Rocky Linux Admin Guide (English version)

A book from the Documentation Team

Version : 2025/07/08

Rocky Documentation Team

Copyright © 2023 The Rocky Enterprise Software Foundation

Table of contents

1. Licence	11
2. Learning Linux with Rocky	12
3. Introduction to the Linux Operating System	14
3.1 What is an operating system?	14
3.2 Generalities UNIX - GNU/Linux	15
3.2.1 History	15
3.2.2 Market share	18
3.2.3 Architectural design	18
3.2.4 The UNIX/Linux Philosophy	20
3.3 The GNU/Linux distributions	20
3.3.1 Desktop environments	20
3.3.2 Free / Open Source	23
3.4 Areas of use	24
3.5 Shell	24
3.5.1 Generalities	24
3.5.2 Functionalities	24
3.5.3 Principle	25
3.6 Check your Knowledge	26
4. Commands for Linux Users	28
4.1 Generalities	28
4.1.1 The users	29
4.1.2 The shell	30
4.2 General commands	31
4.2.1 apropos, whatis and man commands	31
4.2.2 shutdown command	34
4.2.3 history command	34
4.2.4 Auto-complete	35
4.3 Display and Identification	35
4.3.1 clear command	35
4.3.2 echo command	36
4.3.3 date command	37
4.3.4 id, who and whoami commands	38
4.4 File Tree	39
4.4.1 pwd command	41
4.4.2 cd command	41

4.4.3	ls command	42
4.4.4	mkdir command	45
4.4.5	touch command	46
4.4.6	rmdir command	46
4.4.7	rm command	47
4.4.8	mv command	48
4.4.9	cp command	49
4.5 Vis	ualization	50
4.5.1	file command	50
4.5.2	more command	50
4.5.3	less command	51
4.5.4	cat command	51
4.5.5	tac command	52
4.5.6	head command	52
4.5.7	tail command	53
4.5.8	sort command	53
4.5.9	wc command	56
4.6 Sea	arch	57
4.6.1	find command	57
4.6.2	-exec option of the find command	57
4.6.3	whereis command	58
4.6.4	grep command	58
4.6.5	Meta-characters (wildcards)	59
4.7 Red	directs and pipes	60
4.7.1	Standard input and output	60
4.7.2	Input redirection	61
4.7.3	Output redirection	62
4.7.4	Examples of redirection	63
4.7.5	Pipes	63
4.8 Spe	ecial Points	64
4.8.1	tee command	64
4.8.2	alias and unalias commands	65
4.8.3	Aliases and Useful Functions	66
4.8.4	The character ;	68
4.9 Ch	eck your Knowledge	68
5. Advan	ced Commands for Linux users	70
5.1 uni	.q command	70

	5.2 xargs commands	72
	5.3 yum-utils package	74
	5.3.1 repoquery command	75
	5.3.2 yumdownloader command	76
	5.4 psmisc packages	76
	5.5 watch command	77
	5.6 install command	78
	5.7 tree command	80
	5.8 stat command	81
6.	. VI Text Editor	83
	6.1 vicommand	83
	6.2 Operating mode	85
	6.2.1 The Command Mode	85
	6.2.2 The Insert mode	86
	6.2.3 The Ex mode	86
	6.3 Moving the cursor	86
	6.3.1 From a character	86
	6.3.2 From the first character of a word	87
	6.3.3 From any location on a line	87
	6.4 Inserting text	88
	6.4.1 In relation to a character	88
	6.4.2 In relation to a line	88
	6.4.3 In relation to the text	89
	6.5 Characters, words and lines	89
	6.5.1 Characters	89
	6.5.2 Words	90
	6.5.3 Lines	90
	6.5.4 Cancel an action	92
	6.5.5 Cancel cancellation	92
	6.6 EX commands	92
	6.6.1 File line numbers	92
	6.6.2 Search for a string	92
	6.6.3 Replace a string	94
	6.6.4 Deletes the specified row	94
	6.6.5 File operations	95
	6.7 Other functions	96
	6.7.1 vimtutor command	96
	6.7.2 visualization mode	96

7. User Management	98
7.1 General	98
7.2 Group management	99
7.2.1 groupadd command	100
7.2.2 Command groupmod	100
7.2.3 groupdel command	101
7.2.4 /etc/group file	102
7.2.5 /etc/gshadow file	103
7.3 User management	104
7.3.1 Definition	104
7.3.2 useradd command	104
7.3.3 usermod command	107
7.3.4 userdel command	109
7.3.5 /etc/passwd file	109
7.3.6 /etc/shadow file	110
7.4 File owners	112
7.4.1 Modification commands	112
7.4.2 chgrp command	113
7.5 Guest management	114
7.5.1 gpasswd command	114
7.5.2 id command	114
7.5.3 newgrp command	115
7.6 Securing	116
7.6.1 passwd command	116
7.6.2 chage command	117
7.7 Advanced management	119
7.7.1 /etc/default/useradd file	119
7.7.2 /etc/login.defs file	120
7.7.3 /etc/skel directory	121
7.8 Identity change	122
7.8.1 su command	122
8. File System	125
8.1 Partitioning	125
8.1.1 Naming conventions for device file names	127
8.1.2 Device partition number	127
8.1.3 parted command	128
8.1.4 cfdisk command	129

	8.2 Logical Volume Manager (LVM)	130
	8.2.1 The Writing Mechanism of LVM	132
	8.2.2 LVM commands for volume management	133
	8.2.3 LVM commands to view volume information	135
	8.2.4 Preparation of the physical media	136
	8.3 Structure of a file system	136
	8.3.1 mkfs command	136
	8.3.2 Boot sector	137
	8.3.3 Super block	137
	8.3.4 Table of inodes	138
	8.3.5 Data block	139
	8.3.6 Repairing the file system	140
	8.4 Organization of a file system	140
	8.4.1 /etc/fstab file	143
	8.4.2 Mount management commands	144
	8.5 File naming convention	146
	8.5.1 Details of a file name	147
	8.6 File attributes	150
	8.6.1 Basic permissions of files and directories	151
	8.6.2 User type corresponding to basic permission	151
	8.6.3 Attribute management	152
	8.7 Default rights and mask	154
	8.7.1 umask command	155
9.	. Process Management	157
	9.1 Generalities	157
	9.2 Viewing processes	158
	9.3 Types of processes	160
	9.4 Permissions and rights	160
	9.5 Process management	161
	9.5.1 The priority of a process	162
	9.5.2 Modes of operation	162
	9.6 Process management controls	162
	9.6.1 kill command	162
	9.6.2 nohup command	163
	9.6.3 [Ctrl] + [z]	164
	9.6.4 & instruction	164
	9.6.5 fg and bg commands	164
	9.6.6 jobs command	165

165
167
167
168
169
169
171
172
172
173
173
173
174
175
176
176
176
177
177
185
186
189
189
191
191
192
192
193
194
194
194
195
195
196
196
197
201
202

11.3.2 Example of a .service file for the postfix service	204
11.3.3 Using system targets	204
11.3.4 The journald process	207
11.3.5 journalctl command	208
12. Task Management	210
12.1 Generalities	210
12.2 How the service works	211
12.3 Security	211
12.3.1 The cron.allow and cron.deny Files	212
12.3.2 Allowing a user	212
12.3.3 Prohibit a user	212
12.4 Scheduling tasks	213
12.4.1 The crontab command	213
12.4.2 Uses of crontab	214
12.5 The crontab file	215
12.5.1 Task execution process	216
13. Implementing the Network	218
13.1 Generalities	218
13.1.1 MAC address / IP address	220
13.1.2 DNS Domain	221
13.1.3 Reminder of the OSI model	221
13.2 The naming of interfaces	222
13.3 Using the ip command	223
13.4 The hostname	223
13.5 /etc/hosts file	224
13.6 /etc/nsswitch.conf file	225
13.7 /etc/resolv.conf file	225
13.8 ip command	226
13.9 DHCP configuration	226
13.10 Static configuration	228
13.11 Routing	229
13.12 Name resolution	230
13.13 Troubleshooting	230
13.13.1 dig command	232
13.13.2 getent command	233
13.13.3 ipcalc command	234
13.13.4 ss command	235
13.13.5 netstat command	235

13.13.6 IP or MAC address conflicts	236
13.14 Hot configuration	236
13.15 In summary	237
14. Software Management	239
14.1 Generalities	239
14.2 RPM: RedHat Package Manager	239
14.2.1 rpm command	240
14.3 DNF: Dandified Yum	242
14.3.1 dnf command	242
14.3.2 Other useful dnf options	247
14.3.3 How DNF works	252
14.4 DNF modules	253
14.4.1 What are modules	253
14.4.2 Listing modules	253
14.4.3 Enabling Modules	254
14.4.4 Installing packages from the module stream	255
14.4.5 Installing packages from module stream profiles	256
14.4.6 Module Removal and Reset or Switch-To	257
14.4.7 Disable a module stream	258
14.5 The EPEL repository	259
14.5.1 What is EPEL and how is it used?	259
14.5.2 Installation	259
14.5.3 Using EPEL	262
14.5.4 Conclusion	264
14.6 DNF Plugins	264
14.6.1 config-manager plugin	264
14.6.2 copr plugin	265
14.6.3 download plugin	265
14.6.4 needs-restarting plugin	265
14.6.5 versionlock plugin	266
15. Review basic permissions	267
15.1 Seven file types	268
15.2 The meaning of basic permissions	268
15.3 Special authority	269
15.3.1 ACL permissions	269
15.3.2 SetUID	275
15.3.3 SetGID	277
15.3.4 Sticky BIT	279

15.3.5	chattr 2	280
15.3.6	sudo 2	283

1. Licence

RockyLinux offers Linux courseware for trainers or people wishing to learn how to administer a Linux system on their own.

RockyLinux materials are published under Creative Commons-BY-SA. This means you are free to share and transform the material, while respecting the author's rights.

BY : **Attribution**. You must cite the name of the original author.

SA : Share Alike.

• Creative Commons-BY-SA licence : https://creativecommons.org/licenses/by-sa/ 4.0/

The documents and their sources are freely downloadable from:

- https://docs.rockylinux.org
- https://github.com/rocky-linux/documentation

Our media sources are hosted at github.com. You'll find the source code repository where the version of this document was created.

From these sources, you can generate your own personalized training material using mkdocs. You will find instructions for generating your document here.

How can I contribute to the documentation project?

You'll find all the information you need to join us on our git project home page.

We wish you all a pleasant reading and hope you enjoy the content.

2. Learning Linux with Rocky

The Administrator's Guide is a collection of educational documents focused on System Administrators. They can be used by future System Administrators trying to get up to speed, by current System Administrators who would like a refresher, or by any Linux user who'd like to learn more about the Linux environment, commands, processes, and more. Like all documents of this type, it will evolve and update over time.

We start with Introduction to Linux, which outlines Linux, distributions, and the whole ecosystem around our operating system.

User Commands contains essential commands for getting up to speed with Linux. More experienced users should also consult the following chapter on Advanced Linux Commands.

The VI Text Editor deserves its own chapter. While Linux comes with many editors, VI is one of the most powerful. Other commands sometimes use identical syntax to VI (sed comes to mind). So, knowing something about VI, or at least demystifying its essential functions (how to open a file, save, quit or quit without saving), is very important. The user will become more comfortable with the other functions of VI as they use the editor. An alternative would be to use nano which comes installed by default in Rocky Linux. While not as versatile, it is simple to use, straightforward, and gets the job done.

We can then get into the deep functioning of Linux to discover how the system addresses:

- User Management
- File Systems
- Process Management

Backup and Restoration is essential info for the System Administrator. Linux comes with many software solutions to enhance backups (rsnapshot, lsyncd, etcetera.). It is valuable to know the essential components of the backup that exist within the operating system. We investigate two tools: tar and the less widespread cpio in this chapter. System Startup is also an important read because management of the system during the boot process has evolved significantly in recent years since the arrival of the systemd.

Final chapters address Task Management, Implementing the Network, and Software Management including installation.

3. Introduction to the Linux Operating System

In this chapter, you will learn about GNU/Linux distributions.

Objectives: In this chapter, you will learn how to:

- ✓ Describe the features and possible architectures of an operating system.
- ✓ Recount the history of UNIX and GNU/Linux.
- ✓ Choose the right Linux distribution for your needs.
- Explain the philosophy of Free and Open-source Software.
- \checkmark Discover the usefulness of the shell.

🕅 generalities, linux, distributions

Knowledge: ★ Complexity: ★

Reading time: 10 minutes

3.1 What is an operating system?

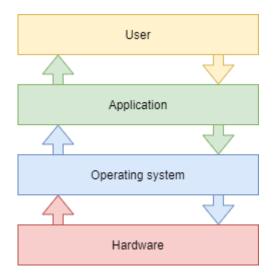
Linux, UNIX, BSD, Windows, and MacOS are all **operating systems**.

Abstract

An operating system is a set of programs that manages the available resources of a computer.

As part of this management of resources, the operating system has to:

- Manage the **physical** or **virtual** memory.
- The **physical memory** is made up of the RAM bars and the processor cache memory, which are used for program execution.
- The **virtual memory** is a location on the hard disk (the **swap** partition) that allows the unloading of the physical memory and the saving of the current state of the system during the electrical shutdown of the computer.
- Intercept **access to peripherals**. Software can rarely access hardware directly (except for graphics cards for specific needs).
- Provide applications with proper **task management**. The operating system is responsible for scheduling processes to occupy the processor
- Protect files from unauthorized access.
- Collect information about programs in use or in progress.



3.2 Generalities UNIX - GNU/Linux

3.2.1 History

UNIX

• **1964** — **1968**: MULTICS (MULTiplexed Information and Computing Service) is developed for MIT, Bell Labs (AT&T) and General Electric.

1969 — 1971: After the withdrawal of Bell (1969) and then General Electric from the project, two developers, Ken Thompson and Dennis Ritchie (joined later by Brian Kernighan), judging MULTICS to be too complex, begin development of UNIX (UNiplexed Information and Computing Service). While it was created in Assembly language, the creators of UNIX eventually developed the B language and then the C language (1971) and completely rewrote UNIX. As it was developed in 1970, the reference (epoch) date for the start of time of UNIX/Linux systems is set at January 01, 1970.

C remains one of the most popular programming languages today. A low-level language close to the hardware, it allows the operating system to be adapted to any machine architecture having a C compiler.

UNIX is an open and evolving operating system that has played a major role in the history of computing. It forms the basis for many other systems, such as Linux, BSD, MacOS, etc.

UNIX remains relevant today (HP-UX, AIX, Solaris, etc.).

GNU Project

- **1984**: Richard Matthew Stallman launched the GNU (GNU's Not Unix) Project, which aims to establish a **free** and **open** Unix system, in which the more important tools are: gcc compiler, bash shell, Emacs editor and so on. GNU is a Unix-like operating system. The development of GNU, started in January 1984, is known as the GNU Project. Many of the programs in GNU are released under the auspices of the GNU Project; those we call GNU packages.
- **1990**: GNU's own kernel, the GNU Hurd, was started in 1990 (before Linux was started).

MINIX

• **1987**: Andrew S. Tanenbaum develops MINIX, a simplified UNIX, to teach operating systems in a simple way. Mr. Tanenbaum makes the source code of his operating system available.

Linux

- **1991**: A Finnish student, **Linus Torvalds**, creates an operating system that runs on his personal computer and names it Linux. He publishes his first version, called 0.02, on the Usenet discussion forum, and other developers help him improve his system. The term Linux is a play on words between the founder's first name, Linus, and UNIX.
- **1993**: The Debian distribution is created. Debian is a non-commercial, community-based distribution. Originally developed for use on servers, it is well-suited for this role; however it is a universal system, usable on a personal computer as well. Debian forms the basis for many other distributions, such as Mint or Ubuntu.
- **1994**: The commercial distribution Red Hat is created by the company Red Hat, which is today the leading distributor of the GNU/Linux operating system. Red Hat supports the community version Fedora and until recently the free distribution CentOS.
- **1997**: The KDE desktop environment is created. It is based on the Qt component library and the C++ development language.
- **1999**: The GNOME desktop environment is created. It is based on the GTK+ component library.
- **2002**: The Arch distribution is created. Its distinctive is that it offers rolling release (continuous update).
- **2004**: Ubuntu is created by the Canonical company (Mark Shuttleworth). It is based on Debian but includes free and proprietary software.
- **2021**: Rocky Linux is created, based on Red Hat distribution.

i Info

Dispute over the name: although people are accustomed to calling the Linux operating system verbally, Linux is strictly a kernel. We must not forget the development and contribution of the GNU project to the open source cause, so! I prefer to call it the GNU/Linux operating system.

3.2.2 Market share

Despite its prevalence, Linux remains relatively unknown to the general public. Linux is hidden within **smartphones**, **televisions**, **internet boxes**, etc. Almost **70% of the websites** in the world are hosted on a Linux or UNIX server!

Linux equips about **3% of personal computers** but more than **82% of smartphones**. The **Android** operating system, for example, uses a Linux kernel.

Linux equips 100% of the top 500 supercomputers since 2018. A supercomputer is a computer designed to achieve the highest possible performance with the techniques known during its design, especially concerning computing speed.

3.2.3 Architectural design

- The **kernel** is the first software component.
- It is the heart of the Linux system.
- It manages the hardware resources of the system.
- The other software components must go through it to access the hardware.
- The **shell** is a utility that interprets user commands and ensures their execution.
- Main shells: Bourne shell, C shell, Korn shell, and Bourne-Again shell (bash).
- Applications are user programs including but not limited to:
- Internet browsers
- Word processors
- Spreadsheets

Multi-task

Linux belongs to the family of time-sharing operating systems. It divides processing time between several programs, transparently switching from one to another for the user. This implies:

- Simultaneous execution of several programs.
- Distribution of CPU time by the scheduler.
- Reduction of problems caused by a failed application.
- Reduced performance in the event of too many programs running.

Multi-user

MULTICS's purpose was to allow users to work from several terminals (screen and keyboard) on a single computer (which was very expensive at the time). Inspired by this operating system, Linux kept this ability to work with several users simultaneously and independently, each with its user account with memory space and access rights to files and software.

Multi-processor

Linux can work with multi-processor computers or with multi-core processors.

Multi-platform

Linux is written in a high-level language that can be adapted to different platforms during compilation. This allows it to run on:

- Home computers (PC and laptop)
- Servers (data and applications)
- Portable computers (smartphones and tablets)
- Embedded systems (car computers)
- Active network elements (routers and switches)
- Household appliances (TVs and refrigerators)

Open

Linux is based on recognized standards such as POSIX, TCP/IP, NFS, and Samba, which allow it to share data and services with other application systems.

3.2.4 The UNIX/Linux Philosophy

- Treat everything as a file.
- Value portability.
- Do one thing and do it well.
- KISS: Keep It Simple Stupid.
- "UNIX is a simple operating system, but you have to be a genius to understand the simplicity." (**Dennis Ritchie**)
- "Unix is user-friendly. It just isn't promiscuous about which users it's friendly with." (**Steven King**)

3.3 The GNU/Linux distributions

A Linux distribution is a **consistent set of software** assembled around the Linux kernel, ready to be installed along with the necessary components to manage itself (installation, removal, configuration). There are **associative** or **community** distributions (Debian, Rocky) and **commercial** distributions (Red Hat, Ubuntu).

Each distribution offers one or more **desktop environments**, and provides a set of pre-installed software and a library of additional software. Configuration options (kernel or services options for example) are specific to each distribution.

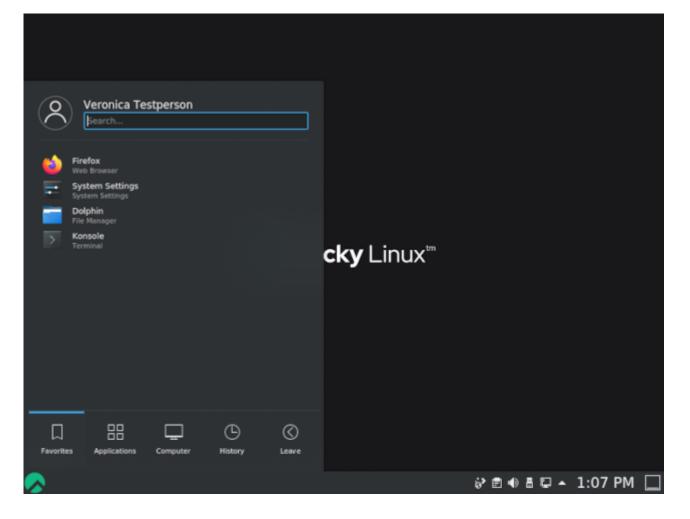
This principle allows distributions to be geared to **beginners** (Ubuntu, Linux Mint...) or fully customizable for **advanced users** (Gentoo, Arch); distributions can also be more adept with **servers** (Debian, Red Hat) or **workstations** (Fedora).

3.3.1 Desktop environments

There are many graphic environments such as **GNOME**, **KDE**, **LXDE**, **XFCE**, etc. There is something for everyone, and their **ergonomics** hold their own against Microsoft or Apple systems. So why is there so little enthusiasm for Linux, when this system is practically **virus free**? Could it be because so many editors (Adobe) and manufacturers (Nvidia) do not play the free game and do not provide a version of their software or **drivers** for GNU/Linux? Perhaps it's fear of change, or the difficulty of finding where to buy a Linux computer, or too few games distributed under Linux. That last excuse, at least, shouldn't be true for long, with the advent of the game engine Steam for Linux.



The **GNOME 3** desktop environment no longer uses the concept of desktop but that of GNOME Shell (not to be confused with the command line shell). It serves as a desktop, a dashboard, a notification area and a window selector. The GNOME desktop environment is based on the **GTK+** component library.



The **KDE** desktop environment is based on the **Qt** component library. It is traditionally recommended for users familiar with a Windows environment.



3.3.2 Free / Open Source

A Microsoft or Mac operating system user must purchase a license to use the system. This license has a cost, although it is usually transparent (the price of the license is included in the price of the computer).

The Free Software movement provides mostly free distributions in the **GNU/Linux** world.

Free does not mean free!

Open Source: the source code is available, so consulting and modifying it under certain conditions is possible.

A free software is necessarily open-source, but the opposite is not true since opensource software is distinct from the freedom offered by the GPL license.

GNU GPL (GNU General Public License)

The **GPL** guarantees the author of a software its intellectual property, but allows modification, redistribution or resale of software by third parties, provided that the source code is included with the software. The GPL is the license that came out of the **GNU** (GNU is Not UNIX) project, which was instrumental in creating Linux.

It implies:

- The freedom to run the program, for any purpose.
- The freedom to study how the program works and adapt it to your needs.
- The freedom to redistribute copies.
- The freedom to improve the program, and publish those improvements for the benefit of the whole community.

On the other hand, even products licensed under the GPL can have a cost. This is not for the product itself, but the **guarantee that a team of developers will continue to work on it to make it evolve and troubleshoot errors, or even provide support to users**.

3.4 Areas of use

A Linux distribution excels for:

- **Servers**: HTTP, email, groupware, file sharing, etc.
- Security: Gateway, firewall, router, proxy, etc.
- Central computers: Banks, insurance, industry, etc.
- Embedded systems: Routers, Internet boxes, SmartTVs, etc.

Linux is a suitable choice for hosting databases or websites, or as a mail server, DNS or firewall. In short, Linux can do just about anything, which explains the quantity of specific distributions.

3.5 Shell

3.5.1 Generalities

The **shell**, known as *command interface*, allows users to send commands to the operating system. It is less visible today since the implementation of graphical interfaces, but remains a privileged means on Linux systems which do not all have graphical interfaces and whose services do not always have a setting interface.

It offers a real programming language including classical structures (loops, alternatives) and common constituents (variables, passing of parameters, and subprograms). It allows the creation of scripts to automate certain actions (backups, creation of users, system monitoring, etc.).

There are several types of shells available and configurable on a platform or according to the user's preference. A few examples include:

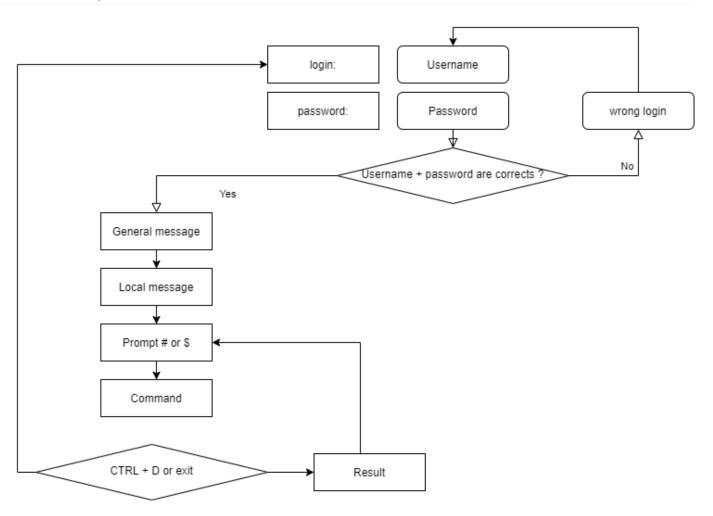
- sh, the POSIX standard shell
- csh, command-oriented shell in C
- bash, Bourne-Again Shell, Linux shell

3.5.2 Functionalities

• Command execution (checks the command given and executes it).

- Input/Output redirection (returns data to a file instead of writing it on the screen).
- Connection process (manages the user's connection).
- Interpreted programming language (allowing the creation of scripts).
- Environment variables (access to information specific to the system during operation).

3.5.3 Principle



3.6 Check your Knowledge

 \checkmark An operating system is a set of programs for managing the available resources of a computer:

True

False

 \checkmark The operating system is necessary to:

Manage physical and virtual memory

Allow direct access to peripherals

Subcontract the management of tasks to the processor

Collect information about the programs used or in use

✓ Among these personalities, which ones participated in the development of UNIX:

Linus Torvalds Ken Thompson Lionel Richie Brian Kernighan Andrew Stuart Tanenbaum

✓ The original nationality of Linus Torvalds, creator of the Linux kernel, is:

Swedish

Finnish

Norwegian

Flemish

French

 \checkmark Which of the following distributions is the oldest:

Debian

Slackware

Red Hat

Arch

 \checkmark Is the Linux kernel:

Multi-tasking

Multi-user

Multi-processor

Multi-core

Cross-platform

Open

✓ Is free software necessarily open-source?

True

False

 \checkmark Is open-source software necessarily free?

True

False

 \checkmark Which of the following is not a shell:

Jason

Jason-Bourne shell (jbsh)

Bourne-Again shell (bash)

C shell (csh)

Korn shell (ksh)

4. Commands for Linux Users

In this chapter you will learn Linux commands and how to use them.

Objectives: In this chapter, future Linux administrators will learn how to:

Move within the system tree.
Create a text file, display its contents and modify it.
Use the most useful Linux commands.

🕅 user commands, linux

Knowledge: ★ Complexity: ★

Reading time: 40 minutes

4.1 Generalities

Current Linux systems have graphical utilities dedicated to the work of an administrator. However, it is important to be able to use the interface in command line mode for several reasons:

- The majority of system commands are common to all Linux distributions, which is not the case for graphical tools.
- It can happen that the system does not start correctly but that a backup command interpreter remains accessible.
- Remote administration is done on the command line with an SSH terminal.
- In order to preserve server resources, the graphical interface is either not installed or launched on demand.
- Administration is done by scripts.

Learning these commands allows the administrator to connect to a Linux terminal, to manage its resources and files, to identify the station, the terminal, and the connected users, etc.

4.1.1 The users

The user of a Linux system will be defined in the /etc/passwd file, by:

- A login name, more commonly called "login", containing no spaces.
- A numeric identifier: **UID** (User Identifier).
- A group identifier: **GID** (Group Identifier).
- A **command interpreter**, e.g., a shell, which can be different from one user to another.
- A connection directory, e.g., the home directory.

In other files the user will be defined by:

- A **password**, which is encrypted before being stored (/etc/shadow).
- A command prompt, or prompt login, which is symbolized by a
- # for administrators
- and a \$ for other users (/etc/profile).

Depending on the security policy implemented on the system, the password will require a certain number of characters and meet certain complexity requirements.

Among the existing command interpreters, the **Bourne-Again Shell** (/bin/bash) is the one most frequently used. It is assigned by default to new users. For various reasons, advanced Linux users can choose alternative shells such as the Korn Shell (ksh), the C Shell (csh), etc.

The user's login directory is by convention stored in the /home directory of the workstation. It will contain the user's personal data and the configuration files of his applications. By default, at login, the login directory is selected as the current directory.

A workstation-type installation (with graphical interface) starts this interface on terminal 1. Linux being multi-user, it is possible to connect several users several times, on different **physical terminals** (TTY) or **virtual terminals** (PTS). Virtual terminals are available within a graphical environment. A user switches from one physical terminal to another using \checkmark Alt + Fx from the command line or using $^{\text{Ctrl}} + \checkmark$ Alt + Fx.

4.1.2 The shell

Once the user is connected to a console, the shell displays the **command prompt**. It then behaves like an infinite loop, repeating the same pattern with each statement entered:

- Displays the command prompt.
- Reads the command.
- Analyzes the syntax.
- Substitutes special characters.
- Executes the command.
- Displays the command prompt.
- Etc.

The key sequence (Ctrl) + C is used to interrupt a running command.

The use of a command generally follows this sequence:

command [option(s)] [argument(s)]

The name of the command is often **lower case**.

A space separates each item.

Short options begin with a dash (-1), while **long options** begin with two dashes (--list). A double dash (--) indicates the end of the option list.

It is possible to group some short options together:

ls -l -i -a

is equivalent to:

ls -lia

There can be several arguments after an option:

ls -lia /etc /home /var

In the literature, the term "option" is equivalent to the term "parameter," which is more commonly used in programming. The optional side of an option or argument is symbolized by enclosing it in square brackets [and]. When more than one option is possible, a vertical bar called a "pipe" separates them [a|e|i].

4.2 General commands

```
4.2.1 apropos, whatis and man commands
```

It is impossible for an administrator at any level to know all the commands and options in detail. A manual is usually available for all installed commands.

apropos command

The command apropos allows you to search by keyword within these manual pages:

Options	Description
-s,sections list orsection list	Limited to manual sections.
-a orand	Displays only the item matching all the provided keywords.

Example:

```
$ apropos clear
clear (1) - clear the terminal screen
clear_console (1) - clear the console
clearenv (3) - clear the environment
clearerr (3) - check and reset stream status
clearerr_unlocked (3) - nonlocking stdio functions
feclearexcept (3) - floating-point rounding and exception handling
```

```
fwup_clear_status (3) - library to support management of system firmware
updates
klogctl (3)
               - read and/or clear kernel message ring buffer; set
console_loglevel
                  - block-clearing puzzle
sgt-samegame (6)
                   - read and/or clear kernel message ring buffer; set
syslog (2)
console_loglevel
timerclear (<mark>3</mark>)
                   - timeval operations
XClearArea (3)
                   - clear area or window
XClearWindow (3) - clear area or window
XSelectionClearEvent (3) - SelectionClear event structure
```

To find the command that will allow changing the password of an account:

```
$ apropos --exact password -a change
chage (1) - change user password expiry information
passwd (1) - change user password
```

whatis command

The whatis command displays the description of the command passed as argument:

whatis clear

Example:

```
$ whatis clear
clear (1) - clear the terminal screen
```

man command

Once found by apropos or whatis, the manual is read by man ("Man is your friend"). This set of manuals is divided into 8 sections, grouping information by topic, the default section being 1:

- 1. Executable programs or commands.
- 2. System calls (functions given by the kernel).
- 3. Library calls (functions given by the library).
- 4. Special files (usually found in /dev).
- 5. File Formats and conventions (configuration files such as etc/passwd).
- 6. Games (such as character-based applications).
- 7. Miscellaneous (e.g. man (7)).
- 8. System administration commands (usually only for root).
- 9. Kernel routines (non-standard).

Information about each section can be accessed by typing man \times intro, where \times is the section number.

The command:

man passwd

will tell the administrator about the passwd command, its options, etc. While a:

man <mark>5</mark> passwd

will inform him about the files related to the command.

Navigate through the manual with the arrows $\square Up$ and $\square Down$. Exit the manual by pressing the Q key.

4.2.2 shutdown command

The shutdown command allows you to **electronically shut down** a Linux server, either immediately or after a certain period of time.

```
shutdown [-h] [-r] time [message]
```

Specify the shutdown time in the format hh:mm for a precise time, or +mm for a delay in minutes.

To force an immediate stop, use the word <u>now</u> in place of the time. In this case, the optional message is not sent to other users of the system.

Examples:

```
[root]# shutdown -h 0:30 "Server shutdown at 0:30"
[root]# shutdown -r +5
```

Options:

Options	Remarks
- h	Shuts down the system electronically.
- r	Restarts the system.

4.2.3 history command

The history command displays the history of commands that have been entered by the user.

The commands are stored in the .bash_history file in the user's login directory.

Example of a history command

\$ history 147 man ls 148 man history

Options	Comments
- W	Writes the current history to the history file.
- C	Deletes the history of the current session (but not the contents of the .bash_history file).

• Manipulating history:

To manipulate the history, the following commands entered from the command prompt will:

Keys	Function
! + !	Recalls the last command placed.
! + N	Recalls the command by its number in the list.
! + string	Recalls the most recent command beginning with the string.
t Up	Navigates through your history working backward in time from the most recent command.
↓ Down	Navigates through your history working forward in time.

4.2.4 Auto-complete

Auto-completion is a great help.

- Completes commands, entered paths, or file names.
- Press the Tab key to complete the entry in the case of a single solution.
- In the case of multiple solutions, press Tab a second time to see options.

If double-pressing the Tab - key presents no options, then there is no solution to the current completion.

4.3 Display and Identification

4.3.1 clear command

The clear command clears the contents of the terminal screen. More accurately, it shifts the display so that the command prompt is at the top of the screen on the first line.

On a physical terminal, the display will be permanently hidden, whereas in a graphical interface, a scrollbar will allow you to go back in the history of the virtual terminal.

ó Tip		
(^ctrl)+(L) will have the same effect as the clear command		

4.3.2 echo command

The echo command is used to display a string of characters.

This command is most commonly used in administration scripts to inform the user during execution.

The -n option indicates no newline output string (by default, newline output string).

```
shell > echo -n "123";echo "456"
123456
shell > echo "123";echo "456"
123
456
```

For various reasons, the script developer may need to use special sequences (starting with a $\$ character). In this case, the -e option will be stipulated, allowing interpretation of the sequences.

Among the frequently used sequences, we can mention:

Sequence	Result
\a	Sends a sonar beep
\b	Back
\n	Adds a line break
١t	Adds a horizontal tab
\v	Adds a vertical tab

4.3.3 date command

The date command displays the date and time. The command has the following syntax:

date [-d yyyyMMdd] [format]

Examples:

```
$ date
Mon May 24 16:46:53 CEST 2021
$ date -d 20210517 +%j
137
```

In this last example, the -d option displays a given date. The +%j option formats this date to show only the day of the year.

🛕 Warning

The format of a date can change depending on the value of the language defined in the environment variable <code>\$LANG</code> .

Option	Format	
+%A	Locale's full weekday name (e.g., Sunday)	
+%B	Locale's full month name (e.g., January)	
+%C	Locale's date and time (e.g., Thu Mar 3 23:05:25 2005)	
+%d	Day of month (e.g., 01)	
+%F	Date in YYYY-MM-DD format	
+%G	Year	
+%H	Hour (0023)	
+%j	Day of the year (001366)	
+%m	Month number (0112)	
+%M	Minute (0059)	
+%R	Time in hh:mm format	
+%s	Seconds since January 1, 1970	
+%S	Second (0060)	
+%T	Time in hh:mm:ss format	
+%u	Day of the week (1 for Monday)	
+%V	Week number (+%v)	
+%X	Date in format DD/MM/YYYY	

The date display can follow the following formats:

The date command also allows you to change the system date and time. In this case, the -s option will be used.

[root]# date -s "2021-05-24 10:19"

The format to be used following the -s option is this:

date -s "yyyy-MM-dd hh:mm[:ss]"

4.3.4 id, who and whoami commands

The id command is used to display information about users and groups. By default, no user parameter is added, and the information of the currently logged in user and group is displayed

```
$ id rockstar
uid=1000(rockstar) gid=1000(rockstar) groups=1000(rockstar),10(wheel)
```

The -g, -G, -n and -u options display the main group GID, subgroup GIDs, names instead of numeric identifiers, and the user's UID respectively.

The whoami command displays the login of the current user.

The who command alone displays the names of logged in users:

\$ who
rockstar tty1 2021-05-24 10:30
root pts/0 2021-05-24 10:31

Since Linux is multi-user, it is possible that multiple sessions are open on the same station, either physically or over the network. It is interesting to know which users are logged in, if only to communicate with them by sending messages.

- tty: represents a terminal.
- pts/: represents a virtual console in a graphical environment with the number after representing the instance of the virtual console (0, 1, 2...)

The -r option also displays the runlevel (see chapter "startup").

4.4 File Tree

In Linux, the file tree is an inverted tree, called a **single hierarchical tree**, whose root is the directory /.

The **current directory** is the directory where the user is located.

The **connection directory** is the working directory associated with the user. The login directories are, by default, stored in the */home* directory.

When the user logs in, the current directory is the login directory.

An **absolute path** references a file from the root by traversing the entire tree to the file level:

```
    /home/groupA/alice/file
```

The **relative path** references that same file by traversing the entire tree from the current directory:

• ../alice/file

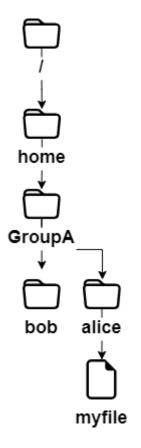
In the above example, the " . . " refers to the parent directory of the current directory.

A directory, even if it is empty, will necessarily contain at least **two references**:

- . : reference to itself.
- ...: reference to the parent directory of the current directory.

A relative path can thus start with ./ or ../. When the relative path refers to a subdirectory or file in the current directory, then the ./ is often omitted. Mentioning the first ./ in the tree will only really be required to run an executable file.

Errors in paths can cause many problems: creating folders or files in the wrong places, unintentional deletions, etc. It is therefore strongly recommended to use auto-completion when entering paths.



In the above example, we are looking to give the location of the file myfile from the directory of bob.

- By an **absolute path**, the current directory does not matter. We start at the root, and work our way down to the directories home, groupA, alice and finally the file myfile: /home/groupA/alice/myfile.
- By a **relative path**, our starting point being the current directory bob, we go up one level through .. (i.e., into the groupA directory), then down into the alice directory, and finally the myfile file: ../alice/myfile.

4.4.1 pwd command

The pwd (Print Working Directory) command displays the absolute path of the current directory.

\$ pwd
/home/rockstar

To use a relative path to reference a file or directory, or use the cd command to move to another directory, you must know its location in the file tree.

Depending on the type of shell and the different parameters of its configuration file, the terminal prompt (also known as the command prompt) will display the absolute or relative path of the current directory.

4.4.2 cd command

The cd (Change Directory) command allows you to change the current directory -in other words, to move through the tree.

\$ cd /tmp
\$ pwd
/tmp
\$ cd ../
\$ pwd
/
\$ cd
\$ pwd
/
\$ cd
\$ pwd
/home/rockstar

As you can see in the last example above, the command cd with no arguments moves the current directory to the home directory.

4.4.3 ls command

The ls command displays the contents of a directory.

```
ls [-a] [-i] [-l] [directory1] [directory2] [...]
```

Example:

```
$ ls /home
. .. rockstar
```

The main options of the ls command are:

Option	Information	
- a	Displays all files, even hidden ones. Hidden files in Linux are those beginning with \ldots	
-i	Displays inode numbers.	
-1	Use a long listing format, that is, each line displays long format information for a file or directory.	

Option	Information
- d	Displays information about a directory instead of listing its contents.
- g	Like -l option, but do not list owner.
-h	Displays file sizes in the most appropriate format (byte, kilobyte, megabyte, gigabyte,). h stands for Human Readable. Needs to be used with -l option.
- S	Displays the allocated size of each file, in blocks. In the Ls command, the default size of a single block is 1024-Byte. In the GNU/Linux operating system, "block" is the smallest unit of storage in the file system, and generally speaking, one block is equal to 4096-Byte. In the Windows operating system, taking the NTFS file system as an example, its smallest storage unit is called a "Cluster". The definition of the minimum storage unit name may vary depending on different file systems.
-A	Displays all files in the directory except $\ . \ $ and $\ . \ .$
- R	Displays the contents of subdirectories recursively.
- F	Displays the type of files. Prints a $/$ for a directory, \star for executables, $@$ for a symbolic link, and nothing for a text file.
- X	Sorts files according to their extensions.

The ls command, however, has a lot of options (see man):

• Description of columns generated by running the ls -lia command:

\$ ls -lia /home

78489 drwx----- 4 rockstar rockstar 4096 25 oct. 08:10 rockstar

Value	Information
78489	Inode Number.
drwx	File type (d) and rights (rwx).
4	Number of subdirectories (. and included). For a file, it represents the number of hard links, and 1 represents itself.
rockstar	User ownership.
rockstar	Group ownership.
4096	For files, it shows the size of the file. For directories, it shows the fixed value of 4096 bytes occupied by the file naming. To calculate the total size of a directory, use du -sh rockstar/
25 oct. 08:10	Last modified date.
rockstar	The name of the file (or directory).

🖍 Note

Aliases are frequently positioned in common distributions.

This is the case of the alias \mathfrak{ll} :

alias ll='ls -l --color=auto'

The ls command has many options. Here are some advanced examples of uses:

• List the files in /etc in order of last modification:

```
$ ls -ltr /etc
total 1332
-rw-r--r-. 1 root root 662 29 may 2021 logrotate.conf
-rw-r--r-. 1 root root 272 17 may. 2021 mailcap
-rw-----. 1 root root 122 12 may. 2021 securetty
...
-rw-r--r-. 2 root root 85 18 may. 17:04 resolv.conf
-rw-r--r-. 1 root root 44 18 may. 17:04 adjtime
-rw-r--r-. 1 root root 283 18 may. 17:05 mtab
```

• List /var files larger than 1 megabyte but less than 1 gigabyte. The example here uses advanced grep commands with regular expressions. Novices don't have to struggle too much, there will be a special tutorial to introduce these regular expressions in the future.

```
$ ls -lhR /var/ | grep ^\- | grep -E "[1-9]*\.[0-9]*M"
...
-rw-r--r-. 1 apache apache 1.2M 10 may. 13:02 XB RiyazBdIt.ttf
-rw-r--r-. 1 apache apache 1.2M 10 may. 13:02 XB RiyazBd.ttf
-rw-r--r-. 1 apache apache 1.1M 10 may. 13:02 XB RiyazIt.ttf
...
```

Of course, we highly recommend that you use the find command.

find /var -size +1M -a -size -1024M -a -type f -exec ls -lh {} \;

• Show the rights on a folder:

To find out the rights to a folder, in our example /etc , the following command would **not** be appropriate:

The above command will display the contents of the folder (inside) by default. For the folder itself, you can use the -d option.

ls -ld /etc drwxr-xr-x. 69 root root 4096 18 nov. 17:05 /etc

• Sort by file size, largest first:

ls -lhS

• time/date format with -1:

```
$ ls -l --time-style="+%Y-%m-%d %m-%d %H:%M" /
total 12378
dr-xr-xr-x. 2 root root 4096 2014-11-23 11-23 03:13 bin
dr-xr-xr-x. 5 root root 1024 2014-11-23 11-23 05:29 boot
```

• Add the *trailing slash* to the end of folders:

By default, the ls command does not display the last slash of a folder. In some cases, like for scripts for example, it is useful to display them:

```
$ ls -dF /etc
/etc/
```

• Hide some extensions:

ls /etc --hide=*.conf

4.4.4 mkdir command

The mkdir command creates a directory or directory tree.

```
mkdir [-p] directory [directory] [...]
```

Example:

mkdir /home/rockstar/work

The "rockstar" directory must exist to create the "work" directory.

Otherwise, the -p option should be used. The -p option creates the parent directories if they do not exist.

• Danger
It is not recommended to use Linux command names as directory or file names.

4.4.5 touch command

The touch command changes the timestamp of a file or creates an empty file if the file does not exist.

touch [-t date] file

Example:

touch /home/rockstar/myfile

Option	Information
-t date	Changes the date of last modification of the file with the specified date.

Date format: [AAAA]MMJJhhmm[ss]

Ó Tip
The touch command is primarily used to create an empty file, but it can be useful for incremental or differential backups for example. Indeed, the only effect of executing a touch on a file will be to force it to be saved during the next backup.

4.4.6 rmdir command

The rmdir command deletes an empty directory.

Example:

rmdir /home/rockstar/work

-p Removes the parent directory or directories provided if they are empty.	Option	Information
() Tim	- p	Removes the parent directory or directories provided if they are empty.
	b Tip	

To delete both a non-empty directory and its contents, use the rm command.

4.4.7 rm command

The rm command deletes a file or directory.

```
rm [-f] [-r] file [file] [...]
```

O Danger

Any deletion of a file or directory is final.

Options	Information	
-f	Do not ask whether to delete.	
-i	Ask whether to delete.	
- r	Delete a directory and recursively delete its subdirectories.	

🖍 Note

The rm command itself does not ask for confirmation when deleting files. However, with a Red Hat/Rocky distribution, rm does ask for confirmation of deletion because the rm command is an alias of the rm -i command. Don't be surprised if on another distribution, like Debian for example, you don't get a confirmation request.

Deleting a folder with the rm command, whether the folder is empty or not, will require the -r option to be added.

The end of the options is signaled to the shell by a double dash --.

In the example:

```
$ >-hard-hard # To create an empty file called -hard-hard
hard-hard
[CTRL+C] To interrupt the creation of the file
$ rm -f -- -hard-hard
```

The hard-hard file name starts with a -. Without the use of the -- the shell would have interpreted the -d in -hard-hard as an option.

4.4.8 mv command

The mv command moves and renames a file.

```
mv file [file ...] destination
```

Examples:

```
mv /home/rockstar/file1 /home/rockstar/file2
```

mv /home/rockstar/file1 /home/rockstar/file2 /tmp

Options	Information	
- f	Don't ask for confirmation if overwriting the destination file.	
-i	Request confirmation if overwriting destination file (default).	

A few concrete cases will help you understand the difficulties that can arise:

```
mv /home/rockstar/file1 /home/rockstar/file2
```

Renames file1 to file2. If file2 already exists, replace the contents of the file with file1.

mv /home/rockstar/file1 /home/rockstar/file2 /tmp

Moves file1 and file2 into the /tmp directory.

mv file1 /repexist/file2

Moves file1 into repexist and renames it file2.

mv file1 file2

file1 is renamed to file2.

mv file1 /repexist

If the destination directory exists, file1 is moved to /repexist.

mv file1 /wrongrep

If the destination directory does not exist, file1 is renamed to wrongrep in the root directory.

4.4.9 cp command

The cp command copies a file.

cp file [file ...] destination

Example:

cp -r /home/rockstar /tmp

Options	Information	
-i	Request confirmation if overwriting (default).	
- f	Do not ask for confirmation if overwriting the destination file.	
- p	Keeps the owner, permissions and timestamp of the copied file.	
- r	Copies a directory with its files and subdirectories.	
- S	Creates a symbolic link rather than copying.	

cp file1 /repexist/file2

file1 is copied to /repexist under the name file2.

cp file1 file2

file1 is copied as file2 to this directory.

cp file1 /repexist

If the destination directory exists, file1 is copied to /repexist.

cp file1 /wrongrep

If the destination directory does not exist, file1 is copied under the name wrongrep to the root directory.

4.5 Visualization

4.5.1 file command

The file command displays the type of a file.

```
file file1 [files]
```

Example:

```
$ file /etc/passwd /etc
/etc/passwd: ASCII text
/etc: directory
```

4.5.2 more command

The more command displays the contents of one or more files screen by screen.

```
more file1 [files]
```

Example:

```
$ more /etc/passwd
root:x:0:0:root:/root:/bin/bash
...
```

Using the Enter a key, the move is line by line. Using the Space key, the move is page by page. /text allows you to search for the occurrence in the file.

4.5.3 less command

The less command displays the contents of one or more files. The less command is interactive and has its own commands for use.

```
less file1 [files]
```

The commands specific to less are:

Command	Action
h or H	Help.
(↑ Up) (↓ Down) (→ Right) (← Left)	Move up, down a line, or to the right or left.
Enter 🕫	Move down one line.
Space	Move down one page.
(* Page Up) and (* Page Down)	Move up or down one page.
g and G	Move to the first and last pages
/text	Search for text.
q or Q	Quit the less command.

4.5.4 cat command

The cat command concatenates the contents of multiple files and displays the result on the standard output.

cat file1 [files]

Example 1 - Displaying the contents of a file to the standard output:

cat /etc/passwd

Example 2 - Displaying the contents of multiple files to standard output:

cat /etc/passwd /etc/group

Example 3 - Combining the contents of multiple files into one file using output redirection:

cat /etc/passwd /etc/group > usersAndGroups.txt

Example 4 - Displaying the line numbering:

```
$ cat -n /etc/profile

1  # /etc/profile: system-wide .profile file for the Bourne shell

(sh(1))

2  # and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).

3

4  if [ "`id -u`" -eq 0 ]; then

5     PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"

6  else

...
```

Example 5 - Shows the numbering of non-empty lines:

```
$ cat -b /etc/profile
1 # /etc/profile: system-wide .profile file for the Bourne shell
(sh(1))
2 # and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).
3 if [ "`id -u`" -eq 0 ]; then
4 PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
5 else
...
```

4.5.5 tac command

The tac command does almost the opposite of the cat command. It displays the contents of a file starting from the end (which is particularly interesting for reading logs!).

Example: Display a log file by displaying the last line first:

```
[root]# tac /var/log/messages | less
```

4.5.6 head command

The head command displays the beginning of a file.

```
    head [-n x] file

    Option
    Description

    -n x
    Display the first x lines of the file
```

By default (without the -n option), the head command will display the first 10 lines of the file.

4.5.7 tail command

The tail command displays the end of a file.

```
tail [-f] [-n x] file
```

Option	Description
-n x	Displays the last \times lines of the file
- f	Displays changes to the file in real time

Example:

```
tail -n 3 /etc/passwd
sshd:x:74:74:Privilege-separeted sshd:/var/empty /sshd:/sbin/nologin
tcpdump::x:72:72::/:/sbin/nologin
user1:x:500:500:grp1:/home/user1:/bin/bash
```

With the -f option, the change information of the file will always be output unless the user exits the monitoring state with ctrl + c. This option is very frequently used to track log files (the logs) in real time.

Without the -n option, the tail command displays the last 10 lines of the file.

4.5.8 sort command

The sort command sorts the lines of a file.

It allows you to order the result of a command or the content of a file in a given order, numerically, alphabetically, by size (KB, MB, GB) or in reverse order.

sort [-k] [-n] [-u] [-o file] [-t] file

Example:

```
$ sort -k 3,4 -t ":" -n /etc/passwd
root:x:0:0:root:/root:/bin/bash
adm:x:3:4:adm:/var/adm/:/sbin/nologin
```

Option	Description
- k	Specify the columns to be separated. You can specify multiple columns.
- n	Requests a numeric sort.
-o file	Saves the sort to the specified file.
-t	Specify a delimiter, which requires that the contents of the corresponding file must be regularly delimited column contents, otherwise they cannot be sorted properly.
- r	Reverse the order of the result. Used in conjunction with the -n option to sort in order from largest to smallest.
- u	Remove duplicates after sorting. Equivalent to sort FILE uniq command.

The sort command sorts the file only on the screen. The file is not modified by the sorting. To save the sort, use the -o option or an output redirection >.

By default, the numbers are sorted according to their character. Thus, "110" will be before "20", which will itself be before "3". The -n option must be specified so that the numeric character blocks are sorted by their value.

The sort command reverses the order of the results, with the -r option:

```
$ sort -k 3 -t ":" -n -r /etc/passwd
nobody:x:65534:65534:Kernel Overflow User:/:/sbin/nologin
systemd-coredump:x:999:997:systemd Core Dumper:/:/sbin/nologin
polkitd:x:998:996:User for polkitd:/:/sbin/nologin
```

In this example, the sort command will sort the contents of the /etc/passwd file this time from largest uid (user identifier) to smallest.

Some advanced examples of using the sort command:

• Shuffling values

The sort command also allows you to shuffle values with the -R option:

sort -R /etc/passwd

• Sorting IP addresses

A system administrator is quickly confronted with the processing of IP addresses from the logs of his services such as SMTP, VSFTP or Apache. These addresses are typically extracted with the cut command.

Here is an example with the file dns-client.txt:

192.168.1.10
192.168.1.200
5.1.150.146
208.128.150.98
208.128.150.99
\$ sort -nr dns-client.txt
208.128.150.99
208.128.150.98
192.168.1.200
192.168.1.10
5.1.150.146

• Sorting file by removing duplicates

Dod

The sort command knows how to remove the duplicates from the file output using -u as option.

Here is an example with the file colours.txt:

Red		
Green		
Blue		
Red		
Pink		
\$ sort Blue Green	-u colours.txt	

Pink			
Red			

• Sorting file by sizes

The sort command knows how to recognize file sizes, from commands like 1s with the -h option.

Here is an example with the file size.txt:

18M 69K
69K
2.4M
1.2M
4.2G
6M
124M
12.4M
4G
\$ sort -hr size.txt
4.2G
4G
1.7G
124M
18M
12. 4M
6M
2 .4M
1 .2M
69K

4.5.9 wc command

The wc command counts the number of lines, words and/or bytes in a file.

wc [-l] [-m] [-w] file [files]

Option	Description
- C	Count the number of bytes.
- m	Count the number of characters.
-1	Counts the number of lines.
- W	Counts the number of words.

4.6 Search

4.6.1 find command

The find command searches for files or directories location.

find directory [-name name] [-type type] [-user login] [-date date]

Since there are so many options to the find command, it is best to refer to the man .

If the search directory is not specified, the find command will search from the current directory.

Option	Description
-perm permissions	Search for files by their permissions.
-size size	Search for files by size.

4.6.2 -exec option of the find command

It is possible to use the -exec option of the find command to execute a command on each result line:

find /tmp -name *.txt -exec rm -f {} \;

The previous command searches for all files in the /tmp directory named *.txt and deletes them.

b Understand the -exec option

In the example above, the find command will construct a string representing the command to be executed.

If the find command finds three files named log1.txt, log2.txt, and log3.txt, then the find command will construct the string by replacing in the string rm -f {} \; the braces with one of the results of the search, and do this as many times as there are results.

This will give us:

rm -f /tmp/log1.txt ; rm -f /tmp/log2.txt ; rm -f /tmp/log3.txt ;

The ; character is a special shell character that must be protected by a \times to prevent it from being interpreted too early by the find command (and not in the -exec).

🜢 Tip

\$ find /tmp -name *.txt -delete does the same thing.

4.6.3 whereis command

The whereis command searches for files related to a command.

```
whereis [-b] [-m] [-s] command
```

Example:

```
$ whereis -b ls
ls: /bin/ls
```

Option	Description
- b	Searches only the binary file.
- m	Searches only for man pages.
- S	Searches only for source files.

4.6.4 grep command

The grep command searches for a string in a file.

grep [-w] [-i] [-v] "string" file

Example:

```
$ grep -w "root:" /etc/passwd
root:x:0:0:root:/root:/bin/bash
```

Option	Description
-i	Ignores the case of the searched string.
- V	Excludes lines containing the string.
- W	Searches for the exact word.

The grep command returns the complete line containing the string you are looking for.

- The ^ special character is used to search for a string at the beginning of a line.
- The special character \$ searches for a string at the end of a line.

```
grep -w "^root" /etc/passwd
```

```
Note
This command is very powerful and it is highly recommended to consult its manual. It has many derivatives.
```

It is possible to search for a string in a file tree with the **-**R option.

```
grep -R "Virtual" /etc/httpd
```

4.6.5 Meta-characters (wildcards)

Meta-characters replace one or more characters (or even an absence of characters) during a search. These meta-characters are also known as wildcards.

They can be combined.

The ***** character replaces a string composed of any characters. The ***** character can also represent an absence of character.

```
$ find /home -name "test*"
/home/rockstar/test1
/home/rockstar/test11
```

```
/home/rockstar/tests
/home/rockstar/test362
```

Meta-characters allow more complex searches by replacing all or part of a word. Simply replace the unknowns with these special characters.

The character ? replaces a single character, whatever it is.

```
$ find /home -name "test?"
/home/rockstar/test1
/home/rockstar/tests
```

The square brackets [and] are used to specify the values that a single character can take.

```
$ find /home -name "test[123]*"
/home/rockstar/test1
/home/rockstar/test11
/home/rockstar/test362
```

Note

Always surround words containing meta-characters with " to prevent them from being replaced by the names of files that meet the criteria.

🛕 Warning

Do not confuse shell meta-characters with regular expression meta-characters. The grep command uses regular expression meta-characters.

4.7 Redirects and pipes

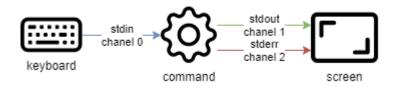
4.7.1 Standard input and output

On UNIX and Linux systems, there are three standard streams. They allow programs, via the stdio.h library, to input or output information.

These streams are called X channel or X file descriptor.

By default:

- the keyboard is the input device for channel 0, called **stdin** ;
- the screen is the output device for channels 1 and 2, called **stdout** and **stderr**.



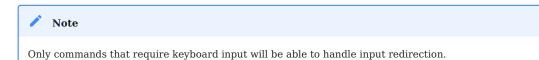
stderr receives the error streams returned by a command. The other streams are directed to **stdout**.

These streams point to peripheral files, but since everything is a file in UNIX/Linux, I/O streams can easily be diverted to other files. This principle is the strength of the shell.

4.7.2 Input redirection

It is possible to redirect the input stream from another file with the character < or <<. The command will read the file instead of the keyboard:

```
ftp -in serverftp << ftp-commands.txt</pre>
```



Input redirection can also be used to simulate user interactivity. The command will read the input stream until it encounters the defined keyword after the input redirection.

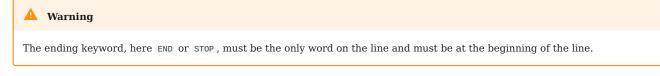
This feature is used to script interactive commands:

```
$ ftp -in serverftp << END
user alice password
put file
bye
END</pre>
```

The keyword END can be replaced by any word.

```
$ ftp -in serverftp << STOP
user alice password
put file
bye
STOP</pre>
```

The shell exits the ftp command when it receives a line containing only the keyword.



The standard input redirection is rarely used because most commands accept a filename as an argument.

The command wc could be used like this:

```
$ wc -l .bash_profile
27 .bash_profile # the number of lines is followed by the file name
$ wc -l < .bash_profile
27 # returns only the number of lines</pre>
```

4.7.3 Output redirection

Standard output can be redirected to other files using the > or >> characters.

The simple > redirection overwrites the contents of the output file:

```
date +%F > date_file
```

When the >> character is used, it indicates that the output result of the command is appended to the file content.

date +%F >> date_file

In both cases, the file is automatically created when it does not exist.

The standard error output can also be redirected to another file. This time it will be necessary to specify the channel number (which can be omitted for channels 0 and 1):

```
ls -R / 2> errors_file
ls -R / 2>> errors_file
```

4.7.4 Examples of redirection

Redirection of 2 outputs to 2 files:

ls -R / >> ok_file 2>> nok_file

Redirection of the 2 outputs to a single file:

ls -R / >> log_file 2>&1

Redirection of *stderr* to a "bottomless pit" (/dev/null):

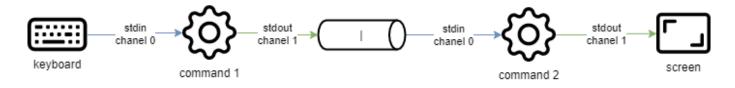
ls -R / 2>> /dev/null

When both output streams are redirected, no information is displayed on the screen. To use both the output redirection and to keep the display, you will have to use the command tee.

4.7.5 Pipes

A **pipe** is a mechanism allowing you to link the standard output of a first command to the standard input of a second command.

This communication is uni directional and is done with the \parallel symbol. The pipe symbol \parallel is obtained by pressing the $\boxed{1 \text{ shift}} + \boxed{1}$ simultaneously.



All data sent by the control on the left of the pipe through the standard output channel is sent to the standard input channel of the control on the right.

The commands particularly used after a pipe are filters.

• Examples:

Display only the beginning:

ls -lia / | head

Display only the end:

ls -lia / | tail

Sort the result:

ls -lia / | sort

Count the number of words / characters:

ls -lia / | wc

Search for a string in the result:

```
ls -lia / | grep fichier
```

4.8 Special Points

4.8.1 tee command

The tee command is used to redirect the standard output of a command to a file while maintaining the screen display.

It is combined with the || pipe to receive as input the output of the command to be redirected:

ls -lia / | tee fic cat fic

The -a option adds to the file instead of overwriting it.

4.8.2 alias and unalias commands

Using **alias** is a way to ask the shell to remember a particular command with its options and give it a name.

For example:

11

will replace the command:

ls -l

The alias command lists the aliases for the current session. Aliases are set by default on Linux distributions. Here, the aliases for a Rocky server:

```
$ alias
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-
tilde'
```

The aliases are only defined temporarily, for the time of the user session.

For permanent use, they must be created in the:

- .bashrc file in the user's login directory;
- /etc/bashrc file for all users.



The unalias command allows you to delete aliases.

To delete a single alias:

unalias ll

To delete all aliases:

```
unalias -a
```

To disable an alias temporarily, the combination is \<alias name>.

For example if we do:

type ls

it might return the following:

ls is an alias to « ls -rt »

Now that this is known, we can see the results of using the alias or disabling it one time with the $\$ by executing the following:

```
$ ls file* # order by time
file3.txt file2.txt file1.txt
$ \ls file* # order by name
file1.txt file2.txt file3.txt
```

4.8.3 Aliases and Useful Functions

• grep alias.

Colorize the result of the grep command: alias grep='grep --color=auto'

• mcd function

It is common to create a folder and then move around in it: mcd() { mkdir -p "\$1"; cd "\$1"; }

• cls function

Move to a folder and list its contents: cls() { cd "\$1"; ls; }

• backup function

Create a backup copy of a file: backup() { cp "\$1"{,.bak}; }

extract function

Extract any type of archive:

```
extract () {
  if [ -f $1 ] ; then
    case $1 in
      *.tar.bz2) tar xjf $1 ;;
      *.tar.gz) tar xzf $1 ;;
      *.bz2) bunzip2 $1 ;;
      *.rar) unrar e $1 ;;
      *.gz) gunzip $1 ;;
      *.tar) tar xf $1 ;;
      *.tbz2) tar xjf $1 ;;
      *.tgz) tar xzf $1 ;;
      *.zip) unzip $1 ;;
      *.Z) uncompress $1 ;;
      *.7z) 7z x $1 ;;
      *)
        echo "'$1' cannot be extracted via extract()" ;;
    esac
  else
    echo "'$1' is not a valid file"
  fi
}
```

• If alias cmount returns the following: alias cmount="mount | column -t"

Then we can use <code>cmount</code> to show all of the system mounts in columns like this: [root]# cmount

/dev/simfs on / type simfs (rw,relatime,usrquota,grpquota) on /proc proc type proc (rw,relatime) sysfs type sysfs on /sys (rw,relatime) none on /dev type devtmpfs (rw, relatime, mode=755) on /dev/pts type devpts none (rw,relatime,mode=600,ptmxmode=000) on /dev/shm type tmpfs none (rw,relatime) on /proc/sys/fs/binfmt_misc type binfmt_misc none (rw,relatime)

which would return our mounted filesystem in the following format:

4.8.4 The character ;

The ; character strings the commands.

The commands will all run sequentially in the order of input once the user presses [Enter -].

```
ls /; cd /home; ls -lia; cd /
```

4.9 Check your Knowledge

✓ What defines a user under Linux? (7 answers)

✓ What characterizes a long option for a command?

✓ Which commands allow you to search for help on a command?

```
google
```

```
chuck --norris
```

info

apropos

whatis

✓ Which command allows you to view a user's history?

 \checkmark Which command allows you to search for text in a file?

find

grep

 \checkmark Which command allows you to search for a file?

find

grep

✓ Which command redirects the error stream of a command to a new errors.log file?

ls -R / 2> errors.log

ls -R / 2>> errors.log

ls -R / 2> errors.log 2>&1

5. Advanced Commands for Linux users

Advanced commands provide greater customization and controls in more specialized situations once you become familiar with basic commands.

Objectives: In this chapter, future Linux administrators will learn:

 \checkmark some useful commands not covered in the previous chapter.

 \checkmark some advanced commands.

🕅 user commands, Linux

Knowledge: \star Complexity: $\star \star \star$

Reading time: 20 minutes

5.1 uniq command

The uniq command is a very powerful command, used with the sort command, especially for log file analysis. It allows you to sort and display entries by removing duplicates.

To illustrate how the uniq command works, let us use a firstnames.txt file containing a list of first names:

antoine xavier steven patrick xavier antoine antoine steven

Note

uniq requires the input file to be sorted before running because it only compares consecutive lines.

With no argument, the uniq command will not display identical lines that follow each other in the firstnames.txt file:

```
$ sort firstnames.txt | uniq
antoine
patrick
steven
xavier
```

To display only the rows that appear only once, use the -u option:

```
$ sort firstnames.txt | uniq -u
patrick
```

Conversely, to display only the lines that appear at least twice in the file, use the -d option:

```
$ sort firstnames.txt | uniq -d
antoine
steven
xavier
```

To simply delete lines that appear only once, use the -D option:

```
$ sort firstnames.txt | uniq -D
antoine
antoine
antoine
steven
steven
xavier
xavier
```

Finally, to count the number of occurrences of each line, use the -c option:

```
$ sort firstnames.txt | uniq -c
3 antoine
1 patrick
2 steven
2 xavier
```

```
$ sort firstnames.txt | uniq -cd
3 antoine
2 steven
2 xavier
```

5.2 xargs commands

The xargs command allows the construction and execution of command lines from standard input.

The xargs command reads whitespace or linefeed delimited arguments from standard input, and executes the command (/bin/echo by default) one or more times using the initial arguments followed by the arguments read from standard input.

A first and simplest example would be the following:

\$ xargs
use
of
xargs
<CTRL+D>
use of xargs

The xargs command waits for an input from the standard **stdin** input. Three lines are entered. The end of the user input is specified to xargs by the keystroke sequence $^{ctrl} + D$. xargs then executes the default command echo followed by the three arguments corresponding to the user input, namely:

```
$ echo "use" "of" "xargs"
use of xargs
```

It is possible to specify a command to be run by xargs.

In the following example, xargs will run the command ls -ld on the set of folders specified in the standard input:

```
$ xargs ls -ld
/home
/tmp
```

```
/root
<CTRL+D>
drwxr-xr-x. 9 root root 4096 5 avril 11:10 /home
dr-xr-x--. 2 root root 4096 5 avril 15:52 /root
drwxrwxrwt. 3 root root 4096 6 avril 10:25 /tmp
```

In practice, the xargs command executed the ls -ld /home /tmp /root command.

What happens if the command to be executed does not accept multiple arguments, such as with the find command?

```
$ xargs find /var/log -name
*.old
*.log
find: paths must precede expression: *.log
```

The xargs command attempted to execute the find command with multiple arguments behind the -name option, which caused find to generate an error:

```
$ find /var/log -name "*.old" "*.log"
find: paths must precede expression: *.log
```

In this case, the xargs command must be forced to execute the find command several times (once per line entered as standard input). The -L option followed by an **integer** allows you to specify the maximum number of entries to be processed with the command at one time:

```
$ xargs -L 1 find /var/log -name
*.old
/var/log/dmesg.old
*.log
/var/log/boot.log
/var/log/anaconda.yum.log
/var/log/anaconda.storage.log
/var/log/anaconda.log
/var/log/anaconda.log
/var/log/audit/audit.log
/var/log/anaconda.ifcfg.log
/var/log/dracut.log
/var/log/dracut.log
/var/log/anaconda.program.log
<CTRL+D>
```

- 73/285 -

To specify both arguments on the same line, use the -n 1 option:

\$ xargs -n 1 find /var/log -name *.old *.log /var/log/dmesg.old /var/log/boot.log /var/log/anaconda.yum.log /var/log/anaconda.storage.log /var/log/anaconda.log /var/log/anaconda.log /var/log/audit/audit.log /var/log/anaconda.ifcfg.log /var/log/dracut.log /var/log/anaconda.program.log <CTRL+D>

Case study of a backup with a tar based on a search:

```
$ find /var/log/ -name "*.log" -mtime -1 | xargs tar cvfP /root/log.tar
$ tar tvfP /root/log.tar
-rw-r--r- root/root 1720 2017-04-05 15:43 /var/log/boot.log
-rw-r--r- root/root 499270 2017-04-06 11:01 /var/log/audit/audit.log
```

The special feature of the xargs command is that it places the input argument at the end of the called command. This works very well with the above example since the files passed in will form the list of files to be added to the archive.

Using the example of the cp command, to copy a list of files in a directory, this list of files will be added at the end of the command... but what the cp command expects at the end of the command is the destination. To do this, use the -I option to put the input arguments somewhere else than at the end of the line.

find /var/log -type f -name "*.log" | xargs -I % cp % /root/backup

The -I option allows you to specify a character (the % character in the above example) where the input files to xargs will be placed.

5.3 yum-utils package

The yum-utils package is a collection of utilities, built for yum by various authors, which make it easier and more powerful to use.

Note

While yum has been replaced by dnf in Rocky Linux 8, the package name has remained yum-utils, although it can be installed as dnf-utils as well. These are classic YUM utilities implemented as CLI shims on top of DNF to maintain backwards compatibility with yum-3.

Here are some examples of these utilities.

5.3.1 repoquery command

The repoquery command is used to query the packages in the repository.

Examples of use:

• Display the dependencies of a package (it can be a software package that has been installed or not installed), equivalent to dnf deplist <package-name>

repoquery --requires <package-name>

• Display the files provided by an installed package (does not work for packages that are not installed), equivalent to rpm -ql <package-name>

```
$ repoquery -l yum-utils
/etc/bash_completion.d
/etc/bash_completion.d/yum-utils.bash
/usr/bin/debuginfo-install
/usr/bin/find-repos-of-install
/usr/bin/needs-restarting
/usr/bin/package-cleanup
/usr/bin/repo-graph
/usr/bin/repo-rss
/usr/bin/repoclosure
/usr/bin/repodiff
/usr/bin/repomanage
/usr/bin/repoquery
/usr/bin/reposync
/usr/bin/repotrack
/usr/bin/show-changed-rco
/usr/bin/show-installed
/usr/bin/verifytree
/usr/bin/yum-builddep
/usr/bin/yum-config-manager
/usr/bin/yum-debug-dump
/usr/bin/yum-debug-restore
```

- 75/285 -

```
/usr/bin/yum-groups-manager
/usr/bin/yumdownloader
```

5.3.2 yumdownloader command

The yumdownloader command downloads RPM packages from the repositories. Equivalent to dnf download --downloadonly --downloaddir ./ package-name

```
Note
This command is very useful to quickly build a local repository of a few rpms!
```

Example: yumdownloader will download the *samba* rpm package and all its dependencies:

```
$ yumdownloader --destdir /var/tmp --resolve samba
or
$ dnf download --downloadonly --downloaddir /var/tmp --resolve samba
```

Options	Comments
destdir	The downloaded packages will be stored in the specified folder.
resolve	Also downloads the package dependencies.

5.4 psmisc packages

The psmisc package contains utilities for managing system processes:

- pstree: the pstree command displays the current processes on the system in a tree-like structure.
- killall: the killall command sends a kill signal to all processes identified by name.
- fuser : the fuser command identifies the PID of processes that use the specified files or file systems.

Examples:

```
$ pstree
systemd—__NetworkManager—_2*[{NetworkManager}]
```

|-agetty |-auditd—_{auditd} |-crond |-dbus-daemon—_{dbus-daemon} |-firewalld—_{firewalld} |-lvmetad |-master___pickup | ___qmgr |-polkitd—_5*[{polkitd}] |-rsyslogd—_2*[{rsyslogd}] |-sshd—_sshd—bash—pstree |-systemd-journal |-systemd-logind |-systemd-udevd |-tuned—_4*[{tuned}]]

killall httpd

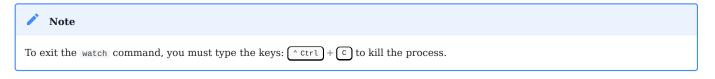
Kill processes (option -k) that access the /etc/httpd/conf/httpd.conf file:

fuser -k /etc/httpd/conf/httpd.conf

5.5 watch command

The watch command regularly executes a command and displays the result in the terminal in full screen.

The -n option allows you to specify the number of seconds between each execution of the command.



Examples:

• Display the end of the /etc/passwd file every 5 seconds:

watch -n 5 tail -n 3 /etc/passwd

Result:

```
Every 5.0s: tail -n 3 /etc/
passwd
rockstar.rockylinux.lan: Thu Jul 1 15:43:59 2021
sssd:x:996:993:User for sssd:/:/sbin/nologin
chrony:x:995:992::/var/lib/chrony:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
```

• Monitoring the number of files in a folder:

```
watch -n 1 'ls -l | wc -l'
```

• Display a clock:

watch -t -n 1 date

5.6 install command

Contrary to what its name might suggest, the install command is not used to install new packages.

This command combines file copying (cp) and directory creation (mkdir), with rights management (chmod , chown) and other useful functionalities (like backups).

```
install source dest
install -t directory source [...]
install -d directory
```

Options:

Options	Remarks
-b orbackup[=suffix]	creates a backup of destination file
- d	treats arguments as directory names
- D	creates all leading components before copying SOURCE to DEST
-g and -o	sets ownership
- m	sets permissions
- p	preserves the timestamps of the sources files
- t	copies all source arguments to the directory

🔪 Note

There are options for managing the SELinux context (see the manual page).

Examples:

Create a directory with the -d option:

install -d ~/samples

Copy a file from a source location to a directory:

install src/sample.txt ~/samples/

These two orders could have been carried out with a single command:

```
$ install -v -D -t ~/samples/ src/sample.txt
install: creating directory '~/samples'
'src/sample.txt' -> '~/samples/sample.txt'
```

This command already saves time. Combine it with owner, owner group, and rights management to improve the time savings:

sudo install -v -o rocky -g users -m 644 -D -t ~/samples/ src/sample.txt

🖍 Note

sudo is required in this case to make property changes.

You can also create a backup of existing files thanks to the -b option:

```
$ install -v -b -D -t ~/samples/ src/sample.txt
'src/sample.txt' -> '~/samples/sample.txt' (archive: '~/samples/sample.txt~')
```

As you can see, the install command creates a backup file with a rilde appended to the original file name.

The suffix can be specified thanks to the -s option:

```
$ install -v -b -S ".bak" -D -t ~/samples/ src/sample.txt
'src/sample.txt' -> '~/samples/sample.txt' (archive: '~/samples/
sample.txt.bak')
```

5.7 tree command

Expand the files or directories in the directory in a tree-like manner.

options	description	
- a	All files are listed	
-h	Prints the size in a more human-readable way	
- U	Displays file owner or UID number	
- g	Displays file group owner or GID number	
- p	Print the protections for each file	

For example:

```
$ tree -hugp /etc/yum.repos.d/
/etc/yum.repos.d/
  — [-rw-r--r-- root
                                           epel-modular.repo
                                   1.6K]
                         root
   - [-rw-r--r-- root
                                           epel.repo
                         root
                                   1.3K]
   [-rw-r--r-- root
                                   1.7K]
                                           epel-testing-modular.repo
                         root
                                           epel-testing.repo
  - [-rw-r--r-- root
                         root
                                   1.4K]
                                     710]
                                           Rocky-AppStream.repo
   · [-rw-r--r-- root
                         root
  - [-rw-r--r-- root
                         root
                                     695]
                                           Rocky-BaseOS.repo
   [-rw-r--r-- root
                         root
                                   1.7K]
                                           Rocky-Debuginfo.repo
                                     360]
                                           Rocky-Devel.repo
   - [-rw-r--r-- root
                         root
   - [-rw-r--r-- root
                         root
                                     695]
                                           Rocky-Extras.repo
                                           Rocky-HighAvailability.repo
  - [-rw-r--r-- root
                         root
                                     731]
                                     680]
                                           Rocky-Media.repo
  - [-rw-r--r-- root
                         root
                                           Rocky-NFV.repo
   - [-rw-r--r-- root
                                    680]
                         root
   - [-rw-r--r-- root
                                    690]
                                           Rocky-Plus.repo
                         root
   [-rw-r--r-- root
                                    715]
                                           Rocky-PowerTools.repo
                         root
  - [-rw-r--r-- root
                                           Rocky-ResilientStorage.repo
                         root
                                    746]
                                           Rocky-RT.repo
   [-rw-r--r-- root
                         root
                                     681]
                                           Rocky-Sources.repo
  - [-rw-r--r-- root
                         root
                                    2.3K]
```

0 directories, 17 files

5.8 stat command

The stat command displays the status of a file or file system.

```
$ stat /root/anaconda-ks.cfg
  File: /root/anaconda-ks.cfg
  Size: 1352
                       Blocks: 8
                                          IO Block: 4096
                                                           regular file
Device: 10302h/66306d
                       Inode: 2757097
                                          Links: 1
Access: (0755/-rwxr-xr-x) Uid: ( 0/
                                          root) Gid: (
                                                            0/
                                                                  root)
Access: 2024-01-20 13:04:57.012033583 +0800
Modify: 2023-09-25 14:04:48.524760784 +0800
Change: 2024-01-24 16:37:34.315995221 +0800
 Birth: 2
```

- File Displays the path location of the file.
- Size Displays the file size in bytes. If this is a directory, it displays the fixed 4096 bytes occupied by the directory name.
- Blocks Displays the number of allocated blocks. Attention, please! The size of each block in this command is 512 bytes. The default size of each block in ls -ls is 1024 bytes.
- Device Device number in decimal or hexadecimal notation.
- Inode Inode is a unique ID number the Linux kernel assigns to a file or directory.
- Links Number of hard links. Hard links are sometimes referred to as physical links.
- Access The last access time of files and directories, i.e. atime in GNU/Linux.
- Modify The last modification time of files and directories, i.e. mtime in GNU/ Linux.
- Change The last time the property is changed, i.e. ctime in GNU/Linux.
- Birth Birth time (Creation time). In some documents, it is abbreviated as btime or crtime. You need a file system and kernel version higher than a certain version to display the creation time.

For files:

atime - After accessing the file content using commands such as cat, less, more, and head, the atime of the file can be updated. Please pay attention! The atime of

the file is not updated in real-time, and for performance reasons, it needs to wait for a period of time before it can be displayed. **mtime** - Modifying the file content can update the mtime of the file (such as appending or overwriting the file content through redirection), because the file size is a property of the file, the ctime will also be updated simultaneously. **ctime** - Changing the owner, group, permissions, file size, and links (soft and hard links) of the file will update ctime.

For directories:

atime - After using the cd command to enter a new directory that has never been accessed before, you can update and fix the atime of that directory. **mtime** - Performing operations such as creating, deleting, and renaming files in this directory will update the mtime and ctime of the directory. **ctime** - When the permissions, owner, group, etc. of a directory change, the ctime of the directory will be updated.

🜢 Tip

- \bullet If you create a new file or directory, its $\verb"atime"$, $\verb"mtime"$, and $\verb"ctime"$ are exactly the same
- If the file content is modified, the mtime and ctime of the file will inevitably be updated.
- If a brand new file is created in the directory, the atime, ctime, and mtime of that directory will be updated simultaneously.
- If the mtime of a directory is updated, then the ctime of that directory must be updated.

6. VI Text Editor

In this chapter you will learn how to work with the VIsual editor.

Objectives: In this chapter, future Linux administrators will learn how to:

Use the main commands of the VI editor;Modify a text with the VI editor.

🕅 user commands, linux

Knowledge: ★ Complexity: ★ ★

Reading time: 20 minutes

Visual (**VI**) is a popular text editor under Linux despite its limited ergonomics. It is indeed an editor entirely in text mode: each action is done with a key on the keyboard or dedicated commands.

Very powerful, it is above all very practical since it is on the whole minimal for basic applications. It is therefore accessible in case of system failure. Its *universality* (it is present on all Linux distributions and under Unix) makes it a *crucial* tool for the administrator.

Its functionalities are:

- Insert, delete, and modify text;
- Copy words, lines, or blocks of text;
- Search and replace characters.

6.1 vi command

The vi command opens the VI text editor.

vi [-c command] [file]

Example:

vi /home/ro	ckstar/file
Option	Information
-c command	Execute VI by specifying a command at the opening

If the file exists at the location mentioned by the path, VI reads it and puts it in **commands** mode.

If the file does not exist, VI opens a blank file, displaying an empty page on the screen. When the file is saved, it will take the name specified with the command.

If the command vi is executed without specifying a file name, VI opens a blank file and displays an empty page on the screen. When the file is saved, VI will ask for a file name.

The vim editor takes the interface and functions of VI with many improvements.

vim [-c command] [file]

Among these improvements, the user has syntax highlighting, which is useful for editing shell scripts or configuration files.

During a session, VI uses a buffer file to record all the user's changes.

Note
The original file is not modified as long as the user has not saved his work.

At startup, VI is in *commands* mode.

🜢 Tip

A line of text is ended by pressing Enter d but if the screen is not wide enough, VI makes automatic line breaks, wrap configuration by default. These line breaks may not be desired, this is the *nowrap* configuration.

To exit VI from the Commands mode, press [:], then type:

- q to exit without saving (quit);
- w to save your work (*write*);
- wq (write quit) or \times (eXit) to save and exit.

In command mode, Click the [z] key of uppercase status twice in a row to save and exit.

You must add ! to the previous commands to force the exit without confirmation.

 Warning

 There is no periodic backup, so you must remember to save your work regularly.

6.2 Operating mode

In VI, there are 3 working modes:

- The *command* mode;
- The *insertion* mode;
- The *ex* mode.

The philosophy of VI is to alternate between the *command* mode and the *insertion* mode.

The third mode, *ex*, is a footer command mode from an old text editor.

6.2.1 The Command Mode

This is the default mode when VI starts up. To access it from any of the other modes, simply press the $\[\mbox{\tiny Esc}\]$ key.

At this time, all keyboard typing is interpreted as commands and the corresponding actions are executed. These are essentially commands for editing text (copy, paste, undo, ...).

The commands are not displayed on the screen.

6.2.2 The Insert mode

This is the text modification mode. To access it from the *command* mode, you have to press special keys that will perform an action in addition to changing the mode.

The text is not entered directly into the file but into a buffer zone in the memory. The changes are only effective when the file is saved.

6.2.3 The Ex mode

This is the file modification mode. To access it, you must first switch to *command* mode, then enter the *ex* command frequently starting with the character : .

The command is validated by pressing the Enter a key.

6.3 Moving the cursor

In *command* mode, there are several ways to move the cursor.

The mouse is not active in a text environment but is in a graphic environment, it is possible to move it character by character, but shortcuts exist to go faster.

VI remains in *command* mode after moving the cursor.

The cursor is placed under the desired character.

6.3.1 From a character

• Move one or n characters to the left:



• Move one or n characters to the right:

→ Right ,	n	\rightarrow Right,	[1] or	n	[1]]
-----------	---	----------------------	----	------	---	-----	---

• Move one or n characters up:



• Move one or n characters down:



- Move to the end of the line:
- \$ Or ↓ End
- Move to the beginning of the line:

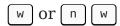
6.3.2 From the first character of a word

Words are made up of letters or numbers. Punctuation characters and apostrophes separate words.

If the cursor is in the middle of a word w moves to the next word, b moves to the beginning of the word.

If the line is finished, VI goes automatically to the next line.

• Move one or n words to the right:



• Move one or n words to the left:



- 6.3.3 From any location on a line
- Move to last line of text:

G

• Move to line n:



• Move to the first line of the screen:

Н

• Move to the middle line of the screen:

M

• Move to the last line of the screen:

L

• Move to the first line of the file content



6.4 Inserting text

There are several ways to insert text in *command* mode.

VI switches to *insert* mode after entering one of these keys.

Note
VI switches to *insertion* mode. So you will have to press the SEC key to return to *command* mode.

6.4.1 In relation to a character

• Inserting text before a character:



• Inserting text after a character:

a (append)

6.4.2 In relation to a line

• Inserting text at the beginning of a line:

I

• Inserting text at the end of a line:

Α

6.4.3 In relation to the text

- Inserting text before a line:
- 0
- Inserting text after a line:
- [0]

6.5 Characters, words and lines

VI allows text editing by managing:

- characters,
- words,
- lines.

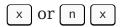
In each case it is possible to :

- delete,
- replace,
- сору,
- cut,
- paste.

These operations are done in *command* mode.

6.5.1 Characters

• Delete one or n characters:



• Replace a character with another:

r + character

• Replace more than one character with others:

$R + Characters + \otimes Esc$
Note
The R command switches to <i>replace</i> mode, which is a kind of <i>insert</i> mode.

6.5.2 Words

• Delete (cut) one or n words:



• Copy one or n words:



• Paste a word once or n times after the cursor:



• Paste a word once or n times before the cursor:

P or n + P

• Replace one word:



🗴 Tip

It is necessary to position the cursor under the first character of the word to cut (or copy) otherwise VI will cut (or copy) only the part of the word between the cursor and the end. To delete a word is to cut it. If it is not pasted afterwards, the buffer is emptied and the word is deleted.

6.5.3 Lines

• Delete (cut) one or n lines:

d + d or n + d + d

• Copy one or n lines:

y + y or n + y + y

• Paste what has been copied or deleted once or n times after the current line:

p or n + p

• Paste what has been copied or deleted once or n times before the current line:

P or n + P

• Delete (cut) from the beginning of the line to the cursor:



• Delete (cut) from the cursor to the end of the line:

• Copy from the beginning of the line to the cursor:

y + 0

• Copy from the cursor to the end of the line:



• Delete (cut) the contents from the cursor line to the last line of the file:



• Delete (cut) the contents from the cursor line to the last line of the screen:



• Copy the content from the cursor line to the end of the file:



• Copy the content from the cursor line to the end of the screen

y + L

6.5.4 Cancel an action

• Undo the last action:

u

• Undo the actions on the current line:

U

6.5.5 Cancel cancellation

• Cancel a cancellation

^ Ctrl + R

6.6 EX commands

The Ex mode allows you to act on the file (saving, layout, options, ...). It is also in Ex mode where search and replace commands are entered. The commands are displayed at the bottom of the page and must be validated with the Enter entered key.

To switch to *Ex* mode, from *command* mode, type [:].

6.6.1 File line numbers

• Show/hide numbering:

:set nu or the longer :set number

:set nonu or the longer :set nonumber

6.6.2 Search for a string

• Search for a string from the cursor:

/string

• Search for a string before the cursor:

?string

• Find the next matching string:

n

• Find the previous matching string:

N

There are regular expressions to facilitate the search in VI.

• [] : Searches for a range of characters or a single character whose possible values are specified.

Example:

/[Ww]ord : search word or Word

/[1-9]word : search 1word, 2word ... x word where x is a number

• 🔼 : Search for lines that begin with characters.

Example:

/^Word

• **\$** : Search for lines that end with characters.

Example:

/Word\$

• . : Search for any single character except newline characters.

Example:

/W.rd : search Word, Ward ...

• * : The number of times the previous character matches, 0 times, or any number of times.

Example:

/W*d

Note: If you want to ignore case (temporary) when matching strings, Please type the :set ic.

6.6.3 Replace a string

From the 1st to the last line of the text, replace the searched string by the specified string:

:1,\$ s/search/replace

Note: You can also use :0,\$s/search/replace to specify starting at the absolute beginning of the file.

From line n to line m, replace the searched string with the specified string:

:n,m s/search/replace

By default, only the first occurrence found of each line is replaced. To force the replacement of each occurrence, you have to add /g at the end of the command:

:n,m s/search/replace/g

Browse an entire file to replace the searched string with the specified string:

:% s/search/replace

6.6.4 Deletes the specified row

• Delete a blank line

:g/^\$/d

• Delete lines with line numbers n to m

:n,md

• Delete the line on which the string is located

:g/string/d

• Delete a line that does not contain a string

:g!/string/d

• Delete all lines that begin with #

:g/^#/d

The **g** here stands for **global**.

6.6.5 File operations

• Save the file:

:w

• Save under another name:

:w file

 \bullet Save from line $\ n$ to line $\ m$ in another file:

:n,m w file

• Reload the last record of the file:

e!

• Paste the content of another file after the cursor:

:r file

• Quit editing a file without saving:

:q

• Quit editing a file that has been modified during the session but not saved:

:q!

• Exit the file and save:

:wq or :x

6.7 Other functions

Executing VI by specifying the options to be loaded for the session is possible. To do this, you must use the -c option:

vi -c "set nu" /home/rockstar/file

It is also possible to enter the *Ex* commands in a file named .exrc in the user's login directory. The commands will be read and applied at each VI or VIM startup.

6.7.1 vimtutor command

There is a tutorial for learning how to use VI. It is accessible with the command vimtutor .

vimtutor

6.7.2 visualization mode

This mode is a sub-item of the command mode. You can complete it by typing \lor or \lor ; the former's operation content is at the character level, and the latter's operation content is at the line level.

i Info

You can use the arrow keys to mark the character or line content you want to operate on.

character level

- Delete (cut) Type the v key to mark the character content you want to delete, and then type x to delete it
- Copy Type the v key to mark the character content to copy, and then type the v key to copy it

line level

- Delete (cut) Type the V key to mark the line to be deleted, and then type × to delete it
- Copy Type the V key to mark the line to copy, and then type the V key to copy it

7. User Management

In this chapter you will learn how to manage users.

Objectives: In this chapter, future Linux administrators will learn how to:

Add, delete or modify a group;
add, delete or modify a user;
understand the files associated with users and groups and learn how to manage them;
change the owner or the group owner of a file;
secure user accounts;
change identity.

Knowledge: $\star \star$ Complexity: $\star \star$

Reading time: 30 minutes

7.1 General

Each user must have a group called the user's **primary group**.

Several users can be part of the same group.

Groups other than the primary group are called the user's **supplementary** groups.



Each user has a primary group and can be invited into one or more supplementary groups.

Groups and users are managed by their unique numerical identifiers GID and UID.

- UID : User IDentifier. Unique user ID.
- GID : Group IDentifier. Unique group identifier.

The kernel recognizes Both UID and GID, meaning that the Super Admin is not necessarily the **root** user, as long as the **uid=0** user is the Super Admin.

The files related to users/groups are:

- /etc/passwd
- /etc/shadow
- /etc/group
- /etc/gshadow
- /etc/skel/
- /etc/default/useradd
- /etc/login.defs

O Danger

You should always use the administration commands instead of manually editing the files.

🖍 Note

Some commands in this chapter require administrator rights. By convention, we will specify the command sudo when commands are to be run with administrator rights. For the examples to work properly, please ensure your account has the right to use the sudo command.

7.2 Group management

Modified files, added lines:

- /etc/group
- /etc/gshadow

7.2.1 groupadd command

The groupadd command adds a group to the system.

groupadd [-f] [-g GID] group

Example:

sudo groupadd -g 1012 GroupeB

Option	Description
-g GID	Defines the GID of the group to create.
-f	The system chooses a GID if the one specified by the -g option already exists.
- r	Creates a system group with a GID between SYS_GID_MIN and SYS_GID_MAX. These two variables are defined in /etc/login.defs.

Group naming rules:

- No accents or special characters;
- Different from the name of an existing user or system files.

Note
inder Debian , the administrator should use, except in scripts intended to be portable to all Linux distributions, the addgroup and elgroup commands as specified in the man :
<pre>\$ man addgroup DESCRIPTION adduser and addgroup add users and groups to the system according to command line options and configuration information in /etc/adduser.conf. They are friendlier front ends to the low-level tools like useradd, groupadd and usermod programs, by default, choosing Debian policy conformant UID and GID values, creating a home directory with skeletal configuration, running a custom script, and other features.</pre>

7.2.2 Command groupmod

The groupmod command allows you to modify an existing group on the system.

```
groupmod [-g GID] [-n nom] group
```

Example:

sudo	groupmod	-g <mark>1016</mark> GroupP	
sudo	groupmod	-n GroupC GroupB	
Option	ı	Description	
-g GID		New GID of the group to mod	lify.

It is possible to change the name of a group, its GID, or both simultaneously.

After modification, the files belonging to the group have an unknown GID. They must be reassigned to the new GID.

sudo find / -gid 1002 -exec chgrp 1016 {} \;

7.2.3 groupdel command

The groupdel command deletes an existing group on the system.

groupdel group

Example:

sudo groupdel GroupC

🜢 Tip

When deleting a group, two conditions can occur:

• If a user has a unique primary group and you issue the groupdel command on that group, you will be prompted that there is a specific user under the group and it cannot be deleted.

• If a user belongs to a supplementary group (not the primary group for the user) and that group is not the primary group for another user on the system, then the groupdel command will delete the group without any additional prompts.

Examples:

```
$ sudo useradd test
$ id test
uid=1000(test) gid=1000(test) group=1000(test)
$ sudo groupdel test
groupdel: cannot remove the primary group of user 'test'
$ sudo usermod -g users -G test test
$ id test
uid=1000(test) gid=100(users) group=100(users),1000(test)
$ sudo groupdel test
```

🜢 Tip

When you delete a user using the userdel -r command, the corresponding primary group is also deleted. The primary group name is usually the same as the username.

🜢 Tip

Each group has a unique GID. Multiple users can use a group as a supplementary group. By convention, The GID of the super administrator is 0. The GIDS reserved for some services or processes is 201-999, called system groups or pseudo-user groups. The GID for users is usually greater than or equal to 1000. These are related to /etc/login.defs, which we will talk about later.

```
# Comment line ignored
shell > cat /etc/login.defs
MAIL_DIR
               /var/spool/mail
UMASK
               022
HOME_MODE
               0700
PASS_MAX_DAYS
               99999
PASS MIN DAYS
              0
PASS MIN LEN
               5
PASS WARN AGE
               7
UID_MIN
                        1000
UID_MAX
                        60000
SYS_UID_MIN
                         201
SYS_UID_MAX
                         999
GID_MIN
                        1000
GID_MAX
                       60000
SYS_GID_MIN
                         201
SYS GID MAX
                         999
CREATE HOME
               yes
USERGROUPS_ENAB yes
ENCRYPT METHOD SHA512
```

🜢 Tip

Since a user is necessarily part of a group, it is best to create the groups before adding the users. Therefore, a group may not have any members.

7.2.4 /etc/group file

This file contains the group information (separated by :).

```
$ sudo tail -1 /etc/group
GroupP:x:516:patrick
(1) (2)(3) (4)
```

- 1: Name of the group.
- 2: The group password is identified by x. The group password is stored in /etc/ gshadow.
- 3: GID.
- 4: Supplementary users in the group (excluding the unique primary user).

Note

 $Each line in the \ /etc/group \ file \ corresponds to \ a \ group. The \ primary \ user \ info \ is \ stored \ in \ /etc/passwd \ .$

7.2.5 /etc/gshadow file

This file contains the security information about the groups (separated by :).

```
$ sudo grep GroupA /etc/gshadow
GroupA:$6$2,9,v...SBn160:alain:rockstar
(1) (2) (3) (4)
```

- 1: Name of the group.
- 2: Encrypted password.
- 3: Name of the group administrator.
- 4: Supplementary users in the group (excluding the unique primary user).

Warning
The name of the group in /etc/group and /etc/gshadow must correspond one by one. That is, each line in the /etc/group file must have a corresponding line in the /etc/gshadow file.

An ! in the password indicates it is locked. Thus, no user can use the password to access the group (since group members do not need it).

7.3 User management

7.3.1 Definition

A user is defined as follows in the /etc/passwd file:

- 1: Login name;
- 2: Password identification, x indicates that the user has a password, the encrypted password is stored in the second field of /etc/shadow;
- 3: UID;
- 4: GID of the primary group;
- 5: Comments;
- 6: Home directory;
- 7: Shell (/bin/bash, /bin/nologin, ...).

There are three types of users:

- root(uid=0): the system administrator;
- system users(uid is one of the 201~999): Used by the system to manage application access rights;
- regular user(uid>=1000): Other account to log in to the system.

Modified files, added lines:

- /etc/passwd
- /etc/shadow

7.3.2 useradd command

The useradd command adds a user.

useradd [-u UID] [-g GID] [-d directory] [-s shell] login

Example:

Option	Description
-u UID	UID of the user to create.
-g GID	GID of the primary group. The GID here can also be a group name.
-G GID1, [GID2]	GID of the supplementary groups. The GID here can also be a group name. It is possible to specify many supplementary groups separated by commas.
-d directory	Creates the home directory.
-s shell	Specifies the user's shell.
-c COMMENT	Adds a comment.
- U	Adds the user to a group with the same name created simultaneously. If not specified, the creation of a group with the same name occurs when creating the user.
- M	Does not create the user's home directory.
- r	Creates a system account.

sudo useradd -u 1000 -g 1013 -d /home/GroupC/carine carine

At creation, the account has no password and is locked.

The user must assign a password to unlock the account.

When invoking the useradd command without any options, the following default settings are set for the new user:

- A home directory with the same name as the username is created;
- A primary group with the same name as the username is created;
- A default shell that points to /bin/bash is assigned to the user;
- The user's UID and primary group GID values are automatically deduced. This is usually a unique value between 1000 and 60,000.

🖍 Note	
	ngs and values are obtained from the following configuration files:
\$ sudo use	radd test1
	1 /etc/passwd 00:1000::/home/test1:/bin/bash
\$ tail -n	1 /etc/shadow

```
test1:!!:19253:0:99999:7:::
$ tail -n 1 /etc/group ; tail -n 1 /etc/gshadow
test1:x:1000:
test1:!::
```

Account naming rules:

- Lowercase letters, numbers, and underscores are allowed; other special characters such as asterisks, percent signs, and full-width symbols are not accepted.
- Although you can use an uppercase user name in RockyLinux, we do not recommend it;
- It is not recommended to start with numbers and underscores, although you may be allowed to do so;
- Different from the name of an existing group or system file;
- The user name can contain up to 32 characters.



The last directory is created by the useradd command, which takes the opportunity to copy the files from /etc/skel into it.

A user can belong to several groups besides their primary group.

Example:

```
sudo useradd -u 1000 -g GroupA -G GroupP,GroupC albert

Note
Under Debian, you will have to specify the -m option to force the creation of the login directory or set the CREATE_HOME variable in
the /etc/login.defs file. In all cases, the administrator should use the adduser and deluser commands as specified in the man,
except in scripts intended to be portable to all Linux distributions:

$ man useradd
DESCRIPTION
**useradd** is a low-level utility for adding users. On Debian, administrators should usually use **adduser(8)**
```

Default value for user creation

Modification of the file /etc/default/useradd.

useradd -D [-b directory] [-g group] [-s shell]

Example:

sudo useradd -D -g 1000 -b /home -s /bin/bash

Option	Description
- D	Sets the default values for user creation.
-b base_directory	Defines the base directory for the user's home directory. If you do not specify this option, use the HOME variable in the /etc/default/useradd file or /home/
-g group	Sets the default group.
-s shell	Sets the default shell.
-f	Sets the number of days after the password expires before disabling the account.
- e	Sets the date for disabling the account.

7.3.3 usermod command

The usermod command allows to modify a user.

usermod [-u UID] [-g GID] [-d directory] [-m] login

Example:

sudo usermod -u 1044 carine

Option	Description
- m	Associated with the -d option. Moves the contents of the old login directory to the new one. If the old home directory does not exist, creation of a new home directory does not occur; Creation of the new home directory occurs when it does not exist.
-l login	Modifies the login name. After you modify the login name, you also need to modify the name of the home directory to match it.
-e YYYY-MM-DD	Modifies the account expiration date.
- L	Locks the account permanently. That is, it adds an ! at the beginning of the /etc/shadow password field.
- U	Unlocks the account.
- a	Appends the user's supplementary groups, which must be used together with the -G option.
- G	Modifies the user's supplementary groups and overwrites previous supplementary groups.

Options identical to the useradd command.

To be modified, a user must be disconnected and have no running processes.

🗴 Tip

After changing the identifier, the files belonging to the user have an unknown UID. It must be reassigned to the new UID.

Where 1000 is the old UID and 1044 is the new one. Examples are as follows:

sudo find / -uid 1000 -exec chown 1044: {} \;

Locking and unlocking of user accounts. Examples are as follows:

```
$ usermod -L test1
$ grep test1 /etc/shadow
test1:!
$6$n.hxglA.X5r7X0ex$qCXeTx.kQVmqsPLeuvIQnNidnSHvFiD7bQTxU7PLUCmB0cPNd5meqX6AEKSQvC
$ usermod -U test1
```

The difference between the -aG option and the -G option can be explained by the following example:

```
$ sudo useradd test1
$ sudo passwd test1
$ sudo groupadd groupA ; sudo groupadd groupB ; sudo groupadd groupC ; sudo
groupadd groupD
$ id test1
```

```
uid=1000(test1) gid=1000(test1) groups=1000(test1)

$ sudo gpasswd -a test1 groupA
$ id test1
uid=1000(test1) gid=1000(test1) groups=1000(test1),1002(groupA)

$ sudo usermod -G groupB,groupC test1
$ id test1
uid=1000(test1) gid=1000(test1) groups=1000(test1),1003(groupB),1004(groupC)

$ sudo usermod -aG groupD test1
$ id test1
uid=1000(test1) gid=1000(test1) groups=1000(test1),1003(groupB),1004(groupC),
1005(groupD)
```

7.3.4 userdel command

The userdel command lets you delete a user's account.

sudo user	del -r carine	
Option	Description	
-r	Deletes the user's home directory and mail files located in the /var/spool/mail/ directory	
ó Tip		
To be deleted, a	user must be logged out and have no running processes.	

The userdel command removes the corresponding lines in /etc/passwd, / etc/ shadow, /etc/group, /etc/gshadow. As mentioned above, userdel -r will also delete the corresponding primary group of the user.

7.3.5 /etc/passwd file

This file contains user information (separated by :).

```
$ sudo head -1 /etc/passwd
root:x:0:0:root:/root:/bin/bash
(1)(2)(3)(4)(5) (6) (7)
```

- 1: Login name;
- 2: Password identification, × indicates that the user has a password, the encrypted password is stored in the second field of /etc/shadow;
- 3: UID;
- 4: GID of the primary group;
- 5: Comments;
- 6: Home directory;
- 7: Shell (/bin/bash, /bin/nologin,...).

7.3.6 /etc/shadow file

This file contains the users' security information (separated by ::).

```
$ sudo tail -1 /etc/shadow
root:$6$...:15399:0:99999:7:::
(1) (2) (3) (4) (5) (6)(7,8,9)
```

- 1: Login name.
- 2: Encrypted password. Uses the SHA512 encryption algorithm, defined by the ENCRYPT_METHOD of /etc/login.defs.
- 3: The time when the password was last changed, the timestamp format, in days. The so-called timestamp is based on January 1, 1970 as the standard time. Every time one day goes by, the timestamp is +1.
- 4: Minimum lifetime of the password. That is, the time interval between two password changes (related to the third field), in days. Defined by the PASS_MIN_DAYS of /etc/login.defs, the default is 0, that is, when you change the password for the second time, there is no restriction. However, if it is 5, it means that it is not allowed to change the password within 5 days, and only after 5 days.
- 5: Maximum lifetime of the password. That is, the validity period of the password (related to the third field). Defined by the PASS_MAX_DAYS of /etc/login.defs.
- 6: The number of warning days before the password expires (related to the fifth field). The default is 7 days, defined by the PASS_WARN_AGE of /etc/login.defs.
- 7: Number of days of grace after password expiration (related to the fifth field).
- 8: Account expiration time, the timestamp format, in days. Note that an account expiration differs from a password expiration. In case of an account expiration, the user shall not be allowed to login. In case of a password expiration, the user is not allowed to login using her password.
- 9: Reserved for future use.

```
Oanger
```

For each line in the /etc/passwd file there must be a corresponding line in the /etc/shadow file.

For time stamp and date conversion, please refer to the following command format:

```
# The timestamp is converted to a date, "17718" indicates the timestamp to be
filled in.
$ date -d "1970-01-01 17718 days"
```

```
# The date is converted to a timestamp, "2018-07-06" indicates the date to be
filled in.
$ echo $(($(date --date="2018-07-06" +%s)/86400+1))
```

7.4 File owners

🗘 Danger		
All files necessarily belong to one user and one group.		

By default, the primary group of the user creating the file is the group that owns the file.

7.4.1 Modification commands

chown command

The chown command allows you to change the owners of a file.

chown [-R] [-v] login[:group] file

Examples:

```
sudo chown root myfile
```

sudo chown albert:GroupA myfile

Option	Description
- R	Recursively changes the owners of the directory and all files under the directory.
- V	Displays the changes.

To change only the owner user:

sudo chown albert file

To modify only the owner group:

```
sudo chown :GroupA file
```

Changing the user and owner group:

sudo chown albert:GroupA file

In the following example the group assigned will be the primary group of the specified user.

```
sudo chown albert: file
```

Change the owner and group of all files in a directory

sudo chown -R albert:GroupA /dir1

7.4.2 chgrp command

The chgrp command allows you to change the owner group of a file.

```
chgrp [-R] [-v] group file
```

Example:

sudo chgrp group1 file

Option	Description
- R	Recursively changes the groups of the directory and all files under the directory.
- V	Displays the changes.

```
🖍 Note
```

It is possible to apply to a file an owner and an owner group by taking as reference those of another file:

chown [options] --reference=RRFILE FILE

For example:

chown --reference=/etc/groups /etc/passwd

7.5 Guest management

7.5.1 gpasswd command

The command gpasswd allows to manage a group.

gpasswd [option] group

Examples:

\$ sudo gpasswd -A alain GroupA
[alain]\$ gpasswd -a patrick GroupA

Option	Description
-a USER	Adds the user to the group. For the added user, this group is a supplementary group.
-A USER,	Sets the list of administrative users.
-d USER	Removes the user from the group.
-M USER,	Sets the list of group members.

The command gpasswd -M acts as a modification, not an addition.

```
# gpasswd GroupeA
New Password:
Re-enter new password:
```

Note

In addition to using gpasswd -a to add users to a group, you can also use the usermod -G or usermod -aG mentioned earlier.

7.5.2 id command

The id command displays the group names of a user.

id USER

Example:

```
$ sudo id alain
uid=1000(alain) gid=1000(GroupA) groupes=1000(GroupA),1016(GroupP)
```

7.5.3 newgrp command

The newgrp command can select a group from the user's supplementary groups as the user's new **temporary** primary group. The newgrp command every time you switch a user's primary group, there will be a new **child shell** child process). Be careful! **child shell** and **sub shell** are different.

```
newgrp [secondarygroups]
```

Example:

```
$ sudo useradd test1
$ sudo passwd test1
$ sudo groupadd groupA ; sudo groupadd groupB
$ sudo usermod -G groupA,groupB test1
$ id test1
uid=1000(test1) gid=1000(test1) groups=1000(test1),1001(groupA),1002(groupB)
$ echo $SHLVL ; echo $BASH_SUBSHELL
1
0
$ su - test1
$ touch a.txt
$ 11
-rw-rw-r-- 1 test1 test1 0 10 7 14:02 a.txt
$ echo $SHLVL ; echo $BASH_SUBSHELL
1
0
# Generate a new child shell
$ newgrp groupA
$ touch b.txt
$ 11
-rw-rw-r-- 1 test1 test1 0 10 7 14:02 a.txt
-rw-r--r-- 1 test1 groupA 0 10 7 14:02 b.txt
$ echo $SHLVL ; echo $BASH_SUBSHELL
2
0
# You can exit the child shell using the `exit` command
$ exit
$ logout
$ whoami
root
```

7.6 Securing

7.6.1 passwd command

The passwd command manages a password.

```
passwd [-d] [-l] [-S] [-u] [login]
```

Examples:

sudo passwd -l albert

sudo passwd -n 60 -x 90 -w 80 -i 10 patrick

Option	Description
- d	Permanently removes the password. For root (uid=0) use only.
-1	Permanently locks the user account. For root (uid=0) use only.
- S	Displays the account status. For root (uid=0) use only.
- U	Permanently unlocks user account. For root (uid=0) use only.
- e	Permanently expires the password. For root $(uid=0)$ use only.
-n DAYS	Defines the minimum password lifetime. Permanent change. For root (uid=0) use only.
-x DAYS	Defines the maximum password lifetime. Permanent change. For root (uid=0) use only.
-w DAYS	Defines the warning time before expiration. Permanent change. For root (uid=0) use only.
-i DAYS	Defines the delay before deactivation when the password expires. Permanent change. For root (uid=0) use only.

Use password -1, that is, add "!!" at the beginning of the password field of the user corresponding to /etc/shadow.

Example:

• Alain changes his password:

[alain]\$ passwd

• root changes Alain's password

sudo passwd alain

Note
Users logged in to the system can use the passwd command to change their passwords (this process requires requesting the user's old password). The root(uid=0) user can change the password of any user.

Changing passwords requires compliance with prescribed security policies, which involves **PAM (Pluggable Authentication Modules)** knowledge.

When managing user accounts by shell script, setting a default password after creating the user may be useful.

This can be done by passing the password to the passwd command.

Example:

sudo echo "azerty,1" | passwd --stdin philippe

🛕 Warning

The password is entered in clear text, passwd encrypts it.

7.6.2 chage command

The chage command is to change user password expiry information.

chage [-d date] [-E date] [-I days] [-l] [-m days] [-M days] [-W days] [login]

Example:

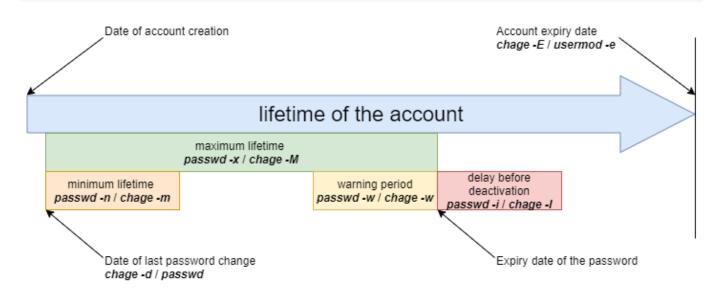
sudo chage -m 60 -M 90 -W 80 -I 10 alain

Option	Description
-I DAYS	Defines the days to delay before deactivation, password expired. Permanent change.
-1	Displays the policy details.
-m DAYS	Defines the minimum lifetime of the password. Permanent change.
-M DAYS	Defines the maximum lifetime of the password. Permanent change.
-d LAST_DAY	Defines the number of days since the password was last changed. You can use the days' timestamp style or the YYYY-MM-DD style. Permanent change.
-E EXPIRE_DATE	Defines the account expiration date. You can use the days' timestamp style or the YYYY-MM-DD style. Permanent change.
-W WARN_DAYS	Defines the number of days warning time before expiration. Permanent change.

Examples:

```
# The `chage` command also offers an interactive mode.
```

- \$ sudo chage philippe
- # The `-d` option changes the password when logging in.
- \$ sudo chage -d 0 philippe



7.7 Advanced management

Configuration files:

- /etc/default/useradd
- /etc/login.defs
- /etc/skel

/ Note

 $Editing \ the \ \verb/etc/default/useradd \ file \ is \ done \ with \ the \ useradd \ command.$

The other files are to be modified with a text editor.

7.7.1 /etc/default/useradd file

This file contains the default data settings.

🜢 Tip

If the options are not specified when creating a user, the system uses the default values defined in /etc/default/useradd.

This file is modified by the command useradd -D (useradd -D entered without any other option displays the contents of the /etc/default/useradd file).

```
Shell > grep -v ^# /etc/default/useradd
GROUP=100
HOME=/home
INACTIVE=-1
EXPIRE=
SHELL=/bin/bash
```

SKEL=/etc/skel CREATE_MAIL_SPOOL=yes

Parameters	Comment
GROUP	Defines the default primary group GID.
HOME	Defines the directory path of the upper level of the common user's home directory.
INACTIVE	Defines the number of days of grace after password expiration. Corresponds to the 7th field of the /etc/shadow file1 value means that the grace period feature is turned off.
EXPIRE	Defines the account expiration date. Corresponds to the 8th field of the /etc/shadow file.
SHELL	Defines the command interpreter.
SKEL	Defines the skeleton directory of the login directory.
CREATE_MAIL_SPOOL	Defines the mailbox creation in /var/spool/mail/.

If you do not need a primary group with the same name when creating users, you can do this:

```
Shell > useradd -N test2
Shell > id test2
uid=1001(test2) gid=100(users) groups=100(users)
```

🖍 Note

GNU/Linux has two group mechanisms:

. Public group, its primary group is GID=100

2. Private group, that is, when adding users, a group with the same name is created as its primary group. This group mechanism is commonly used by RHEL and related downstream distributions.

7.7.2 /etc/login.defs file

# Comment line i	ignored
<pre>shell > cat /et</pre>	tc/login.defs
MAIL_DIR	/var/spool/mail
UMASK	022
HOME_MODE	0700
PASS_MAX_DAYS	99999
PASS_MIN_DAYS	0
PASS_MIN_LEN	5
PASS_WARN_AGE	7
UID_MIN	1000
UID_MAX	60000
SYS_UID_MIN	201

SYS_UID_MAX999GID_MIN1000GID_MAX60000SYS_GID_MIN201SYS_GID_MAX999CREATE_HOMEyesUSERGROUPS_ENAByesENCRYPT_METHODSHA512

UMASK 022 : This means that the permission to create a file is 755 (rwxr-xr-x). However, for security, GNU/Linux does not have **x** permission for newly created files. This restriction applies to root(uid=0) and ordinary users(uid>=1000). For example:

Shell > touch a.txt
Shell > ll
-rw-r--r-- 1 root root 0 Oct 8 13:00 a.txt

HOME_MODE 0700 : The permissions of an ordinary user's home directory. Does not work for root's home directory.

```
Shell > ll -d /root
dr-xr-x---. 10 root root 4096 Oct 8 13:12 /root
Shell > ls -ld /home/test1/
drwx----- 2 test1 test1 4096 Oct 8 13:10 /home/test1/
```

USERGROUPS_ENAB yes: "When you delete a user using the userdel -r command, the corresponding primary group is also deleted." Why? That's the reason.

7.7.3 /etc/skel directory

When a user is created, their home directory and environment files are created. You can think of the files in the /etc/skel/ directory as the file templates you need to create users. These files are automatically copied from the /etc/skel directory.

- .bash_logout
- .bash_profile
- .bashrc

All files and directories placed in this directory will be copied to the user tree when created.

7.8 Identity change

7.8.1 su command

The su command allows you to change the identity of the connected user.

```
su [-] [-c command] [login]
```

Examples:

```
$ sudo su - alain
[albert]$ su - root -c "passwd alain"
```

Option	Description			
	Loads the user's complete environment.			
-c command	Executes the command under the user's identity.			

If the login is not specified, it will be root.

Standard users will have to type the password for the new identity.

```
Tip
You can use the exit / logout command to exit users who have been switched. It should be noted that after switching users, there is no new child shell or sub shell, for example:
$ whoami root
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        $ su - test1
        $ echo $SHLVL ; echo $BASH_SUBSHELL
        1
        0
        }
```

Attention please! su and su - are different, as shown in the following example:

```
$ whoami
test1
$ su root
$ pwd
/home/test1
$ env
. . .
USER=test1
PWD=/home/test1
HOME=/root
MAIL=/var/spool/mail/test1
LOGNAME=test1
. . .
$ whoami
test1
$ su - root
$ pwd
/root
$ env
. . .
USER=root
PWD=/root
HOME=/root
MAIL=/var/spool/mail/root
LOGNAME=root
. . .
```

So, when you want to switch users, remember not to lose the -. Because the necessary environment variable files are not loaded, there may be problems running some programs.

8. File System

In this chapter, you will learn how to work with file systems.

Objectives: In this chapter, future Linux administrators will learn how to:

manage partitions on disk;
use LVM for a better use of disk resources;
provide users with a filesystem and manage the access rights.
and also discover:

how the tree structure is organized in Linux;
the different types of files offered and how to work with them;

🕅 hardware, disk, partition, lvm, linux

Knowledge: $\star \star$ Complexity: $\star \star$

Reading time: 20 minutes

8.1 Partitioning

Partitioning will allow the installation of several operating systems because it is impossible for them to cohabit on the same logical drive. It also allows the separation of data logically (security, access optimization, etc.).

The partition table, stored in the first sector of the disk (MBR: *Master Boot Record*), records the division of the physical disk into partitioned volumes.

For **MBR** partition table types, the same physical disk can be divided into a maximum of 4 partitions:

- Primary partition (or main partition)
- Extended partition

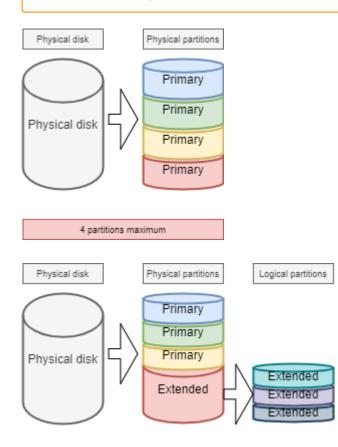
A Warning

There can be only one extended partition per physical disk. That is, a physical disk can have in the MBR partition table up to:

. Three primary partitions plus one extended partition

2. Four primary partitions

The extended partition cannot write data and format and can only contain logical partitions. The largest physical disk that the MBR partition table can recognize is **2TB**.



8.1.1 Naming conventions for device file names

In the world of GNU/Linux, everything is a file. For disks, they are recognized in the system as:

Hardware	Device file name
IDE hard disk	/dev/hd[a-d]
SCSI/SATA/USB hard disk	/dev/sd[a-z]
Optical drive	/dev/cdrom or /dev/sr0
Floppy disk	/dev/fd[0-7]
Printer (25 pins)	/dev/lp[0-2]
Printer (USB)	/dev/usb/lp[0-15]
Mouse	/dev/mouse
Virtual hard disk	/dev/vd[a-z]

The Linux kernel contains drivers for most hardware devices.

What we call *devices* are the files stored without /dev, identifying the different hardware detected by the motherboard.

The service called udev is responsible for applying the naming conventions (rules) and applying them to the devices it detects.

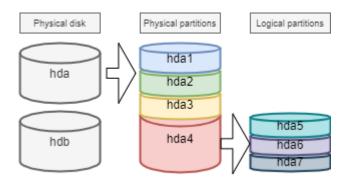
For more information, please see here.

8.1.2 Device partition number

The number after the block device (storage device) indicates a partition. For MBR partition tables, the number 5 must be the first logical partition.

🔺 Warning

Attention please! The partition number we mentioned here mainly refers to the partition number of the block device (storage device).



There are at least two commands for partitioning a disk: fdisk and cfdisk. Both commands have an interactive menu. cfdisk is more reliable and better optimized, so it is best to use it.

The only reason to use fdisk is when you want to list all logical devices with the -1 option. fdisk uses MBR partition tables, so it is not supported for **GPT** partition tables and cannot be processed for disks larger than **2TB**.

sudo fdisk -l sudo fdisk -l /dev/sdc sudo fdisk -l /dev/sdc2

8.1.3 parted command

The parted (*partition editor*) command can partition a disk without the drawbacks of fdisk.

The parted command can be used on the command line or interactively. It also has a recovery function capable of rewriting a deleted partition table.

```
parted [-l] [device]
```

Under the graphical interface, there is the very complete gparted tool: Gnome *PAR*tition *ED*itor.

The gparted -l command lists all logical devices on a computer.

The gparted command, when run without any arguments, will show an interactive mode with its internal options:

- help or an incorrect command will display these options.
- print all in this mode will have the same result as gparted -1 on the command line.
- quit to return to the prompt.

8.1.4 cfdisk command

The cfdisk command is used to manage partitions.

cfdisk device

Example:

\$ sudo cfdisk	/dev/sda					
		Dis	sk: /dev/sda			
	Size: <mark>16</mark> Gi	B, 1717986	6 <mark>9184</mark> bytes,	33554432	sectors	
	Lab	el: dos, i	identifier:	0xcf173747		
Device	Boot	Start	End	Sectors	Size	Id Type
>> /dev/sda1	*	2048	2099199	2097152	1G	<mark>83</mark> Linux
/dev/sda2		2099200	33554431	31455232	15G	8e Linux LVM
lqqqqqqqqqqq	qqqqqqqqqqqqqq	Idddddddddd	qqqqqqqqqqqq	qqqqqqqqqqq	qqqqqqqq	qqqqqqqqqqqqqk
x Partition	cype: Linux (83)				Х
x Attrib	utes: <mark>80</mark>					Х
xFilesystem	JUID: 54a1f5a	17-b8fa-474	47-a87c-2dd6	35914d60		Х
x Filesy	stem: xfs					Х
x Mountp	oint: /boot (mounted)				Х
mqqqqqqqqqqqq	qqqqqqqqqqqqqq	Idddddddddd	qqqqqqqqqqqqq	qqqqqqqqqqq	qqqqqqqq	qqqqqqqqqqqqqj
[Bootabl	e] [Delete] [Resiz	ze] [Qui	t][Ty	/pe]	[Help]
[Write] [Dump]				

The preparation, without *LVM*, of the physical media goes through five steps:

- Setting up the physical disk;
- Partitioning of the volumes (a division of the disk, possibility of installing several systems, ...);
- Creation of the file systems (allows the operating system to manage the files, the tree structure, the rights, ...);
- Mounting of file systems (registration of the file system in the tree structure);
- Manage user access.

8.2 Logical Volume Manager (LVM)

Logical Volume Manager (LVM)

The partition created by the **standard partition** cannot dynamically adjust the resources of the hard disk, once the partition is mounted, the capacity is completely fixed, this constraint is unacceptable on the server. Although the standard partition can be forcibly expanded or shrunk through certain technical means, it can easily cause data loss. LVM can solve this problem very well. LVM is available under Linux from kernel version 2.4, and its main features are:

- More flexible disk capacity;
- Online data movement;
- Disks in *stripe* mode;
- Mirrored volumes (recopy);
- Volume snapshots (*snapshot*).

The principle of LVM is very simple:

- a logical abstraction layer is added between the physical disk (or disk partition) and the file system
- merge multiple disks (or disk partition) into Volume Group(VG)
- perform underlying disk management operations on them through something called Logical Volume(**LV**).

The physical media: The storage medium of the LVM can be the entire hard disk, disk partition, or RAID array. The device must be converted, or initialized, to an LVM Physical Volume(**PV**), before further operations can be performed.

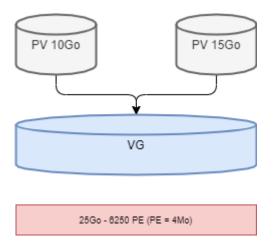
PV(Physical Volume) is the basic storage logic block of LVM. You can create a physical volume by using a disk partition or the disk itself.

VG(Volume Group): Similar to physical disks in a standard partition, a VG consists of one or more PV.

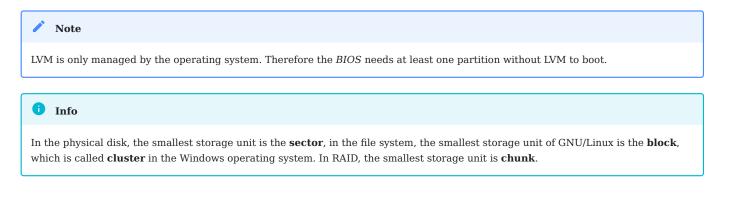
LV(Logical Volume): Similar to hard disk partitions in standard partitions, LV is built on top of VG. You can set up a file system on LV.

PE: The smallest unit of storage that can be allocated in a Physical Volume, default to **4MB**. You can specify an additional size.

LE: The smallest unit of storage that can be allocated in a Logical Volume. In the same VG, PE, and LE are the same and correspond one to one.



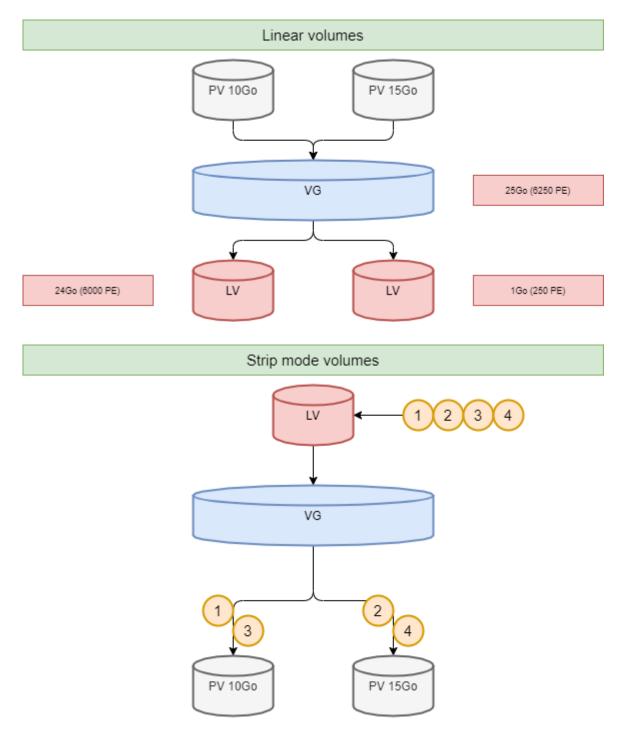
The disadvantage is that if one of the physical volumes becomes out of order, then all the logical volumes that use this physical volume are lost. You will have to use LVM on raid disks.



8.2.1 The Writing Mechanism of LVM

There are several storage mechanisms when storing data to LV, two of which are:

- Linear volumes;
- Volumes in *stripe* mode;
- Mirrored volumes.



8.2.2 LVM commands for volume management

Item	PV	VG	LV
scan	pvscan	vgscan	lvscan
create	pvcreate	vgcreate	lvcreate
display	pvdisplay	vgdisplay	lvdisplay
remove	pvremove	vgremove	lvremove
extend		vgextend	lvextend
reduce		vgreduce	lvreduce
summary information	pvs	vgs	lvs

The main relevant commands are as follows:

pvcreate command

The pvcreate command is used to create physical volumes. It turns Linux partitions (or disks) into physical volumes.

pvcreate [-options] partition

Example:

```
[root]# pvcreate /dev/hdb1
pvcreate -- physical volume « /dev/hdb1 » successfully created
```

You can also use a whole disk (which facilitates disk size increases in virtual environments for example).

```
[root]# pvcreate /dev/hdb
pvcreate -- physical volume « /dev/hdb » successfully created
# It can also be written in other ways, such as
[root]# pvcreate /dev/sd{b,c,d}1
[root]# pvcreate /dev/sd[b-d]1
```

Option	Description
-f	Forces the creation of the volume (disk already transformed into physical volume). Use with extreme caution.

vgcreate command

The vgcreate command creates volume groups. It groups one or more physical volumes into a volume group.

vgcreate <VG_name> <PV_name...> [option]

Example:

[root]# vgcreate volume1 /dev/hdb1
...
vgcreate - volume group « volume1 » successfully created and activated
[root]# vgcreate vg01 /dev/sd{b,c,d}1
[root]# vgcreate vg02 /dev/sd[b-d]1

lvcreate command

The lvcreate command creates logical volumes. The file system is then created on these logical volumes.

lvcreate -L size [-n name] VG_name

Example:

```
[root]# lvcreate -L 600M -n VolLog1 volume1
lvcreate -- logical volume « /dev/volume1/VolLog1 » successfully created
```

Option	Description
-L size	Sets the logical volume size in K, M, or G.
-n name	Sets the LV name. A special file was created in /dev/name_volume with this name.
-l number	Sets the percentage of the capacity of the hard disk to use. You can also use the number of PE. One PE equals 4MB.

i Info

After you create a logical volume with the lvcreate command, the naming rule of the operating system is - /dev/VG_name/LV_name, this file type is a soft link (otherwise known as a symbolic link). The link file points to files like /dev/dm-0 and /dev/dm-1.

8.2.3 LVM commands to view volume information

pvdisplay command

The pvdisplay command allows you to view information about the physical volumes.

pvdisplay /dev/PV_name

Example:

[root]# pvdisplay /dev/PV_name

vgdisplay command

The vgdisplay command allows you to view information about volume groups.

vgdisplay VG_name

Example:

[root]# vgdisplay volume1

lvdisplay command

The lvdisplay command allows you to view information about the logical volumes.

lvdisplay /dev/VG_name/LV_name

Example:

[root]# lvdisplay /dev/volume1/VolLog1

8.2.4 Preparation of the physical media

The preparation with LVM of the physical support is broken down into the following:

- Setting up the physical disk
- Partitioning of the volumes
- LVM physical volume
- LVM volume groups
- LVM logical volumes
- Creating file systems
- Mounting file systems
- Manage user access

8.3 Structure of a file system

A *file system* **FS** is in charge of the following actions:

- Securing access and modification rights to files;
- Manipulating files: create, read, modify, and delete;
- Locating files on the disk;
- Managing partition space.

The Linux operating system is able to use different file systems (ext2, ext3, ext4, FAT16, FAT32, NTFS, HFS, BtrFS, JFS, XFS, ...).

8.3.1 mkfs command

The mkfs (make file system) command allows you to create a Linux file system.

mkfs [-t fstype] filesys

Example:

[root]# m	nkfs -t ext4 /dev/sda1	
Option	Description	
- t	Indicates the type of file system to use.	
A Warning		
Without a file s	Without a file system it is not possible to use the disk space.	

Each file system has an identical structure on each partition. The system initializes a **Boot Sector** and a **Super block**, and then the administrator initializes an **Inode table** and a **Data block**.

✓ Note	
The only exception is the swap partition.	

8.3.2 Boot sector

The boot sector is the first sector of bootable storage media, that is, 0 cylinder, 0 track, 1 sector(1 sector equals 512 bytes). It consists of three parts:

- 1. MBR(master boot record): 446 bytes.
- 2. DPT(disk partition table): 64 bytes.
- 3. BRID(boot record ID): 2 bytes.

Item	Description
MBR	Stores the "boot loader"(or "GRUB"); loads the kernel, passes parameters; provides a menu interface at boot time; transfers to another loader, such as when multiple operating systems are installed.
DPT	Records the partition status of the entire disk.
BRID	Determines whether the device is usable to boot.

8.3.3 Super block

The size of the **Super block** table is defined at creation. It is present on each partition and contains the elements necessary for its utilization.

It describes the File System:

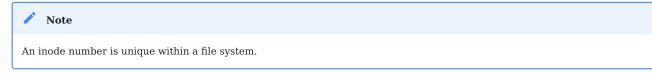
- Name of the Logical Volume;
- Name of the File System;
- Type of the File System;
- File System Status;
- Size of the File System;
- Number of free blocks;
- Pointer to the beginning of the list of free blocks;
- Size of the inode list;
- Number and list of free inodes.

After the system is initialized, a copy is loaded into the central memory. This copy is updated as soon as modified, and the system saves it periodically (command sync).

When the system stops, it copies this table in memory to its block.

8.3.4 Table of inodes

The size of the **inode table** is defined at its creation and is stored on the partition. It consists of records, called inodes, corresponding to the files created. Each record contains the addresses of the data blocks making up the file.



After the system is initialized, a copy is loaded into the central memory. This copy is updated as soon as it is modified, and the system saves it periodically (command sync).

When the system stops, it copies this table in memory to its block.

A file is managed by its inode number.

🖍 Note

The size of the inode table determines the maximum number of files the FS can contain.

Information present in the *inode table* :

- Inode number;
- File type and access permissions;
- Owner identification number;
- Identification number of the owner group;
- Number of links on this file;
- Size of the file in bytes;
- Date the file was last accessed;
- Date the file was last modified;
- Date of the last modification of the inode (= creation);
- Table of several pointers (block table) to the logical blocks containing the file pieces.

8.3.5 Data block

Its size corresponds to the rest of the partition's available space. This area contains the catalogs corresponding to each directory and the data blocks corresponding to the file's contents.

To guarantee the consistency of the file system, an image of the superblock and the inode table is loaded into memory (RAM) when the operating system is loaded so that all I/O operations are done through these system tables. When the user creates or modifies files, this memory image is updated first. The operating system must, therefore, regularly update the superblock of the logical disk (syncl command).

These tables are written to the hard disk when the system is shut down.

4 Attention

In the event of a sudden stop, the file system may lose its consistency and cause data loss.

8.3.6 Repairing the file system

It is possible to check the consistency of a file system with the fsck command.

In case of errors, solutions are proposed to repair the inconsistencies. After repair, files that remain without entries in the inode table are attached to the logical drive's /lost+found folder.

fsck command

The fsck command is a console-mode integrity check and repair tool for Linux file systems.

fsck [-sACVRTNP] [-t fstype] filesys

Example:

[root]# fsck /dev/sda1

To check the root partition, it is possible to create a forcefsck file and reboot or run shutdown with the -F option.

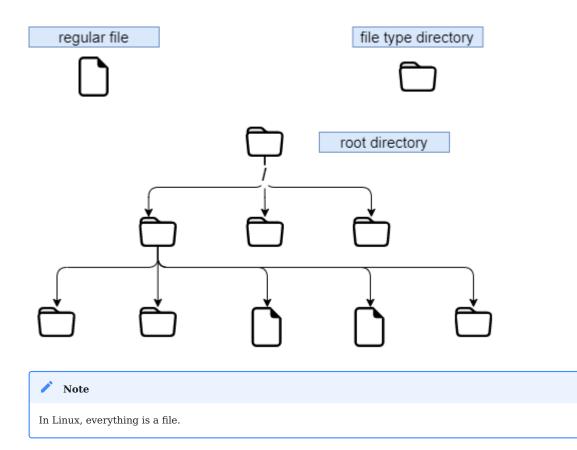
```
[root]# touch /forcefsck
[root]# reboot
or
[root]# shutdown -r -F now
```

🛕 Warning

The partition to be checked must be unmounted.

8.4 Organization of a file system

By definition, a File System is a tree structure of directories built from a root directory (a logical device can only contain one file system).



Text document, directory, binary, partition, network resource, screen, keyboard, Unix kernel, user program, ...

Linux meets the **FHS** (*Filesystems Hierarchy Standard*) (see man hier), which defines the folders' names and roles.

Directory	Functionality	Complete word
/	Contains special directories	
/boot	Files related to system startup	
/sbin	Commands necessary for system startup and repair	system binaries
/bin	Executables of basic system commands	binaries
/usr/bin	System administration commands	
/lib	Shared libraries and kernel modules	libraries
/usr	Saves data resources related to UNIX	UNIX System Resources
/mnt	Temporary mount point directory	mount
/media	For mounting removable media	
/misc	To mount the shared directory of the NFS service.	
/root	Administrator's login directory	
/home	The upper-level directory of a common user's home directory	
/tmp	The directory containing temporary files	temporary
/dev	Special device files	device
/etc	Configuration and script files	editable text configuration
/opt	Specific to installed applications	optional
/proc	This is a mount point for the proc filesystem, which provides information about running processes and the kernel	processes
/var	This directory contains files which may change in size, such as spool and log files	variables
/sys	Virtual file system, similar to /proc	
/run	That is /var/run	
/srv	Service Data Directory	service

- To mount or unmount at the tree level, you must not be under its mount point.
- Mounting on a non-empty directory does not delete the content. It is only hidden.
- Only the administrator can perform mounts.
- Mount points automatically mounted at boot time must be entered in /etc/fstab.

8.4.1 /etc/fstab file

The /etc/fstab file is read at system startup and contains the mounts to be performed. Each file system to be mounted is described on a single line, the fields being separated by spaces or tabs.

Note					
Lines are read sequentially (fsck , mount , umount)).				
/dev/mapper/VolGroup-lv_root	/	ext4	defaults	1	1
UUID=4692	/boot	ext4	defaults	1	2
/dev/mapper/VolGroup-lv_swap	swap	swap	defaults	0	Θ
tmpfs	/dev/shm	tmpfs	defaults	0	Θ
devpts	/dev/pts	devpts	gid=5,mode=620	0	Θ
sysfs	/sys	sysfs	defaults	Θ	Θ
proc	/proc	proc	defaults	0	Θ
1	2	3	4	5	6

Column	Description
1	File system device (/dev/sda1 , UUID=,)
2	Mount point name, absolute path (except swap)
3	Filesystem type (ext4, swap,)
4	Special options for mounting (defaults, ro,)
5	Enable or disable backup management (0:not backed up, 1:backed up). The dump command is used for backup here. This outdated feature was initially designed to back up old file systems on tape.
6	Check order when checking the FS with the fsck command (0:no check, 1:priority, 2:not priority)

The mount -a command allows you to mount automatically based on the contents of the configuration file /etc/fstab. The mounted information is then written to /etc/mtab.

🛕 Warning

Only the mount points listed in /etc/fstab will be mounted on reboot. Generally speaking, we do not recommend writing USB flash disks and removable hard drives to the /etc/fstab file because when the external device is unplugged and rebooted, the system will prompt that the device cannot be found, resulting in a failure to boot. So what am I supposed to do? Temporary mount, for example:

```
Shell > mkdir /mnt/usb
Shell > mount -t vfat /dev/sdb1 /mnt/usb
# Read the information of the USB flash disk
Shell > cd /mnt/usb/
# When not needed, execute the following command to pull out the USB flash disk
Shell > umount /mnt/usb
```

i Info

It is possible to make a copy of the /etc/mtab file or to copy its contents to /etc/fstab. If you want to view the UUID of the device partition number, type the following command: lsblk -o name, uuid. UUID is the abbreviation of Universally Unique Identifier.

8.4.2 Mount management commands

mount command

The mount command allows you to mount and view the logical drives in the tree.

```
mount [-option] [device] [directory]
```

Example:

[root]]#	mount	/c	lev/	′sd	la7 .	/home
--------	----	-------	----	------	-----	-------	-------

Option	Description
- n	Sets mount without writing to $/etc/mtab$.
- t	Indicates the type of file system to use.
- a	Mounts all filesystems mentioned in /etc/fstab.
- r	Mounts the file system read-only (equivalent to $-o$ ro).
- W	Mounts the file system read/write, by default (equivalent $-o$ rw).
-o opts	The opts argument is a comma-separated list (remount , ro ,).

🖍 Note

The mount command alone displays all mounted file systems. If the mount parameter is -o defaults, it is equivalent to -o rw, suid, dev, exec, auto, nouser, async and these parameters are independent of the file system. If you need to browse special mount options related to the file system, please read the "Mount options FS-TYPE" section in man 8 mount (FS-TYPE is replaced with the corresponding file system, such as ntfs, vfat, ufs, etc.)

umount command

The umount command is used to unmount logical drives.

```
umount [-option] [device] [directory]
```

Example:

[root]# umount /home
[root]# umount /dev/sda7

Option	Description
- n	Sets mounting removal without writing to $\ensuremath{/\text{etc/mtab}}$.
- r	Remounts as read-only if umount fails.
- f	Forces mounting removal.
- a	Removes mounts of all filesystems mentioned in \slashed{scalar} /fstab .

🖍 Note

When disassembling, you must not stay below the mounting point. Otherwise, the following error message is displayed: device is busy.

8.5 File naming convention

As in any system, it is important to respect the file naming rules to navigate the tree structure and file management.

- Files are coded on 255 characters;
- All ASCII characters can be used;
- Uppercase and lowercase letters are differentiated;
- Most files do not have a concept for file extension. In the GNU/Linux world, most file extensions are not required, except for a few (for example, .jpg, .mp4, .gif, etc.).

Groups of words separated by spaces must be enclosed in quotation marks:

[root]# mkdir "working dir"
✓ Note
While nothing is technically wrong with creating a file or directory with a space, it is generally a "best practice" to avoid this and replace any space with an underscore.

Note

The . at the beginning of the file name only hides it from a simple ls.

Examples of file extension agreements:

- .c : source file in C language;
- .h : C and Fortran header file;
- .o : object file in C language;
- .tar : data file archived with the tar utility;
- .cpio : data file archived with the cpio utility;
- .gz : data file compressed with the gzip utility;
- .tgz : data file archived with the tar utility and compressed with the gzip utility;
- .html : web page.

8.5.1 Details of a file name

```
[root]# ls -liah /usr/bin/passwd
266037 -rwsr-xr-x 1 root root 59K mars 22 2019 /usr/bin/passwd
1 2 3 4 5 6 7 8 9
```

Part	Description
1	Inode number
2	File type (1st character of the block of 10), "-" means this is an ordinary file.
3	Access rights (last 9 characters of the block of 10)
4	If this is a directory, this number represents how many subdirectories there are in that directory, including hidden ones. If this is a file, it indicates the number of hard links. When the number 1 is, there is only one hard link.
5	Name of the owner
6	Name of the group
7	Size (byte, kilo, mega)
8	Date of last update
9	Name of the file

In the GNU/Linux world, there are seven file types:

File types	Description
-	Represents an ordinary file. Including plain text files (ASCII); binary files (binary); data format files (data); various compressed files.
d	Represents a directory file.
b	Represents a block device file. It includes hard drives, USB drives, and so on.
С	Represents a character device file. Interface device of serial port, such as mouse, keyboard, etc.
S	Represents a socket file. It is a file specially used for network communication.
р	Represents a pipe file. It is a special file type. The main purpose is to solve the errors caused by multiple programs accessing a file simultaneously. FIFO is the abbreviation of first-in-first-out.
1	Represents soft link files, also called symbolic link files, are similar to shortcuts in Windows. Hard link file, also known as physical link file.

Supplementary description of the directory

Each directory has two hidden files: . and ... You need to use <code>ls -al</code> to view, for example:

```
# . Indicates that in the current directory, for example, you need to execute a
script in a directory, usually:
Shell > ./scripts
# .. represents the directory one level above the current directory, for
example:
Shell > cd /etc/
Shell > cd ..
Shell > pwd
/
# For an empty directory, its fourth part must be greater than or equal to 2.
Because there are "." and ".."
Shell > mkdir /tmp/t1
Shell > ls -ldi /tmp/t1
1179657 drwxr-xr-x 2 root root 4096 Nov 14 18:41 /tmp/t1
```

Special files

To communicate with peripherals (hard disks, printers, etc.), Linux uses interface files called special files (*device file* or *special file*). These files allow the peripherals to identify themselves.

These files are special because they do not contain data but specify the access mode to communicate with the device.

They are defined in two modes:

- block mode;
- character mode.

```
# Block device file
Shell > ls -l /dev/sda
brw----- 1 root root 8, 0 jan 1 1970 /dev/sda
# Character device file
Shell > ls -l /dev/tty0
crw----- 1 root root 8, 0 jan 1 1970 /dev/tty0
```

Communication files

These are the pipe (*pipes*) and the *socket* files.

- **Pipe files** pass information between processes by FIFO (*First In, First Out*). One process writes transient information to a *pipe* file, and another reads it. After reading, the information is no longer accessible.
- **Socket files** allow bidirectional inter-process communication (on local or remote systems). They use an *inode* of the file system.

Link files

These files allow the possibility of giving several logical names to the same physical file, creating a new access point to the file.

There are two types of link files:

- Soft link files, also called symbolic link files;
- Hard link files, also called physical link files.

Their main features are:

Link types	Description
Soft link file	This file is similar to a shortcut for Windows. It has permission of 0777 and points to the original file. When the original file is deleted, you can use ls -l to view the output information of the soft link file. In the output information, the file name of the soft link appears in red, and the pointed original file appears in red with a flashing prompt.
Hard link file	This file represents different mappings occupying the same <i>inode</i> number. They can be updated synchronously (including file content, modification time, owner, group affiliation, access time, etc.). Hard-linked files cannot span partitions and file systems and cannot be used in directories.

Specific examples are as follows:

```
# Permissions and the original file to which they point
Shell > ls -l /etc/rc.locol
lrwxrwxrwx 1 root root 13 Oct 25 15:41 /etc/rc.local -> rc.d/rc.local
# When deleting the original file. "-s" represents the soft link option
Shell > touch /root/Afile
```

```
Shell > ln -s /root/Afile /root/slink1
Shell > rm -rf /root/Afile
```

[root@Ansible ~]# ls -l /root/slink1
lrwxrwxrwx 1 root root 11 11月 14 18:05 /root/slink1 -> /root/Afile
[root@Ansible ~]#

```
Shell > cd /home/paul/
Shell > ls -li letter
666 -rwxr--r-- 1 root root ... letter
# The ln command does not add any options, indicating a hard link
Shell > ln /home/paul/letter /home/jack/read
# The essence of hard links is the file mapping of the same inode number in
different directories.
Shell > ls -li /home/*/*
666 -rwxr--r-- 2 root root ... letter
666 -rwxr--r-- 2 root root ... read
# If you use a hard link to a directory, you will be prompted:
Shell > ln /etc/ /root/etc_hardlink
ln: /etc: hard link not allowed for directory
```

8.6 File attributes

Linux is a multi-user operating system where the control of access to files is essential.

These controls are functions of:

- file access permissions ;
- users (ugo Users Groups Others).

8.6.1 Basic permissions of files and directories

File permissions	Description
r	Read. Allows reading a file (cat , less ,) and copying a file (cp ,).
W	Write. Allows modification of the file content (${\tt cat}$, >> , ${\tt vim}$,).
х	Execute. Considers the file as an e \mathbf{X} ecutable (binary or script).
-	No right

The description of **file permissions** is as follows:

The description of **directory permissions** is as follows:

Directory permissions	Description
r	Read. Allows reading the contents of a directory ($\ensuremath{\mbox{ls}}\xspace$ -R).
W	Write. Allows you to create, and delete files/directories in this directory, such as commands mkdir , rmdir , rm , touch , and so on.
x	Execute. Allows entry into directory (cd).
-	No right

i Info

For a directory's permissions, r and x usually appear at the same time. Moving or renaming a file depends on whether the directory where it is located has w permission, and so does deleting a file.

8.6.2 User type corresponding to basic permission

Description
Owner
Owner group
Others users

i Info

In some commands, you can use *a* (*all*) to represent *ugo*. For example: chmod a+x FileName is equivalent to chmod u+x,g+x,o+x FileName or chmod ugo+x FileName.

8.6.3 Attribute management

The display of rights is done with the command ls -1. It is the last 9 characters of the block of 10. More precisely 3 times 3 characters.

```
[root]# ls -l /tmp/myfile
-rwxrw-r-x 1 root sys ... /tmp/myfile
1 2 3 4 5
```

Part	Description
1	Owner (user) permissions, here rwx
2	Owner group permissions (group), here $\ensuremath{rw}\xspace$
3	Other users' permissions (others), here $\ r\mbox{-}x$
4	File owner
5	Group owner of the file

By default, the *owner* of a file is the one who created it. The *group* of the file is the group of the owner who created the file. The *others* are those not concerned by the previous cases.

The attributes are changed with the chmod command.

Only the administrator and the owner of a file can change the rights of a file.

chmod command

The chmod command allows you to change the access permissions to a file.

chmod [option] mode file

Option	Observation
- R	Recursively change the permissions of the directory and all files under the directory.

🛕 Warning

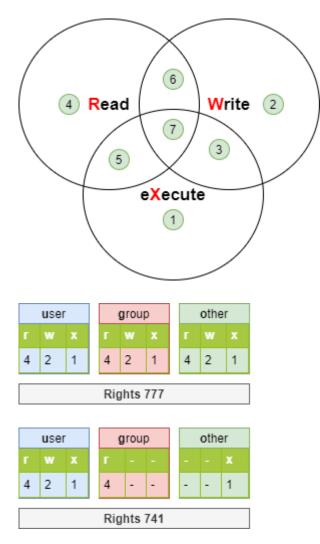
The rights of files and directories are not dissociated. For some operations, it will be necessary to know the rights of the directory containing the file. A write-protected file can be deleted by another user as long as the rights of the directory containing it allow this user to perform this operation.

The mode indication can be an octal representation (e.g. 744) or a symbolic representation ([ugoa][+=-][rwxst]).

OCTAL OR NUMBER REPRESENTATION

Number	Description
4	r
2	W
1	x
0	-

Add the three numbers together to get one user type permission. E.g. **755=rwxr-xr-x**.



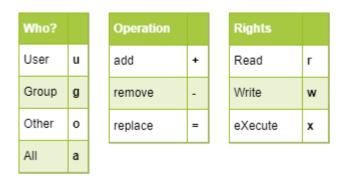
i Info

Sometimes you will see chmod 4755. The number 4 here refers to the special permission **set uid**. Special permissions will not be expanded here for the moment, just as a basic understanding.

```
[root]# ls -l /tmp/fil*
-rwxrwx--- 1 root root ... /tmp/file1
-rwx--x--- 1 root root ... /tmp/file2
-rwx--xr-- 1 root root ... /tmp/file3
[root]# chmod 741 /tmp/file1
[root]# chmod -R 744 /tmp/file2
[root]# ls -l /tmp/fic*
-rwxr----x 1 root root ... /tmp/file1
-rwxr--r-- 1 root root ... /tmp/file2
```

SYMBOLIC REPRESENTATION

This method can be considered as a "literal" association between a user type, an operator, and rights.



[root]# chmod -R u+rwx,g+wx,o-r /tmp/file1 [root]# chmod g=x,o-r /tmp/file2 [root]# chmod -R o=r /tmp/file3

8.7 Default rights and mask

When a file or directory is created, it already has permissions.

- For a directory: rwxr-xr-x or 755.
- For a file: rw-r-r- or 644.

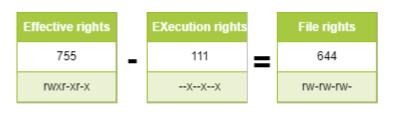
This behavior is defined by the **default mask**.

The principle is to remove the value defined by the mask at maximum rights without the execution right.

For a directory:



For a file, the execution rights are removed:



1 Info

The /etc/login.defs file defines the default UMASK, with a value of 022. This means the permission to create a file is 755 (rwxr-xr-x). However, for the sake of security, GNU/Linux does not have x permission for newly created files. This restriction applies to root(uid=0) and ordinary users(uid>=1000).

```
# root user
Shell > touch a.txt
Shell > ll
-rw-r--r-- 1 root root 0 Oct 8 13:00 a.txt
```

8.7.1 umask command

The umask command allows you to display and modify the mask.

```
umask [option] [mode]
```

Example:

```
$ umask 033
$ umask
0033
$ umask -S
u=rwx,g=r,o=r
$ touch umask_033
$ ls -la umask_033
-rw-r--r-- 1 rockstar rockstar 0 nov. 4 16:44 umask_033
```

\$ umask 025 \$ umask -S u=rwx,g=rx,o=w \$ touch umask_025 \$ ls -la umask_025 -rw-r---w- 1 rockstar rockstar 0 nov. 4 16:44 umask_025

Option	Description
- S	Symbolic display of file rights.

Warning

umask does not affect existing files. umask -s displays the file rights (without the execute right) of the files that will be created. So, it is not the display of the mask used to subtract the maximum value.

Note

In the above example, using commands to modify masks applies only to the currently connected session.

i Info

The umask command belongs to bash's built-in commands, so when you use man umask, all built-in commands will be displayed. If you only want to view the help of umask, you must use the help umask command.

To keep the value, you have to modify the following profile files

For all users:

- /etc/profile
- /etc/bashrc

For a particular user:

• ~/.bashrc

When the above file is written, it actually overrides the UMASK parameter of /etc/login.defs. If you want to improve the security of the operating system, you can set umask to **027** or **077**.

9. Process Management

In this chapter, you will learn how to work with processes.

Objectives: In this chapter, future Linux administrators will learn how to:

Recognize the PID and PPID of a process;
View and search for processes;
Manage processes.

🕫 process, linux

Knowledge: $\star \star$ Complexity: \star

Reading time: 20 minutes

9.1 Generalities

An operating system consists of processes. These processes are executed in a specific order and are related. There are two categories of processes, those focused on the user environment and those focused on the hardware environment.

When a program runs, the system will create a process by placing the program data and code in memory and creating a **runtime stack**. A process is an instance of a program with an associated processor environment (ordinal counter, registers, etc...) and memory environment.

Each process has:

- a PID: Process IDentifier, a unique process identifier
- a PPID: Parent Process IDentifier, unique identifier of parent process

By successive filiations, the init process is the father of all processes.

- A parent process always creates a process
- A parent process can have multiple child processes

There is a parent/child relationship between processes. A child process results from the parent calling the *fork()* primitive and duplicating its code to create a child. The *PID* of the child is returned to the parent process so that it can talk to it. Each child has its parent's identifier, the *PPID*.

The *PID* number represents the process at the time of execution. When the process finishes, the number is available again for another process. Running the same command several times will produce a different *PID* each time.

```
Note
Processes are not to be confused with threads. Each process has its memory context (resources and address space), while threads
```

9.2 Viewing processes

The ps command displays the status of running processes.

```
ps [-e] [-f] [-u login]
```

from the same process share this context.

Example:

ps -fu root

Option	Description
- e	Displays all processes.
-f	Displays full format list.
-u login	Displays the user's processes.

Some additional options:

Option	Description
- g	Displays the processes in the group.
-t tty	Displays the processes running from the terminal.
-p PID	Displays the process information.
- H	Displays the information in a tree structure.
-1	Displays in long format.
sort COL	Sort the result according to a column.
headers	Displays the header on each terminal page.
format "%a %b %c"	Customize the output display format.

Without an option specified, the ps command only displays processes running from the current terminal.

The result is displayed in the following columns:

# ps	-ef						
UID	PID	PPID	С	STIME	TTY	TIME	CMD
root	1	Θ	0	Jan01	?	00:00/03	/sbin/init

Column	Description
UID	Owner user.
PID	Process identifier.
PPID	Parent process identifier.
C	Priority of the process.
STIME	Date and time of execution.
TTY	Execution terminal.
TIME	Processing duration.
CMD	Command executed.

The behavior of the control can be fully customized:

```
# ps -e --format "%P %p %c %n" --sort ppid --headers
PPID PID COMMAND NI
0 1 systemd 0
0 2 kthreadd 0
1 516 systemd-journal 0
1 538 systemd-udevd 0
```

1	598	lvmetad	Θ
1	643	auditd	-4
1	668	rtkit-daemon	1
1	670	sssd	Θ

9.3 Types of processes

The user process:

- $\ensuremath{\bullet}$ is started from a terminal associated with a user
- accesses resources via requests or daemons

The system process (*daemon*):

- is started by the system
- is not associated with any terminal and is owned by a system user (often root)
- is loaded at boot time, resides in memory, and is waiting for a call
- \bullet is usually identified by the letter $\ensuremath{ \ensuremath{ d} }$ associated with the process name

System processes are therefore called daemons (*D****isk** And Execution MON**itor*).

9.4 Permissions and rights

The user's credentials are passed to the created process when a command is executed.

By default, the process's actual UID and GID (of the process) are identical to the **actual** UID and GID (the UID and GID of the user who executed the command).

When a SUID (and/or SGID) is set on a command, the actual UID (and/or GID) becomes that of the owner (and/or owner group) of the command and no longer that of the user or user group that issued the command. Effective and real **UIDs** are therefore **different**.

Each time a file is accessed, the system checks the rights of the process according to its effective identifiers.

9.5 Process management

A process cannot be run indefinitely, as this would be to the detriment of other running processes and would prevent multitasking.

Therefore, the total processing time available is divided into small ranges, and each process (with a priority) accesses the processor sequentially. The process will take several states during its life among the states:

- ready: waiting for the availability of the process
- in execution: accesses the processor
- suspended: waiting for an I/O (input/output)
- stopped: waiting for a signal from another process
- zombie: request for destruction
- dead: the parent process ends the child process

The end-of-process sequencing is as follows:

- 1. Closing of the open files
- 2. Release of the used memory
- 3. Sending a signal to the parent and child processes

When a parent process dies, their children are said to be orphans. They are then adopted by the init process, which will destroy them.

9.5.1 The priority of a process

GNU/Linux belongs to the family of time-sharing operating systems. Processors work in a time-sharing manner, and each process takes up some processor time. Processes are classified by priority:

- Real-time process: the process with priority of **0-99** is scheduled by real-time scheduling algorithm.
- Ordinary processes: processes with dynamic priorities of **100-139** are scheduled using a fully fair scheduling algorithm.
- Nice value: a parameter used to adjust the priority of an ordinary process. The range is **-20-19**.

The default priority of a process is $\mathbf{0}$.

9.5.2 Modes of operation

Processes can run in two ways:

- **synchronous**: the user loses access to the shell during command execution. The command prompt reappears at the end of the process execution.
- **asynchronous**: the process is processed in the background. The command prompt is displayed again immediately.

The constraints of the asynchronous mode:

- the command or script must not wait for keyboard input
- the command or script must not return any result on the screen
- quitting the shell ends the process

9.6 Process management controls

9.6.1 kill command

The kill command sends a stop signal to a process.

kill [-signal] PID

Example:

kill -9 1664

Code	Signal	Description
2	SIGINT	Immediate termination of the process
9	SIGKILL	Interrupts the process $(\underline{(ctrl)} + \underline{d})$
15	SIGTERM	Clean termination of the process
18	SIGCONT	Resumes the process. Processes that use the SIGSTOP signal can use it to continue running
19	SIGSTOP	Suspends the process (Stops process). The effect of this signal is equivalent to $(ctrl) + z$

Signals are the means of communication between processes. The kill command sends a signal to a process.

Tip The complete list of signals taken into account by the kill command is available by typing the command: \$ man 7 signal

9.6.2 nohup command

nohup allows the launching of a process independently of a connection.

nohup command

Example:

nohup myprogram.sh 0</dev/null &</pre>

nohup ignores the SIGHUP signal sent when a user logs out.

Note

nohup handles standard output and error but not standard input, hence the redirection of this input to /dev/null.

9.6.3 [Ctrl] + [z]

The synchronous process is temporarily suspended by pressing the $\colored{ctrl} + \colored{z}$ keys simultaneously. Access to the prompt is restored after displaying the number of the process that has just been suspended.

9.6.4 & instruction

The & statement executes the command asynchronously (the command is then called *job*) and displays the number of *job*. Access to the prompt is then returned.

Example:

```
$ time ls -lR / > list.ls 2> /dev/null &
[1] 15430
$
```

The *job* number is obtained during background processing and is displayed in square brackets, followed by the PID number.

9.6.5 fg and bg commands

The fg command puts the process in the foreground:

```
$ time ls -lR / > list.ls 2>/dev/null &
$ fg 1
time ls -lR / > list.ls 2/dev/null
```

while the command bg places it in the background:

```
[CTRL]+[Z]
^Z
[1]+ Stopped
$ bg 1
[1] 15430
$
```

Whether it was put in the background when it was created with the & argument or later with the $^{ctrl} + z$ keys, a process can be brought back to the foreground with the fg command and its job number.

9.6.6 jobs command

The jobs command displays the list of processes running in the background and specifies their job number.

Example:

```
$ jobs
[1]- Running sleep 1000
[2]+ Running find / > arbo.txt
```

The columns represent:

- 1. job number
- 2. the order that the processes run:
- 3. a + : The process selected by default for the fg and bg commands when no job number is specified
- 4. a : This process is the next process to take the +
- 5. Running (running process) or Stopped (suspended process)
- 6. the command

9.6.7 nice and renice commands

The command nice allows the execution of a command by specifying its priority.

nice priority command

Usage example:

```
nice --adjustment=-5 find / -name "file"
nice -n -5 find / -name "file"
nice --5 find / -name "file"
nice -n 5 find / -name "file"
nice find / -name "file"
```

Unlike root, a standard user can only reduce the priority of a process and only values between 0 and 19 will be accepted.

As shown in the example above, the first three commands indicate setting the Nice value to "-5", while the second command is our recommended usage. The fourth command indicates setting the Nice value to "5". For the fifth command, not typing any options means that the Nice value is set to "10".

b Tip
"Nice" is the abbreviation for "niceness".
Directly typing the nice command will return the Nice value of the current shell.
You can lift the Nice value limit for each user or group by modifying the /etc/security/limits.conf file.

The renice command allows you to change the priority of a running process.

```
renice priority [-g GID] [-p PID] [-u UID]
```

Example:

renice -n 15 -p 1664

Option	Description				
- g	GID of the process owner group.				
- p	PID of the process.				
- U	UID of the process owner.				

The renice command acts on existing processes. Therefore, it is possible to change the priority of a specific process and several processes belonging to a user or a group.



To adapt to different distributions, you should try to use command forms such as nice -n 5 or renice -n 6 as much as possible.

9.6.8 top command

The top command displays the processes and their resource consumption.

Descrip	otion			
Process	identifier			
Owner u				
Process	priority.			
Nice val				
Processo	or load.			
Memory				
Processo	or usage t	time.		
Commai	nd execut	ed.		
	20 0 Descrip Process Owner u Process Nice val Process Memory Process	20 0 15 20 Description 10 20 Process identifier 10 20 Process priority. 20 20 Processor load. 20 20 Memory load. 20 20 Processor usage to	20 0 15 5.5 Description Process identifier. Owner user. Process priority. Nice value. Processor load.	Process identifier. Owner user. Process priority. Nice value. Processor load. Memory load. Processor usage time.

The top command allows control of the processes in real-time and in interactive mode.

9.6.9 pgrep and pkill commands

The pgrep command searches the running processes for a process name and displays the *PID* matching the selection criteria on the standard output.

The pkill command will send each process the specified signal (by default *SIGTERM*).

```
pgrep process
pkill [option] [-signal] process
```

Examples:

• Get the process number from sshd :

pgrep -u root sshd

• Kill all tomcat processes:

pkill tomcat

🖍 Note

Before you kill a process, it's best to know exactly what it is for; otherwise, it can lead to system crashes or other unpredictable problems.

In addition to sending signals to the relevant processes, the pkill command can also end the user's connection session according to the terminal number, such as:

pkill -t pts/1

9.6.10 killall command

This command's function is roughly the same as that of the pkill command. The usage is — killall [option] [-s SIGNAL | -SIGNAL] NAME. The default signal is *SIGTERM*.

Options	Description
-1	lists all known signal names
-i	asks for confirmation before killing
- I	case insensitive process name match

Example:

killall tomcat

9.6.11 pstree command

This command displays the progress in a tree style, and its usage is - pstree [option].

Option	Description
- p	Displays the PID of the process
- n	sorts output by PID
- h	highlights the current process and its ancestors
- U	shows uid transitions



9.6.12 Orphan process and zombie process

orphan process: When a parent process dies, their children are said to be orphans. The init process adopts these special state processes, and status collection is completed until they are destroyed. Conceptually speaking, the orphanage process does not pose any harm.

zombie process: After a child process completes its work and is terminated, its parent process needs to call the signal processing function wait() or waitpid() to obtain the termination status of the child process. If the parent process does not do so, although the child process has already exited, it still retains some exit status information in the system process table. Because the parent process cannot obtain the status information of the child process, these processes will continue to occupy resources in the process table. We refer to processes in this state as zombies.

Hazard:

- They are occupying system resources and causing a decrease in machine performance.
- Unable to generate new child processes.

How can we check for any zombie processes in the current system?

ps -lef | awk '{print \$2}' | grep Z

These characters may appear in this column:

- **D** uninterruptible sleep (usually IO)
- I Idle kernel thread
- ${\bf R}$ running or runnable (on run queue)
- ${\bf S}$ interruptible sleep (waiting for an event to complete)
- T stopped by job control signal
- \boldsymbol{t} stopped by debugger during the tracing
- W paging (not valid since the 2.6.xx kernel)
- ${\bf X}$ dead (should never be seen)
- \mathbf{Z} defunct ("zombie") process, terminated but not reaped by its parent

10. Backup and Restore

In this chapter, you will learn how to back up and restore your data using Linux.

Objectives: In this chapter, future Linux administrators will learn how to:

✓ use the tar and cpio command to make a backup;

Check their backups and restore data;

 \checkmark compress or decompress their backups.

🕅 backup, restore, compression

Knowledge: $\bigstar \bigstar \bigstar$ Complexity: $\bigstar \bigstar$

Reading time: 40 minutes

Note

Throughout this chapter, the command structures use "device" to specify both a target location for backup and the source location when restoring. The device can be either external media or a local file. You should get a feel for this as the chapter unfolds, but you can always refer back to this note for clarification if you need to.

The backup will answer the need to conserve and restore data effectively.

The backup allows you to protect yourself from the following:

- Destruction: voluntary or involuntary. Human or technical. Virus, ...
- Deletion: voluntary or involuntary. Human or technical. Virus, ...
- Integrity: data becomes unusable.

No system is infallible, and no human is infallible, so to avoid losing data, it must be backed up to restore it after a problem.

The backup media should be kept in another room (or building) than the server so that a disaster does not destroy the server and the backups.

In addition, the administrator must regularly check that the media are still readable.

10.1 Generalities

There are two principles: the **backup** and the **archive**.

- The archive destroys the information source after the operation.
- The backup preserves the source of information after the operation.

These operations consist of saving information in a file, on a peripheral, or a supported media (tapes, disks, and so on).

10.1.1 The process

Backups require a lot of discipline and rigor from the system administrator. System administrators need to consider the following issues before performing backup operations:

- What is the appropriate medium?
- What should be backed up?
- How many copies?
- How long will the backup take?
- Method?
- How often?
- Automatic or manual?
- Where to store it?
- How long will it be kept?
- Is there a cost issue to consider?

In addition to these issues, system administrators should also consider factors such as performance, data importance, bandwidth consumption, and maintenance complexity based on actual situations.

10.1.2 Backup methods

- **Full backup**: Refers to a one-time copy of all files, folders, or data in the hard disk or database.
- **Incremental backup**: Refers to the backup of the data updated after the last Full backup or Incremental backup.
- **Differential backup**: Refers to the backup of the changed files after the Full backup.
- Selective backup (Partial backup): Refers to backing up a part of the system.
- **Cold backup**: Refers to the backup when the system is in a shutdown or maintenance state. The backed-up data is precisely the same as the data in the system during this period.
- **Hot backup**: Refers to the backup when the system is operating normally. As the data in the system is updated at any time, the backed-up data has a certain lag relative to the system's real data.
- **Remote backup**: Refers to backing up data in another geographic location to avoid data loss and service interruption caused by fire, natural disasters, theft, and more.

10.1.3 Frequency of backups

- **Periodic**: Backup within a specific period before a major system update (usually during off-peak hours)
- cycle: Backup in units of days, weeks, months, etc

🜢 Tip

Before a system change, it can be useful to make a backup. However, there is no point in backing up data every day that only changes every month.

10.1.4 Recover methods

Depending on the utilities available, performing several types of recovery will be possible.

In some relational database management systems, the corresponding operations of "recover" (sometimes "recovery" is used in the documentation) and "restore" are different. For further information, consult the official documentation. This basic document will not go into too much detail regarding this part of RDBMS.

- **Full recover**: Data recovery based on Full backup or "Full backup + Incremental backup" or "Full backup + Differential backup".
- Selective recover: Data recovery based on Selective backup (Partial backup).

We do not recommend directly deleting directories or files in the currently active operating system before performing a recovery operation (unless you know what will happen after deletion). If you don't know what will happen, you can perform a 'snapshot' operation on the current operating system.

• **Tip** For security reasons, storing the restored directory or file in the /tmp directory before performing the recovery operation is recommended to avoid situations where old files (old directory) overwrite new files (new directory).

10.1.5 The tools and related technologies

There are many utilities to make backups.

- editor tools;
- graphical tools;
- command line tools: tar, cpio, pax, dd, dump,...

The commands we will use here are tar and cpio. If you want to learn about the dump tool, please refer to this document.

- tar:
- easy to use;
- allows adding files to an existing backup.
- cpio:
- retains owners;
- retains groups, dates and rights;
- skips damaged files;
- can be used for the entire file system.

/	Note				

These commands save in a proprietary and standardized format. $% \label{eq:commands}$

Replication: A backup technology that copies a set of data from one data source to another or multiple data sources, mainly divided into **Synchronous Replication** and **Asynchronous Replication**. This is an advanced backup part for novice system administrators, so this basic document will not elaborate on these contents.

10.1.6 Naming convention

Using a naming convention allows one to quickly target a backup file's contents and thus avoid hazardous restorations.

- name of the directory;
- utility used;
- options used;
- date.

🗴 Tip

The name of the backup must be explicit.

Note

In the Linux world, most files do not have the extension concept except for a few exceptions in GUI environments (such as .jpg, .mp4, .gif). In other words, most file extensions are not required. The reason for artificially adding suffixes is to facilitate recognition by human users. If the systems administrator sees a .tar.gz or .tgz file extension, for instance, then he knows how to deal with the file.

10.1.7 Properties of the backup file

A single backup file can include the following properties:

- file name (including manually added suffixes);
- backup the atime, ctime, mtime, btime (crtime) of the file itself;
- file size of the backup file itself;
- the properties or characteristics of files or directories in the backup file will be partially preserved. For example, mtime for files or directories will be retained, but inode number will not be retained.

10.1.8 Storage methods

There are two different storage methods:

- Internal: Store backup files on the current working disk.
- External: Store backup files on external devices. External devices can be USB drives, CDs, disks, servers, or NAS, and more.

10.2 Tape ArchiveR - tar

The tar command allows saving on several successive media (multi-volume options).

It is possible to extract all or part of a backup.

tar implicitly backs up in relative mode even if the path of the information to be backed up is mentioned in absolute mode. However, backups and restores in absolute mode are possible. If you want to see a separate example of the usage of tar, please refer to this document.

10.2.1 Restoration guidelines

The right questions to ask are:

- what: partial or complete;
- where: the place where the data will be restored;
- how: absolute or relative.

🛕 Warning

Before a restoration, it is important to consider and determine the most appropriate method to avoid mistakes.

Restorations are usually performed after a problem has occurred that needs to be resolved quickly. A poor restoration can, in some cases, make the situation worse.

10.2.2 Backing up with tar

The default utility for creating backups on UNIX systems is the tar command. These backups can be compressed by bzip2, xz, lzip, lzma, lzop, gzip, compress or zstd.

tar allows you to extract a single file or a directory from a backup, view its contents, or validate its integrity.

Estimate the size of a backup

The following command estimates the size in bytes of a possible *tar* file:

```
$ tar cf - /directory/to/backup/ | wc -c
20480
$ tar czf - /directory/to/backup/ | wc -c
508
$ tar cjf - /directory/to/backup/ | wc -c
428
```

🛕 Warning

Beware, the presence of "-" in the command line disturbs zsh. Switch to bash!

Naming convention for a tar backup

Here is an example of a naming convention for a tar backup, knowing that the date will be added to the name.

keys	Files	Suffix	Functionality
cvf	home	home.tar	/home in relative mode, uncompressed form
cvfP	/etc	etc.A.tar	/etc in absolute mode, no compression
cvfz	usr	usr.tar.gz	/usr in relative mode, gzip compression
cvfj	usr	usr.tar.bz2	/usr in relative mode, <i>bzip2</i> compression
cvfPz	/home	home.A.tar.gz	/home in absolute mode, gzip compression
cvfPj	/home	home.A.tar.bz2	/home in absolute mode, <i>bzip2</i> compression

Create a backup

CREATE A BACKUP IN RELATIVE MODE

Creating a non-compressed backup in relative mode is done with the cvf keys:

```
tar c[vf] [device] [file(s)]
```

Example:

[root]# tar cvf /backups/home.133.tar /home/

Key	Description
С	Creates a backup.
V	Displays the name of the processed files.
f	Allows you to specify the name of the backup (medium).

💧 Tip

The hyphen (-) in front of the tar keys is optional!

CREATE A BACKUP IN ABSOLUTE MODE

Creating a non-compressed backup explicitly in absolute mode is done with the cvfP keys:

```
tar c[vf]P [device] [file(s)]
```

Example:

```
[root]# tar cvfP /backups/home.133.P.tar /home/
```

Кеу	Description
Ρ	Creates a backup in absolute mode.

🛕 Warning

With the P key, the path of the files to be backed up must be entered as **absolute**. If the two conditions (P key and **absolute** path) are not indicated, the backup is in relative mode.

CREATING A COMPRESSED BACKUP WITH gzip

Creating a compressed backup with gzip is done with the cvfz keys:

tar cvzf backup.tar.gz dirname/

Key	Description
Z	Compresses the backup in gzip.

/ Note

The .tgz extension is equivalent to .tar.gz.

```
🖍 Note
```

Keeping the cvf (tvf or xvf) keys unchanged for all backup operations and simply adding the compression key to the end of the keys makes the command easier to understand (such as: cvfz or cvfj, and others).

CREATING A COMPRESSED BACKUP WITH bzip2

Creating a compressed backup with bzip2 is done with the keys cvfj:

```
      Key
      Description

      j
      Compresses the backup in bzip2.

      Note
      The .tbz and .tb2 extensions are equivalent to .tar.bz2 extensions.
```

COMPARISON OF COMPRESSION EFFICIENCY

Compression, and consequently decompression, will impact resource consumption (time and CPU usage).

Here is a ranking of the compression of a set of text files from least to most efficient:

- compress (.tar.Z)
- gzip(.tar.gz)
- bzip2 (.tar.bz2)
- lzip(.tar.lz)
- xz(.tar.xz)

Add a file or directory to an existing backup

It is possible to add one or more items to an existing backup.

```
tar {r|A}[key(s)] [device] [file(s)]
```

To add /etc/passwd to the backup /backups/home.133.tar:

[root]# tar rvf /backups/home.133.tar /etc/passwd

Adding a directory is similar. Here add dirtoadd to backup_name.tar:

tar rvf backup_name.tar dirtoadd

Кеу	Description
r	Appends the files or directories to the end of the archive.
А	Appends all files in one archive to the end of another archive.

🖍 Note

It is not possible to add files or folders to a compressed backup.

```
$ tar rvfz backup.tgz filetoadd
tar: Cannot update compressed archives
Try `tar --help' or `tar --usage' for more information.
```

🖍 Note

If the backup was performed in relative mode, add files in relative mode. If the backup was done in absolute mode, add files in absolute mode.

Mixing modes can cause problems when restoring.

List the contents of a backup

Viewing the contents of a backup without extracting it is possible.

```
tar t[key(s)] [device]
```

Key	Description
t	Displays the content of a backup (compressed or not).

Examples:

```
tar tvf backup.tar
tar tvfz backup.tar.gz
tar tvfj backup.tar.bz2
```

When the number of files in the backup increases, you can use pipe characters (|) and some commands (less, more, most, and others) to achieve the effect of paging viewing:

tar tvf backup.tar | less

🜢 Tip

To list or retrieve the contents of a backup, it is not necessary to mention the compression algorithm used when the backup was created. That is, a tar tvf is equivalent to tar tvfj, to read the contents. The compression type or algorithm **must** only be selected when creating a compressed backup.

🜢 Tip

You should always check and view the backup file's contents before performing a restore operation.

Check the integrity of a backup

The integrity of a backup can be tested with the w key at the time of its creation:

```
tar cvfW file_name.tar dir/
```

The integrity of a backup can be tested with the key d after its creation:

```
tar vfd file_name.tar dir/
```

🜢 Tip

By adding a second v to the previous key, you will get the list of archived files as well as the differences between the archived files and those present in the file system.

```
$ tar vvfd /tmp/quodlibet.tar .quodlibet/
drwxr-x--- rockstar/rockstar 0 2021-05-21 00:11 .quodlibet/
-rw-r--r-- rockstar/rockstar 0 2021-05-19 00:59 .quodlibet/queue
[...]
-rw------ rockstar/rockstar 3323 2021-05-21 00:11 .quodlibet/config
.quodlibet/config: Mod time differs
.quodlibet/config: Size differs
[...]
```

The w key is also used to compare the content of an archive against the filesystem:

```
$ tar tvfW file_name.tar
Verify 1/file1
1/file1: Mod time differs
1/file1: Size differs
Verify 1/file2
Verify 1/file3
```

You cannot verify the compressed archive with the w key. Instead, you must use the d key.

```
tar dfz file_name.tgz
tar dfj file_name.tar.bz2
```

Extract (untar) a backup

Extract (*untar*) a *.tar backup is done with the xvf keys:

Extract the etc/exports file from the /savings/etc.133.tar backup into the etc directory of the current directory:

tar xvf /backups/etc.133.tar etc/exports

Extract all files from the compressed backup /backups/home.133.tar.bz2 into the current directory:

[root]# tar xvfj /backups/home.133.tar.bz2

Extract all files from the backup /backups/etc.133.P.tar to their original directory:

```
tar xvfP /backups/etc.133.P.tar
```

A Warning		
5	easons, you should use caution when extracting backup files saved in a efore performing extraction operations, you should always check the co ute mode).	
Key	Description	
x	Extracts files from backups (whether compressed or not)	

Extracting a *tar-gzipped* (*.tar.gz) backup is done with the xvfz keys:

```
tar xvfz backup.tar.gz
```

Extracting a *tar-bzipped* (*.tar.bz2) backup is done with the xvfj keys:

tar xvfj backup.tar.bz2

🜢 Tip

To extract or list the contents of a backup, it is not necessary to mention the compression algorithm used to create the backup. That is, a tar xvf is equivalent to tar xvfj, to extract the contents, and a tar tvf is equivalent to tar tvfj, to list.

🔒 Warning

To restore the files in their original directory (key P of a tar xvf), you must have generated the backup with the absolute path. That is, with the P key of a tar cvf.

EXTRACT ONLY A FILE FROM A TAR BACKUP

To extract a specific file from a *tar* backup, specify the name of that file at the end of the tar xvf command.

tar xvf backup.tar /path/to/file

The previous command extracts only the /path/to/file file from the backup.tar backup. This file will be restored to the /path/to/ directory created, or already present, in the active directory.

```
tar xvfz backup.tar.gz /path/to/file
tar xvfj backup.tar.bz2 /path/to/file
```

EXTRACT A FOLDER FROM A BACKUP TAR

To extract only one directory (including its subdirectories and files) from a backup, specify the directory name at the end of the tar xvf command.

```
tar xvf backup.tar /path/to/dir/
```

To extract multiple directories, specify each of the names one after the other:

```
tar xvf backup.tar /path/to/dir1/ /path/to/dir2/
tar xvfz backup.tar.gz /path/to/dir1/ /path/to/dir2/
tar xvfj backup.tar.bz2 /path/to/dir1/ /path/to/dir2/
```

EXTRACT A GROUP OF FILES FROM A TAR BACKUP USING WILDCARD

Specify a wildcard to extract the files matching the specified selection pattern.

For example, to extract all files with the extension .conf :

tar xvf backup.tar --wildcards '*.conf'

keys:

• --wildcards *.conf corresponds to files with the extension .conf.

b Expanded Knowledge

Although wildcard characters and regular expressions usually have the same symbol or style, the objects they match are completely different, so people often confuse them.

wildcard (wildcard character): used to match file or directory names. regular expression: used to match the content of a file.

You can see an introduction with extra detail in this document.

10.3 CoPy Input Output - cpio

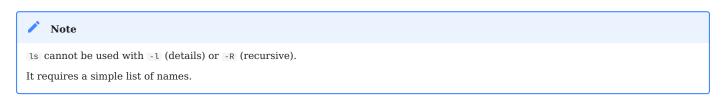
The cpio command allows saving on several successive media without specifying any options.

It is possible to extract all or part of a backup.

Unlike the tar command, there is no option to backup and compress simultaneously. So, it is done in two steps: backup and compression.

cpio has three operating modes, each corresponding to a different function:

- copy-out mode Creates a backup (archive). Enable this mode through the -o or
 -create options. In this mode, you must generate a list of files with a specific command (find, ls, or cat) and pass it to cpio.
- 2. find : browses a tree, recursive or not;
- 3. ls: lists a directory, recursive or not;
- 4. cat : reads a file containing the trees or files to be saved.



- 5. **copy-in mode** extracts files from an archive. You can enable this mode through the -i option.
- 6. **copy-pass mode** copies files from one directory to another. You can enable this mode through the -p or --pass-through options.

Like the tar command, users must consider how the file list is saved (**absolute path** or **relative path**) when creating an archive.

Secondary function:

- 1. -t Prints a table of input contents.
- 2. A Appends to an existing archive. It only works in copy-in mode.

```
Note
Some options of cpio need to be combined with the correct operating mode to work correctly. See man 1 cpio
```

10.3.1 copy-out mode

Syntax of the cpio command:

```
[files command |] cpio {-o| --create} [-options] [< file-list] [> device]
```

Example:

With a redirection of the output of cpio:

find /etc | cpio -ov > /backups/etc.cpio

Using the name of a backup media:

find /etc | cpio -ovF /backups/etc.cpio

The result of the find command is sent as input to the cpio command via a *pipe* (character ||, || Left Shift + ||).

Here, the find /etc command returns a list of files corresponding to the contents of the /etc directory (recursively) to the cpio command, which performs the backup.

Do not forget the > sign when saving or the F save_name_cpio.

Options	Description
- 0	Creates a backup through <i>cp-out</i> mode.
- V	Displays the name of the processed files.
- F	Backup to specific media, which can replace standard input ("<") and standard output (">") in the cpio command

Backup to a media:

find /etc | cpio -ov > /dev/rmt0

The media can be of several types:

- tape drive: /dev/rmt0;
- a partition: /dev/sda5 , /dev/hda5 , etc.

Relative and absolute paths of the file list

```
cd /
find etc | cpio -o > /backups/etc.cpio
```

find /etc | cpio -o > /backups/etc.A.cpio

🛕 Warning

If the path specified in the find command is **absolute**, the backup will be performed in **absolute**.

If the path indicated in the find command is **relative**, the backup will be done in **relative**.

Append files to existing backups

[files command |] cpio {-o| --create} -A [-options] [< fic-list] {F| > device}

Example:

```
find /etc/shadow | cpio -o -AF SystemFiles.A.cpio
```

Adding files is only possible on direct access media.

Option	Description
- A	Appends one or more files to an existing backup.
- F	Designates the backup to be modified.

Compressing a backup

• Save then compress

```
$ find /etc | cpio -o > etc.A.cpio
$ gzip /backups/etc.A.cpio
$ ls /backups/etc.A.cpio*
/backups/etc.A.cpio.gz
```

• Save and compress

find /etc | cpio -o | gzip > /backups/etc.A.cpio.gz

Unlike the tar command, there is no option to save and compress simultaneously. So, it is done in two steps: saving and compressing.

The syntax of the first method is easier to understand and remember because it is done in two steps.

For the first method, the backup file is automatically renamed by the gzip utility, which adds .gz to the end of the file name. Similarly, the bzip2 utility automatically adds .bz2.

10.3.2 Read the contents of a backup

Syntax of the cpio command to read the contents of a *cpio* backup:

```
cpio -t [-options] [< fic-list]</pre>
```

Example:

cpio -tv < /backups/etc.152.cpio | less</pre>

Options	Description
- t	Reads a backup.
- V	Displays file attributes.

After making a backup, you need to read its contents to ensure there are no errors.

In the same way, before performing a restore, you must read the contents of the backup that will be used.

10.3.3 copy-in mode

Syntax of the cpio command to restore a backup:

cpio {-i| --extract} [-E file] [-options] [< device]</pre>

Example:

cpio -iv < /backups/etc.152.cpio | less</pre>

Options	Description
-1	Restores a complete backup.
-E file	Restores only the files whose name is contained in file.
make-directories or -d	Rebuilds the missing tree structure.
- U	Replaces all files even if they exist.
no-absolute-filenames	Allows to restore a backup made in absolute mode in a relative way.

🔺 Warning

By default, at the time of restoration, files on the disk whose last modification date is more recent or equal to the date of the backup are not restored (to avoid overwriting recent information with older information).

On the other hand, the u option allows you to restore older versions of the files.

Examples:

• Absolute restoration of an absolute backup

cpio -ivF home.A.cpio

Absolute restoration on an existing tree structure

The u option allows you to overwrite existing files at the location where the restore takes place.

cpio -iuvF home.A.cpio

• Restore an absolute backup in relative mode

The long option no-absolute-filenames allows a restoration in relative mode. Indeed, the / at the beginning of the path will be removed.

cpio --no-absolute-filenames -divuF home.A.cpio

🗴 Tip

The creation of directories is perhaps necessary, hence the use of the $\,{\rm d}\,$ option

• Restore a relative backup

cpio -iv < etc.cpio

• Absolute restoration of a file or directory

Restoring a particular file or directory requires the creation of a list file that must then be deleted.

```
echo "/etc/passwd" > tmp
cpio -iuE tmp -F etc.A.cpio
rm -f tmp
```

10.4 Compression - decompression utilities

Using compression at the time of a backup can have a number of drawbacks:

• Lengthens the backup time as well as the restore time.

It is, therefore, better to make a backup and compress it than to compress it during the backup.

• It makes it impossible to add files to the backup.

10.4.1 Compressing with gzip

The gzip command compresses data.

Syntax of the gzip command:

```
gzip [options] [file ...]
```

Example:

Note

\$ gzip usr.tar
\$ ls
usr.tar.gz

The file receives the extension .gz.

It keeps the same rights and the same last access and modification dates.

10.4.2 Compressing with bzip2

The bzip2 command also compresses data.

Syntax of the bzip2 command:

```
bzip2 [options] [file ...]
```

Example:

```
$ bzip2 usr.cpio
$ ls
usr.cpio.bz2
```

The file name is given the extension .bz2.

Compression by bzip2 is better than compression by gzip, but executing it takes longer.

10.4.3 Decompressing with gunzip

The gunzip command decompresses compressed data.

Syntax of the gunzip command:

```
gunzip [options] [file ...]
```

Example:

```
$ gunzip usr.tar.gz
$ ls
usr.tar
```

The file name is truncated by gunzip and the extension .gz is removed.

gunzip also decompresses files with the following extensions:

- .z;
- -Z;
- _z;
- -gz;

10.4.4 Decompressing with bunzip2

The bunzip2 command decompresses compressed data.

Syntax of the bzip2 command:

bzip2 [options] [file ...]

Example:

```
$ bunzip2 usr.cpio.bz2
$ ls
usr.cpio
```

The file name is truncated by bunzip2, and the extension .bz2 is removed.

bunzip2 also decompresses the file with the following extensions:

- -bz;
- .tbz2;
- tbz.

11. System Startup

In this chapter, you will learn how the system starts.

Objectives: In this chapter, future Linux administrators will learn:

✓ The different stages of the booting process;

- ✓ How Rocky Linux supports this boot by using GRUB2 and systemd;
- ✓ How to protect GRUB2 from an attack;
- \checkmark How to manage the services;
- \checkmark How to access logs from journald.

🕅 users .

Knowledge: $\star \star$ Complexity: $\star \star \star$

Reading time: 20 minutes

11.1 The boot process

It is essential to understand the boot process of Linux to solve problems that might occur.

The boot process includes:

11.1.1 The BIOS startup

The **BIOS** (Basic Input/Output System) performs the **POST** (power on self-test) to detect, test, and initialize the system hardware components.

It then loads the **MBR** (Master Boot Record).

```
11.1.2 The Master boot record (MBR)
```

The Master Boot Record is the first 512 bytes of the boot disk. The MBR discovers the boot device, loads the bootloader **GRUB2** into memory, and transfers control to it.

The next 64 bytes contain the partition table of the disk.

11.1.3 The GRUB2 bootloader

The Rocky 8 distribution's default bootloader is **GRUB2** (GRand Unified Bootloader). GRUB2 replaces the old GRUB bootloader (also called GRUB legacy).

You can locate the GRUB2 configuration file under /boot/grub2/grub.cfg, but you should not edit this file directly.

You can find the GRUB2 menu configuration settings under /etc/default/grub. The grub2-mkconfig command uses these to generate the grub.cfg file.

```
# cat /etc/default/grub
GRUB_TIMEOUT=5
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=rhel/swap crashkernel=auto rd.lvm.lv=rhel/root
rhgb quiet net.ifnames=0"
GRUB_DISABLE_RECOVERY="true"
```

If you change one or more of these parameters, you must run the grub2-mkconfig command to regenerate the /boot/grub2/grub.cfg file.

[root] # grub2-mkconfig -o /boot/grub2/grub.cfg

- GRUB2 looks for the compressed kernel image (the vmlinuz file) in the /boot directory.
- GRUB2 loads the kernel image into memory and extracts the contents of the initramfs image file into a temporary folder in memory using the tmpfs file system.

11.1.4 The kernel

The kernel starts the systemd process with PID 1.

```
root 1 0 0 02:10 ? 00:00:02 /usr/lib/systemd/systemd --
switched-root --system --deserialize 23
```

11.1.5 systemd

systemd is the parent of all system processes. It reads the target of the /etc/ systemd/system/default.target link (e.g., /usr/lib/systemd/system/multi-user.target) to determine the default target of the system. The file defines the services to start.

systemd then places the system in the target-defined state by performing the following initialization tasks:

- 1. Set the machine name
- 2. Initialize the network
- 3. Initialize SELinux
- 4. Display the welcome banner
- 5. Initialize the hardware based on the arguments given to the kernel at boot time
- 6. Mount the file systems, including virtual file systems like /proc
- 7. Clean up directories in /var
- 8. Start the virtual memory (swap)

11.2 Protecting the GRUB2 bootloader

Why protect the bootloader with a password?

- 1. Prevent *Single* user mode access If an attacker can boot into single user mode, he becomes the root user.
- 2. Prevent access to GRUB console If an attacker manages to use the GRUB console, he can change its configuration or collect information about the system by using the cat command.
- Prevent access to insecure operating systems. If the system has dual boot, an attacker can select an operating system like DOS at boot time that ignores access controls and file permissions.

To password-protect the GRUB2 bootloader:

1. Log in to the operating system as root user and execute the grub2-mkpasswd-pbkdf2 command. The output of this command is as follows:

```
Enter password:
Reenter password:
PBKDF2 hash of your password is
grub.pbkdf2.sha512.10000.D0182EDB28164C19454FA94421D1ECD6309F076F1135A2E5BFE91A50
88BD9EC87687FE14794BE7194F67EA39A8565E868A41C639572F6156900C81C08C1E8413.40F6981C
22F1F81B32E45EC915F2AB6E2635D9A62C0BA67105A9B900D9F365860E84F1B92B2EF3AA0F83CECC6
8E13BA9F4174922877910F026DED961F6592BB7
```

You need to enter your password in the interaction. The ciphertext of the password is the long string "grub.pbkdf2.sha512...".

2. Paste the password ciphertext in the last line of the **/etc/grub.d/00_header** file. The pasted format is as follows:

```
cat <<EOF
set superusers='frank'
password_obkdf2 frank
grub.pbkdf2.sha512.10000.D0182EDB28164C19454FA94421D1ECD6309F076F1135A2E5BFE91A50
88BD9EC87687FE14794BE7194F67EA39A8565E868A41C639572F6156900C81C08C1E8413.40F6981C
22F1F81B32E45EC915F2AB6E2635D9A62C0BA67105A9B900D9F365860E84F1B92B2EF3AA0F83CECC6
8E13BA9F4174922877910F026DED961F6592BB7
E0F</pre>
```

You can replace the 'frank' user with any custom user.

You can also set a plaintext password, for example:

```
cat <<EOF
set superusers='frank'
password frank rockylinux8.x
EOF</pre>
```

- 3. The final step is to run the command grub2-mkconfig -o /boot/grub2/grub.cfg to update GRUB2's settings.
- 4. Restart the operating system to verify GRUB2's encryption. Select the first boot menu item, type the e key, and then enter the corresponding user and password.

Enter username: frank Enter password:

After successful verification, enter $\left[Ctrl + x \right]$ to start the operating system.

Sometimes, you may see in some documents that the grub2-set-password (grub2setpassword) command is used to protect the GRUB2 bootloader:

command	Core functions	Configuration file modification method	automaticity
grub2-set-password	Sets password and update configuration	Auto Completion	high
grub2-mkpasswd- pbkdf2	Only generates encrypted hash values	Requires manual editing	low

Log in to the operating system as the root user and execute the gurb2-set-password command as follows:

```
[root] # grub2-set-password
Enter password:
Confirm password:
[root] # cat /boot/grub2/user.cfg
GRUB2_PASSWORD=grub.pbkdf2.sha512.10000.32E5BAF2C2723B0024C1541F444B8A3656E0A04429
[root] # grub2-mkconfig -o /boot/grub2/grub.cfg
[root] # reboot
```

After executing the grub2-set-password command, the **/boot/grub2/user.cfg** file will be automatically generated.

Select the first boot menu item and type the e key, and then enter the corresponding user and password:

Enter username: root Enter password:

11.3 Systemd

Systemd is a service manager for the Linux operating systems.

The development of systemd was to:

- remain compatible with older SysV initialization scripts,
- provide many features, such as parallel start of system services at system startup, on-demand activation of daemons, support for snapshots, or management of dependencies between services.



systemd introduces the concept of unit files, also known as systemd units.

Туре	File extension	Functionality
Service unit	.service	System service
Target unit	.target	A group of systemd units
Mount unit	.automount	An automatic mount point for file system

Note

There are many types of units: Device unit, Mount unit, Path unit, Scope unit, Slice unit, Snapshot unit, Socket unit, Swap unit, and Timer unit.

- systemd supports system state snapshots and restore.
- You can configure mount points as systemd targets.
- At startup, systemd creates listening sockets for all system services that support this type of activation and passes these sockets to these services as soon as they start. This makes it possible to restart a service without losing a single message sent to it by the network during its unavailability. The corresponding socket remains accessible while all messages queue up.
- System services that use D-BUS for inter-process communications can start ondemand the first time the client uses them.
- systemd stops or restarts only running services. Previous versions (before RHEL7) attempted to stop services directly without checking their current status.
- System services do not inherit any context (like HOME and PATH environment variables). Each service operates in its execution context.

All service unit operations are subject to a 5-minute default timeout to prevent a malfunctioning service from freezing the system.

Due to space limitations, this document will not provide a detailed introduction to systemd. If you have an interest in exploring systemd further, there is a very detailed introduction in this document.

11.3.1 Managing system services

Service units end with the .service file extension and have a similar purpose to init scripts. The use of systemctl command is to display, start, stop, or restart a system service. Except for very few cases, the systemctl single line command can

operate on one or more units in most cases (not limited to the unit type of ".service"). You can view it through the help system.

systemctl	Description
systemctl start name.service	Start one or more services
systemctl stop name.service	Stop one or more services
systemctl restart <i>name</i> .service	Restart one or more services
systemctl reload name.service	Reload one or more services
systemctl status name.service	Check one or more services status
systemctl try-restart name.service	Restart one or more services (If they are running)
systemctl list-unitstype serviceall	Displays the status of all services

The systemctl command is also used for the enable or disable of a system service and displaying associated services:

systemctl	Description
systemctl enable name.service	Activates one or more services
systemctl disable name.service	Disables one or more services
systemctl list-unit-filestype service	Lists all services and checks if they are running
systemctl list-dependenciesafter	Lists the services that start before the specified unit
systemctl list-dependenciesbefore	Lists the services that start after the specified unit

Examples:

```
systemctl stop nfs-server.service
# or
systemctl stop nfs-server
```

To list all units currently loaded:

systemctl list-units --type service

To check the activation status of all units, you can list them with:

systemctl list-unit-files --type service

```
systemctl enable httpd.service
systemctl disable bluetooth.service
```

11.3.2 Example of a .service file for the postfix service

```
postfix.service Unit File
What follows is the content of the /usr/lib/systemd/system/postfix.service
unit file as currently provided by the postfix package:
[Unit]
Description=Postfix Mail Transport Agent
After=syslog.target network.target
Conflicts=sendmail.service exim.service
[Service]
Type=forking
PIDFile=/var/spool/postfix/pid/master.pid
EnvironmentFile=-/etc/sysconfig/network
ExecStartPre=-/usr/libexec/postfix/aliasesdb
ExecStartPre=-/usr/libexec/postfix/chroot-update
ExecStart=/usr/sbin/postfix start
ExecReload=/usr/sbin/postfix reload
ExecStop=/usr/sbin/postfix stop
[Install]
WantedBy=multi-user.target
```

11.3.3 Using system targets

systemd targets replace the concept of run levels on SysV or Upstart.

The representation of systemd targets is by target units. Target units end with the .target file extension, and their sole purpose is to group other systemd units into a chain of dependencies.

For example, the graphical.target unit that starts a graphical session starts system services such as the **GNOME display manager** (gdm.service) or the **accounts service** (accounts-daemon.service) and also activates the multi-user.target unit. If you need to view the dependencies of a certain "target", run the systemctl list-dependencies command. (For example, systemctl list-dependencies multi-user.target).

sysinit.target and basic.target are checkpoints during the startup process. Although one of the design goals of systemd is to start system services in parallel, it is necessary to start the "targets" of certain services and features before starting other services and "targets". Any error in sysinit.target or basic target will cause the initialization of systemd to fail. At this time, your terminal may have entered "emergency mode" (emergency.target).

Target Units	Description
poweroff.target	Shuts down the system and turns it off
rescue.target	Activates a rescue shell
multi-user.target	Activates a multi-user system without a graphical interface
graphical.target	Activates a multi-user system with a graphical interface
reboot.target	Shuts down and restarts the system

The default target

To determine the default target used by default:

systemctl get-default

This command searches for the target of the symbolic link located at /etc/systemd/ system/default.target and displays the result.

```
$ systemctl get-default
graphical.target
```

The systemctl command can also provide a list of available targets:

systemctl list-unitstype target				
UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
basic.target	loaded	active	active	Basic System
bluetooth.target	loaded	active	active	Bluetooth
cryptsetup.target	loaded	active	active	Encrypted Volumes
getty.target	loaded	active	active	Login Prompts
graphical.target	loaded	active	active	Graphical Interface
local-fs-pre.target	loaded	active	active	Local File Systems (Pre)
local-fs.target	loaded	active	active	Local File Systems
multi-user.target	loaded	active	active	Multi-User System
network-online.target	loaded	active	active	Network is Online
network.target	loaded	active	active	Network

nss-user-lookup.target	loaded active	e active User and Group Name Lookups
paths.target	loaded active	active Paths
remote-fs.target	loaded active	e active Remote File Systems
slices.target	loaded active	active Slices
sockets.target	loaded active	active Sockets
sound.target	loaded active	e active Sound Card
swap.target	loaded active	e active Swap
sysinit.target	loaded active	active System Initialization
timers.target	loaded active	active Timers

To configure the system to use a different default target:

systemctl set-default name.target

Example:

```
# systemctl set-default multi-user.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/multi-user.target' '/etc/systemd/system/
default.target'
```

To switch to a different target unit in the current session:

systemctl isolate name.target

The **Rescue mode** provides a simple environment for repairing your system in cases where a normal boot process is impossible.

In rescue mode, the system attempts to mount all local file systems and start several important system services but does not enable a network interface or allow other users to connect to the system simultaneously.

On Rocky 8, the rescue mode is equivalent to the old single user mode and requires the root password.

To change the current target and enter rescue mode in the current session:

```
systemctl rescue
```

Emergency mode provides the most minimalist environment possible and allows the system to be repaired even in situations where it is unable to enter rescue

mode. In emergency mode, the operating system mounts the root file system with the read-only option. It will not attempt to mount any other local file system, will not activate any network interface, and will start some essential services.

To change the current target and enter emergency mode in the current session:

systemctl emergency

Shutdown, suspension, and hibernation

The systemctl command replaces many power management commands used in previous versions:

Old command	New command	Description
halt	systemctl halt	Shuts down the system.
poweroff	systemctl poweroff	Turns off the system.
reboot	systemctl reboot	Restarts the system.
pm-suspend	systemctl suspend	Suspends the system.
pm-hibernate	systemctl hibernate	Hibernates the system.
pm-suspend-hybrid	systemctl hybrid-sleep	Hibernates and suspends the system.

11.3.4 The journald process

You can manage log files with the journald daemon, a component of systemd 'in addition to ' rsyslogd.

The journald daemon is responsible for capturing the following types of log messages:

- Syslog messages
- Kernel log messages
- Initramfs and system startup logs
- Standard output (stdout) and standard error output (stderr) information of all services

After capture, journald will index these logs and provide them to users through structured storage mechanism This mechanism stores logs in binary format,

supports tracking events in chronological order, and provides flexible filtering, searching and output capabilities in multiple formats (such as text/JSON). Note that journald does not enable log persistence by default, which means this component only retain and record all logs since startup. After the operating system restarts, the deletion of historical logs occurs. By default, all temporarily saved log files are in the **/run/log/journal/** directory.

11.3.5 journalctl command

The journalctl command is used to parse log files saved in binary, such as viewing log files, filtering logs, and controlling output entries.

journalctl

If you do not enter the command with any other options, the output log content is similar to the /var/log/messages file, but journalctl provides the following improvements:

- shows the priority of entries is visually marked
- shows the conversion of timestamps to the local time zone of your system
- all logged data is displayed, including rotating logs
- shows the marking of the beginning of a start with a special line

Using continuous display

With continuous display, log messages are displayed in real time.

journalctl -f

This command returns a list of the ten most recent log lines. The journalctl utility then continues to run and waits for new changes to occur before displaying them immediately.

Filtering messages

It is possible to use different filtering methods to extract information that fits different needs. Log messages are often used to track erroneous behavior on the system. To view entries with a selected or higher priority:

```
journalctl -p priority
```

You must replace priority with one of the following keywords (or a number):

- debug (7),
- info (6),
- notice (5),
- warning (4),
- err (3),
- crit (2),
- alert (1),
- emerg (0).

If you want to know more about the content of logs, there are more comprehensive introductions and descriptions in this document.

12. Task Management

In this chapter, you will learn how to manage scheduled tasks.

Objectives: In this chapter, future Linux administrators will learn how to:

Linux deals with the task scheduling;
restrict the use of cron to certain users;
schedule tasks.

🕅 crontab, crond, scheduling, linux

Knowledge: $\star \star$ Complexity: $\star \star$

Reading time: 15 minutes

12.1 Generalities

The scheduling of tasks is managed with the cron utility. It allows the periodic execution of tasks.

It is reserved for administrators to perform system tasks, but can also be used by normal users for tasks or scripts that they have access to. To access the cron utility, we use: crontab.

The cron service is used for:

- Repetitive administration operations;
- Backups;
- Monitoring of system activity;
- Program execution.

crontab is short for **cron table**, but can be thought of as a task scheduling table.

🛕 Warning

To set up a schedule, the system must have the correct time set.

12.2 How the service works

A crond daemon present runs the cron service in memory.

To check its status:

[root] # systemctl status crond

🜢 Tip

If the crond daemon is not running, you will have to initialize it manually and/or automatically at startup. Indeed, even if tasks are scheduled, they will not be launched.

Initialization of the crond daemon in the manual:

[root]# systemctl {status|start|restart|stop} crond

Initialization of the crond daemon at startup:

[root]# systemctl enable crond

12.3 Security

To implement a schedule, a user must have permission to use the cron service.

This permission varies according to the information contained in the files below:

- /etc/cron.allow
- /etc/cron.deny

🛕 Warning

If neither file is present, all users can use cron.

12.3.1 The cron.allow and cron.deny Files

File /etc/cron.allow

Only users contained in this file are allowed to use cron.

If it exists and is empty, no users can use cron.



```
File /etc/cron.deny
```

Users in this file are not allowed to use cron.

If it is empty, all users can use cron.

By default, /etc/cron.deny exists and is empty and /etc/cron.allow does not exist. When two files exist at the same time, the system only uses the cron.allow content as the judgment basis and completely ignores the existence of cron.deny files.

12.3.2 Allowing a user

Only **user1** will be able to use cron.

```
[root]# vi /etc/cron.allow
user1
```

12.3.3 Prohibit a user

Only **user2** will not be able to use cron. Note that the /etc/cron.allow file cannot exist.

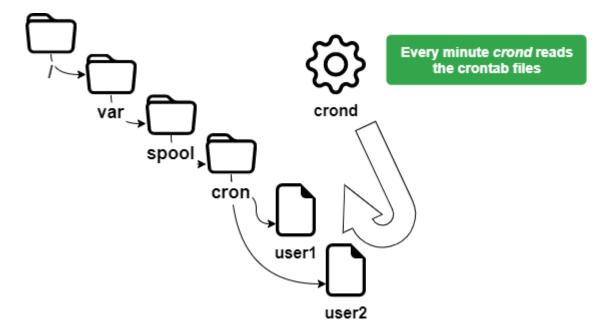
```
[root]# vi /etc/cron.deny
user2
```

If the same user exists in /etc/cron.deny and /etc/cron.allow at the same time, the user can use cron normally.

12.4 Scheduling tasks

When a user schedules a task, there is a file created with their name under /var/ spool/cron/.

This file contains all the information the crond needs to know regarding tasks created by this user, including the commands or programs to run, and the schedule for running them (hour, minute, day, etc.). Note that the minimum time unit that crond can recognize is 1 minute. There are similar scheduling tasks in RDBMS (such as MySQL), where time-based scheduling tasks are referred to as the "Event Scheduler" (whose recognizable time unit is 1 second), and event-based scheduling tasks are referred to as "Triggers".



12.4.1 The crontab command

The crontab command is used to manage the schedule file.

crontab [-u user] [-e | -l | -r]

Example:

[root]# crontab -u user1 -e

Option	Description
- e	Edits the schedule file with vi
-1	Displays the contents of the schedule file
-u <user></user>	Specify a single user to operate
- r	Deletes the schedule file

🛕 Warning

crontab without options deletes the old schedule file and waits for the user to enter new lines. You have to press ctrl + d to exit this editing mode.

Only the root can use the -u <user> option to manage another user's schedule file.

The example above allows the root to schedule a task for user1.

12.4.2 Uses of crontab

The uses of crontab are many and include:

- Modifications to the crontab files taken into account immediately;
- No need to restart.

On the other hand, the following points must be taken into account:

- The program must be autonomous;
- Provide redirections (stdin, stdout, stderr);
- It is not relevant to run commands that use input/output requests on a terminal.

Note

It is important to understand that the purpose of scheduling is to perform tasks automatically, without the need for external intervention.

12.5 The crontab file

The crontab file is structured according to the following rules.

- Each line of this file corresponds to a schedule;
- Each line has six fields, 5 for the time and 1 for the order;
- A space or a tab separates each field;
- Each line ends with a carriage return;
- A # at the beginning of the line comments it.

```
[root]# crontab -e
10 4 1 * * /root/scripts/backup.sh
1 2 3 4 5 6
```

Field	Description	Detail
1	Minute(s)	From 0 to 59
2	Hour(s)	From 0 to 23
3	Day(s) of the month	From 1 to 31
4	Month of the year	From 1 to 12
5	Day(s) of the week	From 0 to 7 ($0=7=sunday$)
6	Task to execute	Full command or script

🛕 Warning

The tasks to be executed must use absolute paths, and if possible, use redirects.

To simplify the notation for the definition of time, it is advisable to use special symbols.

Special symbol	Description
*	Indicates all the time values of the field
	Indicates a continuous time range
7	Indicates the discontinuous time range
Ζ	Indicateds time interval

Examples:

Script executed on April 15 at 10:25 am:

25 10 15 04 * /root/scripts/script > /log/...

Run the task once a day at 11 am and once a day at 4 pm:

00 11,16 * * * /root/scripts/script > /log/...

The task runs once an hour from 11 am to 4 pm every day:

00 11-16 * * * /root/scripts/script > /log/...

Run every 10 minutes during working hours on weekdays:

*/10 8-17 * * 1-5 /root/scripts/script > /log/...

For the root user, crontab also has some special time settings:

Setting	Description
@reboot	Runs a command on system reboot
@hourly	Runs a command every hour
@daily	Runs daily just after midnight
@weekly	Runs command every Sunday just after midnight
@monthly	Runs command on the first day of the month just after midnight
@annually	Runs January 1st just after midnight

12.5.1 Task execution process

A user, rockstar, wants to edit his crontab file:

- The crond daemon checks to see if the user is allowed (/etc/cron.allow and /etc/ cron.deny).
- 2. If the user is allowed, they access their crontab file (/var/spool/cron/rockstar).

The crond daemon:

- Reads Reads the scheduled task files of all users every minute.
- Runs Runs tasks according to the schedule.
- Writes Writes the corresponding events and messages to the (/var/log/cron) file.

13. Implementing the Network

In this chapter you will learn how to work with and manage the network.

Objectives: In this chapter you will learn how to:

- ✓ Configure a workstation to use DHCP;
- ✓ Configure a workstation to use a static configuration;
- ✓ Configure a workstation to use a gateway;
- ✓ Configure a workstation to use DNS servers;
- ✓ Troubleshoot the network of a workstation.

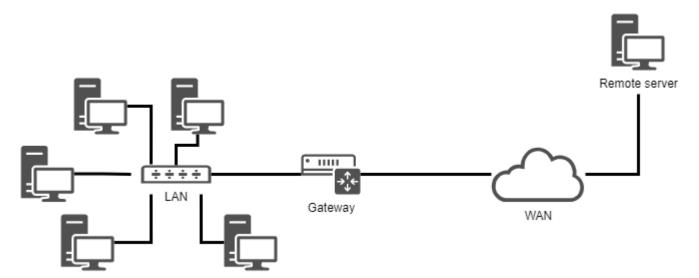
🛤 network, linux, ip

Knowledge: $\star \star$ Complexity: $\star \star$

Reading time: 30 minutes

13.1 Generalities

To illustrate this chapter, we will use the following architecture.



It will allow us to consider:

- integration in a LAN (local area network);
- the configuration of a gateway to reach a remote server;
- the configuration of a DNS server and the implementation of name resolution.

The minimum parameters to be defined for the machine are:

- the name of the machine;
- the IP address;
- the subnet mask.

Example:

- pc-rocky;
- 192.168.1.10;
- 255.255.255.0.

The notation called CIDR is more and more frequent: 192.168.1.10/24

IP addresses are used for the proper routing of messages (packets). They are divided into two parts:

- the fixed part, identifying the network;
- the identifier of the host in the network.

The subnet mask is a set of **4 bytes** intended to isolate:

- the network address (NetID or SubnetID) by performing a bitwise logical AND between the IP address and the mask;
- the host address (**HostID**) by performing a bitwise logical AND between the IP address and the complement of the mask.

There are also specific addresses within a network, which must be identified. The first address of a range as well as the last one have a particular role:

- The first address of a range is the **network address**. It is used to identify networks and to route information from one network to another.
- The last address of a range is the **broadcast address**. It is used to broadcast information to all the machines on the network.

13.1.1 MAC address / IP address

A **MAC address** is a physical identifier written in the factory onto the device. This is sometimes referred to as the hardware address. It consists of 6 bytes often given in hexadecimal form (for example 5E:FF:56:A2:AF:15). It is composed of: 3 bytes of the manufacturer identifier and 3 bytes of the serial number.

🔺 Warning

This last statement is nowadays a little less true with virtualization. There are also software solutions for changing the MAC address.

An Internet Protocol (**IP**) address is an identification number permanently or temporarily assigned to each device connected to a computer network using the Internet Protocol. One part defines the network address (NetID or SubnetID as the case may be), the other part defines the address of the host in the network (HostID). The relative size of each part varies according to the network (sub)mask.

An IPv4 address defines an address on 4 bytes. The number of available addresses being close to saturation a new standard was created, the IPv6 defined on 16 bytes.

IPv6 is often represented by 8 groups of 2 bytes separated by a colon. Insignificant zeros can be omitted, one or more groups of 4 consecutive zeros can be replaced by a double colon.

```
Subnet masks have from 0 to 128 bits. (for example 21ac: 0000:0000:0611:21e0:00ba:321b:54da/64 or 21ac::611:21e0321b:54da/64)
```

In a web address or URL (Uniform Resource Locator), an ip address can be followed by a colon and the port address (which indicates the application to which the data is destined). Also to avoid confusion in a URL, the IPv6 address is written in square brackets [], colon, port address.

IP and MAC addresses must be unique on a network!

13.1.2 DNS Domain

Client machines can be part of a DNS (**Domain Name System**, e.g., mydomain.lan) domain.

The fully qualified machine name (FQDN) becomes pc-rocky.mydomain.lan.

A set of computers can be grouped into a logical, name-resolving, set called a DNS domain. A DNS domain is not, of course, limited to a single physical network.

In order for a computer to be part of a DNS domain, it must be given a DNS suffix (here mydomain.lan) as well as servers that it can query.

13.1.3 Reminder of the OSI model

Memory aid		
To remember the ord- Alligator.	er of the layers of the OSI model, remember the foll	owing sentence: Please Do Not Touch Steven's Pet
Layer	Protocols	
7 - Application	POP, IMAP, SMTP, SSH, SNMP, HTTP, FTP,	
6 - Presentation	ASCII, MIME,	
5 - Session	TLS, SSL, NetBIOS,	
4 - Transport	TLS, SSL, TCP, UDP,	
3 - Network	IPv4, IPv6, ARP,	
2 - Data Link	Ethernet, WiFi, Token Ring,	
1 - Physical	Cables, optical fibers, radio waves,	

Layer 1 (Physical) supports transmission over a communication channel (Wifi, Optical fiber, RJ cable, etc.). Unit: the bit.

Layer 2 (Data Link) supports network topology (token-ring, star, bus, etc.), data splitting and transmission errors. Unit: the frame.

Layer 3 (Network) supports end-to-end data transmission (IP routing = Gateway). Unit: the packet.

Layer 4 (Transport) supports service type (connected or unconnected) encryption and flow control. Unit: the segment or the datagram.

Layer 5 (Session) supports the communication between two computers.

Layer 6 (Presentation) represents the area that is independent of data at the application layer. Essentially this layer translates from network format to the application format, or from the application format to the network format.

Layer 7 (Application) represents the contact with the user. It provides the services offered by the network: http, dns, ftp, imap, pop, smtp, etc.

13.2 The naming of interfaces

lo is the "**loopback**" interface which allows TCP/IP programs to communicate with each other without leaving the local machine. This enables testing if the **network module of the system is working properly** and also allows pinging the localhost. All packets that enter through localhost leave through localhost. The packets received are the packets sent.

The Linux kernel assigns interface names with a specific prefix depending on the type. Traditionally, all **Ethernet** interfaces, for example, began with **eth**. The prefix was followed by a number, the first being 0 (eth0, eth1, eth2...). The wifi interfaces were given a wlan prefix.

On Rocky8 Linux distributions, systemd will name interfaces with the new following policy where "X" represents a number:

- enoX : on-board devices
- ensX : PCI Express hotplug slot
- enpXsX : physical/geographical location of the connector of the hardware

• ...

13.3 Using the ip command

Forget the old ifconfig command! Think ip!

✓ Note
Comment for administrators of older Linux systems:
The historical network management command is ifconfig. This command has been replaced by the ip command, which is already well known to network administrators.
The ip command is the only command to manage IP address, ARP, routing, etc. .
The ifconfig command is no longer installed by default in Rocky8.
It is important to get into good habits now.

13.4 The hostname

The hostname command displays or sets the host name of the system

This command is used by various network programs to identify the machine.

To assign a host name, it is possible to use the hostname command, but the changes will not be retained at the next boot. The command with no arguments displays the host name.

To set the host name, the file /etc/sysconfig/network must be modified:

```
NETWORKING=yes
HOSTNAME=pc-rocky.mondomaine.lan
```

The RedHat boot script also consults the /etc/hosts file to resolve the host name of the system.

When the system boots, Linux evaluates the HOSTNAME value in the /etc/sysconfig/ network file.

It then uses the /etc/hosts file to evaluate the main IP address of the server and its host name. It deduces the DNS domain name.

It is therefore essential to fill in these two files before any configuration of network services.



13.5 /etc/hosts file

The /etc/hosts file is a static host name mapping table, which follows the following format:

@IP <hostname> [alias] [# comment]

Example of /etc/hosts file:

127.0.0.1localhost localhost.localdomain::1localhost localhost.localdomain192.168.1.10rockstar.rockylinux.lan rockstar

The /etc/hosts file is still used by the system, especially at boot time when the system FQDN is determined.

ó Tip	
RedHat recommends that at least one line containing the system name be filled in.	

If the **DNS** service (**D**omain **N**ame **S**ervice) is not in place, you must fill in all the names in the hosts file for each of your machines.

The /etc/hosts file contains one line per entry, with the IP address, the FQDN, then the host name (in that order) and a series of aliases (alias1 alias2 ...). The alias is an option.

13.6 /etc/nsswitch.conf file

The **NSS** (Name Service Switch) allows configuration files (e.g., /etc/ passwd, /etc/group, /etc/hosts) to be substituted for one or more centralized databases.

The /etc/nsswitch.conf file is used to configure the name service databases.

```
passwd: files
shadow: files
group: files
hosts: files dns
```

In this case, Linux will first look for a host name match (hosts: line) in the /etc/ hosts file (files value) before querying DNS (dns value)! This behavior can simply be changed by editing the /etc/nsswitch.conf file.

Of course, it is possible to imagine querying an LDAP, MySQL or other server by configuring the name service to respond to system requests for hosts, users, groups, etc.

The resolution of the name service can be tested with the getent command that we will see later in this course.

13.7 /etc/resolv.conf file

The /etc/resolv.conf file contains the DNS name resolution configuration.

```
#Generated by NetworkManager
domain mondomaine.lan
search mondomaine.lan
nameserver 192.168.1.254
```

🗴 Tip

This file is historical. It is no longer filled in directly!

Newer generations of distributions have generally integrated the NetworkManager service. This service allows you to manage the configuration more efficiently, either in graphical or console mode.

It allows for the addition of DNS servers from the configuration file of a network interface. It then dynamically populates the /etc/resolv.conf file which should never be edited directly, otherwise the configuration changes will be lost the next time the network service is started.

13.8 ip command

The ip command from the iproute2 package allows you to configure an interface and its routing table.

Display interfaces:

[root]# ip link

Display interfaces information:

[root]# ip addr show

Display the information of an interface:

[root]# ip addr show eth0

Display the ARP table:

[root]# ip neigh

All historical network management commands have been grouped under the ip command, which is well known to network administrators.

13.9 DHCP configuration

The **DHCP** protocol (**D**ynamic **H**ost **C**ontrol **P**rotocol) allows you to obtain a complete IP configuration via the network. This is the default configuration mode of

a network interface under Rocky Linux, which explains why a system connected to the network of an Internet router can function without additional configuration.

The configuration of interfaces under Rocky Linux is done in the /etc/sysconfig/ network-scripts/ folder.

For each Ethernet interface, a ifcfg-ethx file allows for the configuration of the associated interface.

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
HWADDR=00:0c:29:96:32:e3
```

• Interface name: (must be in the file name)

DEVICE=eth0

• Automatically start the interface:

ONBOOT=yes

• Make a DHCP request when the interface starts up:

```
B00TPR0T0=dhcp
```

• Specify the MAC address (optional but useful when there are several interfaces):

HWADDR=00:0c:29:96:32:e3

🜢 Tip

If NetworkManager is installed, the changes are taken into account automatically. If not, you have to restart the network service.

• Restart the network service:

```
[root]# systemctl restart NetworkManager
```

13.10 Static configuration

The static configuration requires at least:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=none
IPADDR=192.168.1.10
NETMASK=255.255.255.0
```

• Here we are replacing "dhcp" with "none" which equals static configuration:

B00TPR0T0=none

• IP Address:

IPADDR=192.168.1.10

• Subnet mask:

NETMASK=255.255.255.0

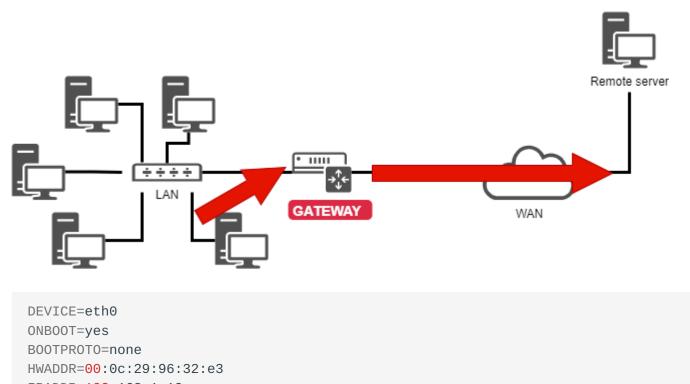
• The mask can be specified with a prefix:

PREFIX=24

🛕 Warning

You must use NETMASK OR PREFIX - Not both!

13.11 Routing



IPADDR=**192**.168.1.10 NETMASK=**255**.255.255.0

GATEWAY=192.168.1.254

The ip route command:

[root]# ip route show
192.168.1.0/24 dev eth0 [...] src 192.168.1.10 metric 1
default via 192.168.1.254 dev eth0 proto static

It is a good idea to know how to read a routing table, especially in an environment with multiple network interfaces.

- In the example shown, the 192.168.1.0/24 network is reachable directly from the eth0 device, so there is a metric at 1 (does not traverse a router).
- All other networks than the previous one will be reachable, again from the etho device, but this time the packets will be addressed to a 192.168.1.254 gateway. The routing protocol is a static protocol (although it is possible to add a route to a dynamically assigned address in Linux).

13.12 Name resolution

A system needs to resolve:

• FQDNs into IP addresses

www.free.fr = 212.27.48.10

• IP addresses into names

212.27.48.10 = www.free.fr

• or to obtain information about an area:

MX de free.fr = 10 mx1.free.fr + 20 mx2.free.fr

```
DEVICE=eth0
ONB00T=yes
B00TPR0T0=none
HWADDR=00:0c:29:96:32:e3
IPADDR=192.168.1.10
NETMASK=255.255.255.0
GATEWAY=192.168.1.254
DNS1=172.16.1.2
DNS2=172.16.1.3
DOMAIN=rockylinux.lan
```

In this case, to reach the DNS, you have to go through the gateway.

```
#Generated by NetworkManager
domain mondomaine.lan
search mondomaine.lan
nameserver 172.16.1.2
nameserver 172.16.1.3
```

The file has been updated by NetworkManager.

13.13 Troubleshooting

The ping command sends datagrams to another machine and waits for a response.

It is the basic command for testing the network because it checks the connectivity between your network interface and another.

Syntax of the ping command:

ping [-c numerical] destination

The -c (count) option allows you to stop the command after the countdown in seconds.

Example:

[root]# ping -c 4 localhost

🗴 Tip

Validate connectivity from near to far

1. Validate the TCP/IP software layer

```
[root]# ping localhost
```

"Pinging" the inner loop does not detect a hardware failure on the network interface. It simply determines whether the IP software configuration is correct.

2. Validate the network card

[root]# ping 192.168.1.10

To determine the functionality of the network card, we must ping its IP address. If the network cable is not connected to the network card, it should be in a "down" state.

If the ping does not work, first check the network cable to your network switch and reassemble the interface (see the if up command), then check the interface itself.

- 232/285 -

3. Validate the connectivity of the gateway

[root]# ping 192.168.1.254

4. Validate the connectivity of a remote server

[root]# ping 172.16.1.2

5. Validate the DNS service

```
[root]# ping www.free.fr
```

13.13.1 dig command

The dig command is used to query the DNS server.

The dig command syntax:

dig [-t type] [+short] [name]

Examples:

```
[root]# dig +short rockylinux.org
76.223.126.88
[root]# dig -t MX +short
rockylinux.org
5 alt1.aspmx.l.google.com.
...
```

The dig command is used to query DNS servers. It is verbose by default, but the +short option can change this behavior.

It is also possible to specify a DNS **record type** to resolve, such as an MX **type** to get information about the mail exchangers for a domain.

13.13.2 getent command

The getent (get entry) command gets an NSSwitch entry (hosts + dns)

Syntax of the getent command:

getent hosts name

Example:

```
[root]# getent hosts rockylinux.org
76.223.126.88 rockylinux.org
```

Querying only a DNS server may return an erroneous result that does not consider the contents of a hosts file, although this should be rare nowadays.

To take the /etc/hosts file into account, the NSSwitch name service must be queried, which will take care of any DNS resolution.

13.13.3 ipcalc command

The ipcalc (**ip calculation**) command calculates the address of a network or broadcast from an IP address and a mask.

Syntax of the ipcalc command:

ipcalc [options] IP <netmask>

Example:

[root]# ipcalc -b 172.16.66.203 255.255.240.0
BROADCAST=172.16.79.255

🜢 Tip

This command is interesting, followed by a redirection to fill in the configuration files of your interfaces automatically:

[root]# ipcalc -b 172.16.66.203 255.255.240.0 >> /etc/sysconfig/network-scripts/ifcfg-eth0

Option	Description
- b	Displays the broadcast address.
- n	Displays the network address and mask.

ipcalc is a simple way to calculate a host's IP information. The various options indicate what information ipcalc should display on the standard output. You can specify multiple options. You must specify an IP address on which to operate. Most operations also require a network mask or CIDR prefix.

Option short	Option long	Description
- b	broadcast	Displays the broadcast address of the given IP address and the network mask.
- h	hostname	Displays the hostname of the IP address given via DNS.
-n	netmask	Calculates the network mask for the given IP address. Assumes that the IP address is part of a complete class A, B, or C network. Many networks do not use default network masks, in which case an incorrect value will be returned.
- p	prefix	Indicates the prefix of the mask/IP address.
- n	network	Indicates the network address of the given IP address and mask.
- S	silent	Does not display any error messages.

13.13.4 ss command

The ss (socket statistics) command displays the listening ports on the network.

Syntax of the ss command:

ss [-tuna]

Example:

[root]# ss -tuna
tcp LISTEN 0 128 *:22 *:*

The commands ss and netstat (to follow) will be very important for the rest of your Linux life.

When implementing network services, it is common to check with one of these two commands that the service is listening on the expected ports.

13.13.5 netstat command

🛕 Warning

The netstat command is now deprecated and is no longer installed by default on Rocky Linux. You may still find some Linux versions that have it installed, but it is best to move on to using ss for everything that you would have used netstat for.

The netstat command (**network statistics**) displays the listening ports on the network.

Syntax of the netstat command:

netstat -tapn

Example:

```
[root]# netstat -tapn
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN 2161/sshd
```

13.13.6 IP or MAC address conflicts

A misconfiguration can cause multiple interfaces to use the same IP address. This can happen when a network has multiple DHCP servers, or the same IP address is manually assigned numerous times.

When the network is malfunctioning, and when an IP address conflict could be the cause, it is possible to use the arp-scan software (requires the EPEL repository):

dnf install arp-scan

Example:

```
$ arp-scan -I eth0 -l
172.16.1.104 00:01:02:03:04:05
172.16.1.107 00:0c:29:1b:eb:97
172.16.1.250 00:26:ab:b1:b7:f6
172.16.1.252 00:50:56:a9:6a:ed
172.16.1.253 00:50:56:b6:78:ec
172.16.1.253 00:50:56:b6:78:ec
172.16.1.253 00:50:56:b6:78:ec
172.16.1.253 00:50:56:b6:78:ec
172.16.1.253 00:50:56:b6:78:ec
172.16.1.253 00:50:56:b6:78:ec
```

VMware, Inc. (Unknown) VMWare, Inc. VMWare, Inc. VMWare, Inc. (DUP: 2) VMWare, Inc. (DUP: 3) VMWare, Inc. (DUP: 4) (Unknown) (DUP: 2)

3COM CORPORATION

🜢 Tip

As the above example shows, MAC address conflicts are possible! Virtualization technologies and the copying of virtual machines cause these problems.

13.14 Hot configuration

The ip command can hot add an IP address to an interface.

ip addr add @IP dev DEVICE

Example:

[root]# ip addr add 192.168.2.10 dev eth1

The ip command allows for the activation or deactivation of an interface:

ip link set DEVICE up
ip link set DEVICE down

Example:

[root]# ip link set eth1 up
[root]# ip link set eth1 down

The ip command adds a route:

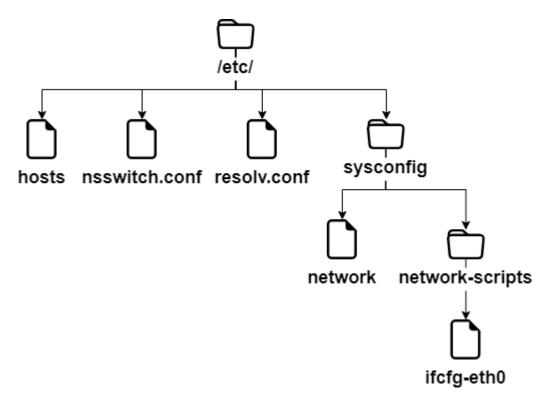
```
ip route add [default|netaddr] via @IP [dev device]
```

Example:

```
[root]# ip route add default via 192.168.1.254
[root]# ip route add 192.168.100.0/24 via 192.168.2.254 dev eth1
```

13.15 In summary

The files used in this chapter are:



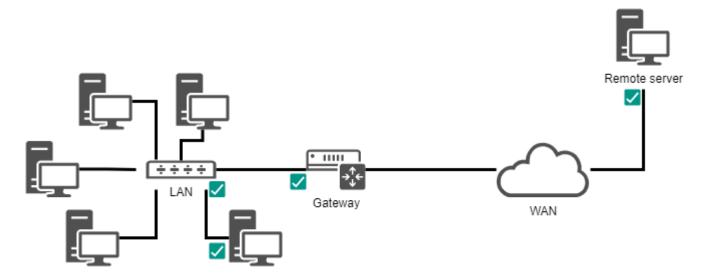
A complete interface configuration could be this (file /etc/sysconfig/network-

scripts/ifcfg-eth0):

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=none
HWADDR=00:0c:29:96:32:e3
IPADDR=192.168.1.10
NETMASK=255.255.255.0
GATEWAY=192.168.1.254
DNS1=172.16.1.1
DNS2=172.16.1.2
DOMAIN=rockylinux.lan
```

The troubleshooting method should go from closest to farthest:

- 1. ping localhost (software test)
- 2. ping IP-address (hardware test)
- 3. ping gateway (connectivity test)
- 4. ping remote server (routing test)
- 5. DNS query (dig or ping)



14. Software Management

14.1 Generalities

On a Linux system, it is possible to install software in two ways:

- Using an installation package;
- Compiling from source files.

Note
Installing from source is not covered here. As a rule, you should use the package method unless the software you want is not
available via the package manager. The reason for this is that dependencies are generally managed by the package system, whereas
with source, you need to manage the dependencies manually.

The package: This is a single file containing all the data needed to install the program. It can be executed directly on the system from a software repository.

The source files: Some software is not provided in packages ready to be installed, but via an archive containing the source files. It is up to the administrator to prepare these files and compile them to install the program.

14.2 RPM: RedHat Package Manager

RPM (RedHat Package Manager) is a software management system. It is possible to install, uninstall, update or check software contained in packages.

RPM is the format used by all RedHat based distributions (RockyLinux, Fedora, CentOS, SuSe, Mandriva, ...). Its equivalent in the Debian world is DPKG (Debian Package).

The name of an RPM package follows a specific nomenclature:

Release Architecture version 7 0-15 Distribution Extension Package name

14.2.1 rpm command

The rpm command allows you to install a package.

rpm [-i][-U] package.rpm [-e] package

Example (for a package named 'package'):

rpm -ivh package.rpm

Option	Description
-i package.rpm	Installs the package.
-U package.rpm	Updates an already installed package.
-e package.rpm	Uninstalls the package.
-h	Displays a progress bar.
- V	Informs about the progress of the operation.
test	Tests the command without executing it.

The rpm command also allows you to query the system package database by adding the -q option.

It is possible to execute several types of queries to obtain different information about the installed packages. The RPM database is located in the directory /var/lib/rpm.

Example:

rpm -qa

This command queries all the packages installed on the system.

rpm -q [-a][-i][-l] package [-f] file

Example:

rpm -qil package rpm -qf /path/to/file

Option	Description
-a	Lists all packages installed on the system.
-ipackage	Displays the package information.
-lpackage	Lists the files contained in the package.
- f	Shows the name of the package containing the specified file.
last	The list of packages is given by installation date (the last installed packages appear first).

🛕 Warning

After the -q option, the package name must be exact. Metacharacters (wildcards) are not supported.

🜢 Tip

However, it is possible to list all installed packages and filter with the grep command.

Example: list the last installed packages:

```
sudo rpm -qa --last | head
NetworkManager-config-server-1.26.0-13.el8.noarch Mon 24 May 2021 02:34:00 PM
CEST
iwl2030-firmware-18.168.6.1-101.el8.1.noarch Mon 24 May 2021 02:34:00 PM CEST
iwl2000-firmware-18.168.6.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
iwl135-firmware-18.168.6.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
iwl105-firmware-18.168.6.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
iwl100-firmware-39.31.5.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
iwl1000-firmware-39.31.5.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
alsa-sof-firmware-1.5-2.el8.noarch
                                              Mon 24 May 2021 02:34:00 PM CEST
iwl7260-firmware-25.30.13.0-101.el8.1.noarch
                                              Mon 24 May 2021 02:33:59 PM CEST
iwl6050-firmware-41.28.5.1-101.el8.1.noarch
                                              Mon 24 May 2021 02:33:59 PM CEST
```

Example: list the installation history of the kernel:

sudo rpm -qalast kernel	
kernel-4.18.0-305.el8.x86_64	Tue 25 May 2021 06:04:56 AM CEST
kernel-4.18.0-240.22.1.el8.x86_64	Mon 24 May 2021 02:33:35 PM CEST

Example: list all installed packages with a specific name using grep :

```
sudo dnf list installed | grep httpd
centos-logos-httpd.noarch 80.5-2.el8
@baseos
httpd.x86_64 2.4.37-30.module_el8.3.0+561+97fdbbcc
@appstream
httpd-filesystem.noarch 2.4.37-30.module_el8.3.0+561+97fdbbcc
@appstream
httpd-tools.x86_64 2.4.37-30.module_el8.3.0+561+97fdbbcc
@appstream
```

14.3 DNF: Dandified Yum

DNF (**Dandified Yum**) is a software package manager, successor of **YUM** (**Y**ellow dog **U**pdater **M**odified). It works with **RPM** packages grouped in a local or remote repository (a directory for storing packages). For the most common commands, its usage is identical to that of yum.

The dnf command allows the management of packages by comparing those installed on the system with those in the repositories defined on the server. It also automatically installs dependencies, if they are also present in the repositories.

dnf is the manager used by many RedHat based distributions (RockyLinux, Fedora, CentOS, ...). Its equivalent in the Debian world is **APT** (Advanced **P**ackaging **T**ool).

14.3.1 dnf command

The dnf command allows you to install a package by specifying only the short name.

dnf [install][remove][list all][search][info] package

Example:

dnf install tree

Only the short name of the package is required.

Option	Description
install	Installs the package.
remove	Uninstall the package.
list all	Lists the packages already in the repository.
search	Search for a package in the repository.
provides */command_name	Search for a command.
info	Displays the package information.
autoremove	Removes all packages installed as dependencies but no longer needed.

The dnf install command allows you to install the desired package without worrying about its dependencies, which will be resolved directly by dnf itself.

Package	Archite	- cture		
Version	=========	Repository	:	Size
======================================		=		
nginx	aarch64	1:		
1.14.1-9.module+el8.4.0+542+81547229 Installing dependencies:		appstream	543	k
nginx-all-modules	noarch	1:		
1.14.1-9.module+el8.4.0+542+81547229		appstream	22	k
nginx-mod-http-image-filter	aarch64	1:		
1.14.1-9.module+el8.4.0+542+81547229		appstream	33	k
nginx-mod-http-perl	aarch64	1:		
1.14.1-9.module+el8.4.0+542+81547229		appstream	44	K
nginx-mod-http-xslt-filter 1.14.1-9.module+el8.4.0+542+81547229	aarch64		32	k
nginx-mod-mail	aarch64	appstream 1:	32	ĸ
1.14.1-9.module+el8.4.0+542+81547229		appstream	60	k
nginx-mod-stream	aarch64	• •		IX.
1.14.1-9.module+el8.4.0+542+81547229		appstream	82	k
Transaction Summary				

Total download size: 816 k Installed size: 2.2 M Is this ok [y/N]:

In case you don't remember the exact name of the package, you can search for it with the command dnf search name. As you can see, there is a section that contains the exact name and another one that contains the package correspondence, all of which are highlighted for easier searching.

```
dnf search nginx
Last metadata expiration check: 0:20:55 ago on Wed 23 Mar 2022 10:40:43 AM CET.
nginx.aarch64 : A high performance web server and reverse proxy server
======= Name & Summary Matched: nginx
_____
collectd-nginx.aarch64 : Nginx plugin for collectd
munin-nginx.noarch : NGINX support for Munin resource monitoring
nginx-all-modules.noarch : A meta package that installs all available Nginx
modules
nginx-filesystem.noarch : The basic directory layout for the Nginx server
nginx-mod-http-image-filter.aarch64 : Nginx HTTP image filter module
nginx-mod-http-perl.aarch64 : Nginx HTTP perl module
nginx-mod-http-xslt-filter.aarch64 : Nginx XSLT module
nginx-mod-mail.aarch64 : Nginx mail modules
nginx-mod-stream.aarch64 : Nginx stream modules
pagure-web-nginx.noarch : Nginx configuration for Pagure
pcp-pmda-nginx.aarch64 : Performance Co-Pilot (PCP) metrics for the Nginx
Webserver
python3-certbot-nginx.noarch : The nginx plugin for certbot
```

Another way to search for a package by entering an additional search key is to send the result of the dnf command through a pipe to the grep command with the desired key.

```
dnf search nginx | grep mod
Last metadata expiration check: 3:44:49 ago on Wed 23 Mar 2022 06:16:47 PM CET.
nginx-all-modules.noarch : A meta package that installs all available Nginx
modules
nginx-mod-http-image-filter.aarch64 : Nginx HTTP image filter module
nginx-mod-http-perl.aarch64 : Nginx HTTP perl module
nginx-mod-http-xslt-filter.aarch64 : Nginx XSLT module
```

nginx-mod-mail.aarch64 : Nginx mail modules
nginx-mod-stream.aarch64 : Nginx stream modules

The dnf remove command removes a package from the system and its dependencies. Below is an excerpt of the **dnf remove httpd** command.

Package	Architecture			
Version ====================================		Repository	Size	
Removing: httpd	aarch64	2.		
4.37-43.module+el8.5.0+727+		@appstream	<mark>8</mark> .9 M	
Removing dependent packages		eapps ci eam	0.5 11	
mod_ssl	aarch64	1:		
2.4.37-43.module+el8.5.0+72		@appstream	274 k	
php	aarch64	7.		
4.19-1.module+el8.5.0+696+6		@appstream	4 .4 M	
python3-certbot-apache	noarch	1.		
22.0-1.el8		@epel	539 k	
Removing unused dependencie	S:			
apr	aarch64	1.		
6.3-12.el8		@appstream	299 k	
apr-util	aarch64	1.		
6.1-6.el8.1		@appstream	224 k	
apr-util-bdb	aarch64	1.		
6.1-6.el8.1		@appstream	67 k	
apr-util-openssl	aarch64	1.		
6.1-6.el8.1		@appstream	68 k	
augeas-libs	aarch64	1.		
12.0-6.el8		@baseos	1 .4 M	
httpd-filesystem	noarch	2.		
4.37-43.module+el8.5.0+727+		@appstream	400	
httpd-tools	aarch64	2.		

The dnf list command lists all the packages installed on the system and present in the repository. It accepts several parameters:

Parameter	Description
all	Lists the installed packages and then those available on the repositories.
available	Lists only the packages available for installation.
updates	Lists packages that can be upgraded.
obsoletes	Lists the packages made obsolete by higher versions available.
recent	Lists the latest packages added to the repository.

The dnf info command, as you might expect, provides detailed information about a package:

dnf info firewalld Last metadata expiration check: <mark>15</mark> :47:27 ago on Tue <mark>22</mark> Mar <mark>2022 05</mark> :49:42 PM CET.				
Installed Packages				
Name	: firewalld			
Version				
Release	: 7.el8			
Architecture				
Size				
Source	: firewalld-0.9.3-7.el8.src.rpm			
Repository				
From repo	: baseos			
Summary	: A firewall daemon with D-Bus interface providing a dynamic			
firewall				
URL	: http://www.firewalld.org			
License	: GPLv2+			
Description	: firewalld is a firewall service daemon that provides a dynamic			
customizable				
	: firewall with a D-Bus interface.			
Available Pa	ckages			
Name	: firewalld			
Version	: 0.9.3			
Release	: 7.el8_5.1			
Architecture	: noarch			
Size				
	: firewalld-0.9.3-7.el8_5.1.src.rpm			
Repository				
Summary	: A firewall daemon with D-Bus interface providing a dynamic			
firewall				
URL	: http://www.firewalld.org			

```
License : GPLv2+
Description : firewalld is a firewall service daemon that provides a dynamic
customizable : firewall with a D-Bus interface.
```

Sometimes you only know the executable you want to use but not the package that contains it, in this case you can use the command dnf provides */package_name which will search the database for you for the desired match.

Example of a search for the semanage command:

```
dnf provides */semanage
Last metadata expiration check: 1:12:29 ago on Wed 23 Mar 2022 10:40:43 AM CET.
libsemanage-devel-2.9-6.el8.aarch64 : Header files and libraries used to build
policy manipulation tools
Repo
            : powertools
Matched from:
Filename : /usr/include/semanage
policycoreutils-python-utils-2.9-16.el8.noarch : SELinux policy core python
utilities
Repo
            : baseos
Matched from:
Filename : /usr/sbin/semanage
            : /usr/share/bash-completion/completions/semanage
Filename
```

The dnf autoremove command does not need any parameters. Dnf takes care of searching for candidate packages for removal.

```
dnf autoremove
Last metadata expiration check: 0:24:40 ago on Wed 23 Mar 2022 06:16:47 PM CET.
Dependencies resolved.
Nothing to do.
Complete!
```

14.3.2 Other useful dnf options

Option	Description
repolist	Lists the repositories configured on the system.
grouplist	Lists available package collections.
clean	Removes temporary files.

The dnf repolist command lists the repositories configured on the system. By default, it lists only the enabled repositories but can be used with these parameters:

Parameter	Description
all	Lists all the repositories.
enabled	Default
disabled	Lists only disabled repositories.

Example:

dnf repolist	
repo id	repo name
appstream	Rocky Linux <mark>8</mark> -
AppStream	
baseos	Rocky Linux <mark>8</mark> - BaseOS
epel	Extra Packages for
Enterprise Linux <mark>8</mark> - aarch64	
epel-modular	Extra Packages for
Enterprise Linux Modular <mark>8</mark> - aarch64	
extras	Rocky Linux <mark>8</mark> - Extras
powertools	Rocky Linux <mark>8</mark> -
PowerTools	
rockyrpi	Rocky Linux <mark>8</mark> -
Rasperry Pi	

And an excerpt of the command with the --all flag.

dnf repolistall	
repo id	repo
name	
status	
appstream	Rocky Linux 8 -
AppStream	
enabled	
appstream-debug	Rocky Linux <mark>8</mark> - AppStream
- Source	disabled
appstream-source	Rocky Linux <mark>8</mark> - AppStream
- Source	disabled
baseos	Rocky Linux <mark>8</mark> -
BaseOS	
enabled	

Rocky Linux 8 - BaseOS baseos-debug Source disabled Rocky Linux 8 - BaseOS baseos-source disabled Source devel Rocky Linux 8 - Devel WARNING! FOR BUILDROOT AND KOJI USE disabled epel Extra Packages for Enterprise Linux 8 - aarch64 enabled epel-debuginfo Extra Packages for Enterprise Linux 8 - aarch64 - Debug disabled epel-modular Extra Packages for Enterprise Linux Modular 8 - aarch64 enabled epel-modular-debuginfo Extra Packages for Enterprise Linux Modular 8 - aarch64 - Debug disabled epel-modular-source Extra Packages for Enterprise Linux Modular 8 - aarch64 - Source . . .

And below is an excerpt from the list of disabled repositories.

dnf repolist --disabled repo id repo name appstream-debug Rocky Linux 8 - AppStream - Source Rocky Linux 8 - AppStream - Source appstream-source baseos-debug Rocky Linux 8 - BaseOS - Source baseos-source Rocky Linux 8 - BaseOS - Source devel Rocky Linux 8 - Devel WARNING! FOR BUILDROOT AND KOJI USE epel-debuginfo Extra Packages for Enterprise Linux 8 - aarch64 - Debug epel-modular-debuginfo Extra Packages for Enterprise Linux Modular 8 - aarch64 - Debug epel-modular-source Extra Packages for Enterprise Linux Modular 8 - aarch64 - Source epel-source Extra Packages for Enterprise Linux 8 - aarch64 - Source epel-testing Extra Packages for Enterprise Linux 8 - Testing - aarch64 . . .

Using the -v option enhances the list with a lot of additional information. Below you can see part of the result of the command.

```
dnf repolist -v
. . .
                  : powertools
Repo-id
                  : Rocky Linux 8 - PowerTools
Repo-name
Repo-revision
                   : 8.5
                  : [cpe:/o:rocky:rocky:8]: , , 8, L, R, c, i, k, n, o,
Repo-distro-tags
u, x, y
Repo-updated
                   : Wed 16 Mar 2022 10:07:49 PM CET
Repo-pkgs
                  : 1,650
Repo-available-pkgs: 1,107
Repo-size
                  : 6.4 G
Repo-mirrors
                   : https://mirrors.rockylinux.org/mirrorlist?
arch=aarch64&repo=PowerTools-8
                  : https://example.com/pub/rocky/8.8/PowerTools/x86_64/os/
Repo-baseurl
(30 more)
Repo-expire
                  : 172,800 second(s) (last: Tue 22 Mar 2022 05:49:24 PM CET)
                  : /etc/yum.repos.d/Rocky-PowerTools.repo
Repo-filename
. . .
```

i Using Groups

Groups are a collection of a set of packages (you can think of them as a virtual packages) that logically groups a set of applications to accomplish a purpose (a desktop environment, a server, development tools, etc.).

The dnf grouplist command lists all available groups.

```
dnf grouplist
Last metadata expiration check: 1:52:00 ago on Wed 23 Mar 2022 02:11:43 PM CET.
Available Environment Groups:
   Server with GUI
   Server
   Minimal Install
   KDE Plasma Workspaces
   Custom Operating System
Available Groups:
   Container Management
   .NET Core Development
   RPM Development Tools
   Development Tools
   Headless Management
   Legacy UNIX Compatibility
   Network Servers
```

Scientific Support Security Tools Smart Card Support System Tools Fedora Packager Xfce

The dnf groupinstall command allows you to install one of these groups.

```
dnf groupinstall "Network Servers"
Last metadata expiration check: 2:33:26 ago on Wed 23 Mar 2022 02:11:43 PM CET.
Dependencies resolved.
_
        Architecture Version
Package
                            Repository
                                   Size
_
Installing Groups:
Network Servers
Transaction Summarv
_
Is this ok [y/N]:
```

Note that it is good practice to enclose the group name in double quotes as without the command it will only execute correctly if the group name does not contain spaces.

So a dnf groupinstall Network Servers produces the following error.

```
dnf groupinstall Network Servers
Last metadata expiration check: 3:05:45 ago on Wed 23 Mar 2022 02:11:43 PM CET.
Module or Group 'Network' is not available.
Module or Group 'Servers' is not available.
Error: Nothing to do.
```

The corresponding command to remove a group is dnf groupremove "name group".

The dnf clean command cleans all caches and temporary files created by dnf. It can be used with the following parameters.

Parameters	Description
all	Removes all temporary files created for enabled repositories.
dbcache	Removes cache files for the repository metadata.
expire-cache	Remove the local cookie files.
metadata	Removes all the repositories metadata.
packages	Removes any cached packages.

14.3.3 How DNF works

The DNF manager relies on one or more configuration files to target the repositories containing the RPM packages.

These files are located in /etc/yum.repos.d/ and must end with .repo in order to be used by DNF.

Example:

/etc/yum.repos.d/Rocky-BaseOS.repo

Each .repo file consists of at least the following information, one directive per line.

Example:

```
[baseos] # Short name of the repository
name=Rocky Linux $releasever - BaseOS # Short name of the repository #Detailed
name
mirrorlist=http://mirrors.rockylinux.org/mirrorlist?arch=$basearch&repo=BaseOS-
$releasever # http address of a list or mirror
#baseurl=http://dl.rockylinux.org/$contentdir/$releasever/BaseOS/$basearch/os/
# http address for direct access
gpgcheck=1 # Repository requiring a signature
enabled=1 # Activated =1, or not activated =0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-rockyofficial # GPG public key path
```

By default, the enabled directive is absent which means that the repository is enabled. To disable a repository, you must specify the enabled=0 directive.

14.4 DNF modules

Modules were introduced in Rocky Linux 8 by the upstream. In order to use modules, the AppStream repository must exist and be enabled.

Package Confusion

The creation of module streams in the AppStream repository caused a lot of people confusion. Since modules are packaged within a stream (see our examples below), a particular package would show up in our RPMs, but if an attempt was made to install it without enabling the module, nothing would happen. Remember to look at modules if you attempt to install a package and it fails to find it.

14.4.1 What are modules

Modules come from the AppStream repository and contain both streams and profiles. These can be described as follows:

- module streams: A module stream can be thought of as a separate repository within the AppStream repository that contains different application versions. These module repositories contain the application RPMs, dependencies, and documentation for that particular stream. An example of a module stream in Rocky Linux 8 would be postgresql. If you install postgresql using the standard sudo dnf install postgresql you will get version 10. However, using modules, you can instead install versions 9.6, 12 or 13.
- module profiles: What a module profile does is take into consideration the use case for the module stream when installing the package. Applying a profile adjusts the package RPMs, dependencies and documentation to account for the module's use. Using the same postgresql stream in our example, you can apply a profile of either "server" or "client". Obviously, you do not need the same packages installed on your system if you are just going to use postgresql as a client to access a server.

14.4.2 Listing modules

You can obtain a list of all modules by executing the following command:

dnf module list

This will give you a long list of the available modules and the profiles that can be used for them. The thing is you probably already know what package you are interested in, so to find out if there are modules for a particular package, add the package name after "list". We will use our postgresql package example again here:

```
dnf module list postgresql
```

This will give you output that looks like this:

Rocky Linux <mark>8</mark>	- AppStream	
Name	Stream	
Profiles		
Summary		
postgresql	<mark>9</mark> .6	client, server
[d]	PostgreSQL server	and client module
postgresql	10 [d]	client, server
[d]	PostgreSQL server	and client module
postgresql	12	client, server
[d]	PostgreSQL server	and client module
postgresql	13	client, server
[d]	PostgreSQL server	and client module

Notice in the listing the "[d]". This means that this is the default. It shows that the default version is 10 and that regardless of which version you choose, if you do not specify a profile, then the server profile will be the profile used, as it is the default as well.

14.4.3 Enabling Modules

Using our example postgresql package, let's say that we want to enable version 12. To do this, you simply use the following:

dnf module enable postgresql:12

Here the enable command requires the module name followed by a ":" and the stream name.

To verify that you have enabled postgresql module stream version 12, use your list command again which should show you the following output:

Rocky Linux	<mark>8</mark> - AppStream			
Name	Stream			
Profiles				
Summary				
postgresql	<mark>9</mark> .6		client,	server
[d]	PostgreSQL server a	nd client m	nodule	
postgresql	10 [d]		client,	server
[d]	PostgreSQL server a	nd client m	nodule	
postgresql	12 [e]		client,	server
[d]	PostgreSQL server a	nd client m	nodule	
postgresql	13		client,	server
[d]	PostgreSQL server a	nd client n	nodule	

Here we can see the "[e]" for "enabled" next to stream 12, so we know that version 12 is enabled.

14.4.4 Installing packages from the module stream

Now that our module stream is enabled, the next step is to install postgresql, the client application for the postgresql server. This can be achieved by running the following command:

dnf install postgresql

Which should give you this output:

Package	Architecture		
Version			
Repository	Size		
		==========	
Installing group/m	odule packages:		
postgresql	x86_64	12.	
12-1.module+el8.6.	0+1049+f8fc4c36	appstream	1.
5 M			
Installing depende	ncies:		
libpq	x86_64	13.	
5-1.el8		appstream	
197 k			

Install 2 Packages
Total download size: 1.7 M
Installed size: 6.1 M
Is this ok [y/N]:

After approving by typing "y" you installed the application.

14.4.5 Installing packages from module stream profiles

It's also possible to directly install packages without even having to enable the module stream! In this example, let's assume that we only want the client profile applied to our installation. To do this, we simply enter this command:

dnf install postgresql:12/client

Which should give you this output:

			==
		======	
Package	Architecture		
Version			
Repository	Size		
			==
		======	
Installing group/module pac	kages:		
postgresql	x86_64	12.	
12-1.module+el8.6.0+1049+f8	3fc4c36	appstream 1	
5 M			
Installing dependencies:			
libpq	x86_64	13.	
5-1.el8		appstream	
197 k			
Installing module profiles:			
postgresql/client			
Enabling module streams:			
postgresql		12	
Transaction Summary			
			==
		======	
Install 2 Packages			

```
Total download size: 1.7 M
Installed size: 6.1 M
Is this ok [y/N]:
```

Answering "y" to the prompt will install everything you need to use postgresql version 12 as a client.

14.4.6 Module Removal and Reset or Switch-To

After you install, you may decide that for whatever reason, you need a different version of the stream. The first step is to remove your packages. Using our example <code>postgresql</code> package again, we would do this with:

```
dnf remove postgresql
```

This will display similar output as the install procedure above, except it will be removing the package and all of its dependencies. Answer "y" to the prompt and hit enter to uninstall postgresql.

Once this step is complete, you can issue the reset command for the module using:

dnf module reset postgresql

Which will give you output like this:

```
Dependencies resolved.
______
Package
          Architecture
Version
          Repository
                    Size
Disabling module profiles:
postgresgl/
client
Resetting modules:
postgresql
Transaction Summary
```

```
Is this ok [y/N]:
```

Answering "y" to the prompt will then reset postgresql back to the default stream with the stream that we had enabled (12 in our example) no longer enabled:

```
Rocky Linux 8 - AppStream
Name
                            Stream
Profiles
Summary
                                                    client, server
postgresql
                            9.6
[d]
                    PostgreSQL server and client module
postgresql
                            10 [d]
                                                    client, server
                    PostgreSQL server and client module
[d]
postgresql
                                                    client, server
                            12
                    PostgreSQL server and client module
[d]
postgresql
                            13
                                                    client, server
                    PostgreSQL server and client module
[d]
```

Now you can use the default.

You can also use the switch-to sub-command to switch from one enabled stream to another. Using this method not only switches to the new stream, but installs the needed packages (either downgrade or upgrade) without a separate step. To use this method to enable postgresql stream version 13 and use the "client" profile, you would use:

dnf module switch-to postgresql:13/client

14.4.7 Disable a module stream

There may be times when you wish to disable the ability to install packages from a module stream. In the case of our postgresql example, this could be because you want to use the repository directly from PostgreSQL so that you could use a newer version (at the time of this writing, versions 14 and 15 are available from this repository). Disabling a module stream, makes installing any of those packages impossible without first enabling them again.

To disable the module streams for postgresql simply do:

dnf module disable postgresql

And if you list out the postgresql modules again, you will see the following showing all postgresql module versions disabled:

```
Rocky Linux 8 - AppStream
Name
                            Stream
Profiles
Summary
postgresql
                            9.6 [X]
                                                     client, server
                   PostgreSQL server and client module
[d]
                                                     client, server
postgresql
                            10 [d][x]
[d]
                   PostgreSQL server and client module
postgresql
                           12 [X]
                                                     client, server
                   PostgreSQL server and client module
[d]
                                                     client, server
postgresql
                            13 [x]
                   PostgreSQL server and client module
[d]
```

14.5 The EPEL repository

14.5.1 What is EPEL and how is it used?

EPEL (Extra Packages for Enterprise Linux) is an open-source and free community-based repository maintained by the EPEL Fedora Special Interest Group that provides a set of additional packages for RHEL (and CentOS, Rocky Linux, and others) from the Fedora sources.

It provides packages that are not included in the official RHEL repositories. These are not included because they are not considered necessary in an enterprise environment or deemed outside the scope of RHEL. We must not forget that RHEL is an enterprise class distribution, and desktop utilities or other specialized software may not be a priority for an enterprise project.

14.5.2 Installation

Installation of the necessary files can be easily done with the package provided by default from Rocky Linux.

If you are behind an internet proxy:

export http_proxy=http://172.16.1.10:8080

Then:

dnf install epel-release

Once installed you can check that the package has been installed correctly with the command dnf info.

dnf info epel-release Last metadata expiration check: 1:30:29 ago on Thu 24 Mar 2022 09:36:42 AM CET.						
Installed Packages						
Name	: epel-release					
Version	: 8					
Release	: 1 4.el8					
Architecture	: noarch					
Size	: 32 k					
Source	: epel-release-8-14.el8.src.rpm					
Repository	: @System					
From repo	: epel					
Summary	ummary : Extra Packages for Enterprise Linux repository configuration					
URL	: http://download.fedoraproject.org/pub/epel					
License	: GPLv2					
Description	: This package contains the Extra Packages for Enterprise Linux					
	: (EPEL) repository GPG key as well as configuration for yum.					

The package, as you can see from the package description above, does not contain executables, libraries, etc... but only the configuration files and GPG keys for setting up the repository.

Another way to verify the correct installation is to query the rpm database.

rpm -qa | grep epel
epel-release-8-14.el8.noarch

Now you need to run an update to let dnf recognize the repository. You will be asked to accept the GPG keys of the repositories. Clearly, you have to answer YES in order to use them.

dnf update

Once the update is complete you can check that the repository has been configured correctly with the dnf repolist command which should now list the new repositories.

dnf repolistrepo idrepo name...epelExtra Packages for Enterprise Linux 8 - aarch64epel-modularExtra Packages for Enterprise Linux Modular 8 - aarch64...

The repository configuration files are located in /etc/yum.repos.d/.

ll /etc/yum.repos.d/ | grep epel -rw-r--r-. 1 root root 1485 Jan 31 17:19 epel-modular.repo -rw-r--r-. 1 root root 1422 Jan 31 17:19 epel.repo -rw-r--r-. 1 root root 1584 Jan 31 17:19 epel-testing-modular.repo -rw-r--r-. 1 root root 1521 Jan 31 17:19 epel-testing.repo

And below we can see the contents of the file epel.repo.

```
[epel]
name=Extra Packages for Enterprise Linux $releasever - $basearch
# It is much more secure to use the metalink, but if you wish to use a local
mirror
# place its address here.
#baseurl=https://download.example/pub/epel/$releasever/Everything/$basearch
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-
$releasever&arch=$basearch&infra=$infra&content=$contentdir
enabled=1
gpgcheck=1
countme=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-8
[epel-debuginfo]
name=Extra Packages for Enterprise Linux $releasever - $basearch - Debug
# It is much more secure to use the metalink, but if you wish to use a local
mirror
# place its address here.
#baseurl=https://download.example/pub/epel/$releasever/Everything/$basearch/
debug
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-debug-
$releasever&arch=$basearch&infra=$infra&content=$contentdir
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-8
```

```
gpgcheck=1
[epel-source]
name=Extra Packages for Enterprise Linux $releasever - $basearch - Source
# It is much more secure to use the metalink, but if you wish to use a local
mirror
# place it's address here.
#baseurl=https://download.example/pub/epel/$releasever/Everything/source/tree/
metalink=https://mirrors.fedoraproject.org/metalink?repo=epel-source-
$releasever&arch=$basearch&infra=$infra&content=$contentdir
enabled=0
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-8
gpgcheck=1
```

14.5.3 Using EPEL

At this point, once configured, we are ready to install the packages from EPEL. To start, we can list the packages available in the repository with the command:

dnf --disablerepo="*" --enablerepo="epel" list available

And an excerpt of the command

dnfdisablerepo="*"enablerepo	="epel" list available le	SS
Last metadata expiration check: 1:		
Available Packages		
3proxy.aarch64		0.
8.13-1.el8	epel	
AMF-devel.noarch		1.
4.23-2.el8	epel	
AMF-samples.noarch	·	1.
4.23-2.el8	epel	
AusweisApp2.aarch64		1.
22.3-1.el8	epel	
AusweisApp2-data.noarch		1.
22.3-1.el8	epel	
AusweisApp2-doc.noarch		1.
22.3-1.el8	epel	
BackupPC.aarch64		4.
4.0-1.el8	epel	
BackupPC-XS.aarch64		0.
62-1.el8	epel	
BibTool.aarch64		2.
68-1.el8	epel	
CCfits.aarch64		2.

5-14.el8	epel	
CCfits-devel.aarch64		2.
5-14.el8	epel	

From the command we can see that to install from EPEL we must force **dnf** to query the requested repository with the options --disablerepo and --enablerepo, this is because otherwise a match found in other optional repositories (RPM Fusion, REMI, ELRepo, etc.) could be newer and therefore have priority. These options are not necessary if you have only installed EPEL as an optional repository because the packages in the repository will never be available in the official ones. At least in the same version!

A Support consideration

One aspect to consider regarding support (updates, bug fixes, security patches) is that EPEL packages have no official support from RHEL and technically their life could last the space of a development of Fedora (six months) and then disappear. This is a remote possibility but one to consider.

So, to install a package from the EPEL repositories you would use:

```
dnf --disablerepo="*" --enablerepo="epel" install nmon
Last metadata expiration check: 2:01:36 ago on Fri 25 Mar 2022 04:28:04 PM CET.
Dependencies resolved.
_____
Package
                  Architecture
Version
                      Repository
Size
_____
Installing:
nmon
                  aarch64
16m-1.el8
                      epel
71 k
Transaction Summary
_____
                         _____
_____
Install 1 Package
Total download size: 71 k
Installed size: 214 k
Is this ok [y/N]:
```

14.5.4 Conclusion

EPEL is not an official repository for RHEL, but it can be useful for administrators and developers who work with RHEL or derivatives and need some utilities prepared for RHEL from a source they can feel confident about.

14.6 DNF Plugins

The dnf-plugins-core package adds plugins to dnf that will be useful for managing your repositories.

```
Note
See more informations here: https://dnf-plugins-core.readthedocs.io/en/latest/index.html
```

Install the package on your system:

dnf install dnf-plugins-core

Not all plugins will be presented here but you can refer to the package documentation for a complete list of plugins and detailed information.

14.6.1 config-manager plugin

Manage DNF options, add repos, or disable them.

Examples:

• Download a .repo file and use it:

dnf config-manager --add-repo https://packages.centreon.com/ui/native/rpmstandard/23.04/el8/centreon-23.04.repo

• You can also set an url as a base url for a repo:

dnf config-manager --add-repo https://repo.rocky.lan/repo

• Enable or disable one or more repos:

dnf config-manager --set-enabled epel centreon
dnf config-manager --set-disabled epel centreon

• Add a proxy to your config file:

dnf config-manager --save --setopt=*.proxy=http://proxy.rocky.lan:3128/

14.6.2 copr plugin

copr is an automatic rpm forge, providing a repo with built packages.

• Activate a copr repo:

copr enable xxxx

14.6.3 download plugin

Download rpm package instead of installing it:

dnf download ansible

If you just want to obtain the remote location url of the package:

dnf download --url ansible

Or if you want to also download the dependencies:

dnf download --resolv --alldeps ansible

14.6.4 needs-restarting plugin

After running a dnf update, the running processes will continue to run but with the old binaries. In order to take into account the code changes and especially the security updates, they have to be restarted.

The needs-restarting plugin will allow you to detect processes that are in this case.

dnf needs-restarting [-u] [-r] [-s]

Options	Description
- u	Only consider processes belonging to the running user.
- r	to check if a reboot may be required.
- S	to check if services need restarting.
-s -r	to do both in one run.

14.6.5 versionlock plugin

Sometimes it is useful to protect packages from all updates or to exclude certain versions of a package (because of known problems for example). For this purpose, the versionlock plugin will be of great help.

You need to install an extra package:

dnf install python3-dnf-plugin-versionlock

Examples:

• Lock the ansible version:

dnf versionlock add ansible
Adding versionlock on: ansible-0:6.3.0-2.el9.*

• List locked packages:

dnf versionlock list
ansible-0:6.3.0-2.el9.*

All of the examples in this document use root actions, with ordinary users actions commented separately. In the markdown code block, the command description will be indicated with # on the previous line.

15. Review basic permissions

It is well known that the basic permissions of GNU/Linux can be viewed using \mbox{ls} - l:

Sh	ell >	ls -	l						
-	rwx	r-x	r-x	1	root	root	1358	Dec <mark>31 14</mark> :50	anaconda-ks.cfg
\downarrow	\downarrow	\downarrow	\downarrow	↓	\downarrow	\downarrow	\downarrow	\downarrow	\downarrow
1	2	3	4	5	6	7	8	9	10

Their meanings are as follows:

Part	Description
1	File type indicates that this is an ordinary file. Seven file types will be introduced later.
2	Permissions of owner user, the meaning of rwx respectively means: read, write, execute.
3	Permissions of the owner group.
4	Permissions of other users.
5	Number of subdirectories (. and included). For a file, it represents the number of hard links, and 1 represents itself.
6	Name of the owner user.
7	Name of the owner group.
8	For files, it shows the size of the file. For directories, it shows the fixed value of 4096 bytes occupied by the file naming. To calculate the total size of a directory, use du -sh
9	Last modified date.
10	The name of the file (or directory).

15.1 Seven file types

File types	Description
-	Represents an ordinary file. Including plain text files (ASCII); binary files (binary); data format files (data); various compressed files.
d	Represents a directory file. By default, there is one in every directory $\ . \ $ and $\ \ldots$
b	Block device file. Including all kinds of hard drives, USB drives and so on.
С	Character device file. Interface device of serial port, such as mouse, keyboard, etc.
S	Socket file. It is a file specially used for network communication.
р	Pipe file. It is a special file type, the main purpose is to solve the errors caused by multiple programs accessing a file at the same time. FIFO is the abbreviation of first-in-first-out.
1	Soft link files, also called symbolic link files, are similar to shortcuts in Windows. Hard link file, also known as physical link file.

15.2 The meaning of basic permissions

For file:

Digital representation	Permissions	Description
4	r(read)	Indicates that you can read this file. You can use commands such as <code>cat</code> , head , more , less , <code>tail</code> , etc.
2	w(write)	Indicates that the file can be modified. Commands such as vim can be used.
1	x(execution)	Permissions for executable files (such as scripts or binaries).

For directory:

Digital representation	Permissions	Description
4	r(read)	Indicates that the contents of the directory can be listed, such as $ls - l$.
2	w(write)	Indicates that you can create, delete, and rename files in this directory, such as commands mkdir, touch, rm, etc.
1	x(execute)	Indicates that you can enter the directory, such as the command $\ \mbox{cd}$.

i Info

For directories, ${\boldsymbol r}$ and ${\boldsymbol x}$ permissions usually appear at the same time.

15.3 Special authority

In GNU/Linux, in addition to the basic permissions mentioned above, there are also some special permissions, which we will introduce one by one.

15.3.1 ACL permissions

What is ACL? ACL(Access Control List), the purpose is to solve the problem that the three identities under Linux can not meet the needs of resource permission allocation.

For example, the teacher gives lessons to the students, and the teacher creates a directory under the root directory of OS. Only the students in this class are allowed to upload and download, and others are not allowed. At this point, the permissions for the directory are 770. One day, a student from another school came to listen to the teacher, how should permissions be assigned? If you put this student in the **owner group**, he will have the same permissions as the students in this class - **rwx**. If the student is put into the **other users**, he will not have any permissions. At this time, the basic permission allocation cannot meet the requirements, and you need to use ACL.

There is a similar feature in the Windows operating system. For example, to assign permissions to a user for a file, for a user-defined directory/file, **right-click** ---> **Properties** ---> **Security** ---> **Edit** ---> **Add** ---> **Advanced** ---> **Find now**, find the corresponding user/group ---> assign specific permissions ---> **apply**, and complete.

The same is true of GNU/Linux: add the specified user/group to the file/directory and grant the appropriate permissions to complete the ACL permission assignment.

How do I enable an ACL? You need to find the file name of the device where the mount point is located and its partition number. For example, on my machine, you could do something like this:

Shell > df -hT						
Filesystem	Туре	Size	Used	Avail	Use%	Mounted on
devtmpfs	devtmpfs	<mark>3</mark> .8G	Θ	<mark>3</mark> .8G	<mark>0</mark> %	/dev
tmpfs	tmpfs	<mark>3</mark> .8G	Θ	<mark>3</mark> .8G	<mark>0</mark> %	/dev/shm
tmpfs	tmpfs	<mark>3</mark> .8G	<mark>8</mark> .9M	<mark>3</mark> .8G	1 %	/run

```
tmpfs
              tmpfs
                        3.8G 0 3.8G 0% /sys/fs/cgroup
/dev/nvme0n1p2 ext4
                         47G
                               11G
                                     35G
                                          24% /
/dev/nvme0n1p1 xfs
                                          19% /boot
                       1014M 187M 828M
tmpfs
                                           0% /run/user/0
              tmpfs
                        774M
                                 0 774M
Shell > dumpe2fs /dev/nvme0n1p2 | head -n 10
dumpe2fs 1.45.6 (20-Mar-2020)
Filesystem volume name:
                         <none>
Last mounted on:
                         /
                         c8e6206d-2892-4c22-a10b-b87d2447a885
Filesystem UUID:
Filesystem magic number: 0xEF53
Filesystem revision #:
                         1 (dynamic)
Filesystem features: has_journal ext_attr resize_inode dir_index filetype
needs_recovery extent 64bit flex_bg sparse_super large_file huge_file
dir_nlink extra_isize metadata_csum
                         signed_directory_hash
Filesystem flags:
Default mount options:
                         user xattr acl
Filesystem state:
                         clean
Errors behavior:
                         Continue
```

When you see the line "**Default mount options: user_xattr acl**", it indicates that ACL has been enabled. If it is not enabled, you can also enable it temporarily -- mount -o remount, acl /. It can also be enabled permanently:

```
Shell > vim /etc/fstab
UUID=c8e6206d-2892-4c22-a10b-b87d2447a885 / ext4 defaults,acl 1 1
Shell > mount -o remount /
# or
Shell > reboot
```

Viewing and setting of ACL

To view ACL, you need to use the getfacle command -- getfacle FILE_NAME

If you want to set ACL permissions, you need to use the setfacl command.

Shell > setfacl <option> <FILE_NAME>

Option	Description
-m	modify the current $ACL(s)$ of file(s)
-X	remove entries from the $ACL(s)$ of file(s)
-b	remove all extended ACL entries
-d	operations apply to the default ACL
-k	remove the default ACL
-R	recurse into subdirectories

Use the teacher's example mentioned at the beginning of the article to illustrate the use of ACL.

```
# The teacher is the root user
Shell > groupadd class1
Shell > mkdir /project
Shell > chown root:class1 /project
Shell > chmod 770 /project
Shell > ls -ld /project/
drwxrwx--- 2 root class1 4096 Jan 12 12:58 /project/
# Put the students in the class into the class1 group
Shell > useradd frank
Shell > passwd frank
Shell > useradd aron
Shell > passwd aron
Shell > gpasswd -a frank class1
Shell > gpasswd -a aron class1
# A student from another school came to listen to the teacher
Shell > useradd tom
Shell > passwd tom
# If it is a group, "u" here should be replaced by "g"
Shell > setfacle -m u:tom:rx /project
# "+" sign is added in the output message
Shell > ls -ld /project/
drwxrwx---+ 2 root class1 4096 Jan 12 12:58 /project/
Shell > getfacl -p /project/
# file: /project/
# owner: root
# group: class1
user::rwx
```

```
user:tom:r-x
group::rwx
mask::rwx
other::---
```

Maximum valid permissions of ACL

When using the getfacl command, what does the "mask:: rwx" in the output message mean? The **mask** is used to specify the maximum valid permissions. The permissions given to the user are not real permissions, the real permissions can only be obtained by using the "logical and" of the user's permissions and mask permissions.

Info		
Logical and" means: that if all are	true, the result is true; if the	re is one false, the
Permissions set by users	Mask permissions	Result
r	r	r
r	-	-
-	r	-
-	-	-
🕽 Info		

Because the default mask is rwx, for any user's ACL permissions, the result is their own permissions.

You can also adjust mask permissions:

```
Shell > setfacl -m u:tom:rwx /project
Shell > setfacl -m m:rx /project
Shell > getfacl -p /project/
# file: project/
# owner: root
# group: class1
user::rwx
user:tom:rwx #effective:r-x
group::rwx #effective:r-x
mask::r-x
other::---
```

Delete ACL permission

Delete the ACL permissions of user/group in the specified directory Shell > setfacl -x u:USER_NAME FILE_NAME Shell > setfacl -x g:GROUP_NAME FILE_NAME # Removes all ACL permissions for the specified directory Shell > setfacl -b FILE_NAME

Default and recursion of ACL permissions

What is the recursion of ACL permissions? For ACL permissions, this means that when the parent directory sets ACL permissions, all subdirectories and sub-files will have the same ACL permissions.

```
• Info
Recursion is suitable for files/directories that already exist in the directory.
```

Look at the following example:

```
Shell > setfacl -m m:rwx /project
Shell > setfacl -m u:tom:rx /project
Shell > cd /project
Shell > touch file1 file2
# Because there is no recursion, the file here does not have ACL permission.
Shell > ls -l
-rw-r--r-- 1 root root 0 Jan 12 14:35 file1
-rw-r--r-- 1 root root 0 Jan 12 14:35 file2
Shell > setfacl -m u:tom:rx -R /project
Shell > ls -l /project
-rw-r-xr--+ 1 root root 0 Jan 12 14:35 file1
-rw-r-xr--+ 1 root root 0 Jan 12 14:35 file1
```

Now there is a question: if I create a new file in this directory, does it have ACL permission? The answer is no, because the newly created file is after the command setfacl-m u:tom:rx -R /project is executed.

```
Shell > touch /project/file3
Shell > ls -l /project/file3
-rw-r--r-- 1 root root 0 Jan 12 14:52 /project/file3
```

If you want the newly created directory/file to also have ACL permissions, you need to use default ACL permissions.

```
Shell > setfacl -m d:u:tom:rx /project
Shell > cd /project && touch file4 && ls -l
-rw-r-xr--+ 1 root root 0 Jan 12 14:35 file1
-rw-r-xr--+ 1 root root 0 Jan 12 14:35 file2
-rw-r--r-- 1 root root 0 Jan 12 14:52 file3
-rw-rw----+ 1 root root 0 Jan 12 14:59 file4
Shell > getfacl -p /project
# file: /project
# owner: root
# group: class1
user::rwx
user:tom:r-x
group::rwx
mask::rwx
other::---
default:user::rwx
default:user:tom:r-x
default:group::rwx
default:mask::rwx
default:other::---
```

i Info

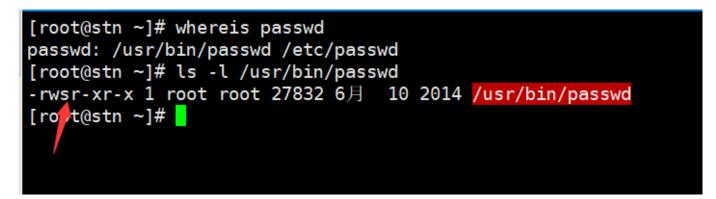
The default and recursion of using ACL permissions require that the operating object of the command be a directory! If the operation object is a file, an error prompt will be output.

15.3.2 SetUID

The role of "SetUID":

- Only executable binaries can set SUID permissions.
- The executor of the command should have x permission to the program.
- The executor of the command obtains the identity of the owner of the program file when executing the program.
- The identity change is only valid during execution, and once the binary program is finished, the executor's identity is restored to the original identity.

Why does GNU/Linux need such strange permissions? Take the most common passwd command as an example:



As you can see, the ordinary users only has r and x, but the owner's x becomes s, proving that the passwd command has SUID permissions.

It is well known that the ordinary users (uid ≥ 1000) can change his own password. The real password is stored in the **/etc/shadow** file, but the permission of the shadows file is 000, and the ordinary users does not have any permissions.

```
Shell > ls -l /etc/shadow
----- 1 root root 874 Jan 12 13:42 /etc/shadow
```

Since the ordinary users can change their password, they must have written the password to the **/etc/shadow** file. When an ordinary user executes the passwd command, it will temporarily change to the owner of the file -- **root**. For **shadow** file, **root** can not be restricted by permissions. This is why passwd command needs SUID permission.

As mentioned earlier, basic permissions can be represented by numbers, such as 755, 644, and so on. SUID is represented by **4**. For executable binaries, you can set permissions like this -- **4755**.

```
# Set SUID permissions
Shell > chmod 4755 FILE_NAME
# or
Shell > chmod u+s FILE_NAME
# Remove SUID permission
Shell > chmod 755 FILE_NAME
# or
Shell > chmod u-s FILE NAME
```

🛕 Warning

When the owner of an executable binary file/program does not have **x**, the use of capital **S** means that the file cannot use SUID permissions.



Because SUID can temporarily change the Ordinary users to root, you need to be especially careful with files with this permission when maintaining the server. You can find files with SUID permissions by using the following command:

Shell > find / -perm -4000 -a -type f -exec ls -l $\{\}$ \;

15.3.3 SetGID

The role of "SetGID":

- Only executable binaries can set SGID permissions.
- The executor of the command should have x permission to the program.
- The executor of the command obtains the identity of the owner group of the program file when executing the program.
- The identity change is only valid during execution, and once the binary program is finished, the executor's identity is restored to the original identity.

Take the locate command for example:

```
Shell > rpm -ql mlocate
/usr/bin/locate
...
/var/lib/mlocate/mlocate.db
Shell > ls -l /var/lib/mlocate/mlocate.db
-rw-r---- 1 root slocate 4151779 1 14 11:43 /var/lib/mlocate/mlocate.db
Shell > ll /usr/bin/locate
-rwx--s--x. 1 root slocate 42248 4 12 2021 /usr/bin/locate
```

The locate command uses the **mlocate.db** database file to quickly search for files.

Because the locate command has SGID permission, when the executor (ordinary users) executes the locate command, the owner group is switched to **slocate**. slocate has r permission for the **/var/lib/mlocate/mlocate.db** file.

The SGID is indicated by the number **2**, so the locate command has a permission of 2711.

```
# Set SGID permissions
Shell > chmod 2711 FILE_NAME
# or
Shell > chmod g+s FILE_NAME
# Remove SGID permission
Shell > chmod 711 FILE_NAME
```

```
# or
Shell > chmod g-s FILE_NAME
```

🛕 Warning

When the owner group of an executable binary file/program does not have \mathbf{x} , use uppercase \mathbf{S} to indicate that the file's SGID permissions cannot be used correctly.

```
# Suppose this is an executable binary file
Shell > touch sgid
Shell > chmod 2741 sgid
Shell > ls -l sgid
-rwxr-S--x 1 root root 0 Jan 14 12:11 sgid
```

SGID can be used not only for executable binary file/program, but also for directories, but it is rarely used.

- Ordinary users must have rwx permissions on the directory.
- For files created by ordinary users in this directory, the default owner group is the owner group of the directory.

For example:

🛕 Warning

Because SGID can temporarily change the owner group of ordinary users to root, you need to pay special attention to the files with this permission when maintaining the server. You can find files with SGID permissions through the following command:

Shell > find / -perm -2000 -a -type f -exec ls -l {} \;

15.3.4 Sticky BIT

The role of "Sticky BIT":

- Only valid for directory.
- Ordinary users have w and x permissions on this directory.
- If there is no Sticky Bit, ordinary users with w permission can delete all files in this directory (including files created by other users). Once the directory is given SBIT permission, only root user can delete all files. Even if ordinary users have w permission, they can only delete files created by themselves (files created by other users cannot be deleted).

SBIT is represented by the number **1**.

Can the file or directory have **7755** permission? No, they are aimed at different objects. SUID is for executable binary files; SGID is used for executable binaries and directories; SBIT is only for directories. That is, you need to set these special permissions according to different objects.

The directory **/tmp** has SBIT permission. The following is an example:

```
# The permissions of the /tmp directory are 1777
Shell > ls -ld /tmp
drwxrwxrwt. 8 root root 4096 Jan 14 12:50 /tmp
Shell > su - tom
Shell > cd /tmp && touch tom_file1
Shell > cd /tmp && touch tom_file1
Shell > su - jack
Shell(jack) > cd /tmp && rm -rf tom_file1
rm: cannot remove 'tom_file1': Operation not permitted
Shell(jack) > exit
# The file has been deleted
Shell > su - tom
Shell(tom) > rm -rf /tmp/tom_file1
```

i Info

root (uid=0) users are not restricted by the permissions of SUID, SGID, and SBIT.

15.3.5 chattr

The function of chattr permission: it is used to protect important files or directories in the system from being deleted by misoperation.

```
Usage of the chattr command -- chattr [ -RVf ] [ -v version ] [ -p project ]
[ mode ] files...
```

The format of a symbolic mode is +-=[aAcCdDeFijPsStTu].

- "+" means to increase permissions;
- "-" means to reduce permissions;
- "=" means equal to a permission.

The most commonly used permissions (also called attribute) are **a** and **i**.

Description of attribute i

	Delete	Free modification	Append file content	View	Create file
file	×	×	×	\checkmark	-
directory	x (Directory and files under the directory)	√ (Files in the directory)	√ (Files in the directory)	√ (Files in the directory)	x

Examples for files:

```
-bash: /tmp/filei: Operation not permitted
Shell > cat /tmp/filei
123
```

Examples for directories:

```
Shell > mkdir /tmp/diri
Shell > cd /tmp/diri && echo "qwer" > f1
Shell > chattr +i /tmp/diri
Shell > lsattr -ad /tmp/diri
----i----e---- /tmp/diri
Shell > rm -rf /tmp/diri
rm: cannot remove '/tmp/diri/f1': Operation not permitted
# Allow modification
Shell > vim /tmp/diri/f1
qwer-tom
Shell > echo "jim" >> /tmp/diri/f1
Shell > cat /tmp/diri/f1
gwer-tom
jim
Shell > touch /tmp/diri/file2
touch: settng time of '/tmp/diri/file2': No such file or directory
```

Remove the i attribute from the above example:

Shell > chattr -i /tmp/filei /tmp/diri

Description of attribute a

	Delete	Free modification	Append file content	View	Create file
file	×	×	\checkmark	\checkmark	-
directory	x (Directory and files under the directory)	x (Files in the directory)	√ (Files in the directory)	√ (Files in the directory)	\checkmark

Examples for files:

```
Shell > touch /etc/tmpfile1
Shell > echo "zxcv" > /etc/tmpfile1
Shell > chattr +a /etc/tmpfile1
Shell > lsattr -a /etc/tmpfile1
-----a------ /etc/tmpfile1
Shell > rm -rf /etc/tmpfile1
rm: cannot remove '/etc/tmpfile1': Operation not permitted
# Cannot be modified freely
Shell > vim /etc/tmpfile1
Shell > echo "new line" >> /etc/tmpfile1
Shell > cat /etc/tmpfile1
zxcv
new line
```

Examples for directories:

```
Shell > mkdir /etc/dira
Shell > cd /etc/dira && echo "asdf" > afile
Shell > chattr +a /etc/dira
Shell > lsattr -ad /etc/dira
-----a-----e----- /etc/dira/
Shell > rm -rf /etc/dira
rm: cannot remove '/etc/dira/afile': Operation not permitted
# Free modification is not allowed
Shell > vim /etc/dira/afile
asdf
Shell > echo "new line" >> /etc/dira/afile
Shell > cat /etc/dira/afile
asdf
new line
# Allow creation of new files
Shell > touch /etc/dira/newfile
```

Remove the a attribute from the above example:

Shell > chattr -a /etc/tmpfile1 /etc/dira/

Question

What happens when I set the ai attribute on a file? You cannot do anything with the file other than to view it. What about the directory? Allowed are: free modification, appending file contents, and viewing. Disallowed: delete and create files.

15.3.6 sudo

The role of "sudo":

- Through the root user, assign the commands that can only be executed by the root user (uid=0) to ordinary users for execution.
- The operation object of "sudo" is the system command.

We know that only the administrator root has permission to use the commands under **/sbin/** and **/usr/sbin/** in the GNU/Linux directory. Generally speaking, a company has a team to maintain a set of servers. This set of servers can refer to a single computer room in one geographic location, or it can refer to a computer room in multiple geographical locations. The team leader uses the permissions of the root user, and other team members may only have the permissions of the ordinary user. As the person in charge has a lot of work, there is no time to maintain the daily work of the server, most of the work needs to be maintained by ordinary users. However, ordinary users have many restrictions on the use of commands, and at this point, you need to use sudo permissions.

To grant permissions to ordinary users, **you must use the root user (uid=0)**.

You can empower ordinary users by using the visudo command, what you're actually changing is the **/etc/sudoers** file.

```
Shell > visudo
...
88 Defaults secure_path = /sbin:/bin:/usr/sbin:/usr/bin
89
90 ## Next comes the main part: which users can run what software on
91 ## which machines (the sudoers file can be shared between multiple
92 ## systems).
93 ## Syntax:
94 ##
```

```
95 ## user MACHINE=COMMANDS
96 ##
97 ## The COMMANDS section may have other options added to it.
98 ##
99 ## Allow root to run any commands anywhere
100 root
           ALL=(ALL)
                            ALL
     Ļ
            \downarrow \downarrow
                             ↓
     1
            2
                 3
                             4
. . .
```

Part	Description
1	User name or owner group name. Refers to which user/group is granted permissions. If it is an owner group, you need to write "%", such as %root .
2	Which machines are allowed to execute commands. It can be a single IP address, a network segment, or ALL.
3	Indicates which identities can be transformed into.
4	The authorized command, which needs to be represented by an absolute path.

For example:

```
Shell > visudo
...
101 tom ALL=/sbin/shutdown -r now
...
# You can use the "-c" option to check for errors in /etc/sudoers writing.
Shell > visudo -c
Shell > su - tom
# View the available sudo commands.
Shell(tom) > sudo -l
# To use the available sudo command, ordinary users need to add sudo before the
command.
Shell(tom) > sudo /sbin/shutdown -r now
```

If your authorization command is /sbin/shutdown, it means that authorized users can use any of the options of the command.

🛕 Warning

Because sudo is a "ultra vires" operation, you need to be careful when dealing with **/etc/sudoers** files!

https://docs.rockylinux.org/