the second entry for the root account empty.

If you have edit access to the GRUB menu entry (see Section 3.1.2) for grub-rescue-pc at boot time, it is even easier with following steps.

1. Boot system with the kernel parameter changed to something like "root=/dev/hda6 rw init=/bin/sh".

2. Edit "/etc/passwd" and make the second entry for the root account empty.

3. Reboot system.

The root shell of the system is now accessible without password.

Note
Once one has root shell access, he can access everything on the system and reset any passwords on the system. Further more, he may compromise password for all user accounts using brute force password cracking tools such as john and crack packages (see Section 9.4.11). This cracked password may lead to compromise other systems.

The only reasonable software solution to avoid all these concerns is to use software encrypted root partition (or "/etc" partition) using dm-crypt and initramfs (see Section 9.8). You always need password to boot the system, though.

Chapter 5

Network setup

Tip
For general guide to the GNU/Linux networking, read the Linux Network Administrators Guide.

Tip
For modern Debian specific guide to the networking, read The Debian Administrator's Handbook —Configuring the Network.

> ⚠ **Warning**
> Instead of using the traditional interface naming scheme ("eth0", "eth1", "wlan0", ···), the newer systemd uses "Predictable Network Interface Names" such as "enp0s25".

> ⚠ **Warning**
> This chapter is getting outdated since this is based on Debian 7.0 (Wheezy) released in 2013.

Tip
Although this document still uses old ifconfig(8) with IPv4 for its network configuration examples, Debian is moving to ip(8) with IPv4+IPv6 in the wheezy release. Patches to update this document are welcomed.

Tip
Under systemd, networkd may be used to manage networks. See systemd-networkd(8).

5.1 The basic network infrastructure

Let's review the basic network infrastructure on the modern Debian system.

packages	popcon	size	type	description
ifupdown	V:581, I:994	203	config::ifupdown	standardized tool to bring up and down the network (Debian specific)
ifplugd	V:4, I:20	214	,,	manage the wired network automatically
ifupdown-extra	V:0, I:1	100	,,	network testing script to enhance "ifupdown" package
ifmetric	V:0, I:1	36	,,	set routing metrics for a network interface
guessnet	V:0, I:0	422	,,	mapping script to enhance "ifupdown" package via "/etc/network/interfaces" file
ifscheme	V:0, I:0	58	,,	mapping scripts to enhance "ifupdown" package
network-manager	V:376, I:481	10583	config::NM	NetworkManager (daemon): manage the network automatically
network-manager-gnome	V:182, I:419	5831	,,	NetworkManager (GNOME frontend)
wicd	I:33	35	config::wicd	wired and wireless network manager (metapackage)
wicd-cli	V:0, I:2	59	,,	wired and wireless network manager (command-line client)
wicd-curses	V:1, I:4	175	,,	wired and wireless network manager (Curses client)
wicd-daemon	V:28, I:37	954	,,	wired and wireless network manager (daemon)
wicd-gtk	V:22, I:34	574	,,	wired and wireless network manager (GTK+ client)
iptables	V:256, I:995	1543	config::Netfilter	administration tools for packet filtering and NAT (Netfilter)
iproute2	V:587, I:819	2057	config::iproute2	iproute2, IPv6 and other advanced network configuration: ip(8), tc(8), etc
ifrename	V:0, I:2	122	,,	rename network interfaces based on various static criteria: ifrename(8)
ethtool	V:109, I:260	375	,,	display or change Ethernet device settings
iputils-ping	V:253, I:997	111	test::iproute2	test network reachability of a remote host by hostname or IP address (iproute2)
iputils-arping	V:40, I:441	58	,,	test network reachability of a remote host specified by the ARP address
iputils-tracepath	V:8, I:148	75	,,	trace the network path to a remote host
net-tools	V:333, I:810	932	config::net-tools	NET-3 networking toolkit (net-tools, IPv4 network configuration): ifconfig(8) etc.
inetutils-ping	V:0, I:1	336	test::net-tools	test network reachability of a remote host by hostname or IP address (legacy, GNU)
arping	V:2, I:27	72	,,	test network reachability of a remote host specified by the ARP address (legacy)
traceroute	V:67, I:975	154	,,	trace the network path to a remote host (legacy, console)
isc-dhcp-client	V:278, I:969	649	config::low-level	DHCP client
wpasupplicant	V:309, I:551	2668	,,	client support for WPA and WPA2 (IEEE 802.11i)
wpagui	V:0, I:3	762	,,	Qt GUI client for wpa_supplicant
wireless-tools	V:171, I:281	286	,,	tools for manipulating Linux Wireless Extensions
ppp	V:200, I:522	940	,,	PPP/PPPoE connection with chat
pppoeconf	V:0, I:10	290	config::helper	configuration helper for PPPoE connection
pppconfig	V:1, I:3	805	,,	configuration helper for PPP connection with chat
wvdial	V:0, I:7	276	,,	configuration helper for PPP connection with wvdial and ppp
mtr-tiny	V:6, I:56	140	test::low-level	trace the network path to a remote host (curses)
mtr	V:5, I:40	190	,,	trace the network path to a remote host (curses and GTK+)
gnome-nettool	V:5, I:138	2101	,,	tools for common network information operations (GNOME)
nmap	V:39, I:327	4673	,,	network mapper / port scanner (Nmap, console)
zenmap	V:4, I:12	2929	,,	network mapper / port scanner (GTK+)
tcpdump	V:21, I:197	1192	,,	network traffic analyzer (Tcpdump, console)
wireshark	I:63	67	,,	network traffic analyzer (Wireshark, GTK+)
tshark	V:3, I:36	383	,,	network traffic analyzer (console)
tcptrace	V:0, I:1	392	,,	produce a summarization of the connections from tcpdump output

5.1.1   The hostname resolution

The hostname resolution is currently supported by the NSS (Name Service Switch) mechanism too. The flow of this resolution is the following.

1. The "/etc/nsswitch.conf" file with stanza like "hosts: files dns" dictates the hostname resolution order. (This replaces the old functionality of the "order" stanza in "/etc/host.conf".)

2. The files method is invoked first. If the hostname is found in the "/etc/hosts" file, it returns all valid addresses for it and exits. (The "/etc/host.conf" file contains "multi on".)

3. The dns method is invoked. If the hostname is found by the query to the Internet Domain Name System (DNS) identified by the "/etc/resolv.conf" file, it returns all valid addresses for it and exits.

For example, "/etc/hosts" looks like the following.

```
127.0.0.1 localhost
127.0.1.1 <host_name>

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

Each line starts with a IP address and it is followed by the associated hostname.

The IP address 127.0.1.1 in the second line of this example may not be found on some other Unix-like systems. The Debian Installer creates this entry for a system without a permanent IP address as a workaround for some software (e.g., GNOME) as documented in the bug #719621.

The <host_name> matches the hostname defined in the "/etc/hostname".

For a system with a permanent IP address, that permanent IP address should be used here instead of 127.0.1.1.

For a system with a permanent IP address and a fully qualified domain name (FQDN) provided by the Domain Name System (DNS), that canonical <host_name>.<domain_name> should be used instead of just <host_name>.

The "/etc/resolv.conf" is a static file if the resolvconf package is not installed. If installed, it is a symbolic link. Either way, it contains information that initialize the resolver routines. If the DNS is found at IP="192.168.11.1", it contains the following.

```
nameserver 192.168.11.1
```

The resolvconf package makes this "/etc/resolv.conf" into a symbolic link and manages its contents by the hook scripts automatically.

For the PC workstation on the typical adhoc LAN environment, the hostname can be resolved via Multicast DNS (mDNS, Zeroconf) in addition to the basic files and dns methods.

• Avahi provides a framework for Multicast DNS Service Discovery on Debian.

• It is equivalent of Apple Bonjour / Apple Rendezvous.

• The libnss-mdns plugin package provides host name resolution via mDNS for the GNU Name Service Switch (NSS) functionality of the GNU C Library (glibc).

• The "/etc/nsswitch.conf" file should have stanza like "hosts: files mdns4_minimal [NOTFOUND=return] dns mdns4".

• Host names ending with the ".local" pseudo-top-level domain (TLD) are resolved.

• The mDNS IPv4 link-local multicast address ”224.0.0.251” or its IPv6 equivalent ”FF02::FB” are used to make DNS query for a name ending with ”.local”.

The hostname resolution via deprecated NETBios over TCP/IP used by the older Windows system can be provided by installing the winbind package. The ”/etc/nsswitch.conf” file should have stanza like ”hosts: files mdns4_minimal [NOTFOUND=return] dns mdns4 wins” to enable this functionality. (Modern Windows system usually use the dns method for the hostname resolution.)

Note
The expansion of generic Top-Level Domains (gTLD) in the Domain Name System is underway. Watch out for the name collision when chosing a domain name used only within LAN.

5.1.2   The network interface name

The network interface name, e.g. eth0, is assigned to each hardware in the Linux kernel through the user space configuration mechanism, udev (see Section 3.3), as it is found. The network interface name is referred as **physical interface** in ifup(8) and interfaces(5).

In order to ensure each network interface to be named persistently for each reboot using MAC address etc., there is a rules file ”/etc/udev/rules.d/70-persistent-net.rules”. This file is automatically generated by the ”/lib/udev/write_net program, probably run by the ”persistent-net-generator.rules” rules file. You can modify it to change naming rule.

Caution
When editing the ”/etc/udev/rules.d/70-persistent-net.rules” rules file, you must keep each rule on a single line and the MAC address in lowercase. For example, if you find ”FireWire device” and ”PCI device” in this file, you probably want to name ”PCI device” as eth0 and configure it as the primary network interface.

5.1.3   The network address range for the LAN

Let us be reminded of the IPv4 32 bit address ranges in each class reserved for use on the local area networks (LANs) by rfc1918. These addresses are guaranteed not to conflict with any addresses on the Internet proper.

Class	network addresses	net mask	net mask /bits	# of subnets
A	10.x.x.x	255.0.0.0	/8	1
B	172.16.x.x —172.31.x.x	255.255.0.0	/16	16
C	192.168.0.x —192.168.255.x	255.255.255.0	/24	256

Table 5.2: List of network address ranges

Note
If one of these addresses is assigned to a host, then that host must not access the Internet directly but must access it through a gateway that acts as a proxy for individual services or else does Network Address Translation (NAT). The broadband router usually performs NAT for the consumer LAN environment.

5.1.4 The network device support

5.2 The modern network configuration for desktop

Network interfaces are typically initialized in "`networking.service`" for the `lo` interface and "`NetworkManager.service`" for other interfaces on modern Debian desktop system under `systemd`.

Debian `squeeze` and newer can manage the network connection via management daemon software such as NetworkManager (NM) (network-manager and associated packages) or Wicd (wicd and associated packages).

- They come with their own GUI and command-line programs as their user interfaces.

- They come with their own daemon as their backend system.

- They allow easy connection of your system to the Internet.

- They allow easy management of wired and wireless network configuration.

- They allow us to configure network independent of the legacy `ifupdown` package.

Note
Do not use these automatic network configuration tools for servers. These are aimed primarily for mobile desktop users on laptops.

These modern network configuration tools need to be configured properly to avoid conflicting with the legacy `ifupdown` package and its configuration file "`/etc/network/interfaces`".

Note
Some features of these automatic network configuration tools may suffer regressions. These are not as robust as the legacy `ifupdown` package. Check BTS of network-manager and BTS of wicd for current issues and limitations.

5.2.1 GUI network configuration tools

Official documentations for NM and Wicd on Debian are provided in "`/usr/share/doc/network-manager/README.Debian`" and "`/usr/share/doc/wicd/README.Debian`", respectively.

Essentially, the network configuration for desktop is done as follows.

1. Make desktop user, e.g. `foo`, belong to group "`netdev`" by the following (Alternatively, do it automatically via D-bus under modern desktop environments such as GNOME and KDE).

   ```
   $ sudo adduser foo netdev
   ```

2. Keep configuration of "`/etc/network/interfaces`" as simple as in the following.

   ```
   auto lo
   iface lo inet loopback
   ```

3. Restart NM or Wicd by the following.

   ```
   $ sudo /etc/init.d/network-manager restart
   ```

   ```
   $ sudo /etc/init.d/wicd restart
   ```

4. Configure your network via GUI.

Note
Only interfaces which are **not** listed in "/etc/network/interfaces" are managed by NM or Wicd to avoid conflict with ifupdown.

Tip
If you wish to extend network configuration capabilities of NM, please seek appropriate plug-in modules and supplemental packages such as network-manager-openconnect, network-manager-openvpn-gnome, network-manager-pptp-gnome, mobile-broadband-provider-info, gnome-bluetooth, etc. The same goes for those of Wicd.

⚠ **Caution**
These automatic network configuration tools may not be compatible with esoteric configurations of legacy ifupdown in "/etc/network/interfaces" such as ones in Section 5.6 and Section 5.7. Check BTS of network-manager and BTS of wicd for current issues and limitations.

5.3 The modern network configuration without GUI

Under systemd, the network may be configured in /etc/systemd/network/ instead. See systemd-resolved(8), resolved.conf(5), and systemd-networkd(8).

This allows the modern network configuration without GUI.

A DHCP client configuration can be set up by creating "/etc/systemd/network/dhcp.network". E.g.:

```
[Match]
Name=en*

[Network]
DHCP=yes
```

A static network configuration can be set up by creating "/etc/systemd/network/static.network". E.g.:

```
[Match]
Name=en*

[Network]
Address=192.168.0.15/24
Gateway=192.168.0.1
```

5.4 The legacy network connection and configuration

When the method described in Section 5.2 does not suffice your needs, you should use the legacy network connection and configuration method which combines many simpler tools.

The legacy network connection is specific for each method (see Section 5.5).

There are 2 types of programs for the low level network configuration on Linux (see Section 5.8.1).

• Old net-tools programs (ifconfig(8), ⋯) are from the Linux NET-3 networking system. Most of these are obsolete now.

- New [Linux iproute2](#) programs (ip(8), ⋯) are the current Linux networking system.

Although these low level networking programs are powerful, they are cumbersome to use. So high level network configuration systems have been created.

The `ifupdown` package is the de facto standard for such high level network configuration system on Debian. It enables you to bring up network simply by doing , e.g., "`ifup eth0`". Its configuration file is the "`/etc/network/interfaces`" file and its typical contents are the following.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp
```

The `resolvconf` package was created to supplement `ifupdown` system to support smooth reconfiguration of network address resolution by automating rewrite of resolver configuration file "`/etc/resolv.conf`". Now, most Debian network configuration packages are modified to use `resolvconf` package (see "`/usr/share/doc/resolvconf/README.Debian`").

Helper scripts to the `ifupdown` package such as `ifplugd`, `guessnet`, `ifscheme`, etc. are created to automate dynamic configuration of network environment such as one for mobile PC on wired LAN. These are relatively difficult to use but play well with existing `ifupdown` system.

These are explained in detail with examples (see Section 5.6 and Section 5.7).

5.5   The network connection method (legacy)

⚠ **Caution**
The connection test methods described in this section are meant for testing purposes. It is not meant to be used directly for the daily network connection. You are advised to use NM, Wicd, or the `ifupdown` package instead (see Section 5.2 and Section 5.6).

The typical network connection method and connection path for a PC can be summarized as the following.

PC	connection method	connection path
Serial port (`ppp0`)	PPP	[modem](#) POTS dial-up access point ISP
Ethernet port (`eth0`)	PPPoE/DHCP/Static	BB-modem BB service BB access point ISP
Ethernet port (`eth0`)	DHCP/Static	LAN BB-router with [network address translation (NAT)](#) (BB-modem ⋯)

Table 5.3: List of network connection methods and connection paths

Here is the summary of configuration scripts for each connection method.

The network connection acronyms mean the following.

Note
The WAN connection services via cable TV are generally served by DHCP or PPPoE. The ones by ADSL and FTTP are generally served by PPPoE. You have to consult your ISP for exact configuration requirements of the WAN connection.

---

connection method	configuration	backend package(s)
PPP	`pppconfig` to create deterministic chat	`pppconfig`, `ppp`
PPP (alternative)	`wvdialconf` to create heuristic chat	`ppp`, `wvdial`
PPPoE	`pppoeconf` to create deterministic chat	`pppoeconf`, `ppp`
DHCP	described in "/etc/dhcp/dhclient.conf"	`isc-dhcp-client`
static IP (IPv4)	described in "/etc/network/interfaces"	`iproute` or `net-tools` (obsolete)
static IP (IPv6)	described in "/etc/network/interfaces"	`iproute`

Table 5.4: List of network connection configurations

acronym	meaning
POTS	plain old telephone service
BB	broadband
BB-service	e.g., the digital subscriber line (DSL), the cable TV, or the fiber to the premises (FTTP)
BB-modem	e.g., the DSL modem, the cable modem, or the optical network terminal (ONT)
LAN	local area network
WAN	wide area network
DHCP	dynamic host configuration protocol
PPP	point-to-point protocol
PPPoE	point-to-point protocol over Ethernet
ISP	Internet service provider

Table 5.5: List of network connection acronyms

> **Note**
> When BB-router is used to create home LAN environment, PCs on LAN are connected to the WAN via BB-router with network address translation (NAT). For such case, PC's network interfaces on the LAN are served by static IP or DHCP from the BB-router. BB-router must be configured to connect the WAN following the instruction by your ISP.

5.5.1 The DHCP connection with the Ethernet

The typical modern home and small business network, i.e. LAN, are connected to the WAN (Internet) using some consumer grade broadband router. The LAN behind this router is usually served by the dynamic host configuration protocol (DHCP) server running on the router.

Just install the `isc-dhcp-client` package for the Ethernet served by the dynamic host configuration protocol (DHCP).

See dhclient.conf(5).

5.5.2 The static IP connection with the Ethernet

No special action is needed for the Ethernet served by the static IP.

5.5.3 The PPP connection with pppconfig

The configuration script `pppconfig` configures the PPP connection interactively just by selecting the following.

- The telephone number
- The ISP user name
- The ISP password
- The port speed
- The modem communication port
- The authentication method

file	function
`/etc/ppp/peers/<isp_name>`	The `pppconfig` generated configuration file for `pppd` specific to <isp_name>
`/etc/chatscripts/<isp_name>`	The `pppconfig` generated configuration file for `chat` specific to <isp_name>
`/etc/ppp/options`	The general execution parameter for `pppd`
`/etc/ppp/pap-secret`	Authentication data for the PAP (security risk)
`/etc/ppp/chap-secret`	Authentication data for the CHAP (more secure)

Table 5.6: List of configuration files for the PPP connection with pppconfig

> ⚠ **Caution**
> The "<isp_name>" value of "`provider`" is assumed if `pon` and `poff` commands are invoked without arguments.

You can test configuration using low level network configuration tools as the following.

```
$ sudo pon <isp_name>
...
$ sudo poff <isp_name>
```

See "/usr/share/doc/ppp/README.Debian.gz".

5.5.4   The alternative PPP connection with wvdialconf

A different approach to using pppd(8) is to run it from wvdial(1) which comes in the `wvdial` package. Instead of `pppd` running chat(8) to dial in and negotiate the connection, `wvdial` does the dialing and initial negotiating and then starts `pppd` to do the rest.

The configuration script `wvdialconf` configures the PPP connection interactively just by selecting the following.

- The telephone number

- The ISP user name

- The ISP password

`wvdial` succeeds in making the connection in most cases and maintains authentication data list automatically.

file	function
/etc/ppp/peers/wvdial	The `wvdialconf` generated configuration file for `pppd` specific to `wvdial`
/etc/wvdial.conf	The `wvdialconf` generated configuration file
/etc/ppp/options	The general execution parameter for `pppd`
/etc/ppp/pap-secret	Authentication data for the PAP (security risk)
/etc/ppp/chap-secret	Authentication data for the CHAP (more secure)

Table 5.7: List of configuration files for the PPP connection with wvdialconf

You can test configuration using low level network configuration tools as the following.

```
$ sudo wvdial
...
$ sudo killall wvdial
```

See wvdial(1) and wvdial.conf(5).

5.5.5   The PPPoE connection with pppoeconf

When your ISP serves you with PPPoE connection and you decide to connect your PC directly to the WAN, the network of your PC must be configured with the PPPoE. The PPPoE stand for PPP over Ethernet. The configuration script `pppoeconf` configures the PPPoE connection interactively.

The configuration files are the following.

You can test configuration using low level network configuration tools as the following.

```
$ sudo /sbin/ifconfig eth0 up
$ sudo pon dsl-provider
...
$ sudo poff dsl-provider
$ sudo /sbin/ifconfig eth0 down
```

See "/usr/share/doc/pppoeconf/README.Debian".

file	function
/etc/ppp/peers/dsl-provider	The pppoeconf generated configuration file for pppd specific to pppoe
/etc/ppp/options	The general execution parameter for pppd
/etc/ppp/pap-secret	Authentication data for the PAP (security risk)
/etc/ppp/chap-secret	Authentication data for the CHAP (more secure)

Table 5.8: List of configuration files for the PPPoE connection with pppoeconf

5.6 The basic network configuration with ifupdown (legacy)

The traditional TCP/IP network setup on the Debian system uses ifupdown package as a high level tool. There are 2 typical cases.

- For **dynamic IP** system such as mobile PCs, you should setup TCP/IP network **with** the resolvconf package and enable you to switch your network configuration easily (see Section 5.6.4).

- For **static IP** system such as servers, you should setup TCP/IP network **without** the resolvconf package and keep your system simple (see Section 5.6.5).

These traditional setup methods are quite useful if you wish to set up advanced configuration; find details in the following.

The ifupdown package provides the standardized framework for the high level network configuration in the Debian system. In this section, we learn the basic network configuration with ifupdown with simplified introduction and many typical examples.

5.6.1 The command syntax simplified

The ifupdown package contains 2 commands: ifup(8) and ifdown(8). They offer high level network configuration dictated by the configuration file "/etc/network/interfaces".

command	action
ifup eth0	bring up a network interface eth0 with the configuration eth0 if "iface eth0" stanza exists
ifdown eth0	bring down a network interface eth0 with the configuration eth0 if "iface eth0" stanza exists

Table 5.9: List of basic network configuration commands with ifupdown

---

⚠ **Warning**
Do not use low level configuration tools such as ifconfig(8) and ip(8) commands to configure an interface in **up** state.

Note
There is no command ifupdown.

5.6.2 The basic syntax of "/etc/network/interfaces"

The key syntax of "/etc/network/interfaces" as explained in interfaces(5) can be summarized as the following.

Lines started with **iface** stanza has the following syntax.

stanza	meaning
"auto <interface_name>"	start interface <interface_name> upon start of the system
"allow-auto <interface_name>"	, ,
"allow-hotplug <interface_name>"	start interface <interface_name> when the kernel detects a hotplug event from the interface
Lines started with "iface <config_name> …"	define the network configuration <config_name>
Lines started with "mapping <interface_name_glob>"	define mapping value of <config_name> for the matching <interface_name>
A line starting with a hash "#"	ignore as comments (end-of-line comments are **not** supported)
A line ending with a backslash "\"	extend the configuration to the next line

Table 5.10: List of stanzas in "/etc/network/interfaces"

```
iface <config_name> <address_family> <method_name>
 <option1> <value1>
 <option2> <value2>
 ...
```

For the basic configuration, the **mapping** stanza is not used and you use the network interface name as the network configuration name (See Section 5.7.5).

> ⚠ **Warning**
> Do not define duplicates of the "iface" stanza for a network interface in "/etc/network/interfaces".

5.6.3   The loopback network interface

The following configuration entry in the "/etc/network/interfaces" file brings up the loopback network interface lo upon booting the system (via **auto** stanza).

```
auto lo
iface lo inet loopback
```

This one always exists in the "/etc/network/interfaces" file.

5.6.4   The network interface served by the DHCP

After preparing the system by Section 5.5.1, the network interface served by the DHCP is configured by creating the configuration entry in the "/etc/network/interfaces" file as the following.

```
allow-hotplug eth0
iface eth0 inet dhcp
```

When the Linux kernel detects the physical interface eth0, the **allow-hotplug** stanza causes ifup to bring up the interface and the **iface** stanza causes ifup to use DHCP to configure the interface.

5.6.5   The network interface with the static IP

The network interface served by the static IP is configured by creating the configuration entry in the "/etc/network/interfaces" file as the following.

```
allow-hotplug eth0
iface eth0 inet static
 address 192.168.11.100
 netmask 255.255.255.0
 gateway 192.168.11.1
 dns-domain example.com
 dns-nameservers 192.168.11.1
```

When the Linux kernel detects the physical interface `eth0`, the **allow-hotplug** stanza causes `ifup` to bring up the interface and the **iface** stanza causes `ifup` to use the static IP to configure the interface.

Here, I assumed the following.

- IP address range of the LAN network: `192.168.11.0` - `192.168.11.255`

- IP address of the gateway: `192.168.11.1`

- IP address of the PC: `192.168.11.100`

- The `resolvconf` package: installed

- The domain name: "`example.com`"

- IP address of the DNS server: `192.168.11.1`

When the `resolvconf` package is not installed, DNS related configuration needs to be done manually by editing the "`/etc/resolv.c` as the following.

```
nameserver 192.168.11.1
domain example.com
```

> ⚠ **Caution**
> The IP addresses used in the above example are not meant to be copied literally. You have to adjust IP numbers to your actual network configuration.

5.6.6   The basics of wireless LAN interface

The wireless LAN (WLAN for short) provides the fast wireless connectivity through the spread-spectrum communication of unlicensed radio bands based on the set of standards called IEEE 802.11.

The WLAN interfaces are almost like normal Ethernet interfaces but require some network ID and encryption key data to be provided when they are initialized. Their high level network tools are exactly the same as that of Ethernet interfaces except interface names are a bit different like `eth1`, `wlan0`, `ath0`, `wifi0`, ⋯depending on the kernel drivers used.

> **Tip**
> The `wmaster0` device is the master device which is an internal device used only by SoftMAC with new mac80211 API of Linux.

Here are some keywords to remember for the WLAN.

The actual choice of protocol is usually limited by the wireless router you deploy.

acronym	full word	meaning
NWID	Network ID	16 bit network ID used by pre-802.11 WaveLAN network (very deprecated)
(E)SSID	(Extended) Service Set Identifier	network name of the Wireless Access Points (APs) interconnected to form an integrated 802.11 wireless LAN, Domain ID
WEP, (WEP2)	Wired Equivalent Privacy	1st generation 64-bit (128-bit) wireless encryption standard with 40-bit key (deprecated)
WPA	Wi-Fi Protected Access	2nd generation wireless encryption standard (most of 802.11i), compatible with WEP
WPA2	Wi-Fi Protected Access 2	3rd generation wireless encryption standard (full 802.11i), non-compatible with WEP

Table 5.11: List of acronyms for WLAN

5.6.7 The wireless LAN interface with WPA/WPA2

You need to install the `wpasupplicant` package to support the WLAN with the new WPA/WPA2.

In case of the DHCP served IP on WLAN connection, the "`/etc/network/interfaces`" file entry should be as the following.

```
allow-hotplug ath0
iface ath0 inet dhcp
 wpa-ssid homezone
 # hexadecimal psk is encoded from a plaintext passphrase
 wpa-psk 000102030405060708090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f
```

See "`/usr/share/doc/wpasupplicant/README.modes.gz`".

5.6.8 The wireless LAN interface with WEP

You need to install the `wireless-tools` package to support the WLAN with the old WEP. (Your consumer grade router may still be using this insecure infrastructure but this is better than nothing.)

> **⚠ Caution**
> Please note that your network traffic on WLAN with WEP may be sniffed by others.

In case of the DHCP served IP on WLAN connection, the "`/etc/network/interfaces`" file entry should be as the following.

```
allow-hotplug eth0
iface eth0 inet dhcp
 wireless-essid Home
 wireless-key1 0123-4567-89ab-cdef
 wireless-key2 12345678
 wireless-key3 s:password
 wireless-defaultkey 2
 wireless-keymode open
```

See "`/usr/share/doc/wireless-tools/README.Debian`".

5.6.9 The PPP connection

You need to configure the PPP connection first as described before (see Section 5.5.3). Then, add the "/etc/network/interfaces" file entry for the primary PPP device ppp0 as the following.

```
iface ppp0 inet ppp
 provider <isp_name>
```

5.6.10 The alternative PPP connection

You need to configure the alternative PPP connection with wvdial first as described before (see Section 5.5.4). Then, add the "/etc/network/interfaces" file entry for the primary PPP device ppp0 as the following.

```
iface ppp0 inet wvdial
```

5.6.11 The PPPoE connection

For PC connected directly to the WAN served by the PPPoE, you need to configure system with the PPPoE connection as described before (see Section 5.5.5). Then, add the "/etc/network/interfaces" file entry for the primary PPPoE device eth0 as the following.

```
allow-hotplug eth0
iface eth0 inet manual
 pre-up /sbin/ifconfig eth0 up
 up ifup ppp0=dsl
 down ifdown ppp0=dsl
 post-down /sbin/ifconfig eth0 down
# The following is used internally only
iface dsl inet ppp
 provider dsl-provider
```

5.6.12 The network configuration state of ifupdown

The "/etc/network/run/ifstate" file stores the **intended** network configuration states for all the currently active network interfaces managed by the ifupdown package. Unfortunately, even if the ifupdown system fails to bring up the interface as intended, the "/etc/network/run/ifstate" file lists it active.

Unless the output of the ifconfig(8) command for an interface does not have a line like following example, it can not be used as a part of IPV4 network.

```
  inet addr:192.168.11.2  Bcast:192.168.11.255  Mask:255.255.255.0
```

Note
For the Ethernet device connected to the PPPoE, the output of the ifconfig(8) command lacks a line which looks like above example.

5.6.13 The basic network reconfiguration

When you try to reconfigure the interface, e.g. eth0, you must disable it first with the "**sudo ifdown eth0**" command. This removes the entry of eth0 from the "/etc/network/run/ifstate" file. (This may result in some error message if eth0 is not active or it is configured improperly previously. So far, it seems to be safe to do this for the simple single user work station at any time.)

You are now free to rewrite the "`/etc/network/interfaces`" contents as needed to reconfigure the network interface, `eth0`.

Then, you can reactivate `eth0` with the "**`sudo ifup eth0`**" command.

Tip
You can (re)initialize the network interface simply by "**`sudo ifdown eth0;sudo ifup eth0`**".

5.6.14   The ifupdown-extra package

The `ifupdown-extra` package provides easy network connection tests for use with the `ifupdown` package.

- The network-test(1) command can be used from the shell.

- The automatic scripts are run for each `ifup` command execution.

The `network-test` command frees you from the execution of cumbersome low level commands to analyze the network problem.

The automatic scripts are installed in "`/etc/network/*/`" and perform the following.

- Check the network cable connection

- Check duplicate use of IP address

- Setup system's static routes based on the "`/etc/network/routes`" definition

- Check if network gateway is reachable

- Record results in the "`/var/log/syslog`" file

This syslog record is quite useful for administration of the network problem on the remote system.

Tip
The   automatic   behavior   of   the   `ifupdown-extra`   package   is   configurable   with   the
"`/etc/default/network-test`". Some of these automatic checks slow down the system boot-up a little
bit since it takes some time to listen for ARP replies.

5.7   The advanced network configuration with ifupdown (legacy)

The functionality of the `ifupdown` package can be improved beyond what was described in Section 5.6 with the advanced knowledge.

The functionalities described here are completely optional. I, being lazy and minimalist, rarely bother to use these.

⚠ **Caution**
If you could not set up network connection by information in Section 5.6, you make situation worse by using
information below.

5.7.1 The ifplugd package

The `ifplugd` package is an older automatic network configuration tool which can manage only Ethernet connections. This solves unplugged/replugged Ethernet cable issues for mobile PC etc. If you have NetworkManager or Wicd (see Section 5.2) installed, you do not need this package.

This package runs a daemon and replaces **auto** or **allow-hotplug** functionalities (see Table 5.10) and starts interfaces upon their connection to the network.

Here is how to use the `ifplugd` package for the internal Ethernet port, e.g. `eth0`.

1. Remove stanza in "`/etc/network/interfaces`": "`auto eth0`" or "`allow-hotplug eth0`".

2. Keep stanza in "`/etc/network/interfaces`": "`iface eth0 inet ⋯`" and "`mapping ⋯`".

3. Install the `ifplugd` package.

4. Run "`sudo dpkg-reconfigure ifplugd`".

5. Put `eth0` as the "static interfaces to be watched by ifplugd".

Now, the network reconfiguration works as you desire.

- Upon power-on or upon hardware discovery, the interface is not brought up by itself.

  - Quick boot process without the long DHCP timeout.
  - No funny activated interface without proper IPv4 address (see Section 5.6.12).

- Upon finding the Ethernet cable, the interface is brought up.

- Upon some time after unplugging the Ethernet cable, the interface is brought down automatically.

- Upon plugging in another Ethernet cable, the interface is brought up under the new network environment.

Tip
The arguments for the ifplugd(8) command can set its behaviors such as the delay for reconfiguring interfaces.

5.7.2 The ifmetric package

The `ifmetric` package enables us to manipulate metrics of routes a posteriori even for DHCP.

The following sets the `eth0` interface to be preferred over the `wlan0` interface.

1. Install the `ifmetric` package.

2. Add an option line with "`metric 0`" just below the "`iface eth0 inet dhcp`" line in "`/etc/network/interfaces`".

3. Add an option line with "`metric 1`" just below the "`iface wlan0 inet dhcp`" line in "`/etc/network/interfaces`".

The metric 0 means the highest priority route and is the default one. The larger metric value means lower priority routes. The IP address of the active interface with the lowest metric value becomes the originating one. See ifmetric(8).

5.7.3   The virtual interface

A single physical Ethernet interface can be configured as multiple virtual interfaces with different IP addresses. Usually the purpose is to connect an interface to several IP subnetworks. For example, IP address based virtual web hosting by a single network interface is one such application.

For example, let's suppose the following.

- A single Ethernet interface on your host is connected to a Ethernet hub (not to the broadband router).

- The Ethernet hub is connected to both the Internet and LAN network.

- The LAN network uses subnet `192.168.0.x/24`.

- Your host uses DHCP served IP address with the physical interface `eth0` for the Internet.

- Your host uses `192.168.0.1` with the virtual interface `eth0:0` for the LAN.

The following stanzas in "`/etc/network/interfaces`" configure your network.

```
iface eth0 inet dhcp
 metric 0
iface eth0:0 inet static
 address 192.168.0.1
 netmask 255.255.255.0
 network 192.168.0.0
 metric 1
```

> **⚠ Caution**
> Although this configuration example with network address translation (NAT) using netfilter/iptables (see Section 5.10) can provide cheap router for the LAN with only single interface, there is no real firewall capability with such set up. You should use 2 physical interfaces with NAT to secure the local network from the Internet.

5.7.4   The advanced command syntax

The `ifupdown` package offers advanced network configuration using the **network configuration** name and the **network interface** name. I use a terminology being slightly different from the one used in ifup(8) and interfaces(5).

manpage terminology	my terminology	examples in the following text	description
physical interface name	**network interface** name	`lo`, `eth0`, `<interface_name>`	name given by the Linux kernel (using `udev` mechanism)
logical interface name	**network configuration** name	`config1`, `config2`, `<config_name>`	name token following **`iface`** in the "`/etc/network/interfaces`"

Table 5.12: List of terminology for network devices

Basic network configuration commands in Section 5.6.1 require the **network configuration** name token of the **`iface`** stanza to match the **network interface** name in the "`/etc/network/interfaces`".

Advanced network configuration commands enables separation of the **network configuration** name and the **network interface** name in the "`/etc/network/interfaces`" as the following.

command	action
ifup eth0=config1	bring up a network interface eth0 with the configuration config1
ifdown eth0=config1	bring down a network interface eth0 with the configuration config1
ifup eth0	bring up a network interface eth0 with the configuration selected by **mapping** stanza
ifdown eth0	bring down a network interface eth0 with the configuration selected by **mapping** stanza

Table 5.13: List of advanced network configuration commands with ifupdown

5.7.5 The mapping stanza

We skipped explaining the **mapping** stanza in the "/etc/network/interfaces" in Section 5.6.2 to avoid complication. This stanza has the following syntax.

```
mapping <interface_name_glob>
 script <script_name>
 map <script_input1>
 map <script_input2>
 map ...
```

This provides advanced features to the "/etc/network/interfaces" file by automating the choice of the configuration with the mapping script specified by <script_name>.

Let's follow the execution of the following.

```
$ sudo ifup eth0
```

When the "<interface_name_glob>" matches "eth0", this execution produces the execution of the following command to configure eth0 automatically.

```
$ sudo ifup eth0=$(echo -e '<script_input1> \n <script_input2> \n ...' | <script_name> eth0 ←
    )
```

Here, script input lines with "map" are optional and can be repeated.

Note
The glob for **mapping** stanza works like shell filename glob (see Section 1.5.6).

5.7.6 The manually switchable network configuration

Here is how to switch manually among several network configurations without rewriting the "/etc/network/interfaces" file as in Section 5.6.13 .

For all the network configuration you need to access, you create a separate stanza in "/etc/network/interfaces" file as the following.

```
auto lo
iface lo inet loopback

iface config1 inet dhcp

iface config2 inet static
 address 192.168.11.100
 netmask 255.255.255.0
 gateway 192.168.11.1
 dns-domain example.com
```

```
 dns-nameservers 192.168.11.1

iface pppoe inet manual
 pre-up /sbin/ifconfig eth0 up
 up ifup ppp0=dsl
 down ifdown ppp0=dsl
 post-down /sbin/ifconfig eth0 down

# The following is used internally only
iface dsl inet ppp
 provider dsl-provider

iface pots inet ppp
 provider provider
```

Please note the **network configuration name** which is the token after **iface** does not use the token for the **network interface name**. Also, there are no **auto** stanza nor **allow-hotplug** stanza to start the network interface eth0 automatically upon events.

Now you are ready to switch the network configuration.

Let's move your PC to a LAN served by the DHCP. You bring up the **network interface** (the physical interface) eth0 by assigning the **network configuration** name (the logical interface name) config1 to it by the following.

```
$ sudo ifup eth0=config1
Password:
...
```

The interface eth0 is up, configured by DHCP and connected to LAN.

```
$ sudo ifdown eth0=config1
...
```

The interface eth0 is down and disconnected from LAN.

Let's move your PC to a LAN served by the static IP. You bring up the **network interface** eth0 by assigning the **network configuration** name config2 to it by the following.

```
$ sudo ifup eth0=config2
...
```

The interface eth0 is up, configured with static IP and connected to LAN. The additional parameters given as dns-* configures "/etc/resolv.conf" contents. This "/etc/resolv.conf" is better manged if the resolvconf package is installed.

```
$ sudo ifdown eth0=config2
...
```

The interface eth0 is down and disconnected from LAN, again.

Let's move your PC to a port on BB-modem connected to the PPPoE served service. You bring up the **network interface** eth0 by assigning the **network configuration** name pppoe to it by the following.

```
$ sudo ifup eth0=pppoe
...
```

The interface eth0 is up, configured with PPPoE connection directly to the ISP.

```
$ sudo ifdown eth0=pppoe
...
```

The interface eth0 is down and disconnected, again.

Let's move your PC to a location without LAN or BB-modem but with POTS and modem. You bring up the **network interface** ppp0 by assigning the **network configuration** name pots to it by the following.

```
$ sudo ifup ppp0=pots
...
```

The interface `ppp0` is up and connected to the Internet with PPP.

```
$ sudo ifdown ppp0=pots
...
```

The interface `ppp0` is down and disconnected from the Internet.

You should check the "`/etc/network/run/ifstate`" file for the current network configuration state of the `ifupdown` system.

> ⚠ **Warning**
> You may need to adjust numbers at the end of `eth*`, `ppp*`, etc. if you have multiple network interfaces.

5.7.7 Scripting with the ifupdown system

The `ifupdown` system automatically runs scripts installed in "`/etc/network/*/`" while exporting environment variables to scripts.

environment variable	value passed
"$IFACE"	physical name (interface name) of the interface being processed
"$LOGICAL"	logical name (configuration name) of the interface being processed
"$ADDRFAM"	<address_family> of the interface
"$METHOD"	<method_name> of the interface (e.g., "static")
"$MODE"	"start" if run from `ifup`, "stop" if run from `ifdown`
"$PHASE"	as per "$MODE", but with finer granularity, distinguishing the `pre-up`, `post-up`, `pre-down` and `post-down` phases
"$VERBOSITY"	indicates whether "`--verbose`" was used; set to 1 if so, 0 if not
"$PATH"	command search path: "`/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbi`
"$IF_<OPTION>"	value for the corresponding option under the **iface** stanza

Table 5.14: List of environment variables passed by the ifupdown system

Here, each environment variable, "`$IF_<OPTION>`", is created from the name for the corresponding option such as <option1> and <option2> by prepending "`$IF_`", converting the case to the upper case, replacing hyphens to underscores, and discarding non-alphanumeric characters.

> **Tip**
> See Section 5.6.2 for <address_family>, <method_name>, <option1> and <option2>.

The `ifupdown-extra` package (see Section 5.6.14) uses these environment variables to extend the functionality of the `ifupdown` package. The `ifmetric` package (see Section 5.7.2) installs the "`/etc/network/if-up.d/ifmetric`" script which sets the metric via the "`$IF_METRIC`" variable. The `guessnet` package (see Section 5.7.8), which provides simple and powerful framework for the auto-selection of the network configuration via the mapping mechanism, also uses these.

Note
For more specific examples of custom network configuration scripts using these environment variables, you should check example scripts in "/usr/share/doc/ifupdown/examples/*" and scripts used in `ifscheme` and `ifupdown-scripts-zg2` packages. These additional scripts have some overlaps of functionalities with basic `ifupdown-extra` and `guessnet` packages. If you install these additional scripts, you should customize these scripts to avoid interferences.

5.7.8 Mapping with guessnet

Instead of manually choosing configuration as described in Section 5.7.6, you can use the mapping mechanism described in Section 5.7.5 to select network configuration automatically with custom scripts.

The guessnet-ifupdown(8) command provided by the `guessnet` package is designed to be used as a mapping script and provides powerful framework to enhance the `ifupdown` system.

- You list test condition as the value for **guessnet** options for each network configuration under **iface** stanza.

- Mapping choses the **iface** with first non-ERROR result as the network configuration.

This dual usage of the "/etc/network/interfaces" file by the mapping script, `guessnet-ifupdown`, and the original network configuration infrastructure, `ifupdown`, does not cause negative impacts since **guessnet** options only export extra environment variables to scripts run by the `ifupdown` system. See details in guessnet-ifupdown(8).

Note
When multiple **guessnet** option lines are required in "/etc/network/interfaces", use option lines started with **guessnet1**, **guessnet2**, and so on, since the `ifupdown` package does not allow starting strings of option lines to be repeated.

5.8 The low level network configuration

5.8.1 Iproute2 commands

Iproute2 commands offer complete low-level network configuration capabilities. Here is a translation table from obsolete net-tools commands to new iproute2 etc. commands.

obsolete net-tools	new iproute2 etc.	manipulation
ifconfig(8)	`ip addr`	protocol (IP or IPv6) address on a device
route(8)	`ip route`	routing table entry
arp(8)	`ip neigh`	ARP or NDISC cache entry
ipmaddr	`ip maddr`	multicast address
iptunnel	`ip tunnel`	tunnel over IP
nameif(8)	ifrename(8)	name network interfaces based on MAC addresses
mii-tool(8)	ethtool(8)	Ethernet device settings

Table 5.15: Translation table from obsolete `net-tools` commands to new `iproute2` commands

See ip(8) and IPROUTE2 Utility Suite Howto.

command	description
`ifconfig`	display the link and address status of active interfaces
`ip addr show`	display the link and address status of active interfaces
`route -n`	display all the routing table in numerical addresses
`ip route show`	display all the routing table in numerical addresses
`arp`	display the current content of the ARP cache tables
`ip neigh`	display the current content of the ARP cache tables
`plog`	display ppp daemon log
`ping yahoo.com`	check the Internet connection to "`yahoo.com`"
`whois yahoo.com`	check who registered "`yahoo.com`" in the domains database
`traceroute yahoo.com`	trace the Internet connection to "`yahoo.com`"
`tracepath yahoo.com`	trace the Internet connection to "`yahoo.com`"
`mtr yahoo.com`	trace the Internet connection to "`yahoo.com`" (repeatedly)
`dig [@dns-server.com] example.com [{a|mx|any}]`	check DNS records of "`example.com`" by "`dns-server.com`" for a "`a`", "`mx`", or "`any`" record
`iptables -L -n`	check packet filter
`netstat -a`	find all open ports
`netstat -l --inet`	find listening ports
`netstat -ln --tcp`	find listening TCP ports (numeric)
`dlint example.com`	check DNS zone information of "`example.com`"

Table 5.16: List of low level network commands

5.8.2 Safe low level network operations

You may use low level network commands as follows safely since they do not change network configuration.

> **Tip**
> Some of these low level network configuration tools reside in "`/sbin/`". You may need to issue full command path such as "`/sbin/ifconfig`" or add "`/sbin`" to the "`$PATH`" list in your "`~/.bashrc`".

5.9 Network optimization

Generic network optimization is beyond the scope of this documentation. I touch only subjects pertinent to the consumer grade connection.

packages	popcon	size	description
iftop	V:8, I:110	93	display bandwidth usage information on an network interface
iperf	V:4, I:54	168	Internet Protocol bandwidth measuring tool
ifstat	V:1, I:10	56	InterFace STATistics Monitoring
bmon	V:1, I:13	140	portable bandwidth monitor and rate estimator
ethstatus	V:0, I:4	39	script that quickly measures network device throughput
bing	V:0, I:1	71	empirical stochastic bandwidth tester
bwm-ng	V:1, I:17	89	small and simple console-based bandwidth monitor
ethstats	V:0, I:1	20	console-based Ethernet statistics monitor
ipfm	V:0, I:0	78	bandwidth analysis tool

Table 5.17: List of network optimization tools

5.9.1   Finding optimal MTU

The Maximum Transmission Unit (MTU) value can be determined experimentally with ping(8) with "`-M do`" option which sends ICMP packets with data size starting from 1500 (with offset of 28 bytes for the IP+ICMP header) and finding the largest size without IP fragmentation.

For example, try the following

```
$ ping -c 1 -s $((1500-28)) -M do www.debian.org
PING www.debian.org (194.109.137.218) 1472(1500) bytes of data.
From 192.168.11.2 icmp_seq=1 Frag needed and DF set (mtu = 1454)

--- www.debian.org ping statistics ---
0 packets transmitted, 0 received, +1 errors
```

Try 1454 instead of 1500

You see ping(8) succeed with 1454.

This process is Path MTU (PMTU) discovery (RFC1191) and the tracepath(8) command can automate this.

Tip
The above example with PMTU value of 1454 is for my previous FTTP provider which used Asynchronous Transfer Mode (ATM) as its backbone network and served its clients with the PPPoE. The actual PMTU value depends on your environment, e.g., 1500 for the my new FTTP provider.

---

network environment	MTU	rationale
Dial-up link (IP: PPP)	576	standard
Ethernet link (IP: DHCP or fixed)	1500	standard and default
Ethernet link (IP: PPPoE)	1492 (=1500-8)	2 bytes for PPP header and 6 bytes for PPPoE header
Ethernet link (ISP's backbone: ATM, IP: DHCP or fixed)	1462 (=48*31-18-8)	author's speculation: 18 bytes for Ethernet header, 8 bytes for SAR trailer
Ethernet link (ISP's backbone: ATM, IP: PPPoE)	1454 (=48*31-8-18-8)	see "Optimal MTU configuration for PPPoE ADSL Connections" for rationale

Table 5.18: Basic guide lines of the optimal MTU value

In addtion to these basic guide lines, you should know the following.

- Any use of tunneling methods (VPN etc.) may reduce optimal MTU further by their overheads.

- The MTU value should not exceed the experimentally determined PMTU value.

- The bigger MTU value is generally better when other limitations are met.

5.9.2   Setting MTU

Here are examples for setting the MTU value from its default 1500 to 1454.

For the DHCP (see Section 5.6.4), you can replace pertinent **iface** stanza lines in the "`/etc/network/interfaces`" with the following.

```
iface eth0 inet dhcp
 pre-up /sbin/ifconfig $IFACE mtu 1454
```

For the static IP (see Section 5.6.5), you can replace pertinent **iface** stanza lines in the "`/etc/network/interfaces`" with the following.

```
iface eth0 inet static
 address 192.168.11.100
 netmask 255.255.255.0
 gateway 192.168.11.1
 mtu 1454
 dns-domain example.com
 dns-nameservers 192.168.11.1
```

For the direct PPPoE (see Section 5.5.5), you can replace pertinent "mtu" line in the "/etc/ppp/peers/dsl-provider" with the following.

```
mtu 1454
```

The maximum segment size (MSS) is used as an alternative measure of packet size. The relationship between MSS and MTU are the following.

- MSS = MTU - 40 for IPv4

- MSS = MTU - 60 for IPv6

Note
The iptables(8) (see Section 5.10) based optimization can clamp packet size by the MSS and is useful for the router. See "TCPMSS" in iptables(8).

5.9.3 WAN TCP optimization

The TCP throughput can be maximized by adjusting TCP buffer size parameters as described in "TCP Tuning Guide" and "TCP tuning" for the modern high-bandwidth and high-latency WAN. So far, the current Debian default settings serve well even for my LAN connected by the fast 1G bps FTTP service.

5.10 Netfilter infrastructure

Netfilter provides infrastructure for stateful firewall and network address translation (NAT) with Linux kernel modules (see Section 3.3.1).

packages	popcon	size	description
iptables	V:256, I:995	1543	administration tools for netfilter (iptables(8) for IPv4, ip6tables(8) for IPv6)
arptables	V:0, I:2	83	administration tools for netfilter (arptables(8) for ARP)
ebtables	V:36, I:59	351	administration tools for netfilter (ebtables(8) for Ethernet bridging)
iptstate	V:0, I:4	116	continuously monitor netfilter state (similar to top(1))
shorewall-init	V:0, I:0	79	Shoreline Firewall initialization
shorewall	V:6, I:14	2448	Shoreline Firewall, netfilter configuration file generator
shorewall-lite	V:0, I:0	76	Shoreline Firewall, netfilter configuration file generator (light version)
shorewall6	V:0, I:2	782	Shoreline Firewall, netfilter configuration file generator (IPv6 version)
shorewall6-lite	V:0, I:0	74	Shoreline Firewall, netfilter configuration file generator (IPv6, light version)

Table 5.19: List of firewall tools

Main user space program of netfilter is iptables(8). You can manually configure netfilter interactively from shell, save its state with iptables-save(8), and restore it via init script with iptables-restore(8) upon system reboot.

Configuration helper scripts such as shorewall ease this process.

See documentations at http://www.netfilter.org/documentation/ (or in "`/usr/share/doc/iptables/html/`").

- Linux Networking-concepts HOWTO

- Linux 2.4 Packet Filtering HOWTO

- Linux 2.4 NAT HOWTO

Tip
Although these were written for Linux **2.4**, both iptables(8) command and netfilter kernel function apply for Linux **2.6** and **3.x** kernel series.

Chapter 6

Network applications

After establishing network connectivity (see Chapter 5), you can run various network applications.

Tip
For modern Debian specific guide to the network infrastructure, read The Debian Administrator's Handbook — Network Infrastructure.

Tip
If you enabled "2-Step Verification" with some ISP, you need to obtain an application password to access POP and SMTP services from your program. You may need to approve your host IP in advance.

6.1 Web browsers

There are many web browser packages to access remote contents with Hypertext Transfer Protocol (HTTP).

package	popcon	size	type	description of web browser
chromium	V:74, I:152	137830	X	Chromium, (open-source browser from Google)
firefox	V:19, I:32	158540	, ,	Firefox, (open-source browser from Mozilla, only available in Debian Unstable)
firefox-esr	V:280, I:454	106910	, ,	Firefox ESR, (Firefox Extended Support Release)
epiphany-browser	V:7, I:33	1476	, ,	GNOME, HIG compliant, Epiphany
konqueror	V:24, I:112	20672	, ,	KDE, Konqueror
dillo	V:1, I:5	1500	, ,	Dillo, (light weight browser, FLTK based)
w3m	V:126, I:564	2302	text	w3m
lynx	V:23, I:104	1903	, ,	Lynx
elinks	V:12, I:31	1583	, ,	ELinks
links	V:15, I:44	2123	, ,	Links (text only)
links2	V:2, I:16	5366	graphics	Links (console graphics without X)

Table 6.1: List of web browsers

6.1.1 Browser configuration

You may be able to use following special URL strings for some browsers to confirm their settings.

- ”about:”

- ”about:config”

- ”about:plugins”

Debian offers many free browser plugin packages in the main archive area which can handle not only Java (software platform) and Flash but also MPEG, MPEG2, MPEG4, DivX, Windows Media Video (.wmv), QuickTime (.mov), MP3 (.mp3), Ogg/Vorbis files, DVDs, VCDs, etc.

package	popcon	size	area	description
icedtea-plugin	I:22	19	main	Java plugin based on OpenJDK and IcedTea
mozilla-plugin-gnash	I:1	39	main	Flash plugin based on Gnash
browser-plugin-vlc	V:4, I:15	202	main	Multimedia plugin based on VLC media player

Table 6.2: List of browser plugin packages

Tip
Although use of above Debian packages are much easier, browser plugins can be still manually enabled by installing ”*.so” into plugin directories (e.g., "/usr/lib/iceweasel/plugins/") and restarting browsers.

Some web sites refuse to be connected based on the user-agent string of your browser. You can work around this situation by spoofing the user-agent string. For example, you can do this by adding following line into user configuration files such as ”~/.gnome2/epiphany/mozilla/epiphany/user.js” or ”~/.mozilla/firefox/*.default/user.js”.

```
user_pref{"general.useragent.override","Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0) ←↩
    "};
```

Alternatively, you can add and reset this variable by typing ”about:config” into URL and right clicking its display contents.

⚠ **Caution**
Spoofed user-agent string may cause bad side effects with Java.

6.2   The mail system

⚠ **Caution**
If you are to set up the mail server to exchange mail directly with the Internet, you should be better than reading this elementary document.

The mail system involves many server programs and many client programs running on multiple hosts. From the functionality, there are 3 types of mail agent programs:

- The mail transport agent (MTA, see Section 6.3) is a program for transferring mails between different hosts.

- The mail delivery agent (MDA, see Section 6.6) is a program for delivering messages to the users' mailboxes within a host.

- The mail user agent (MUA, also known as email client, see Section 6.4) is the program to generate messages and to access delivered messages.

Note
The following configuration examples are only valid for the typical mobile workstation on consumer grade Internet connections.

6.2.1 Email basics

An email message consists of three components, the message envelope, the message header, and the message body.

The "To" and "From" information in the message envelope is used by the SMTP to deliver the email. (The "From" information in the message envelope is also called bounce address, From_, etc.).

The "To" and "From" information in the message header is displayed by the email client. (While it is most common for these to be the same as ones in the message envelope, such is not always the case.)

The email client (MUA) needs to interpret the message header and body data using Multipurpose Internet Mail Extensions (MIME) to deal the content data type and encoding.

6.2.2 Modern mail service basics

In order to minimize exposure to the spam (unwanted and unsolicited email) problems, many ISPs which provide consumer grade Internet connections are implementing counter measures.

- The smarthost service for their customers to send message uses the message submission port (587) specified in rfc4409 with the password (SMTP AUTH service) specified in rfc4954.

- The SMTP port (25) connection from their internal network hosts (except ISP's own outgoing mail server) to the Internet are blocked.

- The SMTP port (25) connection to the ISP's incoming mail server from some suspicious external network hosts are blocked. (The connection from hosts on the dynamic IP address range used by the dial-up and other consumer grade Internet connections are the first ones to be blocked.)

- Anti-spam techniques such as DomainKeys Identified Mail (DKIM), Sender_Policy_Framework (SPF), and Domain-based Message Authentication, Reporting and Conformance (DMARC) are widely used for the email filtering.

- The DomainKeys Identified Mail service may be provided for your mail sent through the smarthost.

- The smarthost may rewrite the source mail address to your mail account on the smarthost.

When configuring your mail system or resolving mail delivery problems, you must consider these new limitations.

Caution
It is not realistic to run SMTP server on consumer grade network to send mail directly to the remote host reliably.

Caution
It is not realistic to expect a single smarthost to send mails of unrelated source mail addresses to the remote host reliably.

> ⚠ **Caution**
> A mail may be rejected by any host en route to the destination quietly. Making your mail to appear as authentic as possible is the only way to send a mail to the remote host reliably.

In light of these hostile Internet situation and limitations, some independent Internet mail ISPs such as Yahoo.com and Gmail.com offer the secure mail service which can be connected from anywhere on the Internet using Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL).

- The smarthost service on port 465 with the deprecated SMTP over SSL (SMTPS protocol).

- The smarthost service on port 587 with the STARTTLS.

- The incoming mail is accessible at the TLS/POP3 port (995) with POP3.

For the simplicity, I assume that the smarthost is located at "`smtp.hostname.dom`", requires SMTP Authentication, and uses the message submission port (587) with the STARTTLS in the following text.

6.2.3   The mail configuration strategy for workstation

The most simple mail configuration is that the mail is sent to the ISP's smarthost and received from ISP's POP3 server by the MUA (see Section 6.4) itself. This type of configuration is popular with full featured GUI based MUA such as icedove(1), evolution(1), etc. If you need to filter mail by their types, you use MUA's filtering function. For this case, the local MTA (see Section 6.3) need to do local delivery only (when sender and receiver are on the same host).

Please note that the Debian system is the multiuser system. Even if you are the only user, there are many programs running as root and they may send you a mail.

The alternative mail configuration is that the mail is sent via local MTA to the ISP's smarthost and received from ISP's POP3 by the mail retriever (see Section 6.5) to the local mailbox. If you need to filter mail by their types, you use MDA with filter (see Section 6.6) to filter mail into separate mailboxes. This type of configuration is popular with simple console based MUA such as mutt(1), mew(1), etc., although this is possible with any MUAs (see Section 6.4). For this case, the local MTA (see Section 6.3) need to do both smarthost delivery and local delivery. Since mobile workstation does not have valid FQDN, you must configure the local MTA to hide and spoof the real local mail name in outgoing mail to avoid mail delivery errors (see Section 6.3.3).

> **Tip**
> You may wish to configure MUA/MDA to use Maildir for storing email messages somewhere under your home directory.

6.3   Mail transport agent (MTA)

For normal workstation, the popular choice for Mail transport agent (MTA) is either `exim4-*` or `postfix` packages. It is really up to you.

Although the popcon vote count of `exim4-*` looks several times popular than that of `postfix`, this does not mean `postfix` is not popular with Debian developers. The Debian server system uses both `exim4` and `postfix`. The mail header analysis of mailing list postings from prominent Debian developers also indicate both of these MTAs are as popular.

The `exim4-*` packages are known to have very small memory consumption and very flexible for its configuration. The `postfix` package is known to be compact, fast, simple, and secure. Both come with ample documentation and are as good in quality and license.

There are many choices for mail transport agent (MTA) packages with different capability and focus in Debian archive.

package	popcon	size	description
exim4-daemon-light	V:503, I:593	1289	Exim4 mail transport agent (MTA: Debian default)
exim4-base	V:511, I:604	1585	Exim4 documentation (text) and common files
exim4-doc-html	I:1	3484	Exim4 documentation (html)
exim4-doc-info	I:1	593	Exim4 documentation (info)
postfix	V:145, I:160	3966	Postfix mail transport agent (MTA: alternative)
postfix-doc	I:10	4099	Postfix documentation (html+text)
sasl2-bin	V:6, I:22	410	Cyrus SASL API implementation (supplement postfix for SMTP AUTH)
cyrus-sasl2-doc	I:1	247	Cyrus SASL - documentation

Table 6.3: List of basic mail transport agent related packages for workstation

package	popcon	size	capability and focus
exim4-daemon-light	V:503, I:593	1289	full
postfix	V:145, I:160	3966	full (security)
exim4-daemon-heavy	V:8, I:9	1430	full (flexible)
sendmail-bin	V:12, I:13	1811	full (only if you are already familiar)
nullmailer	V:5, I:8	423	strip down, no local mail
ssmtp	V:13, I:19	2	strip down, no local mail
courier-mta	V:0, I:0	2079	very full (web interface etc.)
masqmail	V:0, I:0	337	light
esmtp	V:0, I:0	124	light
esmtp-run	V:0, I:0	32	light (sendmail compatibility extension to esmtp)
msmtp	V:2, I:7	260	light
msmtp-mta	V:1, I:1	35	light (sendmail compatibility extension to msmtp)

Table 6.4: List of choices for mail transport agent (MTA) packages in Debian archive

6.3.1 The configuration of exim4

> **⚠ Caution**
> Configuring `exim4` to send the Internet mail via multiple corresponding smarthosts for multiple source email addresses is non-trivial. Please set up `exim4` only for a single email address for the system programs such as `popcon` and `cron` and set up `msmtp` for multiple source email addresses for the user programs such as `mutt`.

For the Internet mail via smarthost, you (re)configure `exim4-*` packages as the following.

```
$ sudo /etc/init.d/exim4 stop
$ sudo dpkg-reconfigure exim4-config
```

Select "mail sent by smarthost; received via SMTP or fetchmail" for "General type of mail configuration".

Set "System mail name:" to its default as the FQDN (see Section 5.1.1).

Set "IP-addresses to listen on for incoming SMTP connections:" to its default as "127.0.0.1 ; ::1".

Unset contents of "Other destinations for which mail is accepted:".

Unset contents of "Machines to relay mail for:".

Set "IP address or host name of the outgoing smarthost:" to "smtp.hostname.dom:587".

Select "<No>" for "Hide local mail name in outgoing mail?". (Use "`/etc/email-addresses`" as in Section 6.3.3, instead.)

Reply to "Keep number of DNS-queries minimal (Dial-on-Demand)?" as one of the following.

- "No" if the system is connected to the Internet while booting.

- "Yes" if the system is **not** connected to the Internet while booting.

Set "Delivery method for local mail:" to "mbox format in /var/mail/".

Select "<Yes>" for "Split configuration into small files?:".

Create password entries for the smarthost by editing "`/etc/exim4/passwd.client`".

```
$ sudo vim /etc/exim4/passwd.client
 ...
$ cat /etc/exim4/passwd.client
^smtp.*\.hostname\.dom:username@hostname.dom:password
```

Start `exim4` by the following.

```
$ sudo /etc/init.d/exim4 start
```

The host name in "`/etc/exim4/passwd.client`" should not be the alias. You check the real host name with the following.

```
$ host smtp.hostname.dom
smtp.hostname.dom is an alias for smtp99.hostname.dom.
smtp99.hostname.dom has address 123.234.123.89
```

I use regex in "`/etc/exim4/passwd.client`" to work around the alias issue. SMTP AUTH probably works even if the ISP moves host pointed by the alias.

You can manually update `exim4` configuration by the following:

- Update `exim4` configuration files in "`/etc/exim4/`".

  - creating "`/etc/exim4/exim4.conf.localmacros`" to set MACROs and editing "`/etc/exim4/exim4.conf.templa`
    (non-split configuration)

– creating new files or editing existing files in the "`/etc/exim4/exim4.conf.d`" subdirectories. (split configuration)

• Run "`invoke-rc.d exim4 reload`".

Please read the official guide at: "`/usr/share/doc/exim4-base/README.Debian.gz`" and update-exim4.conf(8).

> ⚠ **Caution**
> Starting `exim4` takes long time if "No" (default value) was chosen for the debconf query of "Keep number of DNS-queries minimal (Dial-on-Demand)?" and the system is **not** connected to the Internet while booting.

> ⚠ **Warning**
> It is insecure to use plain text password without encryption even if your ISP allows it.

Tip
Although use of SMTP with STARTTLS on port 587 is recommended, some ISPs still use deprecated SMTPS (SSL on port 465). Exim4 after 4.77 supports this deprecated SMTPS protocol for both as client and as server.

Tip
If you are looking for a light weight MTA that respects "`/etc/aliases`" for your laptop PC, you should consider to configure exim4(8) with "`QUEUERUNNER='queueonly'`", "`QUEUERUNNER='nodaemon'`", etc. in "`/etc/default/exim4`".

6.3.2 The configuration of postfix with SASL

For the Internet mail via smarthost, you should first read postfix documentation and key manual pages.

command	function
postfix(1)	Postfix control program
postconf(1)	Postfix configuration utility
postconf(5)	Postfix configuration parameters
postmap(1)	Postfix lookup table maintenance
postalias(1)	Postfix alias database maintenance

Table 6.5: List of important postfix manual pages

You (re)configure `postfix` and `sasl2-bin` packages as follows.

```
$ sudo /etc/init.d/postfix stop
$ sudo dpkg-reconfigure postfix
```

Chose "Internet with smarthost".

Set "SMTP relay host (blank for none):" to "`[smtp.hostname.dom]:587`" and configure it by the following.

```
$ sudo postconf -e 'smtp_sender_dependent_authentication = yes'
$ sudo postconf -e 'smtp_sasl_auth_enable = yes'
$ sudo postconf -e 'smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd'
$ sudo postconf -e 'smtp_sasl_type = cyrus'
$ sudo vim /etc/postfix/sasl_passwd
```

Create password entries for the smarthost.

```
$ cat /etc/postfix/sasl_passwd
[smtp.hostname.dom]:587     username:password
$ sudo postmap hush:/etc/postfix/sasl_passwd
```

Start the `postfix` by the following.

```
$ sudo /etc/init.d/postfix start
```

Here the use of "[" and "]" in the `dpkg-reconfigure` dialog and "/etc/postfix/sasl_passwd" ensures not to check MX record but directly use exact hostname specified. See "Enabling SASL authentication in the Postfix SMTP client" in "/usr/share/doc/postfix/html/SASL_README.html".

6.3.3 The mail address configuration

There are a few mail address configuration files for mail transport, delivery and user agents.

file	function	application
/etc/mailname	default host name for (outgoing) mail	Debian specific, mailname(5)
/etc/email-addresses	host name spoofing for outgoing mail	exim(8) specific, exim4-config_files(5)
/etc/postfix/generic	host name spoofing for outgoing mail	postfix(1) specific, activated after postmap(1) command execution.
/etc/aliases	account name alias for incoming mail	general, activated after newaliases(1) command execution.

Table 6.6: List of mail address related configuration files

The **mailname** in the "/etc/mailname" file is usually a fully qualified domain name (FQDN) that resolves to one of the host's IP addresses. For the mobile workstation which does not have a hostname with resolvable IP address, set this **mailname** to the value of "hostname -f". (This is safe choice and works for both `exim4-*` and `postfix`.)

Tip
The contents of "/etc/mailname" is used by many non-MTA programs for their default behavior. For `mutt`, set "hostname" and "from" variables in ~/muttrc file to override the **mailname** value. For programs in the `devscripts` package, such as bts(1) and dch(1), export environment variables "$DEBFULLNAME" and "$DEBEMAIL" to override it.

Tip
The popularity-contest package normally send mail from root account with FQDN. You need to set MAILFROM in /etc/popularity-contest.conf as described in the /usr/share/popularity-contest/default.conf file. Otherwise, your mail will be rejected by the smarthost SMTP server. Although this is tedious, this approach is safer than rewriting the source address for all mails from root by MTA and should be used for other daemons and cron scripts.

When setting the **mailname** to "hostname -f", the spoofing of the source mail address via MTA can be realized by the following.

- "/etc/email-addresses" file for exim4(8) as explained in the exim4-config_files(5)

- "/etc/postfix/generic" file for postfix(1) as explained in the generic(5)

For `postfix`, the following extra steps are needed.

```
# postmap hash:/etc/postfix/generic
# postconf -e 'smtp_generic_maps = hash:/etc/postfix/generic'
# postfix reload
```

You can test mail address configuration using the following.

- exim(8) with `-brw, -bf, -bF, -bV,` ⋯ options

- postmap(1) with `-q` option.

Tip
Exim comes with several utility programs such as exiqgrep(8) and exipick(8). See "`dpkg -L exim4-base|grep man8/`" for available commands.

6.3.4 Basic MTA operations

There are several basic MTA operations. Some may be performed via sendmail(1) compatibility interface.

exim command	postfix command	description
`sendmail`	`sendmail`	read mails from standard input and arrange for delivery (`-bm`)
`mailq`	`mailq`	list the mail queue with status and queue ID (`-bp`)
`newaliases`	`newaliases`	initialize alias database (`-I`)
`exim4 -q`	`postqueue -f`	flush waiting mails (`-q`)
`exim4 -qf`	`postsuper -r ALL deferred; postqueue -f`	flush all mails
`exim4 -qff`	`postsuper -r ALL; postqueue -f`	flush even frozen mails
`exim4 -Mg queue_id`	`postsuper -h queue_id`	freeze one message by its queue ID
`exim4 -Mrm queue_id`	`postsuper -d queue_id`	remove one message by its queue ID
N/A	`postsuper -d ALL`	remove all messages

Table 6.7: List of basic MTA operation

Tip
It may be a good idea to flush all mails by a script in "`/etc/ppp/ip-up.d/*`".

6.4 Mail user agent (MUA)

If you subscribe to Debian related mailing list, it may be a good idea to use such MUA as `mutt` and `mew` which are the de facto standard for the participant and known to behave as expected.

6.4.1 Basic MUA —Mutt

Customize "`~/.muttrc`" as the following to use `mutt` as the mail user agent (MUA) in combination with `vim`.

package	popcon	size	type
evolution	V:42, I:284	441	X GUI program (GNOME3, groupware suite)
thunderbird	V:60, I:134	107407	X GUI program (GNOME2, unbranded Mozilla Thunderbird)
kmail	V:45, I:98	17211	X GUI program (KDE)
mutt	V:133, I:572	5951	character terminal program probably used with `vim`
mew	V:0, I:0	2328	character terminal program under `(x)emacs`

Table 6.8: List of mail user agent (MUA)

```
#
# User configuration file to override /etc/Muttrc
#
# spoof source mail address
set use_from
set hostname=example.dom
set from="Name Surname <username@example.dom>"
set signature="~/.signature"

# vim: "gq" to reformat quotes
set editor="vim -c 'set tw=72 et ft=mail'"

# "mutt" goes to Inbox, while "mutt -y" lists mailboxes
set mbox_type=Maildir          # use qmail Maildir format for creating mbox
set mbox=~/Mail                # keep all mail boxes in $HOME/Mail/
set spoolfile=+Inbox           # mail delivered to $HOME/Mail/Inbox
set record=+Outbox             # save fcc mail to $HOME/Mail/Outbox
set postponed=+Postponed       # keep postponed in $HOME/Mail/postponed
set move=no                    # do not move Inbox items to mbox
set quit=ask-yes               # do not quit by "q" only
set delete=yes                 # always delete w/o asking while exiting
set fcc_clear                  # store fcc as non encrypted

# Mailboxes in Maildir (automatic update)
mailboxes 'cd ~/Mail; /bin/ls -1|sed -e 's/^/+/' | tr "\n" " "'
unmailboxes Maillog *.ev-summary

## Default
#set index_format="%4C %Z %{%b %d} %-15.15L (%4l) %s"
## Thread index with senders (collapse)
set index_format="%4C %Z %{%b %d} %-15.15n %?M?(#%03M)&(%4l)? %s"

## Default
#set folder_format="%2C %t %N %F %2l %-8.8u %-8.8g %8s %d %f"
## just folder names
set folder_format="%2C %t %N %f"
```

Add the following to "`/etc/mailcap`" or "`~/.mailcap`" to display HTML mail and MS Word attachments inline.

```
text/html; lynx -force_html %s; needsterminal;
application/msword; /usr/bin/antiword '%s'; copiousoutput; description="Microsoft Word Text ←
    "; nametemplate=%s.doc
```

Tip
Mutt can be used as the IMAP client and the mailbox format converter. You can tag messages with "t", "T", etc. These tagged messages can be copied with ";C" between different mailboxes and deleted with ";d" in one action.

6.4.2 Advanced MUA —Mutt + msmtp

Mutt can be configured to use multiple source email addresses with multiple corresponding smarthosts using msmtp.

Tip
Msmtp is a sendmail emulator which allows to be installed along another sendmail emulator which provides the
`/usr/sbin/sendmail` command. So you can leave your system mail to be `exim4` or `postfix`.

Let's think about supporting 3 email addresses as an example:

- "My Name1 <myaccount1@gmail.com>"

- "My Name2 <myaccount2@gmail.com>"

- "My Name3 <myaccount3@example.org>"

Here is an example of `~/.muttrc` customization supporting 3 smarthosts for 3 different source email addresses.

```
set use_from
set from="My Name3 <myaccount3@example.org>"
set reverse_name
alternates myaccount1@gmail\.com|myaccount1@gmail\.com|myaccount3@example\.org

# ...

# MACRO
macro compose "1" "<edit-from>^UMy Name1 \<myaccount1@gmail.com\>\n"
macro compose "2" "<edit-from>^UMy Name2 \<myaccount2@gmail.com\>\n"
macro compose "3" "<edit-from>^UMy Name3 \<myaccount3@example.org\>\n"

send2-hook '~f myaccount1@gmail.com' "set sendmail = '/usr/bin/msmtp --read-envelope-from'"
send2-hook '~f myaccount2@gmail.com' "set sendmail = '/usr/bin/msmtp --read-envelope-from'"
send2-hook '~f myaccount3@example.org' "set sendmail = '/usr/bin/msmtp --read-envelope-from ←
    '"

# ...
```

Let's install `msmtp-gnome` and set `~/.msmtprc` as follows.

```
defaults
logfile ~/.msmtp.log
domain myhostname.example.org
tls on
tls_starttls on
tls_certcheck on
tls_trust_file /etc/ssl/certs/ca-certificates.crt
auth on
port 587
auto_from

account myaccount1@gmail.com
host smtp.gmail.com
from  myaccount1@gmail.com
user  myaccount1@gmail.com

account myaccount2@gmail.com
host smtp.gmail.com
from  myaccount2@gmail.com
user  myaccount2@gmail.com
```

```
account myaccount3@example.org
host mail.example.org
from  myaccount3@example.org
user  myaccount3@example.org

account default : myaccount3@example.org
```

Then, add password data into the Gnome key ring. For example:

```
 $ secret-tool store --label=msmtp \
     host smtp.gmail.com \
     service smtp \
     user myaccount1@gmail.com
 ...
```

Tip

If you don't wish to use the Gnome key ring, you can install `msmtp` package instead and add an entry such as "`password secret123`" to each account in `~/.msmtprc`. See [memtp documentation](#) for more.

6.5   The remote mail retrieval and forward utility

Instead of running a MUA to access remote mails and to process them manually, you may wish to automate such process to have all the mails delivered to the local host. The remote mail retrieval and forward utility is the tool for you.

Although fetchmail(1) has been de facto standard for the remote mail retrieval on GNU/Linux, the author likes getmail(1) now. If you want to reject mail before downloading to save bandwidth, `mailfilter` or `mpop` may be useful. Whichever mail retriever utilities are used, it is a good idea to configure system to deliver retrieved mails to MDA, such as `maildrop`, via pipe.

package	popcon	size	description
fetchmail	V:7, I:19	2343	mail retriever (POP3, APOP, IMAP) (old)
getmail4	V:3, I:10	32 637	mail retriever (POP3, IMAP4, and SDPS) (simple, secure, and reliable)
mailfilter	V:0, I:0	274	mail retriever (POP3) with with regex filtering capability
mpop	V:0, I:0	222	mail retriever (POP3) and MDA with filtering capability

Table 6.9: List of remote mail retrieval and forward utilities

6.5.1   getmail configuration

getmail(1) configuration is described in [getmail documentation](#). Here is my set up to access multiple POP3 accounts as user.

Create "`/usr/local/bin/getmails`" as the following.

```
#!/bin/sh
set -e
if [ -f $HOME/.getmail/running ]; then
  echo "getmail is already running ... (if not, remove $HOME/.getmail/running)" >&2
  pgrep -l "getmai[l]"
  exit 1
else
  echo "getmail has not been running ... " >&2
fi
if [ -f $HOME/.getmail/stop ]; then
  echo "do not run getmail ... (if not, remove $HOME/.getmail/stop)" >&2
```

```
  exit
fi
if [ "x$1" = "x-l" ]; then
  exit
fi
rcfiles="/usr/bin/getmail"
for file in $HOME/.getmail/config/* ; do
  rcfiles="$rcfiles --rcfile $file"
done
date -u > $HOME/.getmail/running
eval "$rcfiles $@"
rm $HOME/.getmail/running
```

Configure it as the following.

```
$ sudo chmod 755 /usr/local/bin/getmails
$ mkdir -m 0700 $HOME/.getmail
$ mkdir -m 0700 $HOME/.getmail/config
$ mkdir -m 0700 $HOME/.getmail/log
```

Create configuration files "$HOME/.getmail/config/pop3_name" for each POP3 accounts as the following.

```
[retriever]
type = SimplePOP3SSLRetriever
server = pop.example.com
username =  pop3_name@example.com
password = <your-password>

[destination]
type = MDA_external
path = /usr/bin/maildrop
unixfrom = True

[options]
verbose = 0
delete = True
delivered_to = False
message_log = ~/.getmail/log/pop3_name.log
```

Configure it as the following.

```
$ chmod 0600 $HOME/.getmail/config/*
```

Schedule "/usr/local/bin/getmails" to run every 15 minutes with cron(8) by executing "sudo crontab -e -u <user_name>" and adding following to user's cron entry.

```
5,20,35,50 * * * * /usr/local/bin/getmails --quiet
```

Tip
Problems of POP3 access may not come from getmail. Some popular free POP3 services may be violating the POP3 protocol and their SPAM filter may not be perfect. For example, they may delete messages just after receiving RETR command before receiving DELE command and may quarantined messages into Spam mailbox. You should minimize damages by configuring them to archive accessed messages and not to delete them. See also "Some mail was not downloaded".

6.5.2   fetchmail configuration

fetchmail(1) configuration is set by "/etc/default/fetchmail", "/etc/fetchmailrc" and "$HOME/.fetchmailrc". See its example in "/usr/share/doc/fetchmail/examples/fetchmailrc.example".

6.6   Mail delivery agent (MDA) with filter

Most MTA programs, such as `postfix` and `exim4`, function as MDA (mail delivery agent). There are specialized MDA with filtering capabilities.

Although procmail(1) has been de facto standard for MDA with filter on GNU/Linux, author likes maildrop(1) now. Whichever filtering utilities are used, it is a good idea to configure system to deliver filtered mails to a qmail-style Maildir.

package	popcon	size	description
procmail	V:65, I:571	300	MDA with filter (old)
mailagent	V:0, I:10	1283	MDA with Perl filter
maildrop	V:1, I:3	1093	MDA with structured filtering language

Table 6.10: List of MDA with filter

6.6.1   maildrop configuration

maildrop(1) configuration is described in maildropfilter documentation. Here is a configuration example for "`$HOME/.mailfilter`".

```
# Local configuration
MAILROOT="$HOME/Mail"
# set this to /etc/mailname contents
MAILHOST="example.dom"
logfile $HOME/.maildroplog

# rules are made to override the earlier value by the later one.

# mailing list mails ?
if (     /^Precedence:.*list/:h || /^Precedence:.*bulk/:h )
{
    # rules for mailing list mails
    # default mailbox for mails from mailing list
    MAILBOX="Inbox-list"
    # default mailbox for mails from debian.org
    if ( /^(Sender|Resent-From|Resent-Sender): .*debian.org/:h )
    {
        MAILBOX="service.debian.org"
    }
    # default mailbox for mails from bugs.debian.org (BTS)
    if ( /^(Sender|Resent-From|Resent-sender): .*@bugs.debian.org/:h )
    {
        MAILBOX="bugs.debian.org"
    }
    # mailbox for each properly maintained mailing list with "List-Id: foo" or "List-Id:  ←
        ...<foo.bar>"
    if ( /^List-Id: ([^<]*<)?([^<>]*)>?/:h )
    {
        MAILBOX="$MATCH2"
    }
}
else
{
    # rules for non-mailing list mails
    # default incoming box
    MAILBOX="Inbox-unusual"
    # local mails
    if ( /Envelope-to: .*@$MAILHOST/:h )
    {
```

```
        MAILBOX="Inbox-local"
    }
    # html mails (99% spams)
    if ( /DOCTYPE html/:b ||\
        /^Content-Type: text\/html/ )
    {
        MAILBOX="Inbox-html"
    }
    # blacklist rule for spams
    if ( /^X-Advertisement/:h ||\
        /^Subject:.*BUSINESS PROPOSAL/:h ||\
        /^Subject:.*URGENT.*ASISSTANCE/:h ||\
        /^Subject: *I NEED YOUR ASSISTANCE/:h )
    {
        MAILBOX="Inbox-trash"
    }
    # whitelist rule for normal mails
    if ( /^From: .*@debian.org/:h ||\
        /^(Sender|Resent-From|Resent-Sender): .*debian.org/:h ||\
        /^Subject: .*(debian|bug|PATCH)/:h )
    {
        MAILBOX="Inbox"
    }
    # whiltelist rule for BTS related mails
    if ( /^Subject: .*Bug#.*/:h ||\
        /^(To|Cc): .*@bugs.debian.org/:h )
    {
        MAILBOX="bugs.debian.org"
    }
    # whitelist rule for getmails cron mails
    if ( /^Subject: Cron .*getmails/:h )
    {
        MAILBOX="Inbox-getmails"
    }
}

# check existance of $MAILBOX
'test -d $MAILROOT/$MAILBOX'
if ( $RETURNCODE == 1 )
{
    # create maildir mailbox for $MAILBOX
    'maildirmake $MAILROOT/$MAILBOX'
}
# deliver to maildir $MAILBOX
to "$MAILROOT/$MAILBOX/"
exit
```

> ⚠ **Warning**
> Unlike `procmail`, `maildrop` does not create missing maildir directories automatically. You must create
> them manually using maildirmake(1) in advance as in the example "`$HOME/.mailfilter`".

6.6.2   procmail configuration

Here is a similar configuration with "`$HOME/.procmailrc`" for procmail(1).

```
MAILDIR=$HOME/Maildir
DEFAULT=$MAILDIR/Inbox/
```

```
LOGFILE=$MAILDIR/Maillog
# clearly bad looking mails: drop them into X-trash and exit
:0
* 1^0 ^X-Advertisement
* 1^0 ^Subject:.*BUSINESS PROPOSAL
* 1^0 ^Subject:.*URGENT.*ASISSTANCE
* 1^0 ^Subject: *I NEED YOUR ASSISTANCE
X-trash/

# Delivering mailinglist messages
:0
* 1^0 ^Precedence:.*list
* 1^0 ^Precedence:.*bulk
* 1^0 ^List-
* 1^0 ^X-Distribution:.*bulk
{
:0
* 1^0 ^Return-path:.*debian-devel-admin@debian.or.jp
jp-debian-devel/

:0
* ^Resent-Sender.*debian-user-request@lists.debian.org
debian-user/

:0
* ^Resent-Sender.*debian-devel-request@lists.debian.org
debian-devel/

:0
* ^Resent-Sender.*debian-announce-request@lists.debian.org
debian-announce

:0
mailing-list/
}

:0
Inbox/
```

6.6.3 Redeliver mbox contents

You need to manually deliver mails to the sorted mailboxes in your home directory from "`/var/mail/<username>`" if your home directory became full and procmail(1) failed. After making disk space in the home directory, run the following.

```
# /etc/init.d/${MAILDAEMON} stop
# formail -s procmail </var/mail/<username>
# /etc/init.d/${MAILDAEMON} start
```

6.7 POP3/IMAP4 server

If you are to run a private server on LAN, you may consider to run POP3 / IMAP4 server for delivering mail to LAN clients.

6.8 The print server and utilities

In the old Unix-like system, the BSD Line printer daemon was the standard. Since the standard print out format of the free software is PostScript on the Unix like system, some filter system was used along with Ghostscript to enable printing to the

package	popcon	size	type	description
courier-pop	V:3, I:5	240	POP3	Courier mail server - POP3 server (maildir format only)
cyrus-pop3d	V:0, I:0	147	POP3	Cyrus mail system (POP3 support)
courier-imap	V:5, I:7	545	IMAP	Courier mail server - IMAP server (maildir format only)
cyrus-imapd	V:0, I:1	332	IMAP	Cyrus mail system (IMAP support)

Table 6.11: List of POP3/IMAP4 servers

non-PostScript printer.

Recently, Common UNIX Printing System (CUPS) is the new de facto standard. The CUPS uses Internet Printing Protocol (IPP). The IPP is now supported by other OSs such as Windows XP and Mac OS X and has became new cross-platform de facto standard for remote printing with bi-directional communication capability.

The standard printable data format for the application on the Debian system is the PostScript (PS) which is a page description language. The data in PS format is fed into the Ghostscript PostScript interpreter to produce the printable data specific to the printer. See Section 11.4.1.

Thanks to the file format dependent auto-conversion feature of the CUPS system, simply feeding any data to the lpr command should generate the expected print output. (In CUPS, lpr can be enabled by installing the cups-bsd package.)

The Debian system has some notable packages for the print servers and utilities.

package	popcon	size	port	description
lpr	V:6, I:8	324	printer (515)	BSD lpr/lpd (Line printer daemon)
lprng	V:1, I:1	3852	, ,	, , (Enhanced)
cups	V:260, I:449	764	IPP (631)	Internet Printing CUPS server
cups-client	V:77, I:504	473	, ,	System V printer commands for CUPS: lp(1), lpstat(1), lpoptions(1), cancel(1), lpmove(8), lpinfo(8), lpadmin(8), ⋯
cups-bsd	V:40, I:433	115	, ,	BSD printer commands for CUPS: lpr(1), lpq(1), lprm(1), lpc(8)
printer-driver-gutenprint	V:91, I:431	881	Not applicable	printer drivers for CUPS

Table 6.12: List of print servers and utilities

Tip
You can configure CUPS system by pointing your web browser to "http://localhost:631/" .

6.9 The remote access server and utilities (SSH)

The Secure SHell (SSH) is the **secure** way to connect over the Internet. A free version of SSH called OpenSSH is available as openssh-client and openssh-server packages in Debian.

Caution
See Section 4.7.3 if your SSH is accessible from the Internet.

package	popcon	size	tool	description
openssh-client	V:821, I:996	4151	ssh(1)	Secure shell client
openssh-server	V:680, I:802	873	sshd(8)	Secure shell server
ssh-askpass-fullscreen	V:0, I:1	42	ssh-askpass-fullscreen(1)	asks user for a pass phrase for ssh-add (GNOME2)
ssh-askpass	V:3, I:42	101	ssh-askpass(1)	asks user for a pass phrase for ssh-add (plain X)

Table 6.13: List of remote access server and utilities

Tip
Please use the screen(1) program to enable remote shell process to survive the interrupted connection (see Section 9.1).

6.9.1 Basics of SSH

⚠ Warning
"/etc/ssh/sshd_not_to_be_run" must not be present if one wishes to run the OpenSSH server.

SSH has two authentication protocols.

SSH protocol	SSH method	description
SSH-1	"RSAAuthentication"	RSA identity key based user authentication
,,	"RhostsAuthentication"	".rhosts" based host authentication (insecure, disabled)
,,	"RhostsRSAAuthentication"	".rhosts" based host authentication combined with RSA host key (disabled)
,,	"ChallengeResponseAuthentication"	RSA challenge-response authentication
,,	"PasswordAuthentication"	password based authentication
SSH-2	"PubkeyAuthentication"	public key based user authentication
,,	"HostbasedAuthentication"	"~/.rhosts" or "/etc/hosts.equiv" based host authentication combined with public key client host authentication (disabled)
,,	"ChallengeResponseAuthentication"	challenge-response authentication
,,	"PasswordAuthentication"	password based authentication

Table 6.14: List of SSH authentication protocols and methods

⚠ Caution
Be careful about these differences if you are using a non-Debian system.

See "/usr/share/doc/ssh/README.Debian.gz", ssh(1), sshd(8), ssh-agent(1), and ssh-keygen(1) for details.

Following are the key configuration files.

configuration file	description of configuration file
`/etc/ssh/ssh_config`	SSH client defaults, see ssh_config(5)
`/etc/ssh/sshd_config`	SSH server defaults, see sshd_config(5)
`~/.ssh/authorized_keys`	default public SSH keys that clients use to connect to this account on this SSH server
`~/.ssh/identity`	secret SSH-1 RSA key of the user
`~/.ssh/id_rsa`	secret SSH-2 RSA key of the user
`~/.ssh/id_dsa`	secret SSH-2 DSA key of the user

Table 6.15: List of SSH configuration files

Tip
See ssh-keygen(1), ssh-add(1) and ssh-agent(1) for how to use public and secret SSH keys.

Tip
Make sure to verify settings by testing the connection. In case of any problem, use "`ssh -v`".

Tip
You can change the pass phrase to encrypt local secret SSH keys later with "`ssh-keygen -p`".

Tip
You can add options to the entries in "`~/.ssh/authorized_keys`" to limit hosts and to run specific commands. See sshd(8) for details.

The following starts an ssh(1) connection from a client.

command	description
`ssh username@hostname.domain.ext`	connect with default mode
`ssh -v username@hostname.domain.ext`	connect with default mode with debugging messages
`ssh -1 username@hostname.domain.ext`	force to connect with SSH version 1
`ssh -1 -o RSAAuthentication=no -l username hostname.domain.ext`	force to use password with SSH version 1
`ssh -o PreferredAuthentications=password -l username hostname.domain.ext`	force to use password with SSH version 2

Table 6.16: List of SSH client startup examples

If you use the same user name on the local and the remote host, you can eliminate typing "`username@`". Even if you use different user name on the local and the remote host, you can eliminate it using "`~/.ssh/config`". For Debian Salsa service with account name "`foo-guest`", you set "`~/.ssh/config`" to contain the following.

```
Host salsa.debian.org people.debian.org
    User foo-guest
```

For the user, ssh(1) functions as a smarter and more secure telnet(1). Unlike `telnet` command, `ssh` command does not stop on the `telnet` escape character (initial default CTRL-]).

6.9.2   Port forwarding for SMTP/POP3 tunneling

To establish a pipe to connect to port 25 of `remote-server` from port 4025 of `localhost`, and to port 110 of `remote-server` from port 4110 of `localhost` through `ssh`, execute on the local host as the following.

```
# ssh -q -L 4025:remote-server:25 4110:remote-server:110 username@remote-server
```

This is a secure way to make connections to SMTP/POP3 servers over the Internet. Set the "`AllowTcpForwarding`" entry to "`yes`" in "`/etc/ssh/sshd_config`" of the remote host.

6.9.3   Connecting without remote passwords

One can avoid having to remember passwords for remote systems by using "`RSAAuthentication`" (SSH-1 protocol) or "`PubkeyAuthentication`" (SSH-2 protocol).

On the remote system, set the respective entries, "`RSAAuthentication yes`" or "`PubkeyAuthentication yes`", in "`/etc/ssh/sshd_config`".

Generate authentication keys locally and install the public key on the remote system by the following.

• "`RSAAuthentication`": RSA key for SSH-1 (deprecated because it is superseded.)

```
$ ssh-keygen
$ cat .ssh/identity.pub | ssh user1@remote "cat - >>.ssh/authorized_keys"
```

• "`PubkeyAuthentication`": RSA key for SSH-2

```
$ ssh-keygen -t rsa
$ cat .ssh/id_rsa.pub | ssh user1@remote "cat - >>.ssh/authorized_keys"
```

• "`PubkeyAuthentication`": DSA key for SSH-2 (deprecated because it is slow.)

```
$ ssh-keygen -t dsa
$ cat .ssh/id_dsa.pub | ssh user1@remote "cat - >>.ssh/authorized_keys"
```

Tip
Use of DSA key for SSH-2 is deprecated because key is smaller and slow. There are no more reasons to work around RSA patent using DSA since it has been expired. DSA stands for Digital Signature Algorithm and is slow. Also see DSA-1571-1.

Note
For "HostbasedAuthentication" to work in SSH-2, you must adjust the settings of "HostbasedAuthentication" to "yes" in both "/etc/ssh/sshd_config" on the server host and "/etc/ssh/ssh_config" or "~/.ssh/config" on the client host.

6.9.4   Dealing with alien SSH clients

There are some free SSH clients available for other platforms.

environment	free SSH program
Windows	puTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/) (GPL)
Windows (cygwin)	SSH in cygwin (http://www.cygwin.com/) (GPL)
Macintosh Classic	macSSH (http://www.macssh.com/) (GPL)
Mac OS X	OpenSSH; use `ssh` in the Terminal application (GPL)

Table 6.17: List of free SSH clients for other platforms

6.9.5   Setting up ssh-agent

It is safer to protect your SSH authentication secret keys with a pass phrase. If a pass phrase was not set, use "`ssh-keygen -p`" to set it.

Place your public SSH key (e.g. "`~/.ssh/id_rsa.pub`") into "`~/.ssh/authorized_keys`" on a remote host using a password-based connection to the remote host as described above.

```
$ ssh-agent bash
$ ssh-add ~/.ssh/id_rsa
Enter passphrase for /home/<username>/.ssh/id_rsa:
Identity added: /home/<username>/.ssh/id_rsa (/home/<username>/.ssh/id_rsa)
```

No remote password needed from here on for the next command.

```
$ scp foo <username>@remote.host:foo
```

Press ^D to terminating ssh-agent session.

For the X server, the normal Debian startup script executes `ssh-agent` as the parent process. So you only need to execute `ssh-add` once. For more, read ssh-agent(1) and ssh-add(1).

6.9.6   How to shutdown the remote system on SSH

You need to protect the process doing "`shutdown -h now`" (see Section 1.1.8) from the termination of SSH using the at(1) command (see Section 9.3.13) by the following.

```
# echo "shutdown -h now" | at now
```

Running "`shutdown -h now`" in screen(1) (see Section 9.1) session is another way to do the same.

6.9.7   Troubleshooting SSH

If you have problems, check the permissions of configuration files and run `ssh` with the "`-v`" option.

Use the "`-p`" option if you are root and have trouble with a firewall; this avoids the use of server ports 1 —1023.

If `ssh` connections to a remote site suddenly stop working, it may be the result of tinkering by the sysadmin, most likely a change in "`host_key`" during system maintenance. After making sure this is the case and nobody is trying to fake the remote host by some clever hack, one can regain a connection by removing the "`host_key`" entry from "`~/.ssh/known_hosts`" on the local host.

6.10   Other network application servers

Here are other network application servers.

Common Internet File System Protocol (CIFS) is the same protocol as Server Message Block (SMB) and is used widely by Microsoft Windows.

package	popcon	size	protocol	description
telnetd	V:1, I:3	103	TELNET	TELNET server
telnetd-ssl	V:0, I:0	154	, ,	, , (SSL support)
nfs-kernel-server	V:32, I:88	321	NFS	Unix file sharing
samba	V:111, I:183	10994	SMB	Windows file and printer sharing
netatalk	V:2, I:5	3923	ATP	Apple/Mac file and printer sharing (AppleTalk)
proftpd-basic	V:32, I:40	5028	FTP	General file download
apache2	V:233, I:346	585	HTTP	General web server
squid	V:13, I:16	7112	, ,	General web proxy server
squid3	V:11, I:17	163	, ,	, ,
bind9	V:54, I:78	1870	DNS	IP address for other hosts
isc-dhcp-server	V:22, I:61	1388	DHCP	IP address of client itself

Table 6.18: List of other network application servers

Tip
See Section 4.5.2 for integration of server systems.

Tip
The hostname resolution is usually provided by the DNS server. For the host IP address dynamically assigned by DHCP, Dynamic DNS can be set up for the hostname resolution using bind9 and isc-dhcp-server as described in the DDNS page on the Debian wiki.

Tip
Use of proxy server such as squid is much more efficient for saving bandwidth than use of local mirror server with the full Debian archive contents.

6.11 Other network application clients

Here are other network application clients.

6.12 The diagnosis of the system daemons

The telnet program enables manual connection to the system daemons and its diagnosis.

For testing plain POP3 service, try the following

```
$ telnet mail.ispname.net pop3
```

For testing the TLS/SSL enabled POP3 service by some ISPs, you need TLS/SSL enabled telnet client by the telnet-ssl or openssl packages.

```
$ telnet -z ssl pop.gmail.com 995
```

```
$ openssl s_client -connect pop.gmail.com:995
```

The following RFCs provide required knowledge to each system daemon.

The port usage is described in "/etc/services".

package	popcon	size	protocol	description
netcat	I:33	16	TCP/IP	TCP/IP swiss army knife
openssl	V:794, I:991	1303	SSL	Secure Socket Layer (SSL) binary and related cryptographic tools
stunnel4	V:4, I:17	461	, ,	universal SSL Wrapper
telnet	V:69, I:909	157	TELNET	TELNET client
telnet-ssl	V:1, I:6	208	, ,	, , (SSL support)
nfs-common	V:303, I:563	722	NFS	Unix file sharing
smbclient	V:20, I:205	1710	SMB	MS Windows file and printer sharing client
cifs-utils	V:35, I:119	230	, ,	mount and umount commands for remote MS Windows file
ftp	V:30, I:560	132	FTP	FTP client
lftp	V:6, I:40	2198	, ,	, ,
ncftp	V:4, I:28	1506	, ,	full screen FTP client
wget	V:359, I:998	3069	HTTP and FTP	web downloader
curl	V:120, I:449	377	, ,	, ,
axel	V:0, I:6	186	, ,	accelerated downloader
aria2	V:2, I:17	4557	, ,	accelerated downloader with BitTorrent and Metalink supports
bind9-host	V:442, I:949	172	DNS	host(1) from bind9, ”Priority: standard”
dnsutils	V:80, I:784	489	, ,	dig(1) from bind, ”Priority: standard”
isc-dhcp-client	V:278, I:969	649	DHCP	obtain IP address
ldap-utils	V:18, I:77	680	LDAP	obtain data from LDAP server

Table 6.19: List of network application clients

RFC	description
rfc1939 and rfc2449	POP3 service
rfc3501	IMAP4 service
rfc2821 (rfc821)	SMTP service
rfc2822 (rfc822)	Mail file format
rfc2045	Multipurpose Internet Mail Extensions (MIME)
rfc819	DNS service
rfc2616	HTTP service
rfc2396	URI definition

Table 6.20: List of popular RFCs

Chapter 7

The X Window System

> **⚠ Warning**
> This chapter is getting outdated since this is based on Debian 7.0 (`Wheezy`) released in 2013.

The X Window System on the Debian system is based on the source from X.Org.

7.1 Key packages

There are a few (meta)packages provided to ease installation.

(meta)package	popcon	size	description
xorg	I:513	52	X libraries, an X server, a set of fonts, and a group of basic X clients and utilities (metapackage)
xserver-xorg	V:102, I:553	238	full suite of the X server and its configuration
xbase-clients	I:49	46	miscellaneous assortment of X clients (metapackage)
x11-common	V:408, I:802	305	filesystem infrastructure for the X Window System
xorg-docs	I:7	2032	miscellaneous documentation for the X.Org software suite
menu	V:92, I:278	1435	generate the Debian menu for all menu-aware applications
gksu	V:38, I:253	289	Gtk+ frontend to su(1) or sudo(8)
menu-xdg	V:35, I:139	76	convert the Debian menu structure to the freedesktop.org xdg menu structure
xdg-utils	V:251, I:573	325	utilities to integrate desktop environment provided by the freedesktop.org
task-gnome-desktop	I:213	6	standard GNOME desktop environment (metapackage)
task-kde-desktop	I:70	6	core KDE desktop environment (metapackage)
task-xfce-desktop	I:107	6	Xfce lightweight desktop environment (metapackage)
task-lxde-desktop	I:45	6	LXDE lightweight desktop environment (metapackage)
fluxbox	V:2, I:11	3860	Fluxbox: package for highly configurable and low resource X window manager

Table 7.1: List of key (meta)packages for X Window

For the basics of X, refer to X(7) and the LDP XWindow-User-HOWTO.

7.2 Setting up desktop environment

A desktop environment is usually a combination of a X window manager, a file manager, and a suite of compatible utility programs.

You can setup a full desktop environment such as GNOME, KDE, Xfce, or LXDE, from the `aptitude` under the task menu.

Tip
Task menu may be out of sync with the latest package transition state under Debian `unstable`/`testing` environment. In such situation, you need to deselect some (meta)packages listed under aptitude(8) task menu to avoid package conflicts. When deselecting (meta)packages, you must select certain packages providing their dependencies manually to avoid them deleted automatically.

You may alternatively setup a simple environment manually just with a X window manager such as Fluxbox.

See Window Managers for X for the guide to the X window manager and the desktop environment.

7.2.1 Debian menu

Debian menu system provides a general interface for both text- and X-oriented programs with update-menus(1) from the `menu` package. Each package installs its menu data in the "`/usr/share/menu/`" directory. See "`/usr/share/menu/README`".

7.2.2 Freedesktop.org menu

Each package which is compliant to Freedesktop.org's xdg menu system installs its menu data provided by "`*.desktop`" under "`/usr/share/applications/`". Modern desktop environments which are compliant to Freedesktop.org standard use these data to generate their menu using the `xdg-utils` package. See "`/usr/share/doc/xdg-utils/README`".

7.2.3 Debian menu from Freedesktop.org menu

In order to access the traditional Debian menu from the Freedesktop.org menu compliant window manager environment such as GNOME and KDE, you must install the `menu-xdg` package.

7.3 The server/client relationship

The X Window System is activated as a combination of the server and client programs. The meaning for the words **server** and **client** with respect to the words **local** and **remote** requires attention here.

Modern X servers have the MIT Shared Memory Extension and communicate with their local X clients using the local shared memory. This bypasses the network transparent Xlib interprocess communication channel and gains performance for large images.

7.4 The X server

See xorg(1) for X server information.

type	description
X server	a program run on a **local host** connected to the user's display and input devices.
X client	a program run on a **remote host** that processes data and talks to the X server.
application server	a program run on a **remote host** that processes data and talks to the application clients.
application client	a program run on a **local host** connected to the user's display and input devices.

Table 7.2: List of server/client terminology

7.4.1 The (re)configuration of the X server

The following (re)configures an X server.

```
# dpkg-reconfigure --priority=low x11-common
```

Note
Recent Linux kernels have good graphics and input device supports with DRM, KMS, and udev. X server is rewritten to use them. So "/etc/X11/xorg.conf" is usually not present on your system. These parameters are configured by the kernel. See "fb/modedb.txt" in the Linux kernel documentation.

For the large high resolution CRT monitor, it is a good idea to set the refresh rate as high as your monitor can handle (85 Hz is great, 75 Hz is OK) to reduce flicker. For the LCD monitor, slower standard refresh rate (60Hz) is usually fine due to its slow response.

Note
Be careful not to use too high refresh rate which may cause fatal hardware failure of your monitor system.

7.4.2 The connection methods to the X server

There are several ways of getting the "X server" (**display** side) to accept connections from an "X client" (**application** side).

Warning
Do not use remote TCP/IP connection over **unsecured** network for X connection unless you have very good reason such as use of encryption. A remote TCP/IP socket connection without encryption is prone to the **eavesdropping attack** and is disabled by default on the Debian system. Use "ssh -X".

Warning
Do not use XDMCP connection over **unsecured** network either. It sends data via UDP/IP without encryption and is prone to the **eavesdropping attack**.

Tip
LTSP stands for Linux Terminal Server Project.

---

package	popcon	size	user	encryption	method	pertinent use
xbase-clients	I:49	46	unchecked	no	xhost command	deprecated
xbase-clients	I:49	46	checked	no	xauth command	local connection via pipe
openssh-client	V:821, I:996	4151	checked	yes	ssh -X command	remote network connection
gdm3	V:192, I:281	4697	checked	no (XDMCP)	GNOME display manager	local connection via pipe
sddm	V:36, I:71	1823	checked	no (XDMCP)	KDE display manager	local connection via pipe
xdm	V:3, I:8	665	checked	no (XDMCP)	X display manager	local connection via pipe
wdm	V:126, I:564	2302	checked	no (XDMCP)	WindowMaker display manager	local connection via pipe
ldm	V:0, I:1	413	checked	yes	LTSP display manager	remote SSH network connection (thin client)

Table 7.3: List of connection methods to the X server

7.5 Starting the X Window System

The X Window System is usually started as an X session which is the combination of an X server and connecting X clients. For the normal desktop system, both of them are executed on a workstation.

The X session is started by one of the following.

- startx command started from the command line

- One of the X display manager daemon programs *dm started by systemd based on the dependency of "graphical.target".

Tip
The start up script for the display manager daemons checks the content of the "/etc/X11/default-display-manager" file before actually executing themselves. This ensures to have only one X display manager daemon program activated.

Tip
See Section 8.4.5 for initial environment variables of the X display manager.

Essentially, all these programs execute the "/etc/X11/Xsession" script. Then the "/etc/X11/Xsession" script performs run-parts(8) like action to execute scripts in the "/etc/X11/Xsession.d/" directory. This is essentially an execution of the first program which is found in the following order by the exec builtin command.

1. The script specified as the argument of "/etc/X11/Xsession" by the X display manager, if it is defined.

2. The "~/.xsession" or "~/.Xsession" script, if it is defined.

3. The "/usr/bin/x-session-manager" command, if it is defined.

4. The "/usr/bin/x-window-manager" command, if it is defined.

5. The "/usr/bin/x-terminal-emulator" command, if it is defined.

This process is affected by the content of "/etc/X11/Xsession.options". The exact programs to which these "/usr/bin/x-*"
commands point, are determined by the Debian alternatives system and changed by "update-alternatives --config
x-session-manager", etc.

See Xsession(5) for details.

7.5.1 Starting X session with gdm3

gdm3(1) lets you select the session type (or desktop environment: Section 7.2), and language (or locale: Section 8.4) of the X
session from its menu. It keeps the selected default value in "~/.dmrc" as the following.

```
[Desktop]
Session=default
Language=ja_JP.UTF-8
```

7.5.2 Customizing the X session (classic method)

On a system where "/etc/X11/Xsession.options" contains a line "allow-user-xsession" without preceding "#"
characters, any user who defines "~/.xsession" or "~/.Xsession" is able to customize the action of "/etc/X11/Xsession"
by completely overriding the system code. The last command in the "~/.xsession" file should use form of "exec some-window/s…"
to start your favorite X window/session managers.

If this feature is used, the selection of the display (or login) manager (DM), session manager or window manager (WM) by the
system utility is ignored.

7.5.3 Customizing the X session (new method)

Here are new methods to customize the X session without completely overriding the system code as above.

• The display manager gdm3 can select a specific session and set it as the argument of "/etc/X11/Xsession".

   – "/etc/profile", "~/.profile", "/etc/xprofile", and "~/.xprofile" files are executed as a part of gdm3
     start up process.

• The "~/.xsessionrc" file is executed as a part of start up process. (desktop independent)

   – "#allow-user-xsession" in "/etc/X11/Xsession.options" does not restrict execution of the "~/.xsessionrc"
     file.

• The "~/.gnomerc" file is executed as a part of start up process. (GNOME desktop only)

The selection of the display (or login) manager (DM), session manager or window manager (WM) by the system utility is re-
spected.

These configuration files should not have "exec …" nor "exit" in them.

7.5.4 Connecting a remote X client via SSH

The use of "ssh -X" enables a secure connection from a local X server to a remote application server.

Set "X11Forwarding" entries to "yes" in "/etc/ssh/sshd_config" of the remote host, if you want to avoid "-X"
command-line option.

Start the X server on the local host.

Open an xterm in the local host.

Run ssh(1) to establish a connection with the remote site as the following.

```
localname @ localhost $ ssh -q -X loginname@remotehost.domain
Password:
```

Run an X application command, e.g. "gimp", on the remote site as the following.

```
loginname @ remotehost $ gimp &
```

This method can display the output from a remote X client as if it were locally connected through a local UNIX domain socket.

7.5.5 Secure X terminal via the Internet

Secure X terminal via the Internet, which displays remotely run entire X desktop environment, can easily achieved by using specialized package such as ldm. Your local machine becomes a secure thin client to the remote application server connected via SSH.

7.6   Fonts in the X Window

Fontconfig 2.0 was created to provide a distribution independent library for configuring and customizing font access in 2002. Debian after squeeze uses Fontconfig 2.0 for its font configuration.

Font supports on X Window System can be summarized as follows.

* Legacy X server side font support system

  – The original core X11 font system provides backward compatibility for older version of X client applications.
  – The original core X11 fonts are installed on the X server.

* Modern X client side font support system

  – The modern X system supports all fonts listed below (Section 7.6.1, Section 7.6.2, and Section 7.6.3) with advanced features such as anti-aliasing.
  – Xft 2.0 connects modern X applications such as ones from GNOME, KDE, and LibreOffice with FreeType 2.0 library.
  – FreeType 2.0 provides font rasterization library.
  – Fontconfig provides resolution of the font specification for Xft 2.0. See fonts.conf(5) for its configuration.
  – All modern X applications using Xft 2.0 can talk to modern X server using the X Rendering Extension.
  – The X Rendering Extension moves font access and glyph image generation from the X server to the X client.

package	popcon	size	description
xfonts-utils	V:39, I:613	387	X Window System font utility programs
libxft2	V:121, I:733	122	Xft, a library that connects X applications with the FreeType font rasterization library
libfreetype6	V:471, I:994	920	FreeType 2.0 font rasterization library
fontconfig	V:404, I:857	563	Fontconfig, a generic font configuration library —support binaries
fontconfig-config	V:414, I:916	472	Fontconfig, a generic font configuration library —configuration data

Table 7.4: Table of packages to support X Window font systems

You can check font configuration information by the following.

* "xset q" for core X11 font path

- ”`fc-match`” for fontconfig font default

- ”`fc-list`” for available fontconfig fonts

Tip
”The Penguin and Unicode” is a good overview of modern X Window System. Other documentations at http://unifont.org/ should provide good information on Unicode fonts, Unicode-enabled software, internationalization, and Unicode usability issues on free/libre/open source (FLOSS) operating systems.

7.6.1 Basic fonts

There are 2 major types of computer fonts.

- Bitmap fonts (good for low resolution rasterization)

- Outline/stroke fonts (good for high resolution rasterization)

While scaling of bitmap fonts causes jugged image, scaling of outline/stroke fonts produces smooth image.

Bitmap fonts on the Debian system are usually provided by compressed X11 pcf bitmap font files having their file extension ”`.pcf.gz`”.

Outline fonts on the Debian system are provided by the following.

- PostScript Type 1 font files having their file extension ”`.pfb`” (binary font file) and ”`.afm`” (font metrics file).

- TrueType (or OpenType) font files usually having their file extension ”`.ttf`”.

Tip
OpenType is intended to supersede both TrueType and PostScript Type 1.

---

font package	popcon	size	sans-serif font	serif font	monospace font	source of font
PostScript	N/A	N/A	Helvetica	Times	Courier	Adobe
gsfonts	I:694	4439	Nimbus Sans L	Nimbus Roman No9 L	Nimbus Mono L	URW (Adobe compatible size)
gsfonts-x11	I:120	96	Nimbus Sans L	Nimbus Roman No9 L	Nimbus Mono L	X font support with PostScript Type 1 fonts.
t1-cyrillic	I:24	4884	Free Helvetian	Free Times	Free Courier	URW extended (Adobe compatible size)
lmodern	I:138	33270	LMSans*	LMRoman*	LMTypewriter*	scalable PostScript and OpenType fonts based on Computer Modern (from TeX)

Table 7.5: Table of corresponding PostScript Type 1 fonts

Tip
DejaVu fonts are based on and superset of Bitstream Vera fonts.

---

font package	popcon	size	sans-serif font	serif font	monospace font	source of font
fonts-liberation	I:574	2093	Liberation Sans	Liberation Serif	Liberation Mono	Liberation Fonts project (Microsoft compatible size)
fonts-freefont-ttf	V:45, I:363	10746	FreeSans	FreeSerif	FreeMono	GNU freefont (Microsoft compatible size)
fonts-dejavu	I:514	39	DejaVu Sans	DejaVu Serif	DejaVu Sans Mono	DejaVu, Bitstream Vera with Unicode coverage
fonts-dejavu-core	V:321, I:768	2954	DejaVu Sans	DejaVu Serif	DejaVu Sans Mono	DejaVu, Bitstream Vera with Unicode coverage (sans, sans-bold, serif, serif-bold, mono, mono-bold)
fonts-dejavu-extra	I:539	7217	N/A	N/A	N/A	DejaVu, Bitstream Vera with Unicode coverage (oblique, italic, bold-oblique, bold-italic, condensed)
ttf-unifont	I:22	16943	N/A	N/A	unifont	GNU Unifont, with all printable character code in Unicode 5.1 Basic Multilingual Plane (BMP)

Table 7.6: Table of corresponding TrueType fonts

7.6.2 Additional fonts

aptitude(8) helps you find additional fonts easily.

- The short package list under "Tasks" → "Localization"

- The filtered flat package list of font data with regex on debtag: "~Gmade-of::data:font"

- The filtered flat package list of the BDF (bitmap) font packages with regex on package name: "~nxfonts-"

- The filtered flat package list of the TrueType (outline) font packages with regex on package name: "~nttf-|~nfonts-"

7.6.3 CJK fonts

Here are some key points focused on fonts of CJK characters.

font type	Japanese font name	Chinese font name	Korean font name
sans-serif	gothic, ゴチック	hei, gothic	dodum, gulim, gothic
serif	mincho, 明朝	song, ming	batang

Table 7.7: Table of key words used in CJK font names to indicate font types

Font name such as "VL PGothic" with "P" is a proportional font which corresponds to the fixed width "VL Gothic" font.

For example, Shift_JIS code table comprises 7070 characters. They can be grouped as the following.

- JIS X 0201 single-byte characters (191 characters, a.k.a. half-width characters)

- JIS X 0208 double-byte characters (6879 characters, a.k.a. full-width characters)

Double-byte characters occupy double width on console terminals which uses CJK fixed width fonts. In order to cope with such situation, Hanzi Bitmap Font (HBF) File with file extension ".hbf" may be deployed for fonts containing single-byte and double-byte characters.

In order to save space for TrueType font files, TrueType font collection file with file extension ".ttc" may be used.

In order to cover complicated code space of characters, CID keyed PostScript Type 1 font is used with CMap files starting themselves with "%!PS-Adobe-3.0 Resource-CMap". This is rarely used for normal X display but used for PDF rendering etc. (see Section 7.7.2).

Tip
The multiple glyphs are expected for some Unicode code points due to Han unification. One of the most annoying ones are "U+3001 IDEOGRAPHIC COMMA" and "U+3002 IDEOGRAPHIC FULL STOP" whose character positions differ among CJK countries. Configuring priority of Japanese centric fonts over Chinese ones using "~/.fonts.conf" should give peace of minds to Japanese.

7.7 X applications

7.7.1 X office applications

Here is a list of basic office applications (LO is LibreOffice).

package	popcon	package size	type	description
libreoffice-writer	V:314, I:490	31652	LO	word processor
libreoffice-calc	V:309, I:484	28663	LO	spreadsheet
libreoffice-impress	V:307, I:481	3802	LO	presentation
libreoffice-base	V:296, I:462	8928	LO	database management
libreoffice-draw	V:307, I:481	9881	LO	vector graphics editor (draw)
libreoffice-math	V:310, I:487	1625	LO	mathematical equation/formula editor
abiword	V:4, I:16	4748	GNOME	word processor
gnumeric	V:15, I:24	7888	GNOME	spreadsheet
gimp	V:84, I:506	16413	GTK	bitmap graphics editor (paint)
inkscape	V:153, I:346	130049	GNOME	vector graphics editor (draw)
dia	V:21, I:38	3916	GTK	flowchart and diagram editor
planner	V:3, I:7	1170	GNOME	project management
calligrawords	V:1, I:8	5569	KDE	word processor
calligrasheets	V:1, I:7	10780	KDE	spreadsheet
calligrastage	V:0, I:7	4832	KDE	presentation
calligraplan	V:0, I:7	15222	KDE	project management
kexi	V:0, I:7	7475	KDE	database management
karbon	V:1, I:8	4186	KDE	vector graphics editor (draw)

Table 7.8: List of basic X office applications

7.7.2 X utility applications

Here is a list of basic utility applications which caught my eyes.

package	popcon	package size	type	description
evince	V:179, I:409	1401	GNOME	document(pdf) viewer
okular	V:72, I:123	13100	KDE	document(pdf) viewer
calibre	V:7, I:38	51018	KDE	e-book converter and library management
fbreader	V:2, I:17	3035	GTK	e-book reader
evolution	V:42, I:284	441	GNOME	Personal information Management (groupware and email)
kontact	V:2, I:22	2030	KDE	Personal information Management (groupware and email)
scribus	V:5, I:27	19061	KDE	desktop page layout editor
glabels	V:0, I:4	1389	GNOME	label editor
gnucash	V:4, I:14	9667	GNOME	personal accounting
homebank	V:0, I:3	878	GTK	personal accounting
kmymoney	V:0, I:3	12342	KDE	personal accounting
shotwell	V:20, I:207	6635	GTK	digital photo organizer
xsane	V:21, I:195	909	GTK	scanner frontend

Table 7.9: List of basic X utility applications

> **Caution**
> The poppler-data package (see Section 11.4.1) needs to be installed for evince and okular to display CJK PDF documents using Cmap data (Section 7.6.3).

> **Note**
> Installing softwares such as scribus (KDE) on GNOME desktop environment are quite acceptable since corresponding functionality is not available under GNOME desktop environment. But installing too many packages with duplicated functionalities clutter your menu.

7.8 The X trivia

7.8.1 Clipboard

The X selection using 3 mouse buttons is the native clipboard feature of X (see Section 1.4.4).

> **Tip**
> Shift-Insert can work as the equivalent of the middle-mouse-button click.

The modern Desktop Environments (GNOME, KDE, ···) offer different clipboard system for the cut, copy, and paste using the left mouse button and keys (CTRL-X, CRTL-C, and CTRL-V).

7.8.2 Keymaps and pointer button mappings in X

xmodmap(1) is a utility for modifying keymaps and pointer button mappings in the X Window System. To get the **keycode**, run xev(1) in the X and press keys. To get the meaning of **keysym**, look into the MACRO definition in "/usr/include/X11/keysymdef file (x11proto-core-dev package). All "#define" statements in this file are named as "XK_" prepended to **keysym** names.

package	popcon	package size	type	description
xsel	V:11, I:41	51	X	command line interface to X selections
xclip	V:8, I:37	55	X	command line interface to X selections

Table 7.10: List of basic X selection programs

7.8.3 Classic X clients

Most traditional X client programs, such as xterm(1), can be started with a set of standard command line options to specify geometry, font, and display.

They also use the X resource database to configure their appearance. The system-wide defaults of X resources are stored in "/etc/X11/Xresources/*" and application defaults of them are stored in "/etc/X11/app-defaults/*". Use these settings as the starting points.

The "~/.Xresources" file is used to store user resource specifications. This file is automatically merged into the default X resources upon login. To make changes to these settings and make them effective immediately, merge them into the database using the following command.

```
$ xrdb -merge ~/.Xresources
```

See x(7) and xrdb(1).

7.8.4 The X terminal emulator —xterm

Learn everything about xterm(1) at http://dickey.his.com/xterm/xterm.faq.html.

7.8.5 Running X clients as root

> **Warning**
> Never start the X display/session manager under the root account by typing in root to the prompt of the display manager such as gdm3 because it is considered unsafe (insecure), even when you plan to perform administrative activities. The entire X architecture is considered insecure if run as root. You must always use the lowest privilege level possible, like a regular user account.

Easy ways to run a particular X client, e.g. "foo" as root is to use sudo(8) etc. as the following.

```
$ sudo foo &
```

```
$ sudo -s
# foo &
```

```
$ gksu foo &
```

```
$ ssh -X root@localhost
# foo &
```

> **Caution**
> Use of ssh(1) just for this purpose as above is waste of resource.

In order for the X client to connect to the X server, please note the following.

- Values of the old user's "$XAUTHORITY" and "$DISPLAY" environment variables must be copied to the new user's ones.

- The file pointed by value of the "$XAUTHORITY" environment variable must be readable by the new user.

The gksu package (popcon: V:38, I:253) is a specialized GTK+ GUI package for gaining the root privileges. It can be configured to use su(1) or sudo(8) as its backend depending on the "/apps/gksu/sudo-mode" gconf key. You can edit gconf key using gconf-editor(1) (menu: "Applications" → "System Tools" → "Configuration Editor").

Chapter 8

I18N and L10N

Multilingualization (M17N) or Native Language Support for an application software is done in 2 steps.

- Internationalization (I18N): To make a software potentially handle multiple locales.

- Localization (L10N): To make a software handle an specific locale.

Tip
There are 17, 18, or 10 letters between "m" and "n", "i" and "n", or "l" and "n" in multilingualization, internationalization, and localization which correspond to M17N, I18N, and L10N.

The modern software such as GNOME and KDE are multilingualized. They are internationalized by making them handle UTF-8 data and localized by providing their translated messages through the gettext(1) infrastructure. Translated messages may be provided as separate localization packages. They can be selected simply by setting pertinent environment variables to the appropriate locale.

The simplest representation of the text data is **ASCII** which is sufficient for English and uses less than 127 characters (representable with 7 bits). In order to support much more characters for the international support, many character encoding systems have been invented. The modern and sensible encoding system is **UTF-8** which can handle practically all the characters known to the human (see Section 8.4.1).

See Introduction to i18n for details.

The international hardware support is enabled with localized hardware configuration data.

⚠ **Warning**
This chapter is getting outdated since this is based on Debian 7.0 (`Wheezy`) released in 2013.

8.1   The keyboard input

The Debian system can be configured to work with many international keyboard arrangements using the `keyboard-configuration` and `console-setup` packages.

```
# dpkg-reconfigure keyboard-configuration
# dpkg-reconfigure console-setup
```

This configures the keyboard for the Linux console and the X Window updates configuration parameters in "`/etc/default/keyboard`" and "`/etc/default/console-setup`". This also configures the Linux console font.

Many non-ASCII characters including accented characters used by many European languages can be made available with dead key, AltGr key, and compose key.

For Asian languages, you need more complicated input method support such as IBus discussed next.

8.1.1 The input method support with IBus

Multilingual input to the application is processed as:

```
Keyboard                           Application
   |                                   ^
   |                                   |
   +-> Linux kernel -> Input method -> Gtk, Qt, or X
```
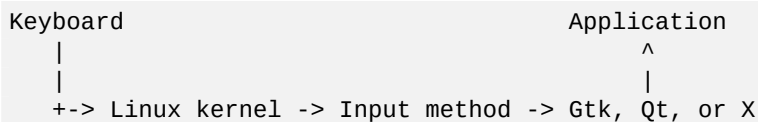
Setup of multilingual input for the Debian system is simplified by using the IBus family of packages with the `im-config` package. The list of IBus packages are the following.

package	popcon	size	supported locale
ibus	V:6, I:9	42508	input method framework using dbus
ibus-mozc	V:1, I:1	927	Japanese
ibus-anthy	V:0, I:1	8450	, ,
ibus-kkc	V:0, I:0	205	, ,
ibus-skk	V:0, I:0	234	, ,
ibus-pinyin	V:0, I:2	1425	Chinese (for zh_CN)
ibus-chewing	V:0, I:0	401	, , (for zh_TW)
ibus-hangul	V:0, I:1	206	Korean
ibus-table	V:0, I:1	908	table engine for IBus
ibus-table-thai	I:0	45	Thai
ibus-unikey	V:0, I:0	256	Vietnamese
ibus-m17n	V:0, I:0	121	Multilingual: Indic, Arabic and others

Table 8.1: List of input method supports with IBus

The kinput2 method and other locale dependent Asian classic input methods still exist but are not recommended for the modern UTF-8 X environment. The SCIM and uim tool chains are an slightly older approach for the international input method for the modern UTF-8 X environment.

8.1.2 An example for Japanese

I find the Japanese input method started under English environment ("`en_US.UTF-8`") very useful. Here is how I did this with IBus for GNOME3:

1. Install the Japanese input tool package `ibus-anthy` with its recommended packages such as `im-config`.

2. Execute "`im-config`" from user's shell and select "`ibus`" as the input method.

3. Select "Settings" → "Keyboard" → "Input Sources" → click "+" in "Input Sources" → "Japanese" → "Japanese (anthy)" and click "Add".

4. Select "Japanese" and click "Add" to support the Japanese layout keyboard without character conversion. (You may chose as many input sources.)

5. Relogin to user's account.

6. Verify setting by "`im-config`".

7. Setup input source by right clicking the GUI toolbar icon.

8. Switch among installed input sources by SUPER-SPACE. (SUPER is normally the Windows key.)

Please note the following.

- im-config(8) behaves differently if command is executed from root or not.

- im-config(8) enables the best input method on the system as default without any user actions.

- The GUI menu entry for im-config(8) is disabled as default to prevent cluttering.

8.1.3 Disabling the input method

If you wish to input without going through XIM (mechanism used by the X), set "`$XMODIFIERS`" value to "none" while starting a program. This may be the case if you use Japanese input infrastructure `egg` on emacs(1) while disabling `ibus`. From shell, execute as the following.

```
$ XMODIFIERS=none emacs
```

In order to adjust the command executed by the Debian menu, place customized configuration in "`/etc/menu/`" following method described in "`/usr/share/doc/menu/html`".

8.2 The display output

Linux console can only display limited characters. (You need to use special terminal program such as jfbterm(1) to display non-European languages on the non-X console.)

X Window can display any characters in the UTF-8 as long as required font data exists. (The encoding of the original font data is taken care by the X Window System and transparent to the user.)

8.3 East Asian Ambiguous Character Width Characters

Under the East Asian locale, the box drawing, Greek, and Cyrillic characters may be displayed wider than your desired width to cause the unaligned terminal output (see Unicode Standard Annex #11).

You can work around this problem:

- `gnome-terminal`: Edit → Preferences → Profiles → Edit → Compatibility → Ambiguous-wide characters → Narrow

- `ncurses`: Set environment `export NCURSES_NO_UTF8_ACS=0`.

8.4 The locale

The following focuses on the locale for applications run under X Window environment started from gdm3(1).

8.4.1 Basics of encoding

The environment variable "LANG=xx_YY.ZZZZ" sets the locale to language code "xx", country code "yy", and encoding "ZZZZ" (see Section 1.5.2).

The current Debian system normally sets the locale as "LANG=xx_YY.UTF-8". This uses the UTF-8 encoding with the Unicode character set. This UTF-8 encoding system is a multibyte code system and uses code points smartly. The ASCII data, which consist only with 7-bit range codes, are always valid UTF-8 data consisting only with 1 byte per character.

The previous Debian system used to set the locale as "LANG=C" or "LANG=xx_YY" (without ".UTF-8").

- The ASCII character set is used for "LANG=C" or "LANG=POSIX".

- The traditional encoding system in Unix is used for "LANG=xx_YY".

Actual traditional encoding system used for "LANG=xx_YY" can be identified by checking "/usr/share/i18n/SUPPORTED". For example, "en_US" uses "ISO-8859-1" encoding and "fr_FR@euro" uses "ISO-8859-15" encoding.

Tip
For meaning of encoding values, see Table 11.2.

8.4.2 Rationale for UTF-8 locale

Unicode character set can represent practically all characters known to human with code point range from 0 to 10FFFF in hexadecimal notation. Its storage requires at least 21 bits.

Text encoding system UTF-8 fits Unicode code points into a sensible 8 bit data stream compatible with the ASCII data processing system. **UTF** stands for Unicode Transformation Format.

I recommend to use UTF-8 locale for your desktop, e.g., "LANG=en_US.UTF-8". The first part of the locale determines messages presented by applications. For example, gedit(1) (text editor for the GNOME Desktop) under "LANG=fr_FR.UTF-8" locale can display and edit Chinese character text data while presenting menus in French, as long as required fonts and input methods are installed.

I also recommend to set the locale only using the "$LANG" environment variable. I do not see much benefit of setting a complicated combination of "LC_*" variables (see locale(1)) under UTF-8 locale.

Even plain English text may contain non-ASCII characters, e.g. slightly curly left and right quotation marks are not available in ASCII.

```
"double quoted text"  is not "double quoted ASCII"
'single quoted text'  is not 'single quoted ASCII'
```

When ASCII plain text data is converted to UTF-8 one, it has exactly the same content and size as the original ASCII one. So you loose nothing by deploying UTF-8 locale.

Some programs consume more memory after supporting I18N. This is because they are coded to use UTF-32(UCS4) internally to support Unicode for speed optimization and consume 4 bytes per each ASCII character data independent of locale selected. Again, you loose nothing by deploying UTF-8 locale.

The vendor specific old non-UTF-8 encoding systems tend to have minor but annoying differences on some characters such as graphic ones for many countries. The deployment of the UTF-8 system by the modern OSs practically solved these conflicting encoding issues.

8.4.3 The reconfiguration of the locale

In order for the system to access a particular locale, the locale data must be compiled from the locale database. (The Debian system does **not** come with all available locales pre-compiled unless you installed the locales-all package.) The full list of supported locales available for compiling is available in "/usr/share/i18n/SUPPORTED". This lists all the proper locale names. The following lists all the available UTF-8 locales already compiled to the binary form.

```
$ locale -a | grep utf8
```

The following command execution reconfigures the `locales` package.

```
# dpkg-reconfigure locales
```

This process involves 3 steps.

1. Update the list of available locales

2. Compile them into the binary form

3. Set the system wide default locale value in "`/etc/default/locale`" for use by PAM (see Section 4.5)

The list of available locale should include "`en_US.UTF-8`" and all the interesting languages with "`UTF-8`".

The recommended default locale is "`en_US.UTF-8`" for US English. For other languages, please make sure to chose locale with "`UTF-8`". Any one of these settings can handle any international characters.

Note
Although setting locale to "C" uses US English message, it handles only ASCII characters.

8.4.4   The value of the "$LANG" environment variable

The value of the "`$LANG`" environment variable is set and changed by many applications.

- Set initially by the PAM mechanism of login(1) for the local Linux console programs

- Set initially by the PAM mechanism of the display manager for all X programs

- Set initially by the PAM mechanism of ssh(1) for the remote console programs

- Changed by some display manager such as gdm3(1) for all X programs

- Changed by the X session startup code via "`~/.xsessionrc`" for all X programs

- Changed by the shell startup code, e.g. "`~/.bashrc`", for all console programs

---

Tip
It is a good idea to install system wide default locale as "en_US.UTF-8" for maximum compatibility.

8.4.5   Specific locale only under X Window

You can chose specific locale only under X Window irrespective of your system wide default locale using PAM customization (see Section 4.5) as follows.

This environment should provide you with your best desktop experience with stability. You have access to the functioning character terminal with readable messages even when the X Window System is not working. This becomes essential for languages which use non-roman characters such as Chinese, Japanese, and Korean.

Note
There may be another way available as the improvement of X session manager package but please read following as the generic and basic method of setting the locale. For gdm3(1), I know you can select the locale of X session via its memu.

The following line defines file location of the language environment in the PAM configuration file, such as "`/etc/pam.d/gdm3`.

```
auth    required        pam_env.so read_env=1 envfile=/etc/default/locale
```

Change this to the following.

```
auth    required        pam_env.so read_env=1 envfile=/etc/default/locale-x
```

For Japanese, create a "`/etc/default/locale-x`" file with "`-rw-r--r-- 1 root root`" permission containing the following.

```
LANG="ja_JP.UTF-8"
```

Keep the default "`/etc/default/locale`" file for other programs as the the following.

```
LANG="en_US.UTF-8"
```

This is the most generic technique to customize locale and makes the menu selection dialog of gdm3(1) itself to be localized.

Alternatively for this case, you may simply change locale using the "`~/.xsessionrc`" file.

8.4.6   Filename encoding

For cross platform data exchanges (see Section 10.1.7), you may need to mount some filesystem with particular encodings. For example, mount(8) for vfat filesystem assumes CP437 if used without option. You need to provide explicit mount option to use UTF-8 or CP932 for filenames.

Note
When auto-mounting a hot-pluggable USB memory stick under modern desktop environment such as GNOME, you may provide such mount option by right clicking the icon on the desktop, click "Drive" tab, click to expand "Setting", and entering "utf8" to "Mount options:". The next time this memory stick is mounted, mount with UTF-8 is enabled.

Note
If you are upgrading system or moving disk drives from older non-UTF-8 system, file names with non-ASCII characters may be encoded in the historic and deprecated encodings such as ISO-8859-1 or eucJP. Please seek help of text conversion tools to convert them to UTF-8. See Section 11.1.

Samba uses Unicode for newer clients (Windows NT, 200x, XP) but uses CP850 for older clients (DOS and Windows 9x/Me) as default. This default for older clients can be changed using "`dos charset`" in the "`/etc/samba/smb.conf`" file, e.g., to CP932 for Japanese.

8.4.7   Localized messages and translated documentation

Translations exist for many of the text messages and documents that are displayed in the Debian system, such as error messages, standard program output, menus, and manual pages. GNU gettext(1) command tool chain is used as the backend tool for most translation activities.

Under "Tasks" → "Localization" aptitude(8) provides an extensive list of useful binary packages which add localized messages to applications and provide translated documentation.

For example, you can obtain the localized message for manpage by installing the `manpages-<LANG>` package. To read the Italian-language manpage for <programname> from "`/usr/share/man/it/`", execute as the following.

```
LANG=it_IT.UTF-8 man <programname>
```

8.4.8 Effects of the locale

The sort order of characters with sort(1) is affected by the language choice of the locale. Spanish and English locale sort differently.

The date format of ls(1) is affected by the locale. The date format of "`LANG=C ls -l`" and "`LANG=en_US.UTF-8`" are different (see Section 9.2.5).

Number punctuation are different for locales. For example, in English locale, one thousand one point one is displayed as "`1,000.1`" while in German locale, it is displayed as "`1.000,1`". You may see this difference in spreadsheet program.

Chapter 9

System tips

Here, I describe basic tips to configure and manage systems, mostly from the console.

9.1 The screen program

screen(1) is a very useful tool for people to access remote sites via unreliable or intermittent connections since it supports interrupted network connections.

package	popcon	size	description
screen	V:185, I:284	987	terminal multiplexer with VT100/ANSI terminal emulation
tmux	V:29, I:107	659	terminal multiplexer alternative (Use "Control-B" instead)

Table 9.1: List of programs to support interrupted network connections

9.1.1 The use scenario for screen(1)

screen(1) not only allows one terminal window to work with multiple processes, but also allows **remote shell process to survive interrupted connections**. Here is a typical use scenario of screen(1).

1. You login to a remote machine.

2. You start screen on a single console.

3. You execute multiple programs in screen windows created with ˆA c ("Control-A" followed by "c").

4. You switch among the multiple screen windows by ˆA n ("Control-A" followed by "n").

5. Suddenly you need to leave your terminal, but you don't want to lose your active work by keeping the connection.

6. You may **detach** the screen session by any methods.

   - Brutally unplug your network connection
   - Type ˆA d ("Control-A" followed by "d") and manually logging out from the remote connection
   - Type ˆA DD ("Control-A" followed by "DD") to have screen detach and log you out

7. You log in again to the same remote machine (even from a different terminal).

8. You start screen as "screen -r".

9. `screen` magically **reattaches** all previous `screen` windows with all actively running programs.

Tip

You can save connection fees with `screen` for metered network connections such as dial-up and packet ones, because you can leave a process active while disconnected, and then re-attach it later when you connect again.

9.1.2 Key bindings for the screen command

In a `screen` session, all keyboard inputs are sent to your current window except for the command keystroke. All `screen` command keystrokes are entered by typing ˆA ("Control-A") plus a single key [plus any parameters]. Here are important ones to remember.

key binding	meaning
ˆA ?	show a help screen (display key bindings)
ˆA c	create a new window and switch to it
ˆA n	go to next window
ˆA p	go to previous window
ˆA 0	go to window number 0
ˆA 1	go to window number 1
ˆA w	show a list of windows
ˆA a	send a Ctrl-A to current window as keyboard input
ˆA h	write a hardcopy of current window to file
ˆA H	begin/end logging current window to file
ˆA ˆX	lock the terminal (password protected)
ˆA d	detach screen session from the terminal
ˆA DD	detach screen session and log out

Table 9.2: List of key bindings for screen

See screen(1) for details.

9.2 Data recording and presentation

9.2.1 The log daemon

Many programs record their activities under the "`/var/log/`" directory.

• The system log daemon: rsyslogd(8)

See Section 3.2.5 and Section 3.2.4.

9.2.2 Log analyzer

Here are notable log analyzers ("`~Gsecurity::log-analyzer`" in aptitude(8)).

Note

CRM114 provides language infrastructure to write **fuzzy** filters with the TRE regex library. Its popular use is spam mail filter but it can be used as log analyzer.

---

package	popcon	size	description
logwatch	V:18, I:20	2214	log analyzer with nice output written in Perl
fail2ban	V:97, I:108	1731	ban IPs that cause multiple authentication errors
analog	V:5, I:125	3529	web server log analyzer
awstats	V:11, I:18	6799	powerful and featureful web server log analyzer
sarg	V:5, I:5	429	squid analysis report generator
pflogsumm	V:1, I:4	111	Postfix log entry summarizer
syslog-summary	V:1, I:4	30	summarize the contents of a syslog log file
fwlogwatch	V:0, I:0	474	firewall log analyzer
squidview	V:0, I:1	192	monitor and analyze squid access.log files
swatch	V:0, I:0	100	log file viewer with regexp matching, highlighting, and hooks
crm114	V:0, I:0	1099	Controllable Regex Mutilator and Spam Filter (CRM114)
icmpinfo	V:0, I:0	39	interpret ICMP messages

Table 9.3: List of system log analyzers

9.2.3 Recording the shell activities cleanly

The simple use of script(1) (see Section 1.4.9) to record shell activity produces a file with control characters. This can be avoided by using col(1) as the following.

```
$ script
Script started, file is typescript
```

Do whatever ⋯and press Ctrl-D to exit script.

```
$ col -bx <typescript >cleanedfile
$ vim cleanedfile
```

If you don't have script (for example, during the boot process in the initramfs), you can use the following instead.

```
$ sh -i 2>&1 | tee typescript
```

Tip
Some x-terminal-emulator such as gnome-terminal can record. You may wish to extend line buffer for scrollback.

Tip
You may use screen(1) with "ˆA H" (see Section 9.1.2) to perform recording of console.

Tip
You may use emacs(1) with "M-x shell", "M-x eshell", or "M-x term" to perform recording of console. You may later use "C-x C-w" to write the buffer to a file.

9.2.4 Customized display of text data

Although pager tools such as more(1) and less(1) (see Section 1.4.5) and custom tools for highlighting and formatting (see Section 11.1.8) can display text data nicely, general purpose editors (see Section 1.4.6) are most versatile and customizable.

Tip
For vim(1) and its pager mode alias view(1), ":set hls" enables highlighted search.

9.2.5   Customized display of time and date

The default display format of time and date by the "ls -l" command depends on the **locale** (see Section 1.2.6 for value). The "$LANG" variable is referred first and it can be overridden by the "$LC_TIME" variable.

The actual default display format for each locale depends on the version of the standard C library (the libc6 package) used. I.e., different releases of Debian had different defaults.

If you really wish to customize this display format of time and date beyond the **locale**, you should set the **time style value** by the "--time-style" argument or by the "$TIME_STYLE" value (see ls(1), date(1), "info coreutils 'ls invocation'").

time style value	locale	display of time and date
iso	any	01-19 00:15
long-iso	any	2009-01-19 00:15
full-iso	any	2009-01-19 00:15:16.000000000 +0900
locale	C	Jan 19 00:15
locale	en_US.UTF-8	Jan 19 00:15
locale	es_ES.UTF-8	ene 19 00:15
+%d.%m.%y %H:%M	any	19.01.09 00:15
+%d.%b.%y %H:%M	C or en_US.UTF-8	19.Jan.09 00:15
+%d.%b.%y %H:%M	es_ES.UTF-8	19.ene.09 00:15

Table 9.4: Display examples of time and date for the "ls -l" command for wheezy

---

Tip
You can eliminate typing long option on commandline using command alias, e.g.   "alias ls='ls --time-style=+%d.%m.%y\ %H:%M'" (see Section 1.5.9).

Tip
ISO 8601 is followed for these iso-formats.

9.2.6   Colorized shell echo

Shell echo to most modern terminals can be colorized using ANSI escape code (see "/usr/share/doc/xterm/ctlseqs.txt.gz"

For example, try the following

```
$ RED=$(printf "\x1b[31m")
$ NORMAL=$(printf "\x1b[0m")
$ REVERSE=$(printf "\x1b[7m")
$ echo "${RED}RED-TEXT${NORMAL} ${REVERSE}REVERSE-TEXT${NORMAL}"
```

9.2.7   Colorized commands

Colorized commands are handy for inspecting their output in the interactive environment. I include the following in my "~/.bashrc".

```
if [ "$TERM" != "dumb" ]; then
    eval "'dircolors -b'"
    alias ls='ls --color=always'
    alias ll='ls --color=always -l'
    alias la='ls --color=always -A'
```

```
    alias less='less -R'
    alias ls='ls --color=always'
    alias grep='grep --color=always'
    alias egrep='egrep --color=always'
    alias fgrep='fgrep --color=always'
    alias zgrep='zgrep --color=always'
else
    alias ll='ls -l'
    alias la='ls -A'
fi
```

The use of alias limits color effects to the interactive command usage. It has advantage over exporting environment variable "export GREP_OPTIONS='--color=auto'" since color can be seen under pager programs such as less(1). If you wish to suppress color when piping to other programs, use "--color=auto" instead in the above example for "~/.bashrc".

Tip
You can turn off these colorizing aliases in the interactive environment by invoking shell with "TERM=dumb bash".

9.2.8   Recording the editor activities for complex repeats

You can record the editor activities for complex repeats.

For Vim, as follows.

- "qa": start recording typed characters into named register "a".

- ···editor activities

- "q": end recording typed characters.

- "@a": execute the contents of register "a".

For Emacs, as follows.

- "C-x (": start defining a keyboard macro.

- ···editor activities

- "C-x)": end defining a keyboard macro.

- "C-x e": execute a keyboard macro.

9.2.9   Recording the graphic image of an X application

There are few ways to record the graphic image of an X application, including an xterm display.

package	popcon	size	command
xbase-clients	I:49	46	xwd(1)
gimp	V:84, I:506	16413	GUI menu
imagemagick	V:64, I:560	205	import(1)
scrot	V:9, I:92	50	scrot(1)

Table 9.5: List of graphic image manipulation tools

package	popcon	size	description
etckeeper	V:25, I:29	153	store configuration files and their metadata with Git (default), Mercurial, or Bazaar (new)
changetrack	V:0, I:0	62	store configuration files with RCS (old)

Table 9.6: List of packages to record configuration history in VCS

9.2.10 Recording changes in configuration files

There are specialized tools to record changes in configuration files with help of DVCS system.

I recommend to use the etckeeper package with git(1) which put entire "/etc" under VCS control. Its installation guide and tutorial are found in "/usr/share/doc/etckeeper/README.gz".

Essentially, running "sudo etckeeper init" initializes the git repository for "/etc" just like the process explained in Section 10.6.5 but with special hook scripts for more thorough setups.

As you change your configuration, you can use git(1) normally to record them. It automatically records changes nicely every time you run package management commands, too.

Tip
You can browse the change history of "/etc" by executing "sudo GIT_DIR=/etc/.git gitk" with clear view for new installed packages, removed packages, and version changes of packages.

9.3 Monitoring, controlling, and starting program activities

Program activities can be monitored and controlled using specialized tools.

package	popcon	size	description
coreutils	V:878, I:999	15016	nice(1): run a program with modified scheduling priority
bsdutils	V:851, I:999	256	renice(1): modify the scheduling priority of a running process
procps	V:813, I:999	695	"/proc" filesystem utilities: ps(1), top(1), kill(1), watch(1), ⋯
psmisc	V:383, I:946	526	"/proc" filesystem utilities: killall(1), fuser(1), peekfd(1), pstree(1)
time	V:28, I:563	82	time(1): run a program to report system resource usages with respect to time
sysstat	V:126, I:146	1454	sar(1), iostat(1), mpstat(1), ⋯: system performance tools for Linux
isag	V:0, I:4	109	Interactive System Activity Grapher for sysstat
lsof	V:193, I:947	440	lsof(8): list files opened by a running process using "-p" option
strace	V:19, I:159	1399	strace(1): trace system calls and signals
ltrace	V:1, I:21	360	ltrace(1): trace library calls
xtrace	V:0, I:1	333	xtrace(1): trace communication between X11 client and server
powertop	V:6, I:236	612	powertop(1): information about system power use
cron	V:875, I:997	251	run processes according to a schedule in background from cron(8) daemon
anacron	V:453, I:539	106	cron-like command scheduler for systems that don't run 24 hours a day
at	V:333, I:579	149	at(1) or batch(1): run a job at a specified time or below certain load level

Table 9.7: List of tools for monitoring and controlling program activities

---

Tip
The `procps` packages provide very basics of monitoring, controlling, and starting program activities. You should learn all of them.

9.3.1 Timing a process

Display time used by the process invoked by the command.

```
# time some_command >/dev/null
real    0m0.035s      # time on wall clock (elapsed real time)
user    0m0.000s      # time in user mode
sys     0m0.020s      # time in kernel mode
```

9.3.2 The scheduling priority

A nice value is used to control the scheduling priority for the process.

nice value	scheduling priority
19	lowest priority process (nice)
0	very high priority process for user
-20	very high priority process for root (not-nice)

Table 9.8: List of nice values for the scheduling priority

```
# nice  -19 top                                  # very nice
# nice --20 wodim -v -eject speed=2 dev=0,0 disk.img # very fast
```

Sometimes an extreme nice value does more harm than good to the system. Use this command carefully.

9.3.3 The ps command

The ps(1) command on a Debian system support both BSD and SystemV features and helps to identify the process activity statically.

style	typical command	feature
BSD	`ps aux`	display %CPU %MEM
System V	`ps -efH`	display PPID

Table 9.9: List of ps command styles

For the zombie (defunct) children process, you can kill them by the parent process ID identified in the "`PPID`" field.

The pstree(1) command display a tree of processes.

9.3.4 The top command

top(1) on the Debian system has rich features and helps to identify what process is acting funny dynamically.

It is an interactive full screen program. You can get its usage help press by pressing the "h"-key and terminate it by pressing the "q"-key.

9.3.5 Listing files opened by a process

You can list all files opened by a process with a process ID (PID), e.g. 1, by the following.

```
$ sudo lsof -p 1
```

PID=1 is usually the `init` program.

9.3.6 Tracing program activities

You can trace program activity with strace(1), ltrace(1), or xtrace(1) for system calls and signals, library calls, or communication between X11 client and server.

You can trace system calls of the `ls` command as the following.

```
$ sudo strace ls
```

9.3.7 Identification of processes using files or sockets

You can also identify processes using files by fuser(1), e.g. for "`/var/log/mail.log`" by the following.

```
$ sudo fuser -v /var/log/mail.log
                     USER        PID ACCESS COMMAND
/var/log/mail.log:   root       2946 F.... rsyslogd
```

You see that file "`/var/log/mail.log`" is open for writing by the rsyslogd(8) command.

You can also identify processes using sockets by fuser(1), e.g. for "`smtp/tcp`" by the following.

```
$ sudo fuser -v smtp/tcp
                     USER        PID ACCESS COMMAND
smtp/tcp:            Debian-exim  3379 F.... exim4
```

Now you know your system runs exim4(8) to handle TCP connections to SMTP port (25).

9.3.8 Repeating a command with a constant interval

watch(1) executes a program repeatedly with a constant interval while showing its output in fullscreen.

```
$ watch w
```

This displays who is logged on to the system updated every 2 seconds.

9.3.9 Repeating a command looping over files

There are several ways to repeat a command looping over files matching some condition, e.g. matching glob pattern "`*.ext`".

- Shell for-loop method (see Section 12.1.4):

```
for x in *.ext; do if [ -f "$x"]; then command "$x" ; fi; done
```

- find(1) and xargs(1) combination:

```
find . -type f -maxdepth 1 -name '*.ext' -print0 | xargs -0 -n 1 command
```

- find(1) with "`-exec`" option with a command:

```
find . -type f -maxdepth 1 -name '*.ext' -exec command '{}' \;
```

- find(1) with "`-exec`" option with a short shell script:

```
find . -type f -maxdepth 1 -name '*.ext' -exec sh -c "command '{}' && echo 'successful'" \;
```

The above examples are written to ensure proper handling of funny file names such as ones containing spaces. See Section 10.1.5 for more advance uses of find(1).

9.3.10 Starting a program from GUI

For the command-line interface (CLI), the first program with the matching name found in the directories specified in the `$PATH` environment variable is executed. See Section 1.5.3.

For the graphical user interface (GUI) compliant to the freedesktop.org standards, the `*.desktop` files in the /usr/share/applica directory provide necessary attributes for the GUI menu display of each program. See Section 7.2.2.

For example, the `chromium.desktop` file defines attributes for the "Chromium Web Browser" such as "Name" for the program name, "Exec" for the program execution path and arguments, "Icon" for the icon used, etc. (see the Desktop Entry Specification) as follows:

```
[Desktop Entry]
Version=1.0
Name=Chromium Web Browser
GenericName=Web Browser
Comment=Access the Internet
Comment[fr]=Explorer le Web
Exec=/usr/bin/chromium %U
Terminal=false
X-MultipleArgs=false
Type=Application
Icon=chromium
Categories=Network;WebBrowser;
MimeType=text/html;text/xml;application/xhtml_xml;x-scheme-handler/http;x-scheme-handler/ ←
    https;
StartupWMClass=Chromium
StartupNotify=true
```

This is an oversimplified description. The `*.desktop` files are scanned as follows.

The desktop environment sets `$XDG_DATA_HOME` and `$XDG_DATA_DIR` environment variables. For example, under the GNOME 3:

- `$XDG_DATA_HOME` is unset. (The default value of `$HOME/.local/share` is used.)

- `$XDG_DATA_DIRS` is set to `/usr/share/gnome:/usr/local/share/:/usr/share/`.

So the base directories (see XDG Base Directory Specification) and the `applications` directories are as follows.

- `$HOME/.local/share/` → `$HOME/.local/share/applications/`

- `/usr/share/gnome/` → `/usr/share/gnome/applications/`

- `/usr/local/share/` → `/usr/local/share/applications/`

- `/usr/share/` → `/usr/share/applications/`

The *.desktop files are scanned in these applications directories in this order.

Tip

A user custom GUI menu entry can be created by adding a *.desktop file in the $HOME/.local/share/applications/ directory.

Tip

Similarly, if a *.desktop file is created in the autostart directory under these base directories, the specified program in the *.desktop file is executed automatically when the desktop environment is started. See Desktop Application Autostart Specification.

Tip

Similarly, if a *.desktop file is created in the $HOME/Desktop directory and the Desktop environment is configured to support the desktop icon launcher feature, the specified program in it is executed upon clicking the icon. Please note that the actual name of the $HOME/Desktop directory is locale dependent. See xdg-user-dirs-update(1).

9.3.11 Customizing program to be started

Some programs start another program automatically. Here are check points for customizing this process.

- Application configuration menu:

  - GNOME3 desktop: "Settings" → "System" → "Details" → "Default Applications"
  - KDE desktop: "K" → "Control Center" → "KDE Components" → "Component Chooser"
  - Iceweasel browser: "Edit" → "Preferences" → "Applications"
  - mc(1): "/etc/mc/mc.ext"

- Environment variables such as "$BROWSER", "$EDITOR", "$VISUAL", and "$PAGER" (see eviron(7))

- The update-alternatives(1) system for programs such as "editor", "view", "x-www-browser", "gnome-www-browser", and "www-browser" (see Section 1.4.7)

- the "~/.mailcap" and "/etc/mailcap" file contents which associate MIME type with program (see mailcap(5))

- The "~/.mime.types" and "/etc/mime.types" file contents which associate file name extension with MIME type (see run-mailcap(1))

Tip

update-mime(8) updates the "/etc/mailcap" file using "/etc/mailcap.order" file (see mailcap.order(5)).

Tip

The debianutils package provides sensible-browser(1), sensible-editor(1), and sensible-pager(1) which make sensible decisions on which editor, pager, and web browser to call, respectively. I recommend you to read these shell scripts.

Tip

In order to run a console application such as mutt under X as your preferred application, you should create an X application as following and set "/usr/local/bin/mutt-term" as your preferred application to be started as described.

```
# cat /usr/local/bin/mutt-term <<EOF
#!/bin/sh
gnome-terminal -e "mutt \$@"
EOF
chmod 755 /usr/local/bin/mutt-term
```

9.3.12 Killing a process

Use kill(1) to kill (or send a signal to) a process by the process ID.

Use killall(1) or pkill(1) to do the same by the process command name and other attributes.

signal value	signal name	function
1	HUP	restart daemon
15	TERM	normal kill
9	KILL	kill hard

Table 9.10: List of frequently used signals for kill command

9.3.13 Scheduling tasks once

Run the at(1) command to schedule a one-time job by the following.

```
$ echo 'command -args'| at 3:40 monday
```

9.3.14 Scheduling tasks regularly

Use cron(8) to schedule tasks regularly. See crontab(1) and crontab(5).

You can schedule to run processes as a normal user, e.g. foo by creating a crontab(5) file as "/var/spool/cron/crontabs/foo" with "crontab -e" command.

Here is an example of a crontab(5) file.

```
# use /bin/sh to run commands, no matter what /etc/passwd says
SHELL=/bin/sh
# mail any output to paul, no matter whose crontab this is
MAILTO=paul
# Min Hour DayOfMonth Month DayOfWeek command (Day... are OR'ed)
# run at 00:05, every day
5  0  *  * *   $HOME/bin/daily.job >> $HOME/tmp/out 2>&1
# run at 14:15 on the first of every month -- output mailed to paul
15 14 1  * *   $HOME/bin/monthly
# run at 22:00 on weekdays(1-5), annoy Joe. % for newline, last % for cc:
0 22 *   * 1-5 mail -s "It's 10pm" joe%Joe,%%Where are your kids?%.%%
23 */2 1 2 *   echo "run 23 minutes after 0am, 2am, 4am ..., on Feb 1"
5  4 *   * sun echo "run at 04:05 every Sunday"
# run at 03:40 on the first Monday of each month
40 3 1-7 * *   [ "$(date +%a)" == "Mon" ] && command -args
```

Tip
For the system not running continuously, install the anacron package to schedule periodic commands at the specified intervals as closely as machine-uptime permits. See anacron(8) and anacrontab(5).

> **Tip**
> For scheduled system maintenance scripts, you can run them periodically from root account by placing such scripts in `"/etc/cron.hourly/"`, `"/etc/cron.daily/"`, `"/etc/cron.weekly/"`, or `"/etc/cron.monthly/"`. Execution timings of these scripts can be customized by `"/etc/crontab"` and `"/etc/anacrontab"`.

9.3.15   Alt-SysRq key

Insurance against system malfunction is provided by the kernel compile option "Magic SysRq key" (SAK key) which is now the default for the Debian kernel. Pressing Alt-SysRq followed by one of the following keys does the magic of rescuing control of the system.

key following Alt-SysRq	description of action
r	restore the keyboard from **r**aw mode after X crashes
0	change the console loglevel to **0** to reduce error messages
k	**k**ill all processes on the **current virtual console**
e	send a SIGT**E**RM to all processes, except for init(8)
i	send a SIGK**I**LL to all processes, except for init(8)
s	**s**ync all mounted filesystems to avoid data corruption
u	remount all mounted filesystems read-only (**u**mount)
b	re**b**oot the system without syncing or unmounting

Table 9.11: List of SAK command keys

> **Tip**
> Read the signal(7), kill(1), and sync(1) manpages to understand the description above.

The combination of "Alt-SysRq s", "Alt-SysRq u", and "Alt-SysRq r" is good for getting out of really bad situations and gaining usable keyboard access without stopping the system.

See `"/usr/share/doc/linux-doc-3.*/Documentation/sysrq.txt.gz"`.

> ⚠ **Caution**
> The Alt-SysRq feature may be considered a security risk by allowing users access to root-privileged functions. Placing `"echo 0 >/proc/sys/kernel/sysrq"` in `"/etc/rc.local"` or `"kernel.sysrq = 0"` in `"/etc/sysctl.conf"` disables the Alt-SysRq feature.

> **Tip**
> From SSH terminal etc., you can use the Alt-SysRq feature by writing to the `"/proc/sysrq-trigger"`. For example, `"echo s > /proc/sysrq-trigger; echo u > /proc/sysrq-trigger"` from the root shell prompt **s**yncs and **u**mounts all mounted filesystems.

9.4   System maintenance tips

9.4.1   Who is on the system?

You can check who is on the system by the following.

- who(1) shows who is logged on.

- w(1) shows who is logged on and what they are doing.

- last(1) shows listing of last logged in user.

- lastb(1) shows listing of last bad logged in users.

Tip

"`/var/run/utmp`", and "`/var/log/wtmp`" hold such user information. See login(1) and utmp(5).

9.4.2 Warning everyone

You can send message to everyone who is logged on to the system with wall(1) by the following.

```
$ echo "We are shutting down in 1 hour" | wall
```

9.4.3 Hardware identification

For the PCI-like devices (AGP, PCI-Express, CardBus, ExpressCard, etc.), lspci(8) (probably with "`-nn`" option) is a good start for the hardware identification.

Alternatively, you can identify the hardware by reading contents of "`/proc/bus/pci/devices`" or browsing directory tree under "`/sys/bus/pci`" (see Section 1.2.12).

package	popcon	size	description
pciutils	V:175, I:993	1220	Linux PCI Utilities: lspci(8)
usbutils	V:103, I:866	721	Linux USB utilities: lsusb(8)
pcmciautils	V:21, I:40	94	PCMCIA utilities for Linux: pccardctl(8)
scsitools	V:0, I:4	365	collection of tools for SCSI hardware management: lsscsi(8)
procinfo	V:1, I:15	122	system information obtained from "`/proc`": lsdev(8)
lshw	V:9, I:86	671	information about hardware configuration: lshw(1)
discover	V:39, I:938	90	hardware identification system: discover(8)

Table 9.12: List of hardware identification tools

9.4.4 Hardware configuration

Although most of the hardware configuration on modern GUI desktop systems such as GNOME and KDE can be managed through accompanying GUI configuration tools, it is a good idea to know some basics methods to configure them.

Here, ACPI is a newer framework for the power management system than APM.

Tip

CPU frequency scaling on modern system is governed by kernel modules such as `acpi_cpufreq`.

---

package	popcon	size	description
console-setup	V:330, I:945	438	Linux console font and keytable utilities
x11-xserver-utils	V:361, I:589	516	X server utilities: xset(1), xmodmap(1)
acpid	V:346, I:628	143	daemon to manage events delivered by the Advanced Configuration and Power Interface (ACPI)
acpi	V:28, I:603	45	utility to display information on ACPI devices
apmd	V:0, I:7	132	daemon to manage events delivered by the Advanced Power Management (APM)
sleepd	V:0, I:0	79	daemon to put a laptop to sleep during inactivity
hdparm	V:345, I:567	246	hard disk access optimization (see Section 9.5.9)
smartmontools	V:120, I:191	1716	control and monitor storage systems using S.M.A.R.T.
setserial	V:5, I:11	113	collection of tools for serial port management
memtest86+	V:1, I:34	2391	collection of tools for memory hardware management
scsitools	V:0, I:4	365	collection of tools for SCSI hardware management
setcd	V:0, I:1	35	compact disc drive access optimization
big-cursor	I:0	27	larger mouse cursors for X

Table 9.13: List of hardware configuration tools

9.4.5 System and hardware time

The following sets system and hardware time to MM/DD hh:mm, CCYY.

```
# date MMDDhhmmCCYY
# hwclock --utc --systohc
# hwclock --show
```

Times are normally displayed in the local time on the Debian system but the hardware and system time usually use UTC(GMT).

If the hardware (BIOS) time is set to UTC, change the setting to "UTC=yes" in the "/etc/default/rcS".

The following reconfigure the timezone used by the Debian system.

```
# dpkg-reconfigure tzdata
```

If you wish to update system time via network, consider to use the NTP service with the packages such as ntp, ntpdate, and chrony.

Tip
Under systemd, use systemd-timesyncd for the network time synchronization instead. See systemd-timesyncd(8).

See the following.

- Managing Accurate Date and Time HOWTO

- NTP Public Services Project

- The ntp-doc package

Tip
ntptrace(8) in the ntp package can trace a chain of NTP servers back to the primary source.

9.4.6 The terminal configuration

There are several components to configure character console and ncurses(3) system features.

• The ```/etc/terminfo/*/*``` file (terminfo(5))

• The ```$TERM``` environment variable (term(7))

• setterm(1), stty(1), tic(1), and toe(1)

If the ```terminfo``` entry for ```xterm``` doesn't work with a non-Debian ```xterm```, change your terminal type, ```$TERM```, from ```xterm``` to one of the feature-limited versions such as ```xterm-r6``` when you log in to a Debian system remotely. See ```/usr/share/doc/lib``` for more. ```dumb``` is the lowest common denominator for ```$TERM```.

9.4.7 The sound infrastructure

Device drivers for sound cards for current Linux are provided by Advanced Linux Sound Architecture (ALSA). ALSA provides emulation mode for previous Open Sound System (OSS) for compatibility.

Tip
Use ```cat /dev/urandom > /dev/audio``` or speaker-test(1) to test speaker (ˆC to stop).

Tip
If you can not get sound, your speaker may be connected to a muted output. Modern sound system has many outputs. alsamixer(1) in the ```alsa-utils``` package is useful to configure volume and mute settings.

Application softwares may be configured not only to access sound devices directly but also to access them via some standardized sound server system.

package	popcon	size	description
alsa-utils	V:389, I:535	2103	utilities for configuring and using ALSA
oss-compat	V:3, I:40	7	OSS compatibility under ALSA preventing ```/dev/dsp not found``` errors
jackd	V:4, I:28	32	JACK Audio Connection Kit. (JACK) server (low latency)
libjack0	V:0, I:16	337	JACK Audio Connection Kit. (JACK) library (low latency)
nas	V:0, I:0	239	Network Audio System (NAS) server
libaudio2	V:43, I:559	161	Network Audio System (NAS) library
pulseaudio	V:372, I:512	6105	PulseAudio server, replacement for ESD
libpulse0	V:269, I:656	989	PulseAudio client library, replacement for ESD
libgstreamer1.0-0	V:249, I:529	4860	GStreamer: GNOME sound engine
libphonon4	I:211	672	Phonon: KDE sound engine

Table 9.14: List of sound packages

There is usually a common sound engine for each popular desktop environment. Each sound engine used by the application can choose to connect to different sound servers.

9.4.8 Disabling the screen saver

For disabling the screen saver, use following commands.

environment	command
The Linux console	`setterm -powersave off`
The X Window (turning off screensaver)	`xset s off`
The X Window (disabling dpms)	`xset -dpms`
The X Window (GUI configuration of screen saver)	`xscreensaver-command -prefs`

Table 9.15: List of commands for disabling the screen saver

9.4.9 Disabling beep sounds

One can always unplug the PC speaker to disable beep sounds. Removing `pcspkr` kernel module does this for you.

The following prevents the readline(3) program used by bash(1) to beep when encountering an alert character (ASCII=7).

```
$ echo "set bell-style none">> ~/.inputrc
```

9.4.10 Memory usage

There are 2 resources available for you to get the memory usage situation.

• The kernel boot message in the "`/var/log/dmesg`" contains the total exact size of available memory.

• free(1) and top(1) display information on memory resources on the running system.

Here is an example.

```
# grep '\] Memory' /var/log/dmesg
[    0.004000] Memory: 990528k/1016784k available (1975k kernel code, 25868k reserved, 931k ←
    data, 296k init)
$ free -k
             total       used       free     shared    buffers     cached
Mem:        997184     976928      20256          0     129592     171932
-/+ buffers/cache:     675404     321780
Swap:      4545576          4    4545572
```

You may be wondering "dmesg tells me a free of 990 MB, and free -k says 320 MB is free. More than 600 MB missing ⋯".

Do not worry about the large size of "`used`" and the small size of "`free`" in the "`Mem:`" line, but read the one under them (675404 and 321780 in the example above) and relax.

For my MacBook with 1GB=1048576k DRAM (video system steals some of this), I see the following.

report	size
Total size in dmesg	1016784k = 1GB - 31792k
Free in dmesg	990528k
Total under shell	997184k
Free under shell	20256k (but effectively 321780k)

Table 9.16: List of memory sizes reported

9.4.11 System security and integrity check

Poor system maintenance may expose your system to external exploitation.

For system security and integrity check, you should start with the following.

- The debsums package, see debsums(1) and Section 2.5.2.

- The chkrootkit package, see chkrootkit(1).

- The clamav package family, see clamscan(1) and freshclam(1).

- Debian security FAQ.

- Securing Debian Manual.

package	popcon	size	description
logcheck	V:11, I:13	152	daemon to mail anomalies in the system logfiles to the administrator
debsums	V:6, I:38	120	utility to verify installed package files against MD5 checksums
chkrootkit	V:7, I:27	954	rootkit detector
clamav	V:15, I:67	893	anti-virus utility for Unix - command-line interface
tiger	V:3, I:3	2485	report system security vulnerabilities
tripwire	V:3, I:4	11480	file and directory integrity checker
john	V:2, I:14	449	active password cracking tool
aide	V:2, I:2	2038	Advanced Intrusion Detection Environment - static binary
integrit	V:0, I:0	313	file integrity verification program
crack	V:0, I:0	128	password guessing program

Table 9.17: List of tools for system security and integrity check

Here is a simple script to check for typical world writable incorrect file permissions.

```
# find / -perm 777 -a \! -type s -a \! -type l -a \! \( -type d -a -perm 1777 \)
```

> ⚠ **Caution**
> Since the debsums package uses MD5 checksums stored locally, it can not be fully trusted as the system security audit tool against malicious attacks.

9.5 Data storage tips

Booting your system with Linux live CDs or debian-installer CDs in rescue mode makes it easy for you to reconfigure data storage on your boot device.

9.5.1 Disk space usage

The disk space usage can be evaluated by programs provided by the mount, coreutils, and xdu packages:

- mount(8) reports all mounted filesystems (= disks).

- df(1) reports the disk space usage for the file system.

- du(1) reports the disk space usage for the directory tree.

Tip
You can feed the output of du(8) to xdu(1x) to produce its graphical and interactive presentation with "du -k .
|xdu", "sudo du -k -x / |xdu", etc.

9.5.2   Disk partition configuration

For disk partition configuration, although fdisk(8) has been considered standard, parted(8) deserves some attention. "Disk partitioning data", "partition table", "partition map", and "disk label" are all synonyms.

Most PCs use the classic Master Boot Record (MBR) scheme to hold disk partitioning data in the first sector, i.e., LBA sector 0 (512 bytes).

Note
Some new PCs with Extensible Firmware Interface (EFI), including Intel-based Macs, use GUID Partition Table (GPT) scheme to hold disk partitioning data not in the first sector.

Although fdisk(8) has been standard for the disk partitioning tool, parted(8) is replacing it.

package	popcon	size	GPT	description
util-linux	V:877, I:999	3328	Not supported	miscellaneous system utilities including fdisk(8) and cfdisk(8)
parted	V:368, I:572	278	Supported	GNU Parted disk partition resizing program
gparted	V:25, I:146	7231	Supported	GNOME partition editor based on libparted
gdisk	V:60, I:506	779	Supported	partition editor for the GPT disk
kpartx	V:14, I:27	84	Supported	program to create device mappings for partitions

Table 9.18: List of disk partition management packages

⚠ **Caution**
Although parted(8) claims to create and to resize filesystem too, it is safer to do such things using best maintained specialized tools such as mkfs(8) (mkfs.msdos(8), mkfs.ext2(8), mkfs.ext3(8), mkfs.ext4(8), ⋯) and resize2fs(8).

Note
In order to switch between GPT and MBR, you need to erase first few blocks of disk contents directly (see Section 9.7.6) and use "parted /dev/sdx mklabel gpt" or "parted /dev/sdx mklabel msdos" to set it. Please note "msdos" is use here for MBR.

9.5.3   Accessing partition using UUID

Although reconfiguration of your partition or activation order of removable storage media may yield different names for partitions, you can access them consistently. This is also helpful if you have multiple disks and your BIOS doesn't give them consistent device names.

• mount(8) with "-U" option can mount a block device using UUID, instead of using its file name such as "/dev/sda3".

• "/etc/fstab" (see fstab(5)) can use UUID.

• Boot loaders (Section 3.1.2) may use UUID too.

Tip
You can probe UUID of a block special device with blkid(8).

Tip
Device nodes of devices such as removable storage media can be made static by using udev rules, if needed. See Section 3.3.

9.5.4 LVM2

LVM2 is a logical volume manager for the Linux kernel. With LVM2, disk partitions can be created on logical volumes instead of the physical harddisks.

LVM requires the following.

- device-mapper support in the Linux kernel (default for Debian kernels)

- the userspace device-mapper support library (`libdevmapper*` package)

- the userspace LVM2 tools (`lvm2` package)

Please start learning LVM2 from the following manpages.

- lvm(8): Basics of LVM2 mechanism (list of all LVM2 commands)

- lvm.conf(5): Configuration file for LVM2

- lvs(8): Report information about logical volumes

- vgs(8): Report information about volume groups

- pvs(8): Report information about physical volumes

9.5.5 Filesystem configuration

For ext4 filesystem, the `e2fsprogs` package provides the following.

- mkfs.ext4(8) to create new ext4 filesystem

- fsck.ext4(8) to check and to repair existing ext4 filesystem

- tune2fs(8) to configure superblock of ext4 filesystem

- debugfs(8) to debug ext4 filesystem interactively. (It has `undel` command to recover deleted files.)

The mkfs(8) and fsck(8) commands are provided by the `e2fsprogs` package as front-ends to various filesystem dependent programs (`mkfs.fstype` and `fsck.fstype`). For ext4 filesystem, they are mkfs.ext4(8) and fsck.ext4(8) (they are symlinked to mke2fs(8) and e2fsck(8)).

Similar commands are available for each filesystem supported by Linux.

Tip
Ext4 filesystem is the default filesystem for the Linux system and strongly recommended to use it unless you have some specific reasons not to.

---

package	popcon	size	description
e2fsprogs	V:535, I:999	1224	utilities for the ext2/ext3/ext4 filesystems
reiserfsprogs	V:10, I:25	891	utilities for the Reiserfs filesystem
dosfstools	V:102, I:570	235	utilities for the FAT filesystem. (Microsoft: MS-DOS, Windows)
xfsprogs	V:18, I:88	4664	utilities for the XFS filesystem. (SGI: IRIX)
ntfs-3g	V:218, I:563	1355	utilities for the NTFS filesystem. (Microsoft: Windows NT, ···)
jfsutils	V:1, I:13	1533	utilities for the JFS filesystem. (IBM: AIX, OS/2)
reiser4progs	V:0, I:5	1325	utilities for the Reiser4 filesystem
hfsprogs	V:0, I:9	303	utilities for HFS and HFS Plus filesystem. (Apple: Mac OS)
btrfs-tools	V:3, I:35	29	utilities for the Btrfs filesystem
zerofree	V:2, I:65	25	program to zero free blocks from ext2/3/4 filesystems

Table 9.19: List of filesystem management packages

Tip

Btrfs filesystem is available in Linux kernel 3.2 (Debian wheezy). It is expected to be the next default filesystem after the ext4 filesystem.

> ⚠ **Warning**
> You should not use the Btrfs filesystem for your critical data yet before it acquires the live kernel space fsck(8) feature and the boot loader support.

Tip

Some tools allow access to filesystem without Linux kernel support (see Section 9.7.2).

9.5.6 Filesystem creation and integrity check

The mkfs(8) command creates the filesystem on a Linux system. The fsck(8) command provides the filesystem integrity check and repair on a Linux system.

Debian now defaults to no periodic `fsck` after filesystem creation.

> ⚠ **Caution**
> It is generally not safe to run `fsck` on **mounted filesystems**.

Tip

You can run the fsck(8) command safely on all filesystems including root filesystem on reboot by setting "enable_periodic_fsck" in "/etc/mke2fs.conf" and the max mount count to 0 using "tune2fs -c0 /dev/<partition_name>". See mke2fs.conf(5) and tune2fs(8).

Tip

Check files in "/var/log/fsck/" for the result of the fsck(8) command run from the boot script.

9.5.7   Optimization of filesystem by mount options

The basic static filesystem configuration is given by "`/etc/fstab`". For example,

```
# <file system> <mount point>   <type>  <options>         <dump>  <pass>
proc            /proc           proc    defaults     0       0
UUID=709cbe4c-80c1-56db-8ab1-dbce3146d2f7 / ext4 noatime,errors=remount-ro 0 1
UUID=817bae6b-45d2-5aca-4d2a-1267ab46ac23 none swap sw 0       0
/dev/scd0       /media/cdrom0   udf,iso9660 user,noauto 0       0
```

> **Tip**
> UUID (see Section 9.5.3) may be used to identify a block device instead of normal block device names such as
> "`/dev/sda1`", "`/dev/sda2`", ⋯

Performance and characteristics of a filesystem can be optimized by mount options used on it (see fstab(5) and mount(8)). Notable ones are the following.

- "`defaults`" option implies default options: "`rw,suid,dev,exec,auto,nouser,async`". (general)

- "`noatime`" or "`relatime`" option is very effective for speeding up the read access. (general)

- "`user`" option allows an ordinary user to mount the filesystem. This option implies "`noexec,nosuid,nodev`" option combination. (general, used for CDs or usb storage devices)

- "`noexec,nodev,nosuid`" option combination is used to enhance security. (general)

- "`noauto`" option limits mounting by explicit operation only. (general)

- "`data=journal`" option for ext3fs can enhance data integrity against power failure with some loss of write speed.

> **Tip**
> You need to provide kernel boot parameter (see Section 3.1.2), e.g. "`rootflags=data=journal`" to de-
> ploy a non-default journaling mode for the root filesystem. For `lenny`, the default jounaling mode is
> "`rootflags=data=ordered`". For `squeeze`, it is "`rootflags=data=writeback`".

9.5.8   Optimization of filesystem via superblock

Characteristics of a filesystem can be optimized via its superblock using the tune2fs(8) command.

- Execution of "`sudo tune2fs -l /dev/hda1`" displays the contents of the filesystem superblock on "`/dev/hda1`".

- Execution of "`sudo tune2fs -c 50 /dev/hda1`" changes frequency of filesystem checks (`fsck` execution during boot-up) to every 50 boots on "`/dev/hda1`".

- Execution of "`sudo tune2fs -j /dev/hda1`" adds journaling capability to the filesystem, i.e. filesystem conversion from ext2 to ext3 on "`/dev/hda1`". (Do this on the unmounted filesystem.)

- Execution of "`sudo tune2fs -O extents,uninit_bg,dir_index /dev/hda1 && fsck -pf /dev/hda1`" converts it from ext3 to ext4 on "`/dev/hda1`". (Do this on the unmounted filesystem.)

> **Tip**
> Despite its name, tune2fs(8) works not only on the ext2 filesystem but also on the ext3 and ext4 filesystems.

9.5.9   Optimization of hard disk

> ⚠ **Warning**
>
> Please check your hardware and read manpage of hdparam(8) before playing with hard disk configuration because this may be quite dangerous for the data integrity.

You can test disk access speed of a hard disk, e.g. "`/dev/hda`", by "`hdparm -tT /dev/hda`". For some hard disk connected with (E)IDE, you can speed it up with "`hdparm -q -c3 -d1 -u1 -m16 /dev/hda`" by enabling the "(E)IDE 32-bit I/O support", enabling the "using_dma flag", setting "interrupt-unmask flag", and setting the "multiple 16 sector I/O" (dangerous!).

You can test write cache feature of a hard disk, e.g. "`/dev/sda`", by "`hdparm -W /dev/sda`". You can disable its write cache feature with "`hdparm -W 0 /dev/sda`".

You may be able to read badly pressed CDROMs on modern high speed CD-ROM drive by slowing it down with "`setcd -x 2`".

9.5.10   Optimization of solid state drive

Performance and disk wear of the solid state drive (SSD) can be optimized as follows.

- Use the latest Linux kernel. (>= 3.2)

- Reduce disk writes for read disk accesses.

  - Set "`noatime`" or "`relatime`" mount option in `/etc/fstab`.

- Enable the TRIM command.

  - Set "`discard`" mount option in `/etc/fstab` for the ext4 filesystem, swap partition, Btrfs, etc. See fstab(5).
  - Set "`discard`" option in `/etc/lvm/lvm.conf` for LVM. See lvm.conf(5).
  - Set "`discard`" option in `/etc/crypttab` for dm-crypt. See crypttab(5).

- Enable the SSD optimized disk space allocation scheme.

  - Set "`ssd`" mount option in `/etc/fstab` for the Btrfs.

- Make system flush data to the disk every 10 minutes for laptop PCs.

  - Set "`commit=600`" mount option in `/etc/fstab`. See fstab(5).
  - Set pm-utils to use laptop-mode even under AC operation. See Debian BTS #659260.

> ⚠ **Warning**
>
> Changing flushing interval from normal 5 seconds to 10 minutes makes your data vulnerable to the power failure.

9.5.11 Using SMART to predict hard disk failure

You can monitor and log your hard disk which is compliant to SMART with the smartd(8) daemon.

1. Enable SMART feature in BIOS.

2. Install the `smartmontools` package.

3. Identify your hard disk drives by listing them with df(1).

   - Let's assume a hard disk drive to be monitored as "`/dev/hda`".

4. Check the output of "`smartctl -a /dev/hda`" to see if SMART feature is actually enabled.

   - If not, enable it by "`smartctl -s on -a /dev/hda`".

5. Enable smartd(8) daemon to run by the following.

   - uncomment "`start_smartd=yes`" in the "`/etc/default/smartmontools`" file.
   - restart the smartd(8) daemon by "`sudo /etc/init.d/smartmontools restart`".

Tip
The smartd(8) daemon can be customized with the `/etc/smartd.conf` file including how to be notified of warnings.

9.5.12 Specify temporary storage directory via $TMPDIR

Applications create temporary files normally under the temporary storage directory "`/tmp`". If "`/tmp`" does not provide enough space, you can specify such temporary storage directory via the `$TMPDIR` variable for well-behaving programs.

9.5.13 Expansion of usable storage space via LVM

For partitions created on Logical Volume Manager (LVM) (Linux feature) at install time, they can be resized easily by concatenating extents onto them or truncating extents from them over multiple storage devices without major system reconfiguration.

9.5.14 Expansion of usable storage space by mounting another partition

If you have an empty partition (e.g., "`/dev/sdx`"), you can format it with mkfs.ext4(1) and mount(8) it to a directory where you need more space. (You need to copy original data contents.)

```
$ sudo mv work-dir old-dir
$ sudo mkfs.ext4 /dev/sdx
$ sudo mount -t ext4 /dev/sdx work-dir
$ sudo cp -a old-dir/* work-dir
$ sudo rm -rf old-dir
```

Tip
You may alternatively mount an empty disk image file (see Section 9.6.5) as a loop device (see Section 9.6.3). The actual disk usage grows with the actual data stored.

9.5.15   Expansion of usable storage space by bind-mounting another directory

If you have an empty directory (e.g., "/path/to/emp-dir") on another partition with usable space, you can mount(8) it with "--bind" option to a directory (e.g., "work-dir") where you need more space.

```
$ sudo mount --bind /path/to/emp-dir work-dir
```

9.5.16   Expansion of usable storage space by overlay-mounting another directory

If you have usable space in another partition (e.g., "/path/to/empty" and "/path/to/work"), you can create a directory in it and stack that on to an old directory (e.g., "/path/to/old") where you need space using the OverlayFS for Linux kernel 3.18 or newer (Debian Stretch 9.0 or newer).

```
$ sudo mount -t overlay overlay \
  -olowerdir=/path/to/old-dir,upperdir=/path/to/empty,workdir=/path/to/work
```

Here, "/path/to/empty" and "/path/to/work" should be on the RW-enabled partition to write on "/path/to/old".

9.5.17   Expansion of usable storage space using symlink

> ⚠ **Caution**
> This is a deprecated method. Some software may not function well with "symlink to a directory". Instead, use the "mounting" approaches described in the above.

If you have an empty directory (e.g., "/path/to/emp-dir") in another partition with usable space, you can create a symlink to the directory with ln(8).

```
$ sudo mv work-dir old-dir
$ sudo mkdir -p /path/to/emp-dir
$ sudo ln -sf /path/to/emp-dir work-dir
$ sudo cp -a old-dir/* work-dir
$ sudo rm -rf old-dir
```

> ⚠ **Warning**
> Do not use "symlink to a directory" for directories managed by the system such as "/opt". Such a symlink may be overwritten when the system is upgraded.

9.6   The disk image

Here, we discuss manipulations of the disk image.

9.6.1   Making the disk image file

The disk image file, "disk.img", of an unmounted device, e.g., the second SCSI or serial ATA drive "/dev/sdb", can be made using cp(1) or dd(1) by the following.

```
# cp /dev/sdb disk.img
# dd if=/dev/sdb of=disk.img
```

The disk image of the traditional PC's master boot record (MBR) (see Section 9.5.2) which reside on the first sector on the primary IDE disk can be made by using dd(1) by the following.

```
# dd if=/dev/hda of=mbr.img bs=512 count=1
# dd if=/dev/hda of=mbr-nopart.img bs=446 count=1
# dd if=/dev/hda of=mbr-part.img skip=446 bs=1 count=66
```

- "`mbr.img`": The MBR with the partition table

- "`mbr-nopart.img`": The MBR without the partition table

- "`mbr-part.img`": The partition table of the MBR only

If you have an SCSI or serial ATA device as the boot disk, substitute "`/dev/hda`" with "`/dev/sda`".

If you are making an image of a disk partition of the original disk, substitute "`/dev/hda`" with "`/dev/hda1`" etc.

9.6.2 Writing directly to the disk

The disk image file, "`disk.img`" can be written to an unmounted device, e.g., the second SCSI drive "`/dev/sdb`" with matching size, by the following.

```
# dd if=disk.img of=/dev/sdb
```

Similarly, the disk partition image file, "`partition.img`" can be written to an unmounted partition, e.g., the first partition of the second SCSI drive "`/dev/sdb1`" with matching size, by the following.

```
# dd if=partition.img of=/dev/sdb1
```

9.6.3 Mounting the disk image file

The disk image "`partition.img`" containing a single partition image can be mounted and unmounted by using the loop device as follows.

```
# losetup -v -f partition.img
Loop device is /dev/loop0
# mkdir -p /mnt/loop0
# mount -t auto /dev/loop0 /mnt/loop0
...hack...hack...hack
# umount /dev/loop0
# losetup -d /dev/loop0
```

This can be simplified as follows.

```
# mkdir -p /mnt/loop0
# mount -t auto -o loop partition.img /mnt/loop0
...hack...hack...hack
# umount partition.img
```

Each partition of the disk image "`disk.img`" containing multiple partitions can be mounted by using the loop device. Since the loop device does not manage partitions by default, we need to reset it as follows.

```
# modinfo -p loop # verify kernel capability
max_part:Maximum number of partitions per loop device
max_loop:Maximum number of loop devices
# losetup -a # verify nothing using the loop device
# rmmod loop
# modprobe loop max_part=16
```

Now, the loop device can manage up to 16 partitions.

```
# losetup -v -f disk.img
Loop device is /dev/loop0
# fdisk -l /dev/loop0

Disk /dev/loop0: 5368 MB, 5368709120 bytes
255 heads, 63 sectors/track, 652 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x452b6464

      Device Boot      Start         End      Blocks   Id  System
/dev/loop0p1              1         600     4819468+  83  Linux
/dev/loop0p2            601         652      417690   83  Linux
# mkdir -p /mnt/loop0p1
# mount -t ext4 /dev/loop0p1 /mnt/loop0p1
# mkdir -p /mnt/loop0p2
# mount -t ext4 /dev/loop0p2 /mnt/loop0p2
...hack...hack...hack
# umount /dev/loop0p1
# umount /dev/loop0p2
# losetup -d /dev/loop0
```

Alternatively, similar effects can be done by using the device mapper devices created by kpartx(8) from the `kpartx` package as follows.

```
# kpartx -a -v disk.img
...
# mkdir -p /mnt/loop0p2
# mount -t ext4 /dev/mapper/loop0p2 /mnt/loop0p2
...
...hack...hack...hack
# umount /dev/mapper/loop0p2
...
# kpartx -d /mnt/loop0
```

Note
You can mount a single partition of such disk image with loop device using offset to skip MBR etc., too. But this is more error prone.

9.6.4 Cleaning a disk image file

A disk image file, "`disk.img`" can be cleaned of all removed files into clean sparse image "`new.img`" by the following.

```
# mkdir old; mkdir new
# mount -t auto -o loop disk.img old
# dd bs=1 count=0 if=/dev/zero of=new.img seek=5G
# mount -t auto -o loop new.img new
# cd old
# cp -a --sparse=always ./ ../new/
# cd ..
# umount new.img
# umount disk.img
```

If "`disk.img`" is in ext2, ext3 or ext4, you can also use zerofree(8) from the `zerofree` package as follows.

```
# losetup -f -v disk.img
Loop device is /dev/loop3
```

```
# zerofree /dev/loop3
# cp --sparse=always disk.img new.img
```

9.6.5   Making the empty disk image file

The empty disk image "`disk.img`" which can grow up to 5GiB can be made using dd(1) as follows.

```
$ dd bs=1 count=0 if=/dev/zero of=disk.img seek=5G
```

You can create an ext4 filesystem on this disk image "`disk.img`" using the loop device as follows.

```
# losetup -f -v disk.img
Loop device is /dev/loop1
# mkfs.ext4 /dev/loop1
...hack...hack...hack
# losetup -d /dev/loop1
$ du  --apparent-size -h disk.img
5.0G  disk.img
$ du -h disk.img
83M disk.img
```

For "`disk.img`", its file size is 5.0 GiB and its actual disk usage is mere 83MiB. This discrepancy is possible since ext4 can hold sparse file.

Tip
The actual disk usage of sparse file grows with data which are written to it.

Using similar operation on devices created by the loop device or the device mapper devices as Section 9.6.3, you can partition this disk image "`disk.img`" using parted(8) or fdisk(8), and can create filesystem on it using mkfs.ext4(8), mkswap(8), etc.

9.6.6   Making the ISO9660 image file

The ISO9660 image file, "`cd.iso`", from the source directory tree at "`source_directory`" can be made using genisoimage(1) provided by cdrkit by the following.

```
#  genisoimage -r -J -T -V volume_id -o cd.iso source_directory
```

Similarly, the bootable ISO9660 image file, "`cdboot.iso`", can be made from `debian-installer` like directory tree at "`source_directory`" by the following.

```
#  genisoimage -r -o cdboot.iso -V volume_id \
   -b isolinux/isolinux.bin -c isolinux/boot.cat \
   -no-emul-boot -boot-load-size 4 -boot-info-table source_directory
```

Here Isolinux boot loader (see Section 3.1.2) is used for booting.

You can calculate the md5sum value and make the ISO9660 image directly from the CD-ROM device as follows.

```
$ isoinfo -d -i /dev/cdrom
CD-ROM is in ISO 9660 format
...
Logical block size is: 2048
Volume size is: 23150592
...
# dd if=/dev/cdrom bs=2048 count=23150592 conv=notrunc,noerror | md5sum
# dd if=/dev/cdrom bs=2048 count=23150592 conv=notrunc,noerror > cd.iso
```

> ⚠️ **Warning**
>
> You must carefully avoid ISO9660 filesystem read ahead bug of Linux as above to get the right result.

9.6.7 Writing directly to the CD/DVD-R/RW

> **Tip**
>
> DVD is only a large CD to wodim(1) provided by cdrkit.

You can find a usable device by the following.

```
# wodim --devices
```

Then the blank CD-R is inserted to the CD drive, and the ISO9660 image file, "cd.iso" is written to this device, e.g., "/dev/hda", using wodim(1) by the following.

```
# wodim -v -eject dev=/dev/hda cd.iso
```

If CD-RW is used instead of CD-R, do this instead by the following.

```
# wodim -v -eject blank=fast dev=/dev/hda cd.iso
```

> **Tip**
>
> If your desktop system mounts CDs automatically, unmount it by "sudo umount /dev/hda" from console before using wodim(1).

9.6.8 Mounting the ISO9660 image file

If "cd.iso" contains an ISO9660 image, then the following manually mounts it to "/cdrom".

```
# mount -t iso9660 -o ro,loop cd.iso /cdrom
```

> **Tip**
>
> Modern desktop system may mount removable media such as ISO9660 formatted CD automatically (see Section 10.1.7).

9.7 The binary data

Here, we discuss direct manipulations of the binary data on storage media.

9.7.1 Viewing and editing binary data

The most basic viewing method of binary data is to use "od -t x1" command.

> **Tip**
>
> HEX is used as an acronym for hexadecimal format with radix 16. OCTAL is for octal format with radix 8. ASCII is for American Standard Code for Information Interchange, i.e., normal English text code. EBCDIC is for Extended Binary Coded Decimal Interchange Code used on IBM mainframe operating systems.

package	popcon	size	description
coreutils	V:878, I:999	15016	basic package which has od(1) to dump files (HEX, ASCII, OCTAL, ⋯)
bsdmainutils	V:854, I:998	586	utility package which has hd(1) to dump files (HEX, ASCII, OCTAL, ⋯)
hexedit	V:1, I:11	62	binary editor and viewer (HEX, ASCII)
bless	V:0, I:5	973	full featured hexadecimal editor (GNOME)
okteta	V:1, I:20	1344	full featured hexadecimal editor (KDE4)
ncurses-hexedit	V:0, I:2	128	binary editor and viewer (HEX, ASCII, EBCDIC)
beav	V:0, I:1	133	binary editor and viewer (HEX, ASCII, EBCDIC, OCTAL, ⋯)

Table 9.20: List of packages which view and edit binary data

9.7.2 Manipulating files without mounting disk

There are tools to read and write files without mounting disk.

package	popcon	size	description
mtools	V:13, I:102	361	utilities for MSDOS files without mounting them
hfsutils	V:0, I:8	1771	utilities for HFS and HFS+ files without mounting them

Table 9.21: List of packages to manipulate files without mounting disk

9.7.3 Data redundancy

Software RAID systems offered by the Linux kernel provide data redundancy in the kernel filesystem level to achieve high levels of storage reliability.

There are tools to add data redundancy to files in application program level to achieve high levels of storage reliability, too.

package	popcon	size	description
par2	V:2, I:10	246	Parity Archive Volume Set, for checking and repair of files
dvdisaster	V:0, I:2	1588	data loss/scratch/aging protection for CD/DVD media
dvbackup	V:0, I:0	412	backup tool using MiniDV camcorders (providing rsbep(1))
vdmfec	V:0, I:0	97	recover lost blocks using Forward Error Correction

Table 9.22: List of tools to add data redundancy to files

9.7.4 Data file recovery and forensic analysis

There are tools for data file recovery and forensic analysis.

Tip
You can undelete files on the ext2 filesystem using `list_deleted_inodes` and `undel` commands of debugfs(8) in the `e2fsprogs` package.

package	popcon	size	description
testdisk	V:3, I:39	1315	utilities for partition scan and disk recovery
magicrescue	V:0, I:3	224	utility to recover files by looking for magic bytes
scalpel	V:0, I:4	82	frugal, high performance file carver
myrescue	V:0, I:3	38	rescue data from damaged harddisks
extundelete	V:1, I:11	148	utility to undelete files on the ext3/4 filesystem
ext4magic	V:0, I:3	232	utility to undelete files on the ext3/4 filesystem
ext3grep	V:0, I:3	278	tool to help recover deleted files on the ext3 filesystem
scrounge-ntfs	V:0, I:3	45	data recovery program for NTFS filesystems
gzrt	V:0, I:0	57	gzip recovery toolkit
sleuthkit	V:2, I:15	1113	tools for forensics analysis. (Sleuthkit)
autopsy	V:0, I:2	1021	graphical interface to SleuthKit
foremost	V:0, I:7	123	forensics application to recover data
guymager	V:0, I:1	1044	forensic imaging tool based on Qt
dcfldd	V:0, I:5	94	enhanced version of dd for forensics and security

Table 9.23: List of packages for data file recovery and forensic analysis

9.7.5 Splitting a large file into small files

When a data is too big to backup as a single file, you can backup its content after splitting it into, e.g. 2000MiB chunks and merge those chunks back into the original file later.

```
$ split -b 2000m large_file
$ cat x* >large_file
```

> **⚠ Caution**
> Please make sure you do not have any files starting with "x" to avoid name crashes.

9.7.6 Clearing file contents

In order to clear the contents of a file such as a log file, do not use rm(1) to delete the file and then create a new empty file, because the file may still be accessed in the interval between commands. The following is the safe way to clear the contents of the file.

```
$ :>file_to_be_cleared
```

9.7.7 Dummy files

The following commands create dummy or empty files.

```
$ dd if=/dev/zero    of=5kb.file bs=1k count=5
$ dd if=/dev/urandom of=7mb.file bs=1M count=7
$ touch zero.file
$ : > alwayszero.file
```

You should find following files.

- "5kb.file" is 5KB of zeros.

- "7mb.file" is 7MB of random data.

- "zero.file" may be a 0 byte file. If it existed, its mtime is updated while its content and its length are kept.

- "alwayszero.file" is always a 0 byte file. If it existed, its mtime is updated and its content is reset.

9.7.8 Erasing an entire hard disk

There are several ways to completely erase data from an entire hard disk like device, e.g., USB memory stick at "/dev/sda".

> ⚠ **Caution**
> Check your USB memory stick location with mount(8) first before executing commands here. The device
> pointed by "/dev/sda" may be SCSI hard disk or serial-ATA hard disk where your entire system resides.

Erase all the disk content by resetting data to 0 with the following.

```
# dd if=/dev/zero of=/dev/sda
```

Erase everything by overwriting with random data as follows.

```
# dd if=/dev/urandom of=/dev/sda
```

Erase everything by overwriting with random data very efficiently as follows.

```
# shred -v -n 1 /dev/sda
```

Since dd(1) is available from the shell of many bootable Linux CDs such as Debian installer CD, you can erase your installed
system completely by running an erase command from such media on the system hard disk, e.g., "/dev/hda", "/dev/sda",
etc.

9.7.9 Erasing unused area of an hard disk

Unused area on an hard disk (or USB memory stick), e.g. "/dev/sdb1" may still contain erased data themselves since they are
only unlinked from the filesystem. These can be cleaned by overwriting them.

```
# mount -t auto /dev/sdb1 /mnt/foo
# cd /mnt/foo
# dd if=/dev/zero of=junk
dd: writing to 'junk': No space left on device
...
# sync
# umount /dev/sdb1
```

> ⚠ **Warning**
> This is usually good enough for your USB memory stick. But this is not perfect. Most parts of erased
> filenames and their attributes may be hidden and remain in the filesystem.

9.7.10 Undeleting deleted but still open files

Even if you have accidentally deleted a file, as long as that file is still being used by some application (read or write mode), it is
possible to recover such a file.

For example, try the following

```
$ echo foo > bar
$ less bar
$ ps aux | grep ' less[ ]'
bozo     4775  0.0  0.0  92200   884 pts/8    S+   00:18   0:00 less bar
```

```
$ rm bar
$ ls -l /proc/4775/fd | grep bar
lr-x------ 1 bozo bozo 64 2008-05-09 00:19 4 -> /home/bozo/bar (deleted)
$ cat /proc/4775/fd/4 >bar
$ ls -l
-rw-r--r-- 1 bozo bozo 4 2008-05-09 00:25 bar
$ cat bar
foo
```

Execute on another terminal (when you have the `lsof` package installed) as follows.

```
$ ls -li bar
2228329 -rw-r--r-- 1 bozo bozo 4 2008-05-11 11:02 bar
$ lsof |grep bar|grep less
less 4775 bozo 4r REG 8,3 4 2228329 /home/bozo/bar
$ rm bar
$ lsof |grep bar|grep less
less 4775 bozo 4r REG 8,3 4 2228329 /home/bozo/bar (deleted)
$ cat /proc/4775/fd/4 >bar
$ ls -li bar
2228302 -rw-r--r-- 1 bozo bozo 4 2008-05-11 11:05 bar
$ cat bar
foo
```

9.7.11   Searching all hardlinks

Files with hardlinks can be identified by "`ls -li`".

```
$ ls -li
total 0
2738405 -rw-r--r-- 1 root root 0 2008-09-15 20:21 bar
2738404 -rw-r--r-- 2 root root 0 2008-09-15 20:21 baz
2738404 -rw-r--r-- 2 root root 0 2008-09-15 20:21 foo
```

Both "`baz`" and "`foo`" have link counts of "2" (>1) showing them to have hardlinks. Their inode numbers are common "2738404". This means they are the same hardlinked file. If you do not happen to find all hardlinked files by chance, you can search it by the inode, e.g., "2738404" as the following.

```
# find /path/to/mount/point -xdev -inum 2738404
```

9.7.12   Invisible disk space consumption

All deleted but open files consume disk space although they are not visible from normal du(1). They can be listed with their size by the following.

```
# lsof -s -X / |grep deleted
```

9.8   Data encryption tips

With physical access to your PC, anyone can easily gain root privilege and access all the files on your PC (see Section 4.7.4). This means that login password system can not secure your private and sensitive data against possible theft of your PC. You must deploy data encryption technology to do it. Although GNU privacy guard (see Section 10.3) can encrypt files, it takes some user efforts.

dm-crypt and eCryptfs facilitates automatic data encryption natively via Linux kernel modules with minimal user efforts.

package	popcon	size	description
cryptsetup	V:34, I:79	376	utilities for encrypted block device (dm-crypt / LUKS)
cryptmount	V:3, I:4	225	utilities for encrypted block device (dm-crypt / LUKS) with focus on mount/unmount by normal users
ecryptfs-utils	V:6, I:9	396	utilities for encrypted stacked filesystem (eCryptfs)

Table 9.24: List of data encryption utilities

Dm-crypt is a cryptographic filesystem using device-mapper. Device-mapper maps one block device to another.

eCryptfs is another cryptographic filesystem using stacked filesystem. Stacked filesystem stacks itself on top of an existing directory of a mounted filesystem.

> ⚠ **Caution**
> Data encryption costs CPU time etc. Please weigh its benefits and costs.

> **Note**
> Entire Debian system can be installed on a encrypted disk by the debian-installer (lenny or newer) using dm-crypt/LUKS and initramfs.

> **Tip**
> See Section 10.3 for user space encryption utility: GNU Privacy Guard.

9.8.1 Removable disk encryption with dm-crypt/LUKS

You can encrypt contents of removable mass devices, e.g. USB memory stick on "/dev/sdx", using dm-crypt/LUKS. You simply format it as the following.

```
# badblocks -c 1024 -s -w -t random -v /dev/sdx
# fdisk /dev/sdx
... "n" "p" "1" "return" "return" "w"
# cryptsetup luksFormat /dev/sdx1
...
# cryptsetup open --type luks /dev/sdx1 sdx1
...
# ls -l /dev/mapper/
total 0
crw-rw---- 1 root root  10, 60 2008-10-04 18:44 control
brw-rw---- 1 root disk 254,   0 2008-10-04 23:55 sdx1
# mkfs.vfat /dev/mapper/sdx1
...
# cryptsetup luksClose sdx1
```

Then, it can be mounted just like normal one on to "/media/<disk_label>", except for asking password (see Section 10.1.7) under modern desktop environment, such as GNOME using gnome-mount(1). The difference is that every data written to it is encrypted. You may alternatively format media in different filesystem, e.g., ext4 with "mkfs.ext4 /dev/mapper/sdx1".

> **Note**
> If you are really paranoid for the security of data, you may need to overwrite multiple times (the "badblocks" command in the above example). This operation is very time consuming though.

9.8.2   Encrypted swap partition with dm-crypt

Let's assume that your original "`/etc/fstab`" contains the following.

```
/dev/sda7 swap sw 0 0
```

You can enable encrypted swap partition using dm-crypt by as the following.

```
# aptitude install cryptsetup
# swapoff -a
# echo "cswap /dev/sda7 /dev/urandom swap" >> /etc/crypttab
# perl -i -p -e "s/\/dev\/sda7/\/dev\/mapper\/cswap/" /etc/fstab
# /etc/init.d/cryptdisks restart
 ...
# swapon -a
```

9.8.3   Mounting encrypted disk with dm-crypt/LUKS

An encrypted disk partition created with dm-crypt/LUKS on "`/dev/sdc5`" can be mounted onto "`/mnt`" as follows:

```
$ sudo cryptsetup open /dev/sdc5 ninja --type luks
Enter passphrase for /dev/sdc5: ****
$ sudo lvm
lvm> lvscan
  inactive          '/dev/ninja-vg/root' [13.52 GiB] inherit
  inactive          '/dev/ninja-vg/swap_1' [640.00 MiB] inherit
  ACTIVE            '/dev/goofy/root' [180.00 GiB] inherit
  ACTIVE            '/dev/goofy/swap' [9.70 GiB] inherit
lvm> lvchange -a y /dev/ninja-vg/root
lvm> exit
  Exiting.
$ sudo mount /dev/ninja-vg/root /mnt
```

9.8.4   Automatically encrypting files with eCryptfs

You can encrypt files written under "`~/Private/`" automatically using eCryptfs and the `ecryptfs-utils` package.

- Run ecryptfs-setup-private(1) and set up "`~/Private/`" by following prompts.

- Activate "`~/Private/`" by running ecryptfs-mount-private(1).

- Move sensitive data files to "`~/Private/`" and make symlinks as needed.

    - Candidates: "`~/.fetchmailrc`", "`~/.ssh/identity`", "`~/.ssh/id_rsa`", "`~/.ssh/id_dsa`" and other files with "`go-rwx`"

- Move sensitive data directories to a subdirectory in "`~/Private/`" and make symlinks as needed.

    - Candidates: "`~/.gnupg`" and other directories with "`go-rwx`"

- Create symlink from "`~/Desktop/Private/`" to "`~/Private/`" for easier desktop operations.

- Deactivate "`~/Private/`" by running ecryptfs-umount-private(1).

- Activate "`~/Private/`" by issuing "`ecryptfs-mount-private`" as you need encrypted data.

Tip
Since eCryptfs selectively encrypt only the sensitive files, its system cost is much less than using dm-crypt on the entire root or "/home" device. It does not require any special on-disk storage allocation effort but cannot keep all filesystem metadata confidential.

9.8.5 Automatically mounting eCryptfs

If you use your login password for wrapping encryption keys, you can automate mounting eCryptfs via PAM (Pluggable Authentication Modules).

Insert the following line just before "`pam_permit.so`" in "`/etc/pam.d/common-auth`".

```
auth required pam_ecryptfs.so unwrap
```

Insert the following line just at the last line in "`/etc/pam.d/common-session`".

```
session optional pam_ecryptfs.so unwrap
```

Insert the following line at first active line in "`/etc/pam.d/common-password`".

```
password required pam_ecryptfs.so
```

This is quite convenient.

> **Warning**
> Configuration errors of PAM may lock you out of your own system. See Chapter 4.

> **Caution**
> If you use your login password for wrapping encryption keys, your encrypted data are as secure as your user login password (see Section 4.3). Unless you are careful to set up a strong password, your data is at risk when someone runs password cracking software after stealing your laptop (see Section 4.7.4).

9.9 The kernel

Debian distributes modularized Linux kernel as packages for supported architectures.

9.9.1 Linux kernel 2.6/3.x

There are few notable features on Linux kernel 2.6/3.x compared to 2.4.

- Devices are created by the udev system (see Section 3.3).
- Read/write accesses to IDE CD/DVD devices do not use the `ide-scsi` module.
- Network packet filtering functions use `iptables` kernel modules.

The version bump from Linux 2.6.39 to Linux 3.0 is not about major technological changes but about the 20th anniversary.

9.9.2 Kernel parameters

Many Linux features are configurable via kernel parameters as follows.

- Kernel parameters initialized by the bootloader (see Section 3.1.2)
- Kernel parameters changed by sysctl(8) at runtime for ones accessible via sysfs (see Section 1.2.12)
- Module parameters set by arguments of modprobe(8) when a module is activated (see Section 9.6.3)

See "`kernel-parameters.txt(.gz)`" and other related documents in the Linux kernel documentation ("`/usr/share/doc/li`
provided by the `linux-doc-3.*` package.

9.9.3   Kernel headers

Most **normal programs** don't need kernel headers and in fact may break if you use them directly for compiling. They should be compiled against the headers in "`/usr/include/linux`" and "`/usr/include/asm`" provided by the `libc6-dev` package (created from the `glibc` source package) on the Debian system.

Note

For compiling some kernel-specific programs such as the kernel modules from the external source and the automounter daemon (`amd`), you must include path to the corresponding kernel headers, e.g. "`-I/usr/src/linux-particular-version/include/`", to your command line. module-assistant(8) (or its short form `m-a`) helps users to build and install module package(s) easily for one or more custom kernels.

9.9.4   Compiling the kernel and related modules

Debian has its own method of compiling the kernel and related modules.

package	popcon	size	description
build-essential	I:445	20	essential packages for building Debian packages: `make`, `gcc`, ⋯
bzip2	V:196, I:946	184	compress and decompress utilities for bz2 files
libncurses5-dev	V:15, I:147	993	developer's libraries and docs for ncurses
git	V:279, I:436	33191	git: distributed revision control system used by the Linux kernel
fakeroot	V:32, I:495	215	provide fakeroot environment for building package as non-root
initramfs-tools	V:256, I:991	105	tool to build an initramfs (Debian specific)
dkms	V:97, I:207	282	dynamic kernel module support (DKMS) (generic)
devscripts	V:9, I:64	2175	helper scripts for a Debian Package maintainer (Debian specific)

Table 9.25: List of key packages to be installed for the kernel recompilation on the Debian system

If you use `initrd` in Section 3.1.2, make sure to read the related information in initramfs-tools(8), update-initramfs(8), mkinitramfs(8) and initramfs.conf(5).

⚠ **Warning**

Do not put symlinks to the directories in the source tree (e.g. "`/usr/src/linux*`") from "`/usr/include/linux`" and "`/usr/include/asm`" when compiling the Linux kernel source. (Some outdated documents suggest this.)

Note

When compiling the latest Linux kernel on the Debian `stable` system, the use of backported latest tools from the Debian `unstable` may be needed.

Note

The dynamic kernel module support (DKMS) is a new distribution independent framework designed to allow individual kernel modules to be upgraded without changing the whole kernel. This is used for the maintenance of out-of-tree modules. This also makes it very easy to rebuild modules as you upgrade kernels.

9.9.5   Compiling the kernel source: Debian Kernel Team recommendation

For building custom kernel binary packages from the upstream kernel source, you should use the "deb-pkg" target provided by it.

```
$ sudo apt-get build-dep linux
$ cd /usr/src
$ wget http://www.kernel.org/pub/linux/kernel/v3.11/linux-<version>.tar.bz2
$ tar -xjvf linux-<version>.tar.bz2
$ cd linux-<version>
$ cp /boot/config-<version> .config
$ make menuconfig
 ...
$ make deb-pkg
```

Tip
The linux-source-<version> package provides the Linux kernel source with Debian patches as "/usr/src/linux-<version>.tar.bz2".

For building specific binary packages from the Debian kernel source package, you should use the "binary-arch_<architecture>_ targets in "debian/rules.gen".

```
$ sudo apt-get build-dep linux
$ apt-get source linux
$ cd linux-3.*
$ fakeroot make -f debian/rules.gen binary-arch_i386_none_686
```

See further information:

- Debian Wiki: KernelFAQ

- Debian Wiki: DebianKernel

- Debian Linux Kernel Handbook: https://kernel-handbook.debian.net

9.9.6   Hardware drivers and firmware

The hardware driver is the code running on the target system. Most hardware drivers are available as free software now and are included in the normal Debian kernel packages in the main area.

- GPU driver

  - Intel GPU driver (main)
  - AMD/ATI GPU driver (main)
  - NVIDIA GPU driver (main for nouveau driver.)

The firmware is the code or data loaded on the device (e.g., CPU microcode, rendering code running on GPU, or FPGA / CPLD data, ···). Some firmware packages are available as free software but many firmware packages are not available as free software since they contain sourceless binary data.

- firmware-linux-free (main)

9.10   Virtualized system

Use of virtualized system enables us to run multiple instances of system simultaneously on a single hardware.

Tip
See http://wiki.debian.org/SystemVirtualization.

9.10.1   Virtualization tools

There are several system virtualization and emulation related packages in Debian beyond simple chroot. Some packages also help you to setup such system.

package	popcon	size	description
schroot	V:7, I:10	2659	specialized tool for executing Debian binary packages in chroot
sbuild	V:1, I:4	283	tool for building Debian binary packages from Debian sources
pbuilder	V:2, I:17	958	personal package builder for Debian packages
debootstrap	V:6, I:65	261	bootstrap a basic Debian system (written in sh)
cdebootstrap	V:0, I:3	112	bootstrap a Debian system (written in C)
virt-manager	V:8, I:33	7722	Virtual Machine Manager: desktop application for managing virtual machines
libvirt-clients	V:27, I:49	1933	programs for the libvirt library
bochs	V:0, I:1	4086	Bochs: IA-32 PC emulator
qemu	I:37	673	QEMU: fast generic processor emulator
qemu-system	I:40	92	QEMU: full system emulation binaries
qemu-user	V:3, I:39	61611	QEMU: user mode emulation binaries
qemu-utils	V:10, I:96	5418	QEMU: utilities
qemu-kvm	V:20, I:69	101	KVM: full virtualization on x86 hardware with the hardware-assisted virtualization
virtualbox	V:33, I:43	76054	VirtualBox: x86 virtualization solution on i386 and amd64
xen-tools	V:0, I:6	666	tools to manage debian XEN virtual server
wine	V:23, I:101	184	Wine: Windows API Implementation (standard suite)
dosbox	V:2, I:20	2706	DOSBox: x86 emulator with Tandy/Herc/CGA/EGA/VGA/SVGA graphics, sound and DOS
dosemu	V:0, I:4	4891	DOSEMU: The Linux DOS Emulator
vzctl	V:1, I:3	1060	OpenVZ server virtualization solution - control tools
vzquota	V:1, I:3	221	OpenVZ server virtualization solution - quota tools
lxc	V:8, I:13	1985	Linux containers user space tools

Table 9.26: List of virtualization tools

See Wikipedia article Comparison of platform virtual machines for detail comparison of different platform virtualization solutions.

9.10.2   Virtualization work flow

Note
Some functionalities described here are only available in squeeze or later.

Note
Default Debian kernels support KVM since lenny.

Typical work flow for virtualization involves several steps.

- Create an empty filesystem (a file tree or a disk image).

  - The file tree can be created by "`mkdir -p /path/to/chroot`".
  - The raw disk image file can be created with dd(1) (see Section 9.6.1 and Section 9.6.5).
  - qemu-img(1) can be used to create and convert disk image files supported by QEMU.
  - The raw and VMDK file format can be used as common format among virtualization tools.

- Mount the disk image with mount(8) to the filesystem (optional).

  - For the raw disk image file, mount it as loop device or device mapper devices (see Section 9.6.3).
  - For disk images supported by QEMU, mount them as network block device (see Section 9.10.3).

- Populate the target filesystem with required system data.

  - The use of programs such as `debootstrap` and `cdebootstrap` helps with this process (see Section 9.10.4).
  - Use installers of OSs under the full system emulation.

- Run a program under a virtualized environment.

  - chroot provides basic virtualized environment enough to compile programs, run console applications, and run daemons in it.
  - QEMU provides cross-platform CPU emulation.
  - QEMU with KVM provides full system emulation by the hardware-assisted virtualization.
  - VirtualBox provides full system emulation on i386 and amd64 with or without the hardware-assisted virtualization.

9.10.3   Mounting the virtual disk image file

For the raw disk image file, see Section 9.6.

For other virtual disk image files, you can use qemu-nbd(8) to export them using network block device protocol and mount them using the `nbd` kernel module.

qemu-nbd(8) supports disk formats supported by QEMU: QEMU supports following disk formats: raw, qcow2, qcow, vmdk, vdi, bochs, cow (user-mode Linux copy-on-write), parallels, dmg, cloop, vpc, vvfat (virtual VFAT), and host_device.

The network block device can support partitions in the same way as the loop device (see Section 9.6.3). You can mount the first partition of "`disk.img`" as follows.

```
# modprobe nbd max_part=16
# qemu-nbd -v -c /dev/nbd0 disk.img
...
# mkdir /mnt/part1
# mount /dev/nbd0p1 /mnt/part1
```

Tip
You may export only the first partition of "`disk.img`" using "`-P 1`" option to qemu-nbd(8).

9.10.4 Chroot system

chroot(8) offers most basic way to run different instances of the GNU/Linux environment on a single system simultaneously without rebooting.

> ⚠ **Caution**
> Examples below assumes both parent system and chroot system share the same CPU architecture.

You can learn how to setup and use chroot(8) by running pbuilder(8) program under script(1) as follows.

```
$ sudo mkdir /sid-root
$ sudo pbuilder --create --no-targz --debug --buildplace /sid-root
```

You see how debootstrap(8) or cdebootstrap(1) populate system data for `sid` environment under "`/sid-root`".

> **Tip**
> These debootstrap(8) or cdebootstrap(1) are used to install Debian by the Debian Installer. These can also be used to install Debian to a system without using a Debian install disk, but instead from another GNU/Linux distribution.

```
$ sudo pbuilder --login --no-targz  --debug --buildplace /sid-root
```

You see how a system shell running under `sid` environment is created as the following.

1. Copy local configuration ("`/etc/hosts`", "`/etc/hostname`", "`/etc/resolv.conf`")

2. Mount "`/proc`" filesystem

3. Mount "`/dev/pts`" filesystem

4. Create "`/usr/sbin/policy-rc.d`" which always exits with 101

5. Run "`chroot /sid-root bin/bash -c 'exec -a -bash bin/bash'`"

> **Note**
> Some programs under chroot may require access to more files from the parent system to function than `pbuilder` provides. For example, "/sys", "/etc/passwd", "/etc/group", "/var/run/utmp", "/var/log/wtmp", etc. may need to be bind-mounted or copied.

> **Note**
> The "/usr/sbin/policy-rc.d" file prevents daemon programs to be started automatically on the Debian system. See "/usr/share/doc/sysv-rc/README.policy-rc.d.gz".

> **Tip**
> The original purpose of the specialized chroot package, `pbuilder` is to construct a chroot system and builds a package inside the chroot. It is an ideal system to use to check that a package's build-dependencies are correct, and to be sure that unnecessary and wrong build dependencies do not exist in the resulting package.

> **Tip**
> Similar `schroot` package may give you an idea to run `i386` chroot system under `amd64` parent system.

9.10.5   Multiple desktop systems

I recommend you to use QEMU or VirtualBox on a Debian `stable` system to run multiple desktop systems safely using virtualization. These enable you to run desktop applications of Debian `unstable` and `testing` without usual risks associated with them.

Since pure QEMU is very slow, it is recommended to accelerate it with KVM when the host system support it.

The virtual disk image "`virtdisk.qcow2`" containing a Debian system for QEMU can be created using debian-installer: Small CDs as follows.

```
$ wget http://cdimage.debian.org/debian-cd/5.0.3/amd64/iso-cd/debian-503-amd64-netinst.iso
$ qemu-img create -f qcow2 virtdisk.qcow2 5G
$ qemu -hda virtdisk.qcow2 -cdrom debian-503-amd64-netinst.iso -boot d -m 256
...
```

See more tips at Debian wiki: QEMU.

VirtualBox comes with Qt GUI tools and quite intuitive. Its GUI and command line tools are explained in VirtualBox User Manual and VirtualBox User Manual (PDF).

Tip
Running other GNU/Linux distributions such as Ubuntu and Fedora under virtualization is a great way to learn configuration tips. Other proprietary OSs may be run nicely under this GNU/Linux virtualization, too.

Chapter 10

Data management

Tools and tips for managing binary and text data on the Debian system are described.

10.1 Sharing, copying, and archiving

⚠ **Warning**
The uncoordinated write access to actively accessed devices and files from multiple processes must not be done to avoid the race condition. File locking mechanisms using flock(1) may be used to avoid it.

The security of the data and its controlled sharing have several aspects.

- The creation of data archive
- The remote storage access
- The duplication
- The tracking of the modification history
- The facilitation of data sharing
- The prevention of unauthorized file access
- The detection of unauthorized file modification

These can be realized by using some combination of tools.

- Archive and compression tools
- Copy and synchronization tools
- Network filesystems
- Removable storage media
- The secure shell
- The authentication system
- Version control system tools
- Hash and cryptographic encryption tools

10.1.1   Archive and compression tools

Here is a summary of archive and compression tools available on the Debian system.

package	popcon	size	extension	command	comment
tar	V:907, I:999	2770	.tar	tar(1)	the standard archiver (de facto standard)
cpio	V:365, I:999	989	.cpio	cpio(1)	Unix System V style archiver, use with find(1)
binutils	V:201, I:699	90	.ar	ar(1)	archiver for the creation of static libraries
fastjar	V:5, I:51	172	.jar	fastjar(1)	archiver for Java (zip like)
pax	V:15, I:42	175	.pax	pax(1)	new POSIX standard archiver, compromise between tar and cpio
gzip	V:876, I:999	231	.gz	gzip(1), zcat(1), ⋯	GNU LZ77 compression utility (de facto standard)
bzip2	V:196, I:946	184	.bz2	bzip2(1), bzcat(1), ⋯	Burrows-Wheeler block-sorting compression utility with higher compression ratio than gzip(1) (slower than gzip with similar syntax)
lzma	V:4, I:48	141	.lzma	lzma(1)	LZMA compression utility with higher compression ratio than gzip(1) (deprecated)
xz-utils	V:441, I:958	515	.xz	xz(1), xzdec(1), ⋯	XZ compression utility with higher compression ratio than bzip2(1) (slower than gzip but faster than bzip2; replacement for LZMA compression utility)
p7zip	V:78, I:385	986	.7z	7zr(1), p7zip(1)	7-Zip file archiver with high compression ratio (LZMA compression)
p7zip-full	V:143, I:531	4659	.7z	7z(1), 7za(1)	7-Zip file archiver with high compression ratio (LZMA compression and others)
lzop	V:6, I:48	97	.lzo	lzop(1)	LZO compression utility with higher compression and decompression speed than gzip(1) (lower compression ratio than gzip with similar syntax)
zip	V:52, I:432	608	.zip	zip(1)	InfoZIP: DOS archive and compression tool
unzip	V:290, I:803	534	.zip	unzip(1)	InfoZIP: DOS unarchive and decompression tool

Table 10.1: List of archive and compression tools

⚠ **Warning**
Do not set the "$TAPE" variable unless you know what to expect. It changes tar(1) behavior.

Note
The gzipped tar(1) archive uses the file extension ".tgz" or ".tar.gz".

Note
The xz-compressed tar(1) archive uses the file extension ".txz" or ".tar.xz".

Note

Popular compression method in FOSS tools such as tar(1) has been moving as follows: `gzip` → `bzip2` → `xz`

Note

cp(1), scp(1) and tar(1) may have some limitation for special files. cpio(1) is most versatile.

Note

cpio(1) is designed to be used with find(1) and other commands and suitable for creating backup scripts since the file selection part of the script can be tested independently.

Note

Internal structure of Libreoffice data files are "`.jar`" file which can be opened also by `unzip`.

Note

The de-facto cross platform archive tool is `zip`. Use it as "`zip -rX`" to attain the maximum compatibility. Use also the "`-s`" option, if the maximum file size matters.

10.1.2   Copy and synchronization tools

Here is a summary of simple copy and backup tools available on the Debian system.

package	popcon	size	tool	function
coreutils	V:878, I:999	15016	GNU cp	locally copy files and directories ("-a" for recursive)
openssh-client	V:821, I:996	4151	scp	remotely copy files and directories (client, "`-r`" for recursive)
openssh-server	V:680, I:802	873	sshd	remotely copy files and directories (remote server)
rsync	V:269, I:602	691	-	1-way remote synchronization and backup
unison	V:4, I:19	3452	-	2-way remote synchronization and backup

Table 10.2: List of copy and synchronization tools

Copying files with rsync(8) offers richer features than others.

- delta-transfer algorithm that sends only the differences between the source files and the existing files in the destination

- quick check algorithm (by default) that looks for files that have changed in size or in last-modified time

- "`--exclude`" and "`--exclude-from`" options similar to tar(1)

- "a trailing slash on the source directory" syntax that avoids creating an additional directory level at the destination.

Tip

Execution of the `bkup` script mentioned in Section 10.2.3 with the "`-gl`" option under cron(8) should provide very similar functionality as Plan9's `dumpfs` for the static data archive.

Tip

Version control system (VCS) tools in Table 10.11 can function as the multi-way copy and synchronization tools.

10.1.3   Idioms for the archive

Here are several ways to archive and unarchive the entire content of the directory "./source" using different tools.

GNU tar(1):

```
$ tar -cvJf archive.tar.xz ./source
$ tar -xvJf archive.tar.xz
```

Alternatively, by the following.

```
$ find ./source -xdev -print0 | tar -cvJf archive.tar.xz --null -F -
```

cpio(1):

```
$ find ./source -xdev -print0 | cpio -ov --null > archive.cpio; xz archive.cpio
$ zcat archive.cpio.xz | cpio -i
```

10.1.4   Idioms for the copy

Here are several ways to copy the entire content of the directory "./source" using different tools.

• Local copy: "./source" directory → "/dest" directory

• Remote copy: "./source" directory at local host → "/dest" directory at "user@host.dom" host

rsync(8):

```
# cd ./source; rsync -aHAXSv . /dest
# cd ./source; rsync -aHAXSv . user@host.dom:/dest
```

You can alternatively use "a trailing slash on the source directory" syntax.

```
# rsync -aHAXSv ./source/ /dest
# rsync -aHAXSv ./source/ user@host.dom:/dest
```

Alternatively, by the following.

```
# cd ./source; find . -print0 | rsync -aHAXSv0 --files-from=- . /dest
# cd ./source; find . -print0 | rsync -aHAXSv0 --files-from=- . user@host.dom:/dest
```

GNU cp(1) and openSSH scp(1):

```
# cd ./source; cp -a . /dest
# cd ./source; scp -pr . user@host.dom:/dest
```

GNU tar(1):

```
# (cd ./source && tar cf - . ) | (cd /dest && tar xvfp - )
# (cd ./source && tar cf - . ) | ssh user@host.dom '(cd /dest && tar xvfp - )'
```

cpio(1):

```
# cd ./source; find . -print0 | cpio -pvdm --null --sparse /dest
```

You can substitute "." with "foo" for all examples containing "." to copy files from "./source/foo" directory to "/dest/foo" directory.

You can substitute "." with the absolute path "/path/to/source/foo" for all examples containing "." to drop "cd ./source;". These copy files to different locations depending on tools used as follows.

- "`/dest/foo`": rsync(8), GNU cp(1), and scp(1)

- "`/dest/path/to/source/foo`": GNU tar(1), and cpio(1)

Tip
rsync(8) and GNU cp(1) have option "`-u`" to skip files that are newer on the receiver.

10.1.5   Idioms for the selection of files

find(1) is used to select files for archive and copy commands (see Section 10.1.3 and Section 10.1.4) or for xargs(1) (see Section 9.3.9). This can be enhanced by using its command arguments.

Basic syntax of find(1) can be summarized as the following.

- Its conditional arguments are evaluated from left to right.

- This evaluation stops once its outcome is determined.

- "Logical **OR**" (specified by "`-o`" between conditionals) has lower precedence than "logical **AND**" (specified by "`-a`" or nothing between conditionals).

- "Logical **NOT**" (specified by "`!`" before a conditional) has higher precedence than "logical **AND**".

- "`-prune`" always returns logical **TRUE** and, if it is a directory, searching of file is stopped beyond this point.

- "`-name`" matches the base of the filename with shell glob (see Section 1.5.6) but it also matches its initial "`.`" with metacharacters such as "`*`" and "`?`". (New POSIX feature)

- "`-regex`" matches the full path with emacs style **BRE** (see Section 1.6.2) as default.

- "`-size`" matches the file based on the file size (value precedented with "`+`" for larger, precedented with "`-`" for smaller)

- "`-newer`" matches the file newer than the one specified in its argument.

- "`-print0`" always returns logical **TRUE** and print the full filename (null terminated) on the standard output.

find(1) is often used with an idiomatic style as the following.

```
# find /path/to \
    -xdev -regextype posix-extended \
    -type f -regex ".*\.cpio|.*~" -prune -o \
    -type d -regex ".*/\.git" -prune -o \
    -type f -size +99M -prune -o \
    -type f -newer /path/to/timestamp -print0
```

This means to do following actions.

1. Search all files starting from "`/path/to`"

2. Globally limit its search within its starting filesystem and uses **ERE** (see Section 1.6.2) instead

3. Exclude files matching regex of "`.*\.cpio`" or "`.*~`" from search by stop processing

4. Exclude directories matching regex of "`.*/\.git`" from search by stop processing

5. Exclude files larger than 99 Megabytes (units of 1048576 bytes) from search by stop processing

6. Print filenames which satisfy above search conditions and are newer than "`/path/to/timestamp`"

Please note the idiomatic use of "`-prune -o`" to exclude files in the above example.

Note
For non-Debian Unix-like system, some options may not be supported by find(1). In such a case, please consider to adjust matching methods and replace "`-print0`" with "`-print`". You may need to adjust related commands too.

10.1.6   Archive media

When choosing computer data storage media for important data archive, you should be careful about their limitations. For small personal data backup, I use CD-R and DVD-R by the brand name company and store in a cool, shaded, dry, clean environment. (Tape archive media seem to be popular for professional use.)

Note
A fire-resistant safe are meant for paper documents. Most of the computer data storage media have less temperature tolerance than paper. I usually rely on multiple secure encrypted copies stored in multiple secure locations.

Optimistic storage life of archive media seen on the net (mostly from vendor info).

- 100+ years : Acid free paper with ink
- 100 years : Optical storage (CD/DVD, CD/DVD-R)
- 30 years : Magnetic storage (tape, floppy)
- 20 years : Phase change optical storage (CD-RW)

These do not count on the mechanical failures due to handling etc.

Optimistic write cycle of archive media seen on the net (mostly from vendor info).

- 250,000+ cycles : Harddisk drive
- 10,000+ cycles : Flash memory
- 1,000 cycles : CD/DVD-RW
- 1 cycles : CD/DVD-R, paper

⚠ **Caution**
Figures of storage life and write cycle here should not be used for decisions on any critical data storage. Please consult the specific product information provided by the manufacture.

Tip
Since CD/DVD-R and paper have only 1 write cycle, they inherently prevent accidental data loss by overwriting. This is advantage!

Tip
If you need fast and frequent backup of large amount of data, a hard disk on a remote host linked by a fast network connection, may be the only realistic option.

10.1.7   Removable storage device

Removable storage devices may be any one of the following.

- USB flash drive
- Hard disk drive

- Optical disc drive

- Digital camera

- Digital music player

They may be connected via any one of the following.

- USB

- IEEE 1394 / FireWire

- PC Card

Modern desktop environments such as GNOME and KDE can mount these removable devices automatically without a matching "/etc/fstab" entry.

- udisks package provides a daemon and associated utilities to mount and unmount these devices.

- D-bus creates events to initiate automatic processes.

- PolicyKit provides required privileges.

Tip
Automounted devices may have the "uhelper=" mount option which is used by umount(8).

Tip
Automounting under modern desktop environment happens only when those removable media devices are not listed in "/etc/fstab".

Mount point under modern desktop environment is chosen as "/media/<disk_label>" which can be customized by the following.

- mlabel(1) for FAT filesystem

- genisoimage(1) with "-V" option for ISO9660 filesystem

- tune2fs(1) with "-L" option for ext2/ext3/ext4 filesystem

Tip
The choice of encoding may need to be provided as mount option (see Section 8.4.6).

Tip
The use of the GUI menu to unmount a filesystem may remove its dynamically generated device node such as "/dev/sdc". If you wish to keep its device node, unmount it with the umount(8) command from the shell prompt.

---

filesystem	description of typical usage scenario
FAT12	cross platform sharing of data on the floppy disk (<32MiB)
FAT16	cross platform sharing of data on the small hard disk like device (<2GiB)
FAT32	cross platform sharing of data on the large hard disk like device (<8TiB, supported by newer than MS Windows95 OSR2)
NTFS	cross platform sharing of data on the large hard disk like device (supported natively on MS Windows NT and later version, and supported by NTFS-3G via FUSE on Linux)
ISO9660	cross platform sharing of static data on CD-R and DVD+/-R
UDF	incremental data writing on CD-R and DVD+/-R (new)
MINIX filesystem	space efficient unix file data storage on the floppy disk
ext2 filesystem	sharing of data on the hard disk like device with older Linux systems
ext3 filesystem	sharing of data on the hard disk like device with older Linux systems
ext4 filesystem	sharing of data on the hard disk like device with current Linux systems

Table 10.3: List of filesystem choices for removable storage devices with typical usage scenarios

10.1.8 Filesystem choice for sharing data

When sharing data with other system via removable storage device, you should format it with common filesystem supported by both systems. Here is a list of filesystem choices.

Tip
See Section 9.8.1 for cross platform sharing of data using device level encryption.

The FAT filesystem is supported by almost all modern operating systems and is quite useful for the data exchange purpose via removable hard disk like media.

When formatting removable hard disk like devices for cross platform sharing of data with the FAT filesystem, the following should be safe choices.

- Partitioning them with fdisk(8), cfdisk(8) or parted(8) (see Section 9.5.2) into a single primary partition and to mark it as the following.

  - Type "6" for FAT16 for media smaller than 2GB.
  - Type "c" for FAT32 (LBA) for larger media.

- Formatting the primary partition with mkfs.vfat(8) with the following.

  - Just its device name, e.g. "`/dev/sda1`" for FAT16
  - The explicit option and its device name, e.g. "`-F 32 /dev/sda1`" for FAT32

When using the FAT or ISO9660 filesystems for sharing data, the following should be the safe considerations.

- Archiving files into an archive file first using tar(1), or cpio(1) to retain the long filename, the symbolic link, the original Unix file permission and the owner information.

- Splitting the archive file into less than 2 GiB chunks with the split(1) command to protect it from the file size limitation.

- Encrypting the archive file to secure its contents from the unauthorized access.

Note
For FAT filesystems by its design, the maximum file size is (2^32 - 1) bytes = (4GiB - 1 byte). For some applications on the older 32 bit OS, the maximum file size was even smaller (2^31 - 1) bytes = (2GiB - 1 byte). Debian does not suffer the latter problem.

Note

Microsoft itself does not recommend to use FAT for drives or partitions of over 200 MB. Microsoft highlights its short comings such as inefficient disk space usage in their "Overview of FAT, HPFS, and NTFS File Systems". Of course, we should normally use the ext4 filesystem for Linux.

Tip

For more on filesystems and accessing filesystems, please read "Filesystems HOWTO".

10.1.9   Sharing data via network

When sharing data with other system via network, you should use common service. Here are some hints.

network service	description of typical usage scenario
SMB/CIFS network mounted filesystem with Samba	sharing files via "Microsoft Windows Network", see smb.conf(5) and The Official Samba 3.x.x HOWTO and Reference Guide or the `samba-doc` package
NFS network mounted filesystem with the Linux kernel	sharing files via "Unix/Linux Network", see exports(5) and Linux NFS-HOWTO
HTTP service	sharing file between the web server/client
HTTPS service	sharing file between the web server/client with encrypted Secure Sockets Layer (SSL) or Transport Layer Security (TLS)
FTP service	sharing file between the FTP server/client

Table 10.4: List of the network service to chose with the typical usage scenario

Although these filesystems mounted over network and file transfer methods over network are quite convenient for sharing data, these may be insecure. Their network connection must be secured by the following.

- Encrypt it with SSL/TLS

- Tunnel it via SSH

- Tunnel it via VPN

- Limit it behind the secure firewall

See also Section 6.10 and Section 6.11.

10.2   Backup and recovery

We all know that computers fail sometime or human errors cause system and data damages. Backup and recovery operations are the essential part of successful system administration. All possible failure modes hit you some day.

Tip

Keep your backup system simple and backup your system often. Having backup data is more important than how technically good your backup method is.

There are 3 key factors which determine actual backup and recovery policy.

1. Knowing what to backup and recover.

- Data files directly created by you: data in "~/"
- Data files created by applications used by you: data in "/var/" (except "/var/cache/", "/var/run/", and "/var/tmp/")
- System configuration files: data in "/etc/"
- Local softwares: data in "/usr/local/" or "/opt/"
- System installation information: a memo in plain text on key steps (partition, ⋯)
- Proven set of data: confirmed by experimental recovery operations in advance

2. Knowing how to backup and recover.

- Secure storage of data: protection from overwrite and system failure
- Frequent backup: scheduled backup
- Redundant backup: data mirroring
- Fool proof process: easy single command backup

3. Assessing risks and costs involved.

- Value of data when lost
- Required resources for backup: human, hardware, software, ⋯
- Failure mode and their possibility

Note
Do not back up the pseudo-filesystem contents found on /proc, /sys, /tmp, and /run (see Section 1.2.12 and Section 1.2.13). Unless you know exactly what you are doing, they are huge useless data.

As for secure storage of data, data should be at least on different disk partitions preferably on different disks and machines to withstand the filesystem corruption. Important data are best stored on a write-once media such as CD/DVD-R to prevent overwrite accidents. (See Section 9.7 for how to write to the storage media from the shell commandline. GNOME desktop GUI environment gives you easy access via menu: "Places→CD/DVD Creator".)

Note
You may wish to stop some application daemons such as MTA (see Section 6.3) while backing up data.

Note
You should pay extra care to the backup and restoration of identity related data files such as "/etc/ssh/ssh_host_dsa_key", "/etc/ssh/ssh_host_rsa_key", "~/.gnupg/*", "~/.ssh/*", "/etc/passwd", "/etc/shadow", "/etc/fetchmailrc", "popularity-contest.conf", "/etc/ppp/pap-secrets", and "/etc/exim4/passwd.client". Some of these data can not be regenerated by entering the same input string to the system.

Note
If you run a cron job as a user process, you must restore files in "/var/spool/cron/crontabs" directory and restart cron(8). See Section 9.3.14 for cron(8) and crontab(1).

---

package	popcon	size	description
dump	V:1, I:6	341	4.4 BSD dump(8) and restore(8) for ext2/ext3/ext4 filesystems
xfsdump	V:0, I:11	838	dump and restore with xfsdump(8) and xfsrestore(8) for XFS filesystem on GNU/Linux and IRIX
backupninja	V:4, I:4	329	lightweight, extensible **meta-backup** system
bacula-common	V:10, I:18	2059	Bacula: network backup, recovery and verification - common support files
bacula-client	I:5	167	Bacula: network backup, recovery and verification - client meta-package
bacula-console	V:1, I:6	68	Bacula: network backup, recovery and verification - text console
bacula-server	I:1	167	Bacula: network backup, recovery and verification - server meta-package
amanda-common	V:1, I:2	9964	Amanda: Advanced Maryland Automatic Network Disk Archiver (Libs)
amanda-client	V:1, I:2	1072	Amanda: Advanced Maryland Automatic Network Disk Archiver (Client)
amanda-server	V:0, I:0	1044	Amanda: Advanced Maryland Automatic Network Disk Archiver (Server)
backup-manager	V:1, I:2	571	command-line backup tool
backup2l	V:1, I:1	113	low-maintenance backup/restore tool for mountable media (disk based)
backuppc	V:4, I:4	2232	BackupPC is a high-performance, enterprise-grade system for backing up PCs (disk based)
duplicity	V:8, I:15	1535	(remote) incremental backup
flexbackup	V:0, I:0	242	(remote) incremental backup
rdiff-backup	V:9, I:17	704	(remote) incremental backup
restic	V:0, I:0	15987	(remote) incremental backup
rsnapshot	V:6, I:12	452	(remote) incremental backup
slbackup	V:0, I:0	146	(remote) incremental backup

Table 10.5: List of backup suite utilities

10.2.1 Backup utility suites

Here is a select list of notable backup utility suites available on the Debian system.

Backup tools have their specialized focuses.

- Mondo Rescue is a backup system to facilitate restoration of complete system quickly from backup CD/DVD etc. without going through normal system installation processes.

- Regular backups of user data can be realized by a simple script (Section 10.2.2) and cron(8).

- Bacula, Amanda, and BackupPC are full featured backup suite utilities which are focused on regular backups over network.

Basic tools described in Section 10.1.1 and Section 10.1.2 can be used to facilitate system backup via custom scripts. Such script can be enhanced by the following.

- The `restic` package enables incremental (remote) backups.

- The `rdiff-backup` package enables incremental (remote) backups.

- The `dump` package helps to archive and restore the whole filesystem incrementally and efficiently.

Tip
See files in "`/usr/share/doc/dump/`" and "Is dump really deprecated?" to learn about the `dump` package.

10.2.2 An example script for the system backup

For a personal Debian desktop system running `unstable` suite, I only need to protect personal and critical data. I reinstall system once a year anyway. Thus I see no reason to backup the whole system or to install a full featured backup utility.

I use a simple script to make a backup archive and burn it into CD/DVD using GUI. Here is an example script for this.

```
#!/bin/sh -e
# Copyright (C) 2007-2008 Osamu Aoki <osamu@debian.org>, Public Domain
BUUID=1000; USER=osamu # UID and name of a user who accesses backup files
BUDIR="/var/backups"
XDIR0=".+/Mail|.+/Desktop"
XDIR1=".+/\.thumbnails|.+/\.?Trash|.+/\.?[cC]ache|.+/\.gvfs|.+/sessions"
XDIR2=".+/CVS|.+/\.git|.+/\.svn|.+/Downloads|.+/Archive|.+/Checkout|.+/tmp"
XSFX=".+\.iso|.+\.tgz|.+\.tar\.gz|.+\.tar\.bz2|.+\.cpio|.+\.tmp|.+\.swp|.+~"
SIZE="+99M"
DATE=$(date --utc +"%Y%m%d-%H%M")
[ -d "$BUDIR" ] || mkdir -p "BUDIR"
umask 077
dpkg --get-selections \* > /var/lib/dpkg/dpkg-selections.list
debconf-get-selections > /var/cache/debconf/debconf-selections

{
find /etc /usr/local /opt /var/lib/dpkg/dpkg-selections.list \
    /var/cache/debconf/debconf-selections -xdev -print0
find /home/$USER /root -xdev -regextype posix-extended \
  -type d -regex "$XDIR0|$XDIR1" -prune -o -type f -regex "$XSFX" -prune -o \
  -type f -size  "$SIZE" -prune -o -print0
find /home/$USER/Mail/Inbox /home/$USER/Mail/Outbox -print0
find /home/$USER/Desktop  -xdev -regextype posix-extended \
  -type d -regex "$XDIR2" -prune -o -type f -regex "$XSFX" -prune -o \
  -type f -size  "$SIZE" -prune -o -print0
} | cpio -ov --null -O $BUDIR/BU$DATE.cpio
chown $BUUID $BUDIR/BU$DATE.cpio
touch $BUDIR/backup.stamp
```

This is meant to be a script example executed from root.

I expect you to change and execute this as follows.

- Edit this script to cover all your important data (see Section 10.1.5 and Section 10.2).

- Replace "find ···-print0" with "find ···-newer $BUDIR/backup.stamp -print0" to make a incremental backup.

- Transfer backup files to the remote host using scp(1) or rsync(1) or burn them to CD/DVD for extra data security. (I use GNOME desktop GUI for burning CD/DVD. See Section 12.1.8 for extra redundancy.)

Keep it simple!

Tip
You can recover debconf configuration data with "debconf-set-selections debconf-selections" and dpkg selection data with "dpkg --set-selection <dpkg-selections.list".

10.2.3   A copy script for the data backup

For the set of data under a directory tree, the copy with "cp -a" provides the normal backup.

For the set of large non-overwritten static data under a directory tree such as the one under the "/var/cache/apt/packages/" directory, hardlinks with "cp -al" provide an alternative to the normal backup with efficient use of the disk space.

Here is a copy script, which I named as bkup, for the data backup. This script copies all (non-VCS) files under the current directory to the dated directory on the parent directory or on a remote host.

```
#!/bin/sh -e
# Copyright (C) 2007-2008 Osamu Aoki <osamu@debian.org>, Public Domain
fdot(){ find . -type d \( -iname ".?*" -o -iname "CVS" \) -prune -o -print0;}
fall(){ find . -print0;}
mkdircd(){ mkdir -p "$1";chmod 700 "$1";cd "$1">/dev/null;}
FIND="fdot";OPT="-a";MODE="CPIOP";HOST="localhost";EXTP="$(hostname -f)"
BKUP="$(basename $(pwd)).bkup";TIME="$(date  +%Y%m%d-%H%M%S)";BU="$BKUP/$TIME"
while getopts gcCsStrlLaAxe:h:T f; do case $f in
g)  MODE="GNUCP";; # cp (GNU)
c)  MODE="CPIOP";; # cpio -p
C)  MODE="CPIOI";; # cpio -i
s)  MODE="CPIOSSH";; # cpio/ssh
t)  MODE="TARSSH";; # tar/ssh
r)  MODE="RSYNCSSH";; # rsync/ssh
l)  OPT="-alv";; # hardlink (GNU cp)
L)  OPT="-av";;  # copy (GNU cp)
a)  FIND="fall";; # find all
A)  FIND="fdot";; # find non CVS/ .???/
x)  set -x;; # trace
e)  EXTP="${OPTARG}";; # hostname -f
h)  HOST="${OPTARG}";; # user@remotehost.example.com
T)  MODE="TEST";; # test find mode
\?) echo "use -x for trace."
esac; done
shift $(expr $OPTIND - 1)
if [ $# -gt 0 ]; then
  for x in $@; do cp $OPT $x $x.$TIME; done
elif [ $MODE = GNUCP ]; then
  mkdir -p "../$BU";chmod 700 "../$BU";cp $OPT . "../$BU/"
elif [ $MODE = CPIOP ]; then
  mkdir -p "../$BU";chmod 700 "../$BU"
  $FIND|cpio --null --sparse -pvd ../$BU
elif [ $MODE = CPIOI ]; then
```

```
  $FIND|cpio -ov --null | ( mkdircd "../$BU"&&cpio -i )
elif [ $MODE = CPIOSSH ]; then
  $FIND|cpio -ov --null|ssh -C $HOST "( mkdircd \"$EXTP/$BU\"&&cpio -i )"
elif [ $MODE = TARSSH ]; then
  (tar cvf - . )|ssh -C $HOST "( mkdircd \"$EXTP/$BU\"&& tar xvfp - )"
elif [ $MODE = RSYNCSSH ]; then
  rsync -aHAXSv ./ "${HOST}:${EXTP}-${BKUP}-${TIME}"
else
  echo "Any other idea to backup?"
  $FIND |xargs -0 -n 1 echo
fi
```

This is meant to be command examples. Please read script and edit it by yourself before using it.

Tip

I keep this `bkup` in my "`/usr/local/bin/`" directory. I issue this `bkup` command without any option in the working directory whenever I need a temporary snapshot backup.

Tip

For making snapshot history of a source file tree or a configuration file tree, it is easier and space efficient to use git(7) (see Section 10.6.5).

10.3 Data security infrastructure

The data security infrastructure is provided by the combination of data encryption tool, message digest tool, and signature tool.

package	popcon	size	command	description
gnupg	V:698, I:998	696	gpg(1)	GNU Privacy Guard - OpenPGP encryption and signing tool
gpgv	V:873, I:999	809	gpgv(1)	GNU Privacy Guard - signature verification tool
paperkey	V:0, I:4	55	paperkey(1)	extract just the secret information out of OpenPGP secret keys
cryptsetup	V:34, I:79	376	cryptsetup(8), …	utilities for dm-crypto block device encryption supporting LUKS
ecryptfs-utils	V:6, I:9	396	ecryptfs(7), …	utilities for ecryptfs stacked filesystem encryption
coreutils	V:878, I:999	15016	md5sum(1)	compute and check MD5 message digest
coreutils	V:878, I:999	15016	sha1sum(1)	compute and check SHA1 message digest
openssl	V:794, I:991	1303	openssl(1ssl)	compute message digest with "openssl dgst" (OpenSSL)

Table 10.6: List of data security infrastructure tools

See Section 9.8 on dm-crypto and ecryptfs which implement automatic data encryption infrastructure via Linux kernel modules.

10.3.1 Key management for GnuPG

Here are GNU Privacy Guard commands for the basic key management.

Here is the meaning of the trust code.

The following uploads my key "1DD8D791" to the popular keyserver "hkp://keys.gnupg.net".

command	description
gpg --gen-key	generate a new key
gpg --gen-revoke my_user_ID	generate revoke key for my_user_ID
gpg --edit-key user_ID	edit key interactively, "help" for help
gpg -o file --export	export all keys to file
gpg --import file	import all keys from file
gpg --send-keys user_ID	send key of user_ID to keyserver
gpg --recv-keys user_ID	recv. key of user_ID from keyserver
gpg --list-keys user_ID	list keys of user_ID
gpg --list-sigs user_ID	list sig. of user_ID
gpg --check-sigs user_ID	check sig. of user_ID
gpg --fingerprint user_ID	check fingerprint of user_ID
gpg --refresh-keys	update local keyring

Table 10.7: List of GNU Privacy Guard commands for the key management

code	description of trust
-	no owner trust assigned / not yet calculated
e	trust calculation failed
q	not enough information for calculation
n	never trust this key
m	marginally trusted
f	fully trusted
u	ultimately trusted

Table 10.8: List of the meaning of the trust code

```
$ gpg --keyserver hkp://keys.gnupg.net --send-keys 1DD8D791
```

A good default keyserver set up in "~/.gnupg/gpg.conf" (or old location "~/.gnupg/options") contains the following.

```
keyserver hkp://keys.gnupg.net
```

The following obtains unknown keys from the keyserver.

```
$ gpg --list-sigs --with-colons | grep '^sig.*\[User ID not found\]' |\
  cut -d ':' -f 5| sort | uniq | xargs gpg --recv-keys
```

There was a bug in OpenPGP Public Key Server (pre version 0.9.6) which corrupted key with more than 2 sub-keys. The newer gnupg (>1.2.1-2) package can handle these corrupted subkeys. See gpg(1) under "--repair-pks-subkey-bug" option.

10.3.2   Using GnuPG on files

Here are examples for using GNU Privacy Guard commands on files.

10.3.3   Using GnuPG with Mutt

Add the following to "~/.muttrc" to keep a slow GnuPG from automatically starting, while allowing it to be used by typing "S" at the index menu.

```
macro index S ":toggle pgp_verify_sig\n"
set pgp_verify_sig=no
```

command	description
`gpg -a -s file`	sign file into ASCII armored file.asc
`gpg --armor --sign file`	,,
`gpg --clearsign file`	clear-sign message
`gpg --clearsign file|mail foo@example.org`	mail a clear-signed message to `foo@example.org`
`gpg --clearsign --not-dash-escaped patchfile`	clear-sign patchfile
`gpg --verify file`	verify clear-signed file
`gpg -o file.sig -b file`	create detached signature
`gpg -o file.sig --detach-sig file`	,,
`gpg --verify file.sig file`	verify file with file.sig
`gpg -o crypt_file.gpg -r name -e file`	public-key encryption intended for name from file to binary crypt_file.gpg
`gpg -o crypt_file.gpg --recipient name --encrypt file`	,,
`gpg -o crypt_file.asc -a -r name -e file`	public-key encryption intended for name from file to ASCII armored crypt_file.asc
`gpg -o crypt_file.gpg -c file`	symmetric encryption from file to crypt_file.gpg
`gpg -o crypt_file.gpg --symmetric file`	,,
`gpg -o crypt_file.asc -a -c file`	symmetric encryption intended for name from file to ASCII armored crypt_file.asc
`gpg -o file -d crypt_file.gpg -r name`	decryption
`gpg -o file --decrypt crypt_file.gpg`	,,

Table 10.9: List of GNU Privacy Guard commands on files

10.3.4 Using GnuPG with Vim

The gnupg plugin let you run GnuPG transparently for files with extension ".gpg", ".asc", and ".ppg".

```
# aptitude install vim-scripts vim-addon-manager
$ vim-addons install gnupg
```

10.3.5 The MD5 sum

md5sum(1) provides utility to make a digest file using the method in rfc1321 and verifying each file with it.

```
$ md5sum foo bar >baz.md5
$ cat baz.md5
d3b07384d113edec49eaa6238ad5ff00  foo
c157a79031e1c40f85931829bc5fc552  bar
$ md5sum -c baz.md5
foo: OK
bar: OK
```

> **Note**
> The computation for the MD5 sum is less CPU intensive than the one for the cryptographic signature by GNU Privacy Guard (GnuPG). Usually, only the top level digest file is cryptographically signed to ensure data integrity.

10.4 Source code merge tools

There are many merge tools for the source code. Following commands caught my eyes.

10.4.1 Extracting differences for source files

The following procedures extract differences between two source files and create unified diff files "file.patch0" or "file.patch1" depending on the file location.

```
$ diff -u file.old file.new > file.patch0
$ diff -u old/file new/file > file.patch1
```

10.4.2 Merging updates for source files

The diff file (alternatively called patch file) is used to send a program update. The receiving party applies this update to another file by the following.

```
$ patch -p0 file < file.patch0
$ patch -p1 file < file.patch1
```

10.4.3 Updating via 3-way-merge

If you have three versions of a source code, you can perform 3-way-merge effectively using diff3(1) by the following.

```
$ diff3 -m file.mine file.old file.yours > file
```

package	popcon	size	command	description
diffutils	V:860, I:983	1394	diff(1)	compare files line by line
diffutils	V:860, I:983	1394	diff3(1)	compare and merges three files line by line
vim	V:121, I:390	2660	vimdiff(1)	compare 2 files side by side in vim
patch	V:122, I:818	235	patch(1)	apply a diff file to an original
dpatch	V:1, I:15	191	dpatch(1)	manage series of patches for Debian package
diffstat	V:19, I:187	70	diffstat(1)	produce a histogram of changes by the diff
patchutils	V:18, I:179	223	combinediff(1)	create a cumulative patch from two incremental patches
patchutils	V:18, I:179	223	dehtmldiff(1)	extract a diff from an HTML page
patchutils	V:18, I:179	223	filterdiff(1)	extract or excludes diffs from a diff file
patchutils	V:18, I:179	223	fixcvsdiff(1)	fix diff files created by CVS that patch(1) mis-interprets
patchutils	V:18, I:179	223	flipdiff(1)	exchange the order of two patches
patchutils	V:18, I:179	223	grepdiff(1)	show which files are modified by a patch matching a regex
patchutils	V:18, I:179	223	interdiff(1)	show differences between two unified diff files
patchutils	V:18, I:179	223	lsdiff(1)	show which files are modified by a patch
patchutils	V:18, I:179	223	recountdiff(1)	recompute counts and offsets in unified context diffs
patchutils	V:18, I:179	223	rediff(1)	fix offsets and counts of a hand-edited diff
patchutils	V:18, I:179	223	splitdiff(1)	separate out incremental patches
patchutils	V:18, I:179	223	unwrapdiff(1)	demangle patches that have been word-wrapped
wiggle	V:0, I:0	166	wiggle(1)	apply rejected patches
quilt	V:4, I:41	711	quilt(1)	manage series of patches
meld	V:16, I:41	3116	meld(1)	compare and merge files (GTK)
dirdiff	V:0, I:2	144	dirdiff(1)	display differences and merge changes between directory trees
docdiff	V:0, I:0	573	docdiff(1)	compare two files word by word / char by char
imediff2	V:0, I:0	34	imediff2(1)	interactive full screen 2-way merge tool
makepatch	V:0, I:0	102	makepatch(1)	generate extended patch files
makepatch	V:0, I:0	102	applypatch(1)	apply extended patch files
wdiff	V:5, I:78	643	wdiff(1)	display word differences between text files

Table 10.10: List of source code merge tools

10.5   Version control systems

Here is a summary of the version control systems (VCS) on the Debian system.

Note
If you are new to VCS systems, you should start learning with **Git**, which is growing fast in popularity.

---

package	popcon	size	tool	VCS type	comment
cssc	V:0, I:2	1979	CSSC	local	clone of the Unix SCCS (deprecated)
rcs	V:4, I:22	539	RCS	local	"Unix SCCS done right"
cvs	V:6, I:55	4597	CVS	remote	previous standard remote VCS
subversion	V:31, I:144	4678	Subversion	remote	"CVS done right", the new de facto standard remote VCS
git	V:279, I:436	33191	Git	distributed	fast DVCS in C (used by the Linux kernel and others)
mercurial	V:12, I:61	552	Mercurial	distributed	DVCS in Python and some C
bzr	V:4, I:22	73	Bazaar	distributed	DVCS influenced by tla written in Python (used by Ubuntu)
darcs	V:0, I:8	31349	Darcs	distributed	DVCS with smart algebra of patches (slow)
tla	V:0, I:7	1011	GNU arch	distributed	DVCS mainly by Tom Lord (Historic)
monotone	V:0, I:0	5815	Monotone	distributed	DVCS in C++
tkcvs	V:0, I:1	1498	CVS, ···	remote	GUI display of VCS (CVS, Subversion, RCS) repository tree
gitk	V:8, I:49	1483	Git	distributed	GUI display of VCS (Git) repository tree

Table 10.11: List of version control system tools

VCS is sometimes known as revision control system (RCS), or software configuration management (SCM).

Distributed VCS such as Git is the tool of choice these days. CVS and Subversion may still be useful to join some existing open source program activities.

Debian provides free VCS services via Debian Salsa service. It supports practically all VCSs. Its documentation can be found at https://wiki.debian.org/Salsa .

There are few basics for creating a shared access VCS archive.

- Use "umask 002" (see Section 1.2.4)

- Make all VCS archive files belonging to a pertinent group

- Enable set group ID on all VCS archive directories (BSD-like file creation scheme, see Section 1.2.3)

- Make user sharing the VCS archive belonging to the group

10.5.1   Comparison of VCS commands

Here is an oversimplified comparison of native VCS commands to provide the big picture. The typical command sequence may require options and arguments.

⚠️ **Caution**
Invoking a git subcommand directly as "git-xyz" from the command line has been deprecated since early 2006.

Git	CVS	Subversion	function
`git init`	`cvs init`	`svn create`	create the (local) repository
-	`cvs login`	-	login to the remote repository
`git clone`	`cvs co`	`svn co`	check out the remote repository as the working tree
`git pull`	`cvs up`	`svn up`	update the working tree by merging the remote repository
`git add .`	`cvs add`	`svn add`	add file(s) in the working tree to the VCS
`git rm`	`cvs rm`	`svn rm`	remove file(s) in working tree from the VCS
-	`cvs ci`	`svn ci`	commit changes to the remote repository
`git commit -a`	-	-	commit changes to the local repository
`git push`	-	-	update the remote repository by the local repository
`git status`	`cvs status`	`svn status`	display the working tree status from the VCS
`git diff`	`cvs diff`	`svn diff`	diff <reference_repository> <working_tree>
`git repack -a -d; git prune`	-	-	repack the local repository into single pack
`gitk`	`tkcvs`	`tkcvs`	GUI display of VCS repository tree

Table 10.12: Comparison of native VCS commands

Tip
If there is a executable file `git-foo` in the path specified by $PATH, entring "`git foo`" without hyphen to the command line invokes this `git-foo`. This is a feature of the `git` command.

Tip
GUI tools such as tkcvs(1) and gitk(1) really help you with tracking revision history of files. The web interface provided by many public archives for browsing their repositories is also quite useful, too.

Tip
Git can work directly with different VCS repositories such as ones provided by CVS and Subversion, and provides the local repository for local changes with `git-cvs` and `git-svn` packages. See git for CVS users, and Section 10.6.4.

Tip
Git has commands which have no equivalents in CVS and Subversion: "fetch", "rebase", "cherry-pick", ···

10.6 Git

Git can do everything for both local and remote source code management. This means that you can record the source code changes without needing network connectivity to the remote repository.

10.6.1 Configuration of Git client

You may wish to set several global configuration in "`~/.gitconfig`" such as your name and email address used by Git by the following.

```
$ git config --global user.name "Name Surname"
$ git config --global user.email yourname@example.com
```

If you are too used to CVS or Subversion commands, you may wish to set several command aliases by the following.

```
$ git config --global alias.ci "commit -a"
$ git config --global alias.co checkout
```

You can check your global configuration by the following.

```
$ git config --global --list
```

10.6.2 Git references

See the following.

- manpage: git(1) (`/usr/share/doc/git-doc/git.html`)
- Git User's Manual (`/usr/share/doc/git-doc/user-manual.html`)
- A tutorial introduction to git (`/usr/share/doc/git-doc/gittutorial.html`)
- A tutorial introduction to git: part two (`/usr/share/doc/git-doc/gittutorial-2.html`)
- Everyday GIT With 20 Commands Or So (`/usr/share/doc/git-doc/everyday.html`)
- git for CVS users (`/usr/share/doc/git-doc/gitcvs-migration.html`)
  - This also describes how to set up server like CVS and extract old data from CVS into Git.
- Other git resources available on the web
  - Git - SVN Crash Course
  - Git Magic (`/usr/share/doc/gitmagic/html/index.html`)

git-gui(1) and gitk(1) commands make using Git very easy.

> **Warning**
> Do not use the tag string with spaces in it even if some tools such as gitk(1) allow you to use it. It may choke some other `git` commands.

10.6.3 Git commands

Even if your upstream uses different VCS, it may be a good idea to use git(1) for local activity since you can manage your local copy of source tree without the network connection to the upstream. Here are some packages and commands used with git(1).

> **Tip**
> With git(1), you work on a local branch with many commits and use something like "`git rebase -i master`" to reorganize change history later. This enables you to make clean change history. See git-rebase(1) and git-cherry-pick(1).

package	popcon	size	command	description
git-doc	I:19	10733	N/A	official documentation for Git
gitmagic	I:1	719	N/A	"Git Magic", easier to understand guide for Git
git	V:279, I:436	33191	git(7)	Git, the fast, scalable, distributed revision control system
gitk	V:8, I:49	1483	gitk(1)	GUI Git repository browser with history
git-gui	V:2, I:28	2210	git-gui(1)	GUI for Git (No history)
git-svn	V:2, I:27	995	git-svnimport(1)	import the data out of Subversion into Git
git-svn	V:2, I:27	995	git-svn(1)	provide bidirectional operation between the Subversion and Git
git-cvs	V:0, I:13	1116	git-cvsimport(1)	import the data out of CVS into Git
git-cvs	V:0, I:13	1116	git-cvsexportcommit(1)	export a commit to a CVS checkout from Git
git-cvs	V:0, I:13	1116	git-cvsserver(1)	CVS server emulator for Git
git-email	V:0, I:13	800	git-send-email(1)	send a collection of patches as email from the Git
stgit	V:0, I:1	1535	stg(1)	quilt on top of git (Python)
git-buildpackage	V:2, I:12	3863	git-buildpackage(1)	automate the Debian packaging with the Git
guilt	V:0, I:0	146	guilt(7)	quilt on top of git (SH/AWK/SED/···)

Table 10.13: List of git related packages and commands

Tip
When you want to go back to a clean working directory without loosing the current state of the working directory, you can use "`git stash`". See git-stash(1).

10.6.4 Git for the Subversion repository

You can check out a Subversion repository at "`svn+ssh://svn.example.org/project/module/trunk`" to a local Git repository at "`./dest`" and commit back to the Subversion repository. E.g.:

```
$ git svn clone -s -rHEAD svn+ssh://svn.example.org/project dest
$ cd dest
... make changes
$ git commit -a
... keep working locally with git
$ git svn dcommit
```

Tip
The use of "`-rHEAD`" enables us to avoid cloning entire historical contents from the Subversion repository.

10.6.5 Git for recording configuration history

You can manually record chronological history of configuration using Git tools. Here is a simple example for your practice to record "`/etc/apt/`" contents.

```
$ cd /etc/apt/
```

```
$ sudo git init
$ sudo chmod 700 .git
$ sudo git add .
$ sudo git commit -a
```

Commit configuration with description.

Make modification to the configuration files.

```
$ cd /etc/apt/
$ sudo git commit -a
```

Commit configuration with description and continue your life.

```
$ cd /etc/apt/
$ sudo gitk --all
```

You have full configuration history with you.

> **Note**
> sudo(8) is needed to work with any file permissions of configuration data. For user configuration data, you may
> skip `sudo`.

> **Note**
> The "`chmod 700 .git`" command in the above example is needed to protect archive data from unauthorized read
> access.

> **Tip**
> For more complete setup for recording configuration history, please look for the `etckeeper` package: Section 9.2.10.

10.7 CVS

CVS is an **older** version control system before Subversion and Git.

> ⚠ **Caution**
> Many URLs found in the below examples for CVS don't exist any more.

See the following.

- cvs(1)

- "/usr/share/doc/cvs/html-cvsclient"

- "/usr/share/doc/cvs/html-info"

- "/usr/share/doc/cvsbook"

- "info cvs"

10.7.1   Configuration of CVS repository

The following configuration allows commits to the CVS repository only by a member of the "src" group, and administration of CVS only by a member of the "staff" group, thus reducing the chance of shooting oneself.

```
# cd /var/lib; umask 002; mkdir cvs
# export CVSROOT=/srv/cvs/project
# cd $CVSROOT
# chown root:src .
# chmod 2775 .
# cvs -d $CVSROOT init
# cd CVSROOT
# chown -R root:staff .
# chmod 2775 .
# touch val-tags
# chmod 664 history val-tags
# chown root:src history val-tags
```

Tip
You may restrict creation of new project by changing the owner of "$CVSROOT" directory to "root:staff" and its permission to "3775".

10.7.2   Local access to CVS

The default CVS repository is pointed by "$CVSROOT". The following sets up "$CVSROOT" for the local access.

```
$ export CVSROOT=/srv/cvs/project
```

10.7.3   Remote access to CVS with pserver

Many public CVS servers provide read-only remote access to them with account name "anonymous" via pserver service. For example, Debian web site contents were maintained by webwml project via CVS at Debian alioth service. The following was used to set up "$CVSROOT" for the remote access to this old CVS repository.

```
$ export CVSROOT=:pserver:anonymous@anonscm.debian.org:/cvs/webwml
$ cvs login
```

Note
Since pserver is prone to eavesdropping attack and insecure, write access is usually disable by server administrators.

10.7.4   Remote access to CVS with ssh

The following was used to set up "$CVS_RSH" and "$CVSROOT" for the remote access to the old CVS repository by webwml project with SSH.

```
$ export CVS_RSH=ssh
$ export CVSROOT=:ext:account@cvs.alioth.debian.org:/cvs/webwml
```

You can also use public key authentication for SSH which eliminates the remote password prompt.

10.7.5   Importing a new source to CVS

Create a new local source tree location at "`~/path/to/module1`" by the following.

```
$ mkdir -p ~/path/to/module1; cd ~/path/to/module1
```

Populate a new local source tree under "`~/path/to/module1`" with files.

Import it to CVS with the following parameters.

- Module name: "`module1`"

- Vendor tag: "`Main-branch`" (tag for the entire branch)

- Release tag: "`Release-initial`" (tag for a specific release)

```
$ cd ~/path/to/module1
$ cvs import -m "Start module1" module1 Main-branch Release-initial
$ rm -Rf . # optional
```

10.7.6   File permissions in CVS repository

CVS does not overwrite the current repository file but replaces it with another one. Thus, write permission to the repository directory is critical. For every new module for "`module1`" in repository at "`/srv/cvs/project`", run the following to ensure this condition if needed.

```
# cd /srv/cvs/project
# chown -R root:src module1
# chmod -R ug+rwX   module1
# chmod    2775      module1
```

10.7.7   Work flow of CVS

Here is an example of typical work flow using CVS.

Check all available modules from CVS project pointed by "`$CVSROOT`" by the following.

```
$ cvs rls
CVSROOT
module1
module2
...
```

Checkout "`module1`" to its default directory "`./module1`" by the following.

```
$ cd ~/path/to
$ cvs co module1
$ cd module1
```

Make changes to the content as needed.

Check changes by making "`diff -u [repository] [local]`" equivalent by the following.

```
$ cvs diff -u
```

You find that you broke some file "`file_to_undo`" severely but other files are fine.

Overwrite "`file_to_undo`" file with the clean copy from CVS by the following.

```
$ cvs up -C file_to_undo
```

Save the updated local source tree to CVS by the following.

```
$ cvs ci -m "Describe change"
```

Create and add "file_to_add" file to CVS by the following.

```
$ vi file_to_add
$ cvs add file_to_add
$ cvs ci -m "Added file_to_add"
```

Merge the latest version from CVS by the following.

```
$ cvs up -d
```

Watch out for lines starting with "C filename" which indicates conflicting changes.

Look for unmodified code in ".#filename.version".

Search for "<<<<<<<" and ">>>>>>>" in files for conflicting changes.

Edit files to fix conflicts as needed.

Add a release tag "Release-1" by the following.

```
$ cvs ci -m "last commit for Release-1"
$ cvs tag Release-1
```

Edit further.

Remove the release tag "Release-1" by the following.

```
$ cvs tag -d Release-1
```

Check in changes to CVS by the following.

```
$ cvs ci -m "real last commit for Release-1"
```

Re-add the release tag "Release-1" to updated CVS HEAD of main by the following.

```
$ cvs tag Release-1
```

Create a branch with a sticky branch tag "Release-initial-bugfixes" from the original version pointed by the tag "Release-initial" and check it out to "~/path/to/old" directory by the following.

```
$ cvs rtag -b -r Release-initial Release-initial-bugfixes module1
$ cd ~/path/to
$ cvs co -r Release-initial-bugfixes -d old module1
$ cd old
```

Tip
Use "-D 2005-12-20" (ISO 8601 date format) instead of "-r Release-initial" to specify particular date as the branch point.

Work on this local source tree having the sticky tag "Release-initial-bugfixes" which is based on the original version.

Work on this branch by yourself ⋯until someone else joins to this "Release-initial-bugfixes" branch.

Sync with files modified by others on this branch while creating new directories as needed by the following.

```
$ cvs up -d
```

Edit files to fix conflicts as needed.

Check in changes to CVS by the following.

```
$ cvs ci -m "checked into this branch"
```

Update the local tree by HEAD of main while removing sticky tag ("-A") and without keyword expansion ("-kk") by the following.

```
$ cvs up -d -kk -A
```

Update the local tree (content = HEAD of main) by merging from the "Release-initial-bugfixes" branch and without keyword expansion by the following.

```
$ cvs up -d -kk -j Release-initial-bugfixes
```

Fix conflicts with editor.

Check in changes to CVS by the following.

```
$ cvs ci -m "merged Release-initial-bugfixes"
```

Make archive by the following.

```
$ cd ..
$ mv old old-module1-bugfixes
$ tar -cvzf old-module1-bugfixes.tar.gz old-module1-bugfixes
$ rm -rf old-module1-bugfixes
```

Tip
"cvs up" command can take "-d" option to create new directories and "-P" option to prune empty directories.

Tip
You can checkout only a sub directory of "module1" by providing its name as "cvs co module1/subdir".

---

option	meaning
-n	dry run, no effect
-t	display messages showing steps of cvs activity

Table 10.14: Notable options for CVS commands (use as first argument(s) to cvs(1))

10.7.8   Latest files from CVS

To get the latest files from CVS, use "tomorrow" by the following.

```
$ cvs ex -D tomorrow module_name
```

10.7.9   Administration of CVS

Add module alias "`mx`" to a CVS project (local server) by the following.

```
$ export CVSROOT=/srv/cvs/project
$ cvs co CVSROOT/modules
$ cd CVSROOT
$ echo "mx -a module1" >>modules
$ cvs ci -m "Now mx is an alias for module1"
$ cvs release -d .
```

Now, you can check out "`module1`" (alias: "`mx`") from CVS to "`new`" directory by the following.

```
$ cvs co -d new mx
$ cd new
```

Note
In order to perform above procedure, you should have appropriate file permissions.

10.7.10   Execution bit for CVS checkout

When you checkout files from CVS, their execution permission bit is retained.

Whenever you see execution permission problems in a checked out file, e.g. "`filename`", change its permission in the corresponding CVS repository by the following to fix it.

```
# chmod ugo-x filename
```

10.8   Subversion

Subversion is an **older** version control system before Git but after CVS. It lacks tagging and branching features found in CVS and Git.

You need to install `subversion`, `libapache2-mod-svn` and `subversion-tools` packages to set up a Subversion server.

10.8.1   Configuration of Subversion repository

Currently, the `subversion` package does not set up a repository, so one must set it up manually. One possible location for a repository is in "`/srv/svn/project`".

Create a directory by the following.

```
# mkdir -p        /srv/svn/project
```

Create the repository database by the following.

```
# svnadmin create /srv/svn/project
```

10.8.2   Access to Subversion via Apache2 server

If you only access Subversion repository via Apache2 server, you just need to make the repository only writable by the WWW server by the following.

```
# chown -R www-data:www-data /srv/svn/project
```

Add (or uncomment) the following in "`/etc/apache2/mods-available/dav_svn.conf`" to allow access to the repository via user authentication.

```
<Location /project>
  DAV svn
  SVNPath /srv/svn/project
  AuthType Basic
  AuthName "Subversion repository"
  AuthUserFile /etc/subversion/passwd
<LimitExcept GET PROPFIND OPTIONS REPORT>
    Require valid-user
</LimitExcept>
</Location>
```

Create a user authentication file with the command by the following.

```
# htpasswd2 -c /etc/subversion/passwd some-username
```

Restart Apache2.

Your new Subversion repository is accessible at URL "`http://localhost/project`" and "`http://example.com/project`" from svn(1) (assuming your URL of web server is "`http://example.com/`").

10.8.3   Local access to Subversion by group

The following sets up Subversion repository for the local access by a group, e.g. `project`.

```
# chmod  2775      /srv/svn/project
# chown -R root:src /srv/svn/project
# chmod -R ug+rwX   /srv/svn/project
```

Your new Subversion repository is group accessible at URL "`file:///localhost/srv/svn/project`" or "`file:///srv/svn/project`" from svn(1) for local users belonging to `project` group. You must run commands, such as `svn`, `svnserve`, `svnlook`, and `svnadmin` under "`umask 002`" to ensure group access.

10.8.4   Remote access to Subversion via SSH

A group accessible Subversion repository is at URL "`example.com:/srv/svn/project`" for SSH, you can access it from svn(1) at URL "`svn+ssh://example.com:/srv/svn/project`".

10.8.5   Subversion directory structure

Many projects uses directory tree similar to the following for Subversion to compensate its lack of branches and tags.

```
  ----- module1
    |   |-- branches
    |   |-- tags
    |   |   |-- release-1.0
    |   |   '-- release-2.0
    |   |
    |   '-- trunk
```

```
    |        |-- file1
    |        |-- file2
    |        '-- file3
    |
    '-- module2
```

Tip
You must use "svn copy ···" command to mark branches and tags. This ensures Subversion to record modification history of files properly and saves storage spaces.

10.8.6   Importing a new source to Subversion

Create a new local source tree location at "~/path/to/module1" by the following.

```
$ mkdir -p ~/path/to/module1; cd ~/path/to/module1
```

Populate a new local source tree under "~/path/to/module1" with files.

Import it to Subversion with the following parameters.

- Module name: "module1"

- Subversion site URL: "file:///srv/svn/project"

- Subversion directory: "module1/trunk"

- Subversion tag: "module1/tags/Release-initial"

```
$ cd ~/path/to/module1
$ svn import file:///srv/svn/project/module1/trunk -m "Start module1"
$ svn cp file:///srv/svn/project/module1/trunk file:///srv/svn/project/module1/tags/Release ←
    -initial
```

Alternatively, by the following.

```
$ svn import ~/path/to/module1 file:///srv/svn/project/module1/trunk -m "Start module1"
$ svn cp file:///srv/svn/project/module1/trunk file:///srv/svn/project/module1/tags/Release ←
    -initial
```

Tip
You can replace URLs such as "file:///···" by any other URL formats such as "http://···" and "svn+ssh:// ···".

10.8.7   Work flow of Subversion

Here is an example of typical work flow using Subversion with its native client.

Tip
Client commands offered by the git-svn package may offer alternative work flow of Subversion using the git command. See Section 10.6.4.

Check all available modules from Subversion project pointed by URL "file:///srv/svn/project" by the following.

```
$ svn list file:///srv/svn/project
module1
module2
...
```

Checkout "module1/trunk" to a directory "module1" by the following.

```
$ cd ~/path/to
$ svn co file:///srv/svn/project/module1/trunk module1
$ cd module1
```

Make changes to the content as needed.

Check changes by making "diff -u [repository] [local]" equivalent by the following.

```
$ svn diff
```

You find that you broke some file "file_to_undo" severely but other files are fine.

Overwrite "file_to_undo" file with the clean copy from Subversion by the following.

```
$ svn revert file_to_undo
```

Save the updated local source tree to Subversion by the following.

```
$ svn ci -m "Describe change"
```

Create and add "file_to_add" file to Subversion by the following.

```
$ vi file_to_add
$ svn add file_to_add
$ svn ci -m "Added file_to_add"
```

Merge the latest version from Subversion by the following.

```
$ svn up
```

Watch out for lines starting with "C filename" which indicates conflicting changes.

Look for unmodified code in, e.g., "filename.r6", "filename.r9", and "filename.mine".

Search for "<<<<<<<" and ">>>>>>>" in files for conflicting changes.

Edit files to fix conflicts as needed.

Add a release tag "Release-1" by the following.

```
$ svn ci -m "last commit for Release-1"
$ svn cp file:///srv/svn/project/module1/trunk file:///srv/svn/project/module1/tags/Release ←
    -1
```

Edit further.

Remove the release tag "Release-1" by the following.

```
$ svn rm file:///srv/svn/project/module1/tags/Release-1
```

Check in changes to Subversion by the following.

```
$ svn ci -m "real last commit for Release-1"
```

Re-add the release tag "Release-1" from updated Subversion HEAD of trunk by the following.

```
$ svn cp file:///srv/svn/project/module1/trunk file:///srv/svn/project/module1/tags/Release ←
    -1
```

Create a branch with a path "module1/branches/Release-initial-bugfixes" from the original version pointed by the path "module1/tags/Release-initial" and check it out to "~/path/to/old" directory by the following.

```
$ svn cp file:///srv/svn/project/module1/tags/Release-initial file:///srv/svn/project/ ←
    module1/branches/Release-initial-bugfixes
$ cd ~/path/to
$ svn co file:///srv/svn/project/module1/branches/Release-initial-bugfixes old
$ cd old
```

Tip
Use "module1/trunk@{2005-12-20}" (ISO 8601 date format) instead of "module1/tags/Release-initial" to specify particular date as the branch point.

Work on this local source tree pointing to branch "Release-initial-bugfixes" which is based on the original version.

Work on this branch by yourself ⋯until someone else joins to this "Release-initial-bugfixes" branch.

Sync with files modified by others on this branch by the following.

```
$ svn up
```

Edit files to fix conflicts as needed.

Check in changes to Subversion by the following.

```
$ svn ci -m "checked into this branch"
```

Update the local tree with HEAD of trunk by the following.

```
$ svn switch file:///srv/svn/project/module1/trunk
```

Update the local tree (content = HEAD of trunk) by merging from the "Release-initial-bugfixes" branch by the following.

```
$ svn merge file:///srv/svn/project/module1/branches/Release-initial-bugfixes
```

Fix conflicts with editor.

Check in changes to Subversion by the following.

```
$ svn ci -m "merged Release-initial-bugfixes"
```

Make archive by the following.

```
$ cd ..
$ mv old old-module1-bugfixes
$ tar -cvzf old-module1-bugfixes.tar.gz old-module1-bugfixes
$ rm -rf old-module1-bugfixes
```

Tip
You can replace URLs such as "file:///⋯" by any other URL formats such as "http://⋯" and "svn+ssh:// ⋯".

Tip
You can checkout only a sub directory of "module1" by providing its name as "svn co file:///srv/svn/project/module1/trunk/subdir module1/subdir", etc.

---

option	meaning
--dry-run	dry run, no effect
-v	display detail messages of svn activity

Table 10.15: Notable options for Subversion commands (use as first argument(s) to svn(1))

Chapter 11

Data conversion

Tools and tips for converting data formats on the Debian system are described.

Standard based tools are in very good shape but support for proprietary data formats are limited.

11.1 Text data conversion tools

Following packages for the text data conversion caught my eyes.

package	popcon	size	keyword	description
libc6	V:930, I:999	11401	charset	text encoding converter between locales by iconv(1) (fundamental)
recode	V:4, I:30	608	charset+eol	text encoding converter between locales (versatile, more aliases and features)
konwert	V:2, I:59	123	charset	text encoding converter between locales (fancy)
nkf	V:0, I:10	347	charset	character set translator for Japanese
tcs	V:0, I:0	479	charset	character set translator
unaccent	V:0, I:0	29	charset	replace accented letters by their unaccented equivalent
tofrodos	V:2, I:32	51	eol	text format converter between DOS and Unix: fromdos(1) and todos(1)
macutils	V:0, I:1	298	eol	text format converter between Macintosh and Unix: frommac(1) and tomac(1)

Table 11.1: List of text data conversion tools

11.1.1 Converting a text file with iconv

Tip
iconv(1) is provided as a part of the `libc6` package and it is always available on practically all Unix-like systems to convert the encoding of characters.

You can convert encodings of a text file with iconv(1) by the following.

```
$ iconv -f encoding1 -t encoding2 input.txt >output.txt
```

Encoding values are case insensitive and ignore "-" and "_" for matching. Supported encodings can be checked by the "`iconv -l`" command.

encoding value	usage
ASCII	American Standard Code for Information Interchange, 7 bit code w/o accented characters
UTF-8	current multilingual standard for all modern OSs
ISO-8859-1	old standard for western European languages, ASCII + accented characters
ISO-8859-2	old standard for eastern European languages, ASCII + accented characters
ISO-8859-15	old standard for western European languages, ISO-8859-1 with euro sign
CP850	code page 850, Microsoft DOS characters with graphics for western European languages, ISO-8859-1 variant
CP932	code page 932, Microsoft Windows style Shift-JIS variant for Japanese
CP936	code page 936, Microsoft Windows style GB2312, GBK or GB18030 variant for Simplified Chinese
CP949	code page 949, Microsoft Windows style EUC-KR or Unified Hangul Code variant for Korean
CP950	code page 950, Microsoft Windows style Big5 variant for Traditional Chinese
CP1251	code page 1251, Microsoft Windows style encoding for the Cyrillic alphabet
CP1252	code page 1252, Microsoft Windows style ISO-8859-15 variant for western European languages
KOI8-R	old Russian UNIX standard for the Cyrillic alphabet
ISO-2022-JP	standard encoding for Japanese email which uses only 7 bit codes
eucJP	old Japanese UNIX standard 8 bit code and completely different from Shift-JIS
Shift-JIS	JIS X 0208 Appendix 1 standard for Japanese (see CP932)

Table 11.2: List of encoding values and their usage

> **Note**
> Some encodings are only supported for the data conversion and are not used as locale values (Section 8.4.1).

For character sets which fit in single byte such as ASCII and ISO-8859 character sets, the character encoding means almost the same thing as the character set.

For character sets with many characters such as JIS X 0213 for Japanese or Universal Character Set (UCS, Unicode, ISO-10646-1) for practically all languages, there are many encoding schemes to fit them into the sequence of the byte data.

• EUC and ISO/IEC 2022 (also known as JIS X 0202) for Japanese

• UTF-8, UTF-16/UCS-2 and UTF-32/UCS-4 for Unicode

For these, there are clear differentiations between the character set and the character encoding.

The code page is used as the synonym to the character encoding tables for some vendor specific ones.

> **Note**
> Please note most encoding systems share the same code with ASCII for the 7 bit characters. But there are some exceptions. If you are converting old Japanese C programs and URLs data from the casually-called shift-JIS encoding format to UTF-8 format, use "CP932" as the encoding name instead of "shift-JIS" to get the expected results: $0x5C \rightarrow$ "\" and $0x7E \rightarrow$ "~". Otherwise, these are converted to wrong characters.

> **Tip**
> recode(1) may be used too and offers more than the combined functionality of iconv(1), fromdos(1), todos(1), frommac(1), and tomac(1). For more, see "info recode".

11.1.2 Checking file to be UTF-8 with iconv

You can check if a text file is encoded in UTF-8 with iconv(1) by the following.

```
$ iconv -f utf8 -t utf8 input.txt >/dev/null || echo "non-UTF-8 found"
```

> **Tip**
> Use "--verbose" option in the above example to find the first non-UTF-8 character.

11.1.3 Converting file names with iconv

Here is an example script to convert encoding of file names from ones created under older OS to modern UTF-8 ones in a single directory.

```
#!/bin/sh
ENCDN=iso-8859-1
for x in *;
 do
 mv "$x" "$(echo "$x" | iconv -f $ENCDN -t utf-8)"
done
```

The "$ENCDN" variable specifies the original encoding used for file names under older OS as in Table 11.2.

For more complicated case, please mount a filesystem (e.g. a partition on a disk drive) containing such file names with proper encoding as the mount(8) option (see Section 8.4.6) and copy its entire contents to another filesystem mounted as UTF-8 with "cp -a" command.

11.1.4 EOL conversion

The text file format, specifically the end-of-line (EOL) code, is dependent on the platform.

platform	EOL code	control	decimal	hexadecimal
Debian (unix)	LF	^J	10	0A
MSDOS and Windows	CR-LF	^M^J	13 10	0D 0A
Apple's Macintosh	CR	^M	13	0D

Table 11.3: List of EOL styles for different platforms

The EOL format conversion programs, fromdos(1), todos(1), frommac(1), and tomac(1), are quite handy. recode(1) is also useful.

> **Note**
> Some data on the Debian system, such as the wiki page data for the `python-moinmoin` package, use MSDOS style CR-LF as the EOL code. So the above rule is just a general rule.

> **Note**
> Most editors (eg. `vim`, `emacs`, `gedit`, ⋯) can handle files in MSDOS style EOL transparently.

> **Tip**
> The use of "`sed -e '/\r$/!s/$/\r/'`" instead of todos(1) is better when you want to unify the EOL style to the MSDOS style from the mixed MSDOS and Unix style. (e.g., after merging 2 MSDOS style files with diff3(1).) This is because `todos` adds CR to all lines.

11.1.5 TAB conversion

There are few popular specialized programs to convert the tab codes.

function	bsdmainutils	coreutils
expand tab to spaces	"col -x"	expand
unexpand tab from spaces	"col -h"	unexpand

Table 11.4: List of TAB conversion commands from `bsdmainutils` and `coreutils` packages

indent(1) from the `indent` package completely reformats whitespaces in the C program.

Editor programs such as `vim` and `emacs` can be used for TAB conversion, too. For example with `vim`, you can expand TAB with "`:set expandtab`" and "`:%retab`" command sequence. You can revert this with "`:set noexpandtab`" and "`:%retab!`" command sequence.

11.1.6 Editors with auto-conversion

Intelligent modern editors such as the `vim` program are quite smart and copes well with any encoding systems and any file formats. You should use these editors under the UTF-8 locale in the UTF-8 capable console for the best compatibility.

An old western European Unix text file, "`u-file.txt`", stored in the latin1 (iso-8859-1) encoding can be edited simply with `vim` by the following.

```
$ vim u-file.txt
```

This is possible since the auto detection mechanism of the file encoding in `vim` assumes the UTF-8 encoding first and, if it fails, assumes it to be latin1.

An old Polish Unix text file, "`pu-file.txt`", stored in the latin2 (iso-8859-2) encoding can be edited with `vim` by the following.

```
$ vim '+e ++enc=latin2 pu-file.txt'
```

An old Japanese unix text file, "`ju-file.txt`", stored in the eucJP encoding can be edited with `vim` by the following.

```
$ vim '+e ++enc=eucJP ju-file.txt'
```

An old Japanese MS-Windows text file, "`jw-file.txt`", stored in the so called shift-JIS encoding (more precisely: CP932) can be edited with `vim` by the following.

```
$ vim '+e ++enc=CP932 ++ff=dos jw-file.txt'
```

When a file is opened with "`++enc`" and "`++ff`" options, "`:w`" in the Vim command line stores it in the original format and overwrite the original file. You can also specify the saving format and the file name in the Vim command line, e.g., "`:w ++enc=utf8 new.txt`".

Please refer to the mbyte.txt "multi-byte text support" in `vim` on-line help and Table 11.2 for locale values used with "`++enc`".

The `emacs` family of programs can perform the equivalent functions.

11.1.7 Plain text extraction

The following reads a web page into a text file. This is very useful when copying configurations off the Web or applying basic Unix text tools such as grep(1) on the web page.

```
$ w3m -dump http://www.remote-site.com/help-info.html >textfile
```

Similarly, you can extract plain text data from other formats using the following.

package	popcon	size	keyword	function
w3m	V:126, I:564	2302	html→text	HTML to text converter with the "`w3m -dump`" command
html2text	V:14, I:58	257	html→text	advanced HTML to text converter (ISO 8859-1)
lynx	V:23, I:104	1903	html→text	HTML to text converter with the "`lynx -dump`" command
elinks	V:12, I:31	1583	html→text	HTML to text converter with the "`elinks -dump`" command
links	V:15, I:44	2123	html→text	HTML to text converter with the "`links -dump`" command
links2	V:2, I:16	5366	html→text	HTML to text converter with the "`links2 -dump`" command
antiword	V:5, I:13	623	MSWord→text,ps	convert MSWord files to plain text or ps
catdoc	V:56, I:100	675	MSWord→text,TeX	convert MSWord files to plain text or TeX
pstotext	V:3, I:6	126	ps/pdf→text	extract text from PostScript and PDF files
unhtml	V:0, I:0	42	html→text	remove the markup tags from an HTML file
odt2txt	V:2, I:7	53	odt→text	converter from OpenDocument Text to text

Table 11.5: List of tools to extract plain text data

package	popcon	size	keyword	description
vim-runtime	V:20, I:429	28409	highlight	Vim MACRO to convert source code to HTML with ":source $VIMRUNTIME/syntax/html.vim"
cxref	V:0, I:0	1182	c → html	converter for the C program to latex and HTML (C language)
src2tex	V:0, I:0	622	highlight	convert many source codes to TeX (C language)
source-highlight	V:1, I:7	2002	highlight	convert many source codes to HTML, XHTML, LaTeX, Texinfo, ANSI color escape sequences and DocBook files with highlight (C++)
highlight	V:1, I:16	1028	highlight	convert many source codes to HTML, XHTML, RTF, LaTeX, TeX or XSL-FO files with highlight (C++)
grc	V:0, I:2	106	text → color	generic colouriser for everything (Python)
txt2html	V:0, I:4	296	text → html	text to HTML converter (Perl)
markdown	V:0, I:7	57	text → html	markdown text document formatter to (X)HTML (Perl)
asciidoc	I:13	80	text → any	AsciiDoc text document formatter to XML/HTML (Python)
pandoc	V:5, I:35	50901	text → any	general markup converter (Haskell)
python-docutils	V:39, I:332	1752	text → any	ReStructured Text document formatter to XML (Python)
txt2tags	V:0, I:1	951	text → any	document conversion from text to HTML, SGML, LaTeX, man page, MoinMoin, Magic Point and PageMaker (Python)
udo	V:0, I:0	564	text → any	universal document - text processing utility (C language)
stx2any	V:0, I:0	264	text → any	document converter from structured plain text to other formats (m4)
rest2web	V:0, I:0	527	text → html	document converter from ReStructured Text to html (Python)
aft	V:0, I:0	235	text → any	"free form" document preparation system (Perl)
yodl	V:0, I:0	615	text → any	pre-document language and tools to process it (C language)
sdf	V:0, I:0	1445	text → any	simple document parser (Perl)
sisu	V:0, I:0	5341	text → any	document structuring, publishing and search framework (Ruby)

Table 11.6: List of tools to highlight plain text data

11.1.8   Highlighting and formatting plain text data

You can highlight and format plain text data by the following.

11.2   XML data

The Extensible Markup Language (XML) is a markup language for documents containing structured information.

See introductory information at XML.COM.

- ”What is XML?”
- ”What Is XSLT?”
- ”What Is XSL-FO?”
- ”What Is XLink?”

11.2.1   Basic hints for XML

XML text looks somewhat like HTML. It enables us to manage multiple formats of output for a document. One easy XML system is the `docbook-xsl` package, which is used here.

Each XML file starts with standard XML declaration as the following.

```
<?xml version="1.0" encoding="UTF-8"?>
```

The basic syntax for one XML element is marked up as the following.

```
<name attribute="value">content</name>
```

XML element with empty content is marked up in the following short form.

```
<name attribute="value"/>
```

The ”`attribute="value"`” in the above examples are optional.

The comment section in XML is marked up as the following.

```
<!-- comment -->
```

Other than adding markups, XML requires minor conversion to the content using predefined entities for following characters.

predefined entity	character to be converted into
"	" : quote
'	' : apostrophe
<	< : less-than
>	> : greater-than
&	& : ampersand

Table 11.7: List of predefined entities for XML

⚠️ **Caution**
”<” or ”&” can not be used in attributes or elements.

Note

When SGML style user defined entities, e.g. "&some-tag:", are used, the first definition wins over others. The entity definition is expressed in "<!ENTITY some-tag "entity value">".

Note

As long as the XML markup are done consistently with certain set of the tag name (either some data as content or attribute value), conversion to another XML is trivial task using Extensible Stylesheet Language Transformations (XSLT).

11.2.2 XML processing

There are many tools available to process XML files such as the Extensible Stylesheet Language (XSL).

Basically, once you create well formed XML file, you can convert it to any format using Extensible Stylesheet Language Transformations (XSLT).

The Extensible Stylesheet Language for Formatting Objects (XSL-FO) is supposed to be solution for formatting. The `fop` package is new to the Debian `main` archive due to its dependence to the Java programing language. So the LaTeX code is usually generated from XML using XSLT and the LaTeX system is used to create printable file such as DVI, PostScript, and PDF.

package	popcon	size	keyword	description
docbook-xml	I:529	2131	xml	XML document type definition (DTD) for DocBook
xsltproc	V:16, I:113	149	xslt	XSLT command line processor (XML $\rightarrow$ XML, HTML, plain text, etc.)
docbook-xsl	V:14, I:222	14998	xml/xslt	XSL stylesheets for processing DocBook XML to various output formats with XSLT
xmlto	V:2, I:32	126	xml/xslt	XML-to-any converter with XSLT
dbtoepub	V:0, I:1	71	xml/xslt	DocBook XML to .epub converter
dblatex	V:6, I:21	4648	xml/xslt	convert Docbook files to DVI, PostScript, PDF documents with XSLT
fop	V:2, I:47	64	xml/xsl-fo	convert Docbook XML files to PDF

Table 11.8: List of XML tools

Since XML is subset of Standard Generalized Markup Language (SGML), it can be processed by the extensive tools available for SGML, such as Document Style Semantics and Specification Language (DSSSL).

package	popcon	size	keyword	description
openjade	V:3, I:44	988	dsssl	ISO/IEC 10179:1996 standard DSSSL processor (latest)
docbook-dsssl	V:2, I:33	2604	xml/dsssl	DSSSL stylesheets for processing DocBook XML to various output formats with DSSSL
docbook-utils	V:1, I:22	281	xml/dsssl	utilities for DocBook files including conversion to other formats (HTML, RTF, PS, man, PDF) with `docbook2*` commands with DSSSL
sgml2x	V:0, I:0	90	SGML/dsssl	converter from SGML and XML using DSSSL stylesheets

Table 11.9: List of DSSSL tools

Tip

GNOME's `yelp` is sometimes handy to read DocBook XML files directly since it renders decently on X.

11.2.3 The XML data extraction

You can extract HTML or XML data from other formats using followings.

package	popcon	size	keyword	description
wv	V:5, I:8	717	MSWord→any	document converter from Microsoft Word to HTML, LaTeX, etc.
texi2html	V:0, I:9	1832	texi→html	converter from Texinfo to HTML
man2html	V:0, I:3	141	manpage→html	converter from manpage to HTML (CGI support)
unrtf	V:1, I:4	145	rtf→html	document converter from RTF to HTML, etc
info2www	V:2, I:4	156	info→html	converter from GNU info to HTML (CGI support)
ooo2dbk	V:0, I:1	217	sxw→xml	converter from OpenOffice.org SXW documents to DocBook XML
wp2x	V:0, I:0	215	WordPerfect→any	WordPerfect 5.0 and 5.1 files to TeX, LaTeX, troff, GML and HTML
doclifter	V:0, I:0	457	troff→xml	converter from troff to DocBook XML

Table 11.10: List of XML data extraction tools

For non-XML HTML files, you can convert them to XHTML which is an instance of well formed XML. XHTML can be processed by XML tools.

package	popcon	size	keyword	description
libxml2-utils	V:23, I:306	167	xml↔html↔xhtml	command line XML tool with xmllint(1) (syntax check, reformat, lint, ⋯)
tidy	V:2, I:17	83	xml↔html↔xhtml	HTML syntax checker and reformatter

Table 11.11: List of XML pretty print tools

Once proper XML is generated, you can use XSLT technology to extract data based on the mark-up context etc.

11.3 Type setting

The Unix troff program originally developed by AT&T can be used for simple typesetting. It is usually used to create manpages.

TeX created by Donald Knuth is a very powerful type setting tool and is the de facto standard. LaTeX originally written by Leslie Lamport enables a high-level access to the power of TeX.

package	popcon	size	keyword	description
texlive	V:6, I:63	69	(La)TeX	TeX system for typesetting, previewing and printing
groff	V:5, I:116	9389	troff	GNU troff text-formatting system

Table 11.12: List of type setting tools

11.3.1 roff typesetting

Traditionally, roff is the main Unix text processing system. See roff(7), groff(7), groff(1), grotty(1), troff(1), groff_mdoc(7), groff_man(7), groff_ms(7), groff_me(7), groff_mm(7), and "info groff".

You can read or print a good tutorial and reference on "-me" macro in "/usr/share/doc/groff/" by installing the groff package.

Tip

"`groff -Tascii -me -`" produces plain text output with ANSI escape code. If you wish to get manpage like output with many "ˆH" and "_", use "`GROFF_NO_SGR=1 groff -Tascii -me -`" instead.

Tip

To remove "ˆH" and "_" from a text file generated by `groff`, filter it by "`col -b -x`".

11.3.2 TeX/LaTeX

The TeX Live software distribution offers a complete TeX system. The `texlive` metapackage provides a decent selection of the TeX Live packages which should suffice for the most common tasks.

There are many references available for TeX and LaTeX.

- The teTeX HOWTO: The Linux-teTeX Local Guide

- tex(1)

- latex(1)

- texdoc(1)

- texdoctk(1)

- "The TeXbook", by Donald E. Knuth, (Addison-Wesley)

- "LaTeX - A Document Preparation System", by Leslie Lamport, (Addison-Wesley)

- "The LaTeX Companion", by Goossens, Mittelbach, Samarin, (Addison-Wesley)

This is the most powerful typesetting environment. Many SGML processors use this as their back end text processor. Lyx provided by the `lyx` package and GNU TeXmacs provided by the `texmacs` package offer nice WYSIWYG editing environment for LaTeX while many use Emacs and Vim as the choice for the source editor.

There are many online resources available.

- The TEX Live Guide - TEX Live 2007 ("`/usr/share/doc/texlive-doc-base/english/texlive-en/live.html`") (`texlive-doc-base` package)

- A Simple Guide to Latex/Lyx

- Word Processing Using LaTeX

- Local User Guide to teTeX/LaTeX

When documents become bigger, sometimes TeX may cause errors. You must increase pool size in "`/etc/texmf/texmf.cnf`" (or more appropriately edit "`/etc/texmf/texmf.d/95NonPath`" and run update-texmf(8)) to fix this.

Note

The TeX source of "The TeXbook" is available at http://tug.ctan.org/tex-archive/systems/knuth/dist/tex/texbook.tex. This file contains most of the required macros. I heard that you can process this document with tex(1) after commenting lines 7 to 10 and adding "`\input manmac \proofmodefalse`". It's strongly recommended to buy this book (and all other books from Donald E. Knuth) instead of using the online version but the source is a great example of TeX input!

11.3.3 Pretty print a manual page

You can print a manual page in PostScript nicely by one of the following commands.

```
$ man -Tps some_manpage | lpr
```

11.3.4 Creating a manual page

Although writing a manual page (manpage) in the plain troff format is possible, there are few helper packages to create it.

package	popcon	size	keyword	description
docbook-to-man	V:1, I:16	187	SGML→manpage	converter from DocBook SGML into roff man macros
help2man	V:0, I:9	464	text→manpage	automatic manpage generator from --help
info2man	V:0, I:0	134	info→manpage	converter from GNU info to POD or man pages
txt2man	V:0, I:1	65	text→manpage	convert flat ASCII text to man page format

Table 11.13: List of packages to help creating the manpage

11.4 Printable data

Printable data is expressed in the PostScript format on the Debian system. Common Unix Printing System (CUPS) uses Ghostscript as its rasterizer backend program for non-PostScript printers.

11.4.1 Ghostscript

The core of printable data manipulation is the Ghostscript PostScript (PS) interpreter which generates raster image.

The latest upstream Ghostscript from Artifex was re-licensed from AFPL to GPL and merged all the latest ESP version changes such as CUPS related ones at 8.60 release as unified release.

package	popcon	size	description
ghostscript	V:149, I:681	219	The GPL Ghostscript PostScript/PDF interpreter
ghostscript-x	V:29, I:72	211	GPL Ghostscript PostScript/PDF interpreter - X display support
libpoppler73	http://qa.debian.org/popcon.php?package=libpoppler73	http://packages.qa.debian.org/libp/libpoppler73.html	PDF rendering library forked from the xpdf PDF viewer
libpoppler-glib8	V:209, I:532	446	PDF rendering library (GLib-based shared library)
poppler-data	V:103, I:674	12219	CMaps for PDF rendering library (for CJK support: Adobe-*)

Table 11.14: List of Ghostscript PostScript interpreters

Tip
"gs -h" can display the configuration of Ghostscript.

11.4.2 Merge two PS or PDF files

You can merge two PostScript (PS) or Portable Document Format (PDF) files using gs(1) of Ghostscript.

```
$ gs -q -dNOPAUSE -dBATCH -sDEVICE=pswrite -sOutputFile=bla.ps -f foo1.ps foo2.ps
$ gs -q -dNOPAUSE -dBATCH -sDEVICE=pdfwrite -sOutputFile=bla.pdf -f foo1.pdf foo2.pdf
```

Note
The PDF, which is a widely used cross-platform printable data format, is essentially the compressed PS format with few additional features and extensions.

Tip
For command line, psmerge(1) and other commands from the `psutils` package are useful for manipulating PostScript documents. pdftk(1) from the `pdftk` package is useful for manipulating PDF documents, too.

11.4.3 Printable data utilities

The following packages for the printable data utilities caught my eyes.

package	popcon	size	keyword	description
poppler-utils	V:49, I:485	592	pdf→ps,text, …	PDF utilities: `pdftops`, `pdfinfo`, `pdfimages`, `pdftotext`, `pdffonts`
psutils	V:9, I:164	219	ps→ps	PostScript document conversion tools
poster	V:0, I:6	49	ps→ps	create large posters out of PostScript pages
enscript	V:2, I:24	2111	text→ps, html, rtf	convert ASCII text to PostScript, HTML, RTF or Pretty-Print
a2ps	V:2, I:22	3620	text→ps	'Anything to PostScript' converter and pretty-printer
pdftk	V:8, I:57	3873	pdf→pdf	PDF document conversion tool: `pdftk`
html2ps	V:0, I:5	249	html→ps	converter from HTML to PostScript
gnuhtml2latex	V:0, I:1	27	html→latex	converter from html to latex
latex2rtf	V:0, I:6	478	latex→rtf	convert documents from LaTeX to RTF which can be read by MS Word
ps2eps	V:6, I:105	94	ps→eps	converter from PostScript to EPS (Encapsulated PostScript)
e2ps	V:0, I:0	112	text→ps	Text to PostScript converter with Japanese encoding support
impose+	V:0, I:1	180	ps→ps	PostScript utilities
trueprint	V:0, I:0	138	text→ps	pretty print many source codes (C, C++, Java, Pascal, Perl, Pike, Sh, and Verilog) to PostScript. (C language)
pdf2svg	V:0, I:4	26	ps→svg	converter from PDF to Scalable vector graphics format
pdftoipe	V:0, I:0	62	ps→ipe	converter from PDF to IPE's XML format

Table 11.15: List of printable data utilities

11.4.4 Printing with CUPS

Both lp(1) and lpr(1) commands offered by Common Unix Printing System (CUPS) provides options for customized printing the printable data.

You can print 3 copies of a file collated using one of the following commands.

```
$ lp -n 3 -o Collate=True filename
```

```
$ lpr -#3 -o Collate=True filename
```

You can further customize printer operation by using printer option such as "-o number-up=2", "-o page-set=even", "-o page-set=odd", "-o scaling=200", "-o natural-scaling=200", etc., documented at Command-Line Printing and Options.

11.5   The mail data conversion

The following packages for the mail data conversion caught my eyes.

package	popcon	size	keyword	description
sharutils	V:7, I:90	1385	mail	shar(1), unshar(1), uuencode(1), uudecode(1)
mpack	V:2, I:23	91	MIME	encoding and decoding of MIME messages: mpack(1) and munpack(1)
tnef	V:5, I:10	98	ms-tnef	unpacking MIME attachments of type "application/ms-tnef" which is a Microsoft only format
uudeview	V:0, I:5	101	mail	encoder and decoder for the following formats: uuencode, xxencode, BASE64, quoted printable, and BinHex
readpst	I:1	21	PST	convert Microsoft Outlook PST files to mbox format

Table 11.16: List of packages to help mail data conversion

> **Tip**
> The Internet Message Access Protocol version 4 (IMAP4) server (see Section 6.7) may be used to move mails out from proprietary mail systems if the mail client software can be configured to use IMAP4 server too.

11.5.1   Mail data basics

Mail (SMTP) data should be limited to series of 7 bit data. So binary data and 8 bit text data are encoded into 7 bit format with the Multipurpose Internet Mail Extensions (MIME) and the selection of the charset (see Section 8.4.1).

The standard mail storage format is mbox formatted according to RFC2822 (updated RFC822). See mbox(5) (provided by the `mutt` package).

For European languages, "Content-Transfer-Encoding: quoted-printable" with the ISO-8859-1 charset is usually used for mail since there are not much 8 bit characters. If European text is encoded in UTF-8, "Content-Transfer-Encoding: quoted-printable" is likely to be used since it is mostly 7 bit data.

For Japanese, traditionally "Content-Type: text/plain; charset=ISO-2022-JP" is usually used for mail to keep text in 7 bits. But older Microsoft systems may send mail data in Shift-JIS without proper declaration. If Japanese text is encoded in UTF-8, Base64 is likely to be used since it contains many 8 bit data. The situation of other Asian languages is similar.

> **Note**
> If your non-Unix mail data is accessible by a non-Debian client software which can talk to the IMAP4 server, you may be able to move them out by running your own IMAP4 server (see Section 6.7).

Note

If you use other mail storage formats, moving them to mbox format is the good first step. The versatile client program such as mutt(1) may be handy for this.

You can split mailbox contents to each message using procmail(1) and formail(1).

Each mail message can be unpacked using munpack(1) from the `mpack` package (or other specialized tools) to obtain the MIME encoded contents.

11.6   Graphic data tools

The following packages for the graphic data conversion, editing, and organization tools caught my eyes.

Tip

Search more image tools using regex "`~Gworks-with::image`" in aptitude(8) (see Section 2.2.6).

Although GUI programs such as gimp(1) are very powerful, command line tools such as imagemagick(1) are quite useful for automating image manipulation via scripts.

The de facto image file format of the digital camera is the Exchangeable Image File Format (EXIF) which is the JPEG image file format with additional metadata tags. It can hold information such as date, time, and camera settings.

The Lempel-Ziv-Welch (LZW) lossless data compression patent has been expired. Graphics Interchange Format (GIF) utilities which use the LZW compression method are now freely available on the Debian system.

Tip

Any digital camera or scanner with removable recording media works with Linux through USB storage readers since it follows the Design rule for Camera Filesystem and uses FAT filesystem. See Section 10.1.7.

11.7   Miscellaneous data conversion

There are many other programs for converting data. Following packages caught my eyes using regex "`~Guse::converting`" in aptitude(8) (see Section 2.2.6).

You can also extract data from RPM format with the following.

```
$ rpm2cpio file.src.rpm | cpio --extract
```

package	popcon	size	keyword	description
gimp	V:84, I:506	16413	image(bitmap)	GNU Image Manipulation Program
imagemagick	V:64, I:560	205	image(bitmap)	image manipulation programs
graphicsmagick	V:6, I:15	4970	image(bitmap)	image manipulation programs (fork of `imagemagick`)
xsane	V:21, I:195	909	image(bitmap)	GTK+-based X11 frontend for SANE (Scanner Access Now Easy)
netpbm	V:29, I:563	4302	image(bitmap)	graphics conversion tools
icoutils	V:11, I:142	192	png↔ico(bitmap)	convert MS Windows icons and cursors to and from PNG formats (favicon.ico)
scribus	V:5, I:27	19061	ps/pdf/SVG/…	Scribus DTP editor
libreoffice-draw	V:307, I:481	9881	image(vector)	LibreOffice office suite - drawing
inkscape	V:153, I:346	130049	image(vector)	SVG (Scalable Vector Graphics) editor
dia	V:21, I:38	3916	image(vector)	diagram editor (Gtk)
xfig	V:10, I:17	1792	image(vector)	Facility for Interactive Generation of figures under X11
pstoedit	V:7, I:213	667	ps/pdf→image(vector)	PostScript and PDF files to editable vector graphics converter (SVG)
libwmf-bin	V:13, I:352	104	Windows/image(vector)	Windows metafile (vector graphic data) conversion tools
fig2sxd	V:0, I:0	149	fig→sxd(vector)	convert XFig files to OpenOffice.org Draw format
unpaper	V:2, I:16	453	image→image	post-processing tool for scanned pages for OCR
tesseract-ocr	V:4, I:30	1020	image→text	free OCR software based on the HP's commercial OCR engine
tesseract-ocr-eng	I:30	4032	image→text	OCR engine data: tesseract-ocr language files for English text
gocr	V:1, I:21	477	image→text	free OCR software
ocrad	V:1, I:6	299	image→text	free OCR software
eog	V:92, I:319	11324	image(Exif)	Eye of GNOME graphics viewer program
gthumb	V:14, I:25	3359	image(Exif)	image viewer and browser (GNOME)
geeqie	V:14, I:23	1989	image(Exif)	image viewer using GTK+
shotwell	V:20, I:207	6635	image(Exif)	digital photo organizer (GNOME)
gtkam	V:0, I:7	1154	image(Exif)	application for retrieving media from digital cameras (GTK+)
gphoto2	V:1, I:13	967	image(Exif)	The gphoto2 digital camera command-line client
gwenview	V:32, I:105	10871	image(Exif)	image viewer (KDE)
kamera	I:105	748	image(Exif)	digital camera support for KDE applications
digikam	V:3, I:16	1832	image(Exif)	digital photo management application for KDE
exiv2	V:4, I:61	246	image(Exif)	EXIF/IPTC metadata manipulation tool
exiftran	V:2, I:25	71	image(Exif)	transform digital camera jpeg images
jhead	V:1, I:12	105	image(Exif)	manipulate the non-image part of Exif compliant JPEG (digital camera photo) files
exif	V:1, I:11	238	image(Exif)	command-line utility to show EXIF information in JPEG files
exiftags	V:0, I:4	288	image(Exif)	utility to read Exif tags from a digital camera JPEG file
exifprobe	V:0, I:3	490	image(Exif)	read metadata from digital pictures
dcraw	V:3, I:23	384	image(Raw)→ppm	decode raw digital camera images
findimagedupes	V:0, I:1	79	image→fingerprint	find visually similar or duplicate images
ale	V:0, I:0	753	image→image	merge images to increase fidelity or create mosaics
imageindex	V:0, I:0	144	image(Exif)→html	generate static HTML galleries from images
outguess	V:0, I:2	216	jpeg,png	universal Steganographic tool
librecad	V:10, I:18	8039	DXF	CAD data editor (KDE)
blender	V:4, I:33	106780	blend, TIFF, VRML, …	3D content editor for animation etc
mm3d	V:0, I:0	4047	ms3d, obj, dxf, …	OpenGL based 3D model editor
open-font-design-toolkit	I:0	10	ttf, ps, …	metapackage for open font design
fontforge	V:1, I:9	91	ttf, ps, …	font editor for PS, TrueType and OpenType fonts
xgridfit	V:0, I:0	876	ttf	program for gridfitting and hinting TrueType fonts

package	popcon	size	keyword	description
alien	V:4, I:52	166	rpm/tgz→deb	converter for the foreign package into the Debian package
freepwing	V:0, I:0	421	EB→EPWING	converter from "Electric Book" (popular in Japan) to a single JIS X 4081 format (a subset of the EPWING V1)
calibre	V:7, I:38	51018	any→EPUB	e-book converter and library management

Table 11.18: List of miscellaneous data conversion tools

Chapter 12

Programming

I provide some pointers for people to learn programming on the Debian system enough to trace the packaged source code. Here are notable packages and corresponding documentation packages for programming.

package	popcon	size	documentation
autoconf	V:35, I:255	1868	"info autoconf" provided by autoconf-doc
automake	V:34, I:250	1710	"info automake" provided by automake1.10-doc
bash	V:830, I:999	5790	"info bash" provided by bash-doc
bison	V:11, I:108	2061	"info bison" provided by bison-doc
cpp	V:370, I:802	42	"info cpp" provided by cpp-doc
ddd	V:1, I:12	3965	"info ddd" provided by ddd-doc
exuberant-ctags	V:7, I:42	333	exuberant-ctags(1)
flex	V:10, I:97	1220	"info flex" provided by flex-doc
gawk	V:411, I:515	2196	"info gawk" provided by gawk-doc
gcc	V:157, I:597	45	"info gcc" provided by gcc-doc
gdb	V:20, I:130	7928	"info gdb" provided by gdb-doc
gettext	V:67, I:356	6496	"info gettext" provided by gettext-doc
gfortran	V:6, I:65	16	"info gfortran" provided by gfortran-doc (Fortran 95)
fpc	I:4	119	fpc(1) and html by fp-docs (Pascal)
glade	V:1, I:9	2267	help provided via menu (UI Builder)
libc6	V:930, I:999	11401	"info libc" provided by glibc-doc and glibc-doc-reference
make	V:179, I:604	1292	"info make" provided by make-doc
xutils-dev	V:1, I:15	1466	imake(1), xmkmf(1), etc.
mawk	V:354, I:997	183	mawk(1)
perl	V:608, I:995	568	perl(1) and html pages provided by perl-doc and perl-doc-html
python	V:615, I:988	624	python(1) and html pages provided by python-doc
tcl	V:30, I:444	21	tcl(3) and detail manual pages provided by tcl-doc
tk	V:31, I:433	21	tk(3) and detail manual pages provided by tk-doc
ruby	V:157, I:340	37	ruby(1) and interactive reference provided by ri
vim	V:121, I:390	2660	help(F1) menu provided by vim-doc
susv2	I:0	15	fetch "The Single UNIX Specifications v2"
susv3	I:0	15	fetch "The Single UNIX Specifications v3"

Table 12.1: List of packages to help programming

Online references are available by typing "man name" after installing manpages and manpages-dev packages. Online references for the GNU tools are available by typing "info program_name" after installing the pertinent documentation packages.

> ⚠ **Warning**
>
> Do not use "`test`" as the name of an executable test file. "`test`" is a shell builtin.

> ⚠ **Caution**
>
> You should install software programs directly compiled from source into "`/usr/local`" or "`/opt`" to avoid collision with system programs.

> **Tip**
>
> Code examples of creating "Song 99 Bottles of Beer" should give you good ideas of practically all the programming languages.

12.1   The shell script

The shell script is a text file with the execution bit set and contains the commands in the following format.

```
#!/bin/sh
 ... command lines
```

The first line specifies the shell interpreter which read and execute this file contents.

Reading shell scripts is the **best** way to understand how a Unix-like system works. Here, I give some pointers and reminders for shell programming. See "Shell Mistakes" (http://www.greenend.org.uk/rjk/2001/04/shell.html) to learn from mistakes.

Unlike shell interactive mode (see Section 1.5 and Section 1.6), shell scripts frequently use parameters, conditionals, and loops.

12.1.1   POSIX shell compatibility

Many system scripts may be interpreted by any one of POSIX shells (see Table 1.13). The default shell for the system is "`/bin/sh`" which is a symlink pointing to the actual program.

- bash(1) for `lenny` or older

- dash(1) for `squeeze` or newer

Avoid writing a shell script with **bashisms** or **zshisms** to make it portable among all POSIX shells. You can check it using checkbashisms(1).

Good: POSIX	Avoid: bashism
`if [ "$foo" = "$bar" ] ; then ⋯`	`if [ "$foo" == "$bar" ] ; then ⋯`
`diff -u file.c.orig file.c`	`diff -u file.c{.orig,}`
`mkdir /foobar /foobaz`	`mkdir /foo{bar,baz}`
`funcname() { ⋯}`	`function funcname() { ⋯}`
octal format: "`\377`"	hexadecimal format: "`\xff`"

Table 12.2: List of typical bashisms

The "`echo`" command must be used with following cares since its implementation differs among shell builtin and external commands.

- Avoid using any command options except ”-n”.

- Avoid using escape sequences in the string since their handling varies.

Note
Although ”-n” option is **not** really POSIX syntax, it is generally accepted.

Tip
Use the ”printf” command instead of the ”echo” command if you need to embed escape sequences in the output string.

12.1.2 Shell parameters

Special shell parameters are frequently used in the shell script.

shell parameter	value
$0	name of the shell or shell script
$1	first (1st) shell argument
$9	ninth (9th) shell argument
$#	number of positional parameters
”$*”	”$1 $2 $3 $4 ···”
”$@”	”$1” ”$2” ”$3” ”$4” ···
$?	exit status of the most recent command
$$	PID of this shell script
$!	PID of most recently started background job

Table 12.3: List of shell parameters

Basic **parameter expansions** to remember are as follows.

parameter expression form	value if **var** is set	value if **var** is not set
${var:-string}	”$var”	”string”
${var:+string}	”string”	”null”
${var:=string}	”$var”	”string” (and run ”var=string”)
${var:?string}	”$var”	echo ”string” to **stderr** (and exit with error)

Table 12.4: List of shell parameter expansions

Here, the colon ”:” in all of these operators is actually optional.

- **with** ”:” = operator test for **exist** and **not null**

- **without** ”:” = operator test for **exist** only

12.1.3 Shell conditionals

Each command returns an **exit status** which can be used for conditional expressions.

- Success: 0 (”True”)

parameter substitution form	result
`${var%suffix}`	remove smallest suffix pattern
`${var%%suffix}`	remove largest suffix pattern
`${var#prefix}`	remove smallest prefix pattern
`${var##prefix}`	remove largest prefix pattern

Table 12.5: List of key shell parameter substitutions

- Error: non 0 ("False")

Note
"0" in the shell conditional context means "True", while "0" in the C conditional context means "False".

Note
"[" is the equivalent of the `test` command, which evaluates its arguments up to "]" as a conditional expression.

Basic **conditional idioms** to remember are the following.

- "<command> && <if_success_run_this_command_too> || true"

- "<command> || <if_not_success_run_this_command_too> || true"

- A multi-line script snippet as the following

```
if [ <conditional_expression> ]; then
 <if_success_run_this_command>
else
 <if_not_success_run_this_command>
fi
```

Here trailing "|| true" was needed to ensure this shell script does not exit at this line accidentally when shell is invoked with "-e" flag.

equation	condition to return logical true
`-e <file>`	<file> exists
`-d <file>`	<file> exists and is a directory
`-f <file>`	<file> exists and is a regular file
`-w <file>`	<file> exists and is writable
`-x <file>`	<file> exists and is executable
`<file1> -nt <file2>`	<file1> is newer than <file2> (modification)
`<file1> -ot <file2>`	<file1> is older than <file2> (modification)
`<file1> -ef <file2>`	<file1> and <file2> are on the same device and the same inode number

Table 12.6: List of file comparison operators in the conditional expression

Arithmetic integer comparison operators in the conditional expression are "`-eq`", "`-ne`", "`-lt`", "`-le`", "`-gt`", and "`-ge`".

12.1.4 Shell loops

There are several loop idioms to use in POSIX shell.

equation	condition to return logical true
`-z <str>`	the length of \<str\> is zero
`-n <str>`	the length of \<str\> is non-zero
`<str1> = <str2>`	\<str1\> and \<str2\> are equal
`<str1> != <str2>`	\<str1\> and \<str2\> are not equal
`<str1> < <str2>`	\<str1\> sorts before \<str2\> (locale dependent)
`<str1> > <str2>`	\<str1\> sorts after \<str2\> (locale dependent)

Table 12.7: List of string comparison operators in the conditional expression

- "`for x in foo1 foo2 ⋯; do command ; done`" loops by assigning items from the list "`foo1 foo2 ⋯`" to variable "`x`" and executing "`command`".

- "`while condition ; do command ; done`" repeats "`command`" while "`condition`" is true.

- "`until condition ; do command ; done`" repeats "`command`" while "`condition`" is not true.

- "`break`" enables to exit from the loop.

- "`continue`" enables to resume the next iteration of the loop.

Tip
The C-language like numeric iteration can be realized by using seq(1) as the "`foo1 foo2 ⋯`" generator.

Tip
See Section 9.3.9.

12.1.5 The shell command-line processing sequence

The shell processes a script roughly as the following sequence.

- The shell reads a line.

- The shell groups a part of the line as **one token** if it is within "`⋯`" or '`⋯`'.

- The shell splits other part of a line into **tokens** by the following.

  – Whitespaces: `<space> <tab> <newline>`
  – Metacharacters: `< > | ; & ( )`

- The shell checks the **reserved word** for each token to adjust its behavior if not within "`⋯`" or '`⋯`'.

  – **reserved word**: `if then elif else fi for in while unless do done case esac`

- The shell expands **alias** if not within "`⋯`" or '`⋯`'.

- The shell expands **tilde** if not within "`⋯`" or '`⋯`'.

  – "`~`" → current user's home directory
  – "`~<user>`" → \<user\>'s home directory

- The shell expands **parameter** to its value if not within '`⋯`'.

  – **parameter**: "`$PARAMETER`" or "`${PARAMETER}`"

- The shell expands **command substitution** if not within '···'.

  - "$(command)" → the output of "command"
  - "` command `" → the output of "command"

- The shell expands **pathname glob** to matching file names if not within "···" or '···'.

  - * → any characters
  - ? → one character
  - [···] → any one of the characters in "···"

- The shell looks up **command** from the following and execute it.

  - **function** definition
  - **builtin** command
  - **executable file** in "$PATH"

- The shell goes to the next line and repeats this process again from the top of this sequence.

Single quotes within double quotes have no effect.

Executing "set -x" in the shell or invoking the shell with "-x" option make the shell to print all of commands executed. This is quite handy for debugging.

12.1.6 Utility programs for shell script

In order to make your shell program as portable as possible across Debian systems, it is a good idea to limit utility programs to ones provided by **essential** packages.

- "aptitude search ~E" lists **essential** packages.

- "dpkg -L <package_name> |grep '/man/man.*/'" lists manpages for commands offered by <package_name> package.

package	popcon	size	description
coreutils	V:878, I:999	15016	GNU core utilities
debianutils	V:930, I:999	213	miscellaneous utilities specific to Debian
bsdmainutils	V:854, I:998	586	collection of more utilities from FreeBSD
bsdutils	V:851, I:999	256	basic utilities from 4.4BSD-Lite
moreutils	V:4, I:22	201	additional Unix utilities

Table 12.8: List of packages containing small utility programs for shell scripts

Tip
Although moreutils may not exist ouside of Debian, it offers interesting small programs. Most notable one is sponge(8) which is quite useful when you wish to overwrite original file.

12.1.7 Shell script dialog

The user interface of a simple shell program can be improved from dull interaction by echo and read commands to more interactive one by using one of the so-called dialog program etc.

package	popcon	size	description
x11-utils	V:252, I:643	637	xmessage(1): display a message or query in a window (X)
whiptail	V:424, I:996	69	displays user-friendly dialog boxes from shell scripts (newt)
dialog	V:20, I:131	1119	displays user-friendly dialog boxes from shell scripts (ncurses)
zenity	V:97, I:416	368	display graphical dialog boxes from shell scripts (gtk2.0)
ssft	V:0, I:0	75	Shell Scripts Frontend Tool (wrapper for zenity, kdialog, and dialog with gettext)
gettext	V:67, I:356	6496	"/usr/bin/gettext.sh": translate message

Table 12.9: List of user interface programs

12.1.8 Shell script example with zenity

Here is a simple script which creates ISO image with RS02 data supplemented by dvdisaster(1).

```
#!/bin/sh -e
# gmkrs02 : Copyright (C) 2007 Osamu Aoki <osamu@debian.org>, Public Domain
#set -x
error_exit()
{
  echo "$1" >&2
  exit 1
}
# Initialize variables
DATA_ISO="$HOME/Desktop/iso-$$.img"
LABEL=$(date +%Y%m%d-%H%M%S-%Z)
if [ $# != 0 ] && [ -d "$1" ]; then
  DATA_SRC="$1"
else
  # Select directory for creating ISO image from folder on desktop
  DATA_SRC=$(zenity --file-selection --directory  \
    --title="Select the directory tree root to create ISO image") \
    || error_exit "Exit on directory selection"
fi
# Check size of archive
xterm -T "Check size $DATA_SRC" -e du -s $DATA_SRC/*
SIZE=$(($(du -s $DATA_SRC | awk '{print $1}')/1024))
if [ $SIZE -le 520 ] ; then
  zenity --info --title="Dvdisaster RS02" --width 640  --height 400 \
    --text="The data size is good for CD backup:\\n $SIZE MB"
elif [ $SIZE -le 3500 ]; then
  zenity --info --title="Dvdisaster RS02" --width 640  --height 400 \
    --text="The data size is good for DVD backup :\\n $SIZE MB"
else
  zenity --info --title="Dvdisaster RS02" --width 640  --height 400 \
    --text="The data size is too big to backup : $SIZE MB"
  error_exit "The data size is too big to backup :\\n $SIZE MB"
fi
# only xterm is sure to have working -e option
# Create raw ISO image
rm -f "$DATA_ISO" || true
xterm -T "genisoimage $DATA_ISO" \
  -e genisoimage -r -J -V "$LABEL" -o "$DATA_ISO" "$DATA_SRC"
# Create RS02 supplemental redundancy
xterm -T "dvdisaster $DATA_ISO" -e  dvdisaster -i "$DATA_ISO" -mRS02 -c
zenity --info --title="Dvdisaster RS02" --width 640  --height 400 \
  --text="ISO/RS02 data ($SIZE MB) \\n created at: $DATA_ISO"
# EOF
```

You may wish to create launcher on the desktop with command set something like "/usr/local/bin/gmkrs02 %d".

12.2 Make

Make is a utility to maintain groups of programs. Upon execution of make(1), `make` read the rule file, "`Makefile`", and updates a target if it depends on prerequisite files that have been modified since the target was last modified, or if the target does not exist. The execution of these updates may occur concurrently.

The rule file syntax is the following.

```
target: [ prerequisites ... ]
 [TAB]  command1
 [TAB]  -command2 # ignore errors
 [TAB]  @command3 # suppress echoing
```

Here "`[TAB]`" is a TAB code. Each line is interpreted by the shell after make variable substitution. Use "`\`" at the end of a line to continue the script. Use "`$$`" to enter "`$`" for environment values for a shell script.

Implicit rules for the target and prerequisites can be written, for example, by the following.

```
%.o: %.c header.h
```

Here, the target contains the character "%" (exactly one of them). The "%" can match any nonempty substring in the actual target filenames. The prerequisites likewise use "%" to show how their names relate to the actual target name.

automatic variable	value
$@	target
$<	first prerequisite
$?	all newer prerequisites
$^	all prerequisites
$*	"%" matched stem in the target pattern

Table 12.10: List of make automatic variables

variable expansion	description
foo1 := bar	one-time expansion
foo2 = bar	recursive expansion
foo3 += bar	append

Table 12.11: List of make variable expansions

Run "`make -p -f/dev/null`" to see automatic internal rules.

12.3 C

You can set up proper environment to compile programs written in the C programming language by the following.

```
# apt-get install glibc-doc manpages-dev libc6-dev gcc build-essential
```

The `libc6-dev` package, i.e., GNU C Library, provides C standard library which is collection of header files and library routines used by the C programming language.

See references for C as the following.

• "`info libc`" (C library function reference)

• gcc(1) and "`info gcc`"

• each_C_library_function_name(3)

• Kernighan & Ritchie, "The C Programming Language", 2nd edition (Prentice Hall)

12.3.1 Simple C program (gcc)

A simple example "example.c" can compiled with a library "libm" into an executable "run_example" by the following.

```
$ cat > example.c << EOF
#include <stdio.h>
#include <math.h>
#include <string.h>

int main(int argc, char **argv, char **envp){
        double x;
        char y[11];
        x=sqrt(argc+7.5);
        strncpy(y, argv[0], 10); /* prevent buffer overflow */
        y[10] = '\0'; /* fill to make sure string ends with '\0' */
        printf("%5i, %5.3f, %10s, %10s\n", argc, x, y, argv[1]);
        return 0;
}
EOF
$ gcc -Wall -g -o run_example example.c -lm
$ ./run_example
        1, 2.915, ./run_exam,     (null)
$ ./run_example 1234567890qwerty
        2, 3.082, ./run_exam, 1234567890qwerty
```

Here, "-lm" is needed to link library "/usr/lib/libm.so" from the libc6 package for sqrt(3). The actual library is in "/lib/" with filename "libm.so.6", which is a symlink to "libm-2.7.so".

Look at the last parameter in the output text. There are more than 10 characters even though "%10s" is specified.

The use of pointer memory operation functions without boundary checks, such as sprintf(3) and strcpy(3), is deprecated to prevent buffer overflow exploits that leverage the above overrun effects. Instead, use snprintf(3) and strncpy(3).

12.4 Debug

Debug is important part of programing activities. Knowing how to debug programs makes you a good Debian user who can produce meaningful bug reports.

12.4.1 Basic gdb execution

Primary debugger on Debian is gdb(1) which enables you to inspect a program while it executes.

Let's install gdb and related programs by the following.

```
# apt-get install gdb gdb-doc build-essential devscripts
```

Good tutorial of gdb is provided by "info gdb" or found elsewhere on the web. Here is a simple example of using gdb(1) on a "program" compiled with the "-g" option to produce debugging information.

```
$ gdb program
(gdb) b 1                # set break point at line 1
(gdb) run args           # run program with args
(gdb) next               # next line
...
(gdb) step               # step forward
...
(gdb) p parm             # print parm
...
(gdb) p parm=12          # set value to 12
...
(gdb) quit
```

> **Tip**
> Many gdb(1) commands can be abbreviated. Tab expansion works as in the shell.

12.4.2 Debugging the Debian package

Since all installed binaries should be stripped on the Debian system by default, most debugging symbols are removed in the normal package. In order to debug Debian packages with gdb(1), either corresponding `*-dbg` packages or `*-dbgsym` packages need to be installed (e.g. `libc6-dbg` in the case of `libc6`, `coreutils-dbgsym` in the case of `coreutils`).

Old-style packages would provide its corresponding `*-dbg` package. It is placed directly inside Debian main archive alongside of the original package itself. For newer packages, they may generate `*-dbgsym` packages automatically when built and those debug packages are placed separately in debian-debug archive. Please refer to articles on Debian Wiki for more information.

If a package to be debugged does not provide either its `*-dbg` package or its `*-dbgsym` package, you need to install it after rebuilding it by the following.

```
$ mkdir /path/new ; cd /path/new
$ sudo apt-get update
$ sudo apt-get dist-upgrade
$ sudo apt-get install fakeroot devscripts build-essential
$ sudo apt-get build-dep source_package_name
$ apt-get source package_name
$ cd package_name*
```

Fix bugs if needed.

Bump package version to one which does not collide with official Debian versions, e.g. one appended with "+debug1" when recompiling existing package version, or one appended with "~pre1" when compiling unreleased package version by the following.

```
$ dch -i
```

Compile and install packages with debug symbols by the following.

```
$ export DEB_BUILD_OPTIONS=nostrip,noopt
$ debuild
$ cd ..
$ sudo debi package_name*.changes
```

You need to check build scripts of the package and ensure to use "CFLAGS=-g -Wall" for compiling binaries.

12.4.3 Obtaining backtrace

When you encounter program crash, reporting bug report with cut-and-pasted backtrace information is a good idea.

The backtrace can be obtained by the following steps.

- Run the program under gdb(1).

- Reproduce crash.

  – It causes you to be dropped back to the gdb prompt.

- Type "bt" at the gdb prompt.

In case of program freeze, you can crash the program by pressing Ctrl-C in the terminal running gdb to obtain gdb prompt.

Tip
Often, you see a backtrace where one or more of the top lines are in "malloc()" or "g_malloc()". When this happens, chances are your backtrace isn't very useful. The easiest way to find some useful information is to set the environment variable "$MALLOC_CHECK_" to a value of 2 (malloc(3)). You can do this while running gdb by doing the following.

```
$ MALLOC_CHECK_=2 gdb hello
```

12.4.4 Advanced gdb commands

command	description for command objectives
(gdb) thread apply all bt	get a backtrace for all threads for multi-threaded program
(gdb) bt full	get parameters came on the stack of function calls
(gdb) thread apply all bt full	get a backtrace and parameters as the combination of the preceding options
(gdb) thread apply all bt full 10	get a backtrace and parameters for top 10 calls to cut off irrelevant output
(gdb) set logging on	write log of gdb output to a file (the default is "gdb.txt")

Table 12.12: List of advanced gdb commands

12.4.5 Debugging X Errors

If a GNOME program preview1 has received an X error, you should see a message as follows.

```
The program 'preview1' received an X Window System error.
```

If this is the case, you can try running the program with "--sync", and break on the "gdk_x_error" function in order to obtain a backtrace.

12.4.6 Check dependency on libraries

Use ldd(1) to find out a program's dependency on libraries by the followings.

```
$ ldd /bin/ls
        librt.so.1 => /lib/librt.so.1 (0x4001e000)
        libc.so.6 => /lib/libc.so.6 (0x40030000)
        libpthread.so.0 => /lib/libpthread.so.0 (0x40153000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

For ls(1) to work in a `chroot`ed environment, the above libraries must be available in your `chroot`ed environment.

See Section 9.3.6.

12.4.7 Memory leak detection tools

There are several memory leak detection tools available in Debian.

12.4.8 Static code analysis tools

There are lint like tools for static code analysis.

package	popcon	size	description
libc6-dev	V:287, I:610	18226	mtrace(1): malloc debugging functionality in glibc
valgrind	V:8, I:54	72217	memory debugger and profiler
kmtrace	V:0, I:14	351	KDE memory leak tracer using glibc's mtrace(1)
electric-fence	V:0, I:5	70	malloc(3) debugger
leaktracer	V:0, I:2	56	memory-leak tracer for C++ programs
libdmalloc5	V:0, I:3	360	debug memory allocation library

Table 12.13: List of memory leak detection tools

package	popcon	size	description
splint	V:0, I:4	2239	tool for statically checking C programs for bugs
flawfinder	V:0, I:0	175	tool to examine C/C++ source code and looks for security weaknesses
perl	V:608, I:995	568	interpreter with internal static code checker: B::Lint(3perl)
pylint	V:7, I:17	1077	Python code static checker
weblint-perl	V:0, I:1	36	syntax and minimal style checker for HTML
linklint	V:0, I:1	343	fast link checker and web site maintenance tool
libxml2-utils	V:23, I:306	167	utilities with xmllint(1) to validate XML files

Table 12.14: List of tools for static code analysis

12.4.9   Disassemble binary

You can disassemble binary code with objdump(1) by the following.

```
$  objdump -m i386 -b binary -D /usr/lib/grub/x86_64-pc/stage1
```

Note
gdb(1) may be used to disassemble code interactively.

12.5   Flex —a better Lex

Flex is a Lex-compatible fast lexical analyzer generator.

Tutorial for flex(1) can be found in "`info flex`".

You need to provide your own "`main()`" and "`yywrap()`". Otherwise, your flex program should look like this to compile without a library. This is because that "`yywrap`" is a macro and "`%option main`" turns on "`%option noyywrap`" implicitly.

```
%option main
%%
.|\n    ECHO ;
%%
```

Alternatively, you may compile with the "`-lfl`" linker option at the end of your cc(1) command line (like AT&T-Lex with "`-ll`"). No "`%option`" is needed in this case.

12.6   Bison —a better Yacc

Several packages provide a Yacc-compatible lookahead LR parser or LALR parser generator in Debian.

package	popcon	size	description
bison	V:11, I:108	2061	GNU LALR parser generator
byacc	V:0, I:6	160	Berkeley LALR parser generator
btyacc	V:0, I:0	207	backtracking parser generator based on byacc

Table 12.15: List of Yacc-compatible LALR parser generators

Tutorial for bison(1) can be found in "info bison".

You need to provide your own "main()" and "yyerror()". "main()" calls "yyparse()" which calls "yylex()", usually created with Flex.

```
%%
```

```
%%
```

12.7   Autoconf

Autoconf is a tool for producing shell scripts that automatically configure software source code packages to adapt to many kinds of Unix-like systems using the entire GNU build system.

autoconf(1) produces the configuration script "configure". "configure" automatically creates a customized "Makefile" using the "Makefile.in" template.

12.7.1   Compile and install a program

> ⚠ **Warning**
> Do not overwrite system files with your compiled programs when installing them.

Debian does not touch files in "/usr/local/" or "/opt". So if you compile a program from source, install it into "/usr/local/" so it does not interfere with Debian.

```
$ cd src
$ ./configure --prefix=/usr/local
$ make
$ make install # this puts the files in the system
```

12.7.2   Uninstall program

If you have the original source and if it uses autoconf(1)/automake(1) and if you can remember how you configured it, execute as follows to uninstall the program.

```
$ ./configure "all-of-the-options-you-gave-it"
# make uninstall
```

Alternatively, if you are absolutely sure that the install process puts files only under "/usr/local/" and there is nothing important there, you can erase all its contents by the following.

```
# find /usr/local -type f -print0 | xargs -0 rm -f
```

If you are not sure where files are installed, you should consider using checkinstall(8) from the checkinstall package, which provides a clean path for the uninstall. It now supports to create a Debian package with "-D" option.

12.8 Perl short script madness

Although any AWK scripts can be automatically rewritten in Perl using a2p(1), one-liner AWK scripts are best converted to one-liner Perl scripts manually.

Let's think following AWK script snippet.

```
awk '($2=="1957") { print $3 }' |
```

This is equivalent to any one of the following lines.

```
perl -ne '@f=split; if ($f[1] eq "1957") { print "$f[2]\n"}' |
```

```
perl -ne 'if ((@f=split)[1] eq "1957") { print "$f[2]\n"}' |
```

```
perl -ne '@f=split; print $f[2] if ( $f[1]==1957 )' |
```

```
perl -lane 'print $F[2] if $F[1] eq "1957"' |
```

```
perl -lane 'print$F[2]if$F[1]eq+1957' |
```

The last one is a riddle. It took advantage of following Perl features.

* The whitespace is optional.

* The automatic conversion exists from number to the string.

See perlrun(1) for the command-line options. For more crazy Perl scripts, Perl Golf may be interesting.

12.9 Web

Basic interactive dynamic web pages can be made as follows.

* Queries are presented to the browser user using HTML forms.

* Filling and clicking on the form entries sends one of the following URL string with encoded parameters from the browser to the web server.

  – "http://www.foo.dom/cgi-bin/program.pl?VAR1=VAL1&VAR2=VAL2&VAR3=VAL3"
  – "http://www.foo.dom/cgi-bin/program.py?VAR1=VAL1&VAR2=VAL2&VAR3=VAL3"
  – "http://www.foo.dom/program.php?VAR1=VAL1&VAR2=VAL2&VAR3=VAL3"

* "%nn" in URL is replaced with a character with hexadecimal nn value.

* The environment variable is set as: "QUERY_STRING="VAR1=VAL1 VAR2=VAL2 VAR3=VAL3"".

* CGI program (any one of "program.*") on the web server executes itself with the environment variable "$QUERY_STRING".

* stdout of CGI program is sent to the web browser and is presented as an interactive dynamic web page.

For security reasons it is better not to hand craft new hacks for parsing CGI parameters. There are established modules for them in Perl and Python. PHP comes with these functionalities. When client data storage is needed, HTTP cookies are used. When client side data processing is needed, Javascript is frequently used.

For more, see the Common Gateway Interface, The Apache Software Foundation, and JavaScript.

Searching "CGI tutorial" on Google by typing encoded URL http://www.google.com/search?hl=en&ie=UTF-8&q=CGI+tutorial directly to the browser address is a good way to see the CGI script in action on the Google server.

12.10 The source code translation

There are programs to convert source codes.

package	popcon	size	keyword	description
perl	V:608, I:995	568	AWK→PERL	convert source codes from AWK to PERL: a2p(1)
f2c	V:0, I:8	430	FORTRAN→C	convert source codes from FORTRAN 77 to C/C++: f2c(1)
intel2gas	V:0, I:0	174	intel→gas	converter from NASM (Intel format) to the GNU Assembler (GAS)

Table 12.16: List of source code translation tools

12.11 Making Debian package

If you want to make a Debian package, read followings.

- Chapter 2 to understand the basic package system

- Section 2.7.13 to understand basic porting process

- Section 9.10.4 to understand basic chroot techniques

- debuild(1), pbuilder(1) and pdebuild(1)

- Section 12.4.2 for recompiling for debugging

- Debian New Maintainers' Guide as tutorial (the maint-guide package)

- Debian Developer's Reference (the developers-reference package)

- Debian Policy Manual (the debian-policy package)

- Guide for Debian Maintainers (the debmake-doc package)

There are packages such as debmake, dh-make, dh-make-perl, etc., which help packaging.

Appendix A

Appendix

Here are backgrounds of this document.

A.1 The Debian maze

The Linux system is a very powerful computing platform for a networked computer. However, learning how to use all its capabilities is not easy. Setting up the LPR printer queue with a non-PostScript printer was a good example of stumble points. (There are no issues anymore since newer installations use the new CUPS system.)

There is a complete, detailed map called the "SOURCE CODE". This is very accurate but very hard to understand. There are also references called HOWTO and mini-HOWTO. They are easier to understand but tend to give too much detail and lose the big picture. I sometimes have a problem finding the right section in a long HOWTO when I need a few commands to invoke.

I hope this "Debian Reference (version 2.72+9.0trisquel2)" (2019-11-15 20:38:45 UTC) provides a good starting direction for people in the Debian maze.

A.2 Copyright history

The Debian Reference was initiated by me, Osamu Aoki <osamu at debian dot org>, as a personal system administration memo. Many contents came from the knowledge I gained from the debian-user mailing list and other Debian resources.

Following a suggestion from Josip Rodin, who was very active with the Debian Documentation Project (DDP), "Debian Reference (version 1, 2001-2007)" was created as a part of DDP documents.

After 6 years, I realized that the original "Debian Reference (version 1)" was outdated and started to rewrite many contents. New "Debian Reference (version 2)" is released in 2008.

The tutorial contents can trace its origin and its inspiration in followings.

• "Linux User's Guide" by Larry Greenfield (December 1996)

  – obsoleted by "Debian Tutorial"

• "Debian Tutorial" by Havoc Pennington. (11 December, 1998)

  – partially written by Oliver Elphick, Ole Tetlie, James Treacy, Craig Sawyer, and Ivan E. Moore II
  – obsoleted by "Debian GNU/Linux: Guide to Installation and Usage"

• "Debian GNU/Linux: Guide to Installation and Usage" by John Goerzen and Ossama Othman (1999)

  – obsoleted by "Debian Reference (version 1)"

The package and archive description can trace some of their origin and their inspiration in following.

• "Debian FAQ" (March 2002 version, when this was maintained by Josip Rodin)

The other contents can trace some of their origin and their inspiration in following.

• "Debian Reference (version 1)" by Osamu Aoki (2001−2007)

  – obsoleted by the newer "Debian Reference (version 2)" in 2008.

The previous "Debian Reference (version 1)" was created with many contributors.

• the major contents contribution on network configuration topics by Thomas Hood

• significant contents contribution on X and VCS related topics by Brian Nelson

• the help on the build scripts and many content corrections by Jens Seidel

• extensive proofreading by David Sewell

• many contributions by the translators, contributors, and bug reporters

Many manual pages and info pages on the Debian system were used as the primary references to write this document. To the extent Osamu Aoki considered within the fair use, many parts of them, especially command definitions, were used as phrase pieces after careful editorial efforts to fit them into the style and the objective of this document.

The gdb debugger description was expanded using Debian wiki contents on backtrace with consent by Ari Pollak, Loïc Minier, and Dafydd Harries.

Contents of the current "Debian Reference (version 2.72+9.0trisquel2)" (2019-11-15 20:38:45 UTC) are mostly my own work except as mentioned above. These has been updated by the contributors too.

The author, Osamu Aoki, thanks all those who helped make this document possible.

A.3 Document format

The source of the English original document is currently written in AsciiDoc text files. AsciiDoc is used as convenience only since it is less typing than straight XML and supports table in the very intuitive format. You should think XML and PO files as real source files. Via build script, it is converted to DocBook XML format and automatically generated data are inserted to form a final Docbook XML source. This final Docbook XML source can be converted to HTML, epub, plain text, PostScript, and PDF. (Some formats may be skipped for distribution.)