Debian Reference

Osamu Aoki
This Debian Reference (version 2.105) (2023-11-02 11:58:17 UTC) is intended to provide a broad overview of the Debian system as a post-installation user’s guide. It covers many aspects of system administration through shell-command examples for non-developers.
### COLLABORATORS

<table>
<thead>
<tr>
<th>ACTION</th>
<th>NAME</th>
<th>DATE</th>
<th>SIGNATURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>WRITTEN BY</td>
<td>Osamu Aoki</td>
<td>November 2, 2023</td>
<td></td>
</tr>
</tbody>
</table>

### REVISION HISTORY

<table>
<thead>
<tr>
<th>NUMBER</th>
<th>DATE</th>
<th>DESCRIPTION</th>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Contents

## 1 GNU/Linux tutorials

1.1 Console basics ................................................................. 1
   1.1.1 The shell prompt ...................................................... 1
   1.1.2 The shell prompt under GUI ......................................... 2
   1.1.3 The root account ..................................................... 2
   1.1.4 The root shell prompt ................................................. 3
   1.1.5 GUI system administration tools .................................... 3
   1.1.6 Virtual consoles .......................................................... 3
   1.1.7 How to leave the command prompt .................................... 3
   1.1.8 How to shutdown the system .......................................... 4
   1.1.9 Recovering a sane console ............................................ 4
   1.1.10 Additional package suggestions for the newbie ....................... 4
   1.1.11 An extra user account ................................................ 5
   1.1.12 sudo configuration .................................................... 5
   1.1.13 Play time ............................................................... 6

1.2 Unix-like filesystem ....................................................... 6
   1.2.1 Unix file basics ........................................................ 6
   1.2.2 Filesystem internals .................................................. 7
   1.2.3 Filesystem permissions ............................................... 8
   1.2.4 Control of permissions for newly created files: umask ............... 10
   1.2.5 Permissions for groups of users (group) ............................. 11
   1.2.6 Timestamps .............................................................. 11
   1.2.7 Links .................................................................... 13
   1.2.8 Named pipes (FIFOs) .................................................. 14
   1.2.9 Sockets ................................................................. 14
   1.2.10 Device files ............................................................. 14
   1.2.11 Special device files .................................................. 15
   1.2.12 procfs and sysfs ........................................................ 15
   1.2.13 tmpfs ................................................................. 16

1.3 Midnight Commander (MC) .................................................... 16
## 1.3 Customization of MC
1.3.1 Customization of MC ................................................. 17
1.3.2 Starting MC ......................................................... 17
1.3.3 File manager in MC ................................................. 17
1.3.4 Command-line tricks in MC ......................................... 17
1.3.5 The internal editor in MC ........................................... 18
1.3.6 The internal viewer in MC .......................................... 18
1.3.7 Auto-start features of MC ........................................... 18
1.3.8 Virtual filesystem of MC ............................................ 19

## 1.4 The basic Unix-like work environment
1.4.1 The login shell ....................................................... 19
1.4.2 Customizing bash ..................................................... 20
1.4.3 Special key strokes .................................................. 20
1.4.4 Mouse operations .................................................... 20
1.4.5 The pager ............................................................. 21
1.4.6 The text editor ....................................................... 22
1.4.7 Setting a default text editor .......................... 22
1.4.8 Using vim ........................................................... 22
1.4.9 Recording the shell activities ........................... 23
1.4.10 Basic Unix commands ............................................ 23

## 1.5 The simple shell command
1.5.1 Command execution and environment variable ................. 24
1.5.2 The "$LANG" variable ............................................... 26
1.5.3 The "$PATH" variable ............................................... 27
1.5.4 The "$HOME" variable ............................................. 27
1.5.5 Command line options .............................................. 28
1.5.6 Shell glob ............................................................ 28
1.5.7 Return value of the command .................................... 29
1.5.8 Typical command sequences and shell redirection ............ 29
1.5.9 Command alias ....................................................... 30

## 1.6 Unix-like text processing
1.6.1 Unix text tools ...................................................... 31
1.6.2 Regular expressions ................................................. 32
1.6.3 Replacement expressions ............................................ 33
1.6.4 Global substitution with regular expressions .................. 33
1.6.5 Extracting data from text file table .............................. 34
1.6.6 Script snippets for piping commands .............................. 35
2 Debian package management

2.1 Debian package management prerequisites

2.1.1 Debian package management system

2.1.2 Package configuration

2.1.3 Basic precautions

2.1.4 Life with eternal upgrades

2.1.5 Debian archive basics

2.1.6 Debian is 100% free software

2.1.7 Package dependencies

2.1.8 The event flow of the package management

2.1.9 First response to package management troubles

2.1.10 How to pick Debian packages

2.1.11 How to cope with conflicting requirements

2.2 Basic package management operations

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

2.2.2 Basic package management operations with the command line

2.2.3 Interactive use of aptitude

2.2.4 Key bindings of aptitude

2.2.5 Package views under aptitude

2.2.6 Search method options with aptitude

2.2.7 The aptitude regex formula

2.2.8 Dependency resolution of aptitude

2.2.9 Package activity logs

2.3 Examples of aptitude operations

2.3.1 Seeking interesting packages

2.3.2 Listing packages with regex matching on package names

2.3.3 Browsing with the regex matching

2.3.4 Purging removed packages for good

2.3.5 Tidying auto/manual install status

2.3.6 System wide upgrade

2.4 Advanced package management operations

2.4.1 Advanced package management operations with command line

2.4.2 Verification of installed package files

2.4.3 Safeguarding for package problems

2.4.4 Searching on the package meta data

2.5 Debian package management internals

2.5.1 Archive meta data

2.5.2 Top level "Release" file and authenticity

2.5.3 Archive level "Release" files
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.5.4 Fetching of the meta data for the package</td>
<td>62</td>
</tr>
<tr>
<td>2.5.5 The package state for APT</td>
<td>63</td>
</tr>
<tr>
<td>2.5.6 The package state for aptitude</td>
<td>63</td>
</tr>
<tr>
<td>2.5.7 Local copies of the fetched packages</td>
<td>63</td>
</tr>
<tr>
<td>2.5.8 Debian package file names</td>
<td>63</td>
</tr>
<tr>
<td>2.5.9 The dpkg command</td>
<td>64</td>
</tr>
<tr>
<td>2.5.10 The update-alternatives command</td>
<td>65</td>
</tr>
<tr>
<td>2.5.11 The dpkg-statoverride command</td>
<td>65</td>
</tr>
<tr>
<td>2.5.12 The dpkg-divert command</td>
<td>65</td>
</tr>
<tr>
<td>2.6 Recovery from a broken system</td>
<td>65</td>
</tr>
<tr>
<td>2.6.1 Failed installation due to missing dependencies</td>
<td>66</td>
</tr>
<tr>
<td>2.6.2 Caching errors of the package data</td>
<td>66</td>
</tr>
<tr>
<td>2.6.3 Incompatibility with old user configuration</td>
<td>66</td>
</tr>
<tr>
<td>2.6.4 Different packages with overlapped files</td>
<td>66</td>
</tr>
<tr>
<td>2.6.5 Fixing broken package script</td>
<td>67</td>
</tr>
<tr>
<td>2.6.6 Rescue with the dpkg command</td>
<td>67</td>
</tr>
<tr>
<td>2.6.7 Recovering package selection data</td>
<td>68</td>
</tr>
<tr>
<td>2.7 Tips for the package management</td>
<td>68</td>
</tr>
<tr>
<td>2.7.1 Who uploaded the package?</td>
<td>68</td>
</tr>
<tr>
<td>2.7.2 Limiting download bandwidth for APT</td>
<td>68</td>
</tr>
<tr>
<td>2.7.3 Automatic download and upgrade of packages</td>
<td>69</td>
</tr>
<tr>
<td>2.7.4 Updates and Backports</td>
<td>69</td>
</tr>
<tr>
<td>2.7.5 External package archives</td>
<td>70</td>
</tr>
<tr>
<td>2.7.6 Packages from mixed source of archives without apt-pinning</td>
<td>70</td>
</tr>
<tr>
<td>2.7.7 Tweaking candidate version with apt-pinning</td>
<td>71</td>
</tr>
<tr>
<td>2.7.8 Blocking packages installed by &quot;Requires&quot;</td>
<td>72</td>
</tr>
<tr>
<td>2.7.9 Tracking testing with some packages from unstable</td>
<td>73</td>
</tr>
<tr>
<td>2.7.10 Tracking unstable with some packages from experimental</td>
<td>74</td>
</tr>
<tr>
<td>2.7.11 Emergency downgrading</td>
<td>74</td>
</tr>
<tr>
<td>2.7.12 The evqs package</td>
<td>75</td>
</tr>
<tr>
<td>2.7.13 Porting a package to the stable system</td>
<td>75</td>
</tr>
<tr>
<td>2.7.14 Proxy server for APT</td>
<td>76</td>
</tr>
<tr>
<td>2.7.15 More readings for the package management</td>
<td>77</td>
</tr>
</tbody>
</table>
## 3 The system initialization

3.1 An overview of the bootstrapping process

- 3.1.1 Stage 1: the UEFI
- 3.1.2 Stage 2: the boot loader
- 3.1.3 Stage 3: the mini-Debian system
- 3.1.4 Stage 4: the normal Debian system

3.2 Systemd init

- 3.2.1 The hostname
- 3.2.2 The filesystem
- 3.2.3 Network interface initialization
- 3.2.4 Cloud system initialization

3.3 The kernel message

3.4 The system message

3.5 System management

3.6 Other system monitors

3.7 Customizing systemd

- 3.7.1 Socket activation
- 3.8 The udev system

3.8.1 The kernel module initialization

## 4 Authentication and access controls

4.1 Normal Unix authentication

4.2 Managing account and password information

4.3 Good password

4.4 Creating encrypted password

4.5 PAM and NSS

- 4.5.1 Configuration files accessed by PAM and NSS
- 4.5.2 The modern centralized system management
- 4.5.3 "Why GNU su does not support the wheel group"
- 4.5.4 Stricter password rule

4.6 Security of authentication

- 4.6.1 Secure password on the Internet
- 4.6.2 Secure Shell
- 4.6.3 Extra security measures for the Internet
- 4.6.4 Securing the root password

4.7 Other access controls

- 4.7.1 sudo
- 4.7.2 PolicyKit
- 4.7.3 Restricting access to some server services
- 4.7.4 Linux security features
5 Network setup

5.1 The basic network infrastructure
   5.1.1 The hostname resolution
   5.1.2 The network interface name
   5.1.3 The network address range for the LAN
   5.1.4 The network device support

5.2 The modern network configuration for desktop
   5.2.1 GUI network configuration tools

5.3 The modern network configuration without GUI

5.4 The modern network configuration for cloud
   5.4.1 The modern network configuration for cloud with DHCP
   5.4.2 The modern network configuration for cloud with static IP
   5.4.3 The modern network configuration for cloud with Network Manger

5.5 The low level network configuration
   5.5.1 Iproute2 commands
   5.5.2 Safe low level network operations

5.6 Network optimization
   5.6.1 Finding optimal MTU
   5.6.2 WAN TCP optimization

5.7 Netfilter infrastructure

6 Network applications

6.1 Web browsers
   6.1.1 Spoofing the User-Agent string
   6.1.2 Browser extension

6.2 The mail system
   6.2.1 Email basics
   6.2.2 Modern mail service limitation
   6.2.3 Historic mail service expectation
   6.2.4 Mail transport agent (MTA)
      6.2.4.1 The configuration of exim4
      6.2.4.2 The configuration of postfix with SASL
      6.2.4.3 The mail address configuration
      6.2.4.4 Basic MTA operations

6.3 The remote access server and utilities (SSH)
   6.3.1 Basics of SSH
   6.3.2 User name on the remote host
   6.3.3 Connecting without remote passwords
   6.3.4 Dealing with alien SSH clients
9 System tips

9.1 The console tips
  9.1.1 Recording the shell activities cleanly
  9.1.2 The screen program
  9.1.3 Navigating around directories
  9.1.4 Readline wrapper
  9.1.5 Scanning the source code tree

9.2 Customizing vim
  9.2.1 Customizing vim with internal features
  9.2.2 Customizing vim with external packages

9.3 Data recording and presentation
  9.3.1 The log daemon
  9.3.2 Log analyzer
  9.3.3 Customized display of text data
  9.3.4 Customized display of time and date
  9.3.5 Colorized shell echo
  9.3.6 Colorized commands
  9.3.7 Recording the editor activities for complex repeats
  9.3.8 Recording the graphic image of an X application
  9.3.9 Recording changes in configuration files

9.4 Monitoring, controlling, and starting program activities
  9.4.1 Timing a process
  9.4.2 The scheduling priority
  9.4.3 The ps command
  9.4.4 The top command
  9.4.5 Listing files opened by a process
  9.4.6 Tracing program activities
  9.4.7 Identification of processes using files or sockets
  9.4.8 Repeating a command with a constant interval
  9.4.9 Repeating a command looping over files
  9.4.10 Starting a program from GUI
  9.4.11 Customizing program to be started
  9.4.12 Killing a process
  9.4.13 Scheduling tasks once
  9.4.14 Scheduling tasks regularly
  9.4.15 Alt-SysRq key

9.5 System maintenance tips
  9.5.1 Who is on the system?
  9.5.2 Warning everyone
9.8.3 Data redundancy ................................................. 168
9.8.4 Data file recovery and forensic analysis ..................... 168
9.8.5 Splitting a large file into small files ......................... 169
9.8.6 Clearing file contents ........................................... 169
9.8.7 Dummy files ...................................................... 170
9.8.8 Erasing an entire hard disk ................................... 170
9.8.9 Erasing unused area of an hard disk ............................ 170
9.8.10 Undeleting deleted but still open files ....................... 171
9.8.11 Searching all hardlinks ...................................... 171
9.8.12 Invisible disk space consumption ........................... 172
9.9 Data encryption tips .............................................. 172
9.9.1 Removable disk encryption with dm-crypt/LUKS ............ 172
9.9.2 Mounting encrypted disk with dm-crypt/LUKS ................ 173
9.10 The kernel ......................................................... 173
9.10.1 Kernel parameters .............................................. 173
9.10.2 Kernel headers .................................................. 174
9.10.3 Compiling the kernel and related modules ................... 174
9.10.4 Compiling the kernel source: Debian Kernel Team recommendation ................................................. 175
9.10.5 Hardware drivers and firmware ............................... 175
9.11 Virtualized system ................................................ 176
9.11.1 Virtualization and emulation tools ............................ 176
9.11.2 Virtualization workflow ....................................... 178
9.11.3 Mounting the virtual disk image file ........................ 178
9.11.4 Chroot system ................................................... 179
9.11.5 Multiple desktop systems .................................... 180
10 Data management .................................................. 181
10.1 Sharing, copying, and archiving .................................. 181
10.1.1 Archive and compression tools ................................ 182
10.1.2 Copy and synchronization tools ............................... 183
10.1.3 Idioms for the archive ......................................... 183
10.1.4 Idioms for the copy ............................................. 184
10.1.5 Idioms for the selection of files ............................... 184
10.1.6 Archive media .................................................... 185
10.1.7 Removable storage device ..................................... 186
10.1.8 Filesystem choice for sharing data ......................... 187
10.1.9 Sharing data via network ...................................... 189
10.2 Backup and recovery .............................................. 189
10.2.1 Backup and recovery policy ................................. 189
11.3.4 Creating a manual page ................................................................. 211
11.4 Printable data ............................................................................. 212
   11.4.1 Ghostscript ........................................................................... 212
   11.4.2 Merge two PS or PDF files .................................................... 212
   11.4.3 Printable data utilities ........................................................... 212
   11.4.4 Printing with CUPS ............................................................... 213
11.5 The mail data conversion .............................................................. 213
   11.5.1 Mail data basics ................................................................. 213
11.6 Graphic data tools ....................................................................... 214
11.7 Miscellaneous data conversion .................................................... 216

12 Programming ...................................................... 217
   12.1 The shell script ................................................................. 217
      12.1.1 POSIX shell compatibility ................................................... 218
      12.1.2 Shell parameters ............................................................. 218
      12.1.3 Shell conditionals ............................................................ 219
      12.1.4 Shell loops .................................................................... 220
      12.1.5 Shell environment variables ............................................. 220
      12.1.6 The shell command-line processing sequence ............ 221
      12.1.7 Utility programs for shell script ...................................... 222
   12.2 Scripting in interpreted languages ........................................... 222
      12.2.1 Debugging interpreted language codes ......................... 223
      12.2.2 GUI program with the shell script ................................... 223
      12.2.3 Custom actions for GUI filer ........................................... 224
      12.2.4 Perl short script madness ................................................ 224
   12.3 Coding in compiled languages ............................................... 225
      12.3.1 C ............................................................................. 225
      12.3.2 Simple C program (gcc) ................................................... 225
      12.3.3 Flex — a better Lex .......................................................... 226
      12.3.4 Bison — a better Yacc ..................................................... 226
   12.4 Static code analysis tools ........................................................... 228
   12.5 Debug .............................................................. 229
      12.5.1 Basic gdb execution ......................................................... 229
      12.5.2 Debugging the Debian package ....................................... 229
      12.5.3 Obtaining backtrace ......................................................... 230
      12.5.4 Advanced gdb commands ............................................... 231
      12.5.5 Check dependency on libraries ....................................... 231
      12.5.6 Dynamic call tracing tools ............................................... 231
      12.5.7 Debugging X Errors ......................................................... 231
12.5.8 Memory leak detection tools .......................................................... 231
12.5.9 Disassemble binary ........................................................................... 231
12.6 Build tools ........................................................................................... 232
12.6.1 Make ................................................................................................. 232
12.6.2 Autotools .......................................................................................... 232
   12.6.2.1 Compile and install a program ....................................................... 233
   12.6.2.2 Uninstall program ....................................................................... 233
12.6.3 Meson ............................................................................................... 234
12.7 Web ....................................................................................................... 234
12.8 The source code translation .................................................................. 234
12.9 Making Debian package ........................................................................ 235

A Appendix ................................................................................................ 236
   A.1 The Debian maze .............................................................................. 236
   A.2 Copyright history ............................................................................... 236
   A.3 Document format .............................................................................. 237
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>List of interesting text-mode program packages</td>
<td>4</td>
</tr>
<tr>
<td>1.2</td>
<td>List of informative documentation packages</td>
<td>5</td>
</tr>
<tr>
<td>1.3</td>
<td>List of usage of key directories</td>
<td>7</td>
</tr>
<tr>
<td>1.4</td>
<td>List of the first character of “ls -l” output</td>
<td>9</td>
</tr>
<tr>
<td>1.5</td>
<td>The numeric mode for file permissions in <code>chmod(1)</code> commands</td>
<td>10</td>
</tr>
<tr>
<td>1.6</td>
<td>The <code>umask</code> value examples</td>
<td>10</td>
</tr>
<tr>
<td>1.7</td>
<td>List of notable system-provided groups for file access</td>
<td>11</td>
</tr>
<tr>
<td>1.8</td>
<td>List of notable system-provided groups for particular command executions</td>
<td>11</td>
</tr>
<tr>
<td>1.9</td>
<td>List of types of timestamps</td>
<td>12</td>
</tr>
<tr>
<td>1.10</td>
<td>List of special device files</td>
<td>15</td>
</tr>
<tr>
<td>1.11</td>
<td>The key bindings of MC</td>
<td>17</td>
</tr>
<tr>
<td>1.12</td>
<td>The reaction to the enter key in MC</td>
<td>19</td>
</tr>
<tr>
<td>1.13</td>
<td>List of shell programs</td>
<td>19</td>
</tr>
<tr>
<td>1.14</td>
<td>List of key bindings for bash</td>
<td>21</td>
</tr>
<tr>
<td>1.15</td>
<td>List of mouse operations and related key actions on Debian</td>
<td>21</td>
</tr>
<tr>
<td>1.16</td>
<td>List of basic Vim key strokes</td>
<td>23</td>
</tr>
<tr>
<td>1.17</td>
<td>List of basic Unix commands</td>
<td>25</td>
</tr>
<tr>
<td>1.18</td>
<td>The 3 parts of locale value</td>
<td>26</td>
</tr>
<tr>
<td>1.19</td>
<td>List of locale recommendations</td>
<td>26</td>
</tr>
<tr>
<td>1.20</td>
<td>List of “$HOME” values</td>
<td>27</td>
</tr>
<tr>
<td>1.21</td>
<td>Shell glob patterns</td>
<td>28</td>
</tr>
<tr>
<td>1.22</td>
<td>Command exit codes</td>
<td>29</td>
</tr>
<tr>
<td>1.23</td>
<td>Shell command idioms</td>
<td>29</td>
</tr>
<tr>
<td>1.24</td>
<td>Predefined file descriptors</td>
<td>30</td>
</tr>
<tr>
<td>1.25</td>
<td>Metacharacters for BRE and ERE</td>
<td>32</td>
</tr>
<tr>
<td>1.26</td>
<td>The replacement expression</td>
<td>33</td>
</tr>
<tr>
<td>1.27</td>
<td>List of script snippets for piping commands</td>
<td>36</td>
</tr>
<tr>
<td>2.1</td>
<td>List of Debian package management tools</td>
<td>38</td>
</tr>
<tr>
<td>2.2</td>
<td>List of Debian archive sites</td>
<td>41</td>
</tr>
</tbody>
</table>
2.3 List of Debian archive area .......................................................... 42
2.4 The relationship between suite and codename ............................ 42
2.5 List of key web site to resolving problems with a specific package ... 47
2.6 Basic package management operations with the commandline using apt(8), aptitude(8) and apt-get(8) /apt-cache(8) ............................. 50
2.7 Notable command options for aptitude(8) ................................. 50
2.8 List of key bindings for aptitude ............................................... 51
2.9 List of views for aptitude ......................................................... 52
2.10 The categorization of standard package views ............................ 53
2.11 List of the aptitude regex formula ........................................... 54
2.12 The log files for package activities ........................................ 55
2.13 List of advanced package management operations .................... 59
2.14 The content of the Debian archive meta data ............................. 60
2.15 The name structure of Debian packages .................................. 63
2.16 The usable characters for each component in the Debian package names ................................................................. 64
2.17 The notable files created by dpkg ............................................ 64
2.18 List of notable Pin-Priority values for apt-pinning technique......... 72
2.19 List of the proxy tools specially for Debian archive .................. 76

3.1 List of boot loaders ..................................................................... 79
3.2 The meaning of the menu entry of the above part of /boot/grub/grub.cfg .......................................................... 80
3.3 List of boot utilities for the Debian system ............................... 81
3.4 List of kernel error levels ....................................................... 83
3.5 List of typical journalctl command snippets ........................... 84
3.6 List of typical systemctl command snippets ........................... 85
3.7 List of other monitoring command snippets under systemd ....... 86

4.1 3 important configuration files for pam_unix(8) ......................... 88
4.2 The second entry content of "/etc/passwd" ................................ 89
4.3 List of commands to manage account information ..................... 90
4.4 List of tools to generate password ......................................... 91
4.5 List of notable PAM and NSS systems ................................... 91
4.6 List of configuration files accessed by PAM and NSS ................ 92
4.7 List of insecure and secure services and ports ......................... 94
4.8 List of tools to provide extra security measures ....................... 95

5.1 List of network configuration tools ........................................... 99
5.2 List of network address ranges .............................................. 101
5.3 Translation table from obsolete net-tools commands to new iproute2 commands ........................................... 104
5.4 List of low level network commands ...................................... 104
11.6 List of tools to highlight plain text data .......................................................... 207
11.7 List of predefined entities for XML ................................................................. 208
11.8 List of XML tools ......................................................................................... 209
11.9 List of DSSSL tools .................................................................................. 209
11.10 List of XML data extraction tools ............................................................... 209
11.11 List of XML pretty print tools ................................................................. 210
11.12 List of typesetting tools ........................................................................ 210
11.13 List of packages to help creating the manpage ............................................. 212
11.14 List of Ghostscript PostScript interpreters ................................................. 212
11.15 List of printable data utilities ..................................................................... 213
11.16 List of packages to help mail data conversion ............................................ 214
11.17 List of graphic data tools ......................................................................... 215
11.18 List of miscellaneous data conversion tools ............................................. 216

12.1 List of typical bashisms ............................................................................ 218
12.2 List of shell parameters ............................................................................. 218
12.3 List of shell parameter expansions ............................................................ 219
12.4 List of key shell parameter substitutions ................................................... 219
12.5 List of file comparison operators in the conditional expression ................. 220
12.6 List of string comparison operators in the conditional expression .......... 220
12.7 List of packages containing small utility programs for shell scripts ............ 222
12.8 List of interpreter related packages ........................................................... 222
12.9 List of dialog programs ........................................................................... 223
12.10 List of compiler related packages ............................................................. 225
12.11 List of Yacc-compatible LALR parser generators ...................................... 226
12.12 List of tools for static code analysis .......................................................... 228
12.13 List of debug packages .......................................................................... 229
12.14 List of advanced gdb commands ............................................................. 231
12.15 List of memory leak detection tools .......................................................... 232
12.16 List of build tool packages ..................................................................... 232
12.17 List of make automatic variables .............................................................. 233
12.18 List of make variable expansions ............................................................. 233
12.19 List of source code translation tools ......................................................... 235
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 2.105) (2023-11-02 11:58:17 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 testing 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 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 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 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/
• The Debian Administrator’s Handbook
• The HOWTOs from The Linux Documentation Project (TLDP) at http://tldp.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.

All command examples are run under the English locale "LANG=en_US.UTF8". Please don't expect the placeholder strings such as `command-in-root-account` and `command-in-user-account` to be translated in command examples. This is an intentional choice to keep all translated examples to be up-to-date.

**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.

```bash
$ 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-and-groups.html"` or by its URL, such as [https://www.debian.org](https://www.debian.org). You are encouraged to read the documentation by typing the following.

```bash
$ zcat "/usr/share/doc/doc/base-passwd/users-and-groups.txt.gz" | sensible-pager
$ sensible-browser ""/usr/share/doc/doc/base-passwd/users-and-groups.html"
```

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.

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

### The popcon

The **popcon** data is presented as the objective measure for the popularity of each package. It was downloaded on 2023-11-01 15:07:51 UTC and contains the total submission of 230686 reports over 195209 binary packages and 27 architectures.

**Note**

Please note that the amd64 unstable archive contains only 70981 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 any GUI environment such as GNOME or KDE desktop system. Suppose your hostname is `foo`, the login prompt looks as follows.

If you installed a GUI environment, then you can still get to the character based login prompt by Ctrl-Alt-F3, and you can return to the GUI environment via Ctrl-Alt-F2 (see Section 1.1.6 below for more).

```
foo login:
```

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 11 foo tty1
foo login: penguin
Password:
```
Now you are in the **shell**. The shell interprets your commands.

### 1.1.2 The shell prompt under GUI

If you installed a **GUI** environment 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 primary click of the mouse.

You can gain the shell prompt under GUI environment by starting a `x-terminal-emulator` program such as `gnome-terminal(1)`, `rxvt(1)` or `xterm(1)`. Under the GNOME desktop environment, press SUPER-key (Windows-key) and typing in “terminal” to the search prompt does the trick.

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.
- 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 terminal emulator, such as `gnome-terminal(1)`, `rxvt(1)`, or `xterm(1)`. See Section 1.1.4 and Section 7.8.

---

**Warning**

Never start the GUI display/session manager under the root account by typing in `root` to the prompt of the display manager such as `gdm3(1)`. 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 in the GUI environment, you gain access to the character console 3 by pressing `Ctrl-Alt-F3` key, i.e., the `left-Ctrl-key`, the `left-Alt-key`, and the `F3-key` are pressed together. You can get back to the GUI environment, normally running on the virtual console 2, by pressing `Alt-F2`.

You can alternatively change to another virtual console, e.g. to the console 3, from the command line.

```
# chvt 3
```

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, `^D`, 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
```

See Section 6.3.8.

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 aptitude
...
```

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

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc</td>
<td>V:50, I:212</td>
<td>1490</td>
<td>A text-mode full-screen file manager</td>
</tr>
<tr>
<td>sudo</td>
<td>V:676, I:835</td>
<td>6222</td>
<td>A program to allow limited root privileges to users</td>
</tr>
<tr>
<td>vim</td>
<td>V:95, I:370</td>
<td>3738</td>
<td>Unix text editor Vi IMPproved, a programmers text editor (standard version)</td>
</tr>
<tr>
<td>vim-tiny</td>
<td>V:57, I:974</td>
<td>1731</td>
<td>Unix text editor Vi IMPproved, a programmers text editor (compact version)</td>
</tr>
<tr>
<td>emacs-nox</td>
<td>V:3, I:16</td>
<td>35109</td>
<td>GNU project Emacs, the Lisp based extensible text editor</td>
</tr>
<tr>
<td>w3m</td>
<td>V:14, I:186</td>
<td>2828</td>
<td>Text-mode WWW browsers</td>
</tr>
<tr>
<td>gpm</td>
<td>V:10, I:12</td>
<td>521</td>
<td>The Unix style cut-and-paste on the text console (daemon)</td>
</tr>
</tbody>
</table>

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
```
### Table 1.2: List of informative documentation packages

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>doc-debian</td>
<td>I:865</td>
<td>187</td>
<td>Debian Project documentation, (Debian FAQ) and other documents</td>
</tr>
<tr>
<td>debian-policy</td>
<td>I:16</td>
<td>4379</td>
<td>Debian Policy Manual and related documents</td>
</tr>
<tr>
<td>developers-reference</td>
<td>V:0, I:5</td>
<td>2604</td>
<td>Guidelines and information for Debian developers</td>
</tr>
<tr>
<td>debmake-doc</td>
<td>I:0</td>
<td>11992</td>
<td>Guide for Debian Maintainers</td>
</tr>
<tr>
<td>debian-history</td>
<td>I:0</td>
<td>4692</td>
<td>History of the Debian Project</td>
</tr>
<tr>
<td>debian-faq</td>
<td>I:863</td>
<td>790</td>
<td>Debian FAQ</td>
</tr>
</tbody>
</table>

### 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.

```bash
# 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.

```bash
# 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.

```bash
# 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.

```bash
# 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. 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).
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-policy/fhs/fhs-2.3.txt") and **hier(7)**. You should remember the following facts as the starter.

<table>
<thead>
<tr>
<th>directory</th>
<th>usage of the directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>the root directory</td>
</tr>
<tr>
<td>/etc/</td>
<td>system wide configuration files</td>
</tr>
<tr>
<td>/var/log/</td>
<td>system log files</td>
</tr>
<tr>
<td>/home/</td>
<td>all the home directories for all non-privileged users</td>
</tr>
</tbody>
</table>

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)
<table>
<thead>
<tr>
<th>character</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>normal file</td>
</tr>
<tr>
<td>d</td>
<td>directory</td>
</tr>
<tr>
<td>l</td>
<td>symlink</td>
</tr>
<tr>
<td>c</td>
<td>character device node</td>
</tr>
<tr>
<td>b</td>
<td>block device node</td>
</tr>
<tr>
<td>p</td>
<td>named pipe</td>
</tr>
<tr>
<td>s</td>
<td>socket</td>
</tr>
</tbody>
</table>

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--  root root 2761 Aug 30 10:38 /etc/passwd
-rw-r----- 1 root shadow 1695 Aug 30 10:38 /etc/shadow
-rw-r--r-- 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
drwxrwxrwt 10 root staff 4096 Sep 29 22:50 /usr/local
drwxrwxrwx 10 root root 4096 Oct 11 00:28 /usr/src
drwxrwxrwx 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.

<table>
<thead>
<tr>
<th>digit</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st optional digit</td>
<td>sum of set user ID (=4), set group ID (=2), and sticky bit (=1)</td>
</tr>
<tr>
<td>2nd digit</td>
<td>sum of read (=4), write (=2), and execute (=1) permissions for user</td>
</tr>
<tr>
<td>3rd digit</td>
<td>ditto for group</td>
</tr>
<tr>
<td>4th digit</td>
<td>ditto for other</td>
</tr>
</tbody>
</table>

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 builtins 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)`.

\[
\text{(file permissions)} = (\text{requested file permissions}) \& \neg (\text{umask value})
\]

<table>
<thead>
<tr>
<th>umask</th>
<th>file permissions created</th>
<th>directory permissions created</th>
<th>usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>0022</td>
<td>-rw-r--r--</td>
<td>-rwxr-xr-x</td>
<td>writable only by the user</td>
</tr>
<tr>
<td>0002</td>
<td>-rw-r--r--</td>
<td>-rwxrwxr-x</td>
<td>writable by the group</td>
</tr>
</tbody>
</table>

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.

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

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.

<table>
<thead>
<tr>
<th>group</th>
<th>accessible commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>sudo</td>
<td>execute sudo without their password</td>
</tr>
<tr>
<td>lpadmin</td>
<td>execute commands to add, modify, and remove printers from printer databases</td>
</tr>
</tbody>
</table>

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.
### Table 1.9: List of types of timestamps

<table>
<thead>
<tr>
<th>type</th>
<th>meaning (historic Unix definition)</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtime</td>
<td>the file modification time (ls -l)</td>
</tr>
<tr>
<td>ctime</td>
<td>the file status change time (ls -lc)</td>
</tr>
<tr>
<td>atime</td>
<td>the last file access time (ls -lu)</td>
</tr>
</tbody>
</table>

**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 localized strings under non-English locale ("fr_FR.UTF-8").

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

**Tip**

See Section 9.3.4 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

```bash
$ 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.

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

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/linux-doc-*/Documentation/filesystems/*") 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/filesystems/*") 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 \texttt{cd} to the directory, I suggest to modify "\texttt{~/.bashrc}" to include a script provided by the \texttt{mc} package.

\begin{verbatim}
. /usr/lib/mc/mc.sh
\end{verbatim}

See \texttt{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.

\textbf{Note}

In some consoles such as \texttt{gnome-terminal(1)}, key strokes of function-keys may be stolen by the console program. You can disable these features in "Preferences" → "General" and "Shortcuts" menu for \texttt{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 9.5.6.

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 \texttt{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.)

\begin{tabular}{|l|l|}
\hline
key & key binding \\
\hline
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 \\
\hline
\end{tabular}

\textbf{Table 1.11:} The key bindings of MC

1.3.4 Command-line tricks in MC

- \texttt{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.

```bash
$ 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-Fn** 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**".

```bash
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.

```bash
$ 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 9.4.11). This is a very convenient MC feature.

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 Virtual filesystem of MC

MC can be used to access files over the Internet. Go to the menu by pressing F9, "Enter" and "h" to activate the Shell filesystem. Enter a URL in the form "sh://[user@]machine[:options]/[remote-dir]", which retrieves a remote directory that appears like a local one using ssh.

### 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

Since the login shell may be used by some system initialization programs, it is prudent to keep it as bash(1) and avoid switching the login shell with chsh(1).

If you want to use a different interactive shell prompt, set it from GUI terminal emulator configuration or start it from ~/.bashrc, e.g., by placing "exec /usr/bin/zsh -i -l" or "exec /usr/bin/fish -i -l" in it.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>POSIX shell</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bash</td>
<td>V:830,I:999</td>
<td>7176</td>
<td>Yes</td>
<td>Bash: the GNU Bourne Again SHell (de facto standard)</td>
</tr>
<tr>
<td>bash-completion</td>
<td>V:32,I:932</td>
<td>1454</td>
<td>N/A</td>
<td>programmable completion for the bash shell</td>
</tr>
<tr>
<td>dash</td>
<td>V:881,I:996</td>
<td>191</td>
<td>Yes</td>
<td>Debian Almquist Shell, good for shell script</td>
</tr>
<tr>
<td>zsh</td>
<td>V:39,I:72</td>
<td>2462</td>
<td>Yes</td>
<td>Z shell: the standard shell with many enhancements</td>
</tr>
<tr>
<td>tcsh</td>
<td>V:6,I:21</td>
<td>1355</td>
<td>No</td>
<td>TENEX C Shell: an enhanced version of Berkeley csh</td>
</tr>
<tr>
<td>mksh</td>
<td>V:4,I:10</td>
<td>1566</td>
<td>Yes</td>
<td>A version of the Korn shell</td>
</tr>
<tr>
<td>csh</td>
<td>V:1,I:6</td>
<td>339</td>
<td>No</td>
<td>OpenBSD C Shell, a version of Berkeley csh</td>
</tr>
<tr>
<td>sash</td>
<td>V:0,I:5</td>
<td>1157</td>
<td>Yes</td>
<td>Stand-alone shell with built-in commands (Not meant for standard &quot;/bin/sh&quot;)</td>
</tr>
<tr>
<td>ksh</td>
<td>V:1,I:11</td>
<td>61</td>
<td>Yes</td>
<td>the real, AT&amp;T version of the Korn shell</td>
</tr>
<tr>
<td>rc</td>
<td>V:0,I:1</td>
<td>178</td>
<td>No</td>
<td>implementation of the AT&amp;T Plan 9 rc shell</td>
</tr>
<tr>
<td>posh</td>
<td>V:0,I:0</td>
<td>190</td>
<td>Yes</td>
<td>Policy-compliant Ordinary SHell (pdksh derivative)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>POSIX shell</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bash</td>
<td>V:830,I:999</td>
<td>7176</td>
<td>Yes</td>
<td>Bash: the GNU Bourne Again SHell (de facto standard)</td>
</tr>
<tr>
<td>bash-completion</td>
<td>V:32,I:932</td>
<td>1454</td>
<td>N/A</td>
<td>programmable completion for the bash shell</td>
</tr>
<tr>
<td>dash</td>
<td>V:881,I:996</td>
<td>191</td>
<td>Yes</td>
<td>Debian Almquist Shell, good for shell script</td>
</tr>
<tr>
<td>zsh</td>
<td>V:39,I:72</td>
<td>2462</td>
<td>Yes</td>
<td>Z shell: the standard shell with many enhancements</td>
</tr>
<tr>
<td>tcsh</td>
<td>V:6,I:21</td>
<td>1355</td>
<td>No</td>
<td>TENEX C Shell: an enhanced version of Berkeley csh</td>
</tr>
<tr>
<td>mksh</td>
<td>V:4,I:10</td>
<td>1566</td>
<td>Yes</td>
<td>A version of the Korn shell</td>
</tr>
<tr>
<td>csh</td>
<td>V:1,I:6</td>
<td>339</td>
<td>No</td>
<td>OpenBSD C Shell, a version of Berkeley csh</td>
</tr>
<tr>
<td>sash</td>
<td>V:0,I:5</td>
<td>1157</td>
<td>Yes</td>
<td>Stand-alone shell with built-in commands (Not meant for standard &quot;/bin/sh&quot;)</td>
</tr>
<tr>
<td>ksh</td>
<td>V:1,I:11</td>
<td>61</td>
<td>Yes</td>
<td>the real, AT&amp;T version of the Korn shell</td>
</tr>
<tr>
<td>rc</td>
<td>V:0,I:1</td>
<td>178</td>
<td>No</td>
<td>implementation of the AT&amp;T Plan 9 rc shell</td>
</tr>
<tr>
<td>posh</td>
<td>V:0,I:0</td>
<td>190</td>
<td>Yes</td>
<td>Policy-compliant Ordinary SHell (pdksh derivative)</td>
</tr>
</tbody>
</table>

#### 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.

```bash
# 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"
fi
export PATH

EDITOR=vim
export EDITOR
```

**Tip**

You can find more bash customization tips, such as Section 9.3.6, 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.

**Tip**

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

1.4.4 Mouse operations

Mouse operations for text on Debian system mix 2 styles with some twists:

- Traditional Unix style mouse operations:
  - use 3 buttons (click)
### Table 1.14: List of key bindings for bash

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

### Table 1.15: List of mouse operations and related key actions on Debian

<table>
<thead>
<tr>
<th>action</th>
<th>response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left-click-and-drag mouse</td>
<td>select range as PRIMARY selection</td>
</tr>
<tr>
<td>Left-click</td>
<td>select the start of range for PRIMARY selection</td>
</tr>
<tr>
<td>Right-click (traditional)</td>
<td>select the end of range for PRIMARY selection</td>
</tr>
<tr>
<td>Right-click (modern)</td>
<td>context dependent menu (cut/copy/paste)</td>
</tr>
<tr>
<td>Middle-click or Shift-Ins</td>
<td>insert PRIMARY selection at the cursor</td>
</tr>
<tr>
<td>Ctrl-X</td>
<td>cut PRIMARY selection to CLIPBOARD</td>
</tr>
<tr>
<td>Ctrl-C (Shift-Ctrl-C in terminal)</td>
<td>copy PRIMARY selection to CLIPBOARD</td>
</tr>
<tr>
<td>Ctrl-V</td>
<td>paste CLIPBOARD at the cursor</td>
</tr>
</tbody>
</table>

Here, the PRIMARY selection is the highlighted text range. Within the terminal program, Shift-Ctrl-C is used instead to avoid terminating a running program.

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/less/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 GUI 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 GUI 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 9.4.11). For the consistency on the Debian system, set these to "/usr/bin/editor". (Historically, "$EDITOR" was "ed" and "$VISUAL" was "vi".)

1.4.8 Using vim

The recent vim(1) starts itself in the same "nocompatible" option and enters into the NORMAL mode.

Please use the "vimitutor" program to learn vim through an interactive tutorial course.

The vim program changes its behavior to typed key strokes based on mode. Typing in key strokes to the buffer is mostly done in INSERT-mode and REPLACE-mode. Moving cursor is mostly done in NORMAL-mode. Interactive selection is done in VISUAL-mode. Typing ":" in NORMAL-mode changes its mode to Ex-mode. Ex-mode accepts commands.

Tip
The Vim comes with the Netrw package. Netrw supports reading files, writing files, browsing directories over a network, and local browsing! Try Netrw with "vim ." (a period as the argument) and read its manual at ":help netrw".

1 Even the older vim can starts in the same "nocompatible" mode by starting it with the "-N" option.
<table>
<thead>
<tr>
<th>mode</th>
<th>key strokes</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORMAL</td>
<td>:help</td>
<td>only</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:e filename.ext</td>
<td>open new buffer to edit filename.ext</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:w filename.ext</td>
<td>overwrite current buffer to the original file</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:w filename.ext</td>
<td>write current buffer to filename.ext</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:q</td>
<td>quit vim</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:q!</td>
<td>force to quit vim</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:only</td>
<td>close all other split open windows</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:set nocompatible?</td>
<td>check if vim is in the same nocompatible mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>:set nocompatible</td>
<td>set vim to the same nocompatible mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>l</td>
<td>enter the INSERT mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>R</td>
<td>enter the REPLACE mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>V</td>
<td>enter the VISUAL mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>Ctrl-V</td>
<td>enter the linewise VISUAL mode</td>
</tr>
<tr>
<td>except TERMINAL-JOB</td>
<td>ESC-key</td>
<td>enter the NORMAL mode</td>
</tr>
<tr>
<td>NORMAL</td>
<td>term</td>
<td>enter the TERMINAL-JOB mode</td>
</tr>
<tr>
<td>TERMINAL-NORMAL</td>
<td>l</td>
<td>enter the TERMINAL-JOB mode</td>
</tr>
<tr>
<td>TERMINAL-JOB</td>
<td>Ctrl-W N (or Ctrl-\ Ctrl-N)</td>
<td>enter the TERMINAL-NORMAL mode</td>
</tr>
<tr>
<td>TERMINAL-JOB</td>
<td>Ctrl-W :</td>
<td>enter the Ex-mode in TERMINAL-NORMAL mode</td>
</tr>
</tbody>
</table>

Table 1.16: List of basic Vim key strokes

For the advanced configuration of vim, see Section 9.2.

### 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.

**Tip**
The new Vim (version>=8.2) can be used to record the shell activities cleanly using TERMINAL-JOB-mode. See Section 1.4.8.

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 9.1.1.

### 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.

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

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.

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:

```bash
$ man man
debian-reference  man  24 / 237
$ 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).

```bash
$ 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.

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.
<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pwd</td>
<td>display name of current/working directory</td>
</tr>
<tr>
<td>whoami</td>
<td>display current user name</td>
</tr>
<tr>
<td>id</td>
<td>display current user identity (name, uid, gid, and associated groups)</td>
</tr>
<tr>
<td>file foo</td>
<td>display a type of file for the file “foo”</td>
</tr>
<tr>
<td>type -p commandname</td>
<td>display a file location of command “commandname”</td>
</tr>
<tr>
<td>which commandname</td>
<td>display information on command “commandname”</td>
</tr>
<tr>
<td>apropos key-word</td>
<td>find commands related to “key-word”</td>
</tr>
<tr>
<td>man -k key-word</td>
<td>display one line explanation on command “commandname”</td>
</tr>
<tr>
<td>whatis commandname</td>
<td>display explanation on command “commandname” (Unix style)</td>
</tr>
<tr>
<td>info commandname</td>
<td>display rather long explanation on command “commandname” (GNU style)</td>
</tr>
<tr>
<td>ls</td>
<td>list contents of directory (non-dot files and directories)</td>
</tr>
<tr>
<td>ls -a</td>
<td>list contents of directory (all files and directories)</td>
</tr>
<tr>
<td>ls -A</td>
<td>list contents of directory (almost all files and directories, i.e., skip “..” and “.”)</td>
</tr>
<tr>
<td>ls -la</td>
<td>list all contents of directory with detail information</td>
</tr>
<tr>
<td>ls -lai</td>
<td>list all contents of directory with inode number and detail information</td>
</tr>
<tr>
<td>ls -d</td>
<td>list all directories under the current directory</td>
</tr>
<tr>
<td>tree</td>
<td>display file tree contents</td>
</tr>
<tr>
<td>lsod foo</td>
<td>list open status of file “foo”</td>
</tr>
<tr>
<td>lsod -p pid</td>
<td>list files opened by the process ID: “pid”</td>
</tr>
<tr>
<td>mkdir -p pid</td>
<td>make a new directory “foo” in the current directory</td>
</tr>
<tr>
<td>rmdir foo</td>
<td>remove a directory “foo” in the current directory</td>
</tr>
<tr>
<td>cd foo</td>
<td>change directory to the directory “foo” in the current directory or in the directory listed in the variable “$CDPATH”</td>
</tr>
<tr>
<td>cd /</td>
<td>change directory to the root directory</td>
</tr>
<tr>
<td>cd</td>
<td>change directory to the current user’s home directory</td>
</tr>
<tr>
<td>cd /foo</td>
<td>change directory to the absolute path directory “/foo”</td>
</tr>
<tr>
<td>cd ..</td>
<td>change directory to the parent directory</td>
</tr>
<tr>
<td>cd ~foo</td>
<td>change directory to the home directory of the user “foo”</td>
</tr>
<tr>
<td>cd -</td>
<td>change directory to the previous directory</td>
</tr>
<tr>
<td>&lt;/etc/motd pager</td>
<td>display contents of “/etc/motd” using the default pager</td>
</tr>
<tr>
<td>touch junkfile</td>
<td>create a new file “junkfile”</td>
</tr>
<tr>
<td>cp foo bar</td>
<td>copy a existing file “foo” to a new file “bar”</td>
</tr>
<tr>
<td>rm junkfile</td>
<td>remove a file “junkfile”</td>
</tr>
<tr>
<td>mv foo bar</td>
<td>rename an existing file “foo” to a new name “bar” (“bar” must not exist)</td>
</tr>
<tr>
<td>mv foo bar</td>
<td>move an existing file “foo” to a new location “bar/foo” (“bar” must exist)</td>
</tr>
<tr>
<td>mv foo bar/baz</td>
<td>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)</td>
</tr>
<tr>
<td>chmod 600 foo</td>
<td>make an existing file “foo” to be non-readable and non-writable by the other people (non-executable for all)</td>
</tr>
<tr>
<td>chmod 644 foo</td>
<td>make an existing file “foo” to be readable but non-writable by the other people (non-executable for all)</td>
</tr>
<tr>
<td>chmod 755 foo</td>
<td>make an existing file “foo” to be readable but non-writable by the other people (executable for all)</td>
</tr>
<tr>
<td>find . -name pattern</td>
<td>find matching filenames using shell “pattern” (slower)</td>
</tr>
<tr>
<td>locate -d . pattern</td>
<td>find matching filenames using shell “pattern” (quicker using regularly generated database)</td>
</tr>
<tr>
<td>grep -e “pattern” *.html</td>
<td>find a “pattern” in all files ending with “.html” in current directory and display them all</td>
</tr>
<tr>
<td>top</td>
<td>display process information using full screen, type “q” to quit</td>
</tr>
<tr>
<td>ps aux</td>
<td>pager</td>
</tr>
<tr>
<td>ps -ef</td>
<td>pager</td>
</tr>
<tr>
<td>ps aux</td>
<td>grep -e “[e]xim4”</td>
</tr>
<tr>
<td>ps aux</td>
<td>pager</td>
</tr>
<tr>
<td>kill 1234</td>
<td>kill a process identified by the process ID: “1234”</td>
</tr>
<tr>
<td>gzip foo</td>
<td>compress “foo” to create “foo.gz” using the Lempel-Ziv coding</td>
</tr>
<tr>
<td>lsof -p pid</td>
<td>list open status of process “pid”</td>
</tr>
<tr>
<td>lsof -p pid</td>
<td>list open status of process “pid”</td>
</tr>
<tr>
<td>cd</td>
<td>change directory to the directory “foo” in the current directory or in the directory listed in the variable “$CDPATH”</td>
</tr>
<tr>
<td>cd /</td>
<td>change directory to the root directory</td>
</tr>
<tr>
<td>cd</td>
<td>change directory to the current user’s home directory</td>
</tr>
<tr>
<td>cd /foo</td>
<td>change directory to the absolute path directory “/foo”</td>
</tr>
<tr>
<td>cd ..</td>
<td>change directory to the parent directory</td>
</tr>
<tr>
<td>cd ~foo</td>
<td>change directory to the home directory of the user “foo”</td>
</tr>
<tr>
<td>cd -</td>
<td>change directory to the previous directory</td>
</tr>
<tr>
<td>&lt;/etc/motd pager</td>
<td>display contents of “/etc/motd” using the default pager</td>
</tr>
<tr>
<td>touch junkfile</td>
<td>create a new file “junkfile”</td>
</tr>
<tr>
<td>cp foo bar</td>
<td>copy a existing file “foo” to a new file “bar”</td>
</tr>
<tr>
<td>rm junkfile</td>
<td>remove a file “junkfile”</td>
</tr>
<tr>
<td>mv foo bar</td>
<td>rename an existing file “foo” to a new name “bar” (“bar” must not exist)</td>
</tr>
<tr>
<td>mv foo bar</td>
<td>move an existing file “foo” to a new location “bar/foo” (“bar” must exist)</td>
</tr>
<tr>
<td>mv foo bar/baz</td>
<td>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)</td>
</tr>
<tr>
<td>chmod 600 foo</td>
<td>make an existing file “foo” to be non-readable and non-writable by the other people (non-executable for all)</td>
</tr>
<tr>
<td>chmod 644 foo</td>
<td>make an existing file “foo” to be readable but non-writable by the other people (non-executable for all)</td>
</tr>
<tr>
<td>chmod 755 foo</td>
<td>make an existing file “foo” to be readable but non-writable by the other people (executable for all)</td>
</tr>
<tr>
<td>find . -name pattern</td>
<td>find matching filenames using shell “pattern” (slower)</td>
</tr>
<tr>
<td>locate -d . pattern</td>
<td>find matching filenames using shell “pattern” (quicker using regularly generated database)</td>
</tr>
<tr>
<td>grep -e “pattern” *.html</td>
<td>find a “pattern” in all files ending with “.html” in current directory and display them all</td>
</tr>
<tr>
<td>top</td>
<td>display process information using full screen, type “q” to quit</td>
</tr>
<tr>
<td>ps aux</td>
<td>pager</td>
</tr>
<tr>
<td>ps -ef</td>
<td>pager</td>
</tr>
<tr>
<td>ps aux</td>
<td>grep -e “[e]xim4”</td>
</tr>
<tr>
<td>ps aux</td>
<td>pager</td>
</tr>
<tr>
<td>kill 1234</td>
<td>kill a process identified by the process ID: “1234”</td>
</tr>
<tr>
<td>gzip foo</td>
<td>compress “foo” to create “foo.gz” using the Lempel-Ziv coding</td>
</tr>
</tbody>
</table>
• The PAM system such as `pam_env` may set environment variables by `/etc/pam.conf`, `/etc/environment` and `/etc/default/locale`.

• The display manager such as `gdm3` may reset environment variables for GUI session by `~/.profile`.

• The user specific program initialization may reset environment variables by `~/.profile`, `~/.bash_profile` and `~/.bashrc`.

### 1.5.2 The "$LANG" variable

The default locale is defined in the "$LANG" environment variable and is configured as "LANG=xx_YY.UTF-8" by the installer or by the subsequent GUI configuration, e.g., "Settings" → "Region & Language" → "Language" / "Formats" for GNOME.

**Note**

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

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

<table>
<thead>
<tr>
<th>locale value</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>xx</td>
<td>ISO 639 language codes (lower case) such as &quot;en&quot;</td>
</tr>
<tr>
<td>YY</td>
<td>ISO 3166 country codes (upper case) such as &quot;US&quot;</td>
</tr>
<tr>
<td>ZZZZ</td>
<td>codeset, always set to &quot;UTF-8&quot;</td>
</tr>
</tbody>
</table>

Table 1.18: The 3 parts of locale value

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

Table 1.19: List of locale recommendations

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

```
$ echo $LANG
en_US.UTF-8
$ date -u
Wed 19 May 2021 03:18:43 PM UTC
$ LANG=fr_FR.UTF-8 date -u
mer. 19 mai 2021 15:19:02 UTC
```
Here, the program `date(1)` is executed with different locale values.

- 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".

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 -u
mer. 19 mai 2021 15:19:24 UTC
```

**Tip**

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

For fine details of the locale configuration, see Section 8.1.

### 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".

<table>
<thead>
<tr>
<th>value of &quot;$HOME&quot;</th>
<th>program execution situation</th>
</tr>
</thead>
<tbody>
<tr>
<td>/</td>
<td>program run by the init process (daemon)</td>
</tr>
<tr>
<td>/root</td>
<td>program run from the normal root shell</td>
</tr>
<tr>
<td>/home/normal_user</td>
<td>program run from the normal user shell</td>
</tr>
<tr>
<td>/home/normal_user</td>
<td>program run from the normal user GUI desktop menu</td>
</tr>
<tr>
<td>/home/normal_user</td>
<td>program run as root with &quot;sudo program&quot;</td>
</tr>
<tr>
<td>/root</td>
<td>program run as root with &quot;sudo -H program&quot;</td>
</tr>
</tbody>
</table>

Table 1.20: 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/".

See Section 12.1.5 if $HOME isn’t available for your program.
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
Thu 20 May 2021 01:08:08 AM JST
$ date -R
Thu, 20 May 2021 01:08:12 +0900
```

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.

<table>
<thead>
<tr>
<th>shell glob pattern</th>
<th>description of match rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>filename (segment) not started with &quot;.&quot;</td>
</tr>
<tr>
<td>.*</td>
<td>filename (segment) started with &quot;.&quot;</td>
</tr>
<tr>
<td>?</td>
<td>exactly one character</td>
</tr>
<tr>
<td>[…]</td>
<td>exactly one character with any character enclosed in brackets</td>
</tr>
<tr>
<td>[a-zA-Z]</td>
<td>exactly one character with any character between &quot;a&quot; and &quot;z&quot;</td>
</tr>
<tr>
<td>[^…]</td>
<td>exactly one character other than any character enclosed in brackets (excluding &quot;]&quot;)</td>
</tr>
</tbody>
</table>

Table 1.21: Shell glob patterns

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 *.h.[hc]
3.c 4.h
$ echo .*
...5.txt ..6.txt
$ echo .*[^.]*
.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.

<table>
<thead>
<tr>
<th>command exit status</th>
<th>numeric return value</th>
<th>logical return value</th>
</tr>
</thead>
<tbody>
<tr>
<td>success</td>
<td>zero, 0</td>
<td>TRUE</td>
</tr>
<tr>
<td>error</td>
<td>non-zero, -1</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

Table 1.22: Command exit codes

For example, try the following.

```bash
$ [ 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.

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.

<table>
<thead>
<tr>
<th>command idiom</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>command &amp;</td>
<td>background execution of command in the subshell</td>
</tr>
<tr>
<td>command1</td>
<td>command2</td>
</tr>
<tr>
<td>command1 2&gt;&amp;1</td>
<td>command2</td>
</tr>
<tr>
<td>command1 ; command2</td>
<td>execute command1 and command2 sequentially</td>
</tr>
<tr>
<td>command1 &amp;&amp; command2</td>
<td>execute command1; if successful, execute command2 sequentially (return success if both command1 and command2 are successful)</td>
</tr>
<tr>
<td>command1</td>
<td></td>
</tr>
<tr>
<td>command &gt; foo</td>
<td>redirect standard output of command to a file foo (overwrite)</td>
</tr>
<tr>
<td>command 2&gt; foo</td>
<td>redirect standard error of command to a file foo (overwrite)</td>
</tr>
<tr>
<td>command &gt;&gt; foo</td>
<td>redirect standard output of command to a file foo (append)</td>
</tr>
<tr>
<td>command 2&gt;&gt; foo</td>
<td>redirect standard error of command to a file foo (append)</td>
</tr>
<tr>
<td>command &gt; foo 2&gt;&amp;1</td>
<td>redirect both standard output and standard error of command to a file foo</td>
</tr>
<tr>
<td>command &lt; foo</td>
<td>redirect standard input of command to a file foo</td>
</tr>
<tr>
<td>command &lt;&lt; delimiter</td>
<td>redirect standard input of command to the following lines until &quot;delimiter&quot; is met (here document)</td>
</tr>
<tr>
<td>command &lt;&lt;- delimiter</td>
<td>redirect standard input of command to the following lines until &quot;delimiter&quot; is met (here document, the leading tab characters are stripped from input lines)</td>
</tr>
</tbody>
</table>

Table 1.23: Shell command idioms
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 "SIGNS", 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.

<table>
<thead>
<tr>
<th>device</th>
<th>description</th>
<th>file descriptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>stdin</td>
<td>standard input</td>
<td>0</td>
</tr>
<tr>
<td>stdout</td>
<td>standard output</td>
<td>1</td>
</tr>
<tr>
<td>stderr</td>
<td>standard error</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1.24: 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
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 9.3.6.

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

- **Extended regular expression (ERE) is used:**
  - `awk(1)` does simple text processing.
  - `egrep(1)` matches text with patterns.
  - `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 `perlr(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.html.

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 or C.UTF-8 locale instead of normal UTF-8 ones (see Section 8.1).

**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.

<table>
<thead>
<tr>
<th>BRE</th>
<th>ERE</th>
<th>description of the regular expression</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\ . [ ] ^ $ *</code></td>
<td><code>\ . [ ] ^ $ *</code></td>
<td>common metacharacters</td>
</tr>
<tr>
<td><code>\+ </code>? <code>\( </code>{ <code>\} </code>|`</td>
<td><code>+ </code>? <code>( ) </code>{ } `</td>
<td>BRE only &quot;&quot; escaped metacharacters</td>
</tr>
<tr>
<td><code>c</code></td>
<td><code>c</code></td>
<td>ERE only non-&quot;&quot; escaped metacharacters</td>
</tr>
<tr>
<td><code>\c</code></td>
<td><code>\c</code></td>
<td>match a literal character &quot;c&quot; even if &quot;c&quot; is metacharacter by itself</td>
</tr>
<tr>
<td>·</td>
<td>.</td>
<td>match any character including newline</td>
</tr>
<tr>
<td><code>^</code></td>
<td>^</td>
<td>position at the beginning of a string</td>
</tr>
<tr>
<td><code>$</code></td>
<td>$</td>
<td>position at the end of a string</td>
</tr>
<tr>
<td><code>\&lt;</code></td>
<td><code>&lt;</code></td>
<td>position at the beginning of a word</td>
</tr>
<tr>
<td><code>\&gt;</code></td>
<td><code>&gt;</code></td>
<td>position at the end of a word</td>
</tr>
<tr>
<td><code>[abc…]</code></td>
<td><code>[abc…]</code></td>
<td>match any characters in &quot;abc…&quot;</td>
</tr>
<tr>
<td><code>[^abc…]</code></td>
<td><code>[^abc…]</code></td>
<td>match any characters except in &quot;abc…&quot;</td>
</tr>
<tr>
<td><code>r*</code></td>
<td><code>r*</code></td>
<td>match zero or more regular expressions identified by &quot;r&quot;</td>
</tr>
<tr>
<td><code>r\+</code></td>
<td><code>r+</code></td>
<td>match one or more regular expressions identified by &quot;r&quot;</td>
</tr>
<tr>
<td><code>r\?</code></td>
<td><code>r?</code></td>
<td>match zero or one regular expressions identified by &quot;r&quot;</td>
</tr>
<tr>
<td><code>r1|r2</code></td>
<td>`r1</td>
<td>r2`</td>
</tr>
<tr>
<td><code>(r1|r2\)</code></td>
<td>`(r1</td>
<td>r2)`</td>
</tr>
</tbody>
</table>

Table 1.25: Metacharacters for BRE and ERE

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

```
$ 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 9.3.6.

## 1.6.3 Replacement expressions

For the replacement expression, some characters have special meanings.

<table>
<thead>
<tr>
<th>replacement expression</th>
<th>description of the text to replace the replacement expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;</td>
<td>what the regular expression matched (use &amp; in emacs)</td>
</tr>
<tr>
<td>\n</td>
<td>what the n-th bracketed regular expression matched (&quot;n&quot; being number)</td>
</tr>
</tbody>
</table>

Table 1.26: 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 command line 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' '+'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.

<table>
<thead>
<tr>
<th>Name</th>
<th>Leader</th>
<th>Month</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ian</td>
<td>Murdock</td>
<td>August</td>
<td>1993</td>
</tr>
<tr>
<td>Bruce</td>
<td>Perens</td>
<td>April</td>
<td>1996</td>
</tr>
<tr>
<td>Ian</td>
<td>Jackson</td>
<td>January</td>
<td>1998</td>
</tr>
<tr>
<td>Wichert</td>
<td>Akkerman</td>
<td>January</td>
<td>1999</td>
</tr>
<tr>
<td>Ben</td>
<td>Collins</td>
<td>April</td>
<td>2001</td>
</tr>
<tr>
<td>Bdale</td>
<td>Garbee</td>
<td>April</td>
<td>2002</td>
</tr>
<tr>
<td>Martin</td>
<td>Michlmayr</td>
<td>March</td>
<td>2003</td>
</tr>
</tbody>
</table>

---

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

For example, try the following

```
$ echo $IFS
```

```
IFS=':'
$ echo $IFS
```

```
IFS=,
$ echo $IFS
```

```
IFS=:
$ echo $IFS
```

Plain text:

```
1.6.6 Script snippets for piping commands

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

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 9.4.9.

When using the shell interactive mode becomes too complicated, please consider to write a shell script (see Section 12.1).
```
<table>
<thead>
<tr>
<th>script snippet (type in one line)</th>
<th>effect of command</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>find /usr -print</code></td>
<td>find all files under &quot;/usr&quot;</td>
</tr>
<tr>
<td><code>seq 1 100</code></td>
<td>print 1 to 100</td>
</tr>
<tr>
<td>`</td>
<td>xargs -n 1 command`</td>
</tr>
<tr>
<td>`</td>
<td>xargs -n 1 echo`</td>
</tr>
<tr>
<td>`</td>
<td>xargs echo`</td>
</tr>
<tr>
<td>`</td>
<td>grep -e regex_pattern`</td>
</tr>
<tr>
<td>`</td>
<td>grep -v -e regex_pattern`</td>
</tr>
<tr>
<td>`</td>
<td>cut -d: -f3 -`</td>
</tr>
<tr>
<td>`</td>
<td>awk '{ print $3 }'`</td>
</tr>
<tr>
<td>`</td>
<td>awk -F'\t' '{ print $3 }'`</td>
</tr>
<tr>
<td>`</td>
<td>col -bx`</td>
</tr>
<tr>
<td>`</td>
<td>expand -`</td>
</tr>
<tr>
<td>`</td>
<td>sort</td>
</tr>
<tr>
<td>`</td>
<td>tr 'A-Z' 'a-z'`</td>
</tr>
<tr>
<td>`</td>
<td>tr -d '\n'`</td>
</tr>
<tr>
<td>`</td>
<td>tr -d 'r'`</td>
</tr>
<tr>
<td>`</td>
<td>sed 's/^/# /'`</td>
</tr>
<tr>
<td>`</td>
<td>sed 's/.*.ext//g'`</td>
</tr>
<tr>
<td>`</td>
<td>sed -n -e 2p`</td>
</tr>
<tr>
<td>`</td>
<td>head -n 2 -`</td>
</tr>
<tr>
<td>`</td>
<td>tail -n 2 -`</td>
</tr>
</tbody>
</table>

Table 1.27: List of script snippets for piping commands
Chapter 2

Debian package management

Note
This chapter is written assuming the latest stable release is codename: bookworm.
The data source of the APT system is collectively referred as the source list in this document. This can be defined anywhere in the "/etc/apt/sources.list" file, "/etc/apt/sources.list.d/*.list" files, or "/etc/apt/sources.list.d/*.source" files.

2.1 Debian package management prerequisites

2.1.1 Debian package management system

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 current Debian package management system which can utilize all these resources is Advanced Packaging Tool (APT).

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 70981 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.
• 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.

2.1.2 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.
### Table 2.1: List of Debian package management tools

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dpkg</td>
<td>V:908,I:999</td>
<td>6447</td>
<td>low level package management system for Debian (file based)</td>
</tr>
<tr>
<td>apt</td>
<td>V:866,I:999</td>
<td>4318</td>
<td>APT front-end to manage packages with CLI: apt/apt-get/apt-cache</td>
</tr>
<tr>
<td>aptitude</td>
<td>V:51,I:268</td>
<td>4268</td>
<td>APT front-end to interactively manage packages with full screen console: aptitude(8)</td>
</tr>
<tr>
<td>tasksel</td>
<td>V:35,I:980</td>
<td>347</td>
<td>APT front-end to install selected tasks: tasksel(8)</td>
</tr>
<tr>
<td>unattended-upgrades</td>
<td>V:209,I:314</td>
<td>301</td>
<td>enhancement package for APT to enable automatic installation of security upgrades</td>
</tr>
<tr>
<td>gnome-software</td>
<td>V:150,I:262</td>
<td>3032</td>
<td>Software Center for GNOME (GUI APT front-end)</td>
</tr>
<tr>
<td>synaptic</td>
<td>V:46,I:375</td>
<td>7627</td>
<td>graphical package manager (GTK APT front-end)</td>
</tr>
<tr>
<td>apt-utils</td>
<td>V:383,I:998</td>
<td>1065</td>
<td>APT utility programs: apt-extracttemplates(1), apt-ftparchive(1), apt-sortpkgs(1)</td>
</tr>
<tr>
<td>apt-listchanges</td>
<td>V:355,I:869</td>
<td>398</td>
<td>package change history notification tool</td>
</tr>
<tr>
<td>apt-listbugs</td>
<td>V:6,I:9</td>
<td>477</td>
<td>lists critical bugs before each APT installation</td>
</tr>
<tr>
<td>apt-file</td>
<td>V:16,I:69</td>
<td>89</td>
<td>APT package searching utility — command-line interface</td>
</tr>
<tr>
<td>apt-rdepends</td>
<td>V:0,I:5</td>
<td>39</td>
<td>recursively lists package dependencies</td>
</tr>
</tbody>
</table>

- 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.
- 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.

#### 2.1.3 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. Until you understand the Debian system very well, you should follow the following precautions.

- Do not include **testing** or **unstable** in the **source list**.
- Do not mix standard Debian with other non-Debian archives such as Ubuntu in the **source 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 violating above precautions 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.

2.1.4 Life with eternal upgrades

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.

Despite my warnings above, I know many readers of this document may wish to run the newer testing or unstable suites. Enlightenment with the following saves a person from the eternal karmic struggle of upgrade hell and let him reach Debian nirvana.

This list is targeted for the self-administered Desktop environment.

• Use the testing suite since it is practically the rolling release automatically managed by the Debian archive QA infrastructure such as the Debian continuous integration, the source only upload practices, and the library transition tracking. The packages in the testing suite are updated frequently enough to offer all the latest features.

• Set the codename corresponding to the testing suite ("trixie" during the bookworm-as-stable release cycle) in the source list.

• Manually update this codename in the source list to the new one only after assessing situation by yourself for about a month after the major suite release. The Debian user and developer mailing list are good sources of information for this, too.

The use of the unstable suite isn’t recommended. The unstable suite is good for debugging packages as a developer but tends to expose you to unnecessary risks for the normal Desktop usage. Even though the unstable suite of the Debian system looks very stable for most of the times, there have been some package problems and a few of them were not so trivial to resolve.

Here are some basic precautionary measure 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

Caution
If you can not do any one of these precautionary actions, you are probably not ready for the testing and unstable suites.
2.1.5 Debian archive basics

**Tip**

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

For a system user, the Debian archive is accessed using the APT system.

The APT system specifies its data source as the source list and it is described in `sources.list(5)`.

For the `bookworm` system with the typical HTTP access, the source list in one-line-style as the following:

```
deb http://deb.debian.org/debian/ bookworm main non-free-firmware contrib non-free
deb-src http://deb.debian.org/debian/ bookworm main non-free-firmware contrib non-free
```

Alternatively, the equivalent source list in deb822-style as the following.

```
Types: deb deb-src
URLs: http://deb.debian.org/debian/
Suites: bookworm
Components: main non-free-firmware contrib non-free

Types: deb deb-src
URLs: http://security.debian.org/debian-security/
Suites: bookworm-security
Components: main non-free-firmware contrib non-free
```

Key points of the source list are followings.

- **One-line-style format**
  - It’s definition files are in the `/etc/apt/sources.list` file and `/etc/apt/sources.list.d/*.list` files.
  - Each line defines the data source for the APT system.
  - 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 using either the suite name or the codename.
  - The 3rd and following arguments are the list of valid archive area names of the Debian archive.

- **Deb822-style format**
  - It’s definition files are in `/etc/apt/sources.list.d/*.source` files.
  - Each block of lines separated by a blank line defines the data source for the APT system.
  - The "Types:" stanza defines the list of types such as "deb" and "deb-src".
  - The "URIs:" stanza defines the list of root URIs of the Debian archive.
  - The "Suites:" stanza defines the list of distribution names using either the suite name or the codename.
  - The "Components:" stanza defines the list of valid archive area names of the Debian archive.
The definition for "deb-src" can safely be omitted 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://", ...

Lines starting with "#" are comments and ignored.

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

**Tip**

If "sid" is used in the above example instead of "bookworm", the "deb: http://security.debian.org/ ... " line or its deb822 equivalent content for security updates in the source 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 after the bookworm release.

<table>
<thead>
<tr>
<th>archive URL</th>
<th>suite name (codename)</th>
<th>purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>stable (bookworm)</td>
<td>stable (bookworm) release</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>testing (trixie)</td>
<td>testing (trixie) release</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>unstable (sid)</td>
<td>unstable (sid) release</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>experimental</td>
<td>experimental pre-release (optional, only for developer)</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>stable-proposed-updates (bookworm-proposed-updates)</td>
<td>Updates for the next stable (bookworm) point release (optional)</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>stable-updates(bookworm-updates)</td>
<td>compatible updates for spam filter, IM clients, etc. for stable (bookworm)</td>
</tr>
<tr>
<td><a href="http://deb.debian.org/debian/">http://deb.debian.org/debian/</a></td>
<td>stable-backports (bookworm-backports)</td>
<td>newer backported packages for stable (bookworm) (optional)</td>
</tr>
<tr>
<td><a href="http://security.debian.org/debian-security/">http://security.debian.org/debian-security/</a></td>
<td>stable-security (bookworm-security)</td>
<td>security updates for stable release (bookworm) (important)</td>
</tr>
<tr>
<td><a href="http://security.debian.org/debian-security/">http://security.debian.org/debian-security/</a></td>
<td>testing-security (trixie-security)</td>
<td>security updates for testing release (This isn’t actively supported by the security team)</td>
</tr>
</tbody>
</table>

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 stable-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 2.7.7).

**Tip**

For the Debian system with the stable suite, it is a good idea to include the content with "http://security.debian.org/" in the source 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.

<table>
<thead>
<tr>
<th>area</th>
<th>number of packages</th>
<th>criteria of package component</th>
</tr>
</thead>
<tbody>
<tr>
<td>main</td>
<td>69633</td>
<td>DFSG compliant and no dependency to non-free</td>
</tr>
<tr>
<td>non-free-firmware</td>
<td>39</td>
<td>not DFSG compliant, firmware required for reasonable system installation experience</td>
</tr>
<tr>
<td>contrib</td>
<td>357</td>
<td>DFSG compliant but having dependency to non-free</td>
</tr>
<tr>
<td>non-free</td>
<td>952</td>
<td>not DFSG compliant and not in non-free-firmware</td>
</tr>
</tbody>
</table>

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 2.1.6).

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.

<table>
<thead>
<tr>
<th>Timing</th>
<th>suite = stable</th>
<th>suite = testing</th>
<th>suite = unstable</th>
</tr>
</thead>
<tbody>
<tr>
<td>after the bookworm release</td>
<td>codename = bookworm</td>
<td>codename = trixie</td>
<td>codename = sid</td>
</tr>
<tr>
<td>after the trixie release</td>
<td>codename = trixie</td>
<td>codename = forky</td>
<td>codename = sid</td>
</tr>
</tbody>
</table>

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](https://lists.debian.org/debian-security/2008/msg00003.html)). 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.

---


• “Sections”
• “Priorities”
• “Base system”
• “Essential packages”

### 2.1.6 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.
• No packages in main depend nor recommend packages in non-free nor non-free-firmware nor contrib.

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

• "Debian will remain 100% free". (First term of [Debian Social Contract](https://www.debian.org/socialcontract))
• Debian servers host some non-free-firmware, non-free and contrib packages.

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.
• Packages outside of the Debian system are hosted by Debian servers in the non-free, non-free-firmware and contrib areas.
These are precisely explained in the 4th and 5th terms of Debian Social Contract:

• Our priorities are our users and free software
  – We will be guided by the needs of our users and the free software community. We will place their interests first in our priorities. We will support the needs of our users for operation in many different kinds of computing environments. We will not object to non-free works that are intended to be used on Debian systems, or attempt to charge a fee to people who create or use such works. We will allow others to create distributions containing both the Debian system and other works, without any fee from us. In furtherance of these goals, we will provide an integrated system of high-quality materials with no legal restrictions that would prevent such uses of the system.

• Works that do not meet our free software standards
  – We acknowledge that some of our users require the use of works that do not conform to the Debian Free Software Guidelines. We have created "non-free", "non-free-firmware" and "contrib" areas in our archive for these works. The packages in these areas are not part of the Debian system, although they have been configured for use with Debian. We encourage CD manufacturers to read the licenses of the packages in these areas and determine if they can distribute the packages on their CDs. Thus, although non-free works are not a part of Debian, we support their use and provide infrastructure for non-free packages (such as our bug tracking system and mailing lists). The Debian official media may include firmware that is otherwise not part of the Debian system to enable use of Debian with hardware that requires such firmware.

Note
The actual text of the 5th term in the current Debian Social Contract 1.2 is slightly different from the above text. This editorial deviation is intensional one to make this user document consistent without changing the real content of the Social Contract.

Users should be aware of the risks of using packages in the non-free, non-free-firmware and contrib areas:

• 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.

Typical non-free, non-free-firmware and contrib packages include freely distributable packages of following types:

• Document packages under GNU Free Documentation License with invariant sections such as ones for GCC and Make. (mostly found in the non-free/doc section.)
• Firmware packages containing sourceless binary data such as ones listed in Section 9.10.5 as non-free-firmware. (mostly found in the non-free-firmware/kernel section.)
• Game and font packages with restriction on commercial use and/or content modification.

Please note that the number of non-free, non-free-firmware and contrib packages is less than 2% of that of main packages. Enabling access to the non-free, non-free-firmware and contrib areas does not obscure the source of packages. Interactive full screen use of aptitude(8) provides you with full visibility and control over what packages are installed from which area to keep your system as free as you wish.
2.1.7 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 weak 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.
2.1.8 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. Choose candidate version which is usually the latest available version for all installed packages (see Section 2.7.7 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. Choose 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. Choose 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. Choose 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.
2.1.9 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_name/README.Debian". 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.

<table>
<thead>
<tr>
<th>web site</th>
<th>command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Home page of the Debian bug tracking system (BTS)</td>
<td>sensible-browser &quot;<a href="http://bugs.debian.org/">http://bugs.debian.org/</a>&quot;</td>
</tr>
<tr>
<td>The bug report of a known package name</td>
<td>sensible-browser &quot;<a href="http://bugs.debian.org/package_name">http://bugs.debian.org/package_name</a>&quot;</td>
</tr>
<tr>
<td>The bug report of known bug number</td>
<td>sensible-browser &quot;<a href="http://bugs.debian.org/bug_number">http://bugs.debian.org/bug_number</a>&quot;</td>
</tr>
</tbody>
</table>

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.

2.1.10 How to pick Debian packages

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
- Area: main > contrib > non-free
- Priority: required > important > standard > optional > extra
- Tasks: packages listed in tasks such as "Desktop environment"
- Packages selected by the dependency package (e.g., gcc-10 by gcc)
- 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.
2.1.11 How to cope with conflicting requirements

Whatever suite of Debian system you may decide to use, you may still wish to run versions of programs which aren’t available in that suite. Even if you find binary packages of such programs in other Debian suites or in other non-Debian resources, their requirements may conflict with your current Debian system.

Although you can tweak package management system with apt-pinning technique etc. as described in Section 2.7.7 to install such out-of-sync binary packages, such tweaking approaches have only limited use cases since they may break those programs and your system.

Before brutally installing such out-of-sync packages, you should seek all available alternative safer technical solutions which are compatible with your current Debian system.

• Install such programs using corresponding sandboxed upstream binary packages (see Section 7.6).
  – Many mostly GUI programs such as LibreOffice and GNOME applications are available as Flatpak, Snap, or AppImage packages.
• Create a chroot or similar environment and run such programs in it (see Section 9.11).
  – CLI commands can be executed easily under its compatible chroot (see Section 9.11.4).
  – Multiple full desktop environments can be tried easily without reboot (see Section 9.11.5).
• Build desired versions of binary packages which are compatible with your current Debian system by yourself.
  – This is a non-trivial task (see Section 2.7.13).

2.2 Basic package management operations

Repository based package management operations on the Debian system can be performed by many APT-based package management 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 successful 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 full screen interactive text user interface.

- **aptitude** offers the command line 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.

### 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).

- **apt / apt-get** and **aptitude** can be mixed without major troubles.

  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.

---

**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.

You may provide command options right after "aptitude".

See **aptitude**(8) and "aptitude user’s manual" at "/usr/share/doc/aptitude/README" for more.
<table>
<thead>
<tr>
<th>apt syntax</th>
<th>aptitude syntax</th>
<th>apt-get/apt-cache syntax</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>apt update</td>
<td>aptitude update</td>
<td>apt-get update</td>
<td>update package archive metadata</td>
</tr>
<tr>
<td>apt install foo</td>
<td>aptitude install foo</td>
<td>apt-get install foo</td>
<td>install candidate version of &quot;foo&quot; package with its dependencies</td>
</tr>
<tr>
<td>apt upgrade</td>
<td>aptitude safe-upgrade</td>
<td>apt-get upgrade</td>
<td>install candidate version of installed packages without removing any other packages</td>
</tr>
<tr>
<td>apt full-upgrade</td>
<td>aptitude full-upgrade</td>
<td>apt-get full-upgrade</td>
<td>install candidate version of installed packages while removing other packages if needed</td>
</tr>
<tr>
<td>apt remove foo</td>
<td>aptitude remove foo</td>
<td>apt-get remove foo</td>
<td>remove &quot;foo&quot; package while leaving its configuration files</td>
</tr>
<tr>
<td>apt autoremove</td>
<td>N/A</td>
<td>apt-get autoremove</td>
<td>remove auto-installed packages which are no longer required</td>
</tr>
<tr>
<td>apt purge foo</td>
<td>aptitude purge foo</td>
<td>apt-get purge foo</td>
<td>purge &quot;foo&quot; package with its configuration files</td>
</tr>
<tr>
<td>apt clean</td>
<td>aptitude clean</td>
<td>apt-get clean</td>
<td>clear out the local repository of retrieved package files completely</td>
</tr>
<tr>
<td>apt autoclean</td>
<td>aptitude autoclean</td>
<td>apt-get autoclean</td>
<td>clear out the local repository of retrieved package files for outdated packages</td>
</tr>
<tr>
<td>apt show foo</td>
<td>aptitude show foo</td>
<td>apt-cache show foo</td>
<td>display detailed information about &quot;foo&quot; package</td>
</tr>
<tr>
<td>apt search regex</td>
<td>aptitude search regex</td>
<td>apt-cache search regex</td>
<td>search packages which match regex</td>
</tr>
<tr>
<td>N/A</td>
<td>aptitude why regex</td>
<td>N/A</td>
<td>explain the reason why regex matching packages should be installed</td>
</tr>
<tr>
<td>N/A</td>
<td>aptitude why-not regex</td>
<td>N/A</td>
<td>explain the reason why regex matching packages cannot be installed</td>
</tr>
<tr>
<td>apt list</td>
<td>N/A</td>
<td>apt-mark showmanual</td>
<td>list manually installed packages</td>
</tr>
<tr>
<td>--manual-installed</td>
<td>N/A</td>
<td>N/A</td>
<td>list manually installed packages</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>command option</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-s</td>
<td>simulate the result of the command</td>
</tr>
<tr>
<td>-d</td>
<td>download only but no install/upgrade</td>
</tr>
<tr>
<td>-b</td>
<td>show brief explanations before the automatic installations and removals</td>
</tr>
</tbody>
</table>

Table 2.7: Notable command options for aptitude(8)
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: [enter 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 keystrokes to browse status of packages and to set "planned action" on them in this full screen mode are the following.

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

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

<table>
<thead>
<tr>
<th>view</th>
<th>description of view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package View</td>
<td>see Table 2.10 (default)</td>
</tr>
<tr>
<td>Audit Recommendations</td>
<td>list packages which are recommended by some installed packages but not yet installed</td>
</tr>
<tr>
<td>Flat Package List</td>
<td>list packages without categorization (for use with regex)</td>
</tr>
<tr>
<td>Debtags Browser</td>
<td>list packages categorized according to their debtags entries</td>
</tr>
<tr>
<td>Source Package View</td>
<td>list packages grouped by source packages</td>
</tr>
</tbody>
</table>

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.

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

<table>
<thead>
<tr>
<th>category</th>
<th>description of view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upgradable Packages</td>
<td>list packages organized as section → area → package</td>
</tr>
<tr>
<td>New Packages</td>
<td></td>
</tr>
<tr>
<td>Installed Packages</td>
<td></td>
</tr>
<tr>
<td>Not Installed Packages</td>
<td></td>
</tr>
<tr>
<td>Obsolete and Locally Created Packages</td>
<td></td>
</tr>
<tr>
<td>Virtual Packages</td>
<td>list packages with the same function</td>
</tr>
<tr>
<td>Tasks</td>
<td>list packages with different functions generally needed for a task</td>
</tr>
</tbody>
</table>

Table 2.10: The categorization of standard package views

### 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 "^", ".*", "$" etc. as in `egrep(1)`, `awk(1)` and `perl(1)`.
- The dependency **type** is one of (depends, predeeps, 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.
<table>
<thead>
<tr>
<th>Description of the Extended Match Rule</th>
<th>Regex Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>match on package name</td>
<td>~nregex_name</td>
</tr>
<tr>
<td>match on description</td>
<td>~dregex_description</td>
</tr>
<tr>
<td>match on task name</td>
<td>~tregex_task</td>
</tr>
<tr>
<td>match on debtag</td>
<td>~Gregex_debtag</td>
</tr>
<tr>
<td>match on maintainer</td>
<td>~mregex_maintainer</td>
</tr>
<tr>
<td>match on package section</td>
<td>~sregex_section</td>
</tr>
<tr>
<td>match on package version</td>
<td>~Vregex_version</td>
</tr>
<tr>
<td>match archive</td>
<td>~A{bookworm, trixie, sid}</td>
</tr>
<tr>
<td>match origin</td>
<td>~O{debian, ...}</td>
</tr>
<tr>
<td>match priority</td>
<td>~p{extra, important, optional, required, standard}</td>
</tr>
<tr>
<td>match essential packages</td>
<td>~E</td>
</tr>
<tr>
<td>match virtual packages</td>
<td>~V</td>
</tr>
<tr>
<td>match new packages</td>
<td>~N</td>
</tr>
<tr>
<td>match with pending action</td>
<td>~a{install, upgrade, downgrade, remove, purge, hold, keep}</td>
</tr>
<tr>
<td>match installed packages</td>
<td>~I</td>
</tr>
<tr>
<td>match installed packages with A-mark (auto installed packages)</td>
<td>~M</td>
</tr>
<tr>
<td>match installed packages without A-mark (administrator selected packages)</td>
<td>~I!~M</td>
</tr>
<tr>
<td>match new packages</td>
<td>~N</td>
</tr>
<tr>
<td>match removed but not purged packages</td>
<td>~C</td>
</tr>
<tr>
<td>match removed, purged or can-be-removed packages</td>
<td>~G</td>
</tr>
<tr>
<td>match packages declaring a broken dependency</td>
<td>~Btype</td>
</tr>
<tr>
<td>match packages declaring broken dependency of type</td>
<td>~D[type:]pattern</td>
</tr>
<tr>
<td>match packages declaring dependency of type</td>
<td>~DB[type:]pattern</td>
</tr>
<tr>
<td>match packages to which the pattern matching package declares dependency type</td>
<td>~R[type:]pattern</td>
</tr>
<tr>
<td>match packages to which the pattern matching package declares broken dependency type</td>
<td>~RB[type:]pattern</td>
</tr>
<tr>
<td>match packages to which some other installed packages depend on</td>
<td><del>R</del>i</td>
</tr>
<tr>
<td>match packages to which no other installed packages depend on</td>
<td>!<del>R</del>i</td>
</tr>
<tr>
<td>match packages to which some other installed packages depend or recommend on</td>
<td><del>R</del>i</td>
</tr>
<tr>
<td>match pattern package with filtered version</td>
<td>~S filter pattern</td>
</tr>
<tr>
<td>match all packages (true)</td>
<td>~T</td>
</tr>
<tr>
<td>match no packages (false)</td>
<td>~F</td>
</tr>
</tbody>
</table>

Table 2.11: List of the aptitude regex formula
• "~Pterm" == "~Dprovides:term"
• "~Cterm" == "~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.

### 2.2.9 Package activity logs

You can check package activity history in the log files.

<table>
<thead>
<tr>
<th>file</th>
<th>content</th>
</tr>
</thead>
<tbody>
<tr>
<td>/var/log/dpkg.log</td>
<td>Log of dpkg level activity for all package activities</td>
</tr>
<tr>
<td>/var/log/apt/term.log</td>
<td>Log of generic APT activity</td>
</tr>
<tr>
<td>/var/log/aptitude.log</td>
<td>Log of aptitude command activity</td>
</tr>
</tbody>
</table>

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 9.3.9 for easier way.

### 2.3 Examples of aptitude operations

Here are few examples of aptitude(8) operations.

#### 2.3.1 Seeking interesting packages

You can seek packages which satisfy your needs with aptitude from the package description or from the list under "Tasks".

#### 2.3.2 Listing packages with regex matching on package names

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

```bash
$ 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.
2.3.3 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.

2.3.4 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.

2.3.5 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-1~Rrecommends:$1)" and type "M" over "Installed Packages" as auto installed.
4. Type "l" to enter the package display limit as "~prequired|~important|~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.
2.3.6 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 source list pointing to a new release and running the "apt update; apt dist-upgrade" command.

To upgrade from stable to testing or unstable during the bookworm-as-stable release cycle, you replace "bookworm" in the source list example of Section 2.1.5 with "trixie" 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).
10. Update the source list 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.

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 the source 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 9.10).
- See Section 12.9 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.
<table>
<thead>
<tr>
<th>command</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>COLUMNS=120 dpkg -l package_name_pattern</code></td>
<td>list status of an installed package for the bug report</td>
</tr>
<tr>
<td><code>dpkg -L package_name</code></td>
<td>list contents of an installed package</td>
</tr>
<tr>
<td>`dpkg -L package_name</td>
<td>egrep '/usr/share/man/man..*/.+.'`</td>
</tr>
<tr>
<td><code>dpkg -S file_name_pattern</code></td>
<td>list installed packages which have matching file name</td>
</tr>
<tr>
<td><code>apt-file search file_name_pattern</code></td>
<td>list packages in archive which have matching file name</td>
</tr>
<tr>
<td><code>apt-file list package_name_pattern</code></td>
<td>list contents of matching packages in archive</td>
</tr>
<tr>
<td><code>dpkg-reconfigure package_name</code></td>
<td>reconfigure the exact package</td>
</tr>
<tr>
<td><code>dpkg-reconfigure -plow package_name</code></td>
<td>reconfigure the exact package with the most detailed question</td>
</tr>
<tr>
<td><code>configure-debian</code></td>
<td>reconfigure packages from the full screen menu</td>
</tr>
<tr>
<td><code>dpkg --audit</code></td>
<td>audit system for partially installed packages</td>
</tr>
<tr>
<td><code>dpkg --configure -a</code></td>
<td>configure all partially installed packages</td>
</tr>
<tr>
<td><code>apt-cache policy binary_package_name</code></td>
<td>show available version, priority, and archive information of a binary package</td>
</tr>
<tr>
<td><code>apt-cache madison package_name</code></td>
<td>show available version, archive information of a package</td>
</tr>
<tr>
<td><code>apt-get install /path/to/package_filename.deb</code></td>
<td>install a local package to the system, meanwhile try to resolve dependency automatically</td>
</tr>
<tr>
<td><code>apt-get build-dep package_name</code></td>
<td>install required packages to build package</td>
</tr>
<tr>
<td><code>aptitude build-dep package_name</code></td>
<td>install required packages to build package</td>
</tr>
<tr>
<td><code>apt-get source package_name</code></td>
<td>download a source from a set of source packages (&quot;<em>.orig.tar.gz&quot; &quot;</em>.debian.tar.gz&quot; &quot;*.diff.gz&quot;)</td>
</tr>
<tr>
<td><code>dget URL for dsc file</code></td>
<td>download a source packages (from other archive)</td>
</tr>
<tr>
<td><code>dpkg-source -x package_name_version-debian.revision-debian.revision debian.revision.dsc</code></td>
<td>build a source tree from a set of source packages (&quot;<em>.orig.tar.gz&quot; &quot;</em>.debian.tar.gz&quot; &quot;*.diff.gz&quot;)</td>
</tr>
<tr>
<td><code>deb</code></td>
<td>build package(s) from a local source tree</td>
</tr>
<tr>
<td><code>make-kpkg kernel_image</code></td>
<td>build a kernel package from a kernel source tree</td>
</tr>
<tr>
<td><code>make-kpkg --initrd kernel_image</code></td>
<td>build a kernel package from a kernel source tree with initramfs enabled</td>
</tr>
<tr>
<td><code>dpkg -i package_name_version-debian.revision_arch.deb</code></td>
<td>install a local package to the system</td>
</tr>
<tr>
<td><code>apt install /path/to/package_filename.deb</code></td>
<td>install a local package to the system, meanwhile try to resolve dependency automatically</td>
</tr>
<tr>
<td><code>deb</code></td>
<td>install local package(s) to the system</td>
</tr>
<tr>
<td><code>dpkg --get-selections '*' &gt;selection.txt</code></td>
<td>save dpkg level package selection state information</td>
</tr>
<tr>
<td><code>dpkg --set-selections &lt;selection.txt</code></td>
<td>set dpkg level package selection state information</td>
</tr>
<tr>
<td>`echo package_name hold</td>
<td>dpkg --set-selections`</td>
</tr>
</tbody>
</table>

Table 2.13: List of advanced package management operations
2.4.3 Safeguarding for package problems

Many users prefer to follow the testing (or unstable) releases 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/](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 to 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.org](http://deb.debian.org). Its archive structure can be browsed by the web browser. There are 6 types of key meta data.

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

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/Release", as follows.

<table>
<thead>
<tr>
<th>Origin: Debian</th>
<th>Label: Debian</th>
<th>Suite: unstable</th>
<th>Codename: sid</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date: Sat, 14 May 2011 08:20:50 UTC</td>
<td>Valid-Until: Sat, 21 May 2011 08:20:50 UTC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Architectures:</td>
<td></td>
<td>alpha amd64 armel hppa hurd-i386 i386 ia64 kfreebsd-amd64 kfreebsd-i386 mips ← mipsel powerpc s390 sparc</td>
<td></td>
</tr>
<tr>
<td>Components: main contrib non-free</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td>Debian</td>
<td>x.y Unstable - Not Released</td>
<td></td>
</tr>
<tr>
<td>MD5Sum:</td>
<td></td>
<td>bdc8fa4b3f5e4a715dd0d56d176fc789 18876880 Contents-alpha.gz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>9469a03c94b85e010d116ae8eb9614c0 19441880 Contents-amd64.gz</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>3d68e206d7faa3aded660dc099f6054fe 1920165 Contents-armel.gz</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note**
Here, you can find my rationale to use the "suite", and "codename" in Section 2.1.5. 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** as described in `apt-secure(8)`.

- 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 keys are locally installed by the latest `debian-archive-keyring` package.
- The **secure APT** system automatically verifies the integrity of the downloaded top level "Release" file cryptographically by this "Release.gpg" file and the locally installed public Debian archive keys.
- 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).

If the source list entry specifies the "signed-by" option, the integrity of its downloaded top level "Release" file is verified using specified public key. This is useful when the source list contains non-Debian archives.

**Tip**
The use of `apt-key(8)` command for APT key management is deprecated.

Also, you can manually verify the integrity of the "Release" file with the "Release.gpg" file and the public Debian archive key posted on ftp-master.debian.org using gpg.
### 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 the source list, such as "http://deb.debian.org/debian/dists/unstable/main/binary-amd64/Release" or "http://deb.debian.org/debian/dists/sid/main/binary-amd64/Release" 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 bookworm-backports, which contain packages which should not be installed automatically, there is an extra line, e.g., "http://deb.debian.org/debian/dists/experimental/main/binary-amd64/Release" 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 2.7.7).

### 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 source list (see Section 2.1.5).

- "/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-architecture_Packages"
- "/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-architecture.gz" (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 the "deb" is specified in the source list. The "Sources" meta data are updated if the "deb-src" is specified in the source 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.

<table>
<thead>
<tr>
<th>package type</th>
<th>name structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>The binary package (a.k.a deb)</td>
<td>package-name_upstream-version-debian.revision_architecture.deb</td>
</tr>
<tr>
<td>The binary package for debian-installer (a.k.a udeb)</td>
<td>package-name_upstream-version-debian.revision_architecture.udeb</td>
</tr>
<tr>
<td>The source package (upstream source)</td>
<td>package-name_upstream-version-debian.revision.orig.tar.gz</td>
</tr>
<tr>
<td>The 1.0 source package (Debian changes)</td>
<td>package-name_upstream-version-debian.revision.diff.gz</td>
</tr>
<tr>
<td>The 3.0 (quilt) source package (Debian changes)</td>
<td>package-name_upstream-version-debian.revision.debian.tar.gz</td>
</tr>
<tr>
<td>The source package (description)</td>
<td>package-name_upstream-version-debian.revision.dsc</td>
</tr>
</tbody>
</table>

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)`.

---

**Note**
You can check package version order by `dpkg(1)`, e.g., "dpkg --compare-versions 7.0 gt 7.~pre1 ; echo $?".
<table>
<thead>
<tr>
<th>name component</th>
<th>usable characters (ERE regex)</th>
<th>existence</th>
</tr>
</thead>
<tbody>
<tr>
<td>package-name</td>
<td>[a-z0-9][-a-z0-9.+]</td>
<td>required</td>
</tr>
<tr>
<td>epoch</td>
<td>[0-9]+:</td>
<td>optional</td>
</tr>
<tr>
<td>upstream-version</td>
<td>[-a-zA-Z0-9.:+]+</td>
<td>required</td>
</tr>
<tr>
<td>debian.revision</td>
<td>[a-zA-Z0-9.~+]</td>
<td>optional</td>
</tr>
</tbody>
</table>

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

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)

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

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

Table 2.17: The notable files created by dpkg

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.
2.5.10 The update-alternatives command

The Debian system has a 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.

```bash
$ 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 testing or unstable system, the administrator is expected to recover from broken package management situation.
2.6.1 Failed installation due to missing dependencies

If you force to install a package by "sudo dpkg -i ..." to a system without all dependency packages installed, the package installation will fail as partially installed. You should install all dependency packages using APT-system or "sudo dpkg -i ...".

Then, configure all partially installed packages with the following command.

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

2.6.2 Caching errors of the package data

Caching errors of the package data cause intriguing errors, such as "GPG error: ... invalid: BADSIG ..." with APT.

You should remove all cached data by "sudo rm -rf /var/lib/apt/* " and try again. (If apt-cacher-ng is used, you should also run "sudo rm -rf /var/cache/apt-cacher-ng/* ".)

2.6.3 Incompatibility with old user configuration

If a desktop GUI program experienced instability after significant upstream version upgrade, you should suspect interference 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.)

2.6.4 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 2.1.7).

Errors by the package maintainer or deployment of inconsistently mixed source of archives (see Section 2.7.6) 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.

```bash
$ sudo dpkg -P old-package
```
2.6.5 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"

Then, follow Section 2.6.

2.6.6 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/archives/".

(If not, you can download it from archive of [https://snapshot.debian.org/](https://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 2.7.11 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.

---

2.6.7 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.status.*". 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 retrieve 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

For simplicity, the source list examples in this section are presented as "/etc/apt/sources.list" in one-line-style after the bookworm release.

2.7.1 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.

2.7.2 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";
```
2.7.3 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-upgrades"` 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 a testing or unstable system, you do not want to use the automatic upgrade since it certainly breaks system some day. Even for such testing or 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";
```

2.7.4 Updates and Backports

There are stable-updates ("bookworm-updates" during the bookworm-as-stable release cycle) and backports.debian.org archives which provide upgrade packages for stable.

In order to use these archives, you list all required archives in the `/etc/apt/sources.list` file as the following.

```
debug http://deb.debian.org/debian/ bookworm main non-free-firmware contrib non-free
deb http://security.debian.org/debian-security bookworm-security main non-free-firmware contrib non-free
deb http://deb.debian.org/debian/ bookworm-updates main non-free-firmware contrib non-free
deb http://deb.debian.org/debian/ bookworm-backports main non-free-firmware contrib non-free
```

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 bookworm-updates.
- Only manually installed older packages from bookworm-backports are upgraded to newer ones from bookworm-backports.

Whenever you wish to install a package named "package-name" with its dependency from bookworm-backports archive manually, you use following command while switching target release with "-t" option.

```
$ sudo apt-get install -t bookworm-backports package-name
```

---

**Warning**

Do not install too many packages from backports.debian.org archives. It may cause package dependency complications. See Section 2.1.11 for alternative solutions.
2.7.5 External package archives

⚠️ **Warning**
You should be aware that the external package gains the root privilege to your system. You should only use the trusted external package archive. See Section 2.1.11 for alternative solutions.

You can use secure APT with Debian-compatible external package archive by adding it to the `source list` and its archive key file into the `/etc/apt/trusted.gpg.d/` directory. See `sources.list(5), apt-secure(8) and apt-key(8).

2.7.6 Packages from mixed source of archives without apt-pinning

⚠️ **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 `stable-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**
In order to make a package to be safer to install, some commercial non-free binary program packages may be provided with completely statically linked libraries. You should still check ABI compatibility issues etc. for them.

**Note**
Except to avoid broken package for a short term, installing binary packages from non-Debian archives is generally bad idea. You should seek all available alternative safer technical solutions which are compatible with your current Debian system (see Section 2.1.11).

### 2.7.7 Tweaking candidate version with 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.

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 2.7.6) 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**.

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.

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.

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 2.6.6 and Section 2.7.6) without **apt-pinning**.

Here is a simplified explanation of **apt-pinning** technique.
### Pin-Priority values for *apt-pinning*

<table>
<thead>
<tr>
<th>Pin-Priority</th>
<th><strong>apt-pinning effects to the package</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>1001</td>
<td>install the package even if this constitutes a downgrade of the package</td>
</tr>
<tr>
<td>990</td>
<td>used as the default for the <strong>target release</strong> archive</td>
</tr>
<tr>
<td>500</td>
<td>used as the default for the <strong>normal</strong> archive</td>
</tr>
<tr>
<td>100</td>
<td>used as the default for the <strong>NotAutomatic</strong> and <strong>ButAutomaticUpgrades</strong> archive</td>
</tr>
<tr>
<td>100</td>
<td>used for the <strong>installed</strong> package</td>
</tr>
<tr>
<td>1</td>
<td>used as the default for the <strong>NotAutomatic</strong> archive</td>
</tr>
<tr>
<td>-1</td>
<td><strong>never install</strong> the package even if recommended</td>
</tr>
</tbody>
</table>

Table 2.18: List of notable Pin-Priority values for *apt-pinning* technique.

The APT system chooses the highest Pin-Priority upgrading package from available package sources defined in the "*/etc/apt/sources.list*" 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 2.7.11).

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 the 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/ bookworm-backports/main Packages**".

#### 2.7.8 Blocking packages installed by "Recommends"

⚠️ **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.

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.

<table>
<thead>
<tr>
<th>Package: <strong>package-1</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin: version *</td>
<td></td>
</tr>
<tr>
<td>Pin-Priority: -1</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Package: <strong>package-2</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin: version *</td>
<td></td>
</tr>
<tr>
<td>Pin-Priority: -1</td>
<td></td>
</tr>
</tbody>
</table>
2.7.9 Tracking testing with some packages from unstable

**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.

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 contrib non-free
deb http://deb.debian.org/debian/ unstable main contrib non-free
deb http://security.debian.org/debian-security testing-security main contrib
```

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-upgrade**" and "**aptitude full-upgrade**") upgrades packages which were installed from **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.

### 2.7.10 Tracking unstable with some packages from experimental

**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.

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 contrib non-free
deb http://deb.debian.org/debian/ experimental main contrib non-free
deb http://security.debian.org/ testing-security main contrib
```

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.

### 2.7.11 Emergency downgrading

**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**
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 2.7.7). This is lazy alternative to tedious actions of many "dpkg -i broken-package_old-version.deb" commands (see Section 2.6.6).

Search lines in the "/etc/apt/sources.list" file tracking unstable as the following.
deb http://deb.debian.org/debian/ sid main contrib non-free

Replace it with the following to track testing.

deb http://deb.debian.org/debian/ testing main contrib non-free

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.

2.7.12 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

Caution
There is no guarantee for the procedure described here to work without extra manual efforts for system differences.
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 contrib non-free
```

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 6.5) as described in apt.conf(5) and in "/usr/share/doc/apt/examples/configure-index.gz".

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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>approx</td>
<td>V:0, I:0</td>
<td>7124</td>
<td>caching proxy server for Debian archive files (compiled OCaml program)</td>
</tr>
<tr>
<td>apt-cacher</td>
<td>V:0, I:0</td>
<td>266</td>
<td>Caching proxy for Debian package and source files (Perl program)</td>
</tr>
<tr>
<td>apt-cacher-ng</td>
<td>V:4, I:4</td>
<td>1816</td>
<td>Caching proxy for distribution of software packages (compiled C++ program)</td>
</tr>
</tbody>
</table>

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 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), taskset(8), apt(8), apt-get(8), apt-config(8), apt-secure(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/index.html" from the apt-doc package; and

- 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:
  - "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. Here is a rough overview of the key points of the Debian system initialization. Since the Debian system is a moving target, you should refer to the latest documentation.

- Debian Linux Kernel Handbook is the primary source of information on the Debian kernel.
- `bootup(7)` describes the system bootup process based on `systemd`. (Recent Debian)
- `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.

3.1.1 Stage 1: the UEFI

The Unified Extensible Firmware Interface (UEFI) defines a boot manager as part of the UEFI specification. When a computer is powered on, the boot manager is the 1st stage of the boot process which checks the boot configuration and based on its settings, then executes the specified OS boot loader or operating system kernel (usually boot loader). The boot configuration is defined by variables stored in NVRAM, including variables that indicate the file system paths to OS loaders or OS kernels.

An EFI system partition (ESP) is a data storage device partition that is used in computers adhering to the UEFI specification. Accessed by the UEFI firmware when a computer is powered up, it stores UEFI applications and the files these applications need to run, including operating system boot loaders. (On the legacy PC system, BIOS stored in the MBR may be used instead.)
3.1.2 Stage 2: the boot loader

The **boot loader** is the 2nd stage of the boot process which is started by the UEFI. 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 5.x Linux kernel is technically the initramfs (initial RAM filesystem) image.

There are many bootloaders and configuration options available.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>initrd</th>
<th>bootloader</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>grub-efi-amd64</td>
<td>I:318</td>
<td>184</td>
<td>Supported</td>
<td>GRUB UEFI</td>
<td>This is smart enough to understand disk partitions and filesystems such as vfat, ext4, … (UEFI)</td>
</tr>
<tr>
<td>grub-pc</td>
<td>V:21, I:652</td>
<td>557</td>
<td>Supported</td>
<td>GRUB 2</td>
<td>This is smart enough to understand disk partitions and filesystems such as vfat, ext4, … (BIOS)</td>
</tr>
<tr>
<td>grub-rescue-pc</td>
<td>V:0, I:0</td>
<td>6613</td>
<td>Supported</td>
<td>GRUB 2</td>
<td>This is GRUB 2 bootable rescue images (CD and floppy) (PC/BIOS version)</td>
</tr>
<tr>
<td>syslinux</td>
<td>V:3, I:36</td>
<td>344</td>
<td>Supported</td>
<td>Isolinux</td>
<td>This understands the ISO9660 filesystem. This is used by the boot CD.</td>
</tr>
<tr>
<td>syslinux</td>
<td>V:3, I:36</td>
<td>344</td>
<td>Supported</td>
<td>Syslinux</td>
<td>This understands the MSDOS filesystem (FAT). This is used by the boot floppy.</td>
</tr>
<tr>
<td>loadlin</td>
<td>V:0, I:0</td>
<td>90</td>
<td>Supported</td>
<td>Loadlin</td>
<td>New system is started from the FreeDOS/MSDOS system.</td>
</tr>
<tr>
<td>mbr</td>
<td>V:0, I:4</td>
<td>50</td>
<td>Not supported</td>
<td>MBR by Neil Turton</td>
<td>This is free software which substitutes MSDOS MBR. This only understands disk partitions.</td>
</tr>
</tbody>
</table>

Table 3.1: List of boot loaders

⚠️ **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 UEFI system, GRUB2 first reads the ESP partition and uses UUID specified for search.fs_uuid in "/boot/efi/EFI/debian/grub.cfg" to determine the partition of the GRUB2 menu configuration file " /boot/grub/grub.cfg".

The key part of the GRUB2 menu configuration file looks like:

```plaintext
menuentry 'Debian GNU/Linux' ... {
    load_video
    insmod gzio
    insmod part_gpt
    insmod ext2
    search --no-floppy --fs-uuid --set=root fe3e1db5-6454-46d6-a14c-071208ebe4b1
    echo 'Loading Linux 5.10.0-6-amd64 ...'
    linux /boot/vmlinuz-5.10.0-6-amd64 root=UUID=fe3e1db5-6454-46d6-a14c-071208ebe4b1 ro quiet
    echo 'Loading initial ramdisk ...'
    initrd /boot/initrd.img-5.10.0-6-amd64
}
```

For this part of /boot/grub/grub.cfg, this menu entry means the following.
<table>
<thead>
<tr>
<th>setting</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRUB2 modules loaded</td>
<td><code>gzio, part_gpt, ext2</code></td>
</tr>
<tr>
<td>root file system partition used</td>
<td>partition identified by UUID=<code>fe3e1db5-6454-46d6-a14c-071208ebe4b1</code></td>
</tr>
<tr>
<td>kernel image path in the root file system</td>
<td><code>/boot/vmlinuz-5.10.0-6-amd64</code></td>
</tr>
<tr>
<td>kernel boot parameter used</td>
<td>&quot;root=UUID=fe3e1db5-6454-46d6-a14c-071208ebe4b1 ro quiet&quot;</td>
</tr>
<tr>
<td>initrd image path in the root file system</td>
<td><code>/boot/initrd.img-5.10.0-6-amd64</code></td>
</tr>
</tbody>
</table>

Table 3.2: The meaning of the menu entry of the above part of `/boot/grub/grub.cfg`

**Tip**
You can enable to see kernel boot log messages by removing `quiet` in `"/boot/grub/grub.cfg"`. For the persistent change, please edit "GRUB_CMDLINE_LINUX_DEFAULT="quiet"" line in "/etc/default/grub".

**Tip**
You can customize GRUB splash image by setting `GRUB_BACKGROUND` variable in "/etc/default/grub" pointing to the image file or placing the image file itself in "/boot/grub/".

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_program". "*/sbin/init" is symlinked to "*/lib/systemd/systemd" after Debian 8 Jessie (released in 2015).

**Tip**
The actual init command on your system can be verified by the "ps --pid 1 -f" command.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemd</td>
<td>V:856,1:963</td>
<td>10423</td>
<td>event-based <em>init</em>(8) daemon for concurrency (alternative to sysvinit)</td>
</tr>
<tr>
<td>cloud-init</td>
<td>V:3,1:5</td>
<td>2870</td>
<td>initialization system for infrastructure cloud instances</td>
</tr>
<tr>
<td>systemd-sysv</td>
<td>V:829,1:961</td>
<td>76</td>
<td>the manual pages and links needed for systemd to replace sysvinit</td>
</tr>
<tr>
<td>init-system-helpers</td>
<td>V:701,1:971</td>
<td>140</td>
<td>helper tools for switching between sysvinit and systemd</td>
</tr>
<tr>
<td>initscripts</td>
<td>V:38,1:148</td>
<td>197</td>
<td>scripts for initializing and shutting down the system</td>
</tr>
<tr>
<td>sysvinit-core</td>
<td>V:5,1:6</td>
<td>371</td>
<td>System-V-like <em>init</em>(8) utilities</td>
</tr>
<tr>
<td>sysv-rc</td>
<td>V:75,1:160</td>
<td>88</td>
<td>System-V-like runlevel change mechanism</td>
</tr>
<tr>
<td>sysvinit-utils</td>
<td>V:895,1:999</td>
<td>102</td>
<td>System-V-like utilities (<em>startpar</em>(8), <em>boot logd</em>(8), ...)</td>
</tr>
<tr>
<td>lsb-base</td>
<td>V:715,1:759</td>
<td>12</td>
<td>Linux Standard Base 3.2 <em>init</em> script functionality</td>
</tr>
<tr>
<td>insserv</td>
<td>V:102,1:158</td>
<td>153</td>
<td>tool to organize boot sequence using LSB <em>init.d</em> script dependencies</td>
</tr>
<tr>
<td>kexec-tools</td>
<td>V:1,1:6</td>
<td>316</td>
<td>kexec tool for <em>kexec</em>(8) reboots (warm reboot)</td>
</tr>
<tr>
<td>systemd-bootchart</td>
<td>V:0,1:0</td>
<td>132</td>
<td>boot process performance analyser</td>
</tr>
<tr>
<td>mingetty</td>
<td>V:0,1:2</td>
<td>38</td>
<td>console-only <em>getty</em>(8)</td>
</tr>
<tr>
<td>mgetty</td>
<td>V:0,1:0</td>
<td>315</td>
<td>smart modem <em>getty</em>(8) replacement</td>
</tr>
</tbody>
</table>

Table 3.3: List of boot utilities for the Debian system

**Tip**
See [Debian wiki: BootProcessSpeedup](https://www.debian.org/wikis/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.

The spawned processes are placed in individual [Linux control groups](https://www.debian.org/doc/debian-policy/ch-controlgroups.html) named after the unit which they belong to in the private systemd hierarchy (see [cgroups](https://www.debian.org/doc/debian-policy/ch-controlgroups.html) and Section 4.7.4).

The unit configuration files 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 "MAPPING 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* and Section 9.6.7. The configuration of the encrypted filesystem is set in "/etc/crypttab". See *crypttab*.

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 Cloud system initialization

The cloud system instance may be launched as a clone of "Debian Official Cloud Images" or similar images. For such system instance, personalities such as hostname, filesystem, networking, locale, SSH keys, users and groups may be configured using functionalities provided by *cloud-init* and *netplan.io* packages with multiple data sources such as files placed in the original system image and external data provided during its launch. These packages enable the declarative system configuration using YAML data.

See more at "Cloud Computing with Debian and its descendants", "Cloud-init documentation" and Section 5.4.

3.3 The kernel message

The kernel error message displayed to the console can be configured by setting its threshold level.

```bash
# dmesg -n3
```

<table>
<thead>
<tr>
<th>error level value</th>
<th>error level name</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KERN_EMERG</td>
<td>system is unusable</td>
</tr>
<tr>
<td>1</td>
<td>KERN_ALERT</td>
<td>action must be taken immediately</td>
</tr>
<tr>
<td>2</td>
<td>KERN_CRIT</td>
<td>critical conditions</td>
</tr>
<tr>
<td>3</td>
<td>KERN_ERR</td>
<td>error conditions</td>
</tr>
<tr>
<td>4</td>
<td>KERN_WARNING</td>
<td>warning conditions</td>
</tr>
<tr>
<td>5</td>
<td>KERN_NOTICE</td>
<td>normal but significant condition</td>
</tr>
<tr>
<td>6</td>
<td>KERN_INFO</td>
<td>informational</td>
</tr>
<tr>
<td>7</td>
<td>KERN_DEBUG</td>
<td>debug-level messages</td>
</tr>
</tbody>
</table>

Table 3.4: List of kernel error levels
3.4 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/journal”.

These binary log data are accessed by the journalctl(1) command. For example, you can display log from the last boot as:

```
$ journalctl -b
```

<table>
<thead>
<tr>
<th>Operation</th>
<th>Command snippets</th>
</tr>
</thead>
<tbody>
<tr>
<td>View log for system services and kernel from</td>
<td>journalctl -b --system</td>
</tr>
<tr>
<td>the last boot</td>
<td></td>
</tr>
<tr>
<td>View log for services of the current user from</td>
<td>journalctl -b --user</td>
</tr>
<tr>
<td>the last boot</td>
<td></td>
</tr>
<tr>
<td>View job log of &quot;$unit&quot; from the last boot</td>
<td>journalctl -b -u $unit</td>
</tr>
<tr>
<td>View job log of &quot;$unit&quot; (&quot;tail -f&quot; style) from</td>
<td>journalctl -b -u $unit -f</td>
</tr>
<tr>
<td>the last boot</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.5: List of typical journalctl command snippets

Under systemd, the system logging utility rsyslogd(8) may be uninstalled. If it is installed, it 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. This 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 9.3.2.

3.5 System management

The systemd offers not only init system but also generic system management operations with the systemctl(1) command. 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.

3.6 Other system monitors

Here are a list of other monitoring command snippets under systemd. Please read the pertinent manpages including cgroups(7).
<table>
<thead>
<tr>
<th>Operation</th>
<th>Command snippets</th>
</tr>
</thead>
<tbody>
<tr>
<td>List all target unit configuration</td>
<td><code>systemctl list-units --type=target</code></td>
</tr>
<tr>
<td>List all service unit configuration</td>
<td><code>systemctl list-units --type=service</code></td>
</tr>
<tr>
<td>List all unit configuration types</td>
<td><code>systemctl list-units --type=help</code></td>
</tr>
<tr>
<td>List all socket units in memory</td>
<td><code>systemctl list-sockets</code></td>
</tr>
<tr>
<td>List all timer units in memory</td>
<td><code>systemctl list-timers</code></td>
</tr>
<tr>
<td>Start &quot;$unit&quot;</td>
<td><code>systemctl start $unit</code></td>
</tr>
<tr>
<td>Stop &quot;$unit&quot;</td>
<td><code>systemctl stop $unit</code></td>
</tr>
<tr>
<td>Reload service-specific configuration</td>
<td><code>systemctl reload $unit</code></td>
</tr>
<tr>
<td>Stop and start all &quot;$unit&quot;</td>
<td><code>systemctl restart $unit</code></td>
</tr>
<tr>
<td>Start &quot;$unit&quot; and stop all others</td>
<td><code>systemctl isolate $unit</code></td>
</tr>
<tr>
<td>Switch to &quot;graphical&quot; (GUI system)</td>
<td><code>systemctl isolate graphical</code></td>
</tr>
<tr>
<td>Switch to &quot;multi-user&quot; (CLI system)</td>
<td><code>systemctl isolate multi-user</code></td>
</tr>
<tr>
<td>Switch to &quot;rescue&quot; (single user CLI system)</td>
<td><code>systemctl isolate rescue</code></td>
</tr>
<tr>
<td>Send kill signal to &quot;$unit&quot;</td>
<td><code>systemctl kill $unit</code></td>
</tr>
<tr>
<td>Check if &quot;$unit&quot; service is active</td>
<td><code>systemctl is-active $unit</code></td>
</tr>
<tr>
<td>Check if &quot;$unit&quot; service is failed</td>
<td><code>systemctl is-failed $unit</code></td>
</tr>
<tr>
<td>Show status of &quot;$unit&quot;</td>
<td>PID</td>
</tr>
<tr>
<td>Reset failed &quot;$unit&quot;</td>
<td><code>systemctl reset-failed $unit</code></td>
</tr>
<tr>
<td>List dependency of all unit services</td>
<td><code>systemctl list-dependencies --all</code></td>
</tr>
<tr>
<td>List unit files installed on the system</td>
<td><code>systemctl list-unit-files</code></td>
</tr>
<tr>
<td>Enable &quot;$unit&quot; (add symlink)</td>
<td><code>systemctl enable $unit</code></td>
</tr>
<tr>
<td>Disable &quot;$unit&quot; (remove symlink)</td>
<td><code>systemctl disable $unit</code></td>
</tr>
<tr>
<td>Unmask &quot;$unit&quot; (remove symlink to &quot;/dev/null&quot;)</td>
<td><code>systemctl unmask $unit</code></td>
</tr>
<tr>
<td>Mask &quot;$unit&quot; (add symlink to &quot;/dev/null&quot;)</td>
<td><code>systemctl mask $unit</code></td>
</tr>
<tr>
<td>Get default-target setting</td>
<td><code>systemctl get-default</code></td>
</tr>
<tr>
<td>Set default-target to &quot;graphical&quot; (GUI system)</td>
<td><code>systemctl set-default graphical</code></td>
</tr>
<tr>
<td>Set default-target to &quot;multi-user&quot; (CLI system)</td>
<td><code>systemctl set-default multi-user</code></td>
</tr>
<tr>
<td>Show job environment</td>
<td><code>systemctl show-environment</code></td>
</tr>
<tr>
<td>Set job environment &quot;variable&quot; to &quot;value&quot;</td>
<td><code>systemctl set-environment variable=value</code></td>
</tr>
<tr>
<td>Unset job environment &quot;variable&quot;</td>
<td><code>systemctl unset-environment variable</code></td>
</tr>
<tr>
<td>Reload all unit files and daemons</td>
<td><code>systemctl daemon-reload</code></td>
</tr>
<tr>
<td>Shut down the system</td>
<td><code>systemctl poweroff</code></td>
</tr>
<tr>
<td>Shut down and reboot the system</td>
<td><code>systemctl reboot</code></td>
</tr>
<tr>
<td>Suspend the system</td>
<td><code>systemctl suspend</code></td>
</tr>
<tr>
<td>Hibernate the system</td>
<td><code>systemctl hibernate</code></td>
</tr>
</tbody>
</table>

Table 3.6: List of typical systemctl command snippets
Table 3.7: List of other monitoring command snippets under systemd

3.7 Customizing systemd

3.7.1 Socket activation

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.

```bash
$ sudo systemctl stop sshd.service
$ sudo systemctl mask sshd.service
```

The on-demand socket activation system of the classic Unix services was through the inetd (or xinetd) 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

```ini
[Unit]
Description=SSH Socket for Per-Connection Servers

[Socket]
ListenStream=22
Accept=yes

[Install]
WantedBy=sockets.target
```

**sshd@.service** as the matching service file of sshd.socket

```ini
[Unit]
Description=SSH Per-Connection Server

[Service]
ExecStart=/usr/sbin/sshd -i
StandardInput=socket
```

Then reload.

```bash
$ sudo systemctl daemon-reload
```
3.8 The udev system

The udev system provides mechanism for the automatic hardware discovery and initialization (see udev(7)) since Linux kernel 2.6. 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 3.8.1), 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.

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 9.6.3.

Since the udev system is somewhat a moving target, I leave details to other documentations and describe the minimum information here.

3.8.1 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 3.8) 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 5.7), 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/blacklist" 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 9.5.3.
You may configure hardware at boot time to activate expected hardware features. See Section 9.5.4.
You can probably add support for your special device by recompiling the kernel. See Section 9.10.
Chapter 4

Authentication and access controls

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.

---

### 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.

<table>
<thead>
<tr>
<th>file</th>
<th>permission</th>
<th>user</th>
<th>group</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/passwd</code></td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td>(sanitized) user account information</td>
</tr>
<tr>
<td><code>/etc/shadow</code></td>
<td>-rw-r-----</td>
<td>root</td>
<td>shadow</td>
<td>secure user account information</td>
</tr>
<tr>
<td><code>/etc/group</code></td>
<td>-rw-r--r--</td>
<td>root</td>
<td>root</td>
<td>group information</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>content</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>(empty)</td>
<td>passwordless account</td>
</tr>
<tr>
<td>x</td>
<td>the encrypted password is in &quot;/etc/shadow&quot;</td>
</tr>
</tbody>
</table>

Table 4.2: The second entry content of "/etc/passwd"

"/etc/shadow" contains the following.

```
user1:$1$Xop9FYH9$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.

<table>
<thead>
<tr>
<th>command</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>getent passwd user_name</td>
<td>browse account information of &quot;user_name&quot;</td>
</tr>
<tr>
<td>getent shadow user_name</td>
<td>browse shadowed account information of &quot;user_name&quot;</td>
</tr>
<tr>
<td>getent group group_name</td>
<td>browse group information of &quot;group_name&quot;</td>
</tr>
<tr>
<td>passwd</td>
<td>manage password for the account</td>
</tr>
<tr>
<td>passwd -e</td>
<td>set one-time password for the account activation</td>
</tr>
<tr>
<td>chage</td>
<td>manage password aging information</td>
</tr>
</tbody>
</table>

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 alpha betics
- 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.
Debian Reference

Table 4.4: List of tools to generate password

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>command</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>whois</td>
<td>V:25, I:262</td>
<td>387</td>
<td>mkpasswd</td>
<td>over-featured front end to the crypt(3) library</td>
</tr>
<tr>
<td>openssl</td>
<td>V:823, I:995</td>
<td>2290</td>
<td>openssl</td>
<td>compute password hashes (OpenSSL). passwd(1ssl)</td>
</tr>
</tbody>
</table>

4.4 Creating encrypted password

There are independent tools to generate encrypted passwords with salt.

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 summarized 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libpam-modules</td>
<td>V:879, I:999</td>
<td>1005</td>
<td>Pluggable Authentication Modules (basic service)</td>
</tr>
<tr>
<td>libpam-ldap</td>
<td>V:1, I:17</td>
<td>249</td>
<td>Pluggable Authentication Module allowing LDAP interfaces</td>
</tr>
<tr>
<td>libpam-cracklib</td>
<td>V:1, I:9</td>
<td>117</td>
<td>Pluggable Authentication Module to enable cracklib support</td>
</tr>
<tr>
<td>libpam-systemd</td>
<td>V:530, I:930</td>
<td>575</td>
<td>Pluggable Authentication Module to register user sessions for logind</td>
</tr>
<tr>
<td>libpam-doc</td>
<td>I:0</td>
<td>1002</td>
<td>Pluggable Authentication Modules (documentation in html and text)</td>
</tr>
<tr>
<td>libc6</td>
<td>V:915, I:999</td>
<td>12989</td>
<td>GNU C Library: Shared libraries which also provides &quot;Name Service Switch&quot; service</td>
</tr>
<tr>
<td>glibc-doc</td>
<td>I:9</td>
<td>3502</td>
<td>GNU C Library: Manpages</td>
</tr>
<tr>
<td>glibc-doc-reference</td>
<td>I:4</td>
<td>13188</td>
<td>GNU C Library: Reference manual in info, pdf and html format (non-free)</td>
</tr>
<tr>
<td>libnss-mdns</td>
<td>I:512</td>
<td>141</td>
<td>NSS module for Multicast DNS name resolution</td>
</tr>
<tr>
<td>libnss-ldap</td>
<td>V:0, I:6</td>
<td>265</td>
<td>NSS module for using LDAP as a naming service</td>
</tr>
<tr>
<td>libnss-ldapd</td>
<td>I:15</td>
<td>129</td>
<td>NSS module for using LDAP as a naming service (new fork of libnss-ldap)</td>
</tr>
</tbody>
</table>

Table 4.5: List of notable PAM and NSS systems

- "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.

<table>
<thead>
<tr>
<th>configuration file</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>/etc/pam.d/program_name</code></td>
<td>set up PAM configuration for the &quot;program_name&quot; program; see pam(7) and pam.d(5)</td>
</tr>
<tr>
<td><code>/etc/nsswitch.conf</code></td>
<td>set up NSS configuration with the entry for each service. See nsswitch.conf(5)</td>
</tr>
<tr>
<td><code>/etc/nologin</code></td>
<td>limit the user login by the pam_nologin(8) module</td>
</tr>
<tr>
<td><code>/etc/security/limits.conf</code></td>
<td>set resource restraint (ulimit, core, ...) by the pam_limits(8) module</td>
</tr>
<tr>
<td><code>/etc/security/time.conf</code></td>
<td>set time restraint by the pam_time(8) module</td>
</tr>
<tr>
<td><code>/etc/systemd/logind.conf</code></td>
<td>set systemd login manager configuration (see logind.conf(5) and systemd-logind.service(8))</td>
</tr>
</tbody>
</table>

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.

On a typical GNOME system which automatically installs libpam-gnome-keyring,"/etc/pam.d/common-password" looks like:

```
# here are the per-package modules (the "Primary" block)
password requisite pam_cracklib.so retry=3 minlen=8 difok=3
password [success=1 default=ignore] pam_unix.so obscure use_authtok try_first_pass ← yescrypt
# here’s the fallback if no module succeeds
password requisite pam_deny.so
# prime the stack with a positive return value if there isn’t one already;
# this avoids us returning an error just because nothing sets a success code
# since the modules above will each just jump around
password required pam_permit.so
# and here are more per-package modules (the "Additional" block)
password optional pam_gnome_keyring.so
# end of pam-auth-update config
```
4.6 Security of authentication

Note
The information here may not be sufficient for your security needs but it should be a good start.

4.6.1 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.

<table>
<thead>
<tr>
<th>insecure service name</th>
<th>port</th>
<th>secure service name</th>
<th>port</th>
</tr>
</thead>
<tbody>
<tr>
<td>www (http)</td>
<td>80</td>
<td>https</td>
<td>443</td>
</tr>
<tr>
<td>smtp (mail)</td>
<td>25</td>
<td>ssmt (smtps)</td>
<td>465</td>
</tr>
<tr>
<td>ftp-data</td>
<td>20</td>
<td>ftps-data</td>
<td>989</td>
</tr>
<tr>
<td>ftp</td>
<td>21</td>
<td>ftps</td>
<td>990</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>telnets</td>
<td>992</td>
</tr>
<tr>
<td>imap2</td>
<td>143</td>
<td>imaps</td>
<td>993</td>
</tr>
<tr>
<td>pop3</td>
<td>110</td>
<td>pop3s</td>
<td>995</td>
</tr>
<tr>
<td>ldap</td>
<td>389</td>
<td>ldaps</td>
<td>636</td>
</tr>
</tbody>
</table>

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.)

4.6.2 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 6.3.

4.6.3 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 5.7) together with the following security tools may improve the security situation.

4.6.4 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
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>knockd</td>
<td>V:0, I:2</td>
<td>110</td>
<td>small port-knock daemon knockd(1) and client knock(1)</td>
</tr>
<tr>
<td>fall2ban</td>
<td>V:100, I:113</td>
<td>2129</td>
<td>ban IPs that cause multiple authentication errors</td>
</tr>
<tr>
<td>libpam-shield</td>
<td>V:0, I:0</td>
<td>115</td>
<td>lock out remote attackers trying password guessing</td>
</tr>
</tbody>
</table>

Table 4.8: List of tools to provide extra security measures

- Lock UEFI/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 UEFI/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. Furthermore, he may compromise password for all user accounts using brute force password cracking tools such as john and crack packages (see Section 9.5.11). 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 9.9). You always need password to boot the system, though.

### 4.7 Other access controls

There are access controls to the system other than the password based authentication and file permissions.

**Note**

See Section 9.4.15 for restricting the kernel secure attention key (SAK) feature.
4.7.1 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”.

4.7.2 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).

4.7.3 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"
- Systemd 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 3.5, Section 4.5.1, and Section 5.7.

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.)
4.7.4 Linux security features

Linux kernel has evolved and supports security features not found in traditional UNIX implementations. Linux supports extended attributes which extend the traditional UNIX attributes (see xattr(7)).

Linux divides the privileges traditionally associated with superuser into distinct units, known as capabilities(7), which can be independently enabled and disabled. Capabilities are a per-thread attribute since kernel version 2.2.

The Linux Security Module (LSM) framework provides a mechanism for various security checks to be hooked by new kernel extensions. For example:

- AppArmor
- Security-Enhanced Linux (SELinux)
- Smack (Simplified Mandatory Access Control Kernel)
- Tomoyo Linux

Since these extensions may tighten privilege model tighter than the ordinary Unix-like security model policies, even the root power may be restricted. You are advised to read the Linux Security Module (LSM) framework document at kernel.org.

Linux namespaces wrap a global system resource in an abstraction that makes it appear to the processes within the namespace that they have their own isolated instance of the global resource. Changes to the global resource are visible to other processes that are members of the namespace, but are invisible to other processes. Since kernel version 5.6, there are 8 kinds of namespaces (see namespaces(7), unshare(1), nsenter(1)).

As of Debian 11 Bullseye (2021), Debian uses unified cgroup hierarchy (a.k.a. cgroups-v2).

Usage examples of namespaces with cgroups to isolate their processes and to allow resource control are:

- Systemd. See Section 3.2.
- Sandbox environment. See Section 7.6.
- Linux containers such as Docker, LXC. See Section 9.11.

These functionalities can’t be realized by Section 4.1. These advanced topics are mostly out-of-scope for this introductory document.
Chapter 5

Network setup

Tip
For modern Debian specific guide to the networking, read The Debian Administrator's Handbook — Configuring the Network.

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.

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 localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```
<table>
<thead>
<tr>
<th>packages</th>
<th>popcon</th>
<th>size</th>
<th>type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network-manager</td>
<td>V:387, I:457</td>
<td>15413</td>
<td>config::NM</td>
<td>NetworkManager (daemon): manage the network automatically</td>
</tr>
<tr>
<td>network-manager</td>
<td>V:122, I:372</td>
<td>5583</td>
<td>config::NM</td>
<td>NetworkManager (GNOME frontend)</td>
</tr>
<tr>
<td>netplan.io</td>
<td>V:2, I:4</td>
<td>249</td>
<td>config::NM</td>
<td>Netplan (generator): Unified, declarative interface to NetworkManager and systemd-networkd backends</td>
</tr>
<tr>
<td>ifupdown</td>
<td>V:614, I:981</td>
<td>199</td>
<td>config::ifupdown</td>
<td>standardized tool to bring up and down the network (Debian specific)</td>
</tr>
<tr>
<td>isc-dhcp-client</td>
<td>V:215, I:982</td>
<td>2866</td>
<td>config::low-level</td>
<td>DHCP client</td>
</tr>
<tr>
<td>ppoeconf</td>
<td>V:0, I:5</td>
<td>186</td>
<td>config::helper</td>
<td>configuration helper for PPPoE connection</td>
</tr>
<tr>
<td>wpasupplicant</td>
<td>V:351, I:513</td>
<td>3846</td>
<td>config::helper</td>
<td>client support for WPA and WPA2 (IEEE 802.11i)</td>
</tr>
<tr>
<td>wpa_gui</td>
<td>V:0, I:2</td>
<td>770</td>
<td>config::helper</td>
<td>Qt GUI client for wpasupplicant</td>
</tr>
<tr>
<td>wireless-tools</td>
<td>V:178, I:243</td>
<td>292</td>
<td>config::helper</td>
<td>tools for manipulating Linux Wireless Extensions</td>
</tr>
<tr>
<td>iw</td>
<td>V:34, I:477</td>
<td>302</td>
<td>config::helper</td>
<td>tool for configuring Linux wireless devices</td>
</tr>
<tr>
<td>iproute2</td>
<td>V:728, I:969</td>
<td>3637</td>
<td>config::iproute2</td>
<td>iproute2, IPv6 and other advanced network configuration: ip(8), tc(8), etc</td>
</tr>
<tr>
<td>iptables</td>
<td>V:315, I:749</td>
<td>2408</td>
<td>config::Netfilter</td>
<td>administration tools for packet filtering and NAT (Netfilter)</td>
</tr>
<tr>
<td>iputils-ping</td>
<td>V:200, I:997</td>
<td>120</td>
<td>test</td>
<td>test network reachability of a remote host by hostname or IP address (iproute2)</td>
</tr>
<tr>
<td>iputils-arping</td>
<td>V:3, I:41</td>
<td>49</td>
<td>test</td>
<td>test network reachability of a remote host specified by the ARP address</td>
</tr>
<tr>
<td>iputils-tracepath</td>
<td>V:2, I:32</td>
<td>45</td>
<td>test</td>
<td>trace the network path to a remote host</td>
</tr>
<tr>
<td>ethtool</td>
<td>V:96, I:266</td>
<td>735</td>
<td>test</td>
<td>display or change Ethernet device settings</td>
</tr>
<tr>
<td>mtr-tiny</td>
<td>V:5, I:46</td>
<td>156</td>
<td>test::low-level</td>
<td>trace the network path to a remote host (curses)</td>
</tr>
<tr>
<td>mtr</td>
<td>V:4, I:41</td>
<td>209</td>
<td>test::low-level</td>
<td>trace the network path to a remote host (curses and GTK)</td>
</tr>
<tr>
<td>gnome-nettool</td>
<td>V:1, I:19</td>
<td>2492</td>
<td>test::low-level</td>
<td>tools for common network information operations (GNOME)</td>
</tr>
<tr>
<td>nmap</td>
<td>V:25, I:202</td>
<td>4498</td>
<td>test::low-level</td>
<td>network mapper / port scanner (Nmap, console)</td>
</tr>
<tr>
<td>tcpdump</td>
<td>V:17, I:177</td>
<td>1340</td>
<td>test::low-level</td>
<td>network traffic analyzer (Tcpdump, console)</td>
</tr>
<tr>
<td>wireshark</td>
<td>I:45</td>
<td>28</td>
<td>test::low-level</td>
<td>network traffic analyzer (Wireshark, GTK)</td>
</tr>
<tr>
<td>tshark</td>
<td>V:2, I:25</td>
<td>398</td>
<td>test::low-level</td>
<td>network traffic analyzer (console)</td>
</tr>
<tr>
<td>tcptrace</td>
<td>V:0, I:2</td>
<td>401</td>
<td>test::low-level</td>
<td>produce a summarization of the connections from tcpdump output</td>
</tr>
<tr>
<td>snort</td>
<td>V:0, I:0</td>
<td>2203</td>
<td>test::low-level</td>
<td>flexible network intrusion detection system (Snort)</td>
</tr>
<tr>
<td>ntopng</td>
<td>V:0, I:1</td>
<td>15904</td>
<td>test::low-level</td>
<td>display network usage in web browser</td>
</tr>
<tr>
<td>dnsutils</td>
<td>V:19, I:299</td>
<td>272</td>
<td>test::low-level</td>
<td>network clients provided with BIND: nslookup(8), nsupdate(8), dig(8)</td>
</tr>
<tr>
<td>dlint</td>
<td>V:0, I:4</td>
<td>53</td>
<td>test::low-level</td>
<td>check DNS zone information using nameserver lookups</td>
</tr>
<tr>
<td>dnstracer</td>
<td>V:0, I:1</td>
<td>61</td>
<td>test::low-level</td>
<td>trace a chain of DNS servers to the source</td>
</tr>
</tbody>
</table>

Table 5.1: List of network configuration tools
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.

```plaintext
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 mdns44 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 choosing a domain name used only within LAN.

### 5.1.2 The network interface name

The systemd uses "Predictable Network Interface Names" such as "enp0s25".

### 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.
<table>
<thead>
<tr>
<th>Class</th>
<th>network addresses</th>
<th>net mask</th>
<th>net mask /bits</th>
<th>number of subnets</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10.x.x.x</td>
<td>255.0.0.0</td>
<td>/8</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>172.16.x.x — 172.31.x.x</td>
<td>255.255.0.0</td>
<td>/16</td>
<td>16</td>
</tr>
<tr>
<td>C</td>
<td>192.168.0.x — 192.168.255.x</td>
<td>255.255.255.0</td>
<td>/24</td>
<td>256</td>
</tr>
</tbody>
</table>

Table 5.2: List of network address ranges

**Note**
IP address written with colon are IPv6 address, e.g., "::1" for localhost.

**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

Although most hardware devices are supported by the Debian system, there are some network devices which require DFSG non-free firmware to support them. Please see Section 9.10.5.

### 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 can manage the network connection via management daemon software such as NetworkManager (NM) (network-manager 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".
5.2.1 GUI network configuration tools

Official documentations for NM on Debian are provided in "/usr/share/doc/network-manager/README.Debian". 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 by the following.

   $ sudo systemctl restart network-manager

4. Configure your network via GUI.

   **Note**
   Only interfaces which are **not** listed in "/etc/network/interfaces" are managed by NM 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.

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 modern network configuration for cloud

The modern network configuration for cloud may use `cloud-init` and `netplan.io` packages (see Section 3.2.4).

The `netplan.io` package supports `systemd-networkd` and `NetworkManager` as its network configuration backends, and enables the declarative network configuration using YAML data. When you change YAML:

- Run "netplan generate" command to generate all the necessary backend configuration from YAML.
- Run "netplan apply" command to apply the generated configuration to the backends.

See "Netplan documentation", `netplan(5)`, `netplan-generate(8)`, and `netplan-apply(8)`.

See also "Cloud-init documentation" (especially around "Configuration sources" and "Netplan Passthrough") for how `cloud-init` can integrate `netplan.io` configuration with alternative data sources.

5.4.1 The modern network configuration for cloud with DHCP

A DHCP client configuration can be set up by creating a data source file `/etc/netplan/50-dhcp.yaml`:

```
network:
  version: 2
  ethernets:
    all-en:
      match:
        name: "en*"
      dhcp4: true
      dhcp6: true
```

5.4.2 The modern network configuration for cloud with static IP

A static network configuration can be set up by creating a data source file `/etc/netplan/50-static.yaml`:

```
network:
  version: 2
  ethernets:
    eth0:
      addresses:
        - 192.168.0.15/24
      routes:
        - to: default
          via: 192.168.0.1
```

5.4.3 The modern network configuration for cloud with Network Manger

The network client configuration using Network Manger infrastructure can be set up by creating a data source file `/etc/netplan/00-network-manager.yaml`:

```
network:
  version: 2
  renderer: NetworkManager
```

5.5 The low level network configuration

For the low level network configuration on Linux, use the `iproute2` programs (`ip(8)`, `⋯`).
5.5.1 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.

<table>
<thead>
<tr>
<th>obsolete net-tools</th>
<th>new iproute2 etc.</th>
<th>manipulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ifconfig(8)</td>
<td>ip addr</td>
<td>protocol (IP or IPv6) address on a device</td>
</tr>
<tr>
<td>route(8)</td>
<td>ip route</td>
<td>routing table entry</td>
</tr>
<tr>
<td>arp(8)</td>
<td>ip neigh</td>
<td>ARP or NDISC cache entry</td>
</tr>
<tr>
<td>ipmaddr</td>
<td>ip maddr</td>
<td>multicast address</td>
</tr>
<tr>
<td>iptunnel</td>
<td>ip tunnel</td>
<td>tunnel over IP</td>
</tr>
<tr>
<td>nameif(8)</td>
<td>ifrename(8)</td>
<td>name network interfaces based on MAC addresses</td>
</tr>
<tr>
<td>mii-tool(8)</td>
<td>ethtool(8)</td>
<td>Ethernet device settings</td>
</tr>
</tbody>
</table>

Table 5.3: Translation table from obsolete net-tools commands to new iproute2 commands

See ip(8) and IPROUTE2 Utility Suite Howto.

5.5.2 Safe low level network operations

You may use low level network commands as follows safely since they do not change network configuration.

<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip addr show</td>
<td>display the link and address status of active interfaces</td>
</tr>
<tr>
<td>route -n</td>
<td>display all the routing table in numerical addresses</td>
</tr>
<tr>
<td>ip route show</td>
<td>display all the routing table in numerical addresses</td>
</tr>
<tr>
<td>arp</td>
<td>display the current content of the ARP cache tables</td>
</tr>
<tr>
<td>ip neigh</td>
<td>display the current content of the ARP cache tables</td>
</tr>
<tr>
<td>ping yahoo.com</td>
<td>check the Internet connection to “yahoo.com”</td>
</tr>
<tr>
<td>whois yahoo.com</td>
<td>check who registered “yahoo.com” in the domains database</td>
</tr>
<tr>
<td>traceroute yahoo.com</td>
<td>trace the Internet connection to “yahoo.com”</td>
</tr>
<tr>
<td>tracepath yahoo.com</td>
<td>trace the Internet connection to “yahoo.com”</td>
</tr>
<tr>
<td>mtr yahoo.com</td>
<td>trace the Internet connection to “yahoo.com” (repeatedly)</td>
</tr>
<tr>
<td>dig [@[dns-server.com] example.com [{a</td>
<td>mx</td>
</tr>
<tr>
<td>iptables -L -n</td>
<td>check packet filter</td>
</tr>
<tr>
<td>netstat -a</td>
<td>find all open ports</td>
</tr>
<tr>
<td>netstat -l --inet</td>
<td>find listening ports</td>
</tr>
<tr>
<td>netstat -ln --tcp</td>
<td>find listening TCP ports (numeric)</td>
</tr>
<tr>
<td>dlint example.com</td>
<td>check DNS zone information of “example.com”</td>
</tr>
</tbody>
</table>

Table 5.4: List of low level network commands

**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”.  

5.6 Network optimization

Generic network optimization is beyond the scope of this documentation. I touch only subjects pertinent to the consumer grade connection.

<table>
<thead>
<tr>
<th>packages</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iftop</td>
<td>V:7,102</td>
<td>93</td>
<td>display bandwidth usage information on an network interface</td>
</tr>
<tr>
<td>iperf</td>
<td>V:3,144</td>
<td>360</td>
<td>Internet Protocol bandwidth measuring tool</td>
</tr>
<tr>
<td>ifstat</td>
<td>V:0,17</td>
<td>58</td>
<td>InterFace STATistics Monitoring</td>
</tr>
<tr>
<td>bmon</td>
<td>V:1,17</td>
<td>144</td>
<td>portable bandwidth monitor and rate estimator</td>
</tr>
<tr>
<td>ethtstatus</td>
<td>V:0,13</td>
<td>40</td>
<td>script that quickly measures network device throughput</td>
</tr>
<tr>
<td>bmon</td>
<td>V:0,11</td>
<td>80</td>
<td>empirical stochastic bandwidth tester</td>
</tr>
<tr>
<td>bwm-ng</td>
<td>V:1,14</td>
<td>95</td>
<td>small and simple console-based bandwidth monitor</td>
</tr>
<tr>
<td>ethtstats</td>
<td>V:0,10</td>
<td>23</td>
<td>console-based Ethernet statistics monitor</td>
</tr>
<tr>
<td>ipfm</td>
<td>V:0,10</td>
<td>82</td>
<td>bandwidth analysis tool</td>
</tr>
</tbody>
</table>

Table 5.5: List of network optimization tools

5.6.1 Finding optimal MTU

NM normally sets optimal Maximum Transmission Unit (MTU) automatically.

In some occasion, you may wish to set MTU manually after experiments with ping(8) with "-M do" option to send a ICMP packet with various data packet size. MTU is the maximum succeeding datapacket size without IP fragmentation plus 28 bytes for the IPv4 and plus 48 bytes for the IPv6. For example the following finds MTU for IPv4 connection to be 1460 and MTU for IPv6 connection to be 1500.

```
$ ping -4 -c 1 -s $((1500-28)) -M do www.debian.org
PING (149.20.4.15) 1472(1500) bytes of data.
ping: local error: message too long, mtu=1460
--- ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

```
$ ping -4 -c 1 -s $((1460-28)) -M do www.debian.org
PING (130.89.148.77) 1432(1460) bytes of data.
1440 bytes from klecker-misc.debian.org (130.89.148.77): icmp_seq=1 ttl=50 time=325 ms
--- ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 325.318/325.318/325.318/0.000 ms
```

```
$ ping -6 -c 1 -s $((1500-48)) -M do www.debian.org
PING www.debian.org (2603:400a:ffff:bb8::801f:3e)) 1452 data bytes
1460 bytes from mirror-csail.debian.org (2603:400a:ffff:bb8::801f:3e): icmp_seq=1 ttl=47 ← time=191 ms
--- www.debian.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 191.332/191.332/191.332/0.000 ms
```

This process is Path MTU (PMTU) discovery (RFC1191) and the tracepath(8) command can automate this.

In addition 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.
Table 5.6: Basic guide lines of the optimal MTU value

<table>
<thead>
<tr>
<th>network environment</th>
<th>MTU</th>
<th>rationale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dial-up link (IP: PPP)</td>
<td>576</td>
<td>standard</td>
</tr>
<tr>
<td>Ethernet link (IP: DHCP or fixed)</td>
<td>1500</td>
<td>standard and default</td>
</tr>
</tbody>
</table>

- The bigger MTU value is generally better when other limitations are met.

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` (see Section 5.7) based optimization can clamp packet size by the MSS and is useful for the router. See "TCPMSS" in `iptables`(8).

5.6.2 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 1Gbps FTTP service.

5.7 Netfilter infrastructure

Netfilter provides infrastructure for stateful firewall and network address translation (NAT) with Linux kernel modules (see Section 3.8.1).

<table>
<thead>
<tr>
<th>packages</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>iptables</td>
<td>V:315, I:799</td>
<td>2408</td>
<td>administration tools for netfilter (iptables(8) for IPv4, ip6tables(8) for IPv6)</td>
</tr>
<tr>
<td>arptables</td>
<td>V:0, I:1</td>
<td>100</td>
<td>administration tools for netfilter (arptables(8) for ARP)</td>
</tr>
<tr>
<td>ebtables</td>
<td>V:14, I:29</td>
<td>264</td>
<td>administration tools for netfilter (ebtables(8) for Ethernet bridging)</td>
</tr>
<tr>
<td>iptstate</td>
<td>V:0, I:2</td>
<td>119</td>
<td>continuously monitor netfilter state (similar to top(1))</td>
</tr>
<tr>
<td>shorewall-init</td>
<td>V:0, I:0</td>
<td>89</td>
<td>Shoreline Firewall initialization</td>
</tr>
<tr>
<td>shorewall</td>
<td>V:4, I:8</td>
<td>3090</td>
<td>Shoreline Firewall, netfilter configuration file generator</td>
</tr>
<tr>
<td>shorewall-lite</td>
<td>V:0, I:0</td>
<td>71</td>
<td>Shoreline Firewall, netfilter configuration file generator (light version)</td>
</tr>
<tr>
<td>shorewall6</td>
<td>V:0, I:1</td>
<td>1334</td>
<td>Shoreline Firewall, netfilter configuration file generator (IPv6 version)</td>
</tr>
<tr>
<td>shorewall6-lite</td>
<td>V:0, I:0</td>
<td>71</td>
<td>Shoreline Firewall, netfilter configuration file generator (IPv6, light version)</td>
</tr>
</tbody>
</table>

Table 5.7: 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/ipables/html/").
Tip
Although these were written for Linux 2.4, both \texttt{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).

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>type</th>
<th>description of web browser</th>
</tr>
</thead>
<tbody>
<tr>
<td>chromium</td>
<td>V:34, I:109</td>
<td>225852</td>
<td>X</td>
<td>Chromium, (open-source browser from Google)</td>
</tr>
<tr>
<td>firefox</td>
<td>V:8, I:12</td>
<td>230564</td>
<td></td>
<td>Firefox, (open-source browser from Mozilla, only available in Debian Unstable)</td>
</tr>
<tr>
<td>firefox-esr</td>
<td>V:197, I:436</td>
<td>228804</td>
<td></td>
<td>Firefox ESR, (Firefox Extended Support Release)</td>
</tr>
<tr>
<td>epiphany-browser</td>
<td>V:3, I:16</td>
<td>2191</td>
<td></td>
<td>GNOME, HIG compliant, Epiphany</td>
</tr>
<tr>
<td>konqueror</td>
<td>V:24, I:102</td>
<td>25892</td>
<td></td>
<td>KDE, Konqueror</td>
</tr>
<tr>
<td>dillo</td>
<td>V:0, I:5</td>
<td>1565</td>
<td></td>
<td>Dillo, (light weight browser, FLTK based)</td>
</tr>
<tr>
<td>w3m</td>
<td>V:14, I:186</td>
<td>2828</td>
<td>text</td>
<td>w3m</td>
</tr>
<tr>
<td>lynx</td>
<td>V:22, I:303</td>
<td>1935</td>
<td></td>
<td>Lynx</td>
</tr>
<tr>
<td>elinks</td>
<td>V:3, I:21</td>
<td>1645</td>
<td></td>
<td>ELinks</td>
</tr>
<tr>
<td>links</td>
<td>V:3, I:29</td>
<td>2314</td>
<td></td>
<td>Links (text only)</td>
</tr>
<tr>
<td>links2</td>
<td>V:1, I:11</td>
<td>5492</td>
<td>graphics</td>
<td>Links (console graphics without X)</td>
</tr>
</tbody>
</table>

Table 6.1: List of web browsers

6.1.1 Spoofing the User-Agent string

In order to access some overly restrictive web sites, you may need to spoof the User-Agent string returned by the web browser program. See:
• MDN Web Docs: userAgent
• Chrome Developers: Override the user agent string
• How to change your user agent
• How to Change User-Agent in Chrome, Firefox, Safari, and more
• How to Change Your Browser’s User Agent Without Installing Any Extensions
• How to change the User Agent in Gnome Web (epiphany)

Caution
Spoofed user-agent string may cause bad side effects with Java.

6.1.2 Browser extension

All modern GUI browsers support source code based browser extension and it is becoming standardized as web extensions.

6.2 The mail system

This section focuses on typical mobile workstations on consumer grade Internet connections.

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.

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 message format covering header and body data is extended by Multipurpose Internet Mail Extensions (MIME) from the plain ASCII text to other character encodings, as well as attachments of audio, video, images, and application programs.

Full featured GUI based email clients offer all the following functions using the GUI based intuitive configuration.

• It creates and interprets the message header and body data using Multipurpose Internet Mail Extensions (MIME) to deal the content data type and encoding.

• It authenticates itself to the ISP’s SMTP and IMAP servers using the legacy basic access authentication or modern OAuth 2.0. (For OAuth 2.0, set it via Desktop environment settings. E.g., "Settings" -> "Online Accounts".)

• It sends the message to the ISP’s smarthost SMTP server listening to the message submission port (587).

• It receives the stored message on the ISP’s server from the TLS/IMAP4 port (993).

• It can filter mails by their attributes.

• It may offer additional functionalities: Contacts, Calendar, Tasks, Memos.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>evolution</td>
<td>V:29, I:240</td>
<td>483</td>
<td>X GUI program (GNOME3, groupware suite)</td>
</tr>
<tr>
<td>thunderbird</td>
<td>V:50, I:120</td>
<td>224509</td>
<td>X GUI program (GTK, Mozilla Thunderbird)</td>
</tr>
<tr>
<td>kmail</td>
<td>V:36, I:94</td>
<td>23871</td>
<td>X GUI program (KDE)</td>
</tr>
<tr>
<td>mutt</td>
<td>V:16, I:161</td>
<td>7104</td>
<td>character terminal program probably used with vim</td>
</tr>
<tr>
<td>mew</td>
<td>V:0, I:0</td>
<td>2319</td>
<td>character terminal program under (x)emacs</td>
</tr>
</tbody>
</table>

Table 6.2: List of mail user agent (MUA)

6.2.2 Modern mail service limitation

Modern mail service are under some limitations in order to minimize exposure to the spam (unwanted and unsolicited email) problems.

- It is not realistic to run SMTP server on the consumer grade network to send mail directly to the remote host reliably.
- A mail may be rejected by any host en route to the destination quietly unless it appears as authentic as possible.
- It is not realistic to expect a single smarthost to send mails of unrelated source mail addresses to the remote host reliably.

This is because:

- The SMTP port (25) connections from hosts serviced by the consumer grade network to the Internet are blocked.
- The SMTP port (25) connections to hosts serviced by the consumer grade network from the Internet are blocked.
- The outgoing messages from hosts serviced by the consumer grade network to the Internet can only be sent via the message submission port (587).
- 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 in the message header to your mail account on the smarthost to prevent email address spoofing.

6.2.3 Historic mail service expectation

Some programs on Debian expect to access the /usr/sbin/sendmail command to send emails as their default or customized setting since the mail service on a UNIX system functioned historically as:

- An email is created as a text file.
- The email is handed to the /usr/sbin/sendmail command.
- For the destination address on the same host, the /usr/sbin/sendmail command makes local delivery of the email by appending it to the /var/mail/$username file.
  - Commands expecting this feature: apt-listchanges, cron, at, ...
- For the destination address on the remote host, the /usr/sbin/sendmail command makes remote transfer of the email to the destination host found by the DNS MX record using SMTP.
  - Commands expecting this feature: popcon, reportbug, bts, ...
### 6.2.4 Mail transport agent (MTA)

Debian mobile workstations can be configured just with full featured GUI based email clients without mail transfer agent (MTA) program after Debian 12 Bookworm.

Debian traditionally installed some MTA program to support programs expecting the `/usr/sbin/sendmail` command. Such MTA on mobile workstations must cope with Section 6.2.2 and Section 6.2.3.

For mobile workstations, the typical choice of MTA is either `exim4-daemon-light` or `postfix` with its installation option such as "Mail sent by smarthost; received via SMTP or fetchmail" selected. These are light weight MTAs that respect "/etc/aliases".

**Tip**
Configuring `exim4` to send the Internet mail via multiple corresponding smarthosts for multiple source email addresses is non-trivial. If you need such capability for some programs, set them up to use `msmtp` which is easy to set up for multiple source email addresses. Then leave main MTA only for a single email address.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>exim4-daemon-light</td>
<td>V:226, I:237</td>
<td>1574</td>
<td>Exim4 mail transport agent (MTA: Debian default)</td>
</tr>
<tr>
<td>exim4-daemon-heavy</td>
<td>V:6, I:6</td>
<td>1742</td>
<td>Exim4 mail transport agent (MTA: flexible alternative)</td>
</tr>
<tr>
<td>exim4-base</td>
<td>V:232, I:245</td>
<td>1702</td>
<td>Exim4 documentation (text) and common files</td>
</tr>
<tr>
<td>exim4-doc-html</td>
<td>I:1</td>
<td>3746</td>
<td>Exim4 documentation (html)</td>
</tr>
<tr>
<td>exim4-doc-info</td>
<td>I:0</td>
<td>637</td>
<td>Exim4 documentation (info)</td>
</tr>
<tr>
<td>postfix</td>
<td>V:128, I:136</td>
<td>4031</td>
<td>Postfix mail transport agent (MTA: secure alternative)</td>
</tr>
<tr>
<td>postfix-doc</td>
<td>I:7</td>
<td>4634</td>
<td>Postfix documentation (html+text)</td>
</tr>
<tr>
<td>sasl2-bin</td>
<td>V:5, I:14</td>
<td>401</td>
<td>Cyrus SASL API implementation (supplement postfix for SMTP AUTH)</td>
</tr>
<tr>
<td>cyrus-sasl2-doc</td>
<td>I:1</td>
<td>2175</td>
<td>Cyrus SASL - documentation</td>
</tr>
<tr>
<td>msmtp</td>
<td>V:6, I:11</td>
<td>667</td>
<td>Light weight MTA</td>
</tr>
<tr>
<td>msmtp-mta</td>
<td>V:4, I:6</td>
<td>125</td>
<td>Light weight MTA (sendmail compatibility extension to msmtp)</td>
</tr>
<tr>
<td>esmtp</td>
<td>V:0, I:0</td>
<td>129</td>
<td>Light weight MTA</td>
</tr>
<tr>
<td>esmtp-run</td>
<td>V:0, I:0</td>
<td>32</td>
<td>Light weight MTA (sendmail compatibility extension to esmtp)</td>
</tr>
<tr>
<td>nullmailer</td>
<td>V:8, I:9</td>
<td>476</td>
<td>Strip down MTA, no local mail</td>
</tr>
<tr>
<td>ssmtp</td>
<td>V:5, I:8</td>
<td>2</td>
<td>Strip down MTA, no local mail</td>
</tr>
<tr>
<td>sendmail-bin</td>
<td>V:13, I:14</td>
<td>1877</td>
<td>Full featured MTA (only if you are already familiar)</td>
</tr>
<tr>
<td>courier-mta</td>
<td>V:0, I:0</td>
<td>2390</td>
<td>Full featured MTA (web interface etc.)</td>
</tr>
</tbody>
</table>

Table 6.3: List of basic mail transport agent related packages

#### 6.2.4.1 The configuration of exim4

For the Internet mail via smarthost, you (re)configure `exim4-*` packages as the following.

```bash
$ sudo systemctl stop exim4
$ 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 smart host:" to "smtp.hostname.dom:587".
Select "No" for "Hide local mail name in outgoing mail?". (Use "/etc/email-addresses" as in Section 6.2.4.3, 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 smart host by editing "/etc/exim4/passwd.client".

$ sudo vim /etc/exim4/passwd.client
...
$ cat /etc/exim4/passwd.client
^smtp.*\..hostname\.dom:username@hostname\.dom:password

Configure exim4(8) with "QUEUERUNNER='queueonly';" , "QUEUERUNNER='nodaemon'", etc. in "/etc/default/exim4" to minimize system resource usages. (optional)
Start exim4 by the following.

$ sudo systemctl start exim4

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.template" (non-split configuration)
  - creating new files or editing existing files in the "/etc/exim4/exim4.conf.d" subdirectories. (split configuration)
- Run "systemctl reload exim4".

**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.

Please read the official guide at: "/usr/share/doc/exim4-base/README.Debian.gz" and update-exim4.conf(8).

**Warning**
For all practical consideration, use SMTP with STARTTLS on port 587 or SMTPS (SMTP over SSL) on port 465, instead of plain SMTP on port 25.
6.2.4.2 The configuration of postfix with SASL

For the Internet mail via smarthost, you should first read postfix documentation and key manual pages.

You (re)configure postfix and sasl2-bin packages as follows.

```bash
$ sudo systemctl stop postfix
$ 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.

```bash
$ 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.

```bash
$ cat /etc/postfix/sasl_passwd
[smtp.hostname.dom]:587 username:password
$ sudo postmap hush:/etc/postfix/sasl_passwd
```

Start the postfix by the following.

```bash
$ sudo systemctl start postfix
```

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".

6.2.4.3 The mail address configuration

There are a few mail address configuration files for mail transport, delivery and user agents.

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.
Table 6.5: List of mail address related configuration files

<table>
<thead>
<tr>
<th>file</th>
<th>function</th>
<th>application</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/mailname</td>
<td>default host name for (outgoing) mail</td>
<td>Debian specific, mailname(5)</td>
</tr>
<tr>
<td>/etc/email-addresses</td>
<td>host name spoofing for outgoing mail</td>
<td>exim(8) specific, exim4-config_files(5)</td>
</tr>
<tr>
<td>/etc/postfix/generic</td>
<td>host name spoofing for outgoing mail</td>
<td>postfix(1) specific, activated after postmap(1) command execution.</td>
</tr>
<tr>
<td>/etc/aliases</td>
<td>account name alias for incoming mail</td>
<td>general, activated after newaliases(1) command execution.</td>
</tr>
</tbody>
</table>

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.

6.2.4.4 Basic MTA operations

There are several basic MTA operations. Some may be performed via sendmail(1) compatibility interface.

Tip
It may be a good idea to flush all mails by a script in "/etc/ppp/ip-up.d/*".

```
<table>
<thead>
<tr>
<th>exim command</th>
<th>postfix command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sendmail</td>
<td>sendmail</td>
<td>read mails from standard input and arrange for delivery (-bm)</td>
</tr>
<tr>
<td>mailq</td>
<td>mailq</td>
<td>list the mail queue with status and queue ID (-bp)</td>
</tr>
<tr>
<td>newaliases</td>
<td>newaliases</td>
<td>initialize alias database (-I)</td>
</tr>
<tr>
<td>exim4 -q</td>
<td>postqueue -f</td>
<td>flush waiting mails (-q)</td>
</tr>
<tr>
<td>exim4 -qf</td>
<td>postsuper -f ALL deferred; postqueue -f</td>
<td>flush all mails</td>
</tr>
<tr>
<td>exim4 -qff</td>
<td>postsuper -f ALL; postqueue -f</td>
<td>flush even frozen mails</td>
</tr>
<tr>
<td>exim4 -Mg queue_id</td>
<td>postsuper -h queue_id</td>
<td>freeze one message by its queue ID</td>
</tr>
<tr>
<td>exim4 -Mrm queue_id</td>
<td>postsuper -d queue_id</td>
<td>remove one message by its queue ID</td>
</tr>
<tr>
<td>N/A</td>
<td>postsuper -d ALL</td>
<td>remove all messages</td>
</tr>
</tbody>
</table>

Table 6.6: List of basic MTA operation

### 6.3 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.

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-`).

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>tool</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openssh-client</td>
<td>V:854, I:995</td>
<td>5814</td>
<td>ssh(1)</td>
<td>Secure shell client</td>
</tr>
<tr>
<td>openssh-server</td>
<td>V:721, I:813</td>
<td>1948</td>
<td>sshd(8)</td>
<td>Secure shell server</td>
</tr>
<tr>
<td>ssh-askpass</td>
<td>I:24</td>
<td>102</td>
<td>ssh-askpass</td>
<td>asks user for a pass phrase for ssh-add (plain X)</td>
</tr>
<tr>
<td>ssh-askpass-gnome</td>
<td>V:0, I:3</td>
<td>209</td>
<td>ssh-askpass-gnome(1)</td>
<td>asks user for a pass phrase for ssh-add (GNOME)</td>
</tr>
<tr>
<td>ssh-askpass-fullscreen</td>
<td>V:0, I:0</td>
<td>48</td>
<td>ssh-askpass-fullscreen(1)</td>
<td>asks user for a pass phrase for ssh-add (GNOME) with extra eye candy</td>
</tr>
<tr>
<td>shellinabox</td>
<td>V:0, I:1</td>
<td>507</td>
<td>shellinabox(1)</td>
<td>web server for browser accessible VT100 terminal emulator</td>
</tr>
</tbody>
</table>

Table 6.7: List of remote access server and utilities

Although shellinabox is not a SSH program, it is listed here as an interesting alternative for the remote terminal access. See also Section 7.8 for connecting to remote X client programs.

**Caution**

See Section 4.6.3 if your SSH is accessible from the Internet.

**Tip**

Please use the screen(1) program to enable remote shell process to survive the interrupted connection (see Section 9.1.2).
6.3.1 Basics of SSH

The OpenSSH SSH daemon supports SSH protocol 2 only. Please read "/usr/share/doc/openssh-client/README.Debian.gz", ssh(1), sshd(8), ssh-agent(1), and ssh-keygen(1), ssh-add(1) and ssh-agent(1).

---

**Warning**

"/etc/ssh/sshd_not_to_be_run" must not be present if one wishes to run the OpenSSH server. Don't enable host based authentication (HostbasedAuthentication in /etc/ssh/sshd_config).

---

<table>
<thead>
<tr>
<th>configuration file</th>
<th>description of configuration file</th>
</tr>
</thead>
<tbody>
<tr>
<td>/etc/ssh/ssh_config</td>
<td>SSH client defaults, see ssh_config(5)</td>
</tr>
<tr>
<td>/etc/ssh/sshd_config</td>
<td>SSH server defaults, see ssdh_config(5)</td>
</tr>
<tr>
<td>~/.ssh/authorized_keys</td>
<td>default public SSH keys that clients use to connect to this account on this SSH server</td>
</tr>
<tr>
<td>~/.ssh/id_rsa</td>
<td>secret SSH-2 RSA key of the user</td>
</tr>
<tr>
<td>~/.ssh/id_key-type-name</td>
<td>secret SSH-2 key-type-name key such as ed25519, ... of the user</td>
</tr>
</tbody>
</table>

Table 6.8: List of SSH configuration files

The following starts an ssh(1) connection from a client.

<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ssh <a href="mailto:username@hostname.domain.ext">username@hostname.domain.ext</a></td>
<td>connect with default mode</td>
</tr>
<tr>
<td>ssh -v <a href="mailto:username@hostname.domain.ext">username@hostname.domain.ext</a></td>
<td>connect with default mode with debugging messages</td>
</tr>
<tr>
<td>ssh -o PreferredAuthentications=password <a href="mailto:username@hostname.domain.ext">username@hostname.domain.ext</a></td>
<td>force to use password with SSH version 2</td>
</tr>
<tr>
<td>ssh -t <a href="mailto:username@hostname.domain.ext">username@hostname.domain.ext</a> passwd</td>
<td>run passwd program to update password on a remote host</td>
</tr>
</tbody>
</table>

Table 6.9: List of SSH client startup examples

6.3.2 User name on the remote host

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
```

6.3.3 Connecting without remote passwords

One can avoid having to remember passwords for remote systems by using "PubkeyAuthentication" (SSH-2 protocol).
On the remote system, set the respective entries, "PubkeyAuthentication yes", in "/etc/ssh/sshd_config". Generate authentication keys locally and install the public key on the remote system by the following.

```
$ ssh-keygen -t rsa
$ cat .ssh/id_rsa.pub | ssh user1@remote "cat - >>.ssh/authorized_keys"
```

You can add options to the entries in "~/.ssh/authorized_keys" to limit hosts and to run specific commands. See sshd(8) "AUTHORIZED_KEYS FILE FORMAT".

### 6.3.4 Dealing with alien SSH clients

There are some free SSH clients available for other platforms.

<table>
<thead>
<tr>
<th>environment</th>
<th>free SSH program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mac OS X</td>
<td>OpenSSH; use ssh in the Terminal application (GPL)</td>
</tr>
</tbody>
</table>

| **Table 6.10:** List of free SSH clients for other platforms |

### 6.3.5 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).

### 6.3.6 Sending a mail from a remote host

If you have an SSH shell account on a server with proper DNS settings, you can send a mail generated on your workstation as an email genuinely sent from the remote server.

```
$ ssh username@example.org /usr/sbin/sendmail -bm -ti -f "username@example.org" < mail_data ← .txt
```
6.3.7 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.

6.3.8 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 9.4.13) by the following.

```
# echo "shutdown -h now" | at now
```

Running "shutdown -h now" in `screen(1)` (see Section 9.1.2) session is another way to do the same.

6.3.9 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.

6.4 The print server and utilities

In the old Unix-like system, the BSD Line printer daemon (lpd) was the standard and the standard print out format of the classic free software was PostScript (PS). Some filter system was used along with Ghostscript to enable printing to the non-PostScript printer. See Section 11.4.1.

In the modern Debian system, the Common UNIX Printing System (CUPS) is the de facto standard and the standard print out format of the modern free software is Portable Document Format (PDF).

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.

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.

**Tip**

You can configure CUPS system by pointing your web browser to "http://localhost:631/".
### Table 6.11: List of print servers and utilities

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>port</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lpr</td>
<td>V:2,I:3</td>
<td>367</td>
<td>printer (515)</td>
<td>BSD lpr/lpd (Line printer daemon)</td>
</tr>
<tr>
<td>lprng</td>
<td>V:0,I:0</td>
<td>3051</td>
<td>..</td>
<td>.. (Enhanced)</td>
</tr>
<tr>
<td>cups</td>
<td>V:92,I:437</td>
<td>1061</td>
<td>IPP (631)</td>
<td>Internet Printing CUPS server</td>
</tr>
<tr>
<td>cups-client</td>
<td>V:112,I:460</td>
<td>426</td>
<td>..</td>
<td>System V printer commands for CUPS: \lp(1), \lpstat(1), \lpoptions(1), cancel(1), \lpmove(8), \lpinfo(8), \lpadmin(8), ..</td>
</tr>
<tr>
<td>cups-bsd</td>
<td>V:29,I:226</td>
<td>131</td>
<td>..</td>
<td>BSD printer commands for CUPS: lpr(1), lpq(1), lprm(1), lpc(8)</td>
</tr>
<tr>
<td>printer-driver-gutenprint</td>
<td>V:548,I:13</td>
<td>1219</td>
<td>Not applicable</td>
<td>printer drivers for CUPS</td>
</tr>
</tbody>
</table>

### Table 6.12: List of other network application servers

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>protocol</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>telnetd</td>
<td>V:0,I:2</td>
<td>53</td>
<td>TELNET</td>
<td>TELNET server</td>
</tr>
<tr>
<td>telnetd-ssl</td>
<td>V:0,I:0</td>
<td>159</td>
<td>..</td>
<td>.. (SSL support)</td>
</tr>
<tr>
<td>nfs-kernel-server</td>
<td>V:48,I:64</td>
<td>677</td>
<td>NFS</td>
<td>Unix file sharing</td>
</tr>
<tr>
<td>samba</td>
<td>V:106,I:133</td>
<td>4039</td>
<td>SMB</td>
<td>Windows file and printer sharing</td>
</tr>
<tr>
<td>netatalk</td>
<td>V:1,I:2</td>
<td>2003</td>
<td>ATP</td>
<td>Apple/Mac file and printer sharing (AppleTalk)</td>
</tr>
<tr>
<td>proftpd-basic</td>
<td>V:9,I:17</td>
<td>452</td>
<td>FTP</td>
<td>General file download</td>
</tr>
<tr>
<td>apache2</td>
<td>V:218,I:269</td>
<td>566</td>
<td>HTTP</td>
<td>General web server</td>
</tr>
<tr>
<td>squid</td>
<td>V:11,I:12</td>
<td>9166</td>
<td>..</td>
<td>General web proxy server</td>
</tr>
<tr>
<td>bind9</td>
<td>V:44,I:50</td>
<td>1113</td>
<td>DNS</td>
<td>IP address for other hosts</td>
</tr>
<tr>
<td>isc-dhcp-server</td>
<td>V:19,I:38</td>
<td>6082</td>
<td>DHCP</td>
<td>IP address of client itself</td>
</tr>
</tbody>
</table>
6.5 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.

**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.

6.6 Other network application clients

Here are other network application clients.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>protocol</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netcat</td>
<td>I:30</td>
<td>16</td>
<td>TCP/IP</td>
<td>TCP/IP swiss army knife</td>
</tr>
<tr>
<td>openssl</td>
<td>V:823, I:995</td>
<td>2290</td>
<td>SSL</td>
<td>Secure Socket Layer (SSL) binary and related cryptographic tools</td>
</tr>
<tr>
<td>stunnel4</td>
<td>V:7, I:12</td>
<td>541</td>
<td>,,</td>
<td>universal SSL Wrapper</td>
</tr>
<tr>
<td>telnet</td>
<td>V:34, I:598</td>
<td>53</td>
<td>TELNET</td>
<td>TELNET client</td>
</tr>
<tr>
<td>telnet-ssl</td>
<td>V:0, I:12</td>
<td>196</td>
<td>,,</td>
<td>,, (SSL support)</td>
</tr>
<tr>
<td>nfs-common</td>
<td>V:155, I:239</td>
<td>1109</td>
<td>NFS</td>
<td>Unix file sharing</td>
</tr>
<tr>
<td>smbclient</td>
<td>V:20, I:201</td>
<td>2074</td>
<td>SMB</td>
<td>MS Windows file and printer sharing client</td>
</tr>
<tr>
<td>cifs-utils</td>
<td>V:30, I:118</td>
<td>317</td>
<td>,,</td>
<td>mount and umount commands for remote MS Windows file</td>
</tr>
<tr>
<td>lftp</td>
<td>V:4, I:31</td>
<td>2361</td>
<td>,,</td>
<td>,,</td>
</tr>
<tr>
<td>ncftp</td>
<td>V:1, I:15</td>
<td>1399</td>
<td>,,</td>
<td>Full screen FTP client</td>
</tr>
<tr>
<td>wget</td>
<td>V:224, I:981</td>
<td>3681</td>
<td>HTTP and FTP</td>
<td>web downloader</td>
</tr>
<tr>
<td>curl</td>
<td>V:165, I:607</td>
<td>505</td>
<td>,,</td>
<td>,,</td>
</tr>
<tr>
<td>axel</td>
<td>V:0, I:13</td>
<td>201</td>
<td>,,</td>
<td>accelerated downloader</td>
</tr>
<tr>
<td>aria2</td>
<td>V:2, I:19</td>
<td>1852</td>
<td>,,</td>
<td>accelerated downloaded with BitTorrent and Metalink supports</td>
</tr>
<tr>
<td>bind9-host</td>
<td>V:129, I:940</td>
<td>385</td>
<td>DNS</td>
<td>host(i) from bind9, &quot;Priority: standard&quot;</td>
</tr>
<tr>
<td>dnsutils</td>
<td>V:19, I:299</td>
<td>272</td>
<td>,,</td>
<td>dig(i) from bind, &quot;Priority: standard&quot;</td>
</tr>
<tr>
<td>isc-dhcp-client</td>
<td>V:215, I:982</td>
<td>2866</td>
<td>DHCP</td>
<td>obtain IP address</td>
</tr>
<tr>
<td>ldap-utils</td>
<td>V:13, I:65</td>
<td>762</td>
<td>LDAP</td>
<td>obtain data from LDAP server</td>
</tr>
</tbody>
</table>

Table 6.13: List of network application clients
6.7 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.

<table>
<thead>
<tr>
<th>RFC</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rfc1939 and rfc2449</td>
<td>POP3 service</td>
</tr>
<tr>
<td>rfc3501</td>
<td>IMAP4 service</td>
</tr>
<tr>
<td>rfc2821 (rfc821)</td>
<td>SMTP service</td>
</tr>
<tr>
<td>rfc2822 (rfc822)</td>
<td>Mail file format</td>
</tr>
<tr>
<td>rfc2045</td>
<td>Multipurpose Internet Mail Extensions (MIME)</td>
</tr>
<tr>
<td>rfc819</td>
<td>DNS service</td>
</tr>
<tr>
<td>rfc2616</td>
<td>HTTP service</td>
</tr>
<tr>
<td>rfc2396</td>
<td>URI definition</td>
</tr>
</tbody>
</table>

Table 6.14: List of popular RFCs

The port usage is described in `"/etc/services"`. 
Chapter 7

GUI System

7.1 GUI desktop environment

There are several choices for the full featured GUI desktop environment on the Debian system.

<table>
<thead>
<tr>
<th>task package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>task-gnome-desktop</td>
<td>1:199</td>
<td>9</td>
<td>GNOME desktop environment</td>
</tr>
<tr>
<td>task-xfce-desktop</td>
<td>1:97</td>
<td>9</td>
<td>Xfce desktop environment</td>
</tr>
<tr>
<td>task-kde-desktop</td>
<td>1:78</td>
<td>6</td>
<td>KDE Plasma desktop environment</td>
</tr>
<tr>
<td>task-mate-desktop</td>
<td>1:44</td>
<td>9</td>
<td>MATE desktop environment</td>
</tr>
<tr>
<td>task-cinnamon-desktop</td>
<td>1:41</td>
<td>9</td>
<td>Cinnamon desktop environment</td>
</tr>
<tr>
<td>task-lxde-desktop</td>
<td>1:30</td>
<td>9</td>
<td>LXDE desktop environment</td>
</tr>
<tr>
<td>task-lxqt-desktop</td>
<td>1:18</td>
<td>9</td>
<td>LXQt desktop environment</td>
</tr>
<tr>
<td>task-gnome-flashback-desktop</td>
<td>1:13</td>
<td>6</td>
<td>GNOME Flashback desktop environment</td>
</tr>
</tbody>
</table>

Table 7.1: List of desktop environment
Tip
Dependency packages selected by a task metapackage may be out of sync with the latest package transition state under the Debian unstable/testing environment. For task-gnome-desktop, you may need to adjust package selections as follows:

- Start aptitude(8) as sudo `aptitude -u`.
- Move cursor to "Tasks" and press "Enter".
- Move cursor to "End-user" press "Enter".
- Move cursor to "GNOME" press "Enter".
- Move cursor to task-gnome-desktop and press "Enter".
- Move cursor to "Depends" and press "m" (manually selected).
- Move cursor to "Recommends" and press "m" (manually selected).
- Move cursor to "task-gnome-desktop and press ".-" (drop)
- Adjust selected packages while dropping problematic ones causing package conflicts.
- Press "g" to start install.

This chapter will focus mostly on the default desktop environment of Debian: task-gnome-desktop offering GNOME on wayland.

7.2 GUI communication protocol

GUI communication protocol used on the GNOME desktop can be:

- **Wayland (display server protocol) (native)**
- **X Window System core protocol (via xwayland)**

Please check freedesktop.org site for how Wayland architecture is different from X Window architecture.

From user’s perspective, differences can be colloquially summarized as:

- Wayland is a same-host GUI communication protocol: new, simpler, faster, no setuid root binary
- X Window is a network-capable GUI communication protocol: traditional, complex, slower, setuid root binary

For applications using Wayland protocol, the access to their display contents from a remote host is supported by the VNC or RDP. See Section 7.7

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. This situation was the background of creating Wayland as a local-only GUI communication protocol.

Using the `xeyes` program started from the GNOME terminal, you can check GUI communication protocol used by each GUI application.

```sh
dxeyes
```

- If the mouse cursor is on an application such as "GNOME terminal" which uses Wayland display server protocol, eyes don’t move with the mouse cursor.
If the mouse cursor is on an application such as "xterm" which uses X Window System core protocol, eyes move with the mouse cursor exposing not-so-isolated nature of X Window architecture.

As of April 2021, many popular GUI applications such as GNOME and LibreOffice (LO) applications have been migrated to the Wayland display server protocol. I see xterm, gitk, chromium, firefox, gimp, dia, and KDE applications still use X Window System core protocol.

Note
For both the xwayland on Wayland or the native X Window System, the old X server configuration file "/etc/X11/xorg.conf" shouldn't exist on the system. The graphics and input devices are now configured by the kernel with DRM, KMS, and udev. The native X server has been rewritten to use them. See "modedb default video mode support" in the Linux kernel documentation.

7.3 GUI infrastructure

Here are notable GUI infrastructure packages for the GNOME on Wayland environment.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>package size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mutter</td>
<td>V:2, I:71</td>
<td>186</td>
<td>GNOME’s mutter window manager [auto]</td>
</tr>
<tr>
<td>xwayland</td>
<td>V:226, I:309</td>
<td>2387</td>
<td>An X server running on top of wayland [auto]</td>
</tr>
<tr>
<td>gnome-remote-desktop</td>
<td>V:20, I:227</td>
<td>1200</td>
<td>Remote desktop daemon for GNOME using PipeWire [auto]</td>
</tr>
<tr>
<td>gnome-tweaks</td>
<td>V:14, I:205</td>
<td>60</td>
<td>Advanced configuration settings for GNOME</td>
</tr>
</tbody>
</table>

Here, "[auto]" means that these packages are automatically installed when task-gnome-desktop is installed.

Tip
gnome-tweaks is the indispensable configuration utility. For example:

- You can force "Over-Amplification" of sound volume from "General".
- You can force "Caps" to become "Esc" from "Keyboard & Mouse" -> "Keyboard" -> "Additional Layout Option".

Tip
Detail features of GNOME desktop environment can be configured with utilities started by typing "settings", "tweaks", or "extensions" after pressing Super-key.

7.4 GUI applications

Many useful GUI applications are available on Debian now. Installing software packages 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 may clutter your system.

Here is a list of GUI applications which caught my eyes.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>package size</th>
<th>type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>evolution</td>
<td>V:29, I:240</td>
<td>483</td>
<td>GNOME</td>
<td>Personal information Management (groupware and email)</td>
</tr>
<tr>
<td>thunderbird</td>
<td>V:50, I:120</td>
<td>224509</td>
<td>GTK</td>
<td>Email client (Mozilla Thunderbird)</td>
</tr>
<tr>
<td>kontakt</td>
<td>V:1, I:12</td>
<td>2208</td>
<td>KDE</td>
<td>Personal information Management (groupware and email)</td>
</tr>
<tr>
<td>libreoffice-writer</td>
<td>V:121, I:436</td>
<td>39522</td>
<td>LO</td>
<td>word processor</td>
</tr>
<tr>
<td>abiword</td>
<td>V:1, I:8</td>
<td>3467</td>
<td>GNOME</td>
<td>word processor</td>
</tr>
<tr>
<td>calligrawords</td>
<td>V:0, I:7</td>
<td>6097</td>
<td>GTK</td>
<td>Email client (Mozilla Thunderbird)</td>
</tr>
<tr>
<td>scribus</td>
<td>V:1, I:17</td>
<td>30242</td>
<td>KDE</td>
<td>desktop publishing editor to edit PDF files</td>
</tr>
<tr>
<td>glabels</td>
<td>V:0, I:3</td>
<td>1338</td>
<td>GNOME</td>
<td>label editor</td>
</tr>
<tr>
<td>libreoffice-calc</td>
<td>V:114, I:433</td>
<td>31154</td>
<td>LO</td>
<td>spreadsheet</td>
</tr>
<tr>
<td>gnumeric</td>
<td>V:4, I:15</td>
<td>10004</td>
<td>GNOME</td>
<td>spreadsheet</td>
</tr>
<tr>
<td>calligrasheets</td>
<td>V:0, I:5</td>
<td>11525</td>
<td>KDE</td>
<td>spreadsheet</td>
</tr>
<tr>
<td>libreoffice-impress</td>
<td>V:77, I:430</td>
<td>9057</td>
<td>LO</td>
<td>presentation</td>
</tr>
<tr>
<td>calligrastage</td>
<td>V:0, I:5</td>
<td>5339</td>
<td>KDE</td>
<td>presentation</td>
</tr>
<tr>
<td>libreoffice-base</td>
<td>V:31, I:133</td>
<td>6010</td>
<td>LO</td>
<td>database management</td>
</tr>
<tr>
<td>kexi</td>
<td>V:0, I:1</td>
<td>7118</td>
<td>KDE</td>
<td>database management</td>
</tr>
<tr>
<td>libreoffice-draw</td>
<td>V:78, I:430</td>
<td>13625</td>
<td>LO</td>
<td>vector graphics editor (draw)</td>
</tr>
<tr>
<td>inkscape</td>
<td>V:16, I:121</td>
<td>99316</td>
<td>GNOME</td>
<td>vector graphics editor (draw)</td>
</tr>
<tr>
<td>karbon</td>
<td>V:0, I:6</td>
<td>3610</td>
<td>KDE</td>
<td>vector graphics editor (draw)</td>
</tr>
<tr>
<td>dia</td>
<td>V:3, I:24</td>
<td>3908</td>
<td>GTK</td>
<td>flowchart and diagram editor</td>
</tr>
<tr>
<td>gimp</td>
<td>V:53, I:258</td>
<td>19877</td>
<td>GTK</td>
<td>bitmap graphics editor (paint)</td>
</tr>
<tr>
<td>shotwell</td>
<td>V:17, I:255</td>
<td>6187</td>
<td>GTK</td>
<td>digital photo organizer</td>
</tr>
<tr>
<td>digikam</td>
<td>V:1, I:9</td>
<td>293</td>
<td>KDE</td>
<td>digital photo organizer</td>
</tr>
<tr>
<td>darktable</td>
<td>V:4, I:13</td>
<td>30572</td>
<td>GTK</td>
<td>lighttable and darkroom for photographers</td>
</tr>
<tr>
<td>planner</td>
<td>V:0, I:4</td>
<td>1394</td>
<td>GNOME</td>
<td>project management</td>
</tr>
<tr>
<td>calligraphan</td>
<td>V:0, I:2</td>
<td>19013</td>
<td>KDE</td>
<td>project management</td>
</tr>
<tr>
<td>gnuclash</td>
<td>V:2, I:8</td>
<td>28051</td>
<td>GNOME</td>
<td>personal accounting</td>
</tr>
<tr>
<td>homebank</td>
<td>V:0, I:2</td>
<td>1209</td>
<td>GTK</td>
<td>personal accounting</td>
</tr>
<tr>
<td>lilypond</td>
<td>V:0, I:7</td>
<td>15747</td>
<td>-</td>
<td>music typesetter</td>
</tr>
<tr>
<td>kmymoney</td>
<td>V:0, I:2</td>
<td>13341</td>
<td>KDE</td>
<td>personal accounting</td>
</tr>
<tr>
<td>libreoffice-math</td>
<td>V:57, I:434</td>
<td>2267</td>
<td>LO</td>
<td>mathematical equation/formula editor</td>
</tr>
<tr>
<td>calibre</td>
<td>V:7, I:28</td>
<td>61995</td>
<td>KDE</td>
<td>e-book converter and library management</td>
</tr>
<tr>
<td>fbreader</td>
<td>V:1, I:9</td>
<td>3783</td>
<td>GTK</td>
<td>e-book reader</td>
</tr>
<tr>
<td>evince</td>
<td>V:93, I:318</td>
<td>941</td>
<td>GNOME</td>
<td>document(pdf) viewer</td>
</tr>
<tr>
<td>okular</td>
<td>V:39, I:120</td>
<td>17728</td>
<td>KDE</td>
<td>document(pdf) viewer</td>
</tr>
<tr>
<td>x11-apps</td>
<td>V:30, I:463</td>
<td>2476</td>
<td>pure X-app</td>
<td>xeyes(1), etc.</td>
</tr>
<tr>
<td>x11-utils</td>
<td>V:193, I:569</td>
<td>712</td>
<td>pure X-app</td>
<td>xev(1), xwininfo(1), etc.</td>
</tr>
</tbody>
</table>

Table 7.3: List of notable GUI applications
7.5 Fonts

Many useful scalable fonts are available for users on Debian. User’s concern is how to avoid redundancy and how to configure parts of installed fonts to be disabled. Otherwise, useless font choices may clutter your GUI application menus.

Debian system uses FreeType 2.0 library to rasterise many scalable font formats for screen and print:

- **Type 1 (PostScript) fonts** which use cubic Bézier curves (almost obsolete format)
- **TrueType fonts** which use quadratic Bézier curves (good choice format)
- **OpenType fonts** which use cubic Bézier curves (best choice format)

### 7.5.1 Basic fonts

The following table is compiled in the hope to help users to chose appropriate scalable fonts with clear understanding of the metric compatibility and the glyph coverage. Most fonts cover all Latin, Greek, and Cyril characters. The final choice of activated fonts can also be affected by your aesthetics. These fonts can be used for the screen display or for the paper printing.

Here:

- "MCM" stands for "metric compatible with fonts provided by Microsoft"
- "MCMATC" stands for "metric compatible with fonts provided by Microsoft: Arial, Times New Roman, Courier New"
- "MCAHTC" stands for "metric compatible with fonts provided by Adobe: Helvetica, Times, Courier"
- Numbers in font type columns stands for the rough relative "M" width for the same point size font.
- "P" in mono font type columns stands for its usability for programming having clearly distinguishable "0"/"O" and "1"/"I"/"l".

The ttf-mscorefonts-installer package downloads Microsoft’s “Core fonts for the Web” and installs Arial, Times New Roman, Courier New, Verdana, ... . These installed font data are non-free data.

Many free Latin fonts have their lineage traced to URW Nimbus family or Bitstream Vera.

---

**Tip**

If your locale needs fonts not covered well by the above fonts, please use aptitude to check under task packages listed under "Tasks" -> "Localization". The font packages listed as "Depends:" or "Recommends:" in the localization task packages are the primary candidates.

---

### 7.5.2 Font rasterization

Debian uses FreeType to rasterize fonts. Its font choice infrastructure is provided by the Fontconfig font configuration library.

**Tip**

Some font packages such as fonts-noto* install too many fonts. You may also want to keep some font packages installed but disabled under the normal use situation. The multiple glyphs are expected for some Unicode code points due to Han unification and unwanted glyphs may be chosen by the unconfigured Fontconfig library. One of the most annoying case is "U+3001 IDEOGRAPHIC COMMA" and "U+3002 IDEOGRAPHIC FULL STOP" among CJK countries. You can avoid this problematic situation easily by configuring font availability using Font Manager GUI (font-manager).

You can list font configuration state from the command line, too.

- "fc-match(1)" for fontconfig font default
- "fc-list(1)" for available fontconfig fonts

You can configure font configuration state from the text editor but this is non-trivial. See fonts.conf(5).
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>sans</th>
<th>serif</th>
<th>mono</th>
<th>note on font</th>
</tr>
</thead>
<tbody>
<tr>
<td>fonts-cantarell</td>
<td>V:213, I:308</td>
<td>572</td>
<td>59</td>
<td>-</td>
<td>-</td>
<td>Cantarell (GNOME 3, display)</td>
</tr>
<tr>
<td>fonts-noto</td>
<td>I:148</td>
<td>31</td>
<td>61</td>
<td>63</td>
<td>40</td>
<td>Noto fonts (Google, multi-lingual with CJK)</td>
</tr>
<tr>
<td>fonts-dejavu</td>
<td>I:428</td>
<td>35</td>
<td>58</td>
<td>68</td>
<td>40</td>
<td>DejaVu (GNOME 2, MCM: Verdana, extended Bitstream Vera)</td>
</tr>
<tr>
<td>fonts-liberation2</td>
<td>V:123, I:419</td>
<td>15</td>
<td>56</td>
<td>60</td>
<td>40</td>
<td>Liberation fonts for LibreOffice (Red Hat, MCMATC)</td>
</tr>
<tr>
<td>fonts-croscore</td>
<td>V:20, I:41</td>
<td>5274</td>
<td>56</td>
<td>60</td>
<td>40</td>
<td>Chrome OS: Arimo, Tinos and Cousine (Google, MCMATC)</td>
</tr>
<tr>
<td>fonts-crosextra-carlito</td>
<td>V:21, I:140</td>
<td>2696</td>
<td>57</td>
<td>-</td>
<td>-</td>
<td>Chrome OS: Carlito (Google, MCM: Calibri)</td>
</tr>
<tr>
<td>fonts-crosextra-caladea</td>
<td>I:138</td>
<td>347</td>
<td>-</td>
<td>55</td>
<td>-</td>
<td>Chrome OS: Caladea (Google, MCM: Cambria) (Latin only)</td>
</tr>
<tr>
<td>fonts-freefont-ttf</td>
<td>V:72, I:219</td>
<td>14460</td>
<td>57</td>
<td>59</td>
<td>40</td>
<td>GNU FreeFont (extended URW Nimbus)</td>
</tr>
<tr>
<td>fonts-quicksand</td>
<td>V:78, I:433</td>
<td>392</td>
<td>56</td>
<td>-</td>
<td>-</td>
<td>Debian task-desktop, Quicksand (display, Latin only)</td>
</tr>
<tr>
<td>fonts-hack</td>
<td>V:23, I:114</td>
<td>2508</td>
<td>-</td>
<td>-</td>
<td>40 P</td>
<td>A typeface designed for source code Hack (Facebook)</td>
</tr>
<tr>
<td>fonts-sil-gentiumplus</td>
<td>I:32</td>
<td>14345</td>
<td>-</td>
<td>54</td>
<td>-</td>
<td>Gentium SIL</td>
</tr>
<tr>
<td>fonts-sil-charis</td>
<td>I:26</td>
<td>6704</td>
<td>-</td>
<td>59</td>
<td>-</td>
<td>Charis SIL</td>
</tr>
<tr>
<td>fonts-urw-base35</td>
<td>V:158, I:452</td>
<td>15558</td>
<td>56</td>
<td>60</td>
<td>40</td>
<td>URW Nimbus (Sans, Roman No. 9 L, Mono L, MCAHTC)</td>
</tr>
<tr>
<td>fonts-ubuntu</td>
<td>V:2, I:5</td>
<td>4339</td>
<td>58</td>
<td>-</td>
<td>33 P</td>
<td>Ubuntu fonts (display)</td>
</tr>
<tr>
<td>fonts-terminus</td>
<td>V:0, I:3</td>
<td>453</td>
<td>-</td>
<td>-</td>
<td>33</td>
<td>Cool retro terminal fonts</td>
</tr>
<tr>
<td>ttf-mscorefonts-installer</td>
<td>V:1, I:51</td>
<td>85</td>
<td>567</td>
<td>60</td>
<td>40</td>
<td>Downloader of Microsoft non-free fonts (see below)</td>
</tr>
</tbody>
</table>

Table 7.4: List of notable TrueType and OpenType fonts

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libfreetype6</td>
<td>V:557, I:997</td>
<td>933</td>
<td>FreeType font rasterization library</td>
</tr>
<tr>
<td>libfontconfig1</td>
<td>V:553, I:851</td>
<td>580</td>
<td>Fontconfig font configuration library</td>
</tr>
<tr>
<td>fontconfig</td>
<td>V:440, I:723</td>
<td>679</td>
<td>Fc-*: CLI commands for Fontconfig</td>
</tr>
<tr>
<td>font-manager</td>
<td>V:2, I:8</td>
<td>1038</td>
<td>Font Manager: GUI command for Fontconfig</td>
</tr>
<tr>
<td>nautilus-font-manager</td>
<td>V:0, I:0</td>
<td>37</td>
<td>Nautilus extension for Font Manager</td>
</tr>
</tbody>
</table>

Table 7.5: List of notable font environment and related packages
7.6 Sandbox

Many mostly GUI applications on Linux are available in binary formats from non-Debian sources.

- **AppImage** -- Linux apps that run anywhere
- **FLATHUB** -- Apps for Linux, right here
- **snapcraft** -- The app store for Linux

---

**Warning**

Binaries from these sites may include proprietary non-free software packages.

---

There is some raison d’être for these binary format distributions for Free Software aficionados using Debian since these can accommodate clean set of libraries used for each application by the respective upstream developer independent of the ones provided by Debian.

The inherent risk of running external binaries can be reduced by using the sandbox environment which leverages modern Linux security features (see Section 4.7.4).

- For binaries from AppImage and some upstream sites, run them in `firejail` with manual configuration.
- For binaries from FLATHUB, run them in `Flatpak`. (No manual configuration required.)
- For binaries from snapcraft, run them in `Snap`. (No manual configuration required. Compatible with daemon programs.)

The `xdg-desktop-portal` package provides a standardized API to common desktop features. See `xdg-desktop-portal (flatpak)` and `xdg-desktop-portal (snap)`.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>flatpak</td>
<td>V:60, I:65</td>
<td>7372</td>
<td>Flatpak application deployment framework for desktop apps</td>
</tr>
<tr>
<td>gnome-software-plugin-flatpak</td>
<td>V:19, I:28</td>
<td>246</td>
<td>Flatpak support for GNOME Software</td>
</tr>
<tr>
<td>snapd</td>
<td>V:65, I:69</td>
<td>62774</td>
<td>Daemon and tooling that enable snap packages</td>
</tr>
<tr>
<td>gnome-software-plugin-snap</td>
<td>V:1, I:2</td>
<td>117</td>
<td>Snap support for GNOME Software</td>
</tr>
<tr>
<td>xdg-desktop-portal</td>
<td>V:288, I:380</td>
<td>1936</td>
<td>desktop integration portal for Flatpak and Snap</td>
</tr>
<tr>
<td>xdg-desktop-portal-kde</td>
<td>V:43, I:60</td>
<td>1432</td>
<td>xdg-desktop-portal backend for Qt (KDE)</td>
</tr>
<tr>
<td>xdg-desktop-portal-wlr</td>
<td>V:0, I:3</td>
<td>131</td>
<td>xdg-desktop-portal backend for wlroots (Wayland)</td>
</tr>
<tr>
<td>firejail</td>
<td>V:1, I:4</td>
<td>1771</td>
<td>a SUID security sandbox program firejail for use with AppImage</td>
</tr>
</tbody>
</table>

Table 7.6: List of notable sandbox environment and related packages

This sandbox environment technology is very much like apps on smart phone OS where apps are executed under controlled resource accesses.

Some large GUI applications such as web browsers on Debian also use sandbox environment technology internally to make them more secure.
### Table 7.7: List of notable remote access servers

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>protocols</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnome-remote-desktop</td>
<td>0.0:1.209</td>
<td>1063</td>
<td>RDP, RFB (VNC)</td>
<td>GNOME Remote Desktop server</td>
</tr>
<tr>
<td>vinagre</td>
<td>0.3:1.78</td>
<td>4249</td>
<td>RDP, RFB (VNC), SPICE, SSH</td>
<td>Vinagre: GNOME remote desktop client</td>
</tr>
<tr>
<td>remmina</td>
<td>0.14:1.71</td>
<td>915</td>
<td>RDP, RFB (VNC), SPICE, SSH</td>
<td>Remmina: GTK remote desktop client</td>
</tr>
<tr>
<td>krdc</td>
<td>0.1:1.17</td>
<td>3904</td>
<td>RDP, RFB (VNC)</td>
<td>KRDC: KDE remote desktop client</td>
</tr>
<tr>
<td>guacd</td>
<td>0.0:1.0</td>
<td>80</td>
<td>RDP, RFB (VNC), SSH / HTML5</td>
<td>Apache Guacamole: clientless remote desktop gateway (HTML5)</td>
</tr>
<tr>
<td>virt-viewer</td>
<td>0.5:1.52</td>
<td>1284</td>
<td>RFB (VNC), SPICE</td>
<td>Virtual Machine Manager’s GUI display client of guest OS</td>
</tr>
</tbody>
</table>

### 7.7 Remote desktop

### 7.8 X server connection

There are several ways to connect from an application on a remote host to the X server including `xwayland` on the local host.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openssh-server</td>
<td>0.721:1.813</td>
<td>1948</td>
<td>sshd with option X11-forwarding</td>
<td>SSH server (secure)</td>
</tr>
<tr>
<td>openssh-client</td>
<td>0.854:1.995</td>
<td>1814</td>
<td>ssh -X</td>
<td>SSH client (secure)</td>
</tr>
<tr>
<td>xauth</td>
<td>0.163:1.959</td>
<td>81</td>
<td>xauth</td>
<td>X authority file utility</td>
</tr>
<tr>
<td>x11-xserver-utils</td>
<td>0.303:1.527</td>
<td>576</td>
<td>xhost</td>
<td>server access control for X</td>
</tr>
</tbody>
</table>

### Table 7.8: List of connection methods to the X server

### 7.8.1 X server local connection

Access to the local X server by the local applications which use X core protocol can be locally connected through a local UNIX domain socket. This can be authorized by the authority file holding `access_cookie`. The authority file location is identified by the "$XAUTHORITY" environment variable and X display is identified by the "$DISPLAY" environment variable. Since these are normally set automatically, no special action is needed, e.g. "gitk" as the following.

```bash
username $ gitk
```

**Note**

For `xwayland`, `XAUTHORITY` holds value like "/run/user/1000/.mutter-Xwaylandauth.YVSU30".


7.8.2 X server remote connection

Access to the local X server display from the remote applications which use X core protocol is supported by using the X11 forwarding feature.

- Open an `gnome-terminal` on the local host.
- Run `ssh(1)` with `-X` option 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. "gitk", on the remote site as the following.

  ```
  loginname @ remotehost $ gitk
  ```

This method can display the output from a remote X client as if it were locally connected through a local UNIX domain socket. See Section 6.3 for SSH/SSHD.

⚠️ **Warning**
A remote TCP/IP connection to the X server is disabled by default on the Debian system for security reasons. Don’t enable them by simply setting "xhost +" nor by enabling XDMCP connection, if you can avoid it.

7.8.3 X server chroot connection

Access to the X server by the applications which use X core protocol and run on the same host but in an environment such as chroot where the authority file is not accessible, can be authorized securely with `xhost` by using the User-based access, e.g. "gitk" as the following.

```
username $ xhost + si:localuser:root ; sudo chroot /path/to
# cd /src
# gitk
# exit
username $ xhost -
```

7.9 Clipboard

For clipping text to clipboard, see Section 1.4.4.

For clipping graphics to clipboard, see Section 11.6.

Some CLI commands can manipulate character clipboard (PRIMARY and CLIPBOARD), too.
<table>
<thead>
<tr>
<th>package</th>
<th>popup</th>
<th>package size</th>
<th>target</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xset</td>
<td>V:9, I:41</td>
<td>55</td>
<td>X</td>
<td>command line interface to X selections (clipboard)</td>
</tr>
<tr>
<td>xclip</td>
<td>V:11, I:60</td>
<td>64</td>
<td>X</td>
<td>command line interface to X selections (clipboard)</td>
</tr>
<tr>
<td>wl-clipboard</td>
<td>V:2, I:12</td>
<td>162</td>
<td>Wayland</td>
<td><code>wl-copy</code> <code>wl-paste</code>: command line interface to Wayland clipboard</td>
</tr>
<tr>
<td>gpm</td>
<td>V:10, I:12</td>
<td>521</td>
<td>Linux console</td>
<td>a daemon that captures mouse events on Linux console</td>
</tr>
</tbody>
</table>

Table 7.9: List of programs related to manipulating character clipboard
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. See *Introduction to i18n* for details.

8.1 The locale

The behavior of programs supporting internationalization are configured by the environment variable "$LANG" to support localization. Actual support of locale dependent features by the libc library requires to install locales or locales-all packages. The locales package requires to be initialized properly.

If neither locales or locales-all package are installed, support of locale features are lost and system uses US English messages and handles data as ASCII. This behavior is the same way as "$LANG" is set by "$LANG=", "$LANG=C", or "$LANG=POSIX".

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.

The current Debian desktop GUI system normally sets the locale under GUI environment as "LANG=xx_YY.UTF-8". Here, "xx" is ISO 639 language codes and "YY" is ISO 3166 country codes. These values are set by the desktop configuration GUI dialogue and change the program behavior. See Section 1.5.2

8.1.1 Rationale for UTF-8 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).

Even plain English text may contain non-ASCII characters, e.g. slightly curly left and right quotation marks are not available in ASCII.

```
b"" "b""double quoted textb"" b"" is not "double quoted ASCII"
b'" 'b'"'single quoted textb'"' b'" is not 'single quoted ASCII'
```
In order to support more characters, many character sets and encoding systems have been used to support many languages (see Table 11.2).

Unicode character set can represent practically all characters known to human with 21 bit code point range (i.e., 0 to 10FFFF in hexadecimal notation).

Text encoding system UTF-8 fits Unicode code points into a sensible 8 bit data stream mostly compatible with the ASCII data processing system. This makes UTF-8 the modern preferred choice. UTF stands for Unicode Transformation Format. 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 lose nothing by deploying UTF-8 locale.

Under UTF-8 locale with the compatible application program, you can display and edit any foreign language text data as long as required fonts and input methods are installed and enabled. For example under "LANG=fr_FR.UTF-8" locale, gedit(1) (text editor for the GNOME desktop) can display and edit Chinese character text data while presenting menus in French.

**Tip**
Both the new standard "en_US.UTF-8" locale and the old standard "C"/"POSIX" locale use the standard US English message, they have subtle differences in sorting order etc. If you want to handle not only ASCII characters but also handle all UTF-8 encoded characters gracefully while maintaining the old "C" local behavior, use the non-standard "C=UTF-8" locale on Debian.

**Note**
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 lose nothing by deploying UTF-8 locale.

### 8.1.2 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 **locales** package does **not** come with pre-compiled locale data. You need to configure it as:

```
# dpkg-reconfigure locales
```

This process involves 2 steps.

1. Select all required locale data to be compiled into the binary form. (Please make sure to include at least one UTF-8 locale)
2. Set the system wide default locale value by creating "/etc/default/locale" for use by PAM (see Section 4.5).

The system wide default locale value set in "/etc/default/locale" may be overridden by the GUI configuration for GUI applications.

**Note**
Actual traditional encoding system can be identified by "/usr/share/i18n/SUPPORTED". Thus, the "LANG=en_US" is "LANG=en_US.ISO-8859-1".

The **locales-all** package comes with pre-compiled locale data for all locale data. Since it doesn't create "/etc/default/locale" file, you may still need to install the **locales** package, too.

**Tip**
The **locales** package of some Debian derivative distributions come with pre-compiled locale data for all locale data. You need to install both **locales** and **locales-all** packages on Debian to emulate such system environment.
8.1.3 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.

8.1.4 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.

```bash
LANG=it_IT.UTF-8 man programname
```

GNU gettext can accommodate priority list of translation languages with $LANGUAGE environment variable. For example:

```bash
$ export LANGUAGE="pt:pt_BR:es:it:fr"
```

For more, see info gettext and read the section "The LANGUAGE variable".

8.1.5 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 9.3.4).

Number punctuation are different for locales. For example, in English locale, one thousand 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.

Each detail feature of "$LANG" environment variable may be overridden by setting "$LC_*" variables. These environment variables can be overridden again by setting "$LC_ALL" variable. See locale(7) manpage for the details. Unless you have strong reason to create complicated configuration, please stay away from them and use only "$LANG" variable set to one of the UTF-8 locales.
8.2 The keyboard input

8.2.1 The keyboard input for Linux console and X Window

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
```

For the Linux console and the X Window system, this 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`.

8.2.2 The keyboard input for Wayland

For GNOME on Wayland desktop system, Section 8.2.1 can’t support non-English European languages. `IBus` was made to support not only Asian languages but also European languages. The package dependency of GNOME desktop Environment recommends "ibus" via "gnome-shell". The code of "ibus" has been updated to integrate `setxkbmap` and XKB option functionalities. You need to configure `ibus` from "GNOME Settings" or "GNOME Tweaks" for the multilingualized keyboard input.

---

**Note**

If `ibus` is active, your classic X keyboard configuration by the `setxkbmap` may be overridden by `ibus` even under classic X-based desktop environment. You can disable installed `ibus` using `im-config` to set input method to "None". For more, see Debian Wiki on keyboard.

---

8.2.3 The input method support with IBus

Since GNOME desktop Environment recommends "ibus" via "gnome-shell", "ibus" is the good choice for input method. Multilingual input to the application is processed as:

```
Keyboard                    Application
|                           |
|+-> Linux kernel -|> Input method (ibus) -+-> Gtk, Qt, X, Wayland
|   +-> Engine --+
```

The list of IBus and its engine packages are the following.

---

**Note**

For Chinese, "fcitx5" may be an alternative input method framework. For Emacs aficionados, "uim" may be an alternative. Either cases, you may need to do extra manual configuration with `im-config`. Some old classic input methods such as "kinput2" may still exist in Debian repository but are not recommended for the modern environment.

---

8.2.4 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 GNOME on Wayland:

1. Install the Japanese input tool package `ibus-mozc` (or `ibus-anthy`) with its recommended packages such as `im-config`. 

---
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>supported locale</th>
</tr>
</thead>
<tbody>
<tr>
<td>ibus</td>
<td>V:192, I:240</td>
<td>1710</td>
<td>input method framework using dbus</td>
</tr>
<tr>
<td>ibus-mozc</td>
<td>V:1, I:3</td>
<td>935</td>
<td>Japanese</td>
</tr>
<tr>
<td>ibus-anthy</td>
<td>V:0, I:1</td>
<td>8856</td>
<td></td>
</tr>
<tr>
<td>ibus-skk</td>
<td>V:0, I:0</td>
<td>242</td>
<td></td>
</tr>
<tr>
<td>ibus-kkc</td>
<td>V:0, I:0</td>
<td>210</td>
<td></td>
</tr>
<tr>
<td>ibus-libpinyin</td>
<td>V:1, I:3</td>
<td>2760</td>
<td>Chinese (for zh_CN)</td>
</tr>
<tr>
<td>ibus-chewing</td>
<td>V:0, I:0</td>
<td>422</td>
<td>(for zh_TW)</td>
</tr>
<tr>
<td>ibus-libzhuuyin</td>
<td>V:0, I:0</td>
<td>40987</td>
<td>(for zh_TW)</td>
</tr>
<tr>
<td>ibus-rime</td>
<td>V:0, I:0</td>
<td>73</td>
<td>(for zh_CN/zh_TW)</td>
</tr>
<tr>
<td>ibus-cangjie</td>
<td>V:0, I:0</td>
<td>119</td>
<td>(for zh_HK)</td>
</tr>
<tr>
<td>ibus-hangul</td>
<td>V:0, I:2</td>
<td>264</td>
<td>Korean</td>
</tr>
<tr>
<td>ibus-libthai</td>
<td>I:0</td>
<td>90</td>
<td>Thai</td>
</tr>
<tr>
<td>ibus-table-thai</td>
<td>I:0</td>
<td>58</td>
<td>Thai</td>
</tr>
<tr>
<td>ibus-unkey</td>
<td>V:0, I:0</td>
<td>318</td>
<td>Vietnamese</td>
</tr>
<tr>
<td>ibus-keyman</td>
<td>V:0, I:0</td>
<td>137</td>
<td>Multilingual: Keyman engine for over 2000 languages</td>
</tr>
<tr>
<td>ibus-table</td>
<td>V:0, I:1</td>
<td>2158</td>
<td>table engine for IBus</td>
</tr>
<tr>
<td>ibus-m17n</td>
<td>V:0, I:1</td>
<td>382</td>
<td>Multilingual: Indic, Arabic and others</td>
</tr>
<tr>
<td>plasma-widgets-addons</td>
<td>V:46, I:96</td>
<td>1992</td>
<td>additional widgets for Plasma 5 containing Keyboard Indicator</td>
</tr>
</tbody>
</table>

Table 8.1: List of IBus and its engine packages

2. Select "Settings" → "Keyboard" → "Input Sources" → click "+" in "Input Sources" → "Japanese" → "Japanese moz (or anthy)" and click "Add" if it has n't been activated.

3. You may chose as many input sources.

4. Re login to user's account.

5. Setup each input source by right clicking the GUI toolbar icon.

6. Switch among installed input sources by SUPER-SPACE. (SUPER is normally the Windows key.)

**Tip**

If you wish to have access to alphabet only keyboard environment with the physical Japanese keyboard on which shift-2 has " (double quotation mark) engraved, you select "Japanese" in the above procedure. You can enter Japanese using "Japanese moz (or anthy)" with physical "US" keyboard on which shift-2 has @ (at mark) engraved.

- The GUI menu entry for `im-config(8)` is "Input method".
- Alternatively, execute "im-config" from user's shell.
- `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.

### 8.3 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-GUI console.)

GUI environment (Chapter 7) can display any characters in the UTF-8 as long as required fonts are installed and enabled. (The encoding of the original font data is taken care and transparent to the user.)
8.4  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**: Preferences → Profiles → *Profile name* → Compatibility → Ambiguous-wide characters → Narrow

• **ncurses**: Set environment `export NCURSES_NO_UTF8_ACS=0`. 
Chapter 9

System tips

Here, I describe basic tips to configure and manage systems, mostly from the console.

9.1 The console tips

There are some utility programs to help your console activities.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc</td>
<td>V:50,1:212</td>
<td>1490</td>
<td>See Section 1.3</td>
</tr>
<tr>
<td>bsdutils</td>
<td>V:551,1:999</td>
<td>356</td>
<td>script(1) command to make a record of terminal session</td>
</tr>
<tr>
<td>screen</td>
<td>V:74,1:234</td>
<td>1003</td>
<td>terminal multiplexer with VT100/ANSI terminal emulation</td>
</tr>
<tr>
<td>tmux</td>
<td>V:41,1:143</td>
<td>1114</td>
<td>terminal multiplexer alternative (Use &quot;Control-B&quot; instead)</td>
</tr>
<tr>
<td>fzf</td>
<td>V:4,1:14</td>
<td>3476</td>
<td>fuzzy text finder</td>
</tr>
<tr>
<td>fzy</td>
<td>V:0,1:10</td>
<td>54</td>
<td>fuzzy text finder</td>
</tr>
<tr>
<td>rlwrap</td>
<td>V:1,1:15</td>
<td>330</td>
<td>readline feature command line wrapper</td>
</tr>
<tr>
<td>ledit</td>
<td>V:0,1:11</td>
<td>331</td>
<td>readline feature command line wrapper</td>
</tr>
<tr>
<td>rlfe</td>
<td>V:0,1:10</td>
<td>42</td>
<td>readline feature command line wrapper</td>
</tr>
<tr>
<td>ripgrep</td>
<td>V:4,1:16</td>
<td>4521</td>
<td>fast recursive string search in the source code tree with automatic filtering</td>
</tr>
</tbody>
</table>

Table 9.1: List of programs to support console activities

9.1.1 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
```

There are alternative methods to record the shell activities:

- Use tee (usable during the boot process in the initramfs):
$ sh -i 2>&1 | tee typescript

- Use `gnome-terminal` with the extend line buffer for scrollback.
- Use `screen` with "^A H" (see Section 9.1.2) to perform recording of console.
- Use `vim` with ":terminal" to enter the terminal mode. Use "Ctrl-W N" to exit from terminal mode to normal mode. Use "w typescript" to write the buffer to a file.
- Use `emacs` with "M-x shell", "M-x eshell", or "M-x term" to enter recording console. Use "C-x C-w" to write the buffer to a file.

### 9.1.2 The screen program

`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 `^A c` ("Control-A" followed by "c").
4. You switch among the multiple `screen` windows by `^A 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.
   - Bravely unplug your network connection
   - Type `^A d` ("Control-A" followed by "d") and manually logging out from the remote connection
   - Type `^A 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.

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 `^A` ("Control-A") plus a single key [plus any parameters]. Here are important ones to remember.

See `screen(1)` for details.

See `tmux(1)` for functionalities of the alternative command.
### Table 9.2: List of key bindings for screen

<table>
<thead>
<tr>
<th>key binding</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>^A ?</td>
<td>show a help screen (display key bindings)</td>
</tr>
<tr>
<td>^A c</td>
<td>create a new window and switch to it</td>
</tr>
<tr>
<td>^A n</td>
<td>go to next window</td>
</tr>
<tr>
<td>^A p</td>
<td>go to previous window</td>
</tr>
<tr>
<td>^A 0</td>
<td>go to window number 0</td>
</tr>
<tr>
<td>^A 1</td>
<td>go to window number 1</td>
</tr>
<tr>
<td>^A w</td>
<td>show a list of windows</td>
</tr>
<tr>
<td>^A a</td>
<td>send a Ctrl-A to current window as keyboard input</td>
</tr>
<tr>
<td>^A h</td>
<td>write a hardcopy of current window to file</td>
</tr>
<tr>
<td>^A H</td>
<td>begin/end logging current window to file</td>
</tr>
<tr>
<td>^A ^X</td>
<td>lock the terminal (password protected)</td>
</tr>
<tr>
<td>^A d</td>
<td>detach screen session from the terminal</td>
</tr>
<tr>
<td>^A DD</td>
<td>detach screen session and log out</td>
</tr>
</tbody>
</table>

#### 9.1.3 Navigating around directories

In Section 1.4.2, 2 tips to allow quick navigation around directories are described: `SCDPATH` and `mc`.

If you use fuzzy text filter program, you can do without typing the exact path. For `fzf`, include following in `~/.bashrc`.

```bash
FZF_KEYBINDINGS_PATH=/usr/share/doc/fzf/examples/key-bindings.bash
if [ -f $FZF_KEYBINDINGS_PATH ]; then
  . $FZF_KEYBINDINGS_PATH
fi

FZF_COMPLETION_PATH=/usr/share/doc/fzf/examples/completion.bash
if [ -f $FZF_COMPLETION_PATH ]; then
  . $FZF_COMPLETION_PATH
fi
```

For example:

- You can jump to a very deep subdirectory with minimal efforts. You first type “cd **” and press Tab. Then you will be prompted with candidate paths. Typing in partial path strings, e.g., `s/d/b foo`, will narrow down candidate paths. You select the path to be used by `cd` with cursor and return keys.

- You can select a command from the command history more efficiently with minimal efforts. You press `Ctrl-R` at the command prompt. Then you will be prompted with candidate commands. Typing in partial command strings, e.g., `vim d`, will narrow down candidates. You select the one to be used with cursor and return keys.

#### 9.1.4 Readline wrapper

Some commands such as `/usr/bin/dash` which lacks command line history editing capability can add such functionality transparently by running under `rlwrap` or its equivalents.

```
$ rlwrap dash -i
```

This provides convenient platform to test subtle points for `dash` with friendly `bash`-like environment.

#### 9.1.5 Scanning the source code tree

The `rg(1)` command in the `ripgrep` package offers a faster alternative to the `grep(1)` command for scanning the source code tree for typical situation. It takes advantage of modern multi-core CPUs and automatically applies reasonable filters to skip some files.
9.2 Customizing vim

After you learn basics of vim through Section 1.4.8, please read Bram Moolenaar’s "Seven habits of effective text editing (2000)” to understand how vim should be used.

\[ Caution \]
Don’t try to change the default key bindings without very good reasons.

9.2.1 Customizing vim with internal features

The behavior of vim can be changed significantly by enabling its internal features through the Ex-mode commands such as "set ...” to set vim options.

These Ex-mode commands can be included in user’s vimrc file, traditional “~/.vimrc” or git-friendly “~/.vim/vimrc”. Here is a very simple example 1:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>colorscheme murphy</td>
<td>&quot; from /usr/share/vim/vim??/colors/*.vim</td>
</tr>
<tr>
<td>filetype plugin indent on</td>
<td>&quot; filetype aware behavior</td>
</tr>
<tr>
<td>syntax enable</td>
<td>&quot; Syntax highlight</td>
</tr>
<tr>
<td>&quot;set spelllang=en_us</td>
<td>&quot; Spell check language as en_us</td>
</tr>
<tr>
<td>&quot;set spell</td>
<td>&quot; Enable spell check</td>
</tr>
<tr>
<td>set autoindent</td>
<td>&quot; Copy indent from current line</td>
</tr>
<tr>
<td>set smartindent</td>
<td>&quot; More than autoindent (Drop/Pop after {/})</td>
</tr>
<tr>
<td>set nosmarttab</td>
<td>&quot; &lt;Tab&gt;-key always inserts blanks</td>
</tr>
<tr>
<td>set backspace=indent,eol,start</td>
<td>Back space through everything</td>
</tr>
<tr>
<td>set laststatus=2</td>
<td>&quot; Always show status line</td>
</tr>
<tr>
<td>set statusline</td>
<td>%=&lt;%f%m%r%h%w%=y[U+%@4B]%2l/%2L=%P,%2c%V</td>
</tr>
</tbody>
</table>

9.2.2 Customizing vim with external packages

Simple customization to enable secure-modelines and classical IDE can be enabled by installing vim-scripts package and appending the following to user's vimrc file.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>packadd! secure-modelines</td>
<td></td>
</tr>
<tr>
<td>packadd! winmanager</td>
<td></td>
</tr>
<tr>
<td>let mapleader = ' '</td>
<td>&quot; Toggle paste mode with &lt;SPACE&gt;p</td>
</tr>
<tr>
<td>set pastetoggle=&lt;leader&gt;p</td>
<td></td>
</tr>
<tr>
<td>&quot; IDE-like UI for files and buffers with &lt;space&gt;w</td>
<td></td>
</tr>
<tr>
<td>nnoremap &lt;leader&gt;w</td>
<td>:WMToggle&lt;CR&gt;</td>
</tr>
<tr>
<td>&quot; Use safer keys &lt;C-?&gt; for moving to another window</td>
<td></td>
</tr>
<tr>
<td>nnoremap &lt;C-H&gt;</td>
<td>&lt;C-W&gt;H</td>
</tr>
<tr>
<td>nnoremap &lt;C-J&gt;</td>
<td>&lt;C-W&gt;j</td>
</tr>
<tr>
<td>nnoremap &lt;C-K&gt;</td>
<td>&lt;C-W&gt;k</td>
</tr>
<tr>
<td>nnoremap &lt;C-L&gt;</td>
<td>&lt;C-W&gt;l</td>
</tr>
</tbody>
</table>

In order for the above keybindings to function properly, the terminal program needs to be configured to generate "ASCII DEL” for Backspace-key and "Escape sequence” for Delete-key.

The new native Vim package system works nicely with "git" and "git submodule”. One such example configuration can be found at my git repository: dot-vim. This does essentially:

\[1\]More elaborate customization examples: “Vim Galore”, "sensible.vim", “#vim Recommendations” ...
• By using "git" and "git submodule", latest external packages, such as "name", are placed into ~/.vim/pack/*/opt/name and similar.
• By adding :packadd! name line to user's vimrc file, these packages are placed on runtimepath.
• Vim loads these packages on runtimepath during its initialization.
• At the end of its initialization, tags for the installed documents are updated with "helptags ALL".

For more, please start vim with "vim --startuptime vimstart.log" to check actual execution sequence and time spent for each step.

Interesting external plugin packages can be found:

• Vim - the ubiquitous text editor -- The official upstream site of Vim and vim scripts
• VimAwesome -- The listing of Vim plugins
• vim-scripts -- Debian package: a collection of vim scripts

It is quite confusing to see too many ways2 to manage and load these external packages to vim. Checking the original information is the best cure.

<table>
<thead>
<tr>
<th>keystrokes</th>
<th>information</th>
</tr>
</thead>
<tbody>
<tr>
<td>:help package</td>
<td>explanation on the vim package mechanism</td>
</tr>
<tr>
<td>:help runtimepath</td>
<td>explanation on the runtimepath mechanism</td>
</tr>
<tr>
<td>:version</td>
<td>internal states including candidates for the vimrc file</td>
</tr>
<tr>
<td>:echo $VIM</td>
<td>the environment variable &quot;$VIM&quot; used to locate the vimrc file</td>
</tr>
<tr>
<td>:set runtimepath?</td>
<td>list of directories which will be searched for all runtime support files</td>
</tr>
<tr>
<td>:echo $VIMRUNTIME</td>
<td>the environment variable &quot;$VIMRUNTIME&quot; used to locate various system provided runtime support files</td>
</tr>
</tbody>
</table>

Table 9.3: Information on the initialization of vim

9.3 Data recording and presentation

9.3.1 The log daemon

Many traditional programs record their activities in the text file format under the "/var/log/" directory.

logrotate(8) is used to simplify the administration of log files on a system which generates a lot of log files.

Many new programs record their activities in the binary file format using systemd-journald(8) Journal service under the "/var/log/journal" directory.

You can log data to the systemd-journald(8) Journal from a shell script by using the systemd-cat(1) command.

See Section 3.4 and Section 3.3.

9.3.2 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.

---

2vim-pathogen was popular.
### 9.3.3 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.

### 9.3.4 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 "$SLANG" variable is referred first and it can be overridden by the "$SLC_TIME" or "$LC_ALL" exported environment variables. 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. For iso-formats, see [ISO8601](https://en.wikipedia.org/wiki/ISO_8601).

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'"`).

<table>
<thead>
<tr>
<th>time style value</th>
<th>locale</th>
<th>display of time and date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>iso</strong></td>
<td>any</td>
<td>01-19 00:15</td>
</tr>
<tr>
<td><strong>long-iso</strong></td>
<td>any</td>
<td>2009-01-19 00:15:00:00:00</td>
</tr>
<tr>
<td><strong>full-iso</strong></td>
<td>any</td>
<td>2009-01-19 00:15:00:00:00</td>
</tr>
<tr>
<td><strong>locale</strong></td>
<td>C</td>
<td>Jan 19 00:15</td>
</tr>
<tr>
<td><strong>locale</strong></td>
<td>en_US.UTF-8</td>
<td>Jan 19 00:15</td>
</tr>
<tr>
<td><strong>locale</strong></td>
<td>es_ES.UTF-8</td>
<td>ene 19 00:15</td>
</tr>
<tr>
<td>+%d.%m.%y %H:%M</td>
<td>any</td>
<td>19.01.09 00:15</td>
</tr>
<tr>
<td>+%d.%b.%y %H:%M</td>
<td>C or en_US.UTF-8</td>
<td>19.Jan.09 00:15</td>
</tr>
<tr>
<td>+%d.%b.%y %H:%M</td>
<td>es_ES.UTF-8</td>
<td>19.ene.09 00:15</td>
</tr>
</tbody>
</table>

Table 9.5: Display examples of time and date for the "`ls -l`" command with the `time style value`.

**Tip**

You can eliminate typing long option on commandline using command alias (see Section 1.5.9):

```
alias ls='ls --time-style=+%d.%m.%y %H:%M'
```
9.3.5 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))"
```

9.3.6 Colorized commands

Colorized commands are handy for inspecting their output in the interactive environment. I include the following in my "~/.bashrc”.

```bash
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”.

9.3.7 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.
### 9.3.8 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>screen</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnome-screenshot</td>
<td>V:20, I:194</td>
<td>1134</td>
<td>Wayland</td>
</tr>
<tr>
<td>flameshot</td>
<td>V:8, I:15</td>
<td>3364</td>
<td>Wayland</td>
</tr>
<tr>
<td>gimp</td>
<td>V:53, I:258</td>
<td>19877</td>
<td>Wayland + X</td>
</tr>
<tr>
<td>x11-apps</td>
<td>V:30, I:463</td>
<td>2476</td>
<td>X</td>
</tr>
<tr>
<td>imagemagick</td>
<td>I:320</td>
<td>73</td>
<td>X</td>
</tr>
<tr>
<td>scrot</td>
<td>V:5, I:65</td>
<td>131</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 9.6: List of graphic image manipulation tools

### 9.3.9 Recording changes in configuration files

There are specialized tools to record changes in configuration files with help of DVCS and to make system snapshots on Btrfs.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>etckeeper</td>
<td>V:26, I:29</td>
<td>164</td>
<td>store configuration files and their metadata with Git (default), Mercurial, or Bazaar (new)</td>
</tr>
<tr>
<td>timeshift</td>
<td>V:5, I:19</td>
<td>3421</td>
<td>system restore utility using rsync or BTRFS snapshots</td>
</tr>
<tr>
<td>snapper</td>
<td>V:4, I:5</td>
<td>2419</td>
<td>Linux filesystem snapshot management tool</td>
</tr>
</tbody>
</table>

Table 9.7: List of packages which can record configuration history

You may also think about local script Section 10.2.3 approach.

### 9.4 Monitoring, controlling, and starting program activities

Program activities can be monitored and controlled using specialized tools.

**Tip**

The procps packages provide very basics of monitoring, controlling, and starting program activities. You should learn all of them.

### 9.4.1 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
```
### Table 9.8: List of tools for monitoring and controlling program activities

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>nice(1): run a program with modified scheduling priority</td>
</tr>
<tr>
<td>bsdutils</td>
<td>V:551, I:999</td>
<td>356</td>
<td>renice(1): modify the scheduling priority of a running process</td>
</tr>
<tr>
<td>procps</td>
<td>V:755, I:999</td>
<td>2390</td>
<td>”/proc” filesystem utilities: ps(1), top(1), kill(1), watch(1), …</td>
</tr>
<tr>
<td>psmisc</td>
<td>V:421, I:782</td>
<td>909</td>
<td>”/proc” filesystem utilities: killall(1), fuser(1), peekfd(1), pstree(1)</td>
</tr>
<tr>
<td>time</td>
<td>V:8, I:143</td>
<td>129</td>
<td>time(1): run a program to report system resource usages with respect to time</td>
</tr>
<tr>
<td>sysstat</td>
<td>V:152, I:173</td>
<td>1785</td>
<td>sar(1), iostat(1), mpstat(1), …: system performance tools for Linux</td>
</tr>
<tr>
<td>iasg</td>
<td>V:0, I:3</td>
<td>106</td>
<td>Interactive System Activity Grapher for sysstat</td>
</tr>
<tr>
<td>lsof</td>
<td>V:413, I:945</td>
<td>482</td>
<td>lsof(8): list files opened by a running process using ”-p” option</td>
</tr>
<tr>
<td>strace</td>
<td>V:12, I:121</td>
<td>2875</td>
<td>strace(1): trace system calls and signals</td>
</tr>
<tr>
<td>ltrace</td>
<td>V:9, I:16</td>
<td>330</td>
<td>ltrace(1): trace library calls</td>
</tr>
<tr>
<td>xtrace</td>
<td>V:9, I:0</td>
<td>353</td>
<td>xtrace(1): trace communication between X11 client and server</td>
</tr>
<tr>
<td>powertop</td>
<td>V:18, I:216</td>
<td>669</td>
<td>powertop(1): information about system power use</td>
</tr>
<tr>
<td>cron</td>
<td>V:862, I:995</td>
<td>235</td>
<td>run processes according to a schedule in background from cron(8) daemon</td>
</tr>
<tr>
<td>anacron</td>
<td>V:393, I:480</td>
<td>92</td>
<td>cron-like command scheduler for systems that don’t run 24 hours a day</td>
</tr>
<tr>
<td>at</td>
<td>V:107, I:165</td>
<td>158</td>
<td>at(1) or batch(1): run a job at a specified time or below certain load level</td>
</tr>
</tbody>
</table>

### Table 9.9: List of nice values for the scheduling priority

<table>
<thead>
<tr>
<th>nice value</th>
<th>scheduling priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>lowest priority process (nice)</td>
</tr>
<tr>
<td>0</td>
<td>very high priority process for user</td>
</tr>
<tr>
<td>-20</td>
<td>very high priority process for root (not-nice)</td>
</tr>
</tbody>
</table>
9.4.2 The scheduling priority

A nice value is used to control the scheduling priority for the process.

```
# 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.

9.4.3 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.

<table>
<thead>
<tr>
<th>style</th>
<th>typical command</th>
<th>feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSD</td>
<td><code>ps aux</code></td>
<td>display %CPU %MEM</td>
</tr>
<tr>
<td>System V</td>
<td><code>ps -efH</code></td>
<td>display PPID</td>
</tr>
</tbody>
</table>

Table 9.10: 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 `pstat(1)` command display a tree of processes.

9.4.4 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.

9.4.5 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.

9.4.6 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
```

Tip

Use `strace-graph` script found in `/usr/share/doc/strace/examples/` to make a nice tree view.
9.4.7 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
```

<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>ACCESS</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>root</td>
<td>2946</td>
<td>F</td>
<td>rsyslogd</td>
</tr>
</tbody>
</table>

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
```

<table>
<thead>
<tr>
<th>USER</th>
<th>PID</th>
<th>ACCESS</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian-exim</td>
<td>3379</td>
<td>F</td>
<td>exim4</td>
</tr>
</tbody>
</table>

Now you know your system runs `exim4(8)` to handle TCP connections to SMTP port (25).

9.4.8 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.

9.4.9 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)`.
9.4.10 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/applications directory provide necessary attributes for the GUI menu display of each program. 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".

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.
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).

### 9.4.11 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 environ(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 GUI as your preferred application, you should create an GUI application as following and set "/usr/local/bin/bin/mutt-term" as your preferred application to be started as described.

```bash
# cat /usr/local/bin/mutt-term <<EOF
#!/bin/sh
gnome-terminal -e "mutt \$@"
EOF
# chmod 755 /usr/local/bin/mutt-term
```
**Tip**

GUI application can be executed easily under specific environment variables if the program for "Exec" in its corresponding `*.desktop` file doesn't specify the full execution path. You can enable non-English keyboard input to `kitty` using an input method framework `ibus` or `fcitx5` by installing required input method packages and creating a shell script "/usr/local/bin/kitty" as follows.

```bash
# cat /usr/local/bin/kitty <<EOF
#!/bin/sh
GLFW_IM_MODULE=ibus exec /usr/bin/kitty "$@
EOF
# chmod 755 /usr/local/bin/kitty
```

This `kitty` script can alternatively be stored in "~/bin/" or "~/.local/bin/" directories as long as they are defined earlier than "/usr/bin/" in "$PATH".

---

### 9.4.12 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.

<table>
<thead>
<tr>
<th>signal value</th>
<th>signal name</th>
<th>action</th>
<th>note</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>---</td>
<td>no signal is sent (see <code>kill(2)</code>)</td>
<td>check if process is running</td>
</tr>
<tr>
<td>1</td>
<td>SIGHUP</td>
<td>terminate the process</td>
<td>disconnected terminal (signal hang up)</td>
</tr>
<tr>
<td>2</td>
<td>SIGINT</td>
<td>terminate the process</td>
<td>interrupt from keyboard (CTRL-C)</td>
</tr>
<tr>
<td>3</td>
<td>SIGQUIT</td>
<td>terminate the process and dump core</td>
<td>quit from keyboard (CTRL-)</td>
</tr>
<tr>
<td>9</td>
<td>SIGKILL</td>
<td>terminate the process</td>
<td>unblockable kill signal</td>
</tr>
<tr>
<td>15</td>
<td>SIGTERM</td>
<td>terminate the process</td>
<td>blockable termination signal</td>
</tr>
</tbody>
</table>

Table 9.11: List of frequently used signals for kill command
9.4.13 Scheduling tasks once

Run the `at(1)` command to schedule a one-time job by the following.

```
$ echo 'command -args' | at 3:40 monday
```

9.4.14 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/yourname" 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 * * $HOME/bk/monthly
# run at 22:00 on weekdays(1-5), annoy Joe. % for newline, last % for cc:
0 22 * * * mail -s "It's 10pm" joe%Joe,%%Where are your kids?%.%
23 */2 1 2 * echo "run 23 minutes after 0am, 2am, 4am ..., on Feb 1"
5 4 * * sun echo "run at 04:05 every Sunday"
# run at 03:40 on the first Monday of each month
40 3 1-7 * * [ "$(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".

Systemd has low level capability to schedule programs to run without `cron` daemon. For example, "/lib/systemd/system/apt-daily.timer" and "/lib/systemd/system/apt-daily.service" set up daily apt download activities. See `systemd.timer(5)`.

9.4.15 Alt-SysRq key

Pressing Alt-SysRq (PrtScr) followed by one keys does the magic of rescuing control of the system.

See more on Linux kernel user’s and administrator’s guide » Linux Magic System Request Key Hacks

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 syncs and unmounts all mounted filesystems.
<table>
<thead>
<tr>
<th>key following Alt-SysRq</th>
<th>description of action</th>
</tr>
</thead>
<tbody>
<tr>
<td>k</td>
<td>kill all processes on the current virtual console (SAK)</td>
</tr>
<tr>
<td>s</td>
<td>sync all mounted filesystems to avoid data corruption</td>
</tr>
<tr>
<td>u</td>
<td>remount all mounted filesystems read-only (umount)</td>
</tr>
<tr>
<td>r</td>
<td>restore the keyboard from raw mode after X crashes</td>
</tr>
</tbody>
</table>

Table 9.12: List of notable SAK command keys

The current (2021) Debian amd64 Linux kernel has /proc/sys/kernel/sysrq=438=0b110110110:

- 2 = 0x2 - enable control of console logging level (ON)
- 4 = 0x4 - enable control of keyboard (SAK, unraw) (ON)
- 8 = 0x8 - enable debugging dumps of processes etc. (OFF)
- 16 = 0x10 - enable sync command (ON)
- 32 = 0x20 - enable remount read-only (ON)
- 64 = 0x40 - enable signaling of processes (term, kill, oom-kill) (OFF)
- 128 = 0x80 - allow reboot/poweroff (ON)
- 256 = 0x100 - allow nicing of all RT tasks (ON)

9.5 System maintenance tips

9.5.1 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).

9.5.2 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

9.5.3 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).
### 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>console-setup</td>
<td>V:93, I:967</td>
<td>428</td>
<td>Linux console font and keytable utilities</td>
</tr>
<tr>
<td>x11-xserver-utils</td>
<td>V:303, I:527</td>
<td>576</td>
<td>X server utilities: xset(1), xmodmap(1)</td>
</tr>
<tr>
<td>acpid</td>
<td>V:91, I:161</td>
<td>154</td>
<td>daemon to manage events delivered by the Advanced Configuration and Power Interface (ACPI)</td>
</tr>
<tr>
<td>acpi</td>
<td>V:10, I:148</td>
<td>47</td>
<td>utility to display information on ACPI devices</td>
</tr>
<tr>
<td>sleepd</td>
<td>V:0, I:0</td>
<td>86</td>
<td>daemon to put a laptop to sleep during inactivity</td>
</tr>
<tr>
<td>hdparm</td>
<td>V:203, I:366</td>
<td>256</td>
<td>hard disk access optimization (see Section 9.6.9)</td>
</tr>
<tr>
<td>smartmontools</td>
<td>V:200, I:244</td>
<td>2199</td>
<td>control and monitor storage systems using S.M.A.R.T.</td>
</tr>
<tr>
<td>setserial</td>
<td>V:4, I:7</td>
<td>103</td>
<td>collection of tools for serial port management</td>
</tr>
<tr>
<td>memtest86+</td>
<td>V:1, I:21</td>
<td>12687</td>
<td>collection of tools for memory hardware management</td>
</tr>
<tr>
<td>scsitools</td>
<td>V:0, I:12</td>
<td>375</td>
<td>collection of tools for SCSI hardware management</td>
</tr>
<tr>
<td>setcd</td>
<td>V:0, I:0</td>
<td>37</td>
<td>compact disc drive access optimization</td>
</tr>
<tr>
<td>big-cursor</td>
<td>I:0</td>
<td>26</td>
<td>larger mouse cursors for X</td>
</tr>
</tbody>
</table>

Table 9.14: List of hardware configuration tools

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.

### 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 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.
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.

9.5.6 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,$"TERM", from "xterm" to one of the feature-limited versions such as "xterm-r6" when you log in to a Debian system remotely. See "/usr/share/doc/libncurses5/FAQ" for more. "dumb" is the lowest common denominator for "$TERM".

9.5.7 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.

Application softwares may be configured not only to access sound devices directly but also to access them via some standardized sound server system. Currently, PulseAudio, JACK, and PipeWire are used as sound server system. See Debian wiki page on Sound for the latest situation.

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.

Tip
Use "cat /dev/urandom > /dev/audio" or speaker-test(1) to test speaker (^C 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.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alsa-utils</td>
<td>V:330, I:467</td>
<td>2607</td>
<td>utilities for configuring and using ALSA</td>
</tr>
<tr>
<td>oss-compat</td>
<td>V:1, I:18</td>
<td>20</td>
<td>OSS compatibility under ALSA preventing &quot;/dev/dsp not found&quot; errors</td>
</tr>
<tr>
<td>pipewire</td>
<td>V:264, I:318</td>
<td>116</td>
<td>audio and video processing engine multimedia server - metapackage</td>
</tr>
<tr>
<td>pipewire-bin</td>
<td>V:275, I:318</td>
<td>1618</td>
<td>audio and video processing engine multimedia server - audio server and CLI programs</td>
</tr>
<tr>
<td>pipewire-alsa</td>
<td>V:85, I:133</td>
<td>205</td>
<td>audio and video processing engine multimedia server - audio server to replace ALSA</td>
</tr>
<tr>
<td>pipewire-pulse</td>
<td>V:136, I:183</td>
<td>49</td>
<td>audio and video processing engine multimedia server - audio server to replace PulseAudio</td>
</tr>
<tr>
<td>pulseaudio</td>
<td>V:278, I:333</td>
<td>6462</td>
<td>PulseAudio server</td>
</tr>
<tr>
<td>libpulse0</td>
<td>V:409, I:581</td>
<td>969</td>
<td>PulseAudio client library</td>
</tr>
<tr>
<td>jackd</td>
<td>V:2, I:19</td>
<td>9</td>
<td>JACK Audio Connection Kit. (JACK) server (low latency)</td>
</tr>
<tr>
<td>libjack0</td>
<td>V:1, I:9</td>
<td>329</td>
<td>JACK Audio Connection Kit. (JACK) library (low latency)</td>
</tr>
<tr>
<td>libgstreamer1.0-9</td>
<td>V:428, I:594</td>
<td>4449</td>
<td>GStreamer: GNOME sound engine</td>
</tr>
<tr>
<td>libphonon4qt5-4</td>
<td>V:72, I:162</td>
<td>572</td>
<td>Phonon: KDE sound engine</td>
</tr>
</tbody>
</table>

Table 9.15: List of sound packages

### 9.5.8 Disabling the screen saver

For disabling the screen saver, use following commands.

<table>
<thead>
<tr>
<th>environment</th>
<th>command</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Linux console</td>
<td><code>setterm -powersave off</code></td>
</tr>
<tr>
<td>The X Window (turning off screensaver)</td>
<td><code>xset s off</code></td>
</tr>
<tr>
<td>The X Window (disabling dpms)</td>
<td><code>xset -dpms</code></td>
</tr>
<tr>
<td>The X Window (GUI configuration of screen saver)</td>
<td><code>xscreensaver-command -prefs</code></td>
</tr>
</tbody>
</table>

Table 9.16: List of commands for disabling the screen saver

### 9.5.9 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
```

### 9.5.10 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.
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.

<table>
<thead>
<tr>
<th>report</th>
<th>size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total size in dmesg</td>
<td>1016784k = 1GB - 31792k</td>
</tr>
<tr>
<td>Free in dmesg</td>
<td>990528k</td>
</tr>
<tr>
<td>Total under shell</td>
<td>997184k</td>
</tr>
<tr>
<td>Free under shell</td>
<td>20256k (but effectively 321780k)</td>
</tr>
</tbody>
</table>

Table 9.17: List of memory sizes reported

### 9.5.11 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**.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logcheck</td>
<td>V6, I8</td>
<td>110</td>
<td>daemon to mail anomalies in the system logfiles to the administrator</td>
</tr>
<tr>
<td>debsums</td>
<td>V3, I36</td>
<td>98</td>
<td>utility to verify installed package files against MD5 checksums</td>
</tr>
<tr>
<td>chkrootkit</td>
<td>V7, I18</td>
<td>926</td>
<td>rootkit detector</td>
</tr>
<tr>
<td>clamav</td>
<td>V9, I46</td>
<td>27703</td>
<td>anti-virus utility for Unix - command-line interface</td>
</tr>
<tr>
<td>tiger</td>
<td>V1, I2</td>
<td>7800</td>
<td>report system security vulnerabilities</td>
</tr>
<tr>
<td>tripwire</td>
<td>V1, I2</td>
<td>12168</td>
<td>file and directory integrity checker</td>
</tr>
<tr>
<td>john</td>
<td>V1, I9</td>
<td>471</td>
<td>active password cracking tool</td>
</tr>
<tr>
<td>aide</td>
<td>V1, I1</td>
<td>293</td>
<td>Advanced Intrusion Detection Environment - static binary</td>
</tr>
<tr>
<td>integrit</td>
<td>V3, I0</td>
<td>2659</td>
<td>file integrity verification program</td>
</tr>
<tr>
<td>crack</td>
<td>V0, I1</td>
<td>152</td>
<td>password guessing program</td>
</tr>
</tbody>
</table>

Table 9.18: 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 cannot be fully trusted as the system security audit tool against malicious attacks.

---

### 9.6 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.

You may need to `umount(8)` some devices manually from the command line before operating on them if they are automatically mounted by the GUI desktop system.

#### 9.6.1 Disk space usage

The disk space usage can be evaluated by programs provided by the `mount`, `coreutils`, and `xdue` 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 `xdue(1x)` to produce its graphical and interactive presentation with "`du -k . | xdue`, "`sudo du -k -x / | xdue`, etc.

#### 9.6.2 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.

Older 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).

Recent PCs with Unified Extensible Firmware Interface (UEFI), 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>util-linux</td>
<td>V:876, I:999</td>
<td>5279</td>
<td>miscellaneous system utilities including <code>fdisk(8)</code> and <code>cfdisk(8)</code></td>
</tr>
<tr>
<td>parted</td>
<td>V:410, I:567</td>
<td>122</td>
<td>GNU Parted disk partition resizing program</td>
</tr>
<tr>
<td>gparted</td>
<td>V:14, I:103</td>
<td>2175</td>
<td>GNOME partition editor based on <code>libparted</code></td>
</tr>
<tr>
<td>gdisk</td>
<td>V:325, I:510</td>
<td>885</td>
<td>partition editor for the GPT/MBR hybrid disk</td>
</tr>
<tr>
<td>kpartx</td>
<td>V:22, I:33</td>
<td>75</td>
<td>program to create device mappings for partitions</td>
</tr>
</tbody>
</table>

Table 9.19: 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 9.8.6) and use "parted /dev/sdx mklabel gpt" or "parted /dev/sdx mklabel msdos" to set it. Please note "msdos" is use here for MBR.

9.6.3 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/UEFI 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).
You can also probe UUID and other information with "lsblk -f".

9.6.4 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
9.6.5 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 `undelete` 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>e2fsprogs</td>
<td>V:748, I:999</td>
<td>1501</td>
<td>utilities for the ext2/ext3/ext4 filesystems</td>
</tr>
<tr>
<td>btrfs-progs</td>
<td>V:45, I:71</td>
<td>4851</td>
<td>utilities for the Btrfs filesystem</td>
</tr>
<tr>
<td>reiserfsprogs</td>
<td>V:12, I:25</td>
<td>473</td>
<td>utilities for the Reiserfs filesystem</td>
</tr>
<tr>
<td>zfs-utils-linux</td>
<td>V:28, I:29</td>
<td>1619</td>
<td>utilities for the OpenZFS filesystem</td>
</tr>
<tr>
<td>dosfstools</td>
<td>V:187, I:535</td>
<td>315</td>
<td>utilities for the FAT filesystem. (Microsoft: MS-DOS, Windows)</td>
</tr>
<tr>
<td>exfatprogs</td>
<td>V:25, I:341</td>
<td>301</td>
<td>utilities for the exFAT filesystem maintained by Samsung.</td>
</tr>
<tr>
<td>exfat-fuse</td>
<td>V:6, I:145</td>
<td>73</td>
<td>read/write exFAT filesystem (Microsoft) driver for FUSE.</td>
</tr>
<tr>
<td>exfat-utils</td>
<td>V:4, I:133</td>
<td>231</td>
<td>utilities for the exFAT filesystem maintained by the exfat-fuse author.</td>
</tr>
<tr>
<td>xfsprogs</td>
<td>V:23, I:95</td>
<td>3493</td>
<td>utilities for the XFS filesystem. (SGI: IRIX)</td>
</tr>
<tr>
<td>ntfs-3g</td>
<td>V:198, I:512</td>
<td>1470</td>
<td>read/write NTFS filesystem (Microsoft: Windows NT, ...) driver for FUSE.</td>
</tr>
<tr>
<td>jfsutils</td>
<td>V:0, I:8</td>
<td>1577</td>
<td>utilities for the JFS filesystem. (IBM: AIX, OS/2)</td>
</tr>
<tr>
<td>reiser4progs</td>
<td>V:0, I:2</td>
<td>1367</td>
<td>utilities for the Reiser4 filesystem</td>
</tr>
<tr>
<td>hfsprogs</td>
<td>V:0, I:5</td>
<td>394</td>
<td>utilities for HFS and HFS Plus filesystem. (Apple: Mac OS)</td>
</tr>
<tr>
<td>zerofree</td>
<td>V:5, I:129</td>
<td>25</td>
<td>program to zero free blocks from ext2/3/4 filesystems</td>
</tr>
</tbody>
</table>

Table 9.20: List of filesystem management packages

**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.

Btrfs status can be found at Debian wiki on btrfs and kernel.org wiki on btrfs. It is expected to be the next default filesystem after the ext4 filesystem.

Some tools allow access to filesystem without Linux kernel support (see Section 9.8.2).

9.6.6 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). Check files in "/var/log/fsck/" for the result of the fsck(8) command run from the boot script.

9.6.7 Optimization of filesystem by mount options

The basic static filesystem configuration is given by "/etc/fstab". For example,

```
<table>
<thead>
<tr>
<th>«file system»</th>
<th>«mount point»</th>
<th>«type»</th>
<th>«options»</th>
<th>«dump»</th>
<th>«pass»</th>
</tr>
</thead>
<tbody>
<tr>
<td>proc</td>
<td>/proc</td>
<td>proc</td>
<td>defaults</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>UUID=709cbe4c-80c1-56db-8ab1-dbce3146d2f7 /</td>
<td>ext4</td>
<td>errors=remount-ro 0 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UUID=817bae6b-45d2-5aca-4d2a-1267ab46ac23 none swap sw 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>/dev/scd0</td>
<td>/media/cdrom</td>
<td>udf,iso9660 user,noauto</td>
<td>0 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Tip
UUID (see Section 9.6.3) may be used to identify a block device instead of normal block device names such as "/dev/sda1","/dev/sda2",...

Since Linux 2.6.30, the kernel defaults to the behavior provided by "relatime" option.

See fstab(5) and mount(8).

9.6.8 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.

9.6.9 Optimization of hard disk

⚠️ Warning
Please check your hardware and read manpage of hdparm(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”.

9.6.10 Optimization of solid state drive

Solid state drive (SSD) is auto detected now.

Reduce unnecessary disk accesses to prevent disk wear out by mounting ”tmpfs” on volatile data path in /etc/fstab.

9.6.11 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 systemctl restart smartmontools”.

Tip
The smartd(8) daemon can be customized with the /etc/smartd.conf file including how to be notified of warnings.

9.6.12 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.

9.6.13 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.
9.6.14 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` and `mount` it to a directory where you need more space. (You need to copy original data contents.)

```bash
$ 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 9.7.5) as a loop device (see Section 9.7.3). The actual disk usage grows with the actual data stored.

---

9.6.15 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.

```bash
$ sudo mount --bind /path/to/emp-dir work-dir
```

---

9.6.16 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).

```bash
$ 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".

---

9.6.17 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`.

```bash
$ 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.

---
## 9.7 The disk image

Here, we discuss manipulations of the disk image.

### 9.7.1 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.

```bash
# 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 9.6.2) which reside on the first sector on the primary IDE disk can be made by using `dd(1)` by the following.

```bash
# 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.

### 9.7.2 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.

```bash
# 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.

```bash
# dd if=partition.img of=/dev/sdb1
```

### 9.7.3 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.

```bash
# 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.
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.

```bash
# 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.

```bash
# 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   417800   83  Linux
```

```bash
# mkdir -p /mnt/loop0p1
# mount -t ext4 /dev/loop0p1 /mnt/loop0p1
# umount /dev/loop0p1
```

```
# mkdir -p /mnt/loop0p2
# mount -t ext4 /dev/loop0p2 /mnt/loop0p2
# umount /dev/loop0p2
```

Alternatively, similar effects can be done by using the device mapper devices created by kpartx(8) from the kpartx package as follows.

```bash
# kpartx -a -v disk.img
...# mkdir -p /mnt/loop0p2
# mount -t ext4 /dev/mapper/loop0p2 /mnt/loop0p2
...
# 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.

## 9.7.4 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.
# Building an Empty Disk Image

The empty disk image "disk.img" which can grow up to 5GiB can be made using `dd` as follows.

```bash
$ dd bs=1 count=0 if=/dev/zero of=disk.img seek=5G
```

Instead of using `dd`, specialized `fallocate` may be used here.

You can create an ext4 filesystem on this disk image "disk.img" using the `loop device` as follows.

```bash
# losetup -f -v disk.img
Loop device is /dev/loop1
# mkfs.ext4 /dev/loop1
...hack...hack...hack
$ 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 9.7.3, 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.

### 9.7.6 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.

```bash
# 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.
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.

```bash
$ 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=no_trunc,noerror | md5sum
# dd if=/dev/cdrom bs=2048 count=23150592 conv=no_trunc,noerror > cd.iso
```

**Warning**

You must carefully avoid ISO9660 filesystem read ahead bug of Linux as above to get the right result.

### 9.7.7 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.

```bash
# 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.

```bash
# wodim -v -eject dev=/dev/hda cd.iso
```

If CD-RW is used instead of CD-R, do this instead by the following.

```bash
# 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).

### 9.7.8 Mounting the ISO9660 image file

If "cd.iso" contains an ISO9660 image, then the following manually mounts it to "/cdrom".

```bash
# 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).
9.8 The binary data

Here, we discuss direct manipulations of the binary data on storage media.

9.8.1 Viewing and editing binary data

The most basic viewing method of binary data is to use "od -t x1" command.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>basic package which has od(1) to dump files (HEX, ASCII, OCTAL, …)</td>
</tr>
<tr>
<td>bsdmainutils</td>
<td>V:13, I:361</td>
<td>17</td>
<td>utility package which has hd(1) to dump files (HEX, ASCII, OCTAL, …)</td>
</tr>
<tr>
<td>hexedit</td>
<td>V:0, I:9</td>
<td>73</td>
<td>binary editor and viewer (HEX, ASCII)</td>
</tr>
<tr>
<td>bless</td>
<td>V:0, I:2</td>
<td>924</td>
<td>full featured hexadecimal editor (GNOME)</td>
</tr>
<tr>
<td>okteta</td>
<td>V:1, I:112</td>
<td>1581</td>
<td>full featured hexadecimal editor (KDE4)</td>
</tr>
<tr>
<td>ncurses-hexedit</td>
<td>V:0, I:1</td>
<td>130</td>
<td>binary editor and viewer (HEX, ASCII, EBCDIC)</td>
</tr>
<tr>
<td>beav</td>
<td>V:0, I:0</td>
<td>137</td>
<td>binary editor and viewer (HEX, ASCII, EBCDIC, OCTAL, …)</td>
</tr>
</tbody>
</table>

Table 9.21: List of packages which view and edit binary data

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.

9.8.2 Manipulating files without mounting disk

There are tools to read and write files without mounting disk.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mtools</td>
<td>V:8, I:164</td>
<td>390</td>
<td>utilities for MSDOS files without mounting them</td>
</tr>
<tr>
<td>hfsutils</td>
<td>V:0, I:4</td>
<td>184</td>
<td>utilities for HFS and HFS+ files without mounting them</td>
</tr>
</tbody>
</table>

Table 9.22: List of packages to manipulate files without mounting disk

9.8.3 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.

9.8.4 Data file recovery and forensic analysis

There are tools for data file recovery and forensic analysis.
### Table 9.23: List of tools to add data redundancy to files

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>par2</td>
<td>V:8,I:87</td>
<td>268</td>
<td>Parity Archive Volume Set, for checking and repair of files</td>
</tr>
<tr>
<td>dvdisaster</td>
<td>V:0,I:1</td>
<td>1422</td>
<td>data loss/scratch/aging protection for CD/DVD media</td>
</tr>
<tr>
<td>dvbackup</td>
<td>V:0,I:0</td>
<td>413</td>
<td>backup tool using MiniDV camcorders (providing rsbeep(1))</td>
</tr>
</tbody>
</table>

### Table 9.24: List of packages for data file recovery and forensic analysis

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>testdisk</td>
<td>V:2,I:29</td>
<td>1413</td>
<td>utilities for partition scan and disk recovery</td>
</tr>
<tr>
<td>magicrescue</td>
<td>V:0,I:2</td>
<td>255</td>
<td>utility to recover files by looking for magic bytes</td>
</tr>
<tr>
<td>scalpel</td>
<td>V:0,I:3</td>
<td>88</td>
<td>frugal, high performance file carver</td>
</tr>
<tr>
<td>myrescue</td>
<td>V:0,I:2</td>
<td>83</td>
<td>rescue data from damaged harddisks</td>
</tr>
<tr>
<td>extundelete</td>
<td>V:0,I:8</td>
<td>147</td>
<td>utility to undelete files on the ext3/4 filesystem</td>
</tr>
<tr>
<td>ext4magic</td>
<td>V:0,I:4</td>
<td>233</td>
<td>utility to undelete files on the ext3/4 filesystem</td>
</tr>
<tr>
<td>ext3grep</td>
<td>V:0,I:2</td>
<td>293</td>
<td>tool to help recover deleted files on the ext3 filesystem</td>
</tr>
<tr>
<td>scrounge-ntfs</td>
<td>V:0,I:2</td>
<td>50</td>
<td>data recovery program for NTFS filesystems</td>
</tr>
<tr>
<td>gzrt</td>
<td>V:0,I:0</td>
<td>33</td>
<td>gzip recovery toolkit</td>
</tr>
<tr>
<td>Sleuthkit</td>
<td>V:3,I:24</td>
<td>1611</td>
<td>tools for forensics analysis. (Sleuthkit)</td>
</tr>
<tr>
<td>autopsy</td>
<td>V:0,I:1</td>
<td>1027</td>
<td>graphical interface to SleuthKit</td>
</tr>
<tr>
<td>foremost</td>
<td>V:0,I:5</td>
<td>102</td>
<td>forensics application to recover data</td>
</tr>
<tr>
<td>guymager</td>
<td>V:0,I:1</td>
<td>1021</td>
<td>forensic imaging tool based on Qt</td>
</tr>
<tr>
<td>dcftdd</td>
<td>V:0,I:3</td>
<td>114</td>
<td>enhanced version of dd for forensics and security</td>
</tr>
</tbody>
</table>

### Tip
You can undelete files on the ext2 filesystem using `list_deleted_inodes` and `undel` commands of `debugfs(8)` in the `e2fsprogs` package.

#### 9.8.5 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.

```bash
$ 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.

#### 9.8.6 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.

```bash
$ :> file_to_be_cleared
```
9.8.7  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.

9.8.8  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
```

You may alternatively use `badblocks(8)` with `-t random` option.

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.

9.8.9  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.

### 9.8.10 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:

```bash
$ 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
$ 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 bar
foo
```

Execute on another terminal (when you have the `ls/of` package installed) as follows.

```bash
$ ls -li bar
2228329 -rw-r--r-- 1 bozo bozo 4 2008-05-11 11:02 bar
$ ls/of | grep bar | grep less
less 4775 bozo 4r REG 8,3 4 2228329 /home/bozo/bar
$ rm bar
$ ls/of | 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
```

### 9.8.11 Searching all hardlinks

Files with hardlinks can be identified by `"ls -li"`.

```bash
$ 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.

```bash
# find /path/to/mount/point -xdev -inum 2738404
```
9.8.12 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.

```
# lsolf -s -X / | grep deleted
```

9.9 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 4.6.4). This means that login password system cannot secure your private and sensitive data against possible theft of your PC. You must employ data encryption technology to do it. Although GNU privacy guard (see Section 10.3) can encrypt files, it takes some user efforts.

**Dm-crypt** facilitates automatic data encryption via native Linux kernel modules with minimal user efforts using *device-mapper*.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cryptsetup</td>
<td>V:35, I:78</td>
<td>410</td>
<td>utilities for encrypted block device (dm-crypt / LUKS)</td>
</tr>
<tr>
<td>cryptmount</td>
<td>V:2, I:3</td>
<td>231</td>
<td>utilities for encrypted block device (dm-crypt / LUKS) with focus on mount/unmount by normal users</td>
</tr>
<tr>
<td>fscrypt</td>
<td>V:0, I:0</td>
<td>4596</td>
<td>utilities for Linux filesystem encryption (fscrypt)</td>
</tr>
<tr>
<td>libpam-fscrypt</td>
<td>I:0</td>
<td>4181</td>
<td>PAM module for Linux filesystem encryption (fscrypt)</td>
</tr>
</tbody>
</table>

Table 9.25: List of data encryption utilities

---

⚠️ **Caution**

Data encryption costs CPU time etc. Encrypted data becomes inaccessible if its password is lost. Please weigh its benefits and costs.

---

**Note**

Entire Debian system can be installed on an 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.

9.9.1 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.

```
# fdisk /dev/sdx
... "n" "p" "1" "return" "return" "w"
# cryptsetup luksFormat /dev/sdx1
...
# cryptsetup open /dev/sdx1 secret
...
# ls -l /dev/mapper/
total 0
```
Then, it can be mounted just like normal one on to "/media/username/disk_label", except for asking password (see Section 10.1.7) under modern desktop environment using the udisks2 package. The difference is that every data written to it is encrypted. The password entry may be automated using keyring (see Section 10.3.6).

You may alternatively format media in different filesystem, e.g., ext4 with "mkfs.ext4 /dev/mapper/sdx1". If btrfs is used instead, the udisks2-btrfs package needs to be installed. For these filesystems, the file ownership and permissions may need to be configured.

9.9.2 Mounting encrypted disk with dm-crypt/LUKS

For example, an encrypted disk partition created with dm-crypt/LUKS on "/dev/sdc5" by Debian Installer 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
```

9.10 The kernel

Debian distributes modularized Linux kernel as packages for supported architectures.

If you are reading this documentation, you probably don’t need to compile Linux kernel by yourself.

9.10.1 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 9.7.3)

See "The Linux kernel user’s and administrator’s guide » The kernel’s command-line parameters” for the detail.
### 9.10.2 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.

### 9.10.3 Compiling the kernel and related modules

Debian has its own method of compiling the kernel and related modules.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>build-essential</td>
<td>I:479</td>
<td>17</td>
<td>essential packages for building Debian packages: make, gcc, …</td>
</tr>
<tr>
<td>bzip2</td>
<td>V:168, I:970</td>
<td>121</td>
<td>compress and decompress utilities for bzip2 files</td>
</tr>
<tr>
<td>libncurses5-dev</td>
<td>I:75</td>
<td>6</td>
<td>developer’s libraries and docs for curses</td>
</tr>
<tr>
<td>git</td>
<td>V:341, I:534</td>
<td>46517</td>
<td>git: distributed revision control system used by the Linux kernel</td>
</tr>
<tr>
<td>fakeroot</td>
<td>V:31, I:487</td>
<td>208</td>
<td>provide fakeroot environment for building package as non-root</td>
</tr>
<tr>
<td>initramfs-tools</td>
<td>V:427, I:989</td>
<td>113</td>
<td>tool to build an initramfs (Debian specific)</td>
</tr>
<tr>
<td>dkms</td>
<td>V:76, I:164</td>
<td>189</td>
<td>dynamic kernel module support (DKMS) (generic)</td>
</tr>
<tr>
<td>module-assistant</td>
<td>V:1, I:21</td>
<td>406</td>
<td>helper tool to make module package (Debian specific)</td>
</tr>
<tr>
<td>devscripts</td>
<td>V:6, I:41</td>
<td>2643</td>
<td>helper scripts for a Debian Package maintainer (Debian specific)</td>
</tr>
</tbody>
</table>

Table 9.26: 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.

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.

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.
9.10.4 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
$ 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_features_flavour" 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

9.10.5 Hardware drivers and firmware

The hardware driver is the code running on the main CPUs of 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, and non-free for binary-only drivers supported by the vendor.)

The firmware is the code or data loaded on the device attach to the target system (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. Installing these firmware data is essential for the device to function as expected.

- The firmware data packages containing data loaded to the volatile memory on the target device.
  - firmware-linux-free (main)
  - firmware-linux-nonfree (non-free-firmware)
– firmware-linux-* (non-free-firmware)
– *-firmware (non-free-firmware)
– intel-microcode (non-free-firmware)
– amd64-microcode (non-free-firmware)

• The firmware update program packages which update data on the non-volatile memory on the target device.
  – fwupd (main): Firmware update daemon which downloads firmware data from Linux Vendor Firmware Service.
  – gnome-firmware (main): GTK front end for fwupd
  – plasma-discover-backend-fwupd (main): Qt front end for fwupd

Please note that access to non-free-firmware packages are provided by the official installation media to offer functional installation experience to the user since Debian 12 Bookworm. The non-free-firmware area is described in Section 2.1.5.

Please also note that the firmware data downloaded by fwupd from Linux Vendor Firmware Service and loaded to the running Linux kernel may be non-free.

### 9.11 Virtualized system

Use of virtualized system enables us to run multiple instances of system simultaneously on a single hardware.

**Tip**

### 9.11.1 Virtualization and emulation tools

There are several virtualization and emulation tool platforms.

• Complete hardware emulation packages such as ones installed by the games-emulator metapackage
• Mostly CPU level emulation with some I/O device emulations such as QEMU
• Mostly CPU level virtualization with some I/O device emulations such as Kernel-based Virtual Machine (KVM)
• OS level container virtualization with the kernel level support such as LXC (Linux Containers), Docker, systemd-nspawn(1), ...
• OS level filesystem access virtualization with the system library call override on the file path such as chroot
• OS level filesystem access virtualization with the system library call override on the file ownership such as fakeroot
• OS API emulation such as Wine
• Interpreter level virtualization with its executable selection and run-time library overrides such as virtualenv and venv for Python

The container virtualization uses Section 4.7.4 and is the backend technology of Section 7.6.

Here are some packages to help you to setup the virtualized system.

See Wikipedia article [Comparison of platform virtual machines](http://en.wikipedia.org/wiki/Comparison_of_platform_virtual_machines) for detail comparison of different platform virtualization solutions.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>GNU core utilities which contain chroot(8)</td>
</tr>
<tr>
<td>systemd-container</td>
<td>V:50, I:57</td>
<td>1320</td>
<td>systemd container/nsspawn tools which contain systemd-nspawn(1)</td>
</tr>
<tr>
<td>schroot</td>
<td>V:5, I:7</td>
<td>2508</td>
<td>specialized tool for executing Debian binary packages in chroot</td>
</tr>
<tr>
<td>sbuild</td>
<td>V:1, I:3</td>
<td>242</td>
<td>tool for building Debian binary packages from Debian sources</td>
</tr>
<tr>
<td>debootstrap</td>
<td>V:3, I:54</td>
<td>309</td>
<td>bootstrap a basic Debian system (written in sh)</td>
</tr>
<tr>
<td>cdebootstrap</td>
<td>V:0, I:1</td>
<td>115</td>
<td>bootstrap a Debian system (written in C)</td>
</tr>
<tr>
<td>cloud-image-utils</td>
<td>V:0, I:16</td>
<td>66</td>
<td>cloud image management utilities</td>
</tr>
<tr>
<td>cloud-guest-utils</td>
<td>V:3, I:12</td>
<td>71</td>
<td>cloud guest utilities</td>
</tr>
<tr>
<td>virt-manager</td>
<td>V:11, I:43</td>
<td>2296</td>
<td>Virtual Machine Manager: desktop application for managing virtual machines</td>
</tr>
<tr>
<td>libvirt-clients</td>
<td>V:46, I:64</td>
<td>1237</td>
<td>programs for the libvirt library</td>
</tr>
<tr>
<td>lxd</td>
<td>V:0, I:10</td>
<td>49313</td>
<td>LXD: system container and virtual machine manager</td>
</tr>
<tr>
<td>podman</td>
<td>V:12, I:14</td>
<td>41407</td>
<td>podman: engine to run OCI-based containers in Pods</td>
</tr>
<tr>
<td>podman-docker</td>
<td>V:0, I:0</td>
<td>249</td>
<td>engine to run OCI-based containers in Pods - wrapper for docker</td>
</tr>
<tr>
<td>docker.io</td>
<td>V:39, I:41</td>
<td>149302</td>
<td>docker: Linux container runtime</td>
</tr>
<tr>
<td>games-emulator</td>
<td>I:0</td>
<td>21</td>
<td>games-emulator: Debian's emulators for games</td>
</tr>
<tr>
<td>bochs</td>
<td>V:0, I:0</td>
<td>6956</td>
<td>Bochs: IA-32 PC emulator</td>
</tr>
<tr>
<td>qemu</td>
<td>I:17</td>
<td>97</td>
<td>QEMU: fast generic processor emulator</td>
</tr>
<tr>
<td>qemu-system</td>
<td>I:21</td>
<td>67</td>
<td>QEMU: full system emulation binaries</td>
</tr>
<tr>
<td>qemu-user</td>
<td>V:1, I:6</td>
<td>93198</td>
<td>QEMU: user mode emulation binaries</td>
</tr>
<tr>
<td>qemu-utils</td>
<td>V:12, I:105</td>
<td>10503</td>
<td>QEMU: utilities</td>
</tr>
<tr>
<td>qemu-system-x86</td>
<td>V:31, I:90</td>
<td>46165</td>
<td>KVM: full virtualization on x86 hardware with the hardware-assisted virtualization</td>
</tr>
<tr>
<td>virtualbox</td>
<td>V:6, I:8</td>
<td>131166</td>
<td>VirtualBox: x86 virtualization solution on i386 and amd64</td>
</tr>
<tr>
<td>gnome-boxes</td>
<td>V:1, I:7</td>
<td>6691</td>
<td>Boxes: Simple GNOME app to access virtual systems</td>
</tr>
<tr>
<td>xen-tools</td>
<td>V:0, I:2</td>
<td>719</td>
<td>tools to manage debian XEN virtual server</td>
</tr>
<tr>
<td>wine</td>
<td>V:14, I:60</td>
<td>135</td>
<td>Wine: Windows API Implementation (standard suite)</td>
</tr>
<tr>
<td>dosbox</td>
<td>V:1, I:15</td>
<td>2671</td>
<td>DOSBox: x86 emulator with Tandy/Herc/CGA/EGA/VGA/SVGA graphics, sound and DOS</td>
</tr>
<tr>
<td>lxctools</td>
<td>V:9, I:12</td>
<td>25866</td>
<td>Linux containers user space tools</td>
</tr>
<tr>
<td>python3-venv</td>
<td>I:79</td>
<td>6</td>
<td>venv for creating virtual python environments (system library)</td>
</tr>
<tr>
<td>python3-virtualenv</td>
<td>V:9, I:50</td>
<td>364</td>
<td>virtualenv for creating isolated virtual python environments</td>
</tr>
<tr>
<td>pipx</td>
<td>V:2, I:13</td>
<td>928</td>
<td>pipx for installing python applications in isolated environments</td>
</tr>
</tbody>
</table>

Table 9.27: List of virtualization tools
9.11.2 Virtualization work flow

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` (see Section 9.7.1 and Section 9.7.5).
  - `qemu-img` can be used to create and convert disk image files supported by QEMU.
  - The raw and VMDK file format can be used as a 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 9.7.3).
  - For disk images supported by QEMU, mount them as `network block device` (see Section 9.11.3).

- Populate the target filesystem with required system data.
  - The use of programs such as `debootstrap` and `cdebootstrap` helps with this process (see Section 9.11.4).
  - 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.

9.11.3 Mounting the virtual disk image file

For the raw disk image file, see Section 9.7.

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 9.7.3). 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)`.
9.11.4 Chroot system

If you wish to try a new Debian environment from a terminal console, I recommend you to use chroot. This enables you to run console applications of Debian unstable and testing without usual risks associated and without rebooting. chroot(8) is the most basic way.

Caution

Examples below assumes both parent system and chroot system share the same amd64 CPU architecture.

Although you can manually create a chroot(8) environment using debootstrap(1), this requires non-trivial efforts.

The sbuild package to build Debian packages from source uses the chroot environment managed by the schroot package. It comes with helper script sbuild-createchroot(1). Let's learn how it works by running it as follows.

```
$ sudo mkdir -p /srv/chroot
$ sudo sbuild-createchroot -v --include=eatmydata,ccache unstable /srv/chroot/unstable- amd64-sbuild http://deb.debian.org/debian
...
```

You see how debootstrap(8) populates system data for unstable environment under”/srv/chroot/unstable-amd64-sbuild” for a minimal build system.

You can login to this environment using schroot(1).

```
$ sudo schroot -v -c chroot:unstable-amd64-sbuild
```

You see how a system shell running under unstable environment is created.

Note

The ”/usr/sbin/policy-rc.d” file which always exits with 101 prevents daemon programs to be started automatically on the Debian system. See ”/usr/share/doc/sysv-rc/README.policy-rc.d.gz”.

Note

Some programs under chroot may require access to more files from the parent system to function than sbuild-createchroot provides as above. For example, ”/sys”, ”/etc/passwd”, ”/etc/group”, ”/var/run/utmp”, ”/var/log/wtmp”, etc. may need to be bind-mounted or copied.

Tip

The sbuild package helps to construct a chroot system and builds a package inside the chroot using schroot as its backend. It is an ideal system to check build-dependencies. See more on sbuild at Debian wiki and sbuild configuration example in ”Guide for Debian Maintainers”.

Tip

The systemd-nspawn(1) command helps to run a command or OS in a light-weight container in similar ways to chroot. It is more powerful since it uses namespaces to fully virtualize the the process tree, IPC, hostname, domain name and, optionally, networking and user databases. See systemd-nspawn.
9.11.5 Multiple desktop systems

If you wish to try a new GUI Desktop environment of any OS, I recommend you to use QEMU or KVM on a Debian stable system to run multiple desktop systems safely using virtualization. These enable you to run any desktop applications including ones of Debian unstable and testing without usual risks associated with them and without rebooting.

Since pure QEMU is very slow, it is recommended to accelerate it with KVM when the host system supports it.

Virtual Machine Manager also known as virt-manager is a convenient GUI tool for managing KVM virtual machines via libvirt.

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 ...
```

**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.

See more tips at Debian wiki: SystemVirtualization.
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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>extension</th>
<th>command</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>tar</td>
<td>V:909,I:999</td>
<td>3144</td>
<td>.tar</td>
<td>tar(1)</td>
<td>the standard archiver (de facto standard)</td>
</tr>
<tr>
<td>cpio</td>
<td>V:392,I:998</td>
<td>1141</td>
<td>.cpio</td>
<td>cpio(1)</td>
<td>Unix System V style archiver, use with find(1)</td>
</tr>
<tr>
<td>binutils</td>
<td>V:173,I:629</td>
<td>143</td>
<td>.ar</td>
<td>ar(1)</td>
<td>archiver for the creation of static libraries</td>
</tr>
<tr>
<td>fastjar</td>
<td>V:1,I:14</td>
<td>183</td>
<td>.jar</td>
<td>fastjar(1)</td>
<td>archiver for Java (zip like)</td>
</tr>
<tr>
<td>pax</td>
<td>V:8,I:15</td>
<td>170</td>
<td>.pax</td>
<td>pax(1)</td>
<td>new POSIX standard archiver, compromise between tar and cpio</td>
</tr>
<tr>
<td>gzip</td>
<td>V:874,I:999</td>
<td>252</td>
<td>.gz</td>
<td>gzip(1), zcat(1),...</td>
<td>GNU LZ77 compression utility (de facto standard)</td>
</tr>
<tr>
<td>bzip2</td>
<td>V:168,I:970</td>
<td>121</td>
<td>.bz2</td>
<td>bzip2(1), bzcat(1),...</td>
<td>Burrows-Wheeler block-sorting compression utility with higher compression ratio than gzip(1) (slower than gzip with similar syntax)</td>
</tr>
<tr>
<td>lzma</td>
<td>V:1,I:17</td>
<td>149</td>
<td>.lzma</td>
<td>lzma(1)</td>
<td>LZMA compression utility with higher compression ratio than gzip(1) (deprecated)</td>
</tr>
<tr>
<td>xz-utils</td>
<td>V:382,I:980</td>
<td>1248</td>
<td>.xz</td>
<td>xz(1), xzdec(1),...</td>
<td>XZ compression utility with higher compression ratio than bzip2(1) (slower than gzip but faster than bzip2; replacement for LZMA compression utility)</td>
</tr>
<tr>
<td>zstd</td>
<td>V:157,I:385</td>
<td>2158</td>
<td>.zstd</td>
<td>zstd(1), zstdcat(1),...</td>
<td>Zstandard fast lossless compression utility</td>
</tr>
<tr>
<td>p7zip</td>
<td>V:122,I:479</td>
<td>987</td>
<td>.7z</td>
<td>7zr(1), p7zip(1)</td>
<td>7-Zip file archiver with high compression ratio (LZMA compression)</td>
</tr>
<tr>
<td>p7zip-full</td>
<td>V:125,I:481</td>
<td>4664</td>
<td>.7z</td>
<td>7z(1), 7za(1)</td>
<td>7-Zip file archiver with high compression ratio (LZMA compression and others)</td>
</tr>
<tr>
<td>lzop</td>
<td>V:15,I:140</td>
<td>164</td>
<td>.lzo</td>
<td>lzop(1)</td>
<td>LZO compression utility with higher compression and decompression speed than gzip(1) (lower compression ratio than gzip with similar syntax)</td>
</tr>
<tr>
<td>zip</td>
<td>V:47,I:383</td>
<td>616</td>
<td>.zip</td>
<td>zip(1)</td>
<td>InfoZIP: DOS archive and compression tool</td>
</tr>
<tr>
<td>unzip</td>
<td>V:105,I:774</td>
<td>379</td>
<td>.zip</td>
<td>unzip(1)</td>
<td>InfoZIP: DOS unarchive and decompression tool</td>
</tr>
</tbody>
</table>

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.

- The gzipped tar(1) archive uses the file extension ".tgz" or ".tar.gz".
- The xz-compressed tar(1) archive uses the file extension ".txz" or ".tar.xz".


• Popular compression method in FOSS tools such as tar(1) has been moving as follows: 
gzip → bzip2 → xz

• cp(1), scp(1) and tar(1) may have some limitation for special files. cpio(1) is most versatile.

• 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.

• Internal structure of Libreoffice data files are ".jar" file which can be opened also by unzip.

• 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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>tool</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>GNU cp</td>
<td>locally copy files and directories (&quot;-a&quot; for recursive)</td>
</tr>
<tr>
<td>openssh-client</td>
<td>V:854, I:995</td>
<td>5814</td>
<td>scp</td>
<td>remotely copy files and directories (client, &quot;-r&quot; for recursive)</td>
</tr>
<tr>
<td>openssh-server</td>
<td>V:721, I:813</td>
<td>1948</td>
<td>sshd</td>
<td>remotely copy files and directories (remote server)</td>
</tr>
<tr>
<td>rsync</td>
<td>V:266, I:351</td>
<td>776</td>
<td></td>
<td>1-way remote synchronization and backup</td>
</tr>
<tr>
<td>unison</td>
<td>V:3, I:15</td>
<td>14</td>
<td></td>
<td>2-way remote synchronization and backup</td>
</tr>
</tbody>
</table>

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
Version control system (VCS) tools in Table 10.14 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):

```bash
$ tar -cvJf archive.tar.xz ./source
$ tar -xvJf archive.tar.xz
```

Alternatively, by the following.

```bash
$ find ./source -xdev -print0 | tar -cvJf archive.tar.xz --null -T -
```

cpio(1):

```bash
$ 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 -aHAXSv --files-from=- . /dest
# cd ./source; find . -print0 | rsync -aHAXSv --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 xvf - )
# (cd ./source && tar cf -. ) | ssh user@host.dom 'cd /dest && tar xvf - ')
```

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 9.4.9). 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 preceded with "+" for larger, preceded 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 \n   -xdev -regextype posix-extended \n   -type f -regex ".*\.cpio|.~" -prune -o \n   -type d -regex ".*\/.git" -prune -o \n   -type f -size +99M -prune -o \n   -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.

---

**Tip**

If you use re-writable media for your backups, use of filesystem such as `btrfs` or `zfs` which supports read-only snapshots may be a good idea.

---

### 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.
Modern desktop environments such as GNOME and KDE can mount these removable devices automatically without a matching "/etc/fstab" entry.

- **udisks2** 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 "helper=mount" 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/username/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 8.1.3).

**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.

### 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 9.9.1 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.
### Table 10.3: List of filesystem choices for removable storage devices with typical usage scenarios

<table>
<thead>
<tr>
<th>filesystem name</th>
<th>typical usage scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>FAT12</td>
<td>cross platform sharing of data on the floppy disk (&lt;32MiB)</td>
</tr>
<tr>
<td>FAT16</td>
<td>cross platform sharing of data on the small hard disk like device (&lt;2GiB)</td>
</tr>
<tr>
<td>FAT32</td>
<td>cross platform sharing of data on the large hard disk like device (&lt;8TiB, supported by newer than MS Windows95 OSR2)</td>
</tr>
<tr>
<td>exFAT</td>
<td>cross platform sharing of data on the large hard disk like device (&lt;512TiB, supported by WindowsXP, Mac OS X Snow Leopard 10.6.5, and Linux kernels since 5.4 release)</td>
</tr>
<tr>
<td>NTFS</td>
<td>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)</td>
</tr>
<tr>
<td>ISO96600</td>
<td>cross platform sharing of static data on CD-R and DVD +/- R</td>
</tr>
<tr>
<td>UDF</td>
<td>incremental data writing on CD-R and DVD +/-R (new)</td>
</tr>
<tr>
<td>MINIX</td>
<td>space efficient unix file data storage on the floppy disk</td>
</tr>
<tr>
<td>ext2</td>
<td>sharing of data on the hard disk like device with older Linux systems</td>
</tr>
<tr>
<td>ext3</td>
<td>sharing of data on the hard disk like device with older Linux systems</td>
</tr>
<tr>
<td>ext4</td>
<td>sharing of data on the hard disk like device with current Linux systems</td>
</tr>
<tr>
<td>btrfs</td>
<td>sharing of data on the hard disk like device with current Linux systems with read-only snapshots</td>
</tr>
</tbody>
</table>

- Partitioning them with `fdisk(8)`, `cfdisk(8)` or `parted(8)` (see Section 9.6.2) 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.

<table>
<thead>
<tr>
<th>network service</th>
<th>description of typical usage scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMB/CIFS network mounted filesystem with Samba</td>
<td>sharing files via &quot;Microsoft Windows Network&quot;, see smb.conf(5) and The Official Samba 3.x.x HOWTO and Reference Guide or the samba-doc package</td>
</tr>
<tr>
<td>NFS network mounted filesystem with the Linux kernel</td>
<td>sharing files via &quot;Unix/Linux Network&quot;, see exports(5) and Linux NFS-HOWTO</td>
</tr>
<tr>
<td>HTTP service</td>
<td>sharing file between the web server/client</td>
</tr>
<tr>
<td>HTTPS service</td>
<td>sharing file between the web server/client with encrypted Secure Sockets Layer (SSL) or Transport Layer Security (TLS)</td>
</tr>
<tr>
<td>FTP service</td>
<td>sharing file between the FTP server/client</td>
</tr>
</tbody>
</table>

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 6.5 and Section 6.6.

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.

10.2.1 Backup and recovery policy

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 programs: 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
  – Cron job as a user process: files in "/var/spool/cron/crontabs" directory and restart cron(8). See Section 9.4.14 for cron(8) and crontab(1).
  – Systemd timer jobs as user processes: files in "~/.config/systemd/user" directory. See systemd.timer(5) and systemd.service(5).
  – Autostart jobs as user processes: files in "~/.config/autostart" directory. See Desktop Application Autostart Specification.

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.
• Risk of data when lost
  – 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 read-only filesystem. 1
• Risk of data when breached
  – Sensitive identity data such as "/etc/ssh/ssh_host_*_key", "/.gnupg/*", "/.ssh/*", "/.local/share/
  "/etc/passwd", "/etc/shadow", "popularity-contest.conf", "/etc/ppp/pap-secrets", and "/etc/ should be backed up as encrypted. 2 (See Section 9.9.)
  – Never hard code system login password nor decryption passphrase in any script even on any trusted system. (See Section 10.3.6.)
• Failure mode and their possibility
  – Hardware (especially HDD) will break
  – Filesystem may be corrupted and data in it may be lost
  – Remote storage system can’t be trusted for security breaches
  – Weak password protection can be easily compromised
  – File permission system may be compromised
• Required resources for backup: human, hardware, software, …
  – Automatic scheduled backup with cron job or systemd timer job

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.

Note
You may wish to stop some application daemons such as MTA (see Section 6.2.4) while backing up data.

10.2.2 Backup utility suites

Here is a select list of notable backup utility suites available on the Debian system.

Backup tools have their specialized focuses.

1A write-once media such as CD/DVD-R can prevent overwrite accidents. (See Section 9.8 for how to write to the storage media from the shell command line. GNOMEdesktop GUI environment gives you easy access via menu: "Places → CD/DVD Creator".)
2Some of these data cannot be regenerated by entering the same input string to the system.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dump</td>
<td>V:0,1:5</td>
<td>351</td>
<td>4.4 BSD dump(8) and restore(8) for ext2/ext3/ext4 filesystems</td>
</tr>
<tr>
<td>xfsdump</td>
<td>V:0,1:7</td>
<td>848</td>
<td>dump and restore with xfsdump(8) and xfsrestore(8) for XFS filesystem on GNU/Linux and IRIX</td>
</tr>
<tr>
<td>backupninja</td>
<td>V:2,1:3</td>
<td>360</td>
<td>lightweight, extensible meta-backup system</td>
</tr>
<tr>
<td>bacula-common</td>
<td>V:9,1:10</td>
<td>2119</td>
<td>Bacula: network backup, recovery and verification - common support files</td>
</tr>
<tr>
<td>bacula-client</td>
<td>V:0,1:2</td>
<td>154</td>
<td>Bacula: network backup, recovery and verification - client meta-package</td>
</tr>
<tr>
<td>bacula-console</td>
<td>V:0,1:3</td>
<td>104</td>
<td>Bacula: network backup, recovery and verification - text console</td>
</tr>
<tr>
<td>bacula-server</td>
<td>I:0</td>
<td>154</td>
<td>Bacula: network backup, recovery and verification - server meta-package</td>
</tr>
<tr>
<td>amanda-common</td>
<td>V:0,1:2</td>
<td>9935</td>
<td>Amanda: Advanced Maryland Automatic Network Disk Archiver (Libs)</td>
</tr>
<tr>
<td>amanda-client</td>
<td>V:0,1:2</td>
<td>1084</td>
<td>Amanda: Advanced Maryland Automatic Network Disk Archiver (Client)</td>
</tr>
<tr>
<td>amanda-server</td>
<td>V:0,1:0</td>
<td>1076</td>
<td>Amanda: Advanced Maryland Automatic Network Disk Archiver (Server)</td>
</tr>
<tr>
<td>backup-manager</td>
<td>V:0,1:1</td>
<td>566</td>
<td>command-line backup tool</td>
</tr>
<tr>
<td>backup2l</td>
<td>V:0,1:0</td>
<td>115</td>
<td>low-maintenance backup/restore tool for mountable media (disk based)</td>
</tr>
<tr>
<td>backuppc</td>
<td>V:2,1:2</td>
<td>3178</td>
<td>BackupPC: is a high-performance, enterprise-grade system for backing up PCs (disk based)</td>
</tr>
<tr>
<td>duplicity</td>
<td>V:29,1:48</td>
<td>1874</td>
<td>(remote) incremental backup</td>
</tr>
<tr>
<td>flexbackup</td>
<td>V:0,1:0</td>
<td>243</td>
<td>(remote) incremental backup</td>
</tr>
<tr>
<td>rdiff-backup</td>
<td>V:4,1:10</td>
<td>1162</td>
<td>(remote) incremental backup</td>
</tr>
<tr>
<td>restic</td>
<td>V:2,1:5</td>
<td>21812</td>
<td>(remote) incremental backup</td>
</tr>
<tr>
<td>slbackup</td>
<td>V:0,1:0</td>
<td>151</td>
<td>(remote) incremental backup</td>
</tr>
</tbody>
</table>

Table 10.5: List of backup suite utilities
• 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.

• Bacula, Amanda, and BackupPC are full featured backup suite utilities which are focused on regular backups over network.

• Regular backups of user data can be realized by a simple script (Section 10.2.3).

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.

10.2.3 Personal backup

For a personal Debian desktop system running testing 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.

At the same time, it is very valuable to have frequent recent snapshots of personal data and system configuration, and occasional full backups of personal data.

I usually make these snapshots and backups with a simple shell script bss. This script is a short shell which uses standard utilities: btrfs subvolume snapshot, rsync. For data encryption, disk image is created by fallocate(1) and configured with cryptsetup(8).

Tip
You can recover debconf configuration data with "debconf-set-selections debconf-selections" and dpkg selection data with "dpkg --set-selection <dpkg-selections.list".

10.3 Data security infrastructure

The data security infrastructure is provided by the combination of data encryption tool, message digest tool, and signature tool. See Section 9.9 on dm-crypt and fscrypt 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".

```bash
$ 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.

```bash
keyserver hkp://keys.gnupg.net
```
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gnupg</td>
<td>V:555, I:911</td>
<td>885</td>
<td>gpg(1)</td>
<td>GNU Privacy Guard - OpenPGP encryption and signing tool</td>
</tr>
<tr>
<td>gpgv</td>
<td>V:891, I:999</td>
<td>917</td>
<td>gpgv(t)</td>
<td>GNU Privacy Guard - signature verification tool</td>
</tr>
<tr>
<td>paperkey</td>
<td>V:1, I:13</td>
<td>58</td>
<td>paperkey(t)</td>
<td>extract just the secret information out of OpenPGP secret keys</td>
</tr>
<tr>
<td>cryptsetup</td>
<td>V:35, I:78</td>
<td>410</td>
<td>cryptsetup</td>
<td>utilities for dm-crypt block device encryption supporting LUKS</td>
</tr>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>md5sum(1)</td>
<td>compute and check MD5 message digest</td>
</tr>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>sha1sum(1)</td>
<td>compute and check SHA1 message digest</td>
</tr>
<tr>
<td>openssl</td>
<td>V:823, I:995</td>
<td>2290</td>
<td>openssl(lss)</td>
<td>compute message digest with &quot;openssl dgst&quot; (OpenSSL)</td>
</tr>
<tr>
<td>libsecret-tools</td>
<td>V:1, I:12</td>
<td>41</td>
<td>secret-tool</td>
<td>store and retrieve passwords (CLI)</td>
</tr>
<tr>
<td>seahorse</td>
<td>V:80, I:270</td>
<td>7987</td>
<td>seahorse(t)</td>
<td>key management tool (GNOME)</td>
</tr>
</tbody>
</table>

Table 10.6: List of data security infrastructure tools

<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gpg --gen-key</td>
<td>generate a new key</td>
</tr>
<tr>
<td>gpg --gen-revoke my_user_ID</td>
<td>generate revoke key for my_user_ID</td>
</tr>
<tr>
<td>gpg --edit-key user_ID</td>
<td>edit key interactively, &quot;help&quot; for help</td>
</tr>
<tr>
<td>gpg -o file --export</td>
<td>export all keys to file</td>
</tr>
<tr>
<td>gpg --import file</td>
<td>import all keys from file</td>
</tr>
<tr>
<td>gpg --send-keys user_ID</td>
<td>send key of user_ID to keys server</td>
</tr>
<tr>
<td>gpg --recv-keys user_ID</td>
<td>recv. key of user_ID from keys server</td>
</tr>
<tr>
<td>gpg --list-keys user_ID</td>
<td>list keys of user_ID</td>
</tr>
<tr>
<td>gpg --list-sigs user_ID</td>
<td>list sig. of user_ID</td>
</tr>
<tr>
<td>gpg --check-sigs user_ID</td>
<td>check sig. of user_ID</td>
</tr>
<tr>
<td>gpg --fingerprint user_ID</td>
<td>check fingerprint of user_ID</td>
</tr>
<tr>
<td>gpg --refresh-keys</td>
<td>update local keyring</td>
</tr>
</tbody>
</table>

Table 10.7: List of GNU Privacy Guard commands for the key management

<table>
<thead>
<tr>
<th>code</th>
<th>description of trust</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>no owner trust assigned / not yet calculated</td>
</tr>
<tr>
<td>e</td>
<td>trust calculation failed</td>
</tr>
<tr>
<td>q</td>
<td>not enough information for calculation</td>
</tr>
<tr>
<td>n</td>
<td>never trust this key</td>
</tr>
<tr>
<td>m</td>
<td>marginally trusted</td>
</tr>
<tr>
<td>f</td>
<td>fully trusted</td>
</tr>
<tr>
<td>u</td>
<td>ultimately trusted</td>
</tr>
</tbody>
</table>

Table 10.8: List of the meaning of the trust code
The following obtains unknown keys from the keys server.

```bash
$ gpg --list-sigs --with-colons | grep '^\{User ID not found\}' | \n   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.

<table>
<thead>
<tr>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gpg -a -s file</td>
<td>sign file into ASCII armored file.asc</td>
</tr>
<tr>
<td>gpg --armor --sign file</td>
<td>,,</td>
</tr>
<tr>
<td>gpg --clearsign file</td>
<td>clear-sign message</td>
</tr>
<tr>
<td>gpg --clearsign file@mail <a href="mailto:foo@example.org">foo@example.org</a></td>
<td>mail a clear-signed message to <a href="mailto:foo@example.org">foo@example.org</a></td>
</tr>
<tr>
<td>gpg --clearsign --not-dash-escaped patchfile</td>
<td>clear-sign patchfile</td>
</tr>
<tr>
<td>gpg --verify file</td>
<td>verify clear-signed file</td>
</tr>
<tr>
<td>gpg -o file.sig -b file</td>
<td>create detached signature</td>
</tr>
<tr>
<td>gpg -o file.sig --detach-sign file</td>
<td>,,</td>
</tr>
<tr>
<td>gpg --verify file.sig file</td>
<td>verify file with file.sig</td>
</tr>
<tr>
<td>gpg -o crypt_file.gpg -r name -e file</td>
<td>public-key encryption intended for name from file to binary crypt_file.gpg</td>
</tr>
<tr>
<td>gpg -o crypt_file.gpg --recipient name --encrypt file</td>
<td>,,</td>
</tr>
<tr>
<td>gpg -o crypt_file.asc -a -r name -e file</td>
<td>public-key encryption intended for name from file to ASCII armored crypt_file.asc</td>
</tr>
<tr>
<td>gpg -o crypt_file.gpg -c file</td>
<td>symmetric encryption from file to crypt_file.gpg</td>
</tr>
<tr>
<td>gpg -o crypt_file.gpg --symmetric file</td>
<td>,,</td>
</tr>
<tr>
<td>gpg -o crypt_file.asc -a -c file</td>
<td>symmetric encryption intended for name from file to ASCII armored crypt_file.asc</td>
</tr>
<tr>
<td>gpg -o file -d crypt_file.gpg -r name</td>
<td>decryption</td>
</tr>
<tr>
<td>gpg -o file --decrypt crypt_file.gpg</td>
<td>,,</td>
</tr>
</tbody>
</table>

Table 10.9: List of 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.

```bash
macro index S ":toggle pgp_verify_sig\n"
set pgp_verify_sig=no
```
10.3.4 Using GnuPG with Vim

The gnupg pluginlet you run GnuPG transparently for files with extension ".gpg", ".asc", and ".pgp".3

```
$ sudo aptitude install vim-scripts
$ echo "packadd! gnupg" >> ~/.vim/vimrc
```

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.

10.3.6 Password keyring

On GNOME system, the GUI tool seahorse(1) manages passwords and stores them securely in the keyring ~/.local/share/keyrings/*.

secret-tool(1) can store password to the keyring from the command line.

Let’s store passphrase used for LUKS/dm-crypt encrypted disk image

```
$ secret-tool store --label='LUKS passphrase for disk.img' LUKS my_disk.img
Password: ********
```

This stored password can be retrieved and fed to other programs, e.g., cryptsetup(8).

```
$ secret-tool lookup LUKS my_disk.img | \ 
cryptsetup open disk.img disk_img --type luks --keyring -
$ sudo mount /dev/mapper/disk_img /mnt
```

**Tip**
Whenever you need to provide password in a script, use secret-tool and avoid directly hardcoding the passphrase in it.

10.4 Source code merge tools

There are many merge tools for the source code. Following commands caught my eyes.

---

3If you use "~/.vimrc" instead of "~/.vim/vimrc", please substitute accordingly.
Table 10.10: List of source code merge tools

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>patch</strong></td>
<td>V:98, I:695</td>
<td>248</td>
<td>patch(1)</td>
<td>apply a diff file to an original</td>
</tr>
<tr>
<td><strong>vim</strong></td>
<td>V:95, I:370</td>
<td>3738</td>
<td>vimdiff(1)</td>
<td>compare 2 files side by side in vim</td>
</tr>
<tr>
<td><strong>imediff</strong></td>
<td>V:0, I:0</td>
<td>169</td>
<td>imediff(1)</td>
<td>interactive full screen 2/3-way merge tool</td>
</tr>
<tr>
<td><strong>meld</strong></td>
<td>V:8, I:30</td>
<td>3500</td>
<td>meld(1)</td>
<td>compare and merge files (GTK)</td>
</tr>
<tr>
<td><strong>wiggly</strong></td>
<td>V:0, I:0</td>
<td>176</td>
<td>wiggly(1)</td>
<td>apply rejected patches</td>
</tr>
<tr>
<td><strong>diffutils</strong></td>
<td>V:861, I:996</td>
<td>1735</td>
<td>diff(1)</td>
<td>compare files line by line</td>
</tr>
<tr>
<td><strong>diffutils</strong></td>
<td>V:861, I:996</td>
<td>1735</td>
<td>diff3(1)</td>
<td>compare and merges three files line by line</td>
</tr>
<tr>
<td><strong>quilt</strong></td>
<td>V:2, I:23</td>
<td>871</td>
<td>quilt(1)</td>
<td>manage series of patches</td>
</tr>
<tr>
<td><strong>wdiff</strong></td>
<td>V:7, I:52</td>
<td>648</td>
<td>wdiff(1)</td>
<td>display word differences between text files</td>
</tr>
<tr>
<td><strong>diffstat</strong></td>
<td>V:13, I:123</td>
<td>74</td>
<td>diffstat(1)</td>
<td>produce a histogram of changes by the diff</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>combinediff(1)</td>
<td>create a cumulative patch from two incremental patches</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>dehtmldiff(1)</td>
<td>extract a diff from an HTML page</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>filterdiff(1)</td>
<td>extract or excludes diffs from a diff file</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>fixcvsdiff(1)</td>
<td>fix diff files created by CVS that patch(1) mis-interprets</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>flipdiff(1)</td>
<td>exchange the order of two patches</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>grepdiff(1)</td>
<td>show which files are modified by a patch matching a regex</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>interdiff(1)</td>
<td>show differences between two unified diff files</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>lsdiff(1)</td>
<td>show which files are modified by a patch</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>reccountdiff(1)</td>
<td>compute counts and offsets in unified context diffs</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>rediff(1)</td>
<td>fix offsets and counts of a hand-edited diff</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>splitdiff(1)</td>
<td>separate out incremental patches</td>
</tr>
<tr>
<td><strong>patchutils</strong></td>
<td>V:16, I:121</td>
<td>232</td>
<td>unwrapdiff(1)</td>
<td>demangle patches that have been word-wrapped</td>
</tr>
<tr>
<td><strong>dirdiff</strong></td>
<td>V:0, I:2</td>
<td>167</td>
<td>dirdiff(1)</td>
<td>display differences and merge changes between directory trees</td>
</tr>
<tr>
<td><strong>docdiff</strong></td>
<td>V:0, I:0</td>
<td>553</td>
<td>docdiff(1)</td>
<td>compare two files word by word / char by char</td>
</tr>
<tr>
<td><strong>makepatch</strong></td>
<td>V:0, I:0</td>
<td>100</td>
<td>makepatch(1)</td>
<td>generate extended patch files</td>
</tr>
<tr>
<td><strong>makepatch</strong></td>
<td>V:0, I:0</td>
<td>100</td>
<td>applypatch(1)</td>
<td>apply extended patch files</td>
</tr>
</tbody>
</table>
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.

```bash
$ 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.

```bash
$ patch -p0 file < file.patch0
$ patch -p1 file < file.patch1
```

10.4.3 Interactive merge

If you have two versions of a source code, you can perform 2-way merge interactively using imediff(1) by the following.

```bash
$ imediff -o file.merged file.old file.new
```

If you have three versions of a source code, you can perform 3-way merge interactively using imediff(1) by the following.

```bash
$ imediff -o file.merged file.yours file.base file.theirs
```

10.5 Git

Git is the tool of choice these days for the version control system (VCS) since Git can do everything for both local and remote source code management.

Debian provides free Git services via Debian Salsa service. Its documentation can be found at https://wiki.debian.org/Salsa.

Here are some Git related packages.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>command</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>git</td>
<td>V:341, I:534</td>
<td>46517</td>
<td>git(7)</td>
<td>Git, the fast, scalable, distributed revision control system</td>
</tr>
<tr>
<td>gitk</td>
<td>V:5, I:34</td>
<td>1830</td>
<td>gitk(1)</td>
<td>GUI Git repository browser with history</td>
</tr>
<tr>
<td>git-gui</td>
<td>V:1, I:18</td>
<td>2423</td>
<td>git-gui(1)</td>
<td>GUI for Git (No history)</td>
</tr>
<tr>
<td>git-email</td>
<td>V:0, I:10</td>
<td>1080</td>
<td>git-send-email(1)</td>
<td>collection of patches as email from the Git</td>
</tr>
<tr>
<td>git-buildpackage</td>
<td>V:1, I:9</td>
<td>1990</td>
<td>git-buildpackage(1)</td>
<td>the Debian packaging with the Git</td>
</tr>
<tr>
<td>dgit</td>
<td>V:0, I:1</td>
<td>484</td>
<td>dgit(1)</td>
<td>git interoperability with the Debian archive</td>
</tr>
<tr>
<td>imediff</td>
<td>V:0, I:0</td>
<td>169</td>
<td>git-ime(1)</td>
<td>interactive git commit split helper tool</td>
</tr>
<tr>
<td>stgit</td>
<td>V:0, I:0</td>
<td>601</td>
<td>stg(1)</td>
<td>quilt on top of git (Python)</td>
</tr>
<tr>
<td>git-doc</td>
<td>T:12</td>
<td>13165</td>
<td>N/A</td>
<td>official documentation for Git</td>
</tr>
<tr>
<td>gitmagic</td>
<td>T:0</td>
<td>721</td>
<td>N/A</td>
<td>&quot;Git Magic&quot;, easier to understand guide for Git</td>
</tr>
</tbody>
</table>

Table 10.11: List of git related packages and commands
### 10.5.1 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
```

You may also customize the Git default behavior by the following.

```
$ git config --global init.defaultBranch main
$ git config --global pull.rebase true
$ git config --global push.default current
```

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
```

### 10.5.2 Basic Git commands

Git operation involves several data.

- The working tree which holds user facing files and to which you make changes.
  - The changes to be recorded must be explicitly selected and staged to the index. This is `git add` and `git rm` commands.
- The index which holds staged files.
  - Staged files will be committed to the local repository upon the subsequent request. This is `git commit` command.
- The local repository which holds committed files.
  - Git records the linked history of the committed data and organizes them as branches in the repository.
  - The local repository can send data to the remote repository by `git push` command.
  - The local repository can receive data from the remote repository by `git fetch` and `git pull` commands.
    - The `git pull` command performs `git merge` or `git rebase` command after `git fetch` command.
    - Here, `git merge` combines two separate branches of history at the end to a point. (This is default of `git pull` without customization and may be good for upstream people who publish branch to many people.)
    - Here, `git rebase` creates one single branch of sequential history of the remote branch one followed by the local branch one. (This is `pull.rebase true` customization case and may be good for rest of us.)
- The remote repository which holds committed files.
  - The communication to the remote repository uses secure communication protocols such as SSH or HTTPS.

The working tree is files outside of the `.git/` directory. Files inside of the `.git/` directory hold the index, the local repository data, and some git configuration text files.

Here is an overview of main Git commands.
<table>
<thead>
<tr>
<th>Git command</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>git init</code></td>
<td>create the (local) repository</td>
</tr>
<tr>
<td><code>git clone URL</code></td>
<td>clone the remote repository to a local repository with the working tree</td>
</tr>
<tr>
<td><code>git pull origin main</code></td>
<td>update the local main branch by the remote repository origin</td>
</tr>
<tr>
<td><code>git add .</code></td>
<td>add file(s) in the working tree to the index for pre-existing files in index only</td>
</tr>
<tr>
<td><code>git add -A .</code></td>
<td>add file(s) in the working tree to the index for all files including removals</td>
</tr>
<tr>
<td><code>git rm filename</code></td>
<td>remove file(s) from the working tree and the index</td>
</tr>
<tr>
<td><code>git commit</code></td>
<td>commit staged changes in the index to the local repository</td>
</tr>
<tr>
<td><code>git commit -a</code></td>
<td>add all changes in the working tree to the index and commit them to the local repository (add + commit)</td>
</tr>
<tr>
<td><code>git push -u origin branch_name</code></td>
<td>update the remote repository origin by the local branch_name branch (initial invocation)</td>
</tr>
<tr>
<td><code>git push origin branch_name</code></td>
<td>update the remote repository origin by the local branch_name branch (subsequent invocation)</td>
</tr>
<tr>
<td><code>git diff treeish1 treeish2</code></td>
<td>show difference between treeish1 commit and treeish2 commit</td>
</tr>
<tr>
<td><code>gitk</code></td>
<td>GUI display of VCS repository branch history tree</td>
</tr>
</tbody>
</table>

Table 10.12: Main Git commands

### 10.5.3 Git tips

Here are some Git tips.

#### Warning

![Warning](warning_icon.png)

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.

---

#### Caution

![Caution](caution_icon.png)

If a local branch which has been pushed to remote repository is rebased or squashed, pushing this branch has risks and requires `-force` option. This is usually not an acceptable for main branch but may be acceptable for a topic branch before merging to main branch.

---

#### Caution

![Caution](caution_icon.png)

Invoking a git subcommand directly as "git-xyz" from the command line has been deprecated since early 2006.

---

#### Tip

![Tip](tip_icon.png)

If there is an executable file `git-foo` in the path specified by `$PATH`, entering "git foo" without hyphen to the command line invokes this `git-foo`. This is a feature of the `git` command.

### 10.5.4 Git references

See the following.

- `manpage: git(1) (/usr/share/doc/git-doc/git.html)`
<table>
<thead>
<tr>
<th>Git command line</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>gitk --all</code></td>
<td>see complete Git history and operate on them such as resetting HEAD to another commit, cheery-picking patches, creating tags and branches ...</td>
</tr>
<tr>
<td><code>git stash</code></td>
<td>get the clean working tree without losing data</td>
</tr>
<tr>
<td><code>git remote -v</code></td>
<td>check settings for remote</td>
</tr>
<tr>
<td><code>git branch -vv</code></td>
<td>check settings for branch</td>
</tr>
<tr>
<td><code>git status</code></td>
<td>show working tree status</td>
</tr>
<tr>
<td><code>git config -l</code></td>
<td>list git settings</td>
</tr>
<tr>
<td><code>git reset --hard HEAD; git clean -x -d -f</code></td>
<td>revert all working tree changes and clean them up completely</td>
</tr>
<tr>
<td><code>git rm --cached filename</code></td>
<td>revert staged index changed by git add filename</td>
</tr>
<tr>
<td><code>git reflog</code></td>
<td>get reference log (useful for recovering commits from the removed branch)</td>
</tr>
<tr>
<td><code>git branch new_branch_name HEAD@{6}</code></td>
<td>create a new branch from reflog information</td>
</tr>
<tr>
<td><code>git remote add new_remote URL</code></td>
<td>add a new remote repository pointed by URL</td>
</tr>
<tr>
<td><code>git remote rename origin upstream</code></td>
<td>rename the remote repository name from origin to upstream</td>
</tr>
<tr>
<td><code>git branch -u upstream/branch_name</code></td>
<td>set the remote tracking to the remote repository upstream and its branch name branch_name.</td>
</tr>
<tr>
<td><code>git remote set-url origin https://foo/bar.git</code></td>
<td>change URL of origin</td>
</tr>
<tr>
<td><code>git remote set-url --push upstream DISABLED</code></td>
<td>disable push to upstream (Edit .git/config to re-enable)</td>
</tr>
<tr>
<td><code>git checkout -b topic_branch; git push -u topic_branch origin</code></td>
<td>make a new topic_branch and push it to origin</td>
</tr>
<tr>
<td><code>git branch -m oldname newname</code></td>
<td>rename local branch name</td>
</tr>
<tr>
<td><code>git push -d origin branch_to_be_removed</code></td>
<td>remove remote branch (new method)</td>
</tr>
<tr>
<td><code>git push origin :branch_to_be_removed</code></td>
<td>remove remote branch (old method)</td>
</tr>
<tr>
<td><code>git checkout --orphan unconnected</code></td>
<td>create a new unconnected branch</td>
</tr>
<tr>
<td><code>git fetch upstream foo:upstream-foo</code></td>
<td>create a local (possibly orphan) upstream-foo branch as a copy of foo branch the upstream repository</td>
</tr>
<tr>
<td><code>git rebase -i origin/main</code></td>
<td>reorder/drop/squish commits from origin/main to clean branch history</td>
</tr>
<tr>
<td><code>git reset HEAD^; git commit --amend</code></td>
<td>squash last 2 commits into one</td>
</tr>
<tr>
<td><code>git checkout topic_branch; git merge --squash topic_branch</code></td>
<td>squash entire topic_branch into a commit</td>
</tr>
<tr>
<td><code>git fetch --unshallow --update-head-ok origin ‘+refs/heads/*:refs/heads/*’</code></td>
<td>convert a shallow clone to the full clone of all branches</td>
</tr>
<tr>
<td><code>git ime</code></td>
<td>split the last commit into a series of file-by-file smaller commits etc. (imediff package required)</td>
</tr>
<tr>
<td><code>git repack -a -d; git prune</code></td>
<td>repack the local repository into single pack (this may limit chance of lost data recovery from erased branch etc.)</td>
</tr>
</tbody>
</table>

Table 10.13: Git tips
• 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/giteveryday.html)
• Git Magic (/usr/share/doc/gitmagic/html/index.html)

10.5.5 Other version control systems

The version control systems (VCS) is sometimes known as the revision control system (RCS), or the software configuration management (SCM).

Here is a summary of the notable other non-Git VCS on the Debian system.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>tool</th>
<th>VCS type</th>
<th>comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>mercurial</td>
<td>V:5, I:33</td>
<td>1053</td>
<td>Mercurial</td>
<td>distributed</td>
<td>DVCS in Python and some C</td>
</tr>
<tr>
<td>darcs</td>
<td>V:0, I:5</td>
<td>34070</td>
<td>Darcs</td>
<td>distributed</td>
<td>DVCS with smart algebra of patches (slow)</td>
</tr>
<tr>
<td>bzr</td>
<td>I:8</td>
<td>28</td>
<td>Bazaar</td>
<td>distributed</td>
<td>DVCS influenced by tla written in Python (historic)</td>
</tr>
<tr>
<td>tla</td>
<td>V:0, I:1</td>
<td>1022</td>
<td>GNU arch</td>
<td>distributed</td>
<td>DVCS mainly by Tom Lord (historic)</td>
</tr>
<tr>
<td>subversion</td>
<td>V:13, I:75</td>
<td>4836</td>
<td>Subversion</td>
<td>remote</td>
<td>&quot;CVS done right&quot;, newer standard remote VCS (historic)</td>
</tr>
<tr>
<td>cvs</td>
<td>V:3, I:30</td>
<td>4609</td>
<td>CVS</td>
<td>remote</td>
<td>previous standard remote VCS (historic)</td>
</tr>
<tr>
<td>tkcvs</td>
<td>V:0, I:1</td>
<td>1498</td>
<td>CVS, &quot;…&quot;</td>
<td>remote</td>
<td>GUI display of VCS (CVS, Subversion, RCS) repository tree</td>
</tr>
<tr>
<td>rcs</td>
<td>V:2, I:13</td>
<td>564</td>
<td>RCS</td>
<td>local</td>
<td>&quot;Unix SCCS done right&quot; (historic)</td>
</tr>
<tr>
<td>cssc</td>
<td>V:0, I:1</td>
<td>2044</td>
<td>CSSC</td>
<td>local</td>
<td>clone of the Unix SCCS (historic)</td>
</tr>
</tbody>
</table>

Table 10.14: List of other version control system tools
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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libc6</td>
<td>V:915, I:999</td>
<td>12989</td>
<td>charset</td>
<td>text encoding converter between locales by <code>iconv</code> (fundamental)</td>
</tr>
<tr>
<td>recode</td>
<td>V:2, I:19</td>
<td>601</td>
<td>charset+eol</td>
<td>text encoding converter between locales (versatile, more aliases and features)</td>
</tr>
<tr>
<td>konwert</td>
<td>V:1, I:49</td>
<td>134</td>
<td>charset</td>
<td>text encoding converter between locales (fancy)</td>
</tr>
<tr>
<td>nkf</td>
<td>V:0, I:9</td>
<td>360</td>
<td>charset</td>
<td>character set translator for Japanese</td>
</tr>
<tr>
<td>tcs</td>
<td>V:0, I:0</td>
<td>518</td>
<td>charset</td>
<td>character set translator</td>
</tr>
<tr>
<td>unaccent</td>
<td>V:0, I:0</td>
<td>35</td>
<td>charset</td>
<td>replace accented letters by their unaccented equivalent</td>
</tr>
<tr>
<td>tofroodos</td>
<td>V:1, I:18</td>
<td>51</td>
<td>eol</td>
<td>text format converter between DOS and Unix: <code>fromdos</code> and <code>todos</code></td>
</tr>
<tr>
<td>macutils</td>
<td>V:0, I:0</td>
<td>312</td>
<td>eol</td>
<td>text format converter between Macintosh and Unix: <code>frommac</code> and <code>tomac</code></td>
</tr>
</tbody>
</table>

Table 11.1: List of text data conversion tools

11.1.1 Converting a text file with `iconv`

Tip

`iconv` 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` by the following.

```bash
$ 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.
<table>
<thead>
<tr>
<th>encoding value</th>
<th>usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange, 7 bit code w/o accented characters</td>
</tr>
<tr>
<td>UTF-8</td>
<td>current multilingual standard for all modern OSs</td>
</tr>
<tr>
<td>ISO-8859-1</td>
<td>old standard for western European languages, ASCII + accented characters</td>
</tr>
<tr>
<td>ISO-8859-2</td>
<td>old standard for eastern European languages, ASCII + accented characters</td>
</tr>
<tr>
<td>ISO-8859-15</td>
<td>old standard for western European languages, ISO-8859-1 with euro sign</td>
</tr>
<tr>
<td>CP850</td>
<td>code page 850, Microsoft DOS characters with graphics for western European languages, ISO-8859-1 variant</td>
</tr>
<tr>
<td>CP932</td>
<td>code page 932, Microsoft Windows style Shift-JIS variant for Japanese</td>
</tr>
<tr>
<td>CP936</td>
<td>code page 936, Microsoft Windows style GB2312, GBK or GB18030 variant for Simplified Chinese</td>
</tr>
<tr>
<td>CP949</td>
<td>code page 949, Microsoft Windows style EUC-KR or Unified Hangul Code variant for Korean</td>
</tr>
<tr>
<td>CP950</td>
<td>code page 950, Microsoft Windows style Big5 variant for Traditional Chinese</td>
</tr>
<tr>
<td>CP1251</td>
<td>code page 1251, Microsoft Windows style encoding for the Cyrillic alphabet</td>
</tr>
<tr>
<td>CP1252</td>
<td>code page 1252, Microsoft Windows style ISO-8859-15 variant for western European languages</td>
</tr>
<tr>
<td>KOI8-R</td>
<td>old Russian UNIX standard for the Cyrillic alphabet</td>
</tr>
<tr>
<td>ISO-2022-JP</td>
<td>standard encoding for Japanese email which uses only 7 bit codes</td>
</tr>
<tr>
<td>eucJP</td>
<td>old Japanese UNIX standard 8 bit code and completely different from Shift-JIS</td>
</tr>
<tr>
<td>Shift-JIS</td>
<td>JIS X 0208 Appendix 1 standard for Japanese (see CP932)</td>
</tr>
</tbody>
</table>

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 8.1).

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 → "\" and 0x7E → "~". 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 8.1.3) 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.

<table>
<thead>
<tr>
<th>platform</th>
<th>EOL code</th>
<th>control</th>
<th>decimal</th>
<th>hexadecimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debian (unix)</td>
<td>LF</td>
<td>^J</td>
<td>10</td>
<td>0A</td>
</tr>
<tr>
<td>MSDOS and Windows</td>
<td>CR-LF</td>
<td>^M^J</td>
<td>13 10</td>
<td>0D 0A</td>
</tr>
<tr>
<td>Apple’s Macintosh</td>
<td>CR</td>
<td>^M</td>
<td>13</td>
<td>0D</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>function</th>
<th>bsdmainutils</th>
<th>coreutils</th>
</tr>
</thead>
<tbody>
<tr>
<td>expand tab to spaces</td>
<td>&quot;col -x&quot;</td>
<td>expand</td>
</tr>
<tr>
<td>unexpand tab from spaces</td>
<td>&quot;col -h&quot;</td>
<td>unexpand</td>
</tr>
</tbody>
</table>

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 '++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 '++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 '++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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>function</th>
</tr>
</thead>
<tbody>
<tr>
<td>w3m</td>
<td>V:14, I:186</td>
<td>2828</td>
<td>html → text</td>
<td>HTML to text converter with the &quot;w3m -dump&quot; command</td>
</tr>
<tr>
<td>html2text</td>
<td>V:3, I:49</td>
<td>274</td>
<td>html → text</td>
<td>advanced HTML to text converter (ISO 8859-1)</td>
</tr>
<tr>
<td>lynx</td>
<td>V:22, I:303</td>
<td>1935</td>
<td>html → text</td>
<td>HTML to text converter with the &quot;lynx -dump&quot; command</td>
</tr>
<tr>
<td>elinks</td>
<td>V:3, I:21</td>
<td>1645</td>
<td>html → text</td>
<td>HTML to text converter with the &quot;elinks -dump&quot; command</td>
</tr>
<tr>
<td>links</td>
<td>V:3, I:29</td>
<td>2314</td>
<td>html → text</td>
<td>HTML to text converter with the &quot;links -dump&quot; command</td>
</tr>
<tr>
<td>links2</td>
<td>V:1, I:11</td>
<td>5492</td>
<td>html → text</td>
<td>HTML to text converter with the &quot;links2 -dump&quot; command</td>
</tr>
<tr>
<td>catdoc</td>
<td>V:14, I:148</td>
<td>686</td>
<td>MSWord → text</td>
<td>convert MSWord files to plain text or TeX</td>
</tr>
<tr>
<td>antiword</td>
<td>V:1, I:7</td>
<td>589</td>
<td>MSWord → text</td>
<td>convert MSWord files to plain text or ps</td>
</tr>
<tr>
<td>pstotext</td>
<td>V:0, I:1</td>
<td>122</td>
<td>ps/pdf → text</td>
<td>extract text from PostScript and PDF files</td>
</tr>
<tr>
<td>unhtml</td>
<td>V:0, I:0</td>
<td>40</td>
<td>html → text</td>
<td>remove the markup tags from an HTML file</td>
</tr>
<tr>
<td>odt2txt</td>
<td>V:2, I:35</td>
<td>60</td>
<td>odt → text</td>
<td>converter from OpenDocument Text to text</td>
</tr>
</tbody>
</table>

Table 11.5: List of tools to extract 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](http://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-xml package, which is used here.

Each XML file starts with standard XML declaration as the following.

```xml
<?xml version="1.0" encoding="UTF-8"?>
```

The basic syntax for one XML element is marked up as the following.

```xml
<name attribute="value">content</name>
```
XML element with empty content is marked up in the following short form.

```xml
<name attribute="value" />
```

The "attribute="value"" in the above examples are optional.

The comment section in XML is marked up as the following.

```xml
<!-- comment -->
```

Other than adding markups, XML requires minor conversion to the content using predefined entities for following characters.

<table>
<thead>
<tr>
<th>predefined entity</th>
<th>character to be converted into</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;</td>
<td>&quot; : quote</td>
</tr>
<tr>
<td>'</td>
<td>’ : apostrophe</td>
</tr>
<tr>
<td>&lt;</td>
<td>&lt; : less-than</td>
</tr>
<tr>
<td>&gt;</td>
<td>&gt; : greater-than</td>
</tr>
<tr>
<td>&amp;</td>
<td>&amp; : ampersand</td>
</tr>
</tbody>
</table>

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 programming 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.

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).

Tip

GNOME’s yelp is sometimes handy to read DocBook XML files directly since it renders decently on X.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>docbook-xml</td>
<td>I:398</td>
<td>2134</td>
<td>xml</td>
<td>XML document type definition (DTD) for DocBook</td>
</tr>
<tr>
<td>docbook-xsl</td>
<td>V:13, I:144</td>
<td>14851</td>
<td>xml/xslt</td>
<td>XSL stylesheets for processing DocBook XML to various output formats with XSLT</td>
</tr>
<tr>
<td>xsltproc</td>
<td>V:16, I:79</td>
<td>162</td>
<td>xslt</td>
<td>XSLT command line processor (XML → XML, HTML, plain text, etc.)</td>
</tr>
<tr>
<td>xmlto</td>
<td>V:0, I:14</td>
<td>130</td>
<td>xml/xslt</td>
<td>XML-to-any converter with XSLT</td>
</tr>
<tr>
<td>fop</td>
<td>V:0, I:12</td>
<td>285</td>
<td>xml/xsl-to</td>
<td>convert Docbook XML files to PDF</td>
</tr>
<tr>
<td>dblatex</td>
<td>V:3, I:10</td>
<td>4643</td>
<td>xml/xslt</td>
<td>convert Docbook files to DVI, PostScript, PDF documents with XSLT</td>
</tr>
<tr>
<td>dbtoepub</td>
<td>V:0, I:0</td>
<td>37</td>
<td>xml/xslt</td>
<td>DocBook XML to .epub converter</td>
</tr>
</tbody>
</table>

Table 11.8: List of XML tools

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>openjade</td>
<td>V:1, I:27</td>
<td>1019</td>
<td>dsssl</td>
<td>ISO/IEC 10179:1996 standard DSSSL processor (latest)</td>
</tr>
<tr>
<td>docbook-dsssl</td>
<td>V:0, I:14</td>
<td>2605</td>
<td>xml/dsssl</td>
<td>DSSSL stylesheets for processing DocBook XML to various output formats with DSSSL</td>
</tr>
<tr>
<td>docbook-utils</td>
<td>V:0, I:10</td>
<td>287</td>
<td>xml/dsssl</td>
<td>utilities for DocBook files including conversion to other formats (HTML, RTF, PS, man, PDF) with docbook2* commands with DSSSL</td>
</tr>
<tr>
<td>sgml2x</td>
<td>V:0, I:0</td>
<td>90</td>
<td>SGML/dsssl</td>
<td>converter from SGML and XML using DSSSL stylesheets</td>
</tr>
</tbody>
</table>

Table 11.9: List of DSSSL tools

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>man2html</td>
<td>V:0, I:1</td>
<td>138</td>
<td>manpage→html</td>
<td>converter from manpage to HTML (CGI support)</td>
</tr>
<tr>
<td>doclifter</td>
<td>V:0, I:0</td>
<td>451</td>
<td>troff→xml</td>
<td>converter from troff to DocBook XML</td>
</tr>
<tr>
<td>texi2html</td>
<td>V:0, I:6</td>
<td>1847</td>
<td>texi→html</td>
<td>converter from Texinfo to HTML</td>
</tr>
<tr>
<td>info2www</td>
<td>V:1, I:2</td>
<td>73</td>
<td>info→html</td>
<td>converter from GNU info to HTML (CGI support)</td>
</tr>
<tr>
<td>wv</td>
<td>V:0, I:5</td>
<td>733</td>
<td>MSWord→any</td>
<td>document converter from Microsoft Word to HTML, LaTeX, etc.</td>
</tr>
<tr>
<td>unrutf</td>
<td>V:0, I:3</td>
<td>148</td>
<td>rtf→html</td>
<td>document converter from RTF to HTML, etc</td>
</tr>
<tr>
<td>wp2x</td>
<td>V:0, I:0</td>
<td>200</td>
<td>WordPerfect→any</td>
<td>WordPerfect 5.0 and 5.1 files to TeX, LaTeX, troff, GML and HTML</td>
</tr>
</tbody>
</table>

Table 11.10: List of XML data extraction tools
11.2.3 The XML data extraction

You can extract HTML or XML data from other formats using followings.

11.2.4 The XML data lint

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.

Syntax of XML files and goodness of URLs found in them may be checked.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>function</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libxml2-utils</td>
<td>V:21,I:212</td>
<td>179</td>
<td>xml ↔ html ↔ xhtml</td>
<td>command line XML tool with xmllint(1) (syntax check, reformat, lint, ...)</td>
</tr>
<tr>
<td>tidy</td>
<td>V:1,I:9</td>
<td>84</td>
<td>xml ↔ html ↔ xhtml</td>
<td>XHTML syntax checker and reformatter</td>
</tr>
<tr>
<td>weblint-perl</td>
<td>V:0,I:1</td>
<td>32</td>
<td>lint</td>
<td>syntax and minimal style checker for HTML</td>
</tr>
<tr>
<td>linklint</td>
<td>V:0,I:0</td>
<td>343</td>
<td>link check</td>
<td>fast link checker and web site maintenance tool</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>texlive</td>
<td>V:3,I:36</td>
<td>57</td>
<td>(La)TeX</td>
<td>TeX system for typesetting, previewing and printing</td>
</tr>
<tr>
<td>groff</td>
<td>V:2,I:39</td>
<td>20/20</td>
<td>troff</td>
<td>GNU troff text-formating system</td>
</tr>
</tbody>
</table>

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 "^H" and "_", use "GROFF_NO_SGR=1 groff -Tascii -me -" instead.

Tip
To remove "^H" 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, Samarim, (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.

- 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.
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.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ghostscript</td>
<td>V:161</td>
<td>180</td>
<td></td>
<td>The GPL Ghostscript PostScript/PDF interpreter</td>
</tr>
<tr>
<td>ghostscript-x</td>
<td>V:3</td>
<td>87</td>
<td></td>
<td>GPL Ghostscript PostScript/PDF interpreter - X display support</td>
</tr>
<tr>
<td>libpoppler102</td>
<td>V:21</td>
<td>4274</td>
<td></td>
<td>PDF rendering library forked from the xpdf PDF viewer</td>
</tr>
<tr>
<td>libpoppler-glib8</td>
<td>V:276</td>
<td>484</td>
<td></td>
<td>PDF rendering library (GLib-based shared library)</td>
</tr>
<tr>
<td>poppler-data</td>
<td>V:121</td>
<td>13086</td>
<td></td>
<td>CMaps for PDF rendering library (for CJK support: Adobe-*)</td>
</tr>
</tbody>
</table>

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.

```bash
$ 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, psmmerge(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.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>poppler-utils</td>
<td>V:196, I:470</td>
<td>717</td>
<td>pdf → ps, text, ...</td>
<td>PDF utilities: pdf2ps, pdfinfo, pdfimages, pdftotext, pdffonts</td>
</tr>
<tr>
<td>psutils</td>
<td>V:4, I:71</td>
<td>219</td>
<td>ps → ps</td>
<td>PostScript document conversion tools</td>
</tr>
<tr>
<td>poster</td>
<td>V:0, I:3</td>
<td>57</td>
<td>ps → ps</td>
<td>create large posters out of PostScript pages</td>
</tr>
<tr>
<td>enscript</td>
<td>V:1, I:14</td>
<td>2130</td>
<td>text → ps, html, rtf</td>
<td>convert ASCII text to PostScript, HTML, RTF or Pretty-Print</td>
</tr>
<tr>
<td>a2ps</td>
<td>V:0, I:10</td>
<td>3979</td>
<td>text → ps</td>
<td>'Anything to PostScript' converter and pretty-printer</td>
</tr>
<tr>
<td>pdf2tk</td>
<td>I:38</td>
<td>28</td>
<td>pdf → pdf</td>
<td>PDF document conversion tool: pdf2tk</td>
</tr>
<tr>
<td>html2ps</td>
<td>V:0, I:2</td>
<td>261</td>
<td>html → ps</td>
<td>converter from HTML to PostScript</td>
</tr>
<tr>
<td>gnuhtml2latex</td>
<td>V:0, I:0</td>
<td>27</td>
<td>html → latex</td>
<td>converter from html to latex</td>
</tr>
<tr>
<td>latex2rtf</td>
<td>V:0, I:4</td>
<td>495</td>
<td>latex → rtf</td>
<td>convert documents from LaTeX to RTF which can be read by MS Word</td>
</tr>
<tr>
<td>ps2eps</td>
<td>V:2, I:44</td>
<td>95</td>
<td>ps → eps</td>
<td>converter from PostScript to EPS (Encapsulated PostScript)</td>
</tr>
<tr>
<td>e2ps</td>
<td>V:0, I:0</td>
<td>109</td>
<td>text → ps</td>
<td>Text to PostScript converter with Japanese encoding support</td>
</tr>
<tr>
<td>impose+</td>
<td>V:0, I:0</td>
<td>118</td>
<td>ps → ps</td>
<td>PostScript utilities</td>
</tr>
<tr>
<td>trueprint</td>
<td>V:0, I:0</td>
<td>149</td>
<td>text → ps</td>
<td>pretty print many source codes (C, C++, Java, Pascal, Perl, Pike, Sh, and Verilog) to PostScript. (C language)</td>
</tr>
<tr>
<td>pdf2svg</td>
<td>V:0, I:3</td>
<td>30</td>
<td>ps → svg</td>
<td>converter from PDF to Scalable vector graphics format</td>
</tr>
<tr>
<td>pdftoipe</td>
<td>V:0, I:0</td>
<td>65</td>
<td>ps → ipe</td>
<td>converter from PDF to IPE's XML format</td>
</tr>
</tbody>
</table>

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.

```bash
$ lp -n 3 -o Collate=True filename
```

```bash
$ 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.

**Tip**
The Internet Message Access Protocol version 4 (IMAP4) server 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 Table 11.2).
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sharutils</td>
<td>V:3, I:38</td>
<td>1415</td>
<td>mail</td>
<td>shar(1), unshar(1), uuencode(1), uudecode(1)</td>
</tr>
<tr>
<td>mpack</td>
<td>V:1, I:12</td>
<td>108</td>
<td>MIME</td>
<td>encoding and decoding of MIME messages: mpack(1) and munpack(1)</td>
</tr>
<tr>
<td>tnef</td>
<td>V:0, I:7</td>
<td>110</td>
<td>ms-tnef</td>
<td>unpacking MIME attachments of type &quot;application/ms-tnef&quot; which is a Microsoft only format</td>
</tr>
<tr>
<td>uuudeview</td>
<td>V:0, I:3</td>
<td>105</td>
<td>mail</td>
<td>encoder and decoder for the following formats: uuencode, xxencode, BASE64, quoted printable, and BinHex</td>
</tr>
</tbody>
</table>

Table 11.16: List of packages to help mail data conversion

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.

**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.
<table>
<thead>
<tr>
<th>Package</th>
<th>Version</th>
<th>Size</th>
<th>Keyword</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gimp</td>
<td>V:53,1:258</td>
<td>19877</td>
<td>image(bitmap)</td>
<td>GNU Image Manipulation Program</td>
</tr>
<tr>
<td>imagemagick</td>
<td>I:320</td>
<td>73</td>
<td>image(bitmap)</td>
<td>image manipulation programs</td>
</tr>
<tr>
<td>graphicsmagick</td>
<td>V:1,1:12</td>
<td>5564</td>
<td>image(bitmap)</td>
<td>image manipulation programs (fork of imagemagick)</td>
</tr>
<tr>
<td>xsane</td>
<td>V:12,1:146</td>
<td>2339</td>
<td>image(bitmap)</td>
<td>GTK-based X11 frontend for SANE (Scanner Access Now Easy)</td>
</tr>
<tr>
<td>netpbm</td>
<td>V:27,1:328</td>
<td>8252</td>
<td>image(bitmap)</td>
<td>graphics conversion tools</td>
</tr>
<tr>
<td>icoutils</td>
<td>V:7,1:54</td>
<td>221</td>
<td>png -- icon(bitmap)</td>
<td>convert MS Windows icons and cursors to and from PNG formats (favicon.ico)</td>
</tr>
<tr>
<td>scribus</td>
<td>V:1,1:17</td>
<td>30242</td>
<td>ps/pdf/SVG/…</td>
<td>Scribus DTP editor</td>
</tr>
<tr>
<td>libreoffice-draw</td>
<td>V:7,1:430</td>
<td>13625</td>
<td>image(vector)</td>
<td>LibreOffice suite - drawing</td>
</tr>
<tr>
<td>inkscape</td>
<td>V:16,1:121</td>
<td>99316</td>
<td>image(vector)</td>
<td>SVG (Scalable Vector Graphics) editor</td>
</tr>
<tr>
<td>dia</td>
<td>V:3,1:24</td>
<td>3908</td>
<td>image(vector)</td>
<td>diagram editor (Gtk)</td>
</tr>
<tr>
<td>xfig</td>
<td>V:0,1:11</td>
<td>7825</td>
<td>image(vector)</td>
<td>Facility for Interactive Generation of figures under X11</td>
</tr>
<tr>
<td>pstoedit</td>
<td>V:2,1:55</td>
<td>1004</td>
<td>ps/pdf -- image(vector)</td>
<td>PostScript and PDF files to editable vector graphics (SVG)</td>
</tr>
<tr>
<td>libwmf-bin</td>
<td>V:7,1:128</td>
<td>151</td>
<td>Windows/image(vector)</td>
<td>Windows metafile (vector graphic data) conversion tools</td>
</tr>
<tr>
<td>Fig2xsd</td>
<td>V:0,1:10</td>
<td>151</td>
<td>fig -- xsd(vector)</td>
<td>Convert XFig files to OpenOffice.org Draw format</td>
</tr>
<tr>
<td>unpaper</td>
<td>V:2,1:17</td>
<td>412</td>
<td>image -- image</td>
<td>post-processing tool for scanned pages for OCR</td>
</tr>
<tr>
<td>tesseract-ocr</td>
<td>V:7,1:33</td>
<td>2135</td>
<td>image -- text</td>
<td>free OCR software based on the HP's commercial OCR engine</td>
</tr>
<tr>
<td>tesseract-ocr-eng</td>
<td>V:7,1:34</td>
<td>4032</td>
<td>image -- text</td>
<td>OCR engine data: tesseract-ocr language files for English text</td>
</tr>
<tr>
<td>gocr</td>
<td>V:0,1:8</td>
<td>545</td>
<td>image -- text</td>
<td>free OCR software</td>
</tr>
<tr>
<td>ocrad</td>
<td>V:0,1:3</td>
<td>578</td>
<td>image -- text</td>
<td>free OCR software</td>
</tr>
<tr>
<td>eog</td>
<td>V:63,1:279</td>
<td>7772</td>
<td>image(Exif)</td>
<td>Eye of GNOME graphics viewer program</td>
</tr>
<tr>
<td>gthumb</td>
<td>V:4,1:17</td>
<td>5036</td>
<td>image(Exif)</td>
<td>image viewer and browser (GNOME)</td>
</tr>
<tr>
<td>geeqle</td>
<td>V:4,1:15</td>
<td>2256</td>
<td>image(Exif)</td>
<td>image viewer using GTK</td>
</tr>
<tr>
<td>shotwell</td>
<td>V:17,1:255</td>
<td>6187</td>
<td>image(Exif)</td>
<td>digital photo organizer (GNOME)</td>
</tr>
<tr>
<td>gtkam</td>
<td>V:0,1:4</td>
<td>1154</td>
<td>image(Exif)</td>
<td>application for retrieving media from digital cameras (GTK)</td>
</tr>
<tr>
<td>gphoto2</td>
<td>V:0,1:9</td>
<td>947</td>
<td>image(Exif)</td>
<td>The gphoto2 digital camera command-line client</td>
</tr>
<tr>
<td>gwenview</td>
<td>V:32,1:103</td>
<td>11755</td>
<td>image(Exif)</td>
<td>image viewer (KDE)</td>
</tr>
<tr>
<td>kamera</td>
<td>I:102</td>
<td>998</td>
<td>image(Exif)</td>
<td>digital camera support for KDE applications</td>
</tr>
<tr>
<td>digikam</td>
<td>V:1,1:9</td>
<td>293</td>
<td>image(Exif)</td>
<td>digital photo management application for KDE</td>
</tr>
<tr>
<td>exiv2</td>
<td>V:2,1:28</td>
<td>278</td>
<td>image(Exif)</td>
<td>EXIF/ IPTC metadata manipulation tool</td>
</tr>
<tr>
<td>exiftran</td>
<td>V:1,1:15</td>
<td>69</td>
<td>image(Exif)</td>
<td>transform digital camera jpeg images</td>
</tr>
<tr>
<td>jhead</td>
<td>V:0,1:8</td>
<td>132</td>
<td>image(Exif)</td>
<td>manipulate the non-image part of Exif compliant JPEG (digital camera photo files)</td>
</tr>
<tr>
<td>exif</td>
<td>V:2,1:37</td>
<td>339</td>
<td>image(Exif)</td>
<td>command-line utility to show EXIF information in JPEG files</td>
</tr>
<tr>
<td>exiftags</td>
<td>V:0,1:3</td>
<td>292</td>
<td>image(Exif)</td>
<td>utility to read Exif tags from a digital camera JPEG file</td>
</tr>
<tr>
<td>exifprobe</td>
<td>V:0,1:3</td>
<td>499</td>
<td>image(Exif)</td>
<td>read metadata from digital pictures</td>
</tr>
<tr>
<td>dcrw</td>
<td>V:1,1:12</td>
<td>583</td>
<td>image(Raw) → RIFF</td>
<td>decode raw digital camera images</td>
</tr>
<tr>
<td>findimagemedups</td>
<td>V:0,1:1</td>
<td>76</td>
<td>image → fingerprint</td>
<td>visually similar or duplicate images</td>
</tr>
<tr>
<td>ale</td>
<td>V:0,1:0</td>
<td>839</td>
<td>image → image</td>
<td>merge images to increase fidelity or create mosaics</td>
</tr>
<tr>
<td>imageindex</td>
<td>V:0,1:1</td>
<td>145</td>
<td>image(Exif) → RIFF</td>
<td>generate static HTML galleries from images</td>
</tr>
<tr>
<td>outguess</td>
<td>V:0,1:1</td>
<td>230</td>
<td>jpeg, png</td>
<td>universal Steganographic tool</td>
</tr>
<tr>
<td>librebreed</td>
<td>V:1,1:15</td>
<td>8798</td>
<td>DXF</td>
<td>CAD data editor (KDE)</td>
</tr>
<tr>
<td>blender</td>
<td>V:3,1:29</td>
<td>82128</td>
<td>blend, TIFF, VRML, ⋮</td>
<td>3D content editor for animation etc</td>
</tr>
<tr>
<td>mm3d</td>
<td>V:0,1:0</td>
<td>3881</td>
<td>ms3d, obj, dxf, ⋮</td>
<td>OpenGL based 3D model editor</td>
</tr>
<tr>
<td>open-font-design</td>
<td>V:0,1:6</td>
<td>3980</td>
<td>ttf, ps, ⋮</td>
<td>font editor for PS, TrueType and OpenType fonts</td>
</tr>
<tr>
<td>fontforge</td>
<td>V:0,1:0</td>
<td>806</td>
<td>ttf</td>
<td>program for gridfitting and hinting TrueType fonts</td>
</tr>
</tbody>
</table>
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).

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alien</td>
<td>V:1, I:21</td>
<td>163</td>
<td>rpm/tgz → deb</td>
<td>converter for the foreign package into the Debian package</td>
</tr>
<tr>
<td>freepwing</td>
<td>V:0, I:0</td>
<td>424</td>
<td>EB → EPWING</td>
<td>converter from &quot;Electric Book&quot; (popular in Japan) to a single JIS X 4081 format (a subset of the EPWING V1)</td>
</tr>
<tr>
<td>calibre</td>
<td>V:7, I:28</td>
<td>61995</td>
<td>any → EPUB</td>
<td>e-book converter and library management</td>
</tr>
</tbody>
</table>

Table 11.18: List of miscellaneous data conversion tools

You can also extract data from RPM format with the following.

```
$ rpm2cpio file.src.rpm | cpio --extract
```
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.

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. You may need to include the contrib and non-free archives in addition to the main archive since some GFDL documentations are not considered to be DFSG compliant.

Please consider to use version control system tools. See Section 10.5.

⚠️ 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.

```bash
#!/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 non-interactive POSIX shell "/bin/sh" is a symlink pointing to /usr/bin/dash and used by many system programs.
- The default interactive POSIX shell is /usr/bin/bash.

Avoid writing a shell script with bashisms or zshisms to make it portable among all POSIX shells. You can check it using checkbashisms(1).

<table>
<thead>
<tr>
<th>Good: POSIX</th>
<th>Avoid: bashism</th>
</tr>
</thead>
<tbody>
<tr>
<td>if [ &quot;$foo&quot; = &quot;$bar&quot; ] ; then …</td>
<td>if [ &quot;$foo&quot; == &quot;$bar&quot; ] ; then …</td>
</tr>
<tr>
<td>diff -u file.c.orig file.c</td>
<td>diff -u file.c{.orig,}</td>
</tr>
<tr>
<td>mkdir /foobar /foobaz</td>
<td>mkdir /foo{bar,baz}</td>
</tr>
<tr>
<td>funcname() { … }</td>
<td>function funcname() { … }</td>
</tr>
<tr>
<td>octal format: &quot;\377&quot;</td>
<td>hexadecimal format: &quot;\xff&quot;</td>
</tr>
</tbody>
</table>

Table 12.1: 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.

<table>
<thead>
<tr>
<th>shell parameter</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0</td>
<td>name of the shell or shell script</td>
</tr>
<tr>
<td>$1</td>
<td>first (1st) shell argument</td>
</tr>
<tr>
<td>$9</td>
<td>ninth (9th) shell argument</td>
</tr>
<tr>
<td>$#</td>
<td>number of positional parameters</td>
</tr>
<tr>
<td>&quot;$*&quot;</td>
<td>&quot;$1 $2 $3 $4 ...&quot;</td>
</tr>
<tr>
<td>&quot;$@&quot;</td>
<td>&quot;$1&quot; &quot;$2&quot; &quot;$3&quot; &quot;$4&quot; &quot;...&quot;</td>
</tr>
<tr>
<td>$?</td>
<td>exit status of the most recent command</td>
</tr>
<tr>
<td>$$</td>
<td>PID of this shell script</td>
</tr>
<tr>
<td>$!</td>
<td>PID of most recently started background job</td>
</tr>
</tbody>
</table>

Table 12.2: List of shell parameters

Basic parameter expansions to remember are as follows.

Here, the colon ":" in all of these operators is actually optional.
### Shell parameter expansions

<table>
<thead>
<tr>
<th>Parameter expression form</th>
<th>Value if ( \text{var} ) is set</th>
<th>Value if ( \text{var} ) is not set</th>
</tr>
</thead>
<tbody>
<tr>
<td>${\text{var}:-string}</td>
<td>$\text{var}&quot;</td>
<td>&quot;string&quot;</td>
</tr>
<tr>
<td>${\text{var}:+string}</td>
<td>&quot;string&quot;</td>
<td>&quot;null&quot;</td>
</tr>
<tr>
<td>${\text{var}:=string}</td>
<td>$\text{var}&quot;</td>
<td>&quot;string&quot; (and run &quot;var=string&quot;)</td>
</tr>
<tr>
<td>${\text{var}?:string}</td>
<td>$\text{var}&quot;</td>
<td>echo &quot;string&quot; to stderr (and exit with error)</td>
</tr>
</tbody>
</table>

Table 12.3: List of shell parameter expansions

- **with ":"** = operator test for exist and not null
- **without ":"** = operator test for exist only

### Key shell parameter substitutions

<table>
<thead>
<tr>
<th>Parameter substitution form</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>${\text{var}}%suffix</td>
<td>remove smallest suffix pattern</td>
</tr>
<tr>
<td>${\text{var}}%%suffix</td>
<td>remove largest suffix pattern</td>
</tr>
<tr>
<td>${\text{var}}#prefix</td>
<td>remove smallest prefix pattern</td>
</tr>
<tr>
<td>${\text{var}}##prefix</td>
<td>remove largest prefix pattern</td>
</tr>
</tbody>
</table>

Table 12.4: List of key shell parameter substitutions

### Shell conditionals

Each command returns an exit status which can be used for conditional expressions.

- **Success:** 0 ("True")
- **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.

- "\text{command} \&\& \text{if\_success\_run\_this\_command\_too} \mid\mid \text{true}"  
- "\text{command} \mid\mid \text{if\_not\_success\_run\_this\_command\_too} \mid\mid \text{true}"  
- A multi-lines script snippet as the following

```bash
if [ conditional_expression ]; then
  if_success_run_this_command
else
  if_not_success_run_this_command
fi
```

Here trailing "\mid\mid \text{true}" was needed to ensure this shell script does not exit at this line accidentally when shell is invoked with "-e" flag.

**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.

- "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 9.4.9.

### 12.1.5 Shell environment variables

Some popular environment variables for the normal shell command prompt may not be available under the execution environment of your script.

- For "$USER", use "$(id -un)"
- For "$UID", use "$(id -u)"
- For "$HOME", use "$(getent passwd $(id -u) | cut -d ':' -f 6)" (this works also on Section 4.5.2)
12.1.6 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.
### 12.1.7 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.

**Tip**

Although moreutils may not exist outside of Debian, it offers interesting small programs. Most notable one is sponge (8) which is quite useful when you wish to overwrite original file.

See Section 1.6 for examples.

### 12.2 Scripting in interpreted languages

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dash</td>
<td>V:881, I:996</td>
<td>191</td>
<td>small and fast POSIX-compliant shell for sh</td>
</tr>
<tr>
<td>coreutils</td>
<td>V:879, I:999</td>
<td>18062</td>
<td>GNU core utilities</td>
</tr>
<tr>
<td>grep</td>
<td>V:783, I:999</td>
<td>1267</td>
<td>GNU grep, egrep and fgrep</td>
</tr>
<tr>
<td>sed</td>
<td>V:786, I:999</td>
<td>987</td>
<td>GNU sed</td>
</tr>
<tr>
<td>mawk</td>
<td>V:429, I:997</td>
<td>284</td>
<td>small and fast awk</td>
</tr>
<tr>
<td>debianutils</td>
<td>V:900, I:999</td>
<td>224</td>
<td>miscellaneous utilities specific to Debian</td>
</tr>
<tr>
<td>bsdutils</td>
<td>V:551, I:999</td>
<td>356</td>
<td>basic utilities from 4.4BSD-Lite</td>
</tr>
<tr>
<td>bsdextrautils</td>
<td>V:557, I:673</td>
<td>339</td>
<td>extra utilities from 4.4BSD-Lite</td>
</tr>
<tr>
<td>moreutils</td>
<td>V:14, I:37</td>
<td>244</td>
<td>additional Unix utilities</td>
</tr>
</tbody>
</table>

**Tip**

When you wish to automate a task on Debian, you should script it with an interpreted language first. The guide line for the choice of the interpreted language is:
• Use **dash**, if the task is a simple one which combines CLI programs with a shell program.
• Use **python3**, if the task isn’t a simple one and you are writing it from scratch.
• Use **perl**, **tcl**, **ruby**, ... if there is an existing code using one of these languages on Debian which needs to be touched up to do the task.

If the resulting code is too slow, you can rewrite only the critical portion for the execution speed in a compiled language and call it from the interpreted language.

### 12.2.1 Debugging interpreted language codes

Most interpreters offer basic syntax check and code tracing functionalities.

- **“dash -n script.sh”** - Syntax check of a Shell script
- **“dash -x script.sh”** - Trace a Shell script
- **“python -m py_compile script.py”** - Syntax check of a Python script
- **“python -mtrace --trace script.py”** - Trace a Python script
- **“perl -I .. /libpath -c script.pl”** - Syntax check of a Perl script
- **“perl -d:Trace script.pl”** - Trace a Perl script

For testing code for dash, try Section 9.1.4 which accommodates bash-like interactive environment.

For testing code for perl, try REPL environment for Perl which accommodates Python-like REPL (=READ + EVAL + PRINT + LOOP) environment for Perl.

### 12.2.2 GUI program with the shell script

The shell script can be improved to create an attractive GUI program. The trick is to use one of so-called dialog programs instead of dull interaction using echo and read commands.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x11-utils</td>
<td>V:193, I:569</td>
<td>712</td>
<td>xmessage(1): display a message or query in a window (X)</td>
</tr>
<tr>
<td>whiptail</td>
<td>V:285, I:996</td>
<td>56</td>
<td>displays user-friendly dialog boxes from shell scripts (newt)</td>
</tr>
<tr>
<td>dialog</td>
<td>V:12, I:102</td>
<td>1224</td>
<td>displays user-friendly dialog boxes from shell scripts (ncurses)</td>
</tr>
<tr>
<td>zenity</td>
<td>V:79, I:367</td>
<td>155</td>
<td>display graphical dialog boxes from shell scripts (GTK)</td>
</tr>
<tr>
<td>ssft</td>
<td>V:0, I:0</td>
<td>75</td>
<td>Shell Scripts Frontend Tool (wrapper for zenity, kdialog, and dialog with gettext)</td>
</tr>
<tr>
<td>gettext</td>
<td>V:56, I:283</td>
<td>5818</td>
<td>&quot;/usr/bin/gettext.sh&quot;: translate message</td>
</tr>
</tbody>
</table>

Table 12.9: List of dialog programs

Here is an example of GUI program to demonstrate how easy it is just with a shell script.

This script uses zenity to select a file (default /etc/motd) and display it.

GUI launcher for this script can be created following Section 9.4.10.

```bash
#!/bin/sh -e
# Copyright (C) 2021 Osamu Aoki <osamu@debian.org>, Public Domain
# vim: set sw=2 sts=2 et:
DATA_FILE=${(zenity --file-selection --filename="/etc/motd" --title="Select a file to check ←")} || 
( echo "E: File selection error" >&2 ; exit 1 )
```
# Check size of archive
if ( file -ib "$DATA_FILE" | grep -qe '^text/' ) ; then
    zenity --info --title="Check file: $DATA_FILE" --width 640 --height 400 \
    --text="$(head -n 20 "$DATA_FILE")"
else
    zenity --info --title="Check file: $DATA_FILE" --width 640 --height 400 \
    --text="The data is MIME=$(file -ib "$DATA_FILE")"
fi

This kind of approach to GUI program with the shell script is useful only for simple choice cases. If you are to write any program with complexities, please consider writing it on more capable platform.

## 12.2.3 Custom actions for GUI filer

GUI filer programs can be extended to perform some popular actions on selected files using additional extension packages. They can also made to perform very specific custom actions by adding your specific scripts.

- For GNOME, see NautilusScriptsHowto.
- For KDE, see Creating Dolphin Service Menus.
- For Xfce, see Thunar - Custom Actions and https://help.ubuntu.com/community/ThunarCustomActions.
- For LXDE, see Custom Actions.

## 12.2.4 Perl short script madness

In order to process data, sh needs to spawn sub-process running cut, grep, sed, etc., and is slow. On the other hand, perl has internal capabilities to process data, and is fast. So many system maintenance scripts on Debian use perl.

Let’s think following one-liner AWK script snippet and its equivalents in Perl.

```
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 )' |
```

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.
- Perl execution tricks via command line options: perlrun(1)
- Perl special variables: perlvar(1)

This flexibility is the strength of Perl. At the same time, this allows us to create cryptic and tangled codes. So be careful. For more crazy Perl scripts, Perl Golf may be interesting.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcc</td>
<td>V:165, I:551</td>
<td>47</td>
<td>GNU C compiler</td>
</tr>
<tr>
<td>libc6-dev</td>
<td>V:249, I:567</td>
<td>12050</td>
<td>GNU C Library: Development Libraries and Header Files</td>
</tr>
<tr>
<td>g++</td>
<td>V:55, I:499</td>
<td>14</td>
<td>GNU C++ compiler</td>
</tr>
<tr>
<td>libstdc++-10-dev</td>
<td>V:18, I:193</td>
<td>17537</td>
<td>GNU Standard C++ Library v3 (development files)</td>
</tr>
<tr>
<td>cpp</td>
<td>V:333, I:730</td>
<td>30</td>
<td>GNU C preprocessor</td>
</tr>
<tr>
<td>gettext</td>
<td>V:56, I:263</td>
<td>5818</td>
<td>GNU Internationalization utilities</td>
</tr>
<tr>
<td>glade</td>
<td>V:0, I:5</td>
<td>1209</td>
<td>GTK User Interface Builder</td>
</tr>
<tr>
<td>valac</td>
<td>V:0, I:14</td>
<td>724</td>
<td>C# like language for the GObject system</td>
</tr>
<tr>
<td>flex</td>
<td>V:7, I:74</td>
<td>1241</td>
<td>LEX-compatible fast lexical analyzer generator</td>
</tr>
<tr>
<td>bison</td>
<td>V:7, I:81</td>
<td>3116</td>
<td>YACC-compatible parser generator</td>
</tr>
<tr>
<td>susv2</td>
<td>I:0</td>
<td>16</td>
<td>fetch &quot;The Single UNIX Specifications v2&quot;</td>
</tr>
<tr>
<td>susv3</td>
<td>I:0</td>
<td>16</td>
<td>fetch &quot;The Single UNIX Specifications v3&quot;</td>
</tr>
<tr>
<td>golang</td>
<td>I:20</td>
<td>11</td>
<td>Go programming language compiler</td>
</tr>
<tr>
<td>rustc</td>
<td>V:3, I:14</td>
<td>8860</td>
<td>Rust systems programming language</td>
</tr>
<tr>
<td>haskell-platform</td>
<td>I:1</td>
<td>12</td>
<td>Standard Haskell libraries and tools</td>
</tr>
<tr>
<td>gfortran</td>
<td>V:7, I:65</td>
<td>16</td>
<td>GNU Fortran 95 compiler</td>
</tr>
<tr>
<td>fpc</td>
<td>I:2</td>
<td>102</td>
<td>Free Pascal</td>
</tr>
</tbody>
</table>

Table 12.10: List of compiler related packages

12.3 Coding in compiled languages

Here, Section 12.3.3 and Section 12.3.4 are included to indicate how compiler-like program can be written in C language by compiling higher level description into C language.

12.3.1 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)

12.3.2 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>
EOF
```
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).

### 12.3.3 Flex — a better Lex

Flex is a Lex-compatible fast lexical analyzer generator.

Tutorial for flex(1) can be found in “info flex”.

Many simple examples can be found under /usr/share/doc/flex/examples/”.

### 12.3.4 Bison — a better Yacc

Several packages provide a Yacc-compatible lookahead LR parser or LALR parser generator in Debian.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bison</td>
<td>V:7,I:81</td>
<td>3116</td>
<td>GNU LALR parser generator</td>
</tr>
<tr>
<td>byacc</td>
<td>V:0,I:4</td>
<td>258</td>
<td>Berkeley LALR parser generator</td>
</tr>
<tr>
<td>btyacc</td>
<td>V:0,I:0</td>
<td>243</td>
<td>backtracking parser generator based on byacc</td>
</tr>
</tbody>
</table>

Table 12.11: 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 "yparse()" which calls "yylex()", usually created with Flex.

Here is an example to create a simple terminal calculator program.

Let’s create example.y:

/* calculator source for bison */
%
#include <stdio.h>
extern int yylex(void);
extern int yyerror(char *);
%
/* declare tokens */
%token NUMBER
%token OP_ADD OP_SUB OP_MUL OP_RGT OP_LFT OP_EQU

calc:
  | calc exp OP_EQU { printf("Y: RESULT = %d\n", $2); }  
  |
exp: factor
  | exp OP_ADD factor { $$ = $1 + $3; }  
  | exp OP_SUB factor { $$ = $1 - $3; }  
  |
factor: term
  | factor OP_MUL term { $$ = $1 * $3; }  
  |
term: NUMBER
  | OP_LFT exp OP_RGT { $$ = $2; }  
  |
%
int main(int argc, char **argv)
{
  yyparse();
}
int yyerror(char *s)
{
  fprintf(stderr, "error: '%s'\n", s);
}

Let's create, example.l:

/* calculator source for flex */
{%
#include "example.tab.h"
%
%
[0-9]+ { printf("L: NUMBER = %s\n", yytext); yylval = atoi(yytext); return NUMBER; }  
"+" { printf("L: OP_ADD\n"); return OP_ADD; }
"-" { printf("L: OP_SUB\n"); return OP_SUB; }
"*" { printf("L: OP_MUL\n"); return OP_MUL; }
"(" { printf("L: OP_LFT\n"); return OP_LFT; }
")" { printf("L: OP_RGT\n"); return OP_RGT; }
"=" { printf("L: OP_EQU\n"); return OP_EQU; }
"exit" { printf("L: exit\n"); return YYEOF; } /* YYEOF = 0 */  
. { /* ignore all other */ }
%

Then execute as follows from the shell prompt to try this:

$ bison -d example.y
$ flex example.l
$ gcc -lfl example.tab.c lex.yy.c -o example
$ ./example
$ ./example
1 + 2 * ( 3 + 1 ) =
Debian Reference

12.4 Static code analysis tools

Lint like tools can help automatic static code analysis.

Indent like tools can help human code reviews by reformatting source codes consistently.

Ctags like tools can help human code reviews by generating an index (or tag) file of names found in source codes.

Tip
Configuring your favorite editor (emacs or vim) to use asynchronous lint engine plugins helps your code writing. These plugins are getting very powerful by taking advantage of Language Server Protocol. Since they are moving fast, using their upstream code instead of Debian package may be a good option.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vim-ale</td>
<td>I:0</td>
<td>2591</td>
<td>Asynchronous Lint Engine for Vim 8 and NeoVim</td>
</tr>
<tr>
<td>vim-syntastic</td>
<td>I:3</td>
<td>1379</td>
<td>Syntax checking hacks for vim</td>
</tr>
<tr>
<td>elpa-flycheck</td>
<td>V:0,I:1</td>
<td>808</td>
<td>modern on-the-fly syntax checking for Emacs</td>
</tr>
<tr>
<td>elpa-relint</td>
<td>V:0,I:0</td>
<td>144</td>
<td>Emacs Lisp regexp mistake finder</td>
</tr>
<tr>
<td>cppcheck-gui</td>
<td>V:0,I:1</td>
<td>6941</td>
<td>tool for static C/C++ code analysis (GUI)</td>
</tr>
<tr>
<td>shellcheck</td>
<td>V:2,I:11</td>
<td>18987</td>
<td>lint tool for shell scripts</td>
</tr>
<tr>
<td>pyflakes3</td>
<td>V:2,I:15</td>
<td>24</td>
<td>passive checker of Python 3 programs</td>
</tr>
<tr>
<td>pylint</td>
<td>V:4,I:19</td>
<td>2047</td>
<td>Python code static checker</td>
</tr>
<tr>
<td>perl</td>
<td>V:7/10, I:990</td>
<td>670</td>
<td>interpreter with internal static code checker: B::Lint(3perl)</td>
</tr>
<tr>
<td>rubocop</td>
<td>V:0,I:10</td>
<td>3247</td>
<td>Ruby static code analyzer</td>
</tr>
<tr>
<td>clang-tidy</td>
<td>V:1,I:10</td>
<td>21</td>
<td>clang-based C++ linter tool</td>
</tr>
<tr>
<td>xmlindent</td>
<td>V:0,I:2</td>
<td>2320</td>
<td>tool for statically checking C programs for bugs</td>
</tr>
<tr>
<td>flawfinder</td>
<td>V:0,I:0</td>
<td>205</td>
<td>tool to examine C/C++ source code and looks for security weaknesses</td>
</tr>
<tr>
<td>black</td>
<td>V:3,I:12</td>
<td>611</td>
<td>uncompromising Python code formatter</td>
</tr>
<tr>
<td>perl tidy</td>
<td>V:0,I:4</td>
<td>2493</td>
<td>Perl script indenter and reformatter</td>
</tr>
<tr>
<td>indent</td>
<td>V:0,I:8</td>
<td>431</td>
<td>C language source code formatting program</td>
</tr>
<tr>
<td>astyle</td>
<td>V:0,I:2</td>
<td>785</td>
<td>Source code indenter for C, C++, Objective-C, C#, and Java</td>
</tr>
<tr>
<td>bcpp</td>
<td>V:0,I:0</td>
<td>111</td>
<td>C(++) beautifier</td>
</tr>
<tr>
<td>xmlindent</td>
<td>V:0,I:1</td>
<td>53</td>
<td>XML stream reformatter</td>
</tr>
<tr>
<td>global</td>
<td>V:0,I:2</td>
<td>1895</td>
<td>Source code search and browse tools</td>
</tr>
<tr>
<td>exuberant-ctags</td>
<td>V:2,I:21</td>
<td>341</td>
<td>build tag file indexes of source code definitions</td>
</tr>
</tbody>
</table>

Table 12.12: List of tools for static code analysis
12.5 Debug

Debug is an important part of programming activities. Knowing how to debug programs makes you a good Debian user who can produce meaningful bug reports.

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>gdb</td>
<td>V:14, I:96</td>
<td>11637</td>
<td>&quot;info gdb&quot; provided by gdb-doc</td>
</tr>
<tr>
<td>ddd</td>
<td>V:0, I:7</td>
<td>4105</td>
<td>&quot;info ddd&quot; provided by ddd-doc</td>
</tr>
</tbody>
</table>

Table 12.13: List of debug packages

12.5.1 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 can be found:

- “info gdb”
- “tutorial 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.

12.5.2 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), *-dbgsym packages need to be installed (e.g. coreutils-dbgsym in the case of coreutils). The source packages generate *-dbgsym packages automatically along with normal binary packages 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 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
$ apt-get source package_name
$ cd package_name*
$ sudo apt-get build-dep .

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.

### 12.5.3 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 `gdb(1)` using one of the following approaches:

- **Crash-in-GDB approach:**
  - Run the program from GDB.
  - Crash the program.
  - Type "bt" at the GDB prompt.

- **Crash-first approach:**
  - Update the "/etc/security/limits.conf" file to include the following:
    
    * soft core unlimited

    - Type "ulimit -c unlimited" to the shell prompt.
    - Run the program from this shell prompt.
    - Crash the program to produce a core dump file.
    - Load the core dump file to GDB as "gdb gdb ./program_binary core".
    - Type "bt" at the GDB prompt.

For infinite loop or frozen keyboard situation, you can force to crash the program by pressing Ctrl-\ or Ctrl-C or executing "kill -ABRT PID". (See Section 9.4.12)

### 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
12.5.4 Advanced gdb commands

<table>
<thead>
<tr>
<th>command</th>
<th>description for command objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>(gdb) thread apply all bt</td>
<td>get a backtrace for all threads for multi-threaded program</td>
</tr>
<tr>
<td>(gdb) bt full</td>
<td>get parameters came on the stack of function calls</td>
</tr>
<tr>
<td>(gdb) thread apply all bt full</td>
<td>get a backtrace and parameters as the combination of the preceding options</td>
</tr>
<tr>
<td>(gdb) thread apply all bt full 10</td>
<td>get a backtrace and parameters for top 10 calls to cut off irrelevant output</td>
</tr>
<tr>
<td>(gdb) set logging on</td>
<td>write log of gdb output to a file (the default is “gdb.txt”)</td>
</tr>
</tbody>
</table>

Table 12.14: List of advanced gdb commands

12.5.5 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 9.4.6.

12.5.6 Dynamic call tracing tools

There are several dynamic call tracing tools available in Debian. See Section 9.4.

12.5.7 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.

12.5.8 Memory leak detection tools

There are several memory leak detection tools available in Debian.

12.5.9 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.
**Table 12.15: List of memory leak detection tools**

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>libc6-dev</td>
<td>V:249,I:567</td>
<td>12050</td>
<td>mtrace(1): malloc debugging functionality in glibc</td>
</tr>
<tr>
<td>valgrind</td>
<td>V:6,I:36</td>
<td>77683</td>
<td>memory debugger and profiler</td>
</tr>
<tr>
<td>electric-fence</td>
<td>V:0,I:3</td>
<td>73</td>
<td>malloc(3) debugger</td>
</tr>
<tr>
<td>libdmalloc5</td>
<td>V:0,I:2</td>
<td>393</td>
<td>debug memory allocation library</td>
</tr>
<tr>
<td>duma</td>
<td>V:0,I:10</td>
<td>293</td>
<td>library to detect buffer overruns and under-runs in C and C++ programs</td>
</tr>
<tr>
<td>leaktracer</td>
<td>V:0,I:2</td>
<td>56</td>
<td>memory-leak tracer for C++ programs</td>
</tr>
</tbody>
</table>

**Table 12.16: List of build tool packages**

<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>make</td>
<td>V:152,I:557</td>
<td>1592</td>
<td>&quot;info make&quot; provided by make-doc</td>
</tr>
<tr>
<td>autoconf</td>
<td>V:30,I:234</td>
<td>2025</td>
<td>&quot;info autoconf&quot; provided by autoconf-doc</td>
</tr>
<tr>
<td>automake</td>
<td>V:30,I:233</td>
<td>1837</td>
<td>&quot;info automake&quot; provided by automake1.10-doc</td>
</tr>
<tr>
<td>libtool</td>
<td>V:25,I:217</td>
<td>1213</td>
<td>&quot;info libtool&quot; provided by libtool-doc</td>
</tr>
<tr>
<td>cmake</td>
<td>V:17,I:114</td>
<td>36146</td>
<td>cmake(1) cross-platform, open-source make system</td>
</tr>
<tr>
<td>ninja-build</td>
<td>V:6,I:38</td>
<td>428</td>
<td>ninja(1) small build system closest in spirit to Make</td>
</tr>
<tr>
<td>meson</td>
<td>V:3,I:21</td>
<td>3685</td>
<td>meson(1) high productivity build system on top of ninja</td>
</tr>
<tr>
<td>xutils-dev</td>
<td>V:0,I:9</td>
<td>1484</td>
<td>imake(1), xmkmf(1), etc.</td>
</tr>
</tbody>
</table>

### 12.6 Build tools

**Make** is a utility to maintain groups of programs. Upon execution of `make(1)`, *make* reads 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.

Run "make -p -f /dev/null" to see automatic internal rules.

**Autotools** is a suite of programming tools designed to assist in making source code packages portable to many Unix-like systems.
<table>
<thead>
<tr>
<th>automatic variable</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$@</td>
<td>target</td>
</tr>
<tr>
<td>$&lt;</td>
<td>first prerequisite</td>
</tr>
<tr>
<td>$?</td>
<td>all newer prerequisites</td>
</tr>
<tr>
<td>$^</td>
<td>all prerequisites</td>
</tr>
<tr>
<td>$^</td>
<td>“%” matched stem in the target pattern</td>
</tr>
</tbody>
</table>

Table 12.17: List of make automatic variables

<table>
<thead>
<tr>
<th>variable expansion</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>foo1 := bar</td>
<td>one-time expansion</td>
</tr>
<tr>
<td>foo2 = bar</td>
<td>recursive expansion</td>
</tr>
<tr>
<td>foo3 += bar</td>
<td>append</td>
</tr>
</tbody>
</table>

Table 12.18: List of make variable expansions

- **Autoconf** is a tool to produce a shell script "configure" from "configure.ac".
  - "configure" is used later to produce "Makefile" from "Makefile.in" template.
- **Automake** is a tool to produce "Makefile.in" from "Makefile.am".
- **Libtool** is a shell script to address the software portability problem when compiling shared libraries from source code.

### 12.6.2.1 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 # this compiles program
$ sudo make install # this installs the files in the system
```

### 12.6.2.2 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
$ sudo 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.
12.6.3 Meson

The software build system has been evolving:

- **Autotools** on the top of **Make** has been the de facto standard for the portable build infrastructure since 1990s. This is extremely slow.

- **CMake** initially released in 2000 improved speed significantly but was originally built on the top of inherently slow **Make**. (Now **Ninja** can be its backend.)

- **Ninja** initially released in 2012 is meant to replace Make for the further improved build speed and is designed to have its input files generated by a higher-level build system.

- **Meson** initially released in 2013 is the new popular and fast higher-level build system which uses **Ninja** as its backend.

See documents found at "The Meson Build system" and "The Ninja build system".

12.7 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.

12.8 The source code translation

There are programs to convert source codes.
<table>
<thead>
<tr>
<th>package</th>
<th>popcon</th>
<th>size</th>
<th>keyword</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>perl</td>
<td>V:710, I:990</td>
<td>670</td>
<td>AWK → PERL</td>
<td>convert source codes from AWK to PERL: a2p(1)</td>
</tr>
<tr>
<td>f2c</td>
<td>V:0, I:3</td>
<td>442</td>
<td>FORTRAN → C</td>
<td>convert source codes from FORTRAN 77 to C/C++: f2c(1)</td>
</tr>
<tr>
<td>intel2gas</td>
<td>V:0, I:0</td>
<td>178</td>
<td>intel → gas</td>
<td>converter from NASM (Intel format) to the GNU Assembler (GAS)</td>
</tr>
</tbody>
</table>

Table 12.19: List of source code translation tools

### 12.9 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 9.11.4 to understand basic chroot techniques
- debuild(1), and sbuild(1)
- Section 12.5.2 for recompiling for debugging
- Guide for Debian Maintainers (the debmake-doc package)
- Debian Developer’s Reference (the developers-reference package)
- Debian Policy Manual (the debian-policy 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 2.105)" (2023-11-02 11:58:17 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, 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.

I have updated "Debian Reference (version 2)" to address new topics (Systemd, Wayland, IMAP, PipeWire, Linux kernel 5.10) and removed outdated topics (SysV init, CVS, Subversion, SSH protocol 1, Linux kernels before 2.5). References to Jessie 8 (2015-2020) release situation or older are mostly removed.

This "Debian Reference (version 2.105)" (2023-11-02 11:58:17 UTC) covers mostly Bookworm (=stable) and Trixie (=testing) Debian releases.

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)
Debian Reference

- 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 as well as upstream web pages and Wikipedia documents 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 2.105)" (2023-11-02 11:58:17 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 DocBook XML files. This Docbook XML source are converted to HTML, plain text, PostScript, and PDF. (Some formats may be skipped for distribution.)