Basic Linux for HPC

Session Outcomes

- Understand the difference between personal computers and HPC.
- Understand some basic components in computer system.
- Understand the concept within HPC environment.
- Understand the difference between Linux and other Operating System.
- Understand the different commands used in Linux to perform different tasks.

Software Required for This Sessions

Operating System	System Built-in	External Installation
Windows OS	`cmd` `powershell`	[PuTTY] [MobaXTerm]
Linux	`terminal`	
macOS	`terminal`	[iTerm2]

What is HPC?

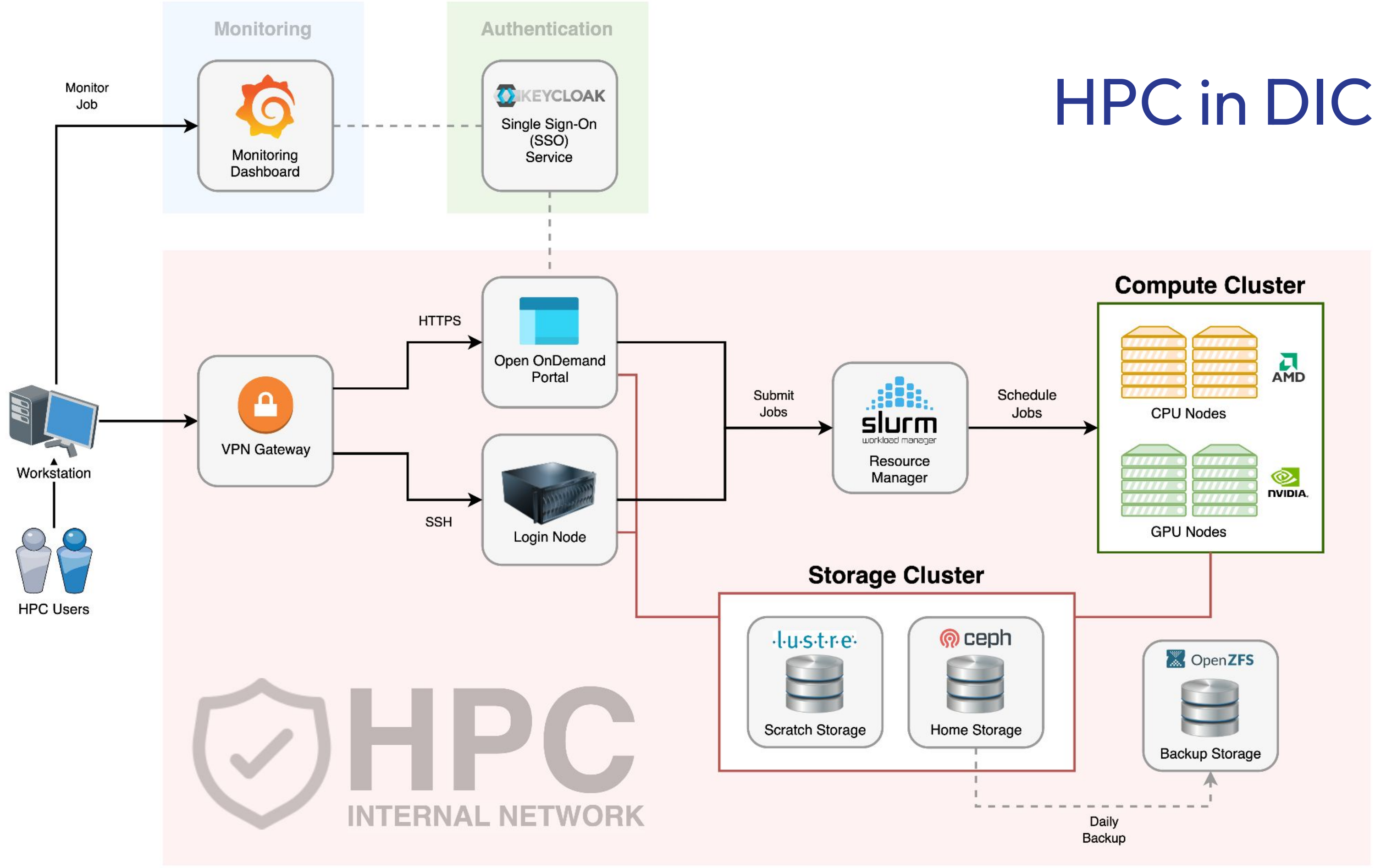
Workstation/Desktop/Laptop

Portable, affordable but limited computing power.

HPC Cluster

More investments and costs, but can be much more powerful.

HPC in DICC


Monitoring
Authentication
Monitor Job
Monitoring Dashboard
KEYCLOAK
Single Sign-On (SSO) Service
Compute Cluster
HTTPS
Open OnDemand Portal
Submit Jobs
slurm workload manager
Resource Manager
Schedule Jobs
CPU Nodes
AMD
GPU Nodes
NVIDIA.
VPN Gateway
Workstation
SSH
Login Node
HPC Users
Storage Cluster
lustre
Scratch Storage
ceph
Home Storage
OpenZFS
Backup Storage
Daily Backup
HPC
INTERNAL NETWORK

Why HPC?

Why do people use HPC?

- **Highly parallel calculation**
  - Can be splitted into multiple small calculations and execute concurrently.
- **Large-scale tightly coupled calculation**
  - Calculation require resources that beyond what a workstation or laptop can supply.
- **Computation require use of GPU**
  - Proven to be able to utilise GPU for massive speedup.

However,

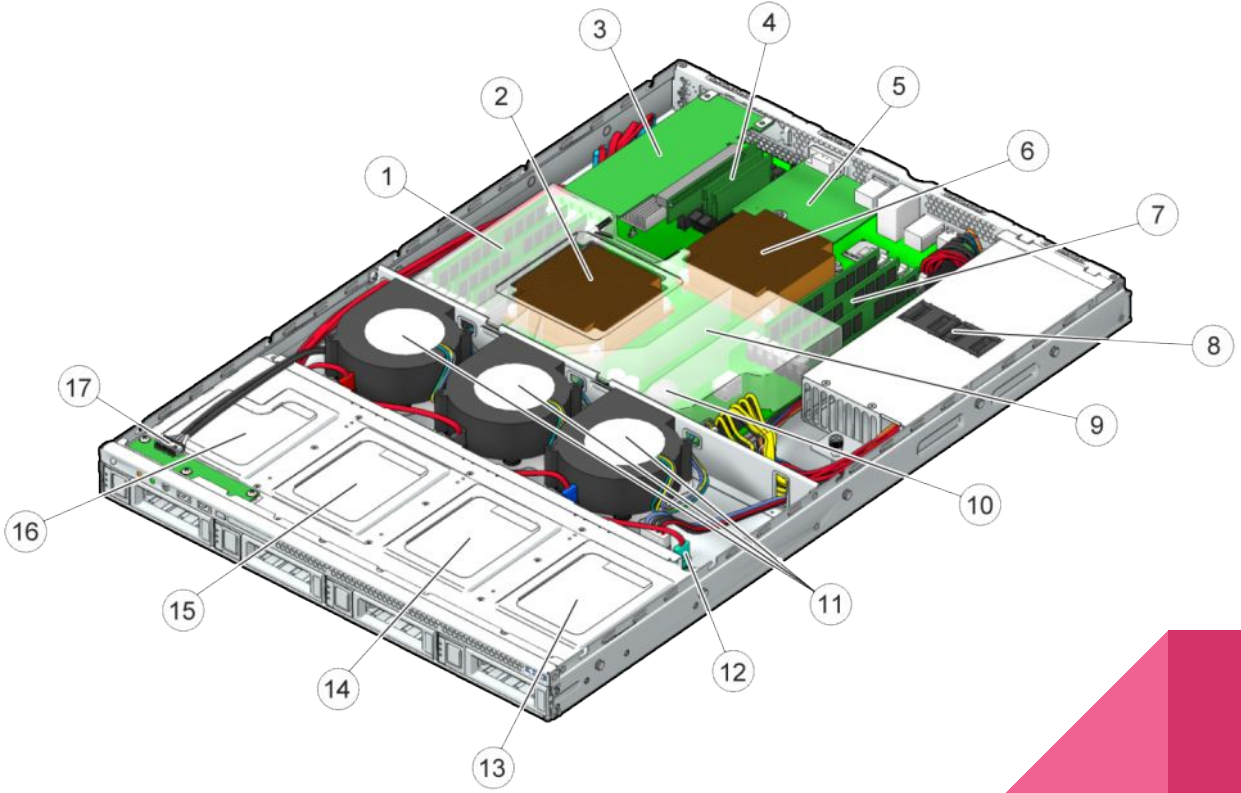
HPC is not the magic solution for everything.

What must you know?

- Basic Application Understanding
  - To be able to run and execute your application in the HPC.
- Basic Computer System Understanding
  - To understand the resources type in the HPC.
- Basic Linux Survival Skills
  - Must have basic Linux knowledge to survive in the HPC environment.

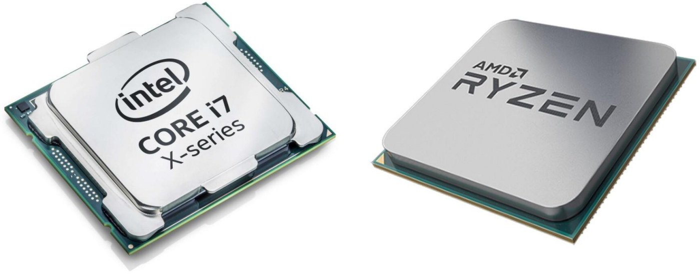
Computer System in Layman

What is inside the Server?



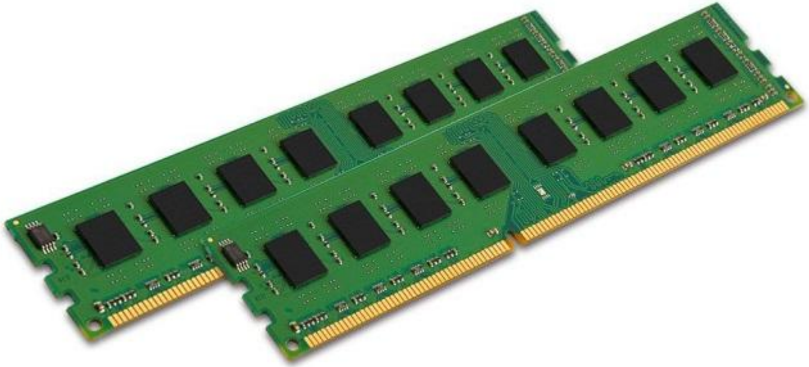
Processor in Layman

- Processor is the brain of any computer system.
  - Core - The processing core in the processor
  - Threads - Number of threads per core
  - Clock Speed - Number of CPU execution per secon
- Consumer Grade Processors:
  - Intel Core i7-1365U (10 Cores) - 5.20 GHz
  - AMD Ryzen 5 5600 (6 Cores) - 4.4 GHz
- Server Grade Processors:
  - Intel Xeon Platinum 8480+ (56 Cores) - 3.80 GHz
  - AMD EPYC 7702P (64 Cores) - 3.35 GHz



Memory in Layman

- Memory is the place where data required for CPU processing is stored.
- Common Memory Size: 1GB - 128GB per memory
- Memory Type: DDR1 - DDR5
- Newer type has lower latency, which mean faster access.
- Larger memory mean more stuff can run concurrently, and larger calculations can be supported.



GPU in Layman

- Super powerful processing unit that can dramatically accelerate additional workloads in high performance computing.
- Usually very expensive.
- Good for graphical processing, AI, accelerated mathematics calculations, and more!
- Example NVIDIA Models:
  - RTX 4090
  - A100
  - V100
  - H100

Storage in Layman

- The location where your files and directories are stored.
- Local Storage:
  - HDD
  - Hybrid-HDD
  - SSD
  - SAS
  - NL-SAS
- Network Storage:
  - NFS
  - Lustre
  - Ceph
  - GlusterFS

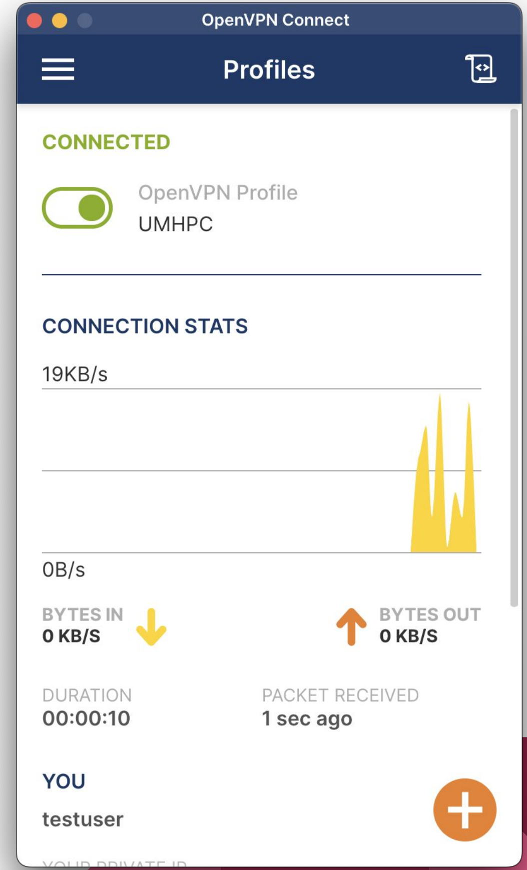
Accessing HPC Login Node

DICC Account

- DICC SSO (sso.dicc.um.edu.my)
  - Update password at DICC SSO.
  - If you forgotten your password, you can also reset your password at DICC SSO.
- Request HPC access at Service Desk.
- DO NOT SHARE YOUR ACCOUNT !!

VPN Connection

- Only account with HPC access can establish connection with the VPN gateway.
- Required software:
  - OpenVPN connect client
  - OpenVPN profile
- VPN Gateway:
  - `vpn01.dicc.um.edu.my`
  - `vpn02.dicc.um.edu.my`



CONNECTED

OpenVPN Profile
UMHPC

CONNECTION STATS

19KB/s

0B/s

BYTES IN
0 KB/S

BYTES OUT
0 KB/S

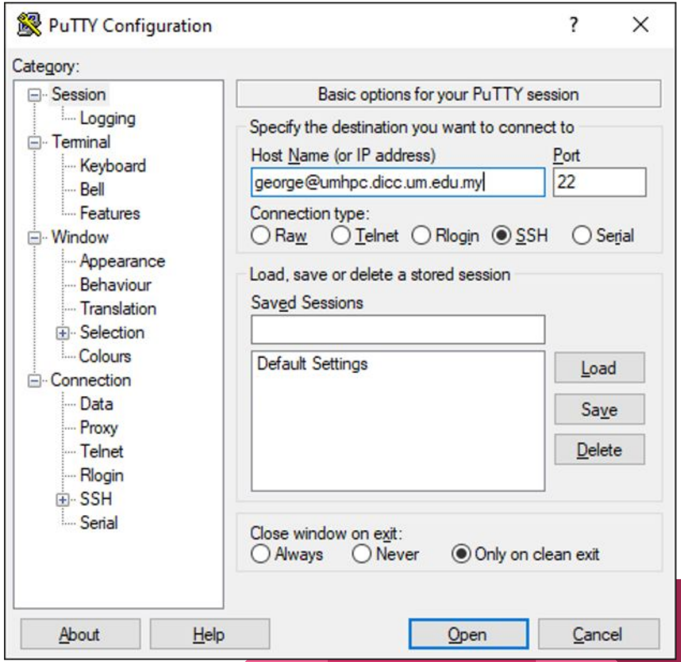
DURATION
00:00:10

PACKET RECEIVED
1 sec ago

YOU
testuser

Connecting to HPC Login Node

- Windows users:
  - PuTTY / MobaXTerm
- Linux / Mac OS users:
  - Use `ssh` command
  - `ssh username@umhpc.dicc.um.edu.my`
- Connection details:
  - Hostname : `umhpc.dicc.um.edu.my`
  - Port : `22`
  - Connection Type : `SSH`



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): george@umhpc.dicc.um.edu.my

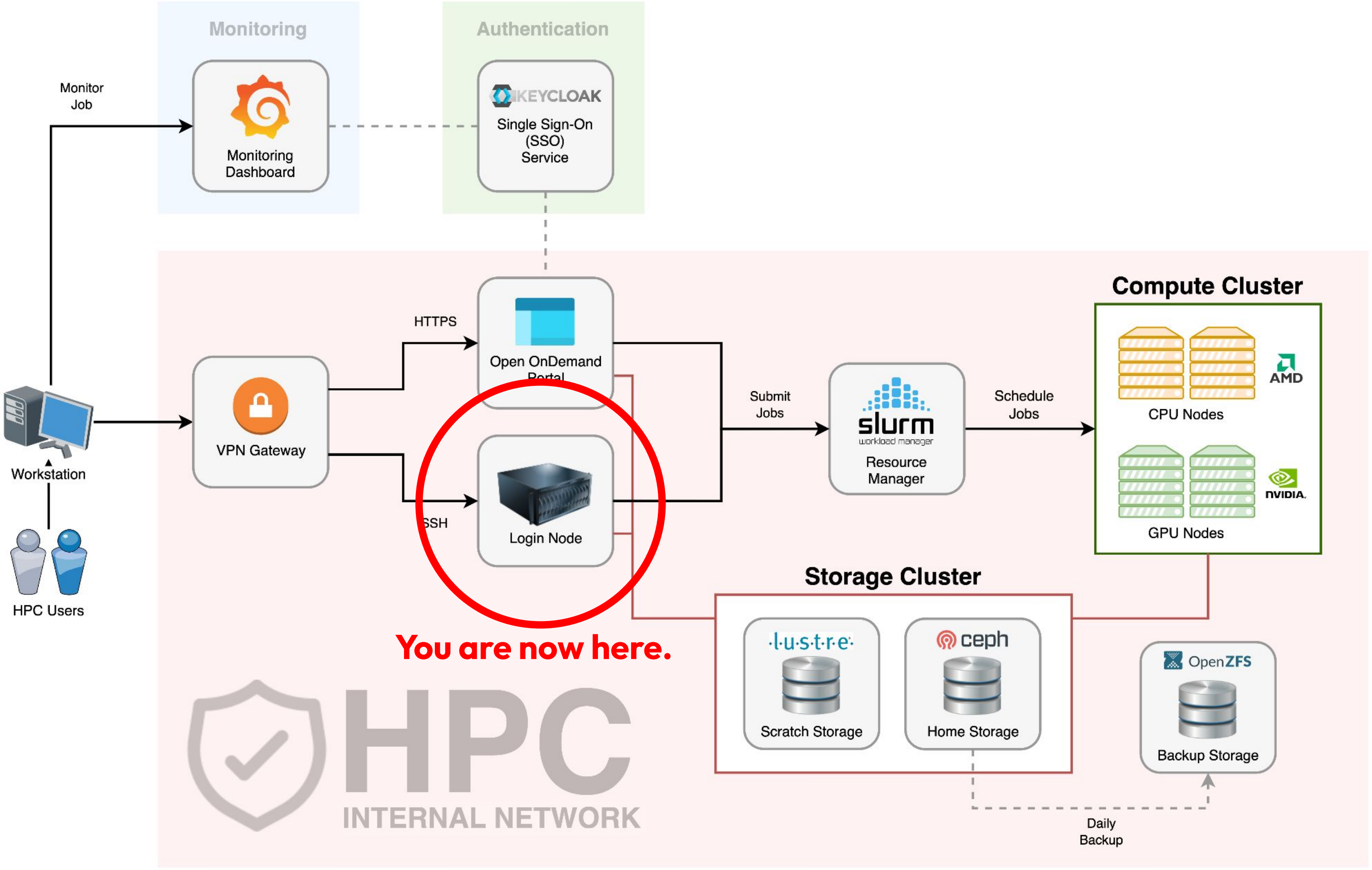
Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

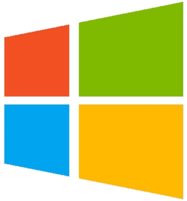
Close window on exit: ○ Always ○ Never ● Only on clean exit



You are now here.

Basic Introduction to Linux

CRAY®
a Hewlett Packard Enterprise company



Windows OS

Ubuntu

openSUSE

Linux OS

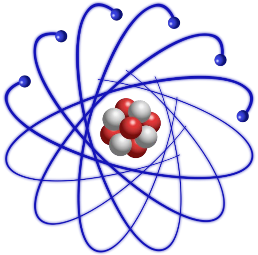
Rocky Linux

CentOS

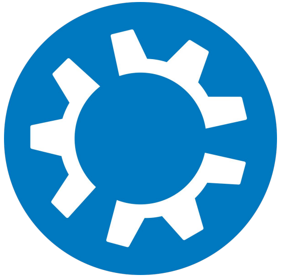
Mac OS

HPE Cray OS

Redhat



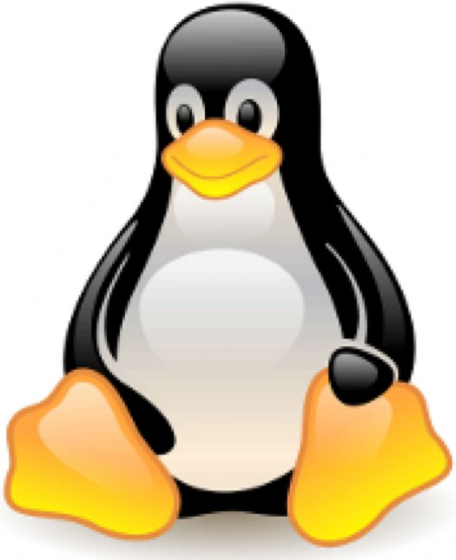
Scientific Linux



Kubuntu

What is Linux?

- Open source UNIX-like operating system.
- Many distributions and flavours:
  - Fedora
    - RedHat, CentOS, Rocky Linux
  - Debian
    - Ubuntu, Kubuntu
  - SUSE
    - SLES, OpenSUSE
- Widely used in server environments where performance matter.



Fedora Linux in DICC

- Free, open source
- Reliable
- Lightweight
- Allow multiple concurrent connections

CentOS

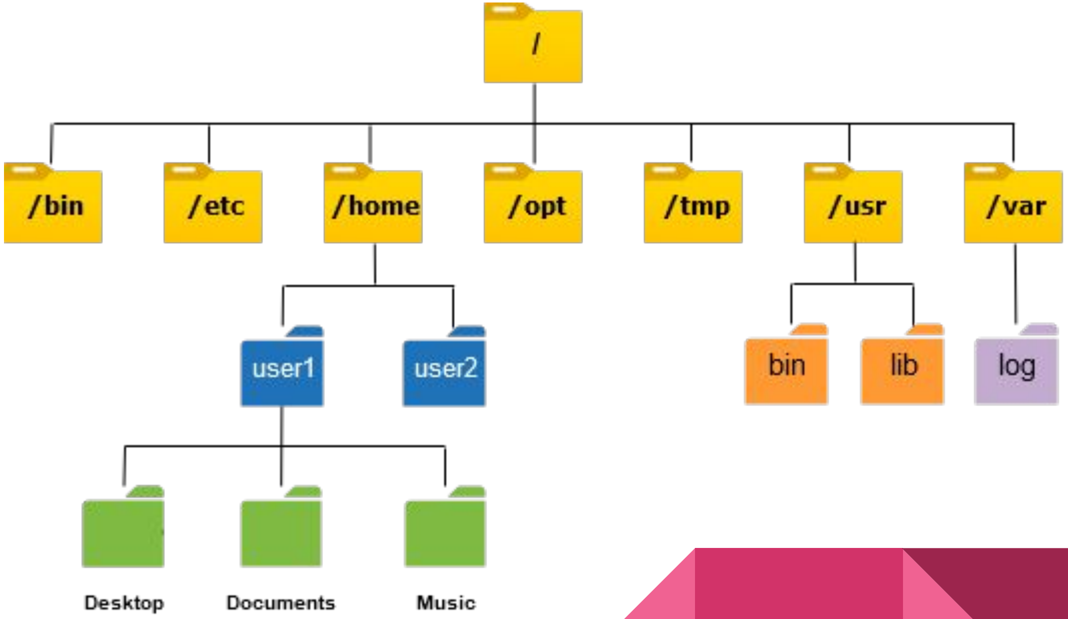
Rocky Linux™

User Roles in Linux

- Super Users
  - System Administrator that can access everything on the system.
- Regular Users
  - Can only access files and directories owned by themselves.
  - All HPC users belongs to this group.
- Service Users
  - System users that are used to run system services.

Directories in Linux

- Tree-like directory structure.
- Everything start with root directory.
  - `/home/user/`
  - `/opt/app/exe/`
  - `/tmp/scratch/`
  - `/dev/usb1/`
- No C/D/E drives like Windows



Linux Shell Variables

- A variable is a character string to which we can assign any value.
- Can contain the following characters:
  - letters (a to z or A to Z)
  - numbers (0 to 9)
  - underscore character (_)
- Example valid variables:
  - `_ALI`
  - `TOKEN_A`
  - `VAR_1`
- Example invalid variables:
  - `2_VAR`
  - `-VARIABLE`
  - `VAR1-VAR2`
  - `VAR_A!`

Linux Shell Variables

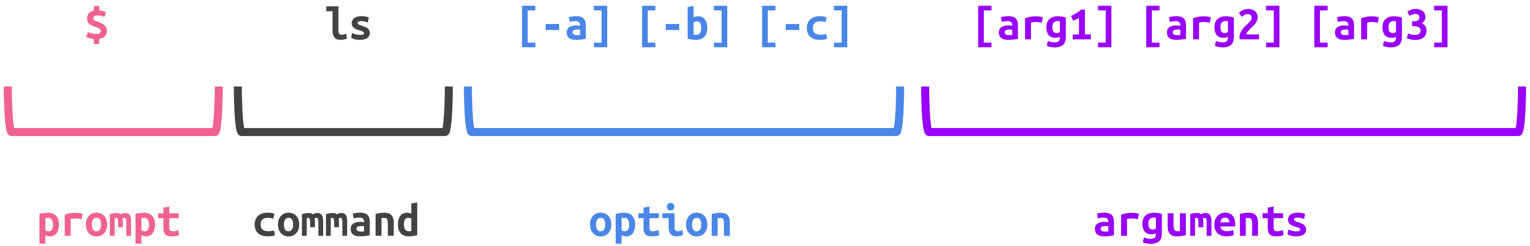
- The following are examples to define a variable:
  - `NAME=“George”`
  - `VAR1=“input.txt”`
  - `VAR2=100`
- Readonly variable is a variable that cannot be changed, once defined:
  - `NAME=“George”; readonly NAME`
  - executing `NAME=“John”` again will give an error message:
    - `/bin/sh: NAME: This variable is read only.`
- Unsetting variables:
  - `unset NAME`

Linux Shell Variables

- Accessing the variables:
  - `echo $NAME`
  - `echo “Hello $NAME - hope you're well.”`
  - `echo “Your file can be located as ${NAME}.txt”`
  - `touch ${NAME}.txt`

Linux Commands

Linux Command Structure



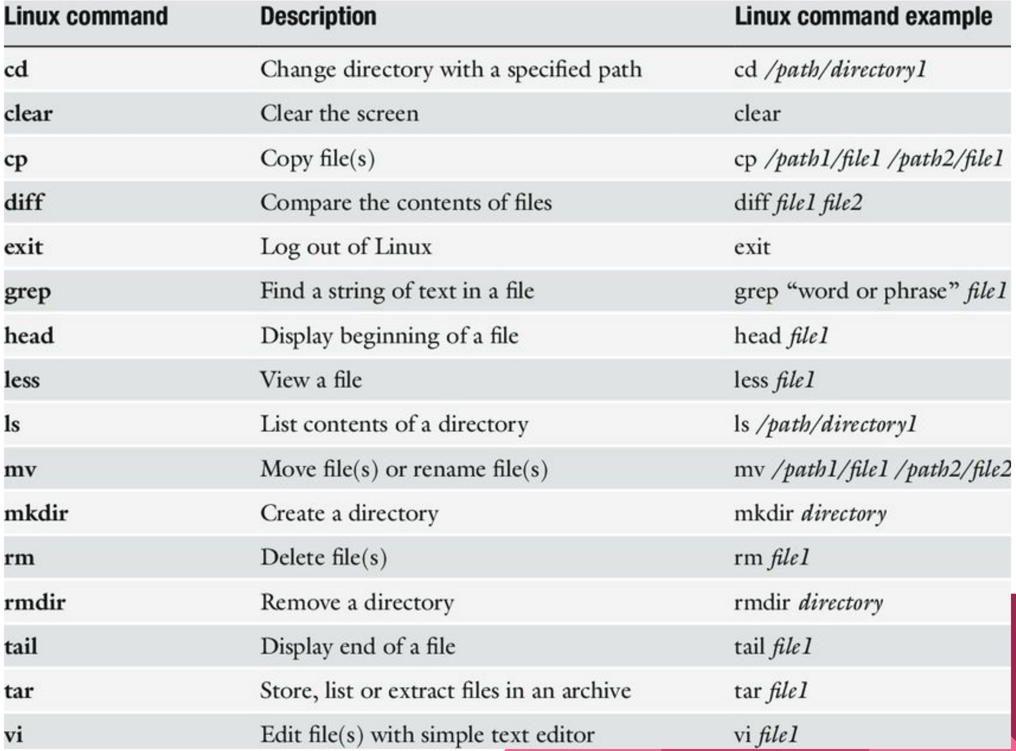
```
$ ls [-a] [-b] [-c] [arg1] [arg2] [arg3]
```

prompt command option arguments

- `ls -lah /home/user/george`
- `cd /tmp`
- `df`

Basic Linux Commands

- Instructions to perform basic actions in Linux.
  - Copy file
  - Move file
  - List directory
  - Navigate to another directory
  - Remove file or directory
  - Create new directory
  - Search for file or directory
- `Ctrl + C` to cancel instruction.



Linux command	Description	Linux command example
cd	Change directory with a specified path	cd */path/directory1*
clear	Clear the screen	clear
cp	Copy file(s)	cp */path1/file1 /path2/file1*
diff	Compare the contents of files	diff *file1 file2*
exit	Log out of Linux	exit
grep	Find a string of text in a file	grep "word or phrase" *file1*
head	Display beginning of a file	head *file1*
less	View a file	less *file1*
ls	List contents of a directory	ls */path/directory1*
mv	Move file(s) or rename file(s)	mv */path1/file1 /path2/file2*
mkdir	Create a directory	mkdir *directory*
rm	Delete file(s)	rm *file1*
rmdir	Remove a directory	rmdir *directory*
tail	Display end of a file	tail *file1*
tar	Store, list or extract files in an archive	tar *file1*
vi	Edit file(s) with simple text editor	vi *file1*

Clear Screen (clear)

Usage : **`clear`**

Clear the entire terminal screen output.

Examples :

- `clear`

Command Manual (man)

Usage : `man <command>`

Display the manual for the command, if any.

Examples :

- `man ls`
- `man cd`
- `man touch`
- `man clear`

List Directory Contents (ls)

Usage : `ls <destination>`

List information about the files (the current directory by default)

Options :

`-a`	list all files including hidden files
`-l`	use long listing format
`-h`	print sizes in human-readable format
`-i`	print index number of each file

Examples :

- `ls -lah /tmp`
- `ls -l /opt`
- `ls /home`

Alias for ls (ll)

Usage : `ll <destination>`

Alias for `ls -l`.

Options :

`-a`	list all files including hidden files
`-h`	print sizes in human-readable format
`-i`	print index number of each file

Examples :

- `ll -ah /tmp`
- `ll /home`
- `ll -h /lustre/user/george`

Navigate to Another Directory (cd)

Usage : `cd <destination>`

Navigate or move to another directory in the system.

Examples :

- `cd /home/user/george`
- `cd /lustre/user/george`
- `cd /tmp`
- `cd`
- `cd -`
- `cd ~`

Print Working Directory (pwd)

Usage : **`pwd`**

Print the full path to current directory.

Examples :

- pwd

Create Directory (mkdir)

Usage : `mkdir <destination>`

Create the directory if it does not already exist.

Options :

- `-p` Make parent directories if necessary

Examples :

- `mkdir -p /home/user/george/sampledir`

Update File Timestamps (touch)

Usage : `touch <filename>`

Update the access and modification times of file to current time.

Will automatically create file if absent.

Options :

- `-c` Do not create file if absent

Examples :

- `touch /home/user/george/empty`

Linux Text Editor

Programmers Text Editor (vi)

Usage : **`vi <filename>`**

Use `vi` editor to edit the specified file.

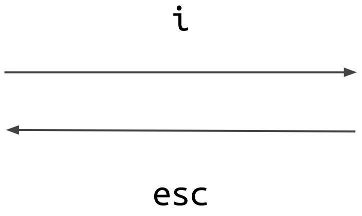
Examples :

- `vi /home/user/george/file01`
- `vi /lustre/user/george/file02`

Mode Switching (vi)

Command Mode (Default)

`arrow keys`	to navigate
`u`	to undo action
`dw`	to cut word
`yy`	to copy the current line
`dd`	to cut whole line
`P`	to paste before your cursor
`p`	to paste after your cursor
`:w`	to save the file
`:wq`	to save the file and quit
`:q!`	to quit without saving



Insert Mode

- Can only type in Insert Mode

WYSIWYG Text Editor (nano)

Usage :

```
nano <filename>
```

Use nano editor to edit the specified file.

Examples :

- `nano /home/user/george/file01`
- `nano /lustre/user/george/file02`

WYSIWYG Text Editor (nano)

- WYSIWYG - What you see is what you get
- Useful shortcuts:

`Ctrl + x`	exit
`Ctrl + o`	save
`Ctrl + w`	search
`Ctrl + k`	cut
`Ctrl + u`	paste
`Ctrl + g`	help menu

Print File Content (cat)

Usage : `cat <filename>`

Print the contents of the file on the terminal.

Examples :

- `cat /home/user/george/file01`
- `cat /lustre/user/george/file02`

Print Last Part of File (tail)

Usage : `tail [-n K] <filename>`

Print the last K lines of the file contents on the terminal.

Print a last 10 lines by default.

Examples :

- `tail /home/user/george/file01`
- `tail -n 25 /lustre/user/george/file02`

Inspect File in Scrollable Mode

less

Usage : **`less <filename>`**

Inspect the file in a scrollable mode.

Options :

- `-S` Do not wrap lines

Examples :

```
less /home/user/george/file01
less -S /lustre/user/george/file02
```

Simple Exercise (Part 01)

Let’s do it

- Create new directory **`training01`** in your home directory.
- Navigate to the created directory.
- Create a file named **`data.txt`** with following content using your desired text editor:

```
I have some sample data.
I have more sample data.
I have even more sample data.
```

- Check the content of current directory and ensure **`data.txt`** present.
- Print out the content of the **`data.txt`** and verify.

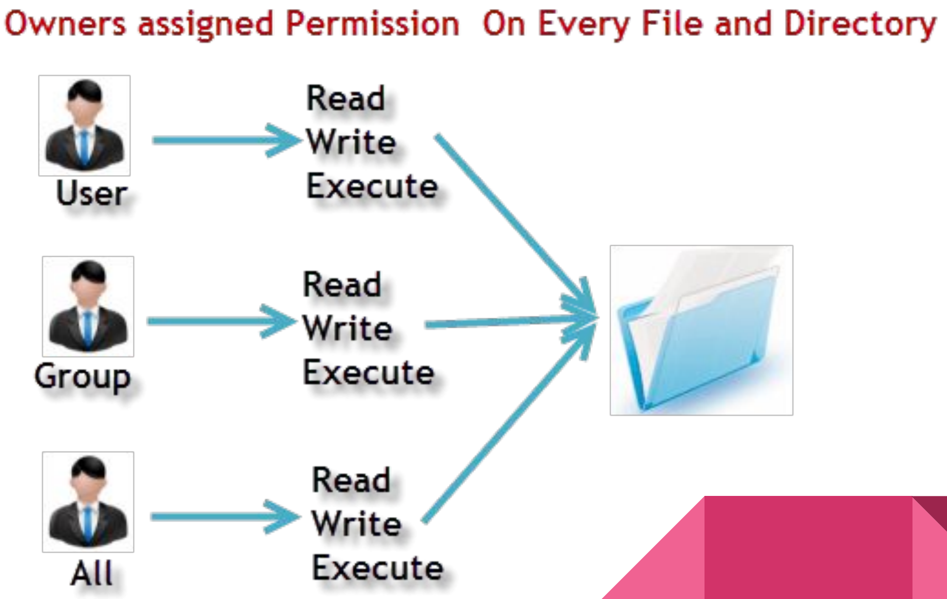
Let's do it Together (Answer)

- `mkdir ~/training01`
- `cd ~/training01`
- `nano data.txt` or `vi data.txt`
- `ls -l` or `ll`
- `cat data.txt` or `less data.txt`

File Permissions and Ownership

File Permissions and Ownership in Linux

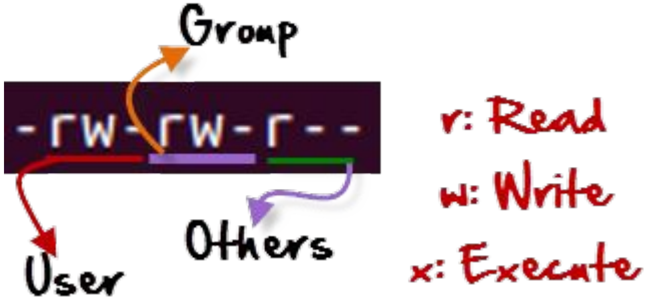
- **File Ownership**
  - User
  - Group
  - All
- **File Permissions**
  - Read
  - Write
  - Execute



Example Permissions & Ownership

- User **john** is a member of **alpha**.
- Can user **john** read the content of the files with following permissions?

○ `rwx --- ---`	`john`	`john`	`1.sh`
○ `rw- r-- r--`	`george`	`alpha`	`2.sh`
○ `rwx rwx rwx`	`william`	`beta`	`3.sh`
○ `rwx --- r--`	`george`	`alpha`	`4.sh`
○ `--- --- ---`	`john`	`john`	`5.sh`



Changing Permission (chmod)

Usage : `chmod <permission> <filename>`

Change the permission bits of the file or directory.

Examples :

```
chmod +x /home/user/george/file01
chmod 700 /home/user/george/file02
chmod u=rwx,g=-,o=- /home/user/george/dir01
chmod 644 /home/user/george/dir02
```

Changing Ownership (chown)

Usage : `chown [-R] <owner>:<group> <filename>`

Change the owner of the file or directory to another owner and group.

Options :

- `-R` Also include changing the owner for children within the directory.

Examples :

- `chown george:george file.txt`
- `chown george:alpha file.txt`
- `chown george:george dir`
- `chown -R george:george dir_with_files`

Important !!

DO NOT

set your home directory or scratch directory to permission 777 !!

Execute Script (source)

Usage : `source <filename>`

Execute script without execution permission bit.

Examples :

- `source example.sh`
- `source dir/example.sh`
- `example.sh`
- `dir/example.sh`

Execute Script (Direct Run)

Usage : **`./<filename> or <filename>`**

Execute script with execution permission bit.

Examples :

- `./example.sh`
- `dir/example.sh`

Copy File (cp)

Usage : `cp [-R] <source> <destination>`

Copy file or directory from source to destination.

Options :

- `-R` Recursively copy directories and files.

Examples :

```
• cp file.txt copy_of_file.txt
• cp -R dir copy_of_dir
• cp dir/* dir2/
• cp dir/file.txt dir2/copy_of_file.txt
```

Move File (mv)

Usage : `mv <source> <destination>`

Move file or directory from source to destination.

Can also be used to rename file.

Examples :

- `mv file.txt file2.txt`
- `mv dir dir2`
- `mv dir/* dir2/`
- `mv file.txt dir2/`

Remove File / Directory (rm)

Usage :

```
rm <file>
```

Remove the specified file or directory

Options :

● `-r`	remove directories and their contents recursively
● `-f`	ignore nonexistent files and arguments, never prompt
● `-i`	prompt before every removal

Examples :

```
● rm /home/user/george/test.txt
● rm -r /home/user/george/test
```

Simple Exercise (Part 02)

Let’s do it Together

- Navigate to the **`training01`** directory created earlier.
- Create a script named **`script.sh`** with the following content.

```
#!/bin/sh

echo Hello HPC!!
echo I am now in $(pwd) directory.
echo These are the contents from $1.
cat $1
```

- Add executable permission to the **`script.sh`**.
- Execute **`script.sh`** with **`data.txt`** as argument.
- Remove the directory **`training01`**.

Let's do it Together (Answer)

- `cd ~/training01`
- `nano script.sh` or `vi script.sh`
- `chmod +x script.sh`
- `./script.sh data.txt`
- `cd ~`
- `rm -r training01` or `rm -rf training01`

Any Questions?