Linux & Terminal Commands

Addressing questions

(note: answers from google)

1. Is Linux mandatory for Hadoop?

> Therefore, having basic knowledge of Linux is good as it is the preferred method for installing and managing Hadoop clusters. Also, **to work with Hadoop HDFS, knowing about the common Linux commands is, without a doubt, a must**.

2. Running Linux Commands in Docker or Running in Git Bash, what is the difference? if I could not install Docker successfully on my end, How Should I do?

> Git Bash is a command-line tool that allows users to run Linux-like commands on Windows . But Docker could launch a really Linux OS. The key disadvantages of using Git Bash in Windows versus running Linux in a virtual environment, like Docker

> 1. Limited functionality: Git Bash, while providing a bash-like environment, has limited functionality compared to a full Linux distribution. Many development tools and workflows are designed primarily for Linux, so using Git Bash may result in compatibility issues or require workarounds.
> 2. Environmental differences: Linux and Windows have fundamental architectural differences that can lead to inconsistencies in how software behaves. Running a virtual machine with a Linux distribution provides a more consistent and predictable development environment that better matches production environments.
> 3. Resource usage: Running a full Linux virtual machine requires more system resources (CPU, memory, storage) than using Git Bash. This can be a consideration if the development machine has limited hardware capabilities.
> 4. Network and firewall interactions: The integration of Git Bash with the Windows networking stack can sometimes introduce complications when interacting with development servers, databases, or other network resources compared to a standalone Linux environment.

> 5. Tooling and ecosystem: The Linux ecosystem has a broader range of development tools, libraries, and frameworks that are well-integrated and optimized for that platform. Some tools may have limited or suboptimal support on Windows with Git Bash.

3. Bash VS Shell

> Bash is a type of shell, while shell is a broad term for any program that provides a command-line interface to interact with an operating system

4. Bash VS Zsh

> Bash and Zsh are both popular UNIX shells used for shell scripting and as command languages. Bash is the default shell for most Linux distributions, while Zsh is the default shell for Kali Linux and macOS Catalina. Here are some ways they compare:
>
> 1. Popularity: Bash is a reliable and popular choice due to its widespread adoption and familiarity.
> 2. Features: Zsh offers advanced features like auto-complete, syntax highlighting, customizable prompts, and more scripting options.
> 3. History: Zsh has a history mechanism.
> 4. Compatibility: Zsh can pretend to be a Bourne shell when run as /bin/sh.
> 5. Variable handling: Zsh has improved variable and array handling with non-zero-based numbering.
> 6. Multi-line commands: Zsh allows editing of multi-line commands in a single buffer.
> 7. Spelling correction: Zsh has built-in spelling correction.
> 8. Command completion: Zsh has programmable command completion.
> 9. Shell functions: Zsh has shell functions with autoloading.
> 10. Plugins and themes: Zsh offers support for plugins and themes.

Commands List

–help tag

- `ls --help`
- `cat --help`

  …

file operation

- `ls`
- `ll -a`
- `mkdir`
- `mkdir -r`
- `touch`
- `mv` (mv to a new location)
- `mv` (rename)
- `rm`
- `rm -r`
- `cat`
- `pwd`
- `cd`

file permission

three types of user and three types of permission(rwx)

- `chmod u=rwx g=rx o=rx names.txt`
- `chmod 777 names.txt`
- `chmod 544 names.txt`
- `chown`

edit, find, search

edit

- `vim`
  type i: switch to the edit mode
  type Esc: switch to the command mode, then :wq means save and exit; :q means no save, just exit

find

- `find .`
- `find ..`
- `find ~/`

- `find ~/Downloads`
- `find . -type f -name "names.txt"`
- `find . -type f -name "*.txt"`
- `find . -type d -name`
- `find . -size +1K`
- `find . -empty`
- `find . -perm 777`

search

```
names.txt
    Dave
```

- `grep Dave names.txt`
- `grep -w Da names.txt` -w complete word
- `grep -w Dave names.txt`
- `grep Dave names.txt`
- `grep -i Dave names.txt` -i not case insentitive
- `grep -n Dave names.txt` -n search line number\
- `grep -B 3 Dave names.txt` -B the three lines that come before Dave

pipe

- `find . -type f -name "*.txt" -exec rm -rf {} +`
- `ls | grep txt`
- `history | grep txt`

compress

- `zip`
- `unzip`

operate system

- `history`
- `sudo`
- `whoami`
- `adduser Dave`

- `passwd Dave`
- `userdel Dave`
- `uname` kernal
- `uname -o` type
- `uname -m` architecture
- `uname -r` kernal version
- `cat /etc/os-release`
- - `hostname`
- `hostname -i`

cpu details

- `lscpu`

memory

- `free`
- `free -h`
- `vmstat`

`lsof` list all the open file, files are opened by any of the processes in the system

network

- `nslookup`
- `netstat` ports are listening
- `ifconfig`

process

- `ps aux`

remote connection

Tips and tricks

terminal shortcuts

1. multiple commands in a single line, use colon ;
2. move to the begining: ==ctrl + A ==
3. move to the end : ==ctrl E ==
4. remove sth, like say after the cursor: ctrl K
5. autocomplete: ==tab ==
6. scroll through various commands: using ==up and down key ==
7. run the previous command : !
8. search commands: ctrl R