



# Linux

## An Introduction

**Mahmoud Parvizi, Ph.D.**

Research Consultant, Institute for Cyber-Enabled Research  
Research Associate, Department of Physics and Astronomy

[parvizim@msu.edu](mailto:parvizim@msu.edu)



# Linux

## An Introduction

The term **Linux** is generally shorthand for the [GNU/Linux Operating System](#)

- Open-source, general-purpose operating system (OS)
- Developed in the 1990's to rival a costly OS like [Unix](#) or [MS-DOS](#)
- Quickly adopted by NASA and public universities
- Now used in mobile devices, personal and supercomputers



# Linux

## An Introduction

A **Linux distribution (distro)** is OS software components bundled together in accordance with the needs of the user

- Typically Includes:
  - [Linux core](#)
  - [GNU tools and libraries](#)
  - [X window system](#)
- Over 1,000 distros available
- Common free and open-source distros include [Ubuntu](#) and [CentOS](#)



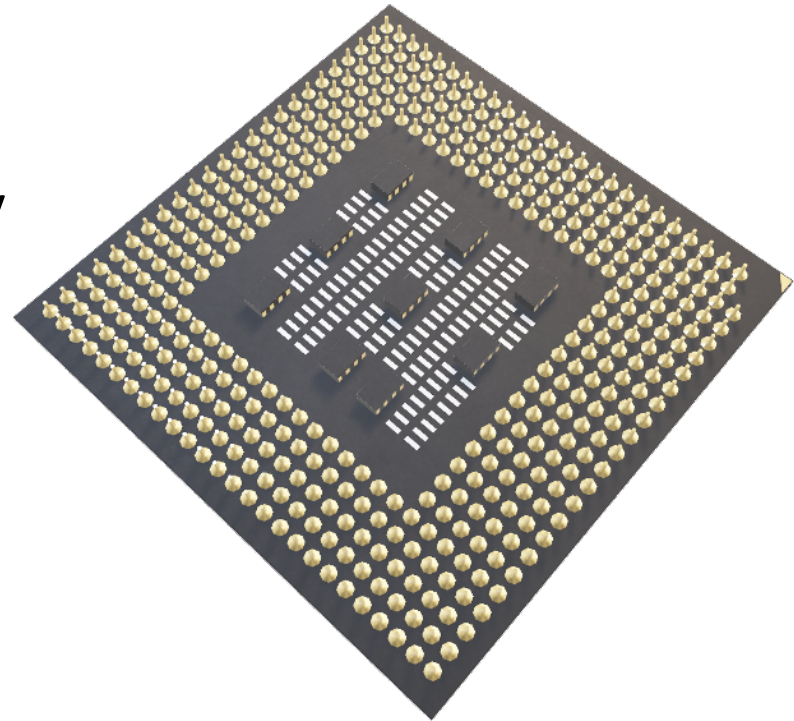
# Linux

## An Introduction



The **Linux Core** (AKA 'kernel') is the OS software in control of the computer hardware

- 1991, created as free replacement for proprietary Unix OS; [kernel.org](https://kernel.org)
- Allows for optimized:
  - parallel computing
  - memory management
  - local/remote filesystems





# Linux

## An Introduction



**GNU Tools and Libraries** is a standard collection of OS programs used by the user installed programs

- ‘GNU’ is a recursive acronym;  
“GNU is Not Unix”
- 1983, [GNU Project](#) developed as free and open-source Unix compatible software
- Example: [GNU C Compilers](#) (GCC) for translating C, C++ code



---

# Linux

## An Introduction



The **X Window System** is software that manages the display

- Framework for drawing and moving graphics on a remote display
- 'X' specifically designed for network connections
- [X.org Foundation](https://www.x.org/) leads free and open-source 'X Project'

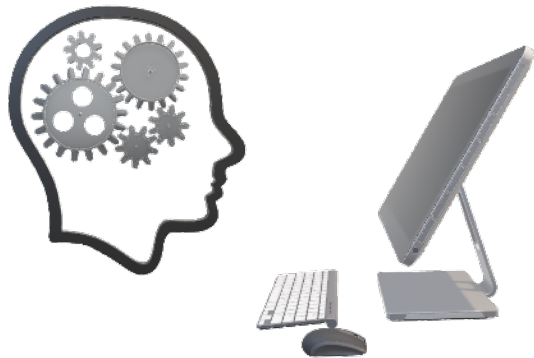


# Linux

## An Introduction

**Three components** for interaction between user and hardware

1. Kernel
2. Shell
3. Terminal Emulator

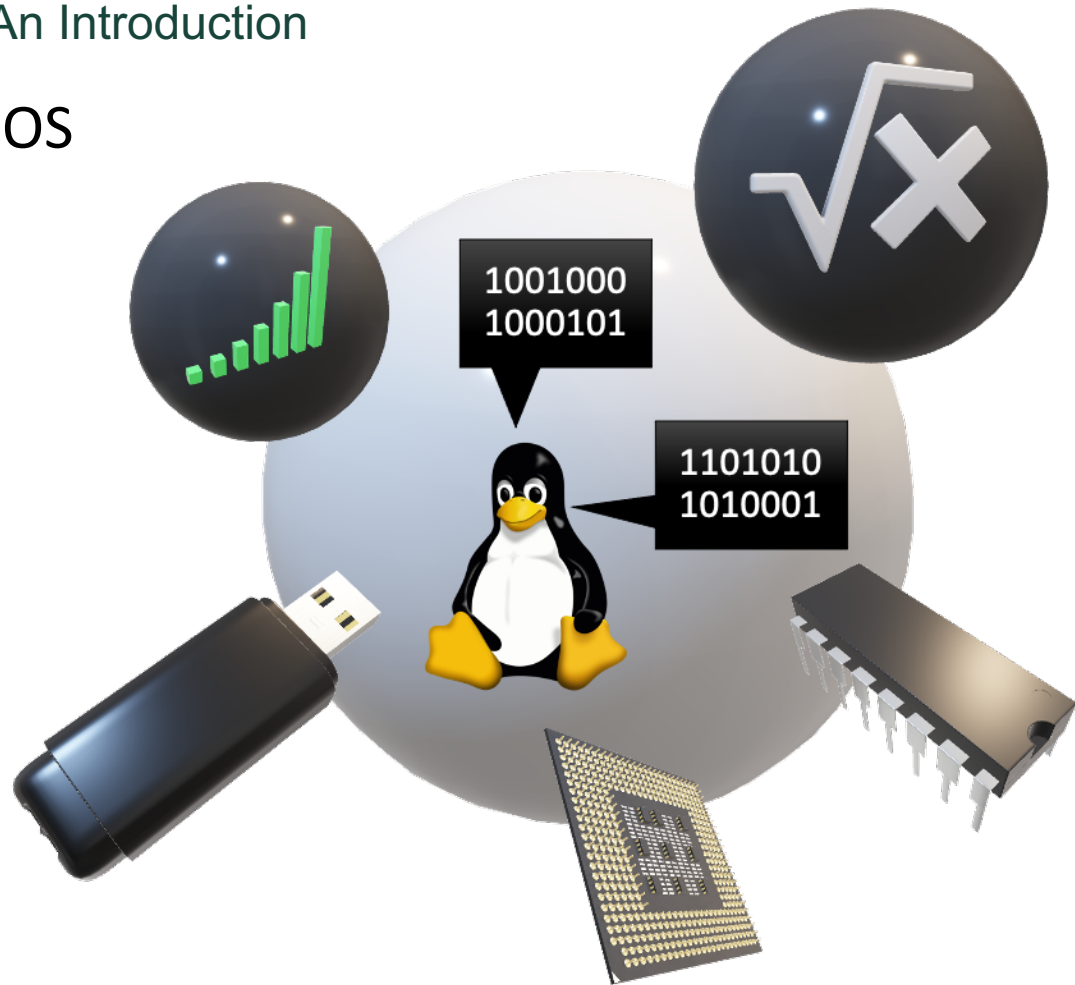


# Linux

## An Introduction

The **kernel** is the core of the OS

- Runs as machine code or 'binary language' layer
- Links instructions from software to computer hardware
- Manages the processors, memory and peripheral devices

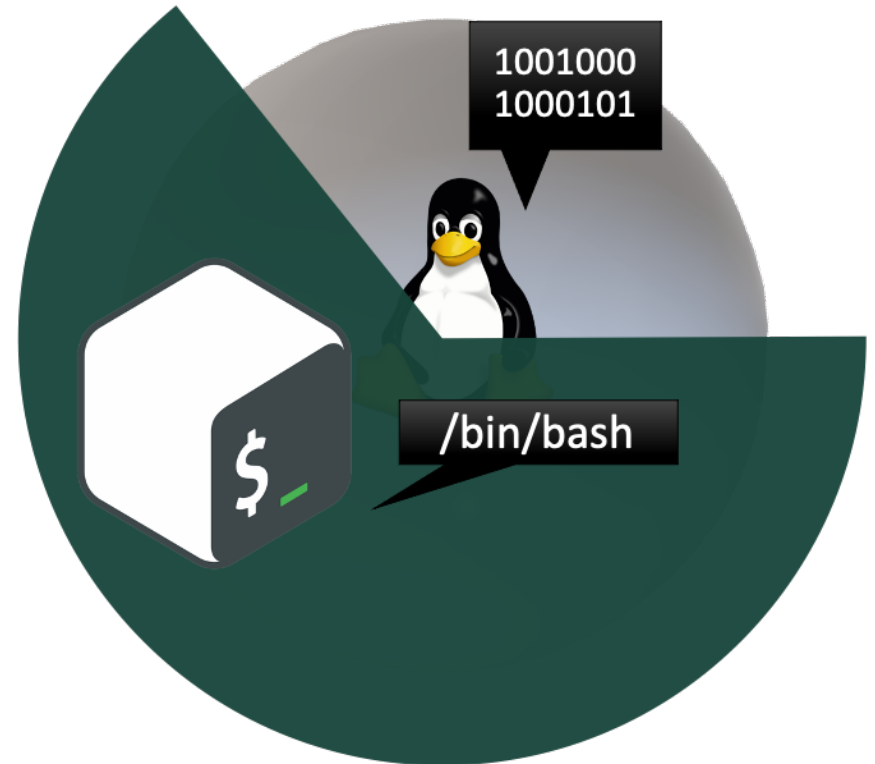


# Linux

## An Introduction

The **shell** is the outer layer of the OS

- Manages instructions from user to kernel
- Both an interactive and scripting language
- 1979, Bourne Shell (sh) developed for easy scripting
- 1989, Bourne-Again Shell (bash) added easy interactive use for GNU OS



# Linux

## An Introduction

The **terminal emulator** application gives the user access to the shell

- Software in combination with keyboard, mouse and display
- Shell access may be local or remote *e.g.*, MSU HPCC
- Emulator may be ‘text terminal’ or graphical user interface (GUI)
- We will emulate a text terminal and use the **command line interface** (CLI)



---

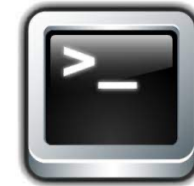
# Linux

## An Introduction

### ICER Recommends these terminal emulators w/ X Windows

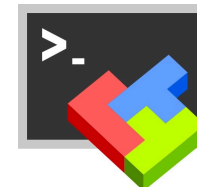
- macOS:

- [Terminal.app](#) (included w/ macOS)
- [Xquartz](#) (installation required)



- MS Windows:

- [MobaXterm](#) (installation required)
- [X Server](#) (included w/ MobaXterm)



# Linux

## An Introduction



**macOS:** Launch the emulated text terminal (AKA '**Terminal**')



- Use the **Finder** app
- Open the **Applications** folder
- Open the **Utilities** folder
- Double-click **Terminal** app





# Linux

## An Introduction



**macOS:** Launch the emulated text terminal (AKA 'Terminal')

- You will see the CLI as a window, with prompt:  
`user@computer ~ $`
- Enter 'commands' in CLI to access the shell



# Linux

## An Introduction



### macOS: Check for the X Window System

- Enter **xeyes** at the command line *i.e.*,

```
user@computer ~ $ xeyes
```

- Check for the 'eyes' window and the 'X' icon in menu bar



# Linux

## An Introduction



### macOS: Check for the X Window System

- If you get the error message...

Error: Can't Open Display

- Download and install 'Xquartz.app' from [Xquartz.org](http://Xquartz.org)



# Linux

An Introduction



## macOS: Verify the Xquartz installation

- Reenter **xeyes** at the command line *i.e.*,

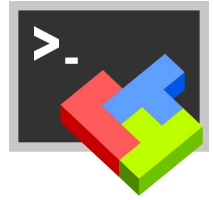
```
user@computer ~ $ xeyes
```

- Check for the 'eyes' window and the 'X' icon in menu bar



# Linux

An Introduction



## Windows: Download and Install [MobaXterm Home Edition \(HE\)](https://mobaxterm.mobatek.net)



- Download **MobaXterm HE - 'Installer Edition'** from:  
[mobaxterm.mobatek.net](https://mobaxterm.mobatek.net)

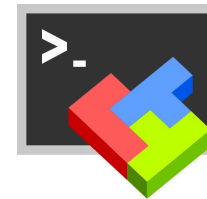


- Right-Click to 'Extract All' from **MobaXterm\_Installer.zip** folder
- Open **MobaXterm\_Installer**
- Double-click **MobaXterm\_Setup**



# Linux

## An Introduction

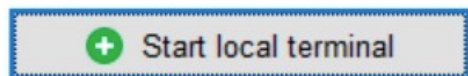


## Windows: Launch MobaXterm Home Edition (HE)

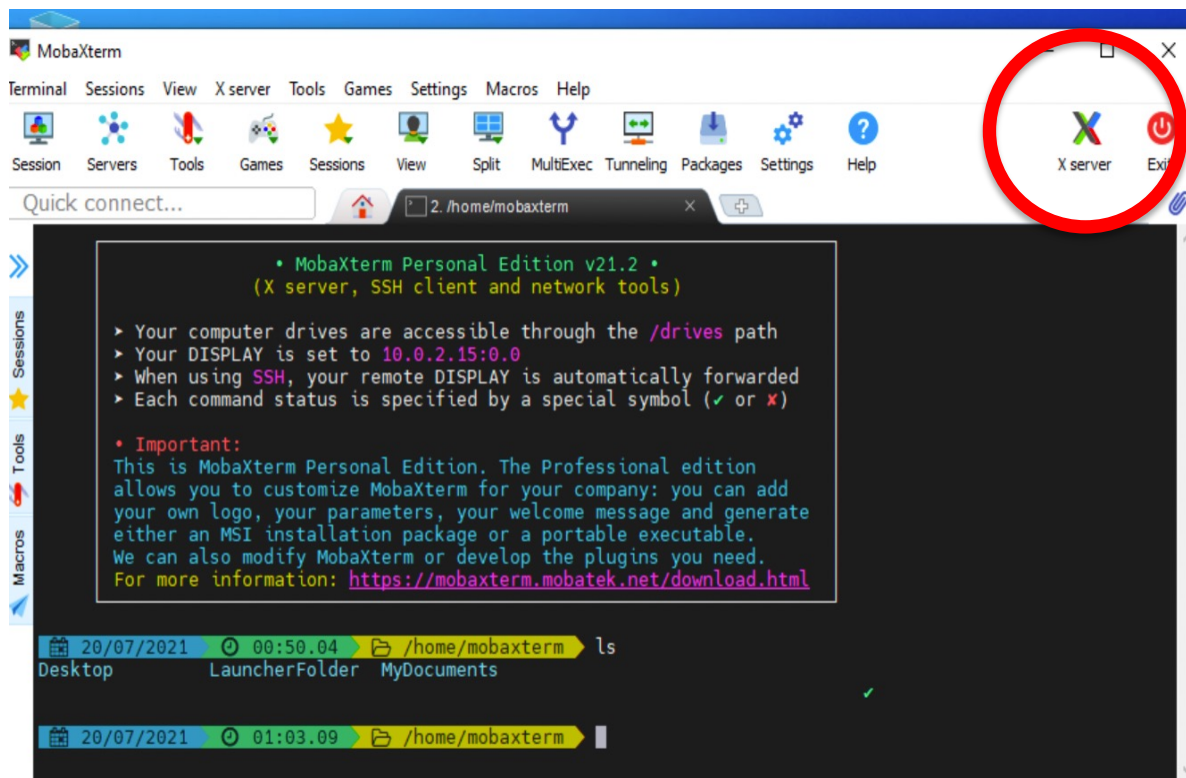
- Double-click **MobaXterm** desktop icon



- Click on button



- Comes with **X Server** installed







# Contact ICER

<https://icer.msu.edu/contact>



INSTITUTE FOR CYBER-ENABLED RESEARCH



# Linux

## The Command Language

**Mahmoud Parvizi, Ph.D.**

Research Consultant, Institute for Cyber-Enabled Research  
Research Associate, Department of Physics and Astronomy

[parvizim@msu.edu](mailto:parvizim@msu.edu)





# Linux

## The Command Language

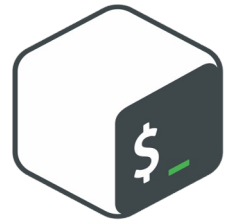
**Linux** is the operating system (OS) for the supercomputer at MSU's high-performance computing center (HPCC)

- User inputs commands through **terminal**; a remote command line interface (CLI) to OS **shell**
- Shell passes these instructions to OS **kernel** that then tasks computer's hardware



# Linux

## The Command Language



The **Command Language** is the syntax employed by user to administer tasks to be passed from shell to kernel



```
Terminal -- bash

Last login: Fri Dec 31 23:59:59

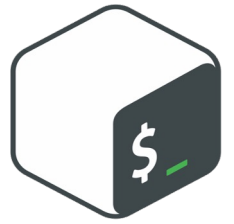
user@computer ~ $ ssh -X sparty21@hpcc.msu.edu
```

- Shell is command language interpreter; bash default on HPCC
- Programming language used interactively and in scripts



# Linux

## The Command Language



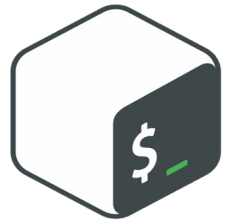
The bash interpreter reads the characters input in the terminal's CLI and groups them into **tokens**

- Sequences of characters interpreted as a unit; separated by a **blank**, *i.e.* 'space' or 'tab'
- May be a single or multiple character sequence



# Linux

## The Command Language



Tokens are then categorized as either a **word** or an **operator** depending on the constituent characters

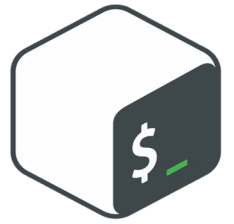
**Word:** sequence of ordinary characters *e.g.*, **a**, **echo** or **my\_file.txt**

**Operator:** sequence of special characters with a specific purpose *e.g.*, **&&** and **||** as logical *and* and *or*



# Linux

The Command Language



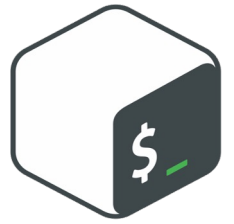
Words DO NOT contain **metacharacters** \*

- Blanks *i.e.*,  
‘space’ *and* ‘tab’
- ‘newline’
- **& , | , ; , ( , ) ,**  
**< , and >**



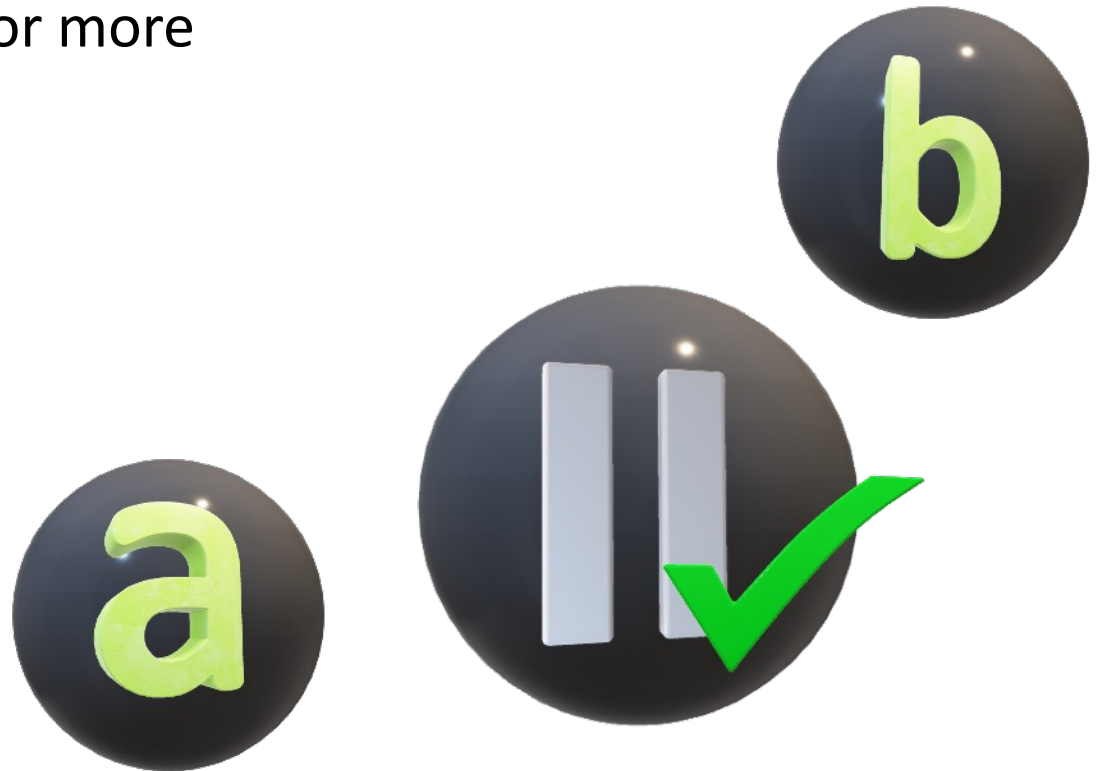
# Linux

## The Command Language



Operators contain at least one **metacharacter** \*

- Sequence of one or more metacharacters
- Examples:
  - ;
  - | |
  - >&
  - 'newline'

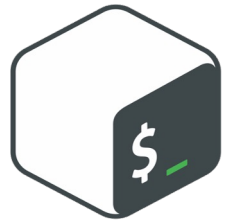


\* Excludes blanks



# Linux

## The Command Language



A sequence of words forms a **simple command**

Terminal -- bash

user@computer ~ \$ ssh -X sparty21@hpcc.msu.edu

- Command to be executed
- Command Option(s)
- Command's Arguments

ssh

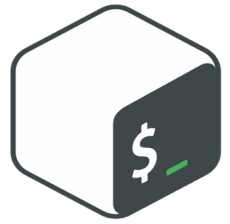
-X

sparty21@hpcc.msu.edu



# Linux

## The Command Language



A **command** uses 'standard streams' for input and output of data

```
Terminal -- bash  
user@computer ~ $ echo "Hello_World"  
Hello_World
```

- **stdin:** default is argument given at CLI
- **stdout** default is write to the display

"Hello\_World"

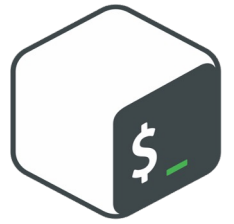
Hello\_World





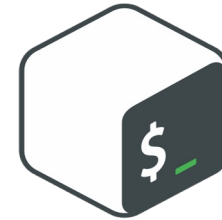
# Linux

## The Command Language



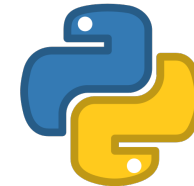
The **command** to be executed may either be **builtin** or **external**

- **Builtin:** command executed internally by the shell *e.g.*, **echo**



```
user@computer $ echo "Hello_World"
```

- **External:** An executable program located elsewhere in the system *e.g.*, **python**



```
user@computer $ python -c "print('Hello_World')"
```



# Linux

## The Command Language



A command **option** (AKA flag or switch) modifies the operation of the command

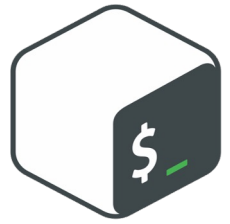
```
user@computer $ ssh -X sparty21@hpcc.msu.edu
```

- Identified by - or -- then a letter or keyword  
*e.g.*, - **h** or -- **help**  
depending on command
- Letters may be combined for brevity *e.g.*,  
- **a** - **b** ↔ - **ab** ↔ - **ba**



# Linux

## The Command Language



Command **arguments** are the constructs used as command input

```
user@computer $ ssh -X sparty21@hpcc.msu.edu
```

- May be no arguments or multiple arguments *e.g.*,

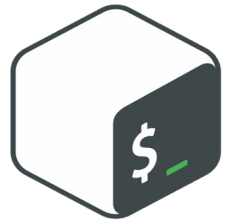
```
sparty21@hpcc $ logout
```

```
user@computer $ echo "Hello" "World"  
Hello World
```



# Linux

## The Command Language



A **pipeline** is sequence of simple commands separated by | operator; output from former is input to latter

- Piped output from **ls** (list) command: '**my\_file.txt**' not displayed
- Input to **wc** (word count) command
- Output from **wc** displayed:  
**1 newline, 1 word, 10 bytes**

```
Terminal -- bash

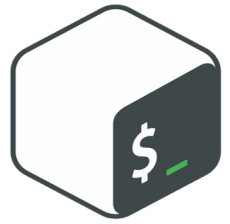
user@computer ~ $ ls
my_file.txt

user@computer ~ $ ls | wc
1 1 10
```



# Linux

## The Command Language



A **list** of commands is a sequence of simple commands or pipelines separated by the **;**, **&&**, or **|** operator

- **;** executes commands sequentially
- **&&** conditional on successful execution of previous
- **|** conditional on unsuccessful execution of previous

```
Terminal -- bash

user@computer ~ $ echo "Hello"; echo "World"
Hello
World
user@computer ~ $
```



# Linux

## The Command Language



**Redirections**, or changing command stdin/stdout from default, carried out by operators `<` and `>` respectively

```
user@computer $ echo "Hello_World" > my_file.txt
```

- `>` creates a file named **my\_file.txt** containing output **Hello\_World**

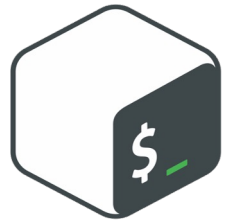
```
user@computer $ echo "I_Am_Here" >> my_file.txt
```

- `>>` appends the output **"I\_Am\_Here"** to the file **my\_file.txt**



# Linux

## The Command Language



An **error** occurs when a command fails to execute successfully; streams message to standard error (**stderr**)

- **stderr**: Error message stream writes to display by default

```
user@computer $ ssh "Hello World"  
ssh: Could not resolve hostname hello world: Name or service not known  
user@computer $
```

- May be redirected using **>&** to **my\_error.txt**

```
user@computer $ ssh "Hello World" >& my_error.txt  
user@computer $
```



# Linux

## The Command Language



To **group** a list of commands, use the **( and )** operators

```
user@computer $(echo "Hello"; echo "World") > my_file.txt ;  
echo "I_Am_Here" >> my_file.txt
```

- First creates a file named **my\_file.txt** containing output **"Hello World"**
- Then appends the output **"I\_Am\_Here"** to the file **my\_file.txt**





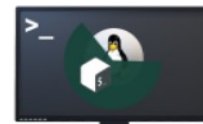
# Linux

The Command Language



## Learning the bash commands

- Remember from training
- Ask your lab mate
- Ask the web
- Consult [docs.icer.msu.edu](https://docs.icer.msu.edu)
- Use the OS **manual pager**



# Linux

## The Command Language



The system's **manual pager**, or **man page**, is the OS embedded reference manual

- Man pages divided into nine sections
- Section 1 contains the shell command pages
- Manual Command; **man** *command name* displays synopsis, description, and options

```
Terminal -- bash

user@computer ~ $ man ssh

SSH(1)                                BSD General Commands Manual                                SSH(1)
NAME
ssh - OpenSSH SSH client (remote login program)
SYNOPSIS
ssh [-1246AaCfGgKkMNnqsTtVvXxYy] [-b bind_address] [-c cipher_spec] [-D [bind_address:]port] [-E log_file]
[-e escape_char] [-F configfile] [-I pkcs11] [-i identity_file] [-J [user@]host[:port]] [-L address]
[-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port] [-Q query_option] [-R address]
[-S ctl_path] [-W host:port] [-w local_tun[:remote_tun]] [user@]hostname [command]
DESCRIPTION
ssh (SSH client) is a program for logging into a remote machine and for executing commands on a remote machine.
It is intended to provide secure encrypted communications between two untrusted hosts over an insecure network.
X11 connections, arbitrary TCP ports and UNIX-domain sockets can also be forwarded over the secure channel.

ssh connects and logs into the specified hostname (with optional user name). The user must prove his/her identity
to the remote machine using one of several methods (see below).

If command is specified, it is executed on the remote host instead of a login shell.

The options are as follows:

-1      Forces ssh to try protocol version 1 only.
-2      Forces ssh to try protocol version 2 only.
-4      Forces ssh to use IPv4 addresses only.
-6      Forces ssh to use IPv6 addresses only.
-A      Enables forwarding of the authentication agent connection. This can also be specified on a per-host
basis in a configuration file.

Agent forwarding should be enabled with caution. Users with the ability to bypass file permissions on
Manual page ssh(1) line 1 (press h for help or q to quit)
```



# Linux

## The Command Language



There is a **man page** for the **manual pager**

- Use **man** as both command and argument
- Details the sections, lists all man page options and provides some helpful examples

Terminal -- bash

```
user@computer ~ $ man man
```

```
MAN(1)                                Manual pager utils                                MAN(1)

NAME
    man - an interface to the on-line reference manuals

SYNOPSIS
    man [-C file] [-d] [-D] [--warnings=warnings] [-R encoding] [-L locale] [-m system[...]] [-M path] [-S
    list] [-e extension] [-i -I] [--regex|--wildcard] [--names-only] [-a] [-u] [--no-subpages] [-P pager] [-r
    prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justification] [-p string] [-t] [-Tdevice] [-Hbrowser]
    [-Xdpi] [-Z] [[section] page ...] ...
    man -k [apropos options] regexp ...
    man -K [-w -W] [-S list] [-i -I] [--regex] [section] term ...
    man -f [whatIs options] page ...
    man -l [-C file] [-d] [-D] [--warnings=warnings] [-R encoding] [-L locale] [-P pager] [-r prompt] [-7] [-E
    encoding] [-p string] [-t] [-Tdevice] [-Hbrowser] [-Xdpi] [-Z] file ...
    man -w -W [-C file] [-d] [-D] page ...
    man -c [-C file] [-d] [-D] page ...
    man [-?V]

DESCRIPTION
    man is the system's manual pager. Each page argument given to man is normally the name of a program, utility
    or function. The manual page associated with each of these arguments is then found and displayed. A section,
    if provided, will direct man to look only in that section of the manual. The default action is to search in
    all of the available sections, following a pre-defined order and to show only the first page found, even if
    page exists in several sections.

    The table below shows the section numbers of the manual followed by the types of pages they contain.

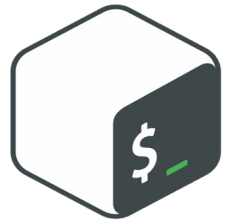
    1 Executable programs or shell commands
    2 System calls (functions provided by the kernel)
    3 Library calls (functions within program libraries)
    4 Special files (usually found in /dev)
    5 File formats and conventions eg /etc/passwd
    6 Games
    7 Miscellaneous (including macro packages and conventions), e.g. man(7), groff(7)

    Manual page man(1) line 1 (press h for help or q to quit)
```



# Linux

The Command Language



Search the **man pages** by keywords

```
Terminal -- bash  
user@computer ~ $ man -k keyword(s)
```

- Command

**man**

- Option

**-k**

- Argument

**Keyword(s)**



# Linux

## The Command Language



**Example:** Search the **man pages** for the command to login into the MSU HPCC

```
Terminal -- bash

user@computer ~ $ man -k login -s 1

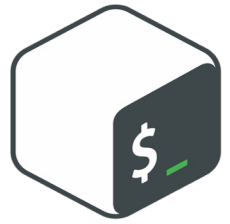
chsh (1)          - change your login shell
git-shell (1)     - Restricted login shell for Git-only SSH access
lchsh (1)         - Change login shell
login (1)         - begin session on the system
loginctl (1)      - Control the systemd login manager
logname (1)       - print user's login name
lslogins (1)      - display information about known users in the system
mate-session-properties (1) - Configure applications to start on login.
singularity-remote-login (1) - Log into a singularity remote endpoint, an OCI/Docker registry or a keyserver using cr...
slogin (1)        - OpenSSH SSH client (remote login program)
ssh (1)           - OpenSSH SSH client (remote login program)
ssh-copy-id (1)   - use locally available keys to authorise logins on a remote machine
```

- Use **-s 1** to restrict search to Section 1 - Shell Commands



# Linux

## The Command Language



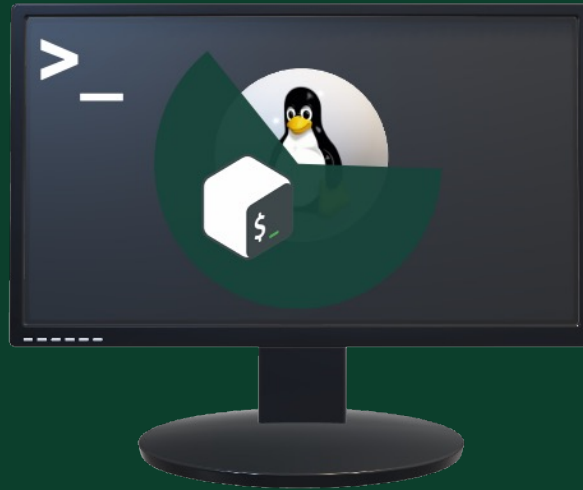
Use the **help** option ( **-h** or **--help** ) if you already know the command name

```
Terminal -- bash

user@computer ~ $ wc --help

Usage: wc [OPTION]... [FILE]...
  or: wc [OPTION]... --files0-from=F
Print newline, word, and byte counts for each FILE, and a total line if
more than one FILE is specified.  With no FILE, or when FILE is -,
read standard input.  A word is a non-zero-length sequence of characters
delimited by white space.
The options below may be used to select which counts are printed, always in
the following order: newline, word, character, byte, maximum line length.
  -c, --bytes          print the byte counts
  -m, --chars          print the character counts
  -l, --lines          print the newline counts
  --files0-from=F      read input from the files specified by
                        NUL-terminated names in file F;
                        If F is - then read names from standard input
  -L, --max-line-length print the length of the longest line
  -w, --words          print the word counts
  --help              display this help and exit
  --version            output version information and exit
```





# HPCC wiki

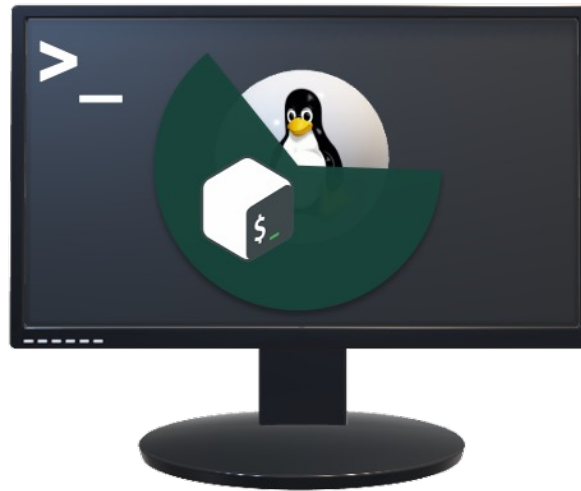
[docs.icer.msu.edu](https://docs.icer.msu.edu)

# Contact ICER

[icer.msu.edu/contact](https://icer.msu.edu/contact)



INSTITUTE FOR CYBER-ENABLED RESEARCH



# Linux

## Linux on the HPCC

**Mahmoud Parvizi, Ph.D.**

Research Consultant, Institute for Cyber-Enabled Research  
Research Associate, Department of Physics and Astronomy

[parvizim@msu.edu](mailto:parvizim@msu.edu)



---

# Linux

Linux on the HPCC

## The MSU High Performance Computing Center (HPCC)

- **Clusters** of individual computers, or **nodes**, connected via a very fast **data interconnect**
- A **clustered filesystem**, a **module system** and a **job scheduler** manage data and perform tasks on the system



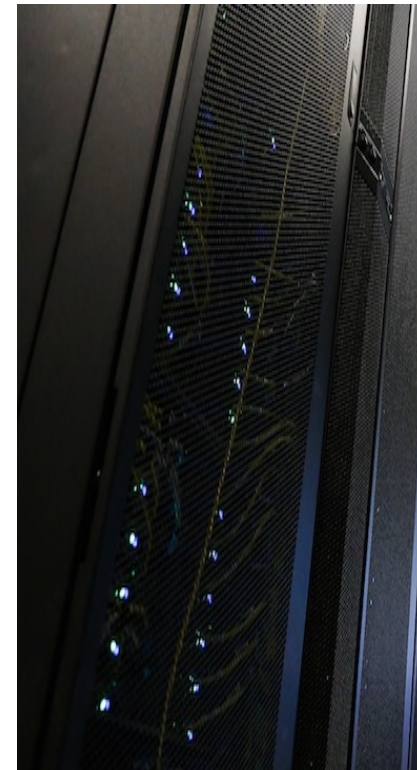
---

# Linux

Linux on the HPCC

## HPCC Hardware Definitions:

- **Cluster:** Set of computers that form a single system; comprises a network (~4 clusters)
- **Node:** Individual computer in a cluster; performs a task (~1,000 nodes)
- **Core:** Individual central processing unit, or CPU, in a node; performs computation (~60,000 CPUs, ~300 Tb memory)



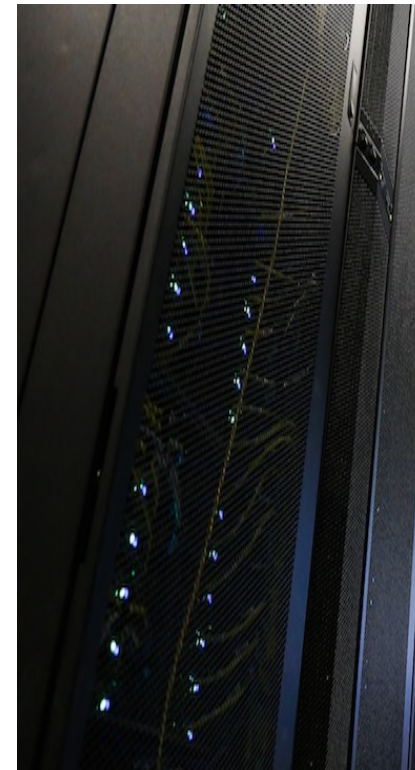
---

# Linux

Linux on the HPCC

## HPCC Hardware Definitions:

- **Data Interconnect:** Networking communications standard with high throughput and low latency
- **Infiniband:** HPCC's data interconnect (~100Gb/s, ~8Pb storage)

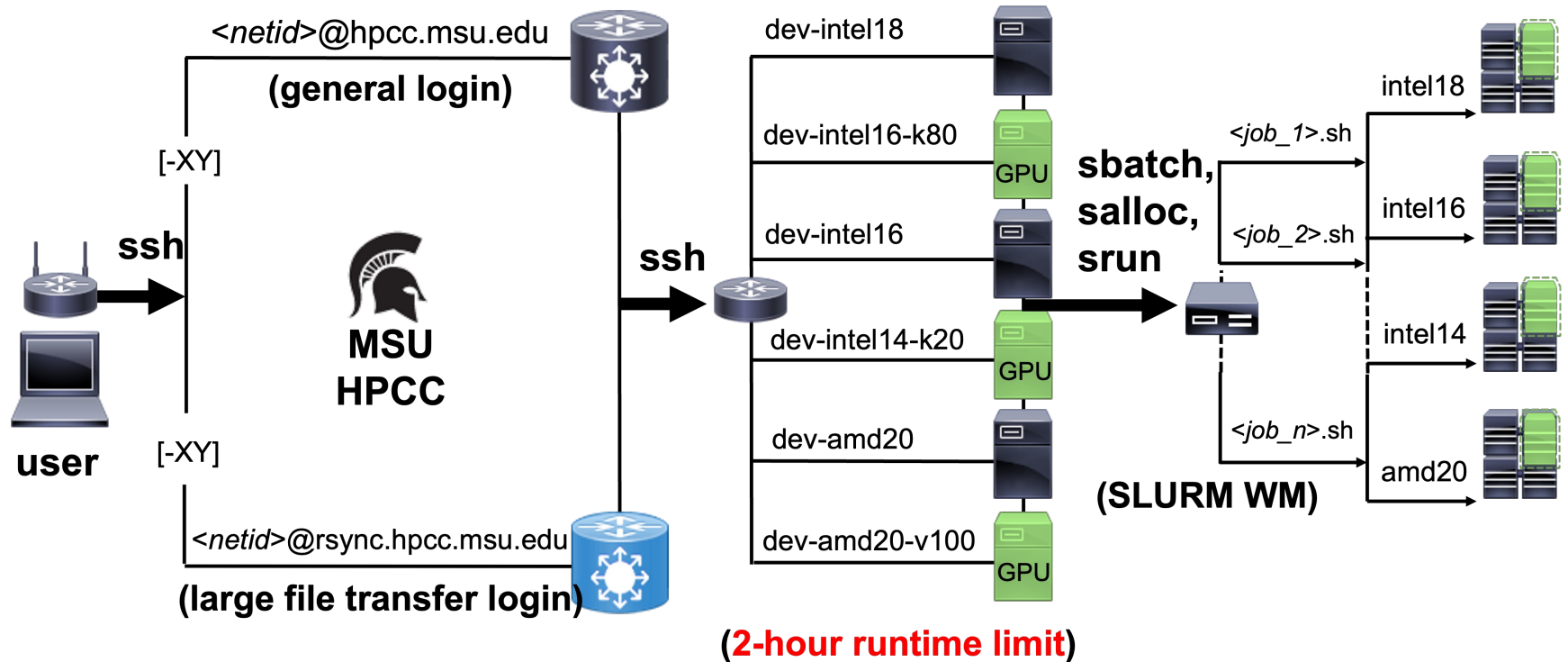


# Linux

Linux on the HPCC

## HPCC System Overview

Terminal ..... Gateways ..... Dev Nodes ..... Clusters



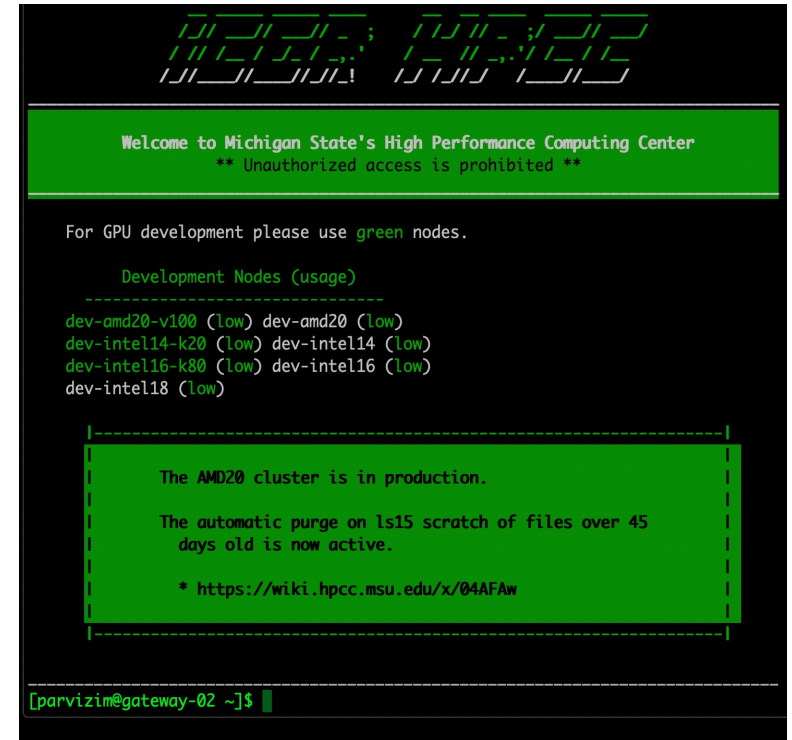
# Linux on the HPC

## Exercise: Login to the HPCC



## Type in Your Terminal:

```
[user@computer] $ ssh -X <NetID>@hpcc.msu.edu
```



# Linux

Linux on the HPCC

**Exercise:** Login to dev-intel18



**Type in Your Terminal:**

**[user@computer] \$ ssh -X dev-intel18**

```
parvizim@dev-intel18:~ — ssh -XY parvizim@hpcc.msu.edu — 80x52

-----
[parvizim@gateway-01 ~]$ ssh dev-intel18
Last login: Mon Jul 20 10:47:36 2020 from gateway-02.dmz

===
Please note that processes on development nodes are limited to two hours of
CPU time; for longer-running jobs, please submit to the queue.

Development nodes are a shared system; for information about performance
considerations please see: https://wiki.hpcc.msu.edu/x/N4JnAg
===

[parvizim@dev-intel18 ~]$
```



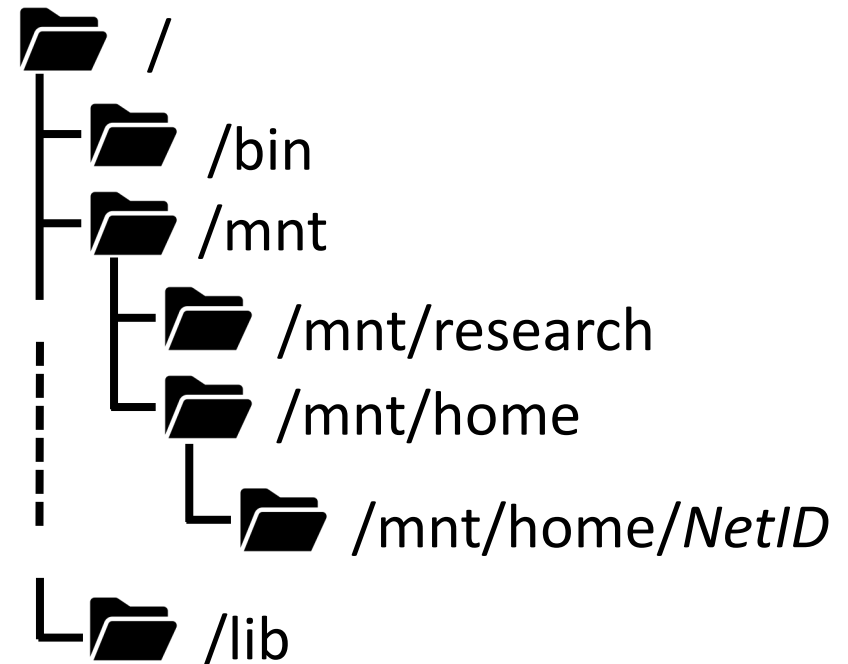


# Linux

Linux on the HPCC

A **clustered filesystem** is a hierarchical collection of files accessible to all compute nodes of a cluster

- **File:** A formatted collection of bytes referenced by the OS
- **Directory:** Any file containing another file
- **Filesystem:** Method use by OS to store and retrieve files

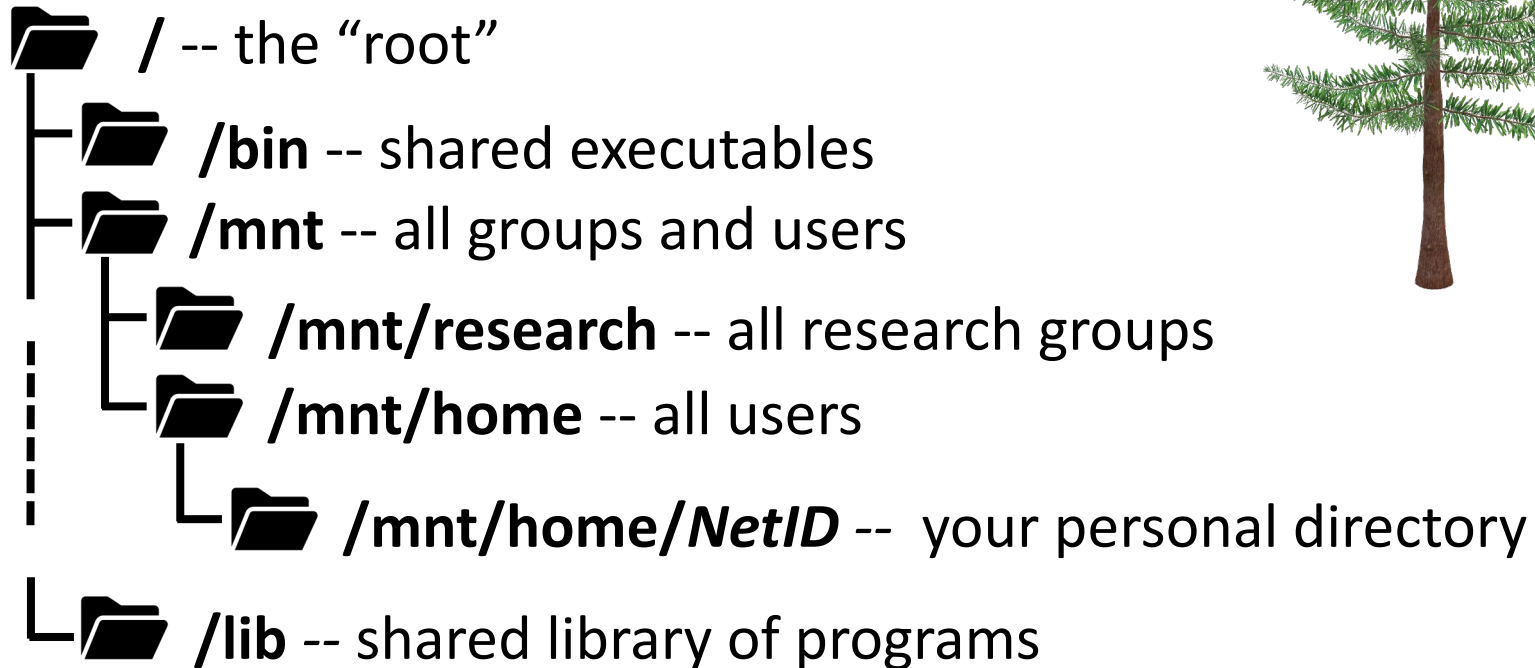


# Linux

Linux on the HPCC

Directories have a **tree-like** structure

- **Examples:**



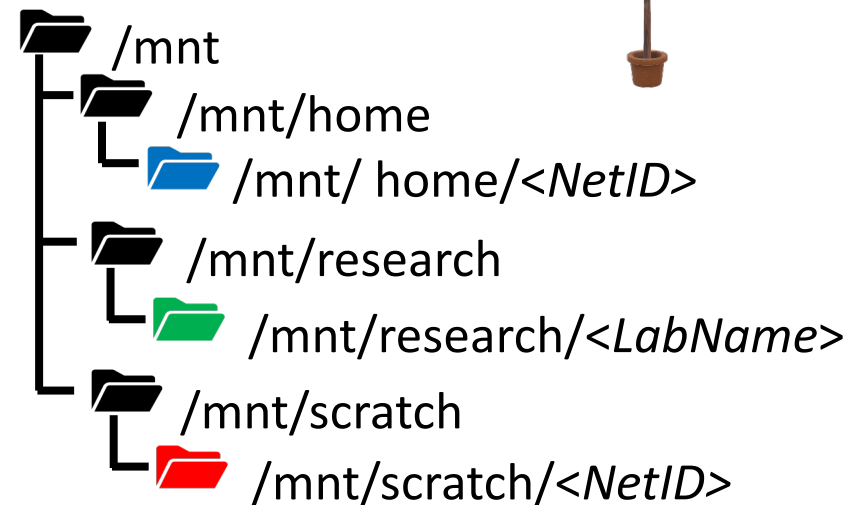


# Linux

Linux on the HPCC

**User Spaces** are directories users can write files to using any nodes in a cluster

- **Home:** Personal files and default login directory (50Gb storage)
- **Research:** Group research files (50Gb - 1Tb storage)
- **Scratch:** Temporary working files (~800 Tb total storage)



# Linux

## Linux on the HPCC

**Exercise:** use the **quota** command to display the details of your user spaces



Type in Your Terminal:

**[user@computer] \$ quota**

```
parvizim@dev-intel18:~ - ssh parvizim@hpcc.msu.edu - 123x49
Development nodes are a shared system; for information about performance
considerations please see: https://wiki.hpcc.msu.edu/x/M4JnAg
-----
[parvizim@dev-intel18 ~]$ quota
Home Directory:
Files
Space      Space      Space      Space      Files      Files      Files
Quota      Used      Available  % Used      Quota      Used      Available
-----
/mnt/home/parvizim 50G       7G        43G        14%        1048576    87299     961277
8%

Research Groups:
Files
Space      Space      Space      Space      Files      Files      Files
Quota      Used      Available  % Used      Quota      Used      Available
-----
TOPMED          4096G     3733G     363G       91%        4194304    1558      4192746
0%
UKBB            9216G     8242G     974G       89%        9437184    6897      9430287
0%
helpdesk        12288G    9888G     2400G      80%        52428800   44415439  8013361
85%

Temporary Filesystems:
-----
/mnt/scratch (/mnt/gsl8) Space Quota Space Used Space Free Space % Used Filess Quota Files Used Files Free
Files % Used
51200G 1G 51199G 0% 1048576 2 1048574
0%

/mnt/lsl15 (legacy scratch) Inodes Used Quota Free
1 1000000 999999

[parvizim@dev-intel18 ~]$
```

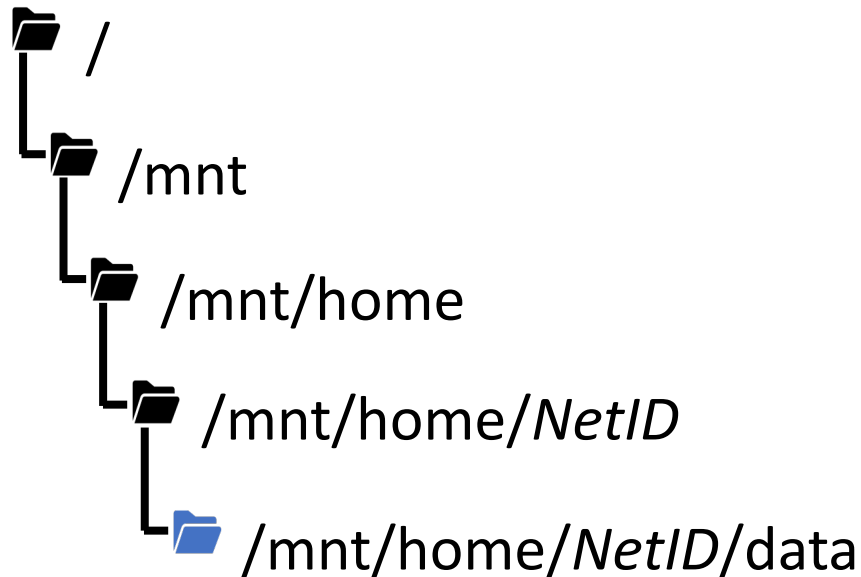


# Linux

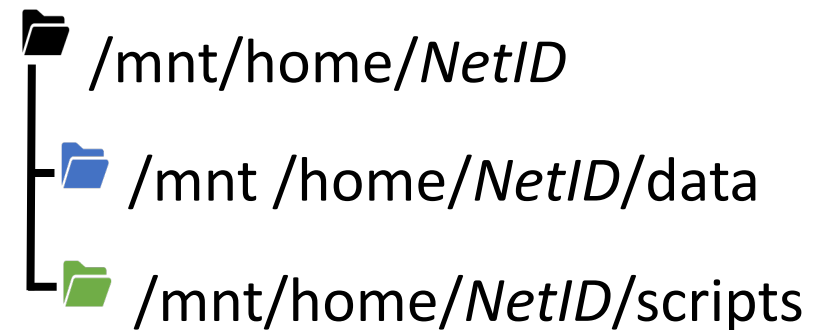
Linux on the HPCC

A **path to a file** is a list of the files containing the file of interest

Absolute `/mnt/home/NetID/data`



Relative `../scripts`



---

# Linux

Linux on the HPCC

**Exercise:** Find path with **pwd**, or print working directory, command



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ pwd  
/mnt/home/parvizim  
[parvizim@dev-intel18 ~]$
```

Type in Your Terminal:

[user@computer] \$ pwd



---

# Linux

Linux on the HPCC

**Exercise:** List files with **ls**, or list information, command



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ ls  
Documents  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ ls**



---

# Linux

Linux on the HPCC

**Exercise:** Enter 'Documents' with **cd**, or change directory, command and list the contents



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ cd Documents; ls  
MATLAB  RT  Workshops  
[parvizim@dev-intel18 Documents]$
```

**Type in Your Terminal:**

**[user@computer] \$ cd Documents; ls**



# Linux

Linux on the HPCC

**Exercise:** Create a file 'test.txt' with the **cat**, or concatenate, command and the > operator



**Type in Your Terminal:**

**[user@computer] \$ cat > test.txt**

```
[parvizim@dev-intel18 Documents]$  
[parvizim@dev-intel18 Documents]$  
[parvizim@dev-intel18 Documents]$ cat > test.txt  
My intro2linux textfile  
the second line  
^Z  
[1]+  Stopped                  cat > test.txt  
[parvizim@dev-intel18 Documents]$
```

- 1) Type text: *My intro2Linux textfile* [return]
- 2) Type text: *the second line* [return]
- 3) Stop cat: [control]+[Z]



# Linux

Linux on the HPCC

**Exercise:** Edit our text file with the **vi**, or vim editor, command



**Type in Your Terminal:**

**[user@computer] \$ vi test.txt**

```
class2@dev-intel18:~/Docum...
My intro2Linux textfile
This is the second line
~
~
~
-- INSERT -- 2,8 All
```

```
class2@dev-intel18:~/Docum...
My intro2Linux textfile
This is the second line
~
~
~
:wq
```

- 1) [ i ], look for the – INSERT – banner at bottom
- 2) Add text: *“This is”* to *“the second line”*
- 3) [esc], – INSERT – banner gone
- 4) [shift]+[ :] and look for : at bottom
- 5) Type *wq* to write and quit





---

# Linux

Linux on the HPCC

**Exercise:** Change path back to 'home' and make a new directory with **mkdir**, or make directory, command



```
[parvizim@dev-intel18 Documents]$ cd ../; mkdir MyIntro2Linux; ls  
MyIntro2Linux  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ cd ../; mkdir MyIntro2Linux; ls**



---

# Linux

Linux on the HPCC

**Exercise:** Move the 'test.txt' file from 'Documents' to the new directory with **mv**, or move, command



```
[parvizim@dev-intel18 ~]$ mv ~/Documents/test.txt ~/MyIntro2Linux  
[parvizim@dev-intel18 ~]$ cd MyIntro2Linux/; ls  
test.txt  
[parvizim@dev-intel18 MyIntro2Linux]$
```

**Type in Your Terminal:**

```
[user@computer] $ mv ~/Documents/test.txt ~/MyIntro2Linux
```



# Linux

Linux on the HPCC

**File Permissions** determine which users may access, modify, and/or execute files

## Permissions

- 

<b>r</b>	<b>w</b>	<b>x</b>
----------	----------	----------

<b>r</b>	-	-
----------	---	---

-	-	-
---	---	---

 (default)

                    ↑                    ↑                    ↑

                    Owner                    Group                    All Users

**r** = Read  
**w** = Write  
**x** = Execute



# Linux

Linux on the HPCC

**chmod**, or change mode, command is used to change file permissions

**chmod** <user> <action> <permission> <FileName>

**u** = Owner  
**g** = Group  
**o** = Others  
**a** = All (u,g,o)

**+** Give  
**-** Take

**r** = Read  
**w** = Write  
**x** = Execute



# Linux

Linux on the HPCC

**Exercise:** list the permissions for 'test.txt' by using the **ls** command with the **-l**, or long, option



```
[parvizim@dev-intel18 MyIntro2Linux]$  
[parvizim@dev-intel18 MyIntro2Linux]$ ls -l  
total 1  
-rw-r----- 1 parvizim helpdesk 40 Jan 24 01:10 test.txt  
[parvizim@dev-intel18 MyIntro2Linux]$
```

-	<div style="border: 1px solid blue; padding: 2px; display: inline-block;">r w -</div>	<div style="border: 1px solid blue; padding: 2px; display: inline-block;">r - -</div>	<div style="border: 1px solid blue; padding: 2px; display: inline-block;">- - -</div>
	↑	↑	↑
	Owner	Group	All Users

**Type in Your Terminal:**

**[user@computer] \$ ls -l**



# Linux

Linux on the HPCC

**Exercise:** Change permissions of 'test.txt' to allow group members to write to the file



```
[parvizim@dev-intel18 MyIntro2Linux]$ chmod g+w test.txt; ls -l  
total 1  
-rw-rw---- 1 parvizim helpdesk 40 Jan 24 01:10 test.txt  
[parvizim@dev-intel18 MyIntro2Linux]$
```

chmod      g + w      test.txt

g = Group

+ Give

w = Write

Type in Your Terminal:

```
[user@computer] $ cd MyIntro2Linux/; chmod g+w test.txt; ls -l
```



---

# Linux

Linux on the HPCC

**Exercise:** Remove the 'test.txt' file from 'MyIntro2Linux' with **rm**, or remove, command



```
[parvizim@dev-intel18 MyIntro2Linux]$  
[parvizim@dev-intel18 MyIntro2Linux]$ rm test.txt; ls  
[parvizim@dev-intel18 MyIntro2Linux]$
```

**Type in Your Terminal:**

**[user@computer] \$ rm test.txt; ls**



---

# Linux

Linux on the HPCC

**Exercise:** Remove the directory 'MyIntro2Linux' with **rmdir**, or  
remove directory, command



```
[parvizim@dev-intel18 MyIntro2Linux]$  
[parvizim@dev-intel18 MyIntro2Linux]$ cd ~; rmdir MyIntro2Linux  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ cd ~; rmdir MyIntro2Linux**





# Linux

## Linux on the HPCC

**Exercise:** Open a 2<sup>nd</sup> terminal, create file 'my.txt', and copy it to the HPCC with **scp**, or secure copy, command



```
parvizim -- zsh -- 106x24
Last login: Tue Jan 24 01:42:12 on ttys001
parvizim@ICERpro84 ~ % cat > my.txt
stuff
^Z
zsh: suspended cat > my.txt
parvizim@ICERpro84 ~ % scp my.txt parvizim@hpcc.msu.edu:~/Documents/my_copy.txt
Enter passphrase for key '/Users/parvizim/.ssh/id_rsa':
my.txt                               100%   6    0.2KB/s   00:00
parvizim@ICERpro84 ~ %
```

**Type in Your Terminal:**

**[user@computer] \$ scp my.txt <NetID>@hpcc.msu.edu:~/Documents/my\_copy.txt**



# Linux

## Linux on the HPCC

**Exercise:** Copy file 'my\_copy.txt', from the HPCC with **scp**, or secure copy, command



```
parvizim@ICERpro84 ~ %  
parvizim@ICERpro84 ~ % scp parvizim@hpcc.msu.edu:~/Documents/my_copy.txt ./  
Enter passphrase for key '/Users/parvizim/.ssh/id_rsa':  
my_copy.txt                                100%    6    0.1KB/s   00:00  
parvizim@ICERpro84 ~ %
```

**Type in Your Terminal:**

**[user@computer] \$ scp <NetID>@hpcc.msu.edu:~/Documents/my\_copy.txt ./**



# Linux

## Linux on the HPCC

**Exercise:** Use **sftp**, or secure file transfer protocol, to move files to and from the HPCC



**Type in Your Terminal:**

```
sftp> lpwd
Local working directory: /Users/mahmoudparvizi
sftp> pwd
Remote working directory: /mnt/ufs18/home-075/parvizim
sftp> cd Documents
sftp> pwd
Remote working directory: /mnt/ufs18/home-075/parvizim/Documents
sftp> get my_copy.txt
Fetching /mnt/ufs18/home-075/parvizim/Documents/my_copy.txt to my_copy.txt
sftp> put my.txt
Uploading my.txt to /mnt/ufs18/home-075/parvizim/Documents/my.txt
my.txt
sftp> quit
mahmoudparvizi@ShredBook-Pro ~ %
```

1. Use **lpwd** for local; **pwd** for HPCC
2. **cd** to your HPCC “Documents” directory
3. Use **get** my\_copy.txt to copy *from* HPCC
4. Use **put** my.txt to copy *to* HPCC
5. Use **quit** to close sftp connection

**[user@computer] \$ sftp <NetID>@hpcc.msu.edu**



# Linux

## Linux on the HPCC

**Exercise:** Download 'iris' dataset from the web to HPCC with **wget**, or web get, command



```
[parvizim@dev-intel18 Documents]$ wget https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data
--2021-06-04 01:45:59-- https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data
Resolving archive.ics.uci.edu (archive.ics.uci.edu)... 128.195.10.252
Connecting to archive.ics.uci.edu (archive.ics.uci.edu)|128.195.10.252|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4551 (4.4K) [application/x-httpd-php]
Saving to: 'iris.data'

100%[=====>] 4,551 --.-K/s in 0s

2021-06-04 01:46:00 (41.1 MB/s) - 'iris.data' saved [4551/4551]

[parvizim@dev-intel18 Documents]$
```

**Type in Your Terminal:**

```
[user@computer] $ wget https://archive.ics.uci.edu/ml/machine-learning-databases/iris/iris.data
```



# Linux

Linux on the HPCC

File transfer GUI **Globus** provided by MSU

<https://www.globus.org/data-transfer>

 **globus**  
a schlegel non-profit service

I Want To... ▾ Pricing ▾ Resources ▾ Support ▾ About ▾ [Log In](#)

## Data Transfer With Globus

Globus provides a secure, unified interface to your research data. Use Globus to 'fire and forget' high-performance data transfers between systems within and across organizations.



Research often requires sophisticated data management capabilities across systems and institutions. Globus was built to provide these capabilities. Laptops, supercomputers, tape archives, cloud storage, HPC clusters, and scientific instruments are some of the systems that can be connected to Globus, as well as cloud storage like [Google Drive](#) and [Amazon S3](#).

Globus lets you use a web browser or command line interface to submit transfer and synchronization requests, optionally choosing encryption. Globus takes it from there. With this 'fire and forget' model you can concentrate on your research while Globus handles the mundane (but important!) details of successful large-scale data transfers, even for [protected data](#) like HIPAA-regulated data.

Your data is transferred directly between the source and destination systems while Globus tunes performance parameters, maintains security, monitors progress, and validates

### Why Subscribe?

Find out why so many leading organizations are Globus subscribers.

### Get Started

Move files now, start a free trial, create storage endpoints, and more.

### What's New

**Globus Connect Server v5.4 released:** Check out the new features

**State of the Craft in Research Data Management:** Read our latest article in Science Node

**Globus at Argonne:** Researchers use Theta for real-time analysis of COVID-19 proteins

*"The system is reliable and secure – and also amazingly easy to use. It just*



---

# Linux

Linux on the HPCC

The HPCC **module system** manages the software environment you need to run your computations

- **Environment:** User specified software applications and their dependencies
- **Dependency:** Any file needed by an executable software application
- **Module:** User loaded file(s) that comprise an environment



---

# Linux

Linux on the HPCC

Examples of HPCC modules include **compilers** and **libraries**

- **Compiler:** Software that translates code e.g., source to machine (GCC, intel, CUDA)
- **Library:** Collection of software resources used by the compiler and other executables; e.g., Math (BLAS, LaPACK)



# Linux

## Linux on the HPCC

**Exercise:** List default HPCC modules with the **module list** command



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ module list  
  
Currently Loaded Modules:  
  1) GCCcore/6.4.0      7) OpenBLAS/0.2.20      13) CMake/3.11.1      19) libffi/3.2.1  
  2) binutils/2.28      8) FFTW/3.3.7          14) ncurses/6.0        20) Python/3.6.4  
  3) GNU/6.4.0-2.28     9) ScaLAPACK/2.0.2-OpenBLAS-0.2.20 15) libreadline/7.0    21) Java/1.8.0_152  
  4) OpenMPI/2.1.2     10) bzip2/1.0.6         16) Tcl/8.6.8          22) MATLAB/2018a  
  5) tbb/2018_U3        11) zlib/1.2.11        17) SQLite/3.21.0  
  6) imkl/2018.1.163    12) Boost/1.67.0       18) GMP/6.1.2
```

**Type in Your Terminal:**

**[user@computer] \$ module list**





# Linux

## Linux on the HPCC

**Exercise:** Find specific modules with the **module spider** command



**Type in Your Terminal:**

**[user@computer] \$ module spider R**

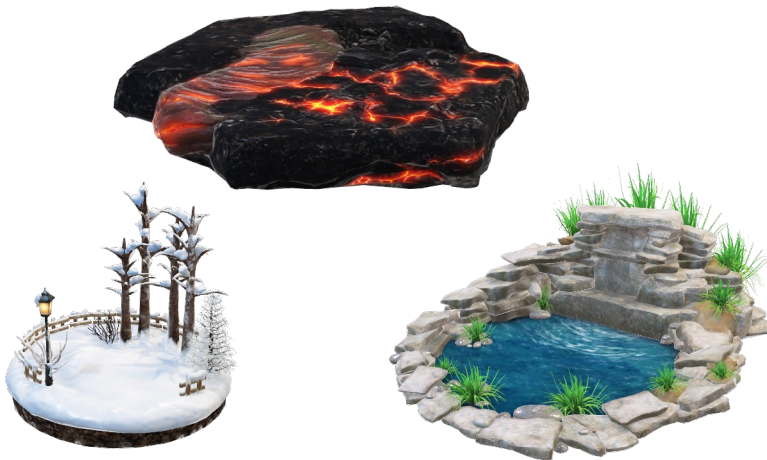
```
[[parvizim@dev-intel18 ~]]$  
[[parvizim@dev-intel18 ~]]$ module spider R  
  
-----  
R:  
-----  
Description:  
  R is a free software environment for statistical computing and graphics.  
  
Versions:  
  R/3.3.1  
  R/3.4.3-X11-20160819  
  R/3.4.3-X11-20171023  
  R/3.4.3xF  
  R/3.4.3x5  
  R/3.4.4-X11-20180131  
  R/3.5.0-X11-20180131  
  R/3.5.1-X11-20180131  
  R/3.5.1-X11-20180604-UR  
  R/3.5.1-X11-20180604  
  R/3.6.0-X11-20180604  
  R/3.6.2-X11-20180604  
  R/3.6.2  
  R/3.6.3  
  R/4.0.0-X11-20180604  
  R/4.0.0  
  R/4.0.2.bak  
  R/4.0.2.test  
  R/4.0.2-X11-20180604  
  R/4.0.2  
  R/4.0.3  
  R/4.1.0  
  R/4.1.2  
  R/4.2.2  
  
Other possible modules matches:  
  ADMIXTURE  AMDuProf  APR  APR-util  Abaqus_parallel  AdapterRemoval  Advisor  Amber  AmrPlusPlus  Armadillo  ...  
  
To find other possible module matches execute:  
  
  $ module -r spider '.*R.*'  
  
-----  
For detailed information about a specific "R" package (including how to load the modules) use the module's full name.
```



# Linux

## Linux on the HPCC

**Exercise:** Find all HPCC modules with the **module avail** command



**Type in Your Terminal:**

**[user@computer] \$ module avail**

```
parvizim@dev-intel18:~$ module avail
/opt/modules/MP1/GCC/6.4.0-2.28/OpenMPI/2.1.2
ABySS/2.0.2
ABySS/2.1.1
ABySS/2.1.5
ANTS/2.3.2
ATK/2.24.1
Armadillo/8.400.0
BAMap/37.93
BCTools/1.9
BLAST/2.7.1
BISCU/2.1.0-Python-3.6.4
BMA/0.7.17
BamTools/2.5.1
BupPerl/1.7.2-Perl-5.26.1
Boost.Python/1.66.0-Python-3.6.4
Boost/1.66.0-Python-3.6.4
Boost/1.66.0
Boost/1.67.0-Python-2.7.14
Boost/1.67.0
Bracken/2.2
CGAL/4.11.1-Python-2.7.14
CLASS/2.1.7
CONVERGE/2.4.21
CONVERGE/2.4.27
CPACK/7.1
DHTOP/2.3.2
ELPA/2018.05.001
ExaML/3.0.21
FFTW/3.3.7
FPrime/3.4.2
FLTK/1.3.4
FoX/4.1.2
FriBidi/1.0.1
GATK/4.0.5.1-Python-3.6.4
GATK/4.1.3.0-Python-3.6.4
GATK/4.1.4.1-Python-3.6.4
GOAL/2.2.3-Python-3.6.4
GDOS/3.6.2-Python-3.6.4
GL2PS/1.4.0
GNAP-GSNAP/2018-05-11
GObject-Introspection/1.54.1-Python-2.7.14
GROMACS/2018
GROMACS/201803
GStreamer/1.8.3
GTK+/2.24.32
Glib-Pixbuf/2.36.11
GraphicsMagick/1.3.28
Pango/1.41.1
ParMETIS/4.0.3
Perl/5.26.1
PyYAML/3.12-Python-2.7.14
PyYAML/3.12-Python-3.6.4
Python/2.7.14
Python/3.6.4
Qt5/5.10.1
QuantumESPRESSO/5.4.0-hybrid
QuantumESPRESSO/6.2
R/3.5.0-x86_64-Python-3.6.4
R/3.5.1-x86_64-Python-3.6.4
SAMtools/1.9
SAMtools/1.9
SCOTCH/6.0.6
Scenes/3.0.1-Python-3.6.4
SDL2/2.0.9
SLIM/2019dev
SLIM/2021dev
SOPDevOps/2/P241
SPAdes/3.11.1
SPAdes/3.13.0
STAR/2.6.0c
SWIG/2.0.12-Python-3.6.4
SageMath/8.8
ScaLAPACK/2.0.2-OpenBLAS-0.2.20
Stacks/2.0beta10a
Stacks/2.4
Subread/1.6.2
SuiteSparse/5.1.2-METIS-5.1.0
TK/8.6.8
Tkinter/2.7.14-Python-2.7.14
Tkinter/3.6.4-Python-3.6.4
Trilinos/12.12.1-Python-3.6.4
VCFtools/0.1.15-Perl-5.26.0
VCFtools/0.1.15-Perl-5.26.1
VTK/7.1.1-Python-3.6.4
VTK/8.1.0-Python-3.6.4
Valgrind/3.13.0
Velvet/1.2.10-mt-1mer_191
Vim/8.2.0236-Python-3.6.4
ZeroMQ/4.2.5
zpack-ng/3.5.0
zpack-ng/3.6.1
zwtcl/1.16.109-Python-3.6.4
```



# Linux

Linux on the HPCC

**Exercise:** Load HPCC modules with the **module load** command



```
[parvizim@dev-intel18 ~]$ module load R/4.0.2
lmod has detected the following error: These module(s) or extension(s) exist but cannot be loaded as requested:
"R/4.0.2"
Try: "module spider R/4.0.2" to see how to load the module(s).

[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ module load R/4.0.2**



# Linux

## Linux on the HPCC

### Exercise: Load HPCC modules with the **module load** command



```
[parvizim@dev-intel18 ~]$ module load GCC/8.3.0 OpenMPI/3.1.4 R/4.0.2
Lmod is automatically replacing "GNU/6.4.0-2.28" with "GCC/8.3.0".

Lmod is automatically replacing "GNU/6.4.0-2.28" with "GCC/8.3.0".

-----
The following dependent module(s) are not currently loaded: OpenBLAS/0.2.20 (required by: ScaLAPACK/2.0.2
-OpenBLAS-0.2.20, Boost/1.67.0, Python/3.6.4, ScaLAPACK/2.0.2-OpenBLAS-0.2.20, Boost/1.67.0, Python/3.6.4
)
-----

Inactive Modules:
 1) CMake/3.11.1      2) imkl/2018.1.163    3) tbb/2018_U3

Due to MODULEPATH changes, the following have been reloaded:
 1) Boost/1.67.0      2) GMP/6.1.2          3) libffi/3.2.1       4) zlib/1.2.11

The following have been reloaded with a version change:
 1) FFTW/3.3.7 => FFTW/3.3.8          8) ScaLAPACK/2.0.2-OpenBLAS-0.2.20 => ScaLAPACK/2.0.2
 2) GCCcore/6.4.0 => GCCcore/8.3.0     9) Tcl/8.6.8 => Tcl/8.6.9
 3) Java/1.8.0_152 => Java/11.0.2       10) binutils/2.28 => binutils/2.32
 4) OpenBLAS/0.2.20 => OpenBLAS/0.3.7  11) bzip2/1.0.6 => bzip2/1.0.8
 5) OpenMPI/2.1.2 => OpenMPI/3.1.4     12) libreadline/7.0 => libreadline/8.0
 6) Python/3.6.4 => Python/3.7.4       13) ncurses/6.0 => ncurses/6.1
 7) SQLite/3.21.0 => SQLite/3.29.0

[parvizim@dev-intel18 ~]$
```

Type in Your Terminal:

**[user@computer] \$ module load GCC/8.3.0 OpenMPI/3.1.4 R/4.0.2**



# Linux

## Linux on the HPCC

**Exercise:** Unload HPCC modules with the **module unload** command



```
parvizim — parvizim@dev-intel18:~ — ssh -XY parvizim@hpcc.msu.edu — 102x27
Currently Loaded Modules:
 1) GCCcore/6.4.0      9) ScaLAPACK/2.0.2-OpenBLAS-0.2.20  17) SQLite/3.21.0
 2) binutils/2.28     10) bzip2/1.0.6                    18) GMP/6.1.2
 3) GNU/6.4.0-2.28    11) zlib/1.2.11                   19) libffi/3.2.1
 4) OpenMPI/2.1.2     12) Boost/1.67.0                  20) Python/3.6.4
 5) tbb/2018_U3       13) CMake/3.11.1                  21) Java/1.8.0_152
 6) imkl/2018.1.163   14) ncurses/6.0                   22) MATLAB/2018a
 7) OpenBLAS/0.2.20   15) libreadline/7.0               23) powertools/1.2
 8) FFTW/3.3.7        16) Tcl/8.6.8

[parvizim@dev-intel18 ~]$ module unload powertools/1.2; module list

Currently Loaded Modules:
 1) GCCcore/6.4.0      9) ScaLAPACK/2.0.2-OpenBLAS-0.2.20  17) SQLite/3.21.0
 2) binutils/2.28     10) bzip2/1.0.6                    18) GMP/6.1.2
 3) GNU/6.4.0-2.28    11) zlib/1.2.11                   19) libffi/3.2.1
 4) OpenMPI/2.1.2     12) Boost/1.67.0                  20) Python/3.6.4
 5) tbb/2018_U3       13) CMake/3.11.1                  21) Java/1.8.0_152
 6) imkl/2018.1.163   14) ncurses/6.0                   22) MATLAB/2018a
 7) OpenBLAS/0.2.20   15) libreadline/7.0
 8) FFTW/3.3.7        16) Tcl/8.6.8

[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ module unload powertools/1.2; module list**



# Linux

Linux on the HPCC

**Exercise:** Unload all HPCC modules with **module purge** command



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ module list  
  
Currently Loaded Modules:  
1) GCCcore/6.4.0      7) OpenBLAS/0.2.20      13) CMake/3.11.1      19) libffi/3.2.1  
2) binutils/2.28      8) FFTW/3.3.7           14) ncurses/6.0       20) Python/3.6.4  
3) GNU/6.4.0-2.28     9) ScaLAPACK/2.0.2-OpenBLAS-0.2.20  15) libreadline/7.0   21) Java/1.8.0_152  
4) OpenMPI/2.1.2      10) bzip2/1.0.6         16) Tcl/8.6.8         22) MATLAB/2018a  
5) tbb/2018_U3        11) zlib/1.2.11         17) SQLite/3.21.0  
6) imkl/2018.1.163    12) Boost/1.67.0        18) GMP/6.1.2  
  
[parvizim@dev-intel18 ~]$ module purge; module list  
No modules loaded  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ module purge; module list**



# Linux

## Linux on the HPCC

**Exercise:** Reload default HPCC modules with the **logout** command



Type in Your Terminal:

[user@computer] \$ logout

```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ module purge; module list  
No modules loaded  
[parvizim@dev-intel18 ~]$ logout  
Connection to dev-intel18 closed.  
  
Currently Loaded Modules:  
  1) gateway/1.0  
  
[parvizim@gateway-03 ~]$ ssh dev-intel18  
Last login: Mon Jan 23 21:39:23 2023 from gateway-03.dmz  
  
===  
Please note that processes on development nodes are limited to two hours of  
CPU time; for longer-running jobs, please submit to the queue.  
  
Development nodes are a shared system; for information about performance  
considerations please see: https://docs.icer.msu.edu/development\_nodes/  
===  
  
[parvizim@dev-intel18 ~]$ module list  
  
Currently Loaded Modules:  
  1) GCCcore/6.4.0      7) OpenBLAS/0.2.20      13) CMake/3.11.1      19) libffi/3.2.1  
  2) binutils/2.28      8) FFTW/3.3.7          14) ncurses/6.0       20) Python/3.6.4  
  3) GNU/6.4.0-2.28     9) ScaLAPACK/2.0.2-OpenBLAS-0.2.20 15) libreadline/7.0   21) Java/1.8.0_152  
  4) OpenMPI/2.1.2      10) bzip2/1.0.6        16) Tcl/8.6.8         22) MATLAB/2018a  
  5) tbb/2018_U3        11) zlib/1.2.11        17) SQLite/3.21.0     23) powertools/1.2  
  6) imkl/2018.1.163    12) Boost/1.67.0       18) GMP/6.1.2
```



# Linux

Linux on the HPCC

**Exercise:** Write a 'Hello World' script with c



**Type in Your Terminal:**

**[user@computer] \$ vi hello.c**

```
#include <stdio.h>

int main()
{
    printf("Hello, World!\n");
    return 0;
}
```

1. #include <stdio.h>
2. int main()
3. {
4. printf("Hello, World!\n");
5. return 0;
6. }





---

# Linux

Linux on the HPCC

**Exercise:** Load the GNU/8.2.0-2.31.1 compiler module



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ module purge; module load GNU/8.2.0-2.31.1  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

**[user@computer] \$ module purge; module load GNU/8.2.0-2.31.1**



# Linux

Linux on the HPCC

**Exercise:** Compile and run the 'Hello World' script with GNU



```
[parvizim@dev-intel18 ~]$  
[parvizim@dev-intel18 ~]$ gcc hello.c -o hello; ./hello  
Hello, World!  
[parvizim@dev-intel18 ~]$
```

**Type in Your Terminal:**

```
[user@computer] $ gcc hello.c -o hello; ./hello
```



# Linux

Linux on the HPCC

The simple Linux utility for resource management (**SLURM**) is the HPCC workload manager i.e., **job scheduler**

- Framework for executing and monitoring jobs
- Allocates nodes/CPU's to users for a specified duration
- Manages the queue of pending jobs; arbitrates contentions



# Linux

## Linux on the HPCC

**Exercise:** Write a bash script 'my\_job.sb' to schedule a SLURM job that runs your 'hello.c' script

```
#!/bin/bash                                     #Tell the shell to interpret bash

##### SLURM Resource Requests #####

#SBATCH --time=0-00:10                         #How long the job will run (days-hours:minutes)
#SBATCH --nodes=1                               #How many compute nodes the job needs
#SBATCH --ntasks=1                             #How many concurrent tasks the job needs
#SBATCH --cpus-per-task=1                      #How many CPUs each task needs
#SBATCH --mem-per-cpu=1G                       #How much memory each CPU needs

##### SLURM Administrative Settings #####

#SBATCH --job-name HelloWorld                  #Name the job for convenience
#SBATCH --output=%x-%j.SLURMout                #Name the output file (JobName-JobNumber.SLURMout)
#SBATCH --mail-type=ALL                       #Tell SLURM to email you when job starts, stops, error
#SBATCH --mail-user=                           #Provide SLURM your email address

##### bash Commands to Run #####

module purge                                  #unload all modules
module load GNU/8.2.0-2.31.1                 #load the GNU compiler
cd /mnt/home/                                #Navigate to the directory containing hello.c
gcc hello.c -o hello                          #Run the command to compile hello.c
./hello                                       #Run the compiled executable hello
```



# Linux

Linux on the HPCC

**Exercise:** Submit 'my\_job.sb' to SLURM with the **sbatch** command



```
[parvizim@dev-intel18 ~]$ sbatch my_job.sb  
Submitted batch job 3479290  
[parvizim@dev-intel18 ~]$
```

Type in Your Terminal:

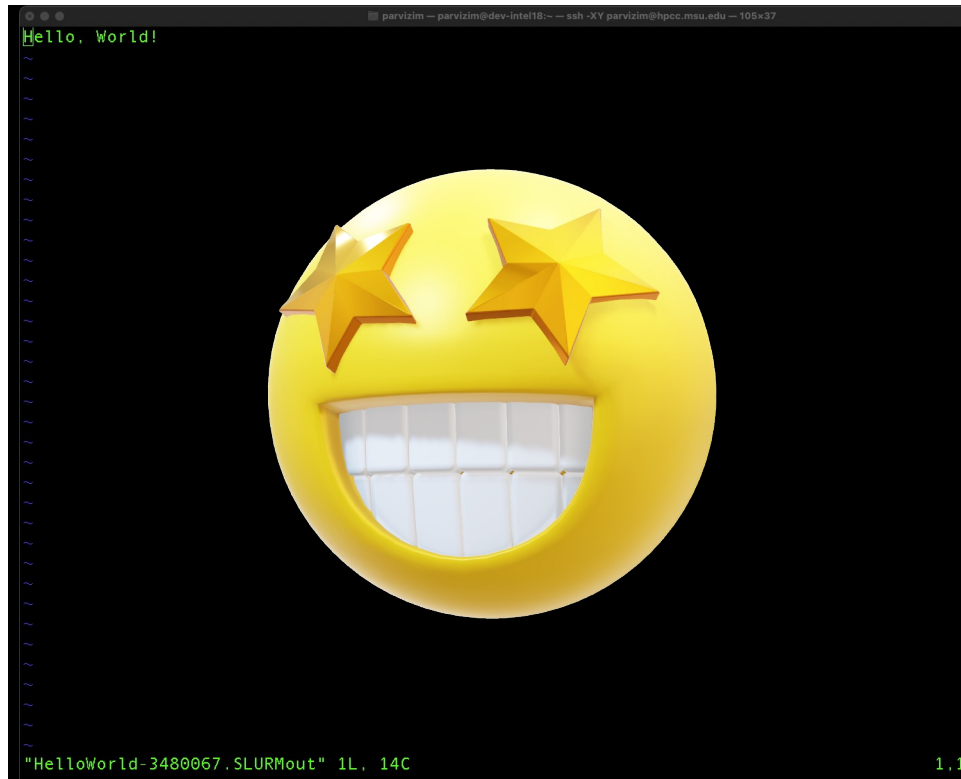
```
[user@computer] $ sbatch my_job.sb
```



# Linux

Linux on the HPCC

**Exercise:** Check the output file 'HelloWorld-xxxxxxx.SLURMout'



A terminal window with a black background. The title bar shows 'parvizim -- parvizim@dev-intel18:~ -- ssh -XY parvizim@hpcc.msu.edu -- 105x37'. The first line of output is 'Hello, World!' in green. Below it is a large yellow smiley face emoji with two orange stars for eyes. At the bottom left, the status bar shows '"HelloWorld-3480067.SLURMout" 1L, 14C' in green. At the bottom right, it shows '1.1' in green.

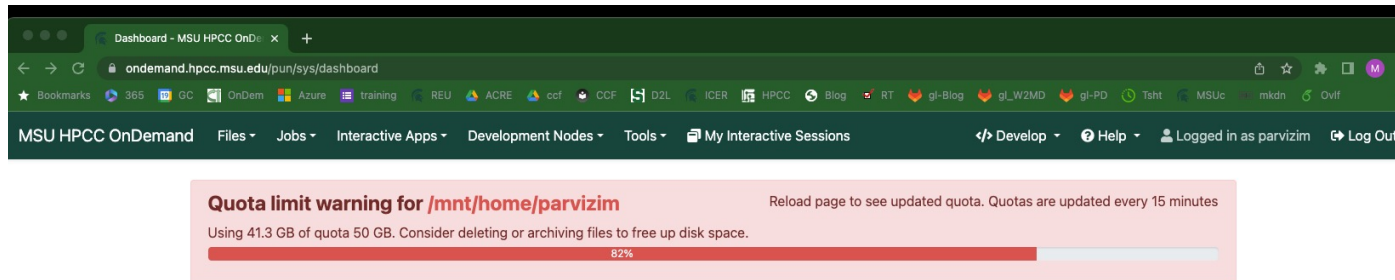


# Linux

Linux on the HPCC

HPCC OnDemand

[ondemand.hpcc.msu.edu](https://ondemand.hpcc.msu.edu)



**MICHIGAN STATE**  
UNIVERSITY

Institute for Cyber-Enabled Research

OnDemand is an integrated access point for the MSU High Performance Computing Center's resources.

Please [Contact Us](#) if you have any questions, feedback, or suggestions.

## Message of the Day

## ICER's OnDemand Resources

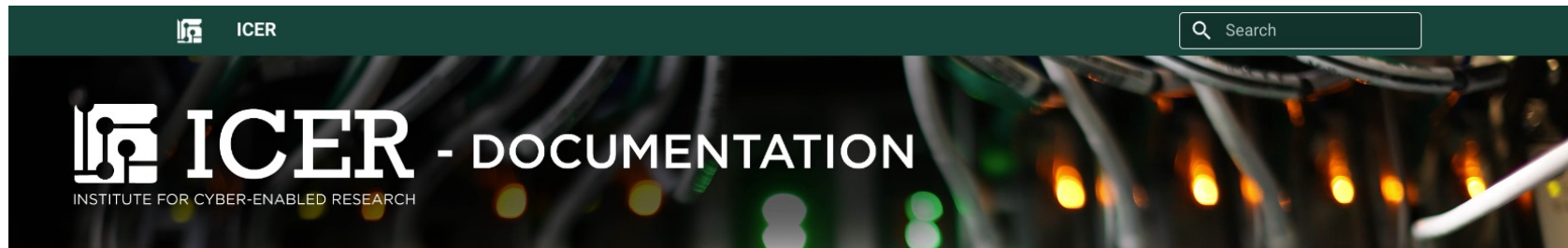
In 30 minutes or less, this non-credit, self-paced training course introduces OnDemand Resources available to utilize the High Performance Computing Center (HPCC) provided by the Institute for Cyber-Enabled Research (ICER) at Michigan State University. No prior knowledge is required for this course.



# Linux

Linux on the HPCC

HPCC Documentation, AKA “The Wiki”, [docs.icer.msu.edu](https://docs.icer.msu.edu)



## MSU HPCC User Documentation

[Home](#)

- Access the HPCC >
- HPCC resources >
- Computing environment >
- File systems >
- File transfer >
- Software module system >
- Powertools >
- Job submission by SLURM >
- Software tutorials >
- Training material >

[FAQ](#)

[Lab Notebooks](#)

[Tags](#)

## Home

### Getting Access to the HPCC

For potential users with an MSU NetID, accounts must be requested by a MSU tenure-track faculty member. Researchers at partner institutions should use the mechanism specified by their institution's agreement with MSU. For more information, see: [Obtain an HPCC Account](#) and on the [ICER website](#).

### CPU and GPU Time Limits

Non-buyin users are limited to 500,000 CPU hours (30,000,000 minutes) and 10,000 GPU hours (600,000 minutes) every year (from January 1st to December 31st). More information is available at [Job Policies](#).



## Table of contents

[Getting Access to the HPCC](#)

[CPU and GPU Time Limits](#)

[Buy-in Options](#)

[Questions?](#)

[Online Helpdesk Hours](#)

[HPCC Workshops and Training](#)

[Acknowledgements](#)



INSTITUTE FOR CYBER-ENABLED RESEARCH





# Contact ICER

[icer.msu.edu/contact](http://icer.msu.edu/contact)



INSTITUTE FOR CYBER-ENABLED RESEARCH