# A tutorial to the Linux platform

This tutorial aims to show a brief introduction to Linux so that we will have sufficient knowledge to work on the Linux environment. The Linux platform is used on nearly all supercomputers. There are many Linux distributions. Ubuntu and CentOS are relatively more used due to well long-term support and easy use. Both of them work fine to practice Linux commands. Our Taco server is using CentOS.

## Access to Linux

1. Run a Linux on a Windows

We can run a local Linux within VirtualBox if our PC is a Windows system. For example, a good tutorial showed detailed steps to install Ubuntu on Windows.

<https://www.freecodecamp.org/news/how-to-install-ubuntu-with-oracle-virtualbox>

Briefly, it can be done in two steps.

1.1 Install VirtualBox

<https://www.virtualbox.org/wiki/Downloads>

Specifically, download and install the VirtualBox binary for Windows hosts.

<https://download.virtualbox.org/virtualbox/6.1.22/VirtualBox-6.1.22-144080-Win.exe>

1.2 Install Ubuntu within VirtualBox

After installing VirtualBox, we need to first download an Ubuntu image file, such as the Ubuntu 20.04.2.0 LTS.

<https://ubuntu.com/download/desktop>

Then, it will be easy to follow the installation steps in the tutorial.

2. Practice on a Mac machine directly

If we use a Mac OS X system, we don’t need to install a Linux distribution via VirtualBox, but to use Mac’s native terminal directly. Mac is a Unix distribution, and it shares a great number of utilities with Ubuntu or CentOS. So, it is desirable to practice Linux commands using the Mac terminal.

3. Our production Linux server: the Taco server

We also have a production Linux system called the Taco server. We can apply the access after checking-in to BCM. After check-in, BCM IT will create a unique ID for us. We need to contact IT to open access to our computer cluster, the Taco server. The Taco server has several hundred total CPUs, and computing nodes may have a large memory of up to 1TB. So, the Taco server has powerful computing resources for working projects. At the same time, it is also a good place to practice Linux.

## An introductory course to learn Linux

- Introduction to Linux

<https://www.edx.org/course/introduction-to-linux>

By its syllabus, this course covers introductory elements of Linux. It will serve the purpose to get familiar with Linux. But most of them are not directly relevant to data analysis, such as network configuration, or system securities. I believe only seven chapters from Chapter 7 to chapter 13 are directly relevant and need careful reading. I will also emphasize several additional points.

1. It is helpful to remember commonly used Linux commands, e.g.,

- ls: to list files or directories

- cd: to change a working directory

- cp: copy a file or directory

- mv: move or rename a file or directory (-i is suggested). To move a large directory, it is encouraged to copy (by cp) or sync (by rsync) it to a new place first, then delete the old one. Because a partial move will cause corrupt source and destination directories.

- rm: remove a file or directory (always be cautious with this remove command because removed files will not be recovered. Think twice before using this command, and -i is suggested)

- more/less: two text file peek commands

- man: return documentation for a command, such as `man ls`.

2. A very critical concept is the file system. We need to know the differences between a file and a directory. A file or directory can configure access permissions, e.g., read, write, or executable. We can visit a file or a directory by its absolute path or a relative path.

3. It is inevitable to master a text editor on Linux. See Chapter 11 for a basic introduction to Vim and Emacs. Vim and Emacs are very powerful for text editing. It may have a steep learning curve, but it will pay off. Basic operations include insert, copy, paste, selection, delete, move, search, and replace. Some cheat sheets will help remember common operations. Such as <https://devhints.io/vim>.

## Bash shell: native scripting language of Linux

Bash scripting language is native to Linux, and it is typically the default shell of most Linux distributions. Bash is especially powerful in executing Linux commands, piping data, and process management, compared to other scripting languages such as Python, or R. To find the Bash manual, we can run the Linux command `man bash`. Alternatively, full Bash documentation is at the below URL.

<https://www.gnu.org/software/bash/manual/bash.html>

It is impossible to learn the full Bash features or Linux commands in a short time. But we want to know where to find the official documents of them. We are going to accumulate knowledge of Bash features along with the script development. A starting feature can be the Linux pipe to redirect output from one command into the input of a second command.

<https://www.gnu.org/software/bash/manual/bash.html#Pipelines>

We will apply Bash features very frequently so that we can process large-scale data efficiently.

## Conda software management across Linux platforms

Open-source software provides a large number of utilities for data analysis. They are not fully installed by default. So, we need to install missing ones manually, such as Git. Linux distributions have maintained their software management system to install hosted software, e.g., apt-get on Ubuntu, yum on CentOS, and brew on Mac. These management systems have successfully extended Linux utilities. However, one big limitation is they require root permission in certain ways. Sometimes, we may not have access to root permission, such as on the Taco server. So, we need a solution to install the software in local directories without root. Conda is a software management system enhancing the portability of software. In general, nearly all Linux utilities have been deposited in the Conda repository for installation. In my opinion, Conda is the most important software for data analysis. Typical Python, R packages, and C++ programs can be easily installed via Conda because Conda will install dependencies automatically, while manual compiling is usually a headache and a nightmare. Below is an example use case of Conda.

1. Install Conda

Conda has two releases, Anaconda and Miniconda. Anaconda includes fully available software or packages, where full packages will consume huge space, and most of them are not relevant for our use. In comparison, Miniconda only installs essential binaries/libraries for Conda. So, it is efficient to install Conda via Miniconda, and we can install missing software or utilities anytime via conda (this is the purpose of Conda anyway). Specifically, download a Conda installer from the below link.

<https://docs.conda.io/en/latest/miniconda.html>

E.g., we want to install the “Python 3.9” Linux installer (on Ubuntu or CentOS).

```

$ mkdir ~/tools/miniconda3 && cd ~/tools/miniconda3

$ wget <https://repo.anaconda.com/miniconda/Miniconda3-py39_4.9.2-Linux-x86_64.sh>

$ bash Miniconda3-py39\_4.9.2-Linux-x86\_64.sh

```

Following the prompt instructions, we can install Miniconda3 in a local directory. After installation, we need to reopen the terminal session (open a new window, or re-login to the server) so that Conda paths will be recognized. By default, the Conda environment is not activated, but we need to activate one to install new utilities. The “base” environment is the default environment of Conda. To activate it, we can simply run

```

$ conda activate

```

We also want to activate the “base” environment upon opening a new terminal session, or logging into the server, we can set auto activate of base, namely run

```

$ conda config --set auto\_activate\_base true

```

Then relogin the terminal. Conda should be well set.

2. Install software via Conda

Once Conda is successfully installed, and the “base” environment has been activated, it is easy to install utilities in the local directory. For example, we can install Git via Conda.

```

(base) $ conda install -y git

```

It is encouraged to install software via Conda, and we want to search them first in the Conda repository at <https://anaconda.org>. For example, the Git package is available in Conda at <https://anaconda.org/conda-forge/git>.

3. Common conda commands

There are several commonly used Conda commands.

```

$ conda env list

$ conda activate

$ conda deactivate

$ conda install

$ conda create -n newenv

$ conda search

```

A good cheat sheet has also summarized commonly used commands. See

<https://www.datamachinist.com/cheat-sheets/anaconda-cheat-sheet/>