1 Learning objectives

The Unix-like operating system provide some commands are quite handy tools at data operations. In the learning objectives, we will go through the specifica commands with the syntax. First we are going to have a look at some specific Unix or Unix-like commands. Second, we will have a look at github. In the end, we will have a tour on Amazon EC2 cloud instance for cloud computing.

1.1 Unix shell or Unix-like shell

chmod is the abbriviation of "change mode" in unix shell. We can imagine the command is a function and the inputs are the arguments of the function. One of the implementation of the function is with three arguments. the syntax of the function is as follows:

$$\text{chmod}[OPTION]\ MODE[,MODE]\ FILE$$

where chmod is the function and the words in *italic* font are non-terminals. We can use the function to change the permissions for a file, for instance:

```
chmod g-w file1
```

where the command removes the group members' write permission. The following table shows the specific *MODE*s are defined for the correspoding classes.

Class	ls -l output
owner	-rwx------
group	----rwx---
other	-------rwx

find is a superb tool which searches files and directories in the file system with good performmance. We can use The syntax is as follows:

$$\text{find}\ [\text{-H}]\ [\text{-L}]\ [\text{-P}]\ [\text{-D}\ debugopts][\text{-O}level]\ [path...][expression]$$

where the text inside square brackets are different kinds of options. The letters with sanssarif represents the different tag for the corresponding options and the the words with italic font represent the non-terminals.

grep stands for "global regular expression print". We can use grep to process text line by line and prints out the lines that match a specified pattern. The following shows the syntax of grep:

$$\text{grep}\ [OPTIONS]\ PATTERN\ FILE$$

sed is the abbriviation of stream editor in Unix shell. We can use the command to filter and transorm texts which matches specified pattern. The syntax of sed is shown as follows:

$$\text{sed}\ OPTIONS\ [SCRIPT]\ FILE$$

apt-get is a command from the APT package where APT is the abbrivation of advanced packaging tool.

nano is a terminal based text editor for Linux. The syntax is as follows:

$$\text{nano } [OPTIONS] \ FILE$$

cut is a command that we can use it for cutting out as our specification.

$$\text{cut } OPTIONS \ [FILE]$$

sort is a command which we can use it for sorting the text line by line of a file.

$$\text{sort } [OPTIONS] \ [FILE]$$

uniq is a command which we can use it for filtering out the repreated items.

$$\text{uniq } [OPTIONS] \ INPUT[OUTPUT]$$

head is a command which we can use it for outputting the head portion of the input file.

$$\text{head } OPTIONS \ [FILE]$$

tail is a command which we can use it for outputting the tail portion of the input file.

$$\text{tail } OPTIONS \ [FILE]$$

less is a command line file viewer.

$$\text{less } [OPTIONS] \ [FILE]$$

cat is an abbrivation of concaternation which describes combinging string without any gaps. We can use the command to display and output the contenst of a file. The syntax is as follows:

$$\text{cat } [OPTIONS] \ [FILE]$$

cat is an abbrivation of secure shell which establishes secure protocol. We can use the command for remote logins.

$$\text{ssh } [OPTIONS] \ HOSTNAME$$

wc counts the words, newlines or bytes of input *FILE*.

$$\text{wc } [OPTIONS] \ [FILE]$$

echo is a command prints text to standard output.

$$\text{echo } [OPTIONS] \ [STRING]$$

echo is a command which we can use it for observing the reference manuals.

man [*OPTIONS*] [*COMMAND*]

awk is an abbrivation with initials of Aho, Weinberger, and Kernighan' who developed the domain-specific language. In Unix-like operating system, awk is gennerally symbolic link to the executable file /usr/bin/gawk. We can use the command for processing text and extract the data. The syntax of the awk is as follows:

awk [*OPTIONS*][*FILE*]

vi is a text editor. The syntax is as follows:

VI [*OPTIONS*] [*FILE*]

vi is a comprehensive comtomizable text editor. The syntax is as follows:

emaxs [*OPTIONS*] [*FILE*]

1.2 Git and Github

In general, version control system are also neccessary tool for backing up digital files to have a clear trace about the changes of the files. First of all, we need to create a new repository on the github wesite by using personal account. Then we can use CLI(Command Line Interface) for version control.

git config is a command for getting and setting a repository or global options. We normally configure email and identifier with the command for others to contact and identifiying who made the git commit, for instance:

git config user.name "*IDENTIFIER*"

git init creates an empty git repository or reinitialize an existing one. i.e.,we can use the command to create a new local file directory a local or turn a existing local file directorry to a git repository. And we would eventually push it to the push to a git repository branch. The syntax is as follows:

git init [*OPTIONS*] [*DIRECTORY*]

git clone clones a git repository into a directory. i.e., We can use the command to copy the git repository to local fire directory. In general, the command requires the correctly formed URL. The syntax is as follows:

git clone [*OPTIONS*] [*URL*]

git add adds changes to the index. i.e., we can use the command to updates the changes to the current content found in the local directory to prepare the next commit.

git add [*OPTIONS*] [*FILE*]

git rm removes filres from the working tree and from the index. The syntax is as follows:

$$\text{git rm } [OPTIONS] \ [FILE]$$

git commit record changes to the repository. i.e. we can use the command to cirmfirm the changes to the repository and also leave a description to the changes that we made to it. The syntax is shown as follows:

$$\text{git commit } [OPTIONS] \ [FILE]$$

git status shows the working tree status. i.e. we can check the changes that we have made in local git repository. The syntax is shown as follows:

$$\text{git status } [OPTIONS] \ [FILE]$$

git checkout switch branches or restore working trees.

$$\text{git checkout } [OPTIONS] \ [BRANCH]$$

git stash stash the changes in a dirty working directory away. i.e., we can use the command for retriving a working tree before any changes have been made to it while save the changes for later use.

$$\text{git stash } [OPTIONS] \ [FILE]$$

git pull fetches and integrate with another repository or a local branch. i.e. we can use the command to update the branches for the repository then merge them into current branch. The syntax is as follows:

$$\text{git pull } [OPTIONS] \ [REPOSITORY \ [REFSPEC]]$$

git push updates remote refs along with associated objects. i.e., we can use the command to update remote repository with local repository. The syntax is as follows:

$$\text{git push } [OPTIONS]$$

First of all, we need to be registered as a user of github in order to create a new remote repository in the github site. Then we can either initialize a local repository or just clone the remote repository. Typically, we should work in a specifica branch then merge the changes to the master if the changes are valid. Then we can follow the syntax of git push and pull to form a correct command.