Linux

Nguyen Quan Ba Hong[1]

October 28, 2022

[1]Independent Researcher, Ben Tre City, Vietnam
e-mail: `nguyenquanbahong@gmail.com`

Contents

I have used the following Linux distributions: Ubuntu, Kubuntu, SUSE, OpenSUSE, where the 1st one is install in my personal notebooks/laptops, the 2nd one is install in my WIAS Dell XPS 15 (KDE provides a lot of good stuffs), while the other ones are install in computer servers at WIAS Berlin. Note that SUSE & OpenSUSE are produced by Germans[1].

Question 0.1. *Which resources I should use to learn deeply Linux in a fast but complete way?*

William Shotts. *The Linux Command Line: A Complete Introduction* (2e) seems a good start.

Question 0.2. *Which e-book readers should I use on Linux?*

---

[1]In my own experiences, I believe that Germans & Frenches prefer building their own facilities over using those provided by the rest of the world.

Chapter 1

Shotts, 2019. The Linux Command Line: A Complete Introduction. 2e

About the Author. "William Shotts has been a software professional for > 30 years & an avid Linux user for > 20 years. He has an extensive background in software development, including technical support, quality assurance, & documentation. He is also the creator of `LinuxCommand.org`, a Linux education & advocacy site featuring news, reviews, & extensive support for using the Linux command line." – Shotts, 2019, p. 6.

About the Technical Reviewer. "Jordi Gutiérrez Hermoso is a coder, mathematician, & hacker-errant. He runs Debian GNU/Linux exclusively since 2022, both at home & at work. Jordi has been involved with GNU Octave, a free numerical computing environment largely compatible with MATLAB, & with Mercurial, a distributed version control system. He enjoys pure & applied mathematics, skating, swimming, & knitting. Nowadays he thinks a lot about environmental mapping, greenhouse gas emissions, & rhino conservation efforts." – Shotts, 2019, p. 7

Introduction

"I want to tell you a story. No, not the story of how, in 1991, Linus Torvalds wrote the 1st version of the Linux kernel. You can read that story in lots of Linux books. Nor am I going to tell you the story of how, some years earlier, Richard Stallman began the GNU Project to create a free Unix-like operating system. That's an important story too, but most other Linux books have that one, as well.

No, I want to tell you the story of how you take back control of your computer.

When I began working with computers as a college student in the late 1970s, there was a revolution going on. The invention of the microprocessor[1] had made it possible for ordinary people like you & me to actually own a computer. It's hard for many people today to imagine what the world was like when only big business & big government ran all the computers. Let's just say, you couldn't get much done.

Today, the world is very different. Computers are everywhere, from tiny wristwatches to giant data centers to everything in between. In addition to ubiquitous[2] computers, we also have a ubiquitous network connecting them together. This has created a wondrous[3] new age of personal empowerment[4] & creative freedom, but over the last couple of decades something else has been happening. A few giant corporations have been imposing their control over most of the world's computers & deciding what you can & cannot do with them. Fortunately, people from all over the world are doing something about it. They are fighting to maintain control of their computers by writing their own software. They are building Linux.

Many people speak of "freedom" with regard to Linux, but I don't think most people know what this freedom really means. Freedom is the power to decide what your computer does, & the only way to have this freedom is to know what your computer is doing. Freedom is a computer that is without secrets, one where everything can be known if you care enough to find out." – Shotts, 2019, pp. 30–31

Why Use the Command Line?

"Have you ever noticed in the movies when the "superhacker" – you know, the guy who can break into the ultra-secure military computer in < 30 seconds – sits down at the computer, he never touches a mouse? It's because filmmakers realize

---

[1]**microprocessor** [n] (*computing*) a small unit of a computer that contains all the functions of the central processing unit.
[2]**ubiquitous** [a] present, appearing or found everywhere.
[3]**wondrous** [a] (*literary*) strange, beautiful, & impressive, SYNONYM: **wonderful**.
[4]**empowerment** [n] [uncountable] a positive feeling that you have some control over your life or the situation you are in.

that we, as human beings, instinctively know the only way to really get anything done on a computer is by typing on a keyboard!

Most computer users today are familiar only with the *graphical user interface* (GUI) & have been taught by vendors[5] & pundits[6] that the *command line interface (CLI)* is a terrifying thing of the past. This is unfortunate because a good command line interface is a marvelously[7] expressive way of communicating with a computer in much the same way the written word is for human beings. It's been said that "graphical user interfaces make easy tasks easy, while command line interfaces make difficult tasks possible," & this is still very true today.

Since Linux is modeled after the Unix family of operating systems, it shares the same rich heritage of command line tools as Unix. Unix came into prominence[8] during the early 1980s (although it was 1st developed a decade earlier), before the widespread adoption of the graphical user interface &, as a result, developed an extensive command line interface instead. In fact, 1 of the strongest reasons early adopters of Linux chose it over, say, Windows NT was the powerful command line interface that made the "difficult tasks possible." – Shotts, 2019, p. 31

What This Book Is About

"This book is a broad overview of "living" on the Linux command line. Unlike some books that concentrate on just a single program, such as the shell program `bash`, this book will try to convey how to get along with the command line interface in a larger sense. How does it all work? What can it do? What's the best way to use it?

This is not a book about Linux system administration. While any serious discussion of the command line will invariably lead to system administration topics, this book touches on only a few administration issues. It will, however, prepare the reader for additional study by providing a solid foundation in the use of the command line, an essential tool for any serious system administration task.

This book is Linux-centric. Many other books try to broaden their appeal by including other platforms such as generic Unix & macOS. In doing so, they "water down" their content to feature only general topics. This book, on the other hand, covers only contemporary Linux distributions. 95% of the content is useful for users of other Unix-like systems, but this book is highly targeted at the modern Linux command line user." – Shotts, 2019, p. 32

Who Should Read This Book

"This book is for new Linux users who have migrated from other platforms. Most likely you are a "power user" of some version of Microsoft Windows. Perhaps your boss has told you to administer a Linux server, or you're entering the exciting new world of single board computers (SBC) such as the Raspberry Pi. You may just be a desktop user who is tired of all the security problems & wants too give Linux a try. That's fine. All are welcome here.

That being said, there is no shortcut to Linux enlightenment. Learning the command line is challenging & takes real effort. It's not that it's so hard, but rather it's so *vast.* The average Linux system has literally *thousands* of programs you can employ on the command line. Consider yourself warned; learning the command line is not a casual[9] endeavor[10].

On the other hand, learning the Linux command line is extremely rewarding. If you think you're a "power user" now, just wait. You don't know what real power is – yet. &, unlike many other computer skills, knowledge of the command line is long-lasting. The skills learned today will still be useful 10 years from now. The command line has survived the test of time.

It is also assumed that you have no programming experience, but don't worry, we'll start you down that path as well." – Shotts, 2019, pp. 32–33

What's in This Book

"This material is presented in a carefully chosen sequence, muck like a tutor sitting next to you guiding you along. Many authors treat this material in a "systematic" fashion, exhaustively[11] covering each topic in order. This makes sense from a writer's perspective but can be very confusing to new users.

---

[5]**vendor** [n] **1.** a person or company that sells things, especially outside on the street; **2.** (*formal*) a company that sells a particular product; **3.** (*law*) a person who is selling something, especially a house.

[6]**pandit** [n] (also **pundit**) **1.** a Hindu priest or wise man; **2.** (*Indian English*) a teacher; **3.** (*Indian English*) a musician with a lot of skill.

[7]**marvellously** [adv] (US English **marvelously**) very; very well, SYNONYM: **wonderfully**.

[8]**prominence** [n] **1.** [uncountable, singular] the state of being important, well known or easy to notice; **2.** [countable, uncountable] (*medical* or *formal*) a thing that sticks out from something; the fact or state of sticking out from something.

[9]**casual** [a] **1.** [usually before noun] without paying attention to detail; **2.** [usually before noun] not showing much care or thought; **3.** [usually before noun] (of a relationship) lasting only a short time & without deep affection; **4.** [usually before noun] (BE) (of work) not permanent; not regular; **5.** not formal; **6.** [only before noun] happening by chance; doing something by chance.

[10]**endeavour** [n] (US **endeavor**) (*formal*) **1.** [uncountable, countable] serious effort to achieve something; an attempt to do something, especially something new or difficult; **2.** [countable, usually plural] something that somebody does; [v] **endeavor to do something** (*formal*) to try hard to do or achieve something, SYNONYM: **strive**.

[11]**exhaustive** [a] including everything possible; very thorough or complete.

Another goal is to acquaint[12] you with the Unix way of thinking, which is different from the Windows way of thinking. Along the way, we'll go on a few side trips to help you understand why certain things work the way they do & how they got that way. Linux is not just a piece of software; it's also a small part of the larger Unix culture, which has its own language & history. I might throw in a rant or 2, as well.

This book is divided into 4 parts, each covering some aspect of the command line experience.

- **Part 1, "Learning the Shell,"** starts our exploration of the basic language of the command line including such things as the structure of commands, file system navigation, command line editing, & finding help & documentation for commands.

- **Part 2, "Configuration & the Environment,"** covers editing configuration files that control the computer's operation from the command line.

- **Part 3, "Common Tasks & Essential Tools,"** explores many of the ordinary tasks that are commonly performed from the command line. Unix-like operating systems, such as Linux, contain many "classic" command line programs that are used to perform powerful operations on data.

- **Part 4, "Writing Shell Scripts,"** introduces shell programming, an admittedly rudimentary but easy-to-learn technique for automating many common computing tasks. By learning shell programming, you will become familiar with concepts that can be applied to many other programming languages." – Shotts, 2019, pp. 33–34

How to Read This Book

"Start at the beginning of the book & follow it to the end. It isn't written as a reference work; it's really more like a story with a beginning, middle, & end." – Shotts, 2019, p. 34

Prerequisites

"To use this book, all you will need is a working Linux installation. You can get this in 1 of 2 ways:

- **Install Linux on a (not so new) computer.** It doesn't matter which distribution you choose, though most people today start out with either Ubuntu, Fedora, or OpenSUSE. If in doubt, try Ubuntu 1st. Installing a modern Linux distribution can be ridiculously easy or ridiculously difficult depending on your hardware. I suggest a desktop computer that is a couple of years gold & has at least 2GB of RAM & 6GB of free hard disk space. Avoid laptops & wireless networks if at all possible, as these are often more difficult to get working.

- **Use a "live CD" or USB flash drive.** 1 of the cool things you can do with many Linux distributions is run them directly from a CD-ROM or USB flash drive without installing them at all. Just go into your BIOS setup & set your computer to boot from a CD-ROM drive or USB device & reboot. Using this method is a great way to test a computer for Linux compatibility prior to installation. The disadvantage is that it may be slow compared to having Linux installed on your hard drive. Both Ubuntu & Fedora (among others) have live versions.

Regardless of how you install Linux, you'll need to have occasional superuser (i.e., administrative) privileges to carry out the lessons in this book.

After you have a working installation, start reading & follow along with your own computer. Most of the material in this book is "hands on," so sit down & get typing!

Why I don't call it "GNU/LINUX". In some quarters, it's politically correct to call the Linux operating system the "GNU/Linux operating system." The problem with "Linux" is that there is no completely correct way to name it because it was written by many different people in a vast, distributed development effort. Technically speaking, Linux is the name of the operating system's kernel, nothing more. The kernel is very important, of course, since it makes the operating system go, but it's not enough to form a complete operating system.

Enter Richard Stallman, the genius-philosopher who founded the Free Software movement, started the Free Software Foundation, formed the GNU Project, wrote the 1st version of the GNU C Compiler (`gcc`), created the GNU General Public License (the GPL), etc., etc., etc. He *insists* that you call it "GNU/Linux" to properly reflect the contributions of the GNU Project. While the GNU Project predates the Linux kernel & the project's contributions are extremely deserving of recognition, placing them in the name is unfair to everyone else who made significant contributions. Besides, I think "Linux/GNU" would be more technically accurate since the kernel boots 1st & everything else runs on top of it.

In popular usage, Linux refers to the kernel & all the other free & open source software found in the typical Linux distribution, i.e., the entire Linux ecosystem, not just the GNU components. The operating system marketplace seems to prefer 1-word names such as DOS, Windows, macOS, Solaris, Irix, & AIX. I have chosen to use the popular format. If, however, you prefer to use "GNU/Linux" instead, perform a mental search-&-replace while reading this book. I won't mind." – Shotts, 2019, pp. 34–36

---

[12]**acquaint** [v] (*formal*) **acquaint somebody/yourself with something** to make somebody/yourself familiar with or aware of something.

What's New in the 2nd Edition

"While the basic structure & content remain the same, this edition of *The Linux Command Line* is peppered with various refinements, classifications, & modernizations, many of which are based on reader feedback. In addition, 2 particular improvements stand out. 1st, the book now assumes `bash` version 4.*x*, which was not in wide use at the time of the original manuscript. This 4th major version of `bash` added several useful new features now covered in this edition. 2nd, Part 4, "Shell Scripting," has been improved to provide better examples of good scripting practice. The scripts included in Part 4 have been revised to make them more robust, & I also fixed a few bugs." "This book is an ongoing project, like many open source software projects." – Shotts, 2019, p. 36

Part I: Learning The Shell

1.1   What Is the Shell?

"When we speak of the command line, we are really referring to the *shell*. The shell is a program that takes keyboard commands & passes them to the operating system to carry out. Almost all Linux distributions supply a shell program from the GNU Project called `bash`. The name is acronym for *b*ourne-*a*gain *sh*ell, a reference to the fact that `bash` is an enhanced replacement for `sh`, the original Unix shell program written by Steve Bourne." – Shotts, 2019, p. 38

1.1.1   Terminal Emulators

"When using a graphical user interface (GUI), we need another program called a *terminal emulator* to interact with the shell. If we look through our desktop menus, we will probably find one. KDE uses konsole, & GNOME uses gnome-terminal, though it's likely called simply Terminal on your menu. A number of other terminal emulators are available for Linux, but they all basically do the same thing: give us access to the shell. You will probably develop a preference for 1 or another terminal emulator based on the number of bells & whistles it has." – Shotts, 2019, p. 38

1.1.2   Making Your 1st Keystrokes

"Launch the terminal emulator. Once it comes up, we should see something like this: `[me@linuxbox ~]$`. This is called a *shell prompt*, & it will appear whenever the shell is ready to accept input. While it might vary in appearance somewhat depending on the distribution, it will typically include your `username@machinename`, followed by the current working directory (more about that in a little bit) & a dollar sign.

   If the last character of the prompt is a hash mark `#` rather than a dollar sign, the terminal session has superuser privileges. This means either we are logged in as the root user or we selected a terminal emulator that provides superuser (administrative) privileges." – Shotts, 2019, p. 39

1.1.3   Command History

"If we press the up arrow, we will see that the previous command entered reappears after the prompt. This is called *command history*. Most Linux distributions remember the last 1,000 commands by default. Press the down arrow & the previous command disappears." – Shotts, 2019, p. 39

1.1.4   Cursor Movement

"Recall the previous command by pressing the up arrow again. If we try the left & right arrows, we'll see that we can position the cursor anywhere on the command line. This makes editing commands easy.

   A few words about mice & focus. While the shell is all about the keyboard, you can also use a mouse with your terminal emulator. A mechanism built into the X Window System (the underlying engine that makes the GUI go) supports a quick copy-&-paste technique. If you highlight some text by holding down the left mouse button & dragging the mouse over it (or double-clicking a word), it is copied into a buffer maintained by X. Pressing the middle mouse button will cause the text to be pasted at the cursor location. Try it.

   Don't be tempted to use CTRL-C & CTRL-V to perform copy & paste inside a terminal window. They don't work. These control codes have different meanings to the shell & were assigned many years before the release of Microsoft Windows.

   Your graphical desktop environment (most likely KDE or GNOME), in an effort to behave like Windows, probably has its focus policy set to "click to focus." This means for a window to get focus (become active), you need to click on it. This is contrary to the traditional X behavior of "focus follows mouse," which means that a window gets focus just by passing the mouse over it. The window will not come to the foreground until you click on it, but it will be able to receive input. Setting

the focus policy to "focus follows mouse" will make the copy-&-paste technique even more useful. Give it a try if you can (some desktop environments such as Ubuntu's Unity no longer support it). I think if you give it a chance, you will prefer it. You will find this setting in the configuration program for your window manager." – Shotts, 2019, p. 40

1.1.5   Try Some Simple Commands

"Now that we have learned to enter text in the terminal emulator, let's try a few simple commands. Let's begin with the `date` command, which displays the current time & date.

```
[me@linuxbox ~]$ date
Fri Feb 2 15:09:41 EST 2018
```

A related command is `cal`, which, by default, displays a calendar of the current month.

```
[me@linuxbox ~]$ cal
February 2018
Su Mo Tu We Th Fr Sa
            1  2  3
 4  5  6  7  8  9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28
```

The console behind the curtain. Even if we have no terminal emulator running, several terminal sessions continue to run behind the graphical desktop. We can access these sessions, called *virtual consoles*, by pressing CTRL-ALT-F1 through CTRL-ALT-F6 on most Linux distributions. When a session is accessed, it presents a login prompt into which we can enter our username & password. To switch from 1 virtual console to another, press `alt-F1` through ALT-F6. On most systems, we can return to the graphical desktop by pressing `alt-F7`.

To see the current amount of free space on our disk drives, enter `df`.

```
[me@linuxbox ~]$ df
tmpfs              783256      2288    780968   1% /run
/dev/nvme0n1p2 491039648 79254728 386768040  18% /
tmpfs             3916268    154944   3761324   4% /dev/shm
tmpfs                5120         4      5116   1% /run/lock
/dev/nvme0n1p1      94759      5329     89430   6% /boot/efi
tmpfs              783252       156    783096   1% /run/user/1000
```

Likewise, to display the amount of free memory, enter the `free` command." – Shotts, 2019, pp. 41–42

```
[me@linuxbox ~]$ free
              total        used        free      shared  buff/cache   available
Mem:        7832540     6709784      217512      327756      905244      502532
Swap:             0           0           0
```

1.1.6   Ending a Terminal Session

"We can end a terminal session by closing the terminal emulator window, by entering the `exit` command at the shell prompt, or by pressing `ctrl-D`. `[me@linuxbox ~]$ exit`." – Shotts, 2019, p. 42

1.2   Navigation

"The 1st thing we need to learn (besides how to type) is how to navigate the file system on our Linux system. In this chapter, we will introduce the following commands: `pwd`: print name of current working directory, `cd`: change directory, `ls`: list directory contents." – Shotts, 2019, p. 43

1.2.1   Understanding the File System Tree

"Like Windows, a Unix-like operating system such as Linux organizes its files in what is called a *hierarchical directory structure*. This means they are organized in a tree-like pattern of directories (sometimes called *folders* in other systems), which may contain files & other directories. The 1st directory in the file system is called the *root directory*. The root directory contains files & subdirectories, which contain more files & subdirectories, & so on.

Note that unlike Windows, which has a separate file system tree for each storage device, Unix-like systems such as Linux always have a single file system tree, regardless of how many drives or storage devices are attached to the computer. Storage devices are attached (or more correctly, *mounted*) at various points on the tree according to the whims of the *system administrator*, the person (or people) responsible for the maintenance of the system." – Shotts, 2019, pp. 43–44

1.3   Exploring the System

1.4   Manipulating Files & Directories

1.5   Working with Commands

1.6   Redirection

1.7   Seeing the World as the Shell Sees It

1.8   Advanced Keyboard Tricks

1.9   Permissions

1.10   Processes

Part II: Configuration & The Environment

1.11   The Environment

1.12   A Gentle Introduction to vi

1.13   Customizing the Prompt

Part III: Common Tasks & Essential Tools

Part IV: Writing Shell Scripts

Chapter 2

Miscellaneous

2.1 Calibre

Official website: https://calibre-ebook.com/. See also, e.g., Wikipedia/Calibre (software). Enable Dark Mode in Calibre:

```
$ sudo nano /etc/profile.d/calibre.sh
```

```
export CALIBRE_USE_SYSTEM_THEME=1
$ sudo service gdm restart
```

& further customization. [inserting]

Bibliography

Shotts, William (2019). "The Linux Command Line: A Complete Introduction". In: p. 640.