United Nations Economic Commission for Africa

Climate Research for Development in Africa

ICPAC

CGIAR RESEARCH PROGRAM ON Climate Change, Agriculture and Food Security

CCAFS

# Introduction to Linux and working on ICPAC High Computing Cluster (HPC)

## Institutional Linkages, South-South Partnerships and Capacity Building Hands-on Workshops on Objective Climate Forecasts for Agriculture and Food Security Sector in Eastern and Southern Africa

**30th August to 04th September 2021**

**Victoria Falls, Zimbabwe**

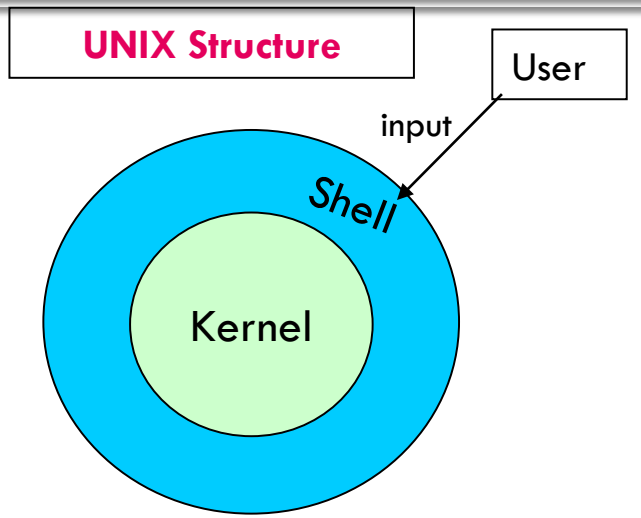
PEACE, PROSPERITY AND REGIONAL INTEGRATION

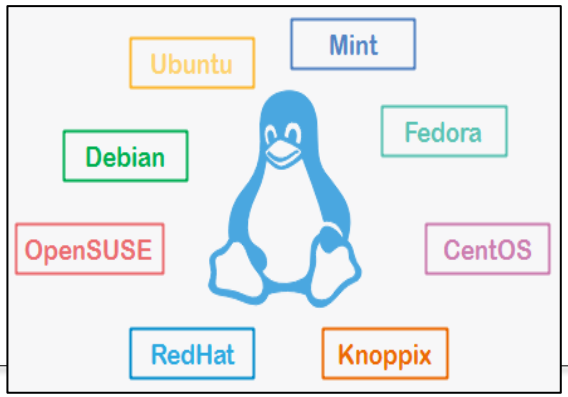
# Outline

- What's Linux?
- Why Linux?
- Directory Structure of Linux System
- Useful Linux Commands
- Specifying Multiple Files
- Absolute and Relative Path
- Redirect, Append and Pipe
- File Permission and Ownership
- Remote Login and File Transfer
- Text Editor
- Running jobs in the background using screen

# What's Linux?

- It is a Unix-like operating system (OS).
- A unix system is described as kernel and shell.
- Kernel is a main program/core component of Unix system. It controls hard wares, CPU, memory, hard disk, network card etc.
- A shell provides an interface between the user and the operating system kernel
- Shell interprets your input as commands and pass them to kernel.
- Linux OS has various distributions.



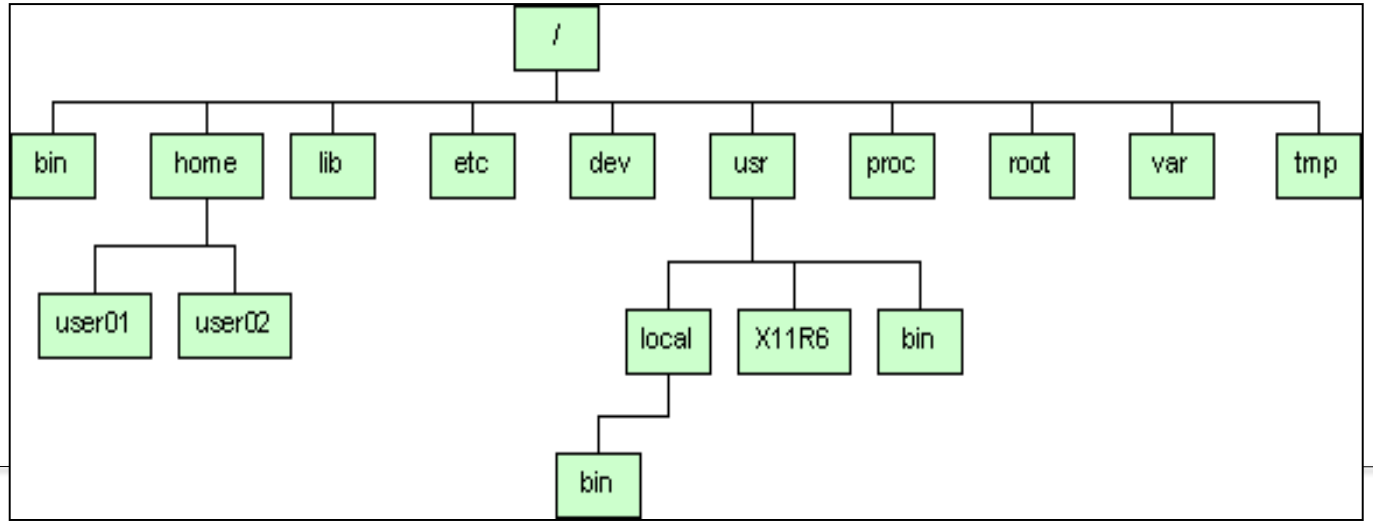



# Why Linux?

- **Multi-user, multi-tasking** - many users can logged into a system simultaneously, and run many programs.
- **Excellent networking facilities** - comes with networking facilities, allowing you to share hardware
- **Task scheduler** - allocate the execution of the CPU to a number of different tasks.
- **Security manager** - the OS maintains the security of the information in the computer's files and controls who can access the computer.
- **Linux operating system is costs less, reliable, stable, and very powerful**
  - **Cost less** – full source code freely available (including several free softwares)
  - **Stable** - the crash of an application is much less likely to bring down the OS.
  - **Reliable** - Linux servers are often up for hundreds of days compared with the regular reboots required with a Windows system.
  - **Extremely powerful**
- **Linux is very easily upgradeable**

# Directory Structure

- Files are put in a directory.
- All directories are in a hierarchical structure (tree structure).
- User can add and remove any directories on the tree.
- Top directory is "/", which is called slash or root.



# Sub-directories

- **/bin** System binaries, including the command shell
- **/dev** Device files for all your peripherals
- **/etc** System configuration files
- **/home** User directories
- **/lib** Shared libraries and modules
- **/lost+found** Lost-cluster files, recovered from a disk-check
- **/mnt** Mounted file-systems
- **/opt** Optional software
- **/proc** Kernel-processes pseudo file-system
- **/root** Administrator's home directory
- **/sbin** System administration binaries
- **/usr** User-oriented software
- **/var** Various other files: mail, spooling and logging

# Linux Command Syntax

- Commands are typed at a shell prompt (usually ends in a dollar sign **$**)
- Most commands consists of three parts, i.e. **command name, options, arguments**.

**Syntax: $ command-name optionA optionB arg1 arg2**

- Space is necessary between command name, options and arguments.
- Options always start with "**–**"

**whoami**

**ls –l .bashrc**

- Unix is case-sensitive. '**ls**' is not the same as '**LS**'.
- Linux remembers your commands. Use the up and down cursor keys to scroll through the list of previous commands
- To exit from the shell, use the exit command.

# Some Useful Commands

- **Working with Directors and Files**

**mkdir** create a directory
**cd** changes the working directory
**pwd** print the working directory
**rmdir** remove directory. **Be careful!!**
**locate** find a file with in Linux OS
**clear** clear the screen

- **Working with Directors and Files**

**cp** copy files
**mv** move, or rename files
**rm** remove files
**ln** make links between files
**touch** create, or update files

- **Listing file system contents**

**ls** list files
**ls -l** list file detail
**ls -a** include hidden files)
**ls -al** list details, including hidden files
**man** get help for a command

- **Displaying file contents**

**cat** list file contents
**more** page-by-page listing
**less** like more, but better
**head** list first 10 lines
**tail** list last 10 lines

# Specifying Multiple Files

- Most programs can be given a list of files

**rm file1.txt file2.txt file3.txt** #To delete several files at once

**mkdir Test1 Test2** #To make several directories in one go

**cat notes.txt morenotes.txt** #To list two files, one after another:

## Specifying Files with Wildcards

- Use the * wildcard to specify multiple filenames to a program:

**rm k*** #Remove all files with names starting k)

**ls *** #List all the files except special files (start with '.' or the hidden ones)

**ls –a *** #Display all files including the hidden ones.

# Paths in Linux

- A path describes the location of a file/directory in the directory tree.
- To express a path, you can use relative path or absolute path.
- A relative path is specified in relation to your current directory. Two special directories:

  `.` #the current directory

  `..` #the parent of the current directory

- So if I'm at /home/hussen/Desktop and wish to specify the path above in a relative:

  **cd ../Downloads/data/observed**

- This indicates, first go up one directory level, then come down through the Downloads directory, followed by the data directory and then to observed.
- To know your absolute path, use the **pwd** command.
- An absolute path is specified from the root ( / ) of the file system e.g.,

  **cd /home/hussen/Downloads/data/observed**

  **ls /usr/share/doc/** #Lists the files in the given directory

# Redirect, Append and Pipe

## Redirect and append

- Output of command is displayed on screen.
- Using "**>**", you can redirect the output from screen to a file.
- Using "**>>**" you can append the output to the bottom of the file.

## Pipe

- Some commands require input from a file or other commands.
- Using "**|**", you can use output from other command as input to the command.

- **Redirect ( >):**
  **head -3 sample.txt > redirect.txt**
- **Append ( >> ):**
  **tail -3 sample.txt >> redirect.txt**

- **Pipe:**
  **ls -l | more**
  **tail redirect.txt | grep Desk**

# File Ownership and Permissions

- All of files and directories have owner and permission.
- There are three types of permission, readable, writeable and executable.
- Permissions are given to three kinds of group. owner, group member and others.
- One can easily view the permissions for a file by invoking a long format listing using ls -l

- For instance, the output of the command ls -l test.sh would look like this

**-rwxr-xr-x 1 training modelers 1423 May 7 2020 test.sh**

- The first item specifies the file type (e.g., **d** a directory, **-** a regular file and **l** a symbolic link)
- The remaining nine slots are three sets of permissions (r: readable, w: writable, x: executable) for three different categories of users (**u**: user/owner, **g**: group, **o**: others **a**: all).

# File Ownership and Permissions

- The ownership of the file or directory can be changed using the command **chown**

  **chown <owner> <file/directory name>**

- The group of the file or directory can be changed using the command **chgrp**

  **chgrp <group> <file/directory name>**

- The permissions of the file can be changed using the command **chmod**

  **chmod -R XXX <filename or directory>**

- -R is optional and when used with directories will traverse all the sub-directories of the target directory changing ALL the permissions to XXX.

  chown hussen logfile.txt #makes the logfile.txt be owned by user hussen
  chgrp training /home/training #changes the group /home/training to training
  chmod a+w filename #add writable permission to all users
  chmod o-x filename #remove executable permission from others

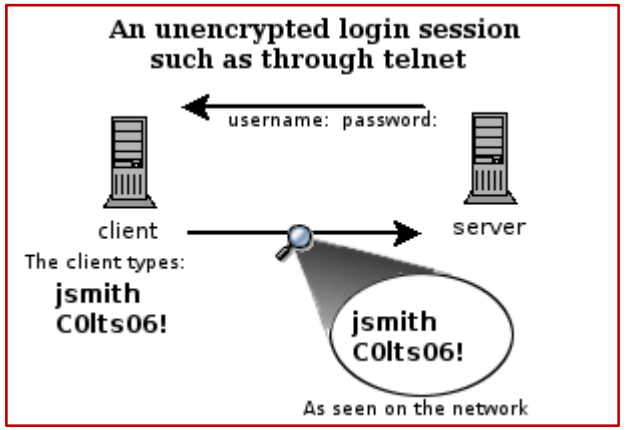
- Further, the numbers can be used to change the permissions of a file or a directory:

**0** = Nothing, **1** = Execute, **2** = Write, **4** = Read

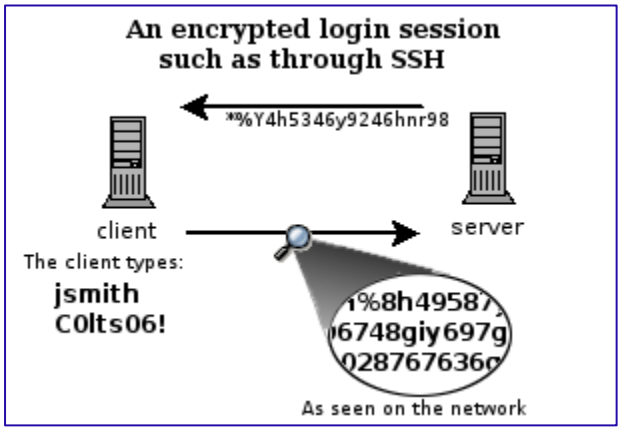
# Remote Login and File Transfer

- Internet allows us to use the resources of a distant computer somewhere else in the world.
- rshd, telnetd, ftpd, sshd are server programs that provide remote login and file transfer services.
- For example, telnet server provides remote login service. ftp server provides file transfer service.

**An unencrypted telnet session**




**An encrypted ssh session**




PEACE, PROSPERITY AND REGIONAL INTEGRATION

# How to Access a Cluster using SSH

- **ssh (secure shell)** is the command use to connect remote machine – the client
- Remote machine can be accessed using IP address of the machine or name of the machine.

**ssh login@your_IP # with –X or –Y option**
**ssh login@server_name # with –X or –Y option**

The **-X** and **-Y** options activate **X11** libraries; these libraries allow for graphical information to be sent via the terminal.

- To copy files between your machine and a remote host, you can use using **scp** (SeCure coPy):

**scp filename :/RemotePath/**
**scp login@your_IP:/RemotePath/ /filename /local/path**

- To connect to the ICPAC cluster, use the following information:

**ssh -X training@197.254.1.14**

# Text Editors

- Several Linux/Unix text editors are available

**vi (or vim)**

**emacs**

**nano**

**pico**

**gedit**

- **vi** is available on all Unix systems and is one of the most popular editor.
- The vi editor has two modes of operation:
- **Command mode**: a mode that enables to perform administrative actions on the file such as saving the files, cutting and pasting words, as well as finding and replacing.
- **Insert mode:** a mode for inserting text in the file.

# Vi Editor

**Command mode commands:**

**:w** = To save a file

**:wq** = To save and quit

**:q** = To close the file without saving

**:q!** = To force a quit without saving (if the file has been modified/changed).

**Changing to Insert Mode:**

**i** = converts to insert mode (to insert text)

**esc** = to exit insert mode (or re-enter command mode)

**x** = deletes character under the cursor (see also dw, de, d$, dd, 2dd, x, X)

**dw** = delete word

**dd** = delete line

**<n> dd** = delete n lines

**/phrase** = Searches for the next instance of the term specified

**:%s/*pattern1*/*pattern2*/g**= Replaces/changes all (global change)

**:set number** = displays line numbers

# Running jobs in the background using screen

- Screen allows you to create a virtual *screen* that helps you to disconnect and reconnect if needed
- Allows to run other commands at the same time that the background process is running
- Keep processes running despite a dropped connection

**$ screen** = to create a screen session or window

***$ ctrl+A and C*** = creates a new *screen* session

**$ screen -ls** = to see the list running sessions/screens)

**$ ctrl+A and D** = to detach the screen (without killing the processes in it )

**$ screen -r [ID]** =to resume your screen session

**$ screen -X -S [ID] kill** =To kill a detached screen

# Practical Examples and Exercises

PEACE, PROSPERITY AND REGIONAL INTEGRATION

# Logging into the ICPAC Cluster

**1. Login to the ICPAC computing cluster using the access credentials**

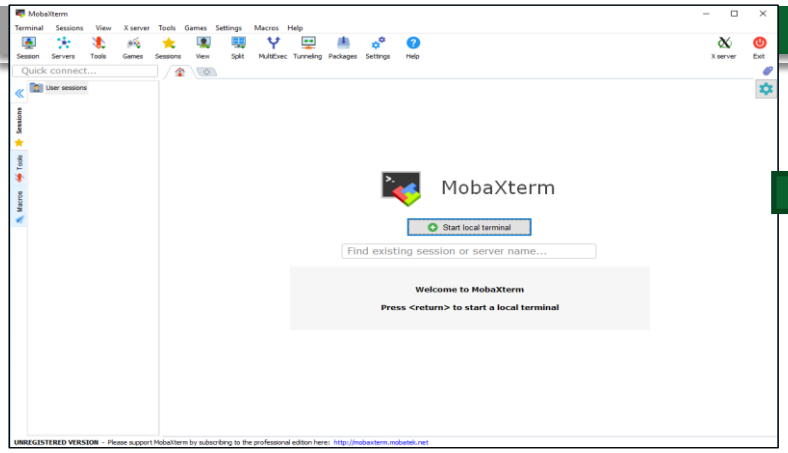
To connect to a remote machine using SSH, you need to have a SSH *client* program installed on your machine. If you are using windows, you need to install MobaXTerm.

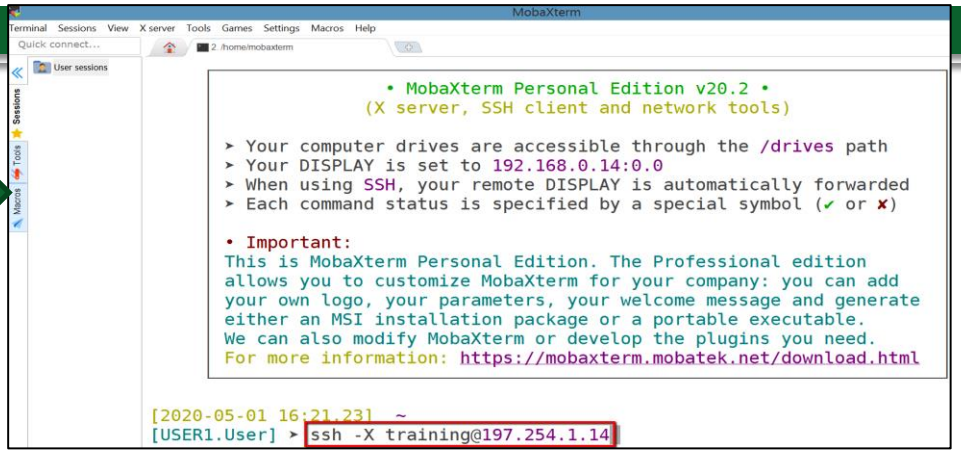
**1.1 Installing MobaXterm on Windows**

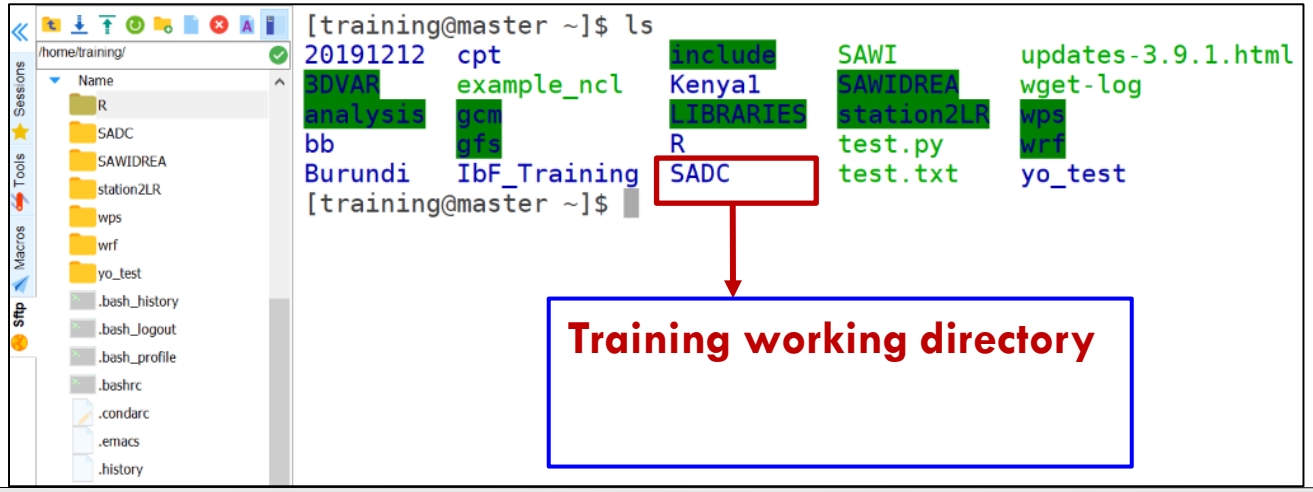
- Download the Home Edition of MobaXterm. Select the Installer edition from the download page.
- Click on the downloaded zip file and uncompress the contents of the downloaded zip file
- Double click the MobaXterm_installer executable file to begin the installation.
- Once the install has finished, open the MobaXterm app..
- From here, you can start a local terminal by clicking on the “Start local terminal” button in the MobaXterm main window

## Starting Local Terminal

## Loging in to the ICPAC computing cluster using SSH




**Password:**

**Tr8iN#432**

PEACE, PROSPERITY AND REGIONAL INTEGRATION

## Some basic linux commands

**2.** Enter each of the following commands in the terminal and try to interpret the outputs.

**date**

**whoami**

**pwd**

**cd**

**ls**

**ls -l**

**ls -al**

**history**

**clear**

## Making Directories and Sub-directories

**3.** Create a working directory with your country name using mkdir command. Note that this will be your working directory.
Example: mkdir zimbabwe

**4.** Go to the directory you have created using cd command.
Example: cd zimbabwe

**5.** Copy the file files under SADC to your working directory
Example: cp -r /home/training/SADC/* .

## Using vi editor

**6** **a.** create a sub-directory called **linux** in your working directory

**b.** Copy the file test.ncl from /home/training/ into **linux** directory

**c.** Edit the file using vi

**d.** Display line number using :set nu

**e.** Replace all occurrences of “olddata” with “newdata” (**:%s/*olddata*/*newdata*/g)**

**d.** Save the file and quit (press esc then :wq to save and close)

## Changing file permissions

**7.** Go to your directory and create a shell script called *hello.sh* using a vi editor. Type the following text then save and close it

```
#!/bin/bash
echo "Hello world“
```

Try to run the bash script as follow:

```
./hello.sh
```

What have you found? Ok, look at the permissions on the file using **ls -l**. Make it executable and run it in the background as follows:

```
chmod a+rx hello.sh
./hello.sh &
```

## Transferring files between your local computer and a remote cluster

**8 a.** Transfer/copy the **hello.sh** file you created above into the current directory (.) on the local machine using **scp** command

scp training@197.254.1.14:~/country/linux/hello.sh .

**b.** Pick a small size file (word or text) from your computer and copy/send it to your working directory in the remote cluster using **scp** command

scp filename training@197.254.1.14:~/country/linux/

**c.** Delete the file you have copied in your working directory in the remote cluster using rm command