


T001 - Basic HPC Linux



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 about how various stuffs in Linux work in general.
 - Understand the different commands used in Linux to perform different tasks.
- 

Software Required for This Session

Operating System	System Built-in	External Installation
Windows OS	<code>cmd</code> <code>powershell</code>	PuTTY MobaXTerm
Linux	<code>terminal</code>	
macOS	<code>terminal</code>	iTerm2



What is
High Performance
Computing (HPC)?



Workstation/Desktop/Laptop

Portable, affordable but limited computing power.



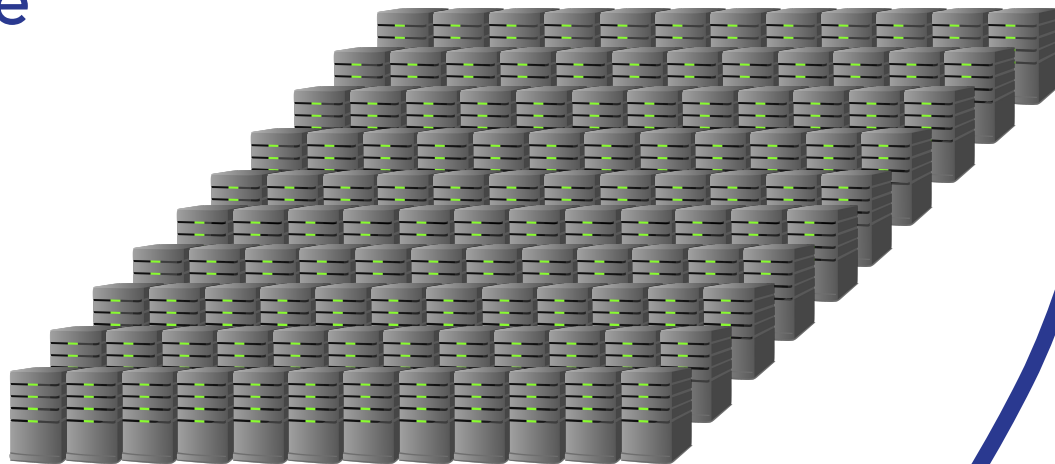
HPC Cluster

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

Comparing the Scale

World Top HPC Data Center (Fugaku)

7,630,848 Cores, 4.85PB Memory



Laptop

4 Cores, 8GB Memory, GPU



Workstation

16 Cores, 128GB Memory, GPU

UM HPC Cluster

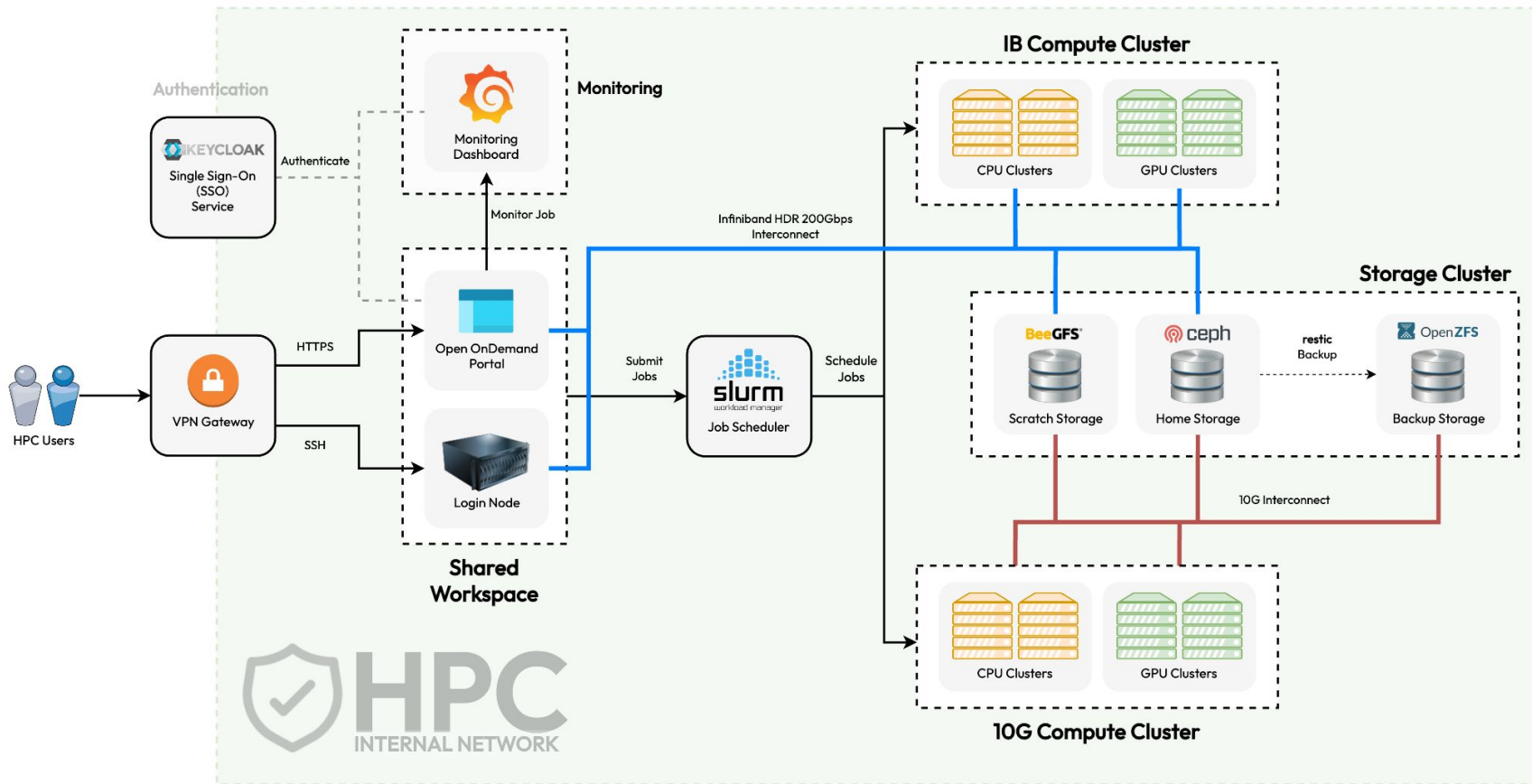
1024 Cores, 4TB Memory, GPUs

Computing
Power
(TFlops)

Total Cost of
Ownership



HPC in DICC





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.**



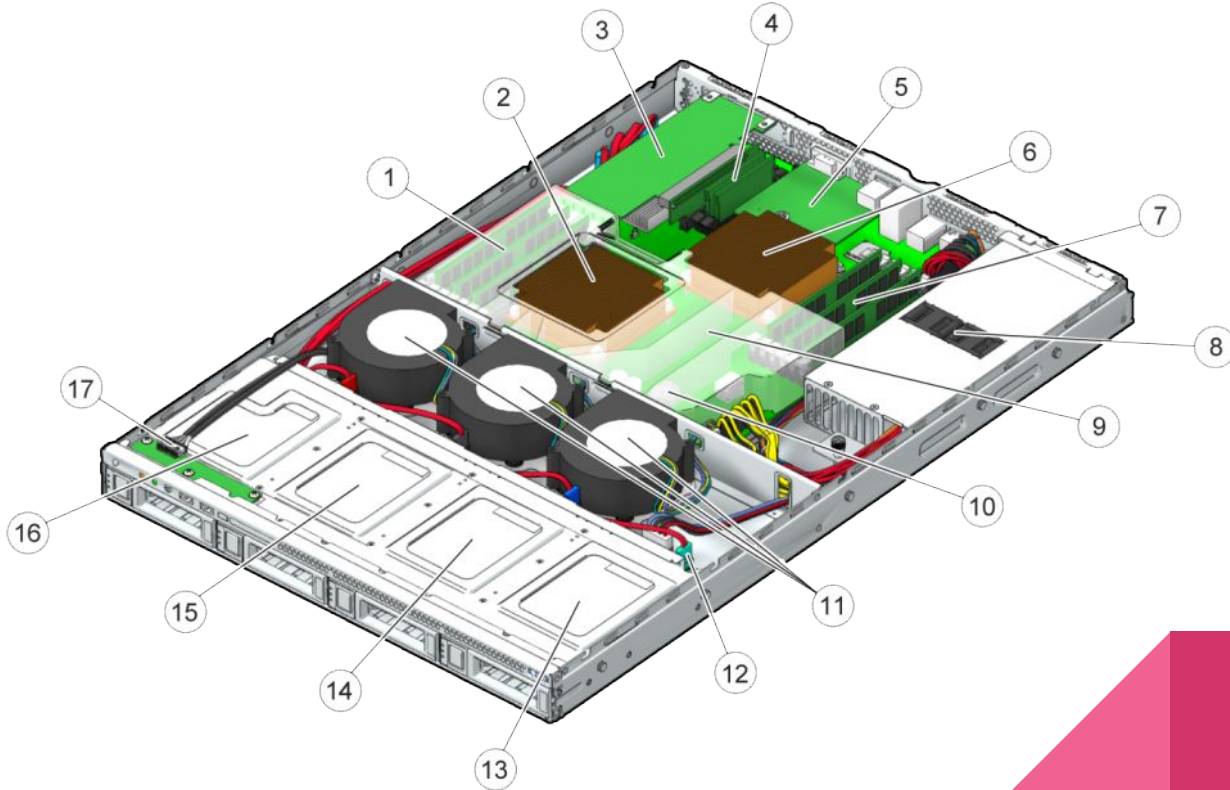
What must you know?

- **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.

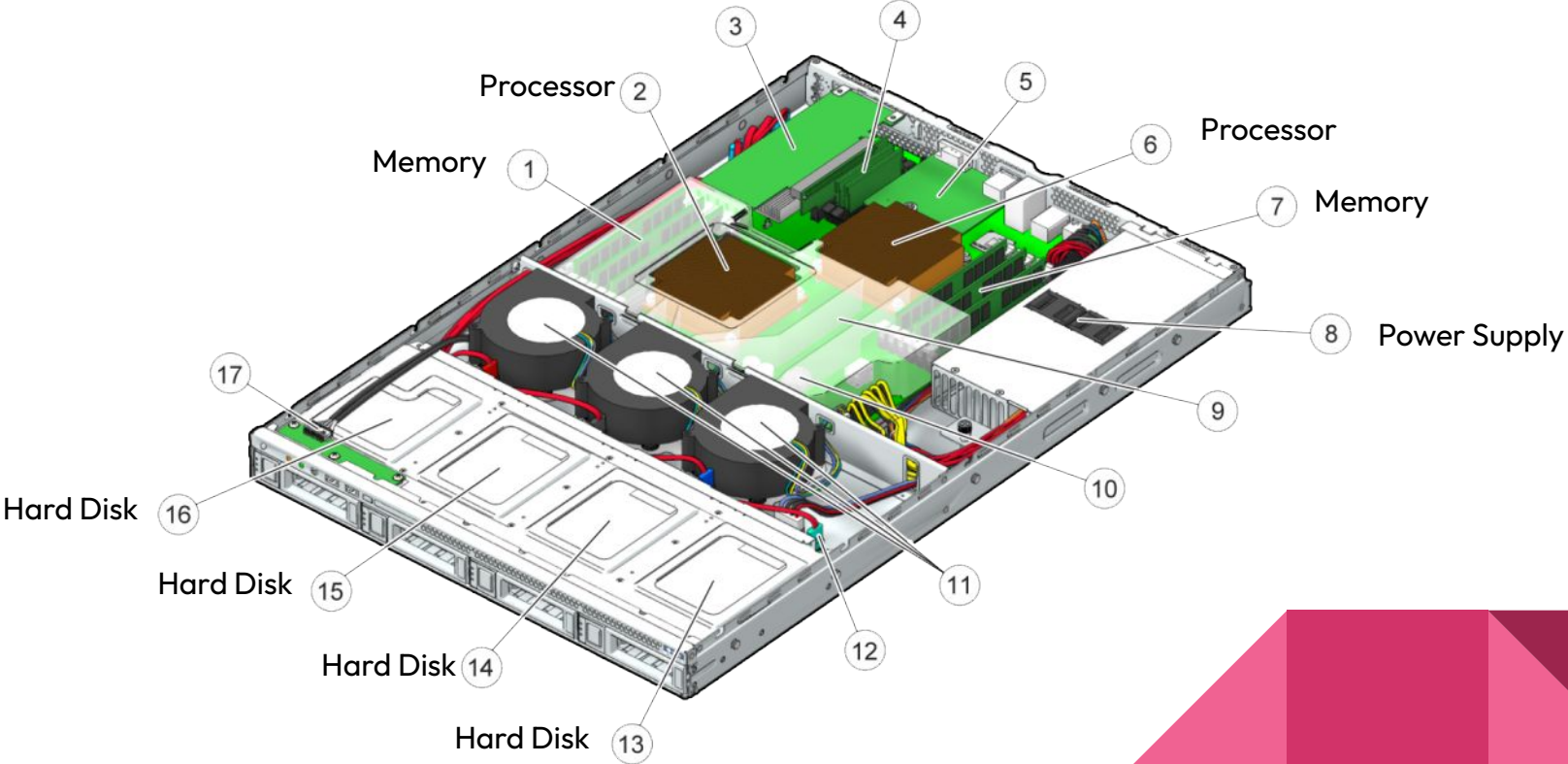


Computer System in Layman

What is inside the Server?

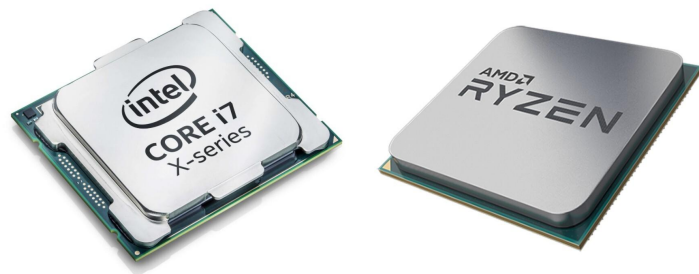


What is inside the Server?



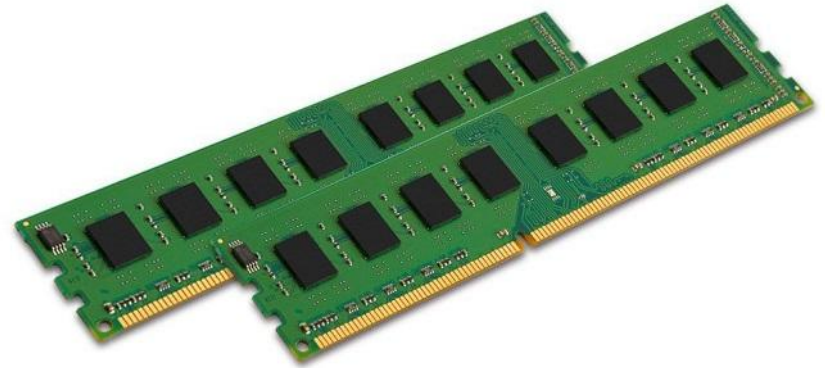
Processor in Layman

- 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 second
- 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
 - Tesla A100
 - Tesla V100
 - Tesla 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

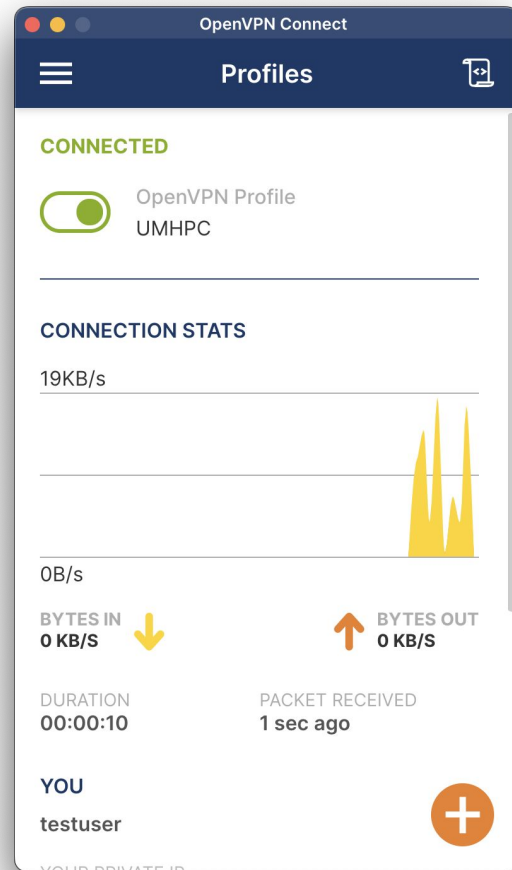
DICC Account

- 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 !!**



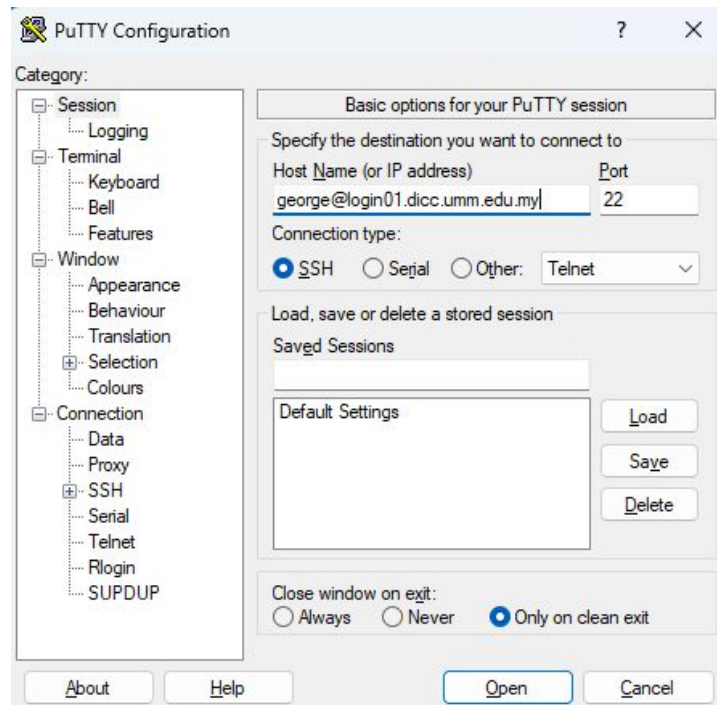
VPN Connection

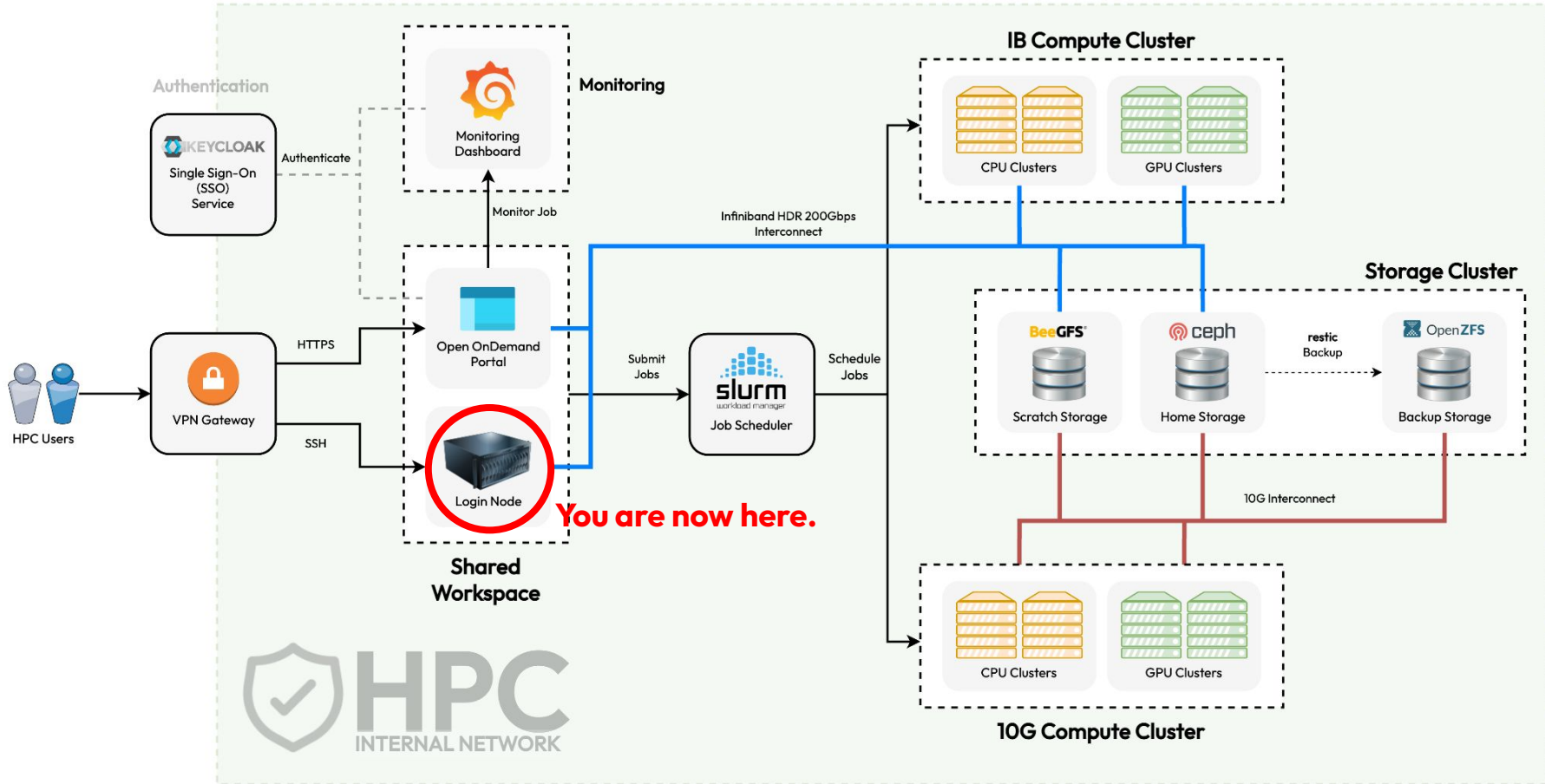
- Only account with HPC access can establish connection with the VPN gateway.
- Required software:
 - OpenVPN connect client
 - OpenVPN profile
- VPN Gateway:
 - `vpn01.dicc.um.edu.my`
 - `vpn02.dicc.um.edu.my`



Connecting to HPC Login Node

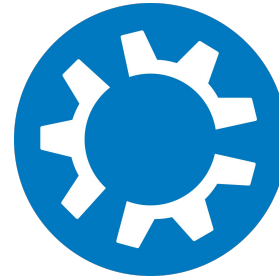
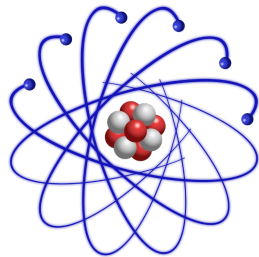
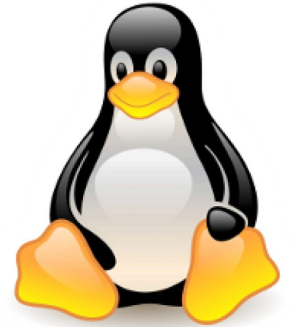
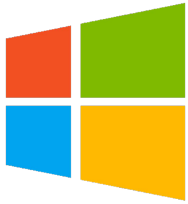
- Windows users:
 - PuTTY / MobaXTerm
- Linux / Mac OS users:
 - Use `ssh` command
 - `ssh username@login01.dicc.um.edu.my`
- Connection details:
 - Hostname : `login01.dicc.um.edu.my`
 - Port : `22`
 - Connection Type : `SSH`







Basic Introduction to Linux



CRAY[®]
a Hewlett Packard Enterprise company



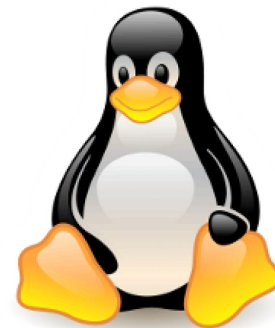
Windows OS



Ubuntu



openSUSE



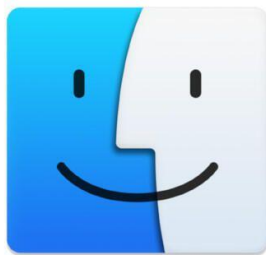
Linux OS



Rocky Linux



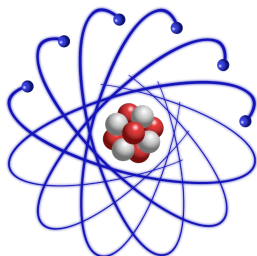
CentOS



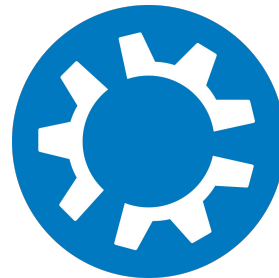
Mac OS



Redhat



Scientific Linux



Kubuntu



HPE Cray OS

What is Linux OS?

- 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 DICCC

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



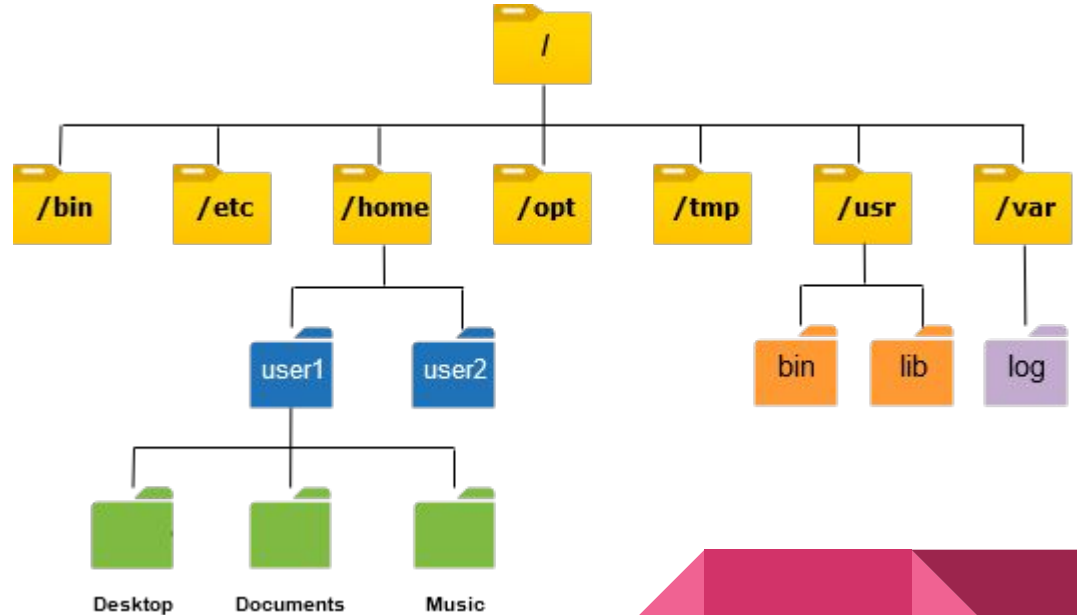
User Roles in Linux

- **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.



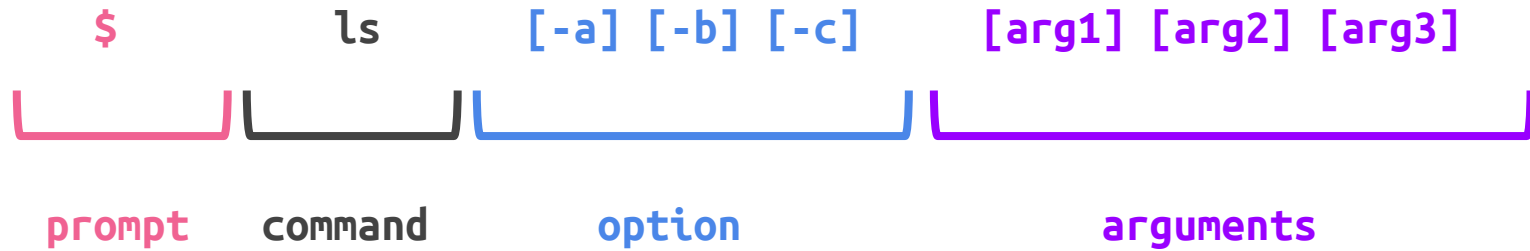
Directories in Linux

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



Linux Commands

Linux Command Structure



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

→ clear

Clear Screen

Usage : `clear`

Clear the entire terminal screen output.

Examples :

- `clear`

→ clear
→ man

Command Manual

Usage : `man <command>`

Display the manual for the command, if any.

Examples :

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

- clear
- man
- ls

List Directory Contents

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`

→ clear
→ man
→ ls
→ ll

Alias for ls command

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

→ clear
→ man
→ ls
→ ll
→ cd

Navigate to Another Directory

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 ~`

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd

Print Working Directory

Usage : `pwd`

Print the full path to current directory.

Examples :

- `pwd`

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ **mkdir**

Create Directory

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/samplendir`

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch

Update File Timestamps

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`

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch
→ vi

Programmers Text Editor

Usage : `vi <filename>`

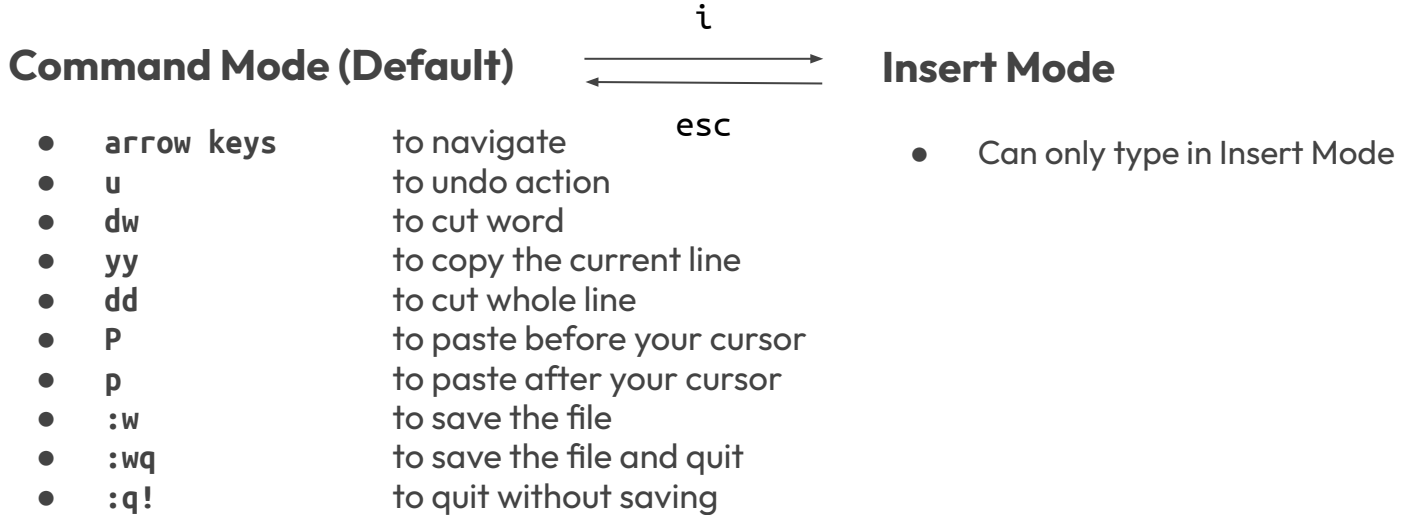
Use vi editor to edit the specified file.

Examples :

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

- clear
- man
- ls
- ll
- cd
- pwd
- mkdir
- touch
- vi

Mode Switching in vi Editor



→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch
→ vi
→ nano

WYSIWYG Text Editor

Usage : `nano <filename>`

Use nano editor to edit the specified file.

Examples :

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

- clear
- man
- ls
- ll
- cd
- pwd
- mkdir
- touch
- vi
- nano

Commands in nano Editor

- 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

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch
→ vi
→ nano
→ **cat**

Print File Content

Usage : `cat <filename>`

Print the contents of the file on the terminal.

Examples :

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

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch
→ vi
→ nano
→ cat
→ tail

Print Last Part of File

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`

→ clear
→ man
→ ls
→ ll
→ cd
→ pwd
→ mkdir
→ touch
→ vi
→ nano
→ cat
→ tail
→ less

Inspect File in Scrollable Mode

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)

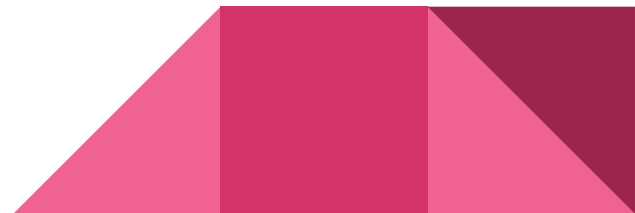
Let's Practise

- 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.

- ❑ `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

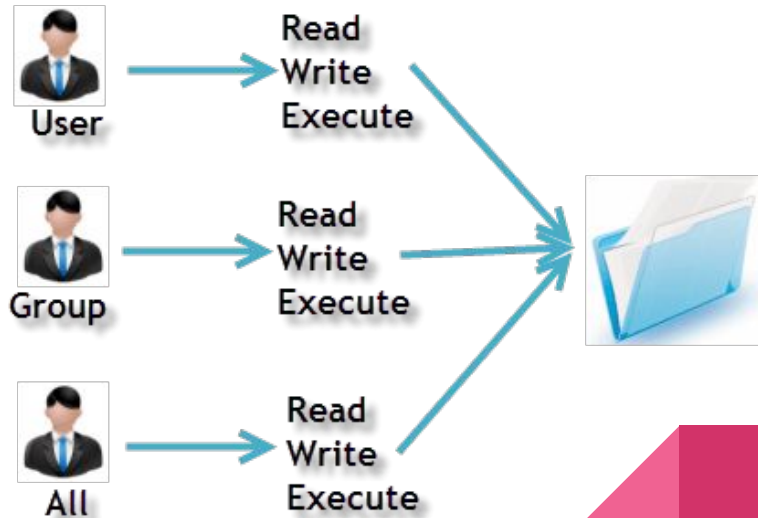
- **File Ownership**

- User
- Group
- All

- **File Permissions**

- Read
- Write
- Execute

Owners assigned Permission On Every File and Directory



Example Permissions & Ownership

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

○	<code>rwX --- ---</code>	<code>john</code>	<code>john</code>	<code>1.sh</code>
○	<code>rw- r-- r--</code>	<code>george</code>	<code>alpha</code>	<code>2.sh</code>
○	<code>rwX rwX rwX</code>	<code>william</code>	<code>beta</code>	<code>3.sh</code>
○	<code>rwX --- r--</code>	<code>george</code>	<code>alpha</code>	<code>4.sh</code>



Important !!

DO NOT

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



→ chmod

Changing Permission

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`

→ chmod
→ chown

Changing Ownership

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`

→ chmod
→ chown
→ source

Script Execution

Usage : `source <filename>`

Execute script without execution permission bit.

Examples :

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

→ chmod
→ chown
→ source

Direct Script Execution

Usage : `./<filename>` or `<path/to/file>`

Execute script with execution permission bit.

Examples :

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

→ chmod
→ chown
→ source
→ cp

Copy File

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`

→ chmod
→ chown
→ source
→ cp
→ mv

Move / Rename File

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/`

→ chmod
→ chown
→ source
→ cp
→ mv
→ **rm**

Remove File / Directory

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`

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`



Bash Profile

- Users can define what they wish to do automatically on login in the user **bash profile**.
- Every single lines defined in **bash profile** will be executed on user login.
- General use case:
 - Export variables
 - Run command
 - Activate custom environment
 - Load application module
 - Customise bash prompt
- Location of the file:
 - `~/.bash_profile`



Simple Exercise (Part 02)

Let's Practise

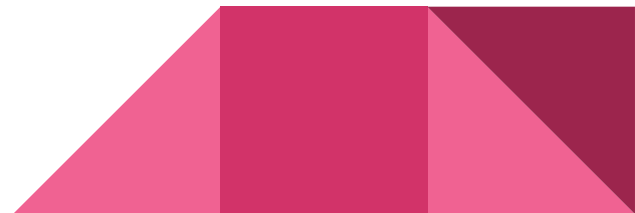
- Navigate to the **training01** directory created earlier.
- Create a script named **script.sh** with the following content.

```
#!/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**.

- ❑ `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?