Chapter

7

Solving Problems and Finding Relevant Information

Contents

For an administrator, the most important skill is to be able to cope with any situation, known or unknown. This chapter gives a number of methods that will — hopefully — allow you to isolate the cause of any problem that you will encounter, so that you may be able to resolve them.

7.1. Documentation Sources

Before you can understand what is really going on when there is a problem, you need to know the theoretical role played by each program involved in the problem. To do this, the best reflex to have is consult their documentation; but since these documentations are many and can be scattered far and wide, you should know all the places where they can be found.

7.1.1. Manual Pages

CULTURE

RTFM

This acronym stands for "Read the F***ing Manual", but can also be expanded in a friendlier variant, "Read the Fine Manual". This phrase is sometimes used in (terse) responses to questions from newbies. It is rather abrupt, and betrays a certain annoyance at a question asked by someone who has not even bothered to read the documentation. Some say that this classic response is better than no response at all (since it indicates that the documentation contains the information sought), or than a more verbose and angry answer.

In any case, if someone responds "RTFM" to you, it is often wise not to take offense. Since this answer may be perceived as vexing, you might want to try and avoid receiving it. If the information that you need is not in the manual, which can happen, you might want to say so, preferably in your initial question. You should also describe the various steps that you have personally taken to find information before you raised a question on a forum. Following Eric Raymond's guidelines is a good way to avoid the most common mistakes and get useful answers.

➨ `http://catb.org/~esr/faqs/smart-questions.html`

Manual pages, while relatively terse in style, contain a great deal of essential information. We will quickly go over the command for viewing them, provided by the *man-db* package. Simply type `man` *`manual-page`* — the manual page usually goes by the same name as the command whose documentation is sought. For example, to learn about the possible options for the `cp` command, you would type the `man cp` command at the shell prompt (see sidebar "The shell, a command line interpreter" page 148).

BACK TO BASICS

The shell, a command line interpreter

A command line interpreter, also called a "shell", is a program that executes commands that are either entered by the user or stored in a script. In interactive mode, it displays a prompt (usually ending in `$` for a normal user, or by `#` for an administrator) indicating that it is ready to read a new command. appendix B, "Short Remedial Course" page 475 describes the basics of using the shell.

The default and most commonly used shell is `bash` (Bourne Again SHell), but there are others, including `dash`, `csh`, `tcsh` and `zsh`.

Among other things, most shells offer help (type `help`) and assistance during input at the prompt, such as the completion of command or file names (which you can generally activate by pressing the tab key), or recalling previous commands (history management; i.e. check out the mappings for "page up" and "page down" in `/etc/inputrc`).

Man pages not only document commands and programs accessible from the command line, but also configuration files, system calls, library functions, and so forth. Sometimes names can collide. For example, the shell's `read` command has the same name as the `read` system call. This is why manual pages are organized in numbered sections:

1 commands that can be executed from the command line;

2 system calls (functions provided by the kernel);

3 library functions (provided by system libraries);

4 devices (on Unix-like systems, these are special files, usually placed in the `/dev/` directory);

5 configuration files (formats and conventions);

6 games;

7 sets of macros and standards;

8 system administration commands;

9 kernel routines.

It is possible to specify the section of the manual page that you are looking for: to view the documentation for the `read` system call, you would type `man 2 read`. When no section is explicitly specified, the first section that has a manual page with the requested name will be shown. Thus, `man shadow` returns `shadow(5)` because there are no manual pages for *shadow* in sections 1 to 4.

TIP **whatis**

If you do not want to look at the full manual page, but only a short description to confirm that it is what you are looking for, simply enter `whatis` *`command`*.

```
$ whatis scp
scp (1)     - secure copy (remote file copy program)
```

This short description is included in the *NAME* section at the beginning of all manual pages.

Of course, if you do not know the names of the commands, the manual is not going to be of much use to you. This is the purpose of the `apropos` command, which helps you conduct a search in the manual pages, or more specifically in their short descriptions. Each manual page begins essentially with a one line summary. `apropos` returns a list of manual pages whose summary mentions the requested keyword(s). If you choose them well, you will find the name of the command that you need.

Example 7.1 *Finding cp with apropos*

```
$ apropos "copy file"
cp (1)               - copy files and directories
cpio (1)             - copy files to and from archives
hpcopy (1)           - copy files from an HFS+ volume
install (1)          - copy files and set attributes
ntfscp (8)           - copy file to an NTFS volume.
```

TIP
Browsing by following links

Many manual pages have a "SEE ALSO" section, usually at the end. It refers to other manual pages relevant to similar commands, or to external documentation. In this way, it is possible to find relevant documentation even when the first choice is not optimal.

The `man` command is not the only means of consulting the manual pages, since `khelpcenter` and `konqueror` (by KDE) and `yelp` (under GNOME) programs also offer this possibility. There is also a web interface, provided by the `man2html` package, which allows you to view manual pages in a web browser. On a computer where this package is installed, use this URL after following the instructions in `/usr/share/doc/man2html/README.Debian`:

➨ `http://localhost/cgi-bin/man/man2html`

This utility requires a web server. This is why you should choose to install this package on one of your servers: all users of the local network could benefit from this service (including non-Linux machines), and this will allow you not to set up an HTTP server on each workstation. If your server is also accessible from other networks, it may be desirable to restrict access to this service only to users of the local network.

Last but not least, you can view all manual pages available in Debian (even those that are not installed on your machine) on the manpages.debian.org service. It offers each manual page in multiple versions, one for each Debian release.

➨ `https://manpages.debian.org`

DEBIAN POLICY
Required man pages

Debian requires each program to have a manual page. If the upstream author does not provide one, the Debian package maintainer will usually write a minimal page that will at the very least direct the reader to the location of the original documentation.

7.1.2. *info* Documents

The GNU project has written manuals for most of its programs in the *info* format; this is why many manual pages refer to the corresponding *info* documentation. This format offers some

advantages, but the default program to view these documents (it is called `info`) is also slightly more complex. You would be well advised to use `pinfo` instead (from the *pinfo* package).

The *info* documentation has a hierarchical structure, and if you invoke `pinfo` without parameters, it will display a list of the nodes available at the first level. Usually, nodes bear the name of the corresponding commands.

With `pinfo` navigating between these nodes is easy to achieve with the arrow keys. Alternatively, you could also use a graphical browser, which is a lot more user-friendly. Again, `konqueror` and `yelp` work; the `info2www` package also provides a web interface.

➡ `http://localhost/cgi-bin/info2www`

Note that the *info* system is not suitable for translation, unlike the `man` page system. *info* documents are thus almost always in English. However, when you ask the `pinfo` program to display a non-existing *info* page, it will fall back on the *man* page by the same name (if it exists), which might be translated.

7.1.3. Specific Documentation

Each package includes its own documentation. Even the least well documented programs generally have a `README` file containing some interesting and/or important information. This documentation is installed in the `/usr/share/doc/`*package*`/` directory (where *package* represents the name of the package). If the documentation is particularly large, it may not be included in the program's main package, but might be offloaded to a dedicated package which is usually named *package*-doc. The main package generally recommends the documentation package so that you can easily find it.

The `/usr/share/doc/`*package*`/` directory also contains some files provided by Debian which complete the documentation by specifying the package's particularities or improvements compared to a traditional installation of the software. The `README.Debian` file also indicates all of the adaptations that were made to comply with the Debian Policy. The `changelog.Debian.gz` file allows the user to follow the modifications made to the package over time: it is very useful to try to understand what has changed between two installed versions that do not have the same behavior. Finally, there is sometimes a `NEWS.Debian.gz` file which documents the major changes in the program that may directly concern the administrator (see section 6.7.2, "Handling Problems after an Upgrade" page 135).

7.1.4. Websites

In most cases, free software programs have websites that are used to distribute it and to bring together the community of its developers and users. These sites are frequently loaded with relevant information in various forms: official documentation, FAQ (Frequently Asked Questions), mailing list archives, etc. Problems that you may encounter have often already been the subject of many questions; the FAQ or mailing list archives may have a solution for it. A good mastery of search engines will prove immensely valuable to find relevant pages quickly (by restricting the

search to the Internet domain or sub-domain dedicated to the program). If the search returns too many pages or if the results do not match what you seek, you can add the keyword **debian** to limit results and target relevant information.

TIP **From error to solution**

If the software returns a very specific error message, enter it into the search engine (between double quotes, ", in order to search not for individual keywords, but for the complete phrase). In most cases, the first links returned will contain the answer that you need.

In other cases, you will get very general errors, such as "Permission denied". In this case, it is best to check the permissions of the elements involved (files, user ID, groups, etc.).

If you do not know the address for the software's website, there are various means of getting it. First, check if there is a Homepage field in the package's meta-information (`apt show` *package*). Alternately, the package description may contain a link to the program's official website. If no URL is indicated, look at `/usr/share/doc/`*package*`/copyright`. The Debian maintainer generally indicates in this file where they got the program's source code, and this is likely to be the website that you need to find. If at this stage your search is still unfruitful, consult a free software directory, such as FSF's Free Software Directory, or search directly with a search engine, such as Google, DuckDuckGo, Yahoo, etc.

➨ https://directory.fsf.org/wiki/Main_Page

You might also want to check the Debian wiki, a collaborative website where anybody, even simple visitors, can make suggestions directly from their browsers. It is used equally by developers who design and specify their projects, and by users who share their knowledge by writing documents collaboratively.

➨ https://wiki.debian.org/

7.1.5. Tutorials (*HOWTO*)

A *HOWTO* is a document that describes, in concrete terms and step by step, "how to" reach a predefined goal. The covered goals are relatively varied, but often technical in nature: for example, setting up IP Masquerading, configuring software RAID, installing a Samba server, etc. These documents often attempt to cover all of the potential problems likely to occur during the implementation of a given technology.

Many such tutorials are managed by the Linux Documentation Project (LDP), whose website hosts all of these documents:

➨ https://www.tldp.org/

Debian also provides tutorials for its users:

➨ https://www.debian.org/doc/

All these documents should be taken with a grain of salt. They are often several years old; the information they contain is sometimes obsolete. This phenomenon is even more frequent for

their translations, since updates are neither systematic nor instant after the publication of a new version of the original documents. Further many tutorials nowadays are provided by bloggers, sharing their individual solution with the interested reader. They often lack important information, i.e. the reason why some configuration has been chosen over another, or why some option has been enabled or disabled. Because blogging and creating own websites made it so easy to share, many of these often short tutorials exist, but only a few are actively maintained and well-kept. This can make it hard, to find the "right" one for you. This is all part of the joys of working in a volunteer environment and without constraints...

7.2. Common Procedures

The purpose of this section is to present some general tips on certain operations that an administrator will frequently have to perform. These procedures will of course not cover every possible case in an exhaustive way, but they may serve as starting points for the more difficult cases.

DISCOVERY

Documentation in other languages

Often, documentation translated into a non-English language is available in a separate package with the name of the corresponding package, followed by `-lang` (where *lang* is the two-letter ISO code for the language).

For instance, the *debian-reference-fr* package is the French version of the reference guides for Debian (initially written in English by Osamu Aoki), and the *manpages-de* package contains the German version of the manual pages about using GNU/Linux.

7.2.1. Configuring a Program

When you want to configure an unknown package, you must proceed in stages. First, you should read what the package maintainer has documented. Reading `/usr/share/doc/`*`package`*`/README.Debian` will indeed allow you to learn of specific provisions made to simplify the use of the software. It is sometimes essential in order to understand the differences from the original behavior of the program, as described in the general documentation, such as howtos. Sometimes this file also details the most common errors in order for you to avoid wasting time on common problems.

Then, you should look at the software's official documentation — refer to section 7.1, "Documentation Sources" page 148 to identify the various existing documentation sources. The `dpkg -L` *`package`* command gives a list of files included in the package; you can therefore quickly identify the available documentation (as well as the configuration files, located in `/etc/`). `dpkg -s` *`package`* displays the package meta-data and shows any possible recommended or suggested packages; in there, you can find documentation or a utility that will ease the configuration of the software.

Finally, the configuration files are often self-documented by many explanatory comments detailing the various possible values for each configuration setting. So much so that it is sometimes

enough to just choose a line to activate from among those available. In some cases, examples of configuration files are provided in the `/usr/share/doc/`*`package`*`/examples/` directory. They may serve as a basis for your own configuration file.

DEBIAN POLICY

Location of examples

All examples must be installed in the `/usr/share/doc/`*`package`*`/examples/` directory. This may be a configuration file, program source code (an example of the use of a library), or a data conversion script that the administrator can use in certain cases (such as to initialize a database). If the example is specific to a particular architecture, it should be installed in `/usr/lib/`*`package`*`/examples/` and there should be a link pointing to that file in the `/usr/share/doc/`*`package`*`/examples/` directory.

7.2.2. Monitoring What Daemons Are Doing

Understanding what a daemon does is somewhat more complicated, since it does not interact directly with the administrator. To check that a daemon is actually working, you need to test it. For example, to check the Apache (web server) daemon, test it with an HTTP request.

BACK TO BASICS

Daemon

A daemon is a program that is not explicitly invoked by the user and that stays in the background, waiting for a certain condition to be met before performing a task. Many server programs are daemons, a term that explains that the letter "d" is frequently present at the end of their name (`sshd`, `smtpd`, `httpd`, etc.).

To allow such tests, each daemon generally records everything that it does, as well as any errors that it encounters, in what are called "log files" or "system logs". Logs are stored in `/var/log/` or one of its subdirectories. To know the precise name of a log file for each daemon, see its documentation. Note: a single test is not always sufficient if it does not cover all the possible usage cases; some problems only occur in particular circumstances.

TOOL

The `rsyslogd` daemon

`rsyslogd` is special: it collects logs (internal system messages) that are sent to it by other programs. Each log entry is associated with a subsystem (e-mail, kernel, authentication, etc.) and a priority; `rsyslogd` processes these two pieces of information to decide on what to do. The log message may be recorded in various log files, and/or sent to an administration console. The details are defined in the `/etc/rsyslog.conf` configuration file (documented in the manual page of the same name provided in the *rsyslog-doc* package).

Certain C functions, which are specialized in sending logs, simplify the use of the `rsyslogd` daemon. However some daemons manage their own log files (this is the case, for example, of `samba`, which implements Windows shares on Linux).

Note that when `systemd` is in use, the logs are actually collected by `systemd` before being forwarded to `rsyslogd`. They are thus also available via `systemd`'s journal and can be consulted with `journalctl` (see section 9.1.1, "The systemd init system" page 198 for details).

As a preventive operation, the administrator should regularly read the most relevant server logs. They can thus diagnose problems before they are even reported by disgruntled users. Indeed users may sometimes wait for a problem to occur repeatedly over several days before reporting it. In many cases, there are specific tools to analyze the contents of the larger log files. In particular, such utilities exist for web servers (such as `analog`, `awstats`, `webalizer` for Apache), for FTP servers, for proxy/cache servers, for firewalls, for e-mail servers, for DNS servers, and even for print servers. Other tools, such as `logcheck` (a software discussed in chapter 14, "Security" page 402), scan these files in search of alerts to be dealt with.

7.2.3. Asking for Help on a Mailing List

If your various searches haven't helped you to get to the root of a problem, it is possible to get help from other, perhaps more experienced people. This is exactly the purpose of the debian-user@lists.debian.org mailing list and its language specific siblings debian-user-*lang*@lists.debian.org. As with any community, it has rules that need to be followed. Before asking any question, you should check that your problem isn't already covered by recent discussions on the list or by any official documentation.

➨ `https://wiki.debian.org/DebianMailingLists`

➨ `https://lists.debian.org/debian-user/`

➨ `https://lists.debian.org/users.html`

BACK TO BASICS

Netiquette applies

In general, for all correspondence on e-mail lists, the rules of Netiquette should be followed. This term refers to a set of common sense rules, from common courtesy to mistakes that should be avoided.

➨ `https://tools.ietf.org/html/rfc1855`

Furthermore, for any communication channel managed by the Debian project, you are bound by the Debian Code of Conduct:

➨ `https://www.debian.org/code_of_conduct`

Once those two conditions are met, you can think of describing your problem to the mailing list. Include as much relevant information as possible: various tests conducted, documentation consulted, how you attempted to diagnose the problem, the packages concerned or those that may be involved, etc. Check the Debian Bug Tracking System (BTS, described in sidebar section 1.3.2.1, "Reporting bugs" page 14) for similar problems, and mention the results of that search, providing links to bugs found. BTS starts on:

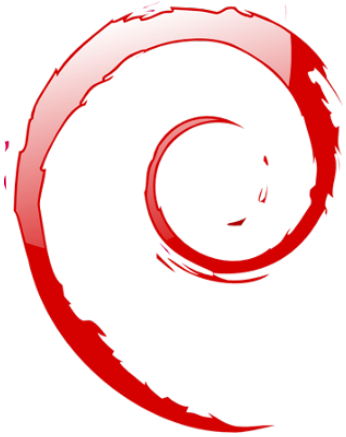
➨ `https://bugs.debian.org/`

The more courteous and precise you have been, the greater your chances are of getting an answer, or, at least, some elements of response. If you receive relevant information by private e-mail, think of summarizing this information publicly so that others can benefit. This also allows the list's archives, searched through various search engines, to show the resolution for others who may have the same question.

7.2.4. Reporting a Bug When a Problem Is Too Difficult

If all of your efforts to resolve a problem fail, it is possible that a resolution is not your responsibility, and that the problem is due to a bug in the program. In this case, the proper procedure is to report the bug to Debian or directly to the upstream developers. To do this, isolate the problem as much as possible and create a minimal test situation in which it can be reproduced. If you know which program is the apparent cause of the problem, you can find its corresponding package using the command, `dpkg -S` *`file_in_question`*. Check the Bug Tracking System (https://bugs.debian.org/*package*) to ensure that the bug has not already been reported. You can then send your own bug report, using the `reportbug` command, including as much information as possible, especially a complete description of those minimal test cases that will allow anyone to recreate the bug.

The elements of this chapter are a means of effectively resolving issues that the following chapters may bring about. Use them as often as necessary!



Keywords

Configuration
Localization
Locales
Network
Name resolution
Users
Groups
Accounts
Command-line interpreter
Shell
Printing
Bootloader
Kernel compiling