Linux Training
for New Users of Cluster

Georgia Advanced Computing Resource Center

University of Georgia

Suchitra Pakala

pakala@uga.edu

Overview

- GACRC
- Linux Operating System
- Shell, Filesystem, and Common Commands
- Scripting and execution

Georgia Advanced Computing Resource Center

Who Are We:

- **G**eorgia **A**dvanced **C**omputing **R**esource **C**enter (**GACRC**)
- Collaboration between the Office of Vice President for Research (**OVPR**) and the Office of the Vice President for Information Technology (**OVPIT**)
- Guided by a faculty advisory committee (GACRC-AC)

Why Are We Here?

- To provide computing hardware and network infrastructure in support of high-performance computing (**HPC**) at UGA

Where Are We?

- http://gacrc.uga.edu (Web)
- http://wiki.gacrc.uga.edu (Wiki)
- http://gacrc.uga.edu/help/ (Web Help)
- https://wiki.gacrc.uga.edu/wiki/Getting_Help (Wiki Help)

- Introduction to Linux
- Connecting to a Linux machine

Linux Operating System

- Operating System (**OS**)
  - Software program
  - Enables hardware to communicate and operate with software
  - Manages all resources and applications
    - Memory, File system, Networking, I/O, etc.
    - Browser, Video player, etc.

- Most popular Operating Systems : Mac, Linux, Windows.

Linux Operating System

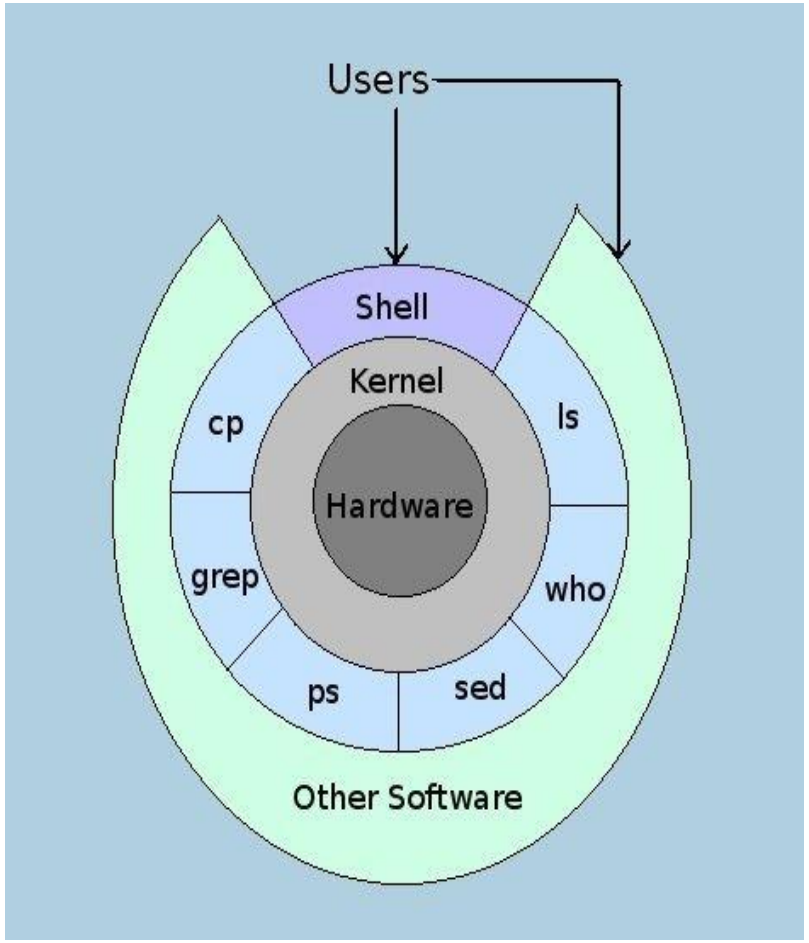
- About Linux OS
  - Developed in 1991 by Linus Torvalds
  - Open Source
  - Multi-user, Multi-tasking operating system
  - Most popular OS in the high performance computing community
  - Several distributions - Ubuntu, CentOS, Fedora, RedHat, etc.

- Why use Linux?
  - Free, Stable, Secure, Portable, Scalable

Linux Operating System

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Two major components of Linux:
  - **Kernel**
    - Core of the OS
    - Schedules processes, and interfaces with hardware
    - Manages resources – memory, I/O, etc
  - **Shell**
    - The shell is an interface between users and the kernel
    - Command-line – Users can type commands
    - Command interpreter – runs commands
    - Programming environment – for scripting



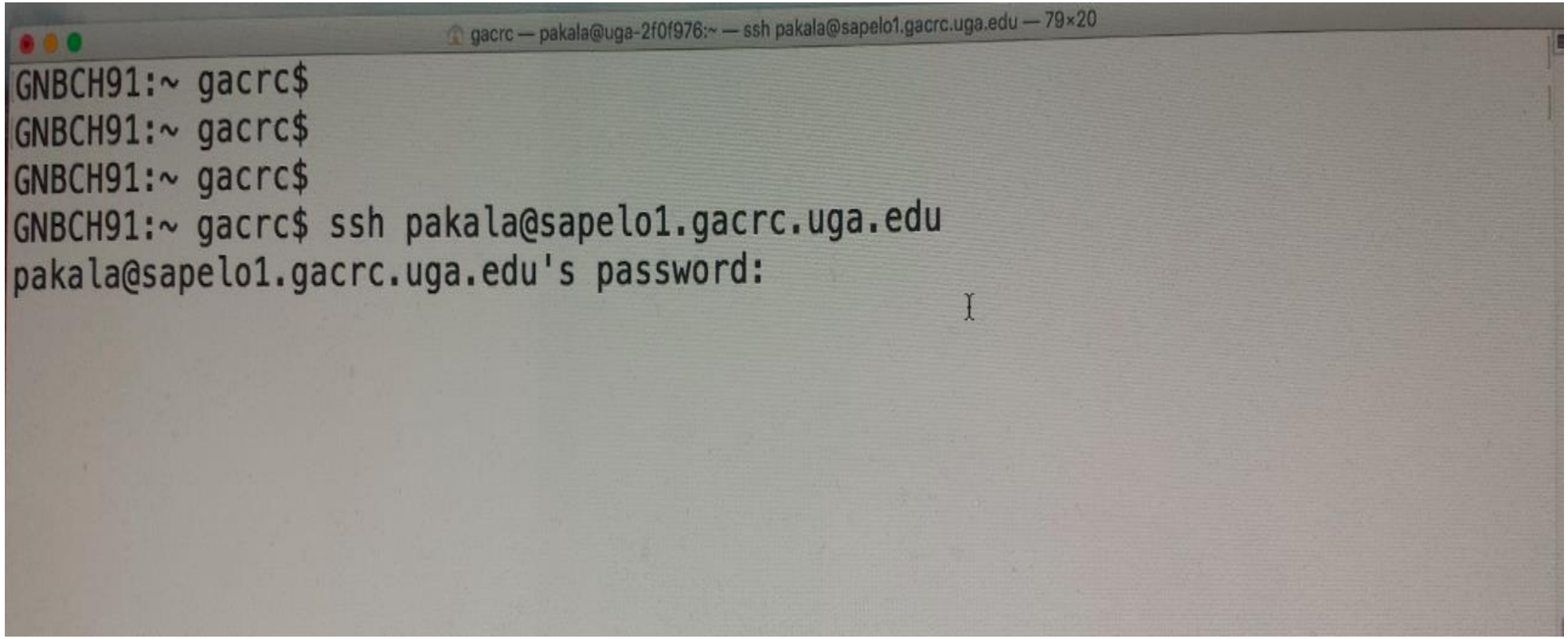
Linux Shell

- "Shell" - **command line interpreter**
  - Interacts between the system and users
    - Reads commands from the keyboard
    - Executes commands
    - Displays the output
  - Provides the "environment"
    - Command-line completion
    - Auto-correction
    - TAB key - Auto-completion
    - Up ⬆ and down ⬇ arrow keys - command history
  - Several shells available
    - **Bash-shell** (bash) is the default one.

Connecting to Shell - on Mac/Linux

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Open a terminal and type: **ssh <UGAMyID>@sapelo2.gacrc.uga.edu**
- Enter your Password when prompted
  - Note: The password entry will not show on the screen. Not even asterisks.



Connecting to Shell – on Windows

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Download SSH Secure Shell from http://eits.uga.edu/hardware_and_software/software/

1. Open the SSH Secure Shell and click on "**Quick connect**".

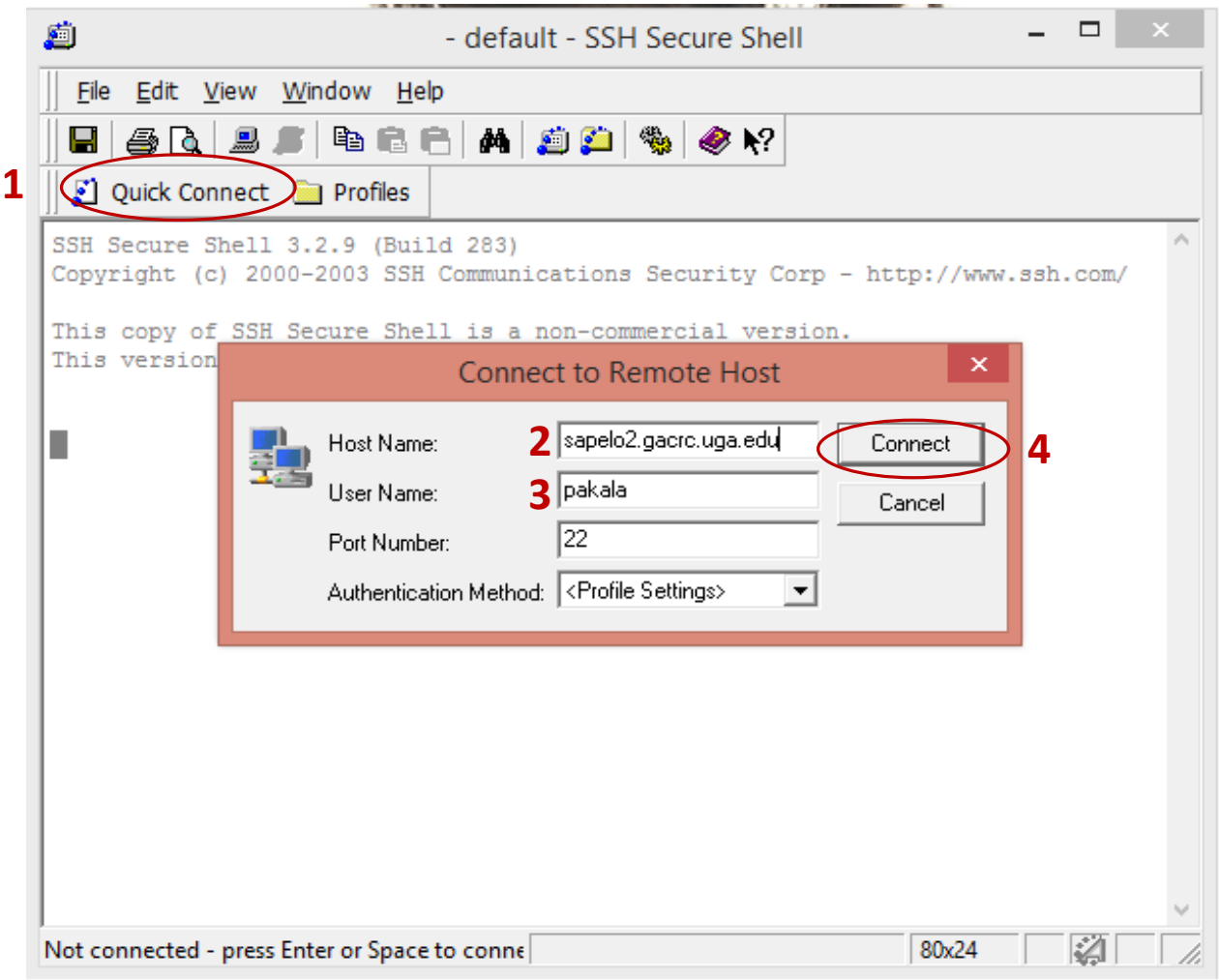
2. Hostname: **sapelo2.gacrc.uga.edu**

3. User Name: **your UGA MyID**

- Port Number: 22
- Authentication Method: Select <Profile Settings>

4. Enter above information and click "**Connect**"

- Enter your password in the next pop up window and click "OK"



Connecting to Shell – on Windows

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

sapelo2.gacrc.uga.edu - default - SSH Secure Shell

File Edit View Window Help

Quick Connect Profiles

```
SSH Secure Shell 3.2.9 (Build 283)
Copyright (c) 2000-2003 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is a non-commercial version.
This version does not include PKI and PKCS #11 functionality.


Duo two-factor login for pakala

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-3898
 2. Phone call to XXX-XXX-3898
 3. SMS passcodes to XXX-XXX-3898

Passcode or option (1-3):
```

Select Duo Login

Connected to sapelo2.gacrc.uga.edu | SSH2 - aes128-cbc - hmac-sha1 - n | 80x24

sapelo2.gacrc.uga.edu - default - SSH Secure Shell

File Edit View Window Help

Quick Connect Profiles

```
SSH Secure Shell 3.2.9 (Build 283)
Copyright (c) 2000-2003 SSH Communications Security Corp - http://www.ssh.com/

This copy of SSH Secure Shell is a non-commercial version.
This version does not include PKI and PKCS #11 functionality.


Duo two-factor login for pakala

Enter a passcode or select one of the following options:

 1. Duo Push to XXX-XXX-3898
 2. Phone call to XXX-XXX-3898
 3. SMS passcodes to XXX-XXX-3898

Passcode or option (1-3): 785502

Success. Logging you in...
pakala@sapelo2-sub2 ~$ pwd
/home/pakala
pakala@sapelo2-sub2 ~$ cd /lustre1/pakala/
pakala@sapelo2-sub2 pakala$
```

home directory

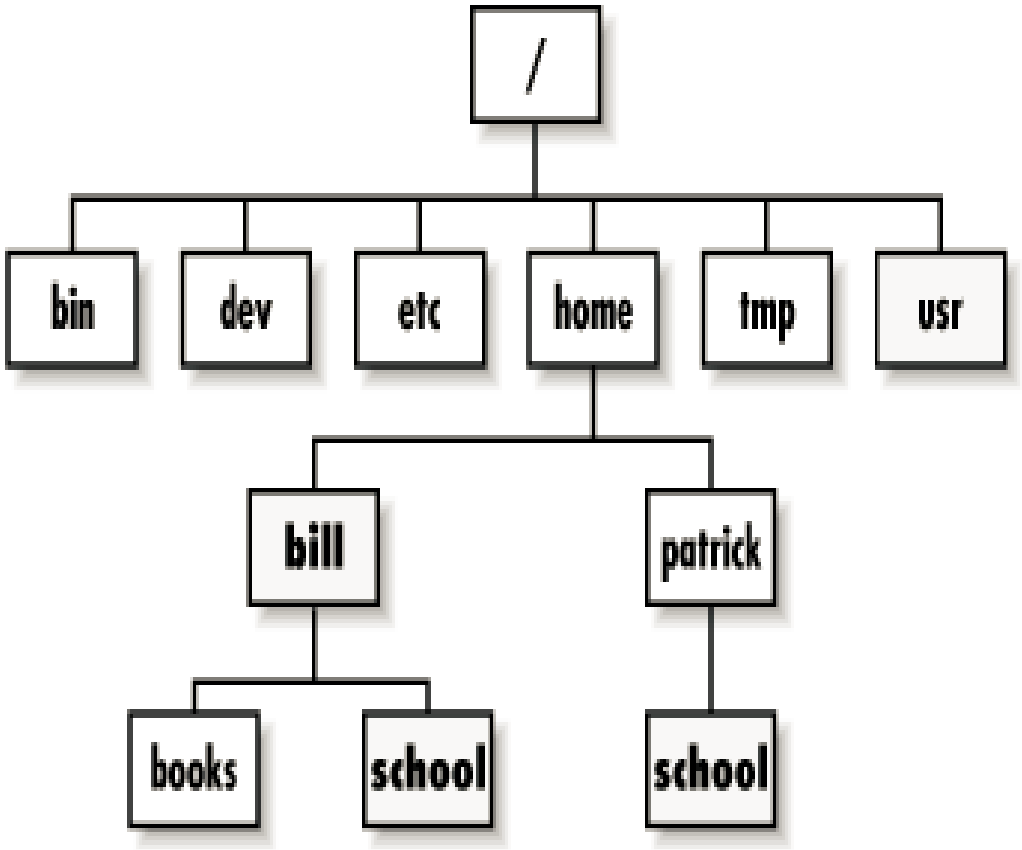
global scratch directory

Connected to sapelo2.gacrc.uga.edu | SSH2 - aes128-cbc - hmac-sha1 - n | 80x24

- Linux Directory Structure
- Navigation Commands

Understanding Linux Directory Structure

- **'upside down tree'**
- **Root directory** ("/" forward slash)
- Organized inside root directory
- Create directories inside - **sub directories**
- Unique name in its containing directory



Relative Path vs Absolute Path

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **Relative path**
  - Path to a file, relative to current location (present working directory)

```
$ pwd
/home/pakala/

$ ls
Suchi_Scripts    Blast

$ ls Blast
AF293          Escherichia_Coli_LF82.fasta
blast.sh       GCF_000002655.1_genomic.fna
```

Present working directory

Relative Path

- **Absolute or Full path**
  - Path to a file, beginning at the root

```
$ ls /home/pakala/Blast/
AF293          Escherichia_Coli_LF82.fasta
blast.sh       GCF_000002655.1_genomic.fna
```

Absolute path

Change Directory (cd)

- **cd** :change your current working directory

```
pakala@uga-2f0f976:~ $ cd /usr/bin      Move to bin subdir of usr dir
pakala@uga-2f0f976:~ $ cd ..            Move up one directory
pakala@uga-2f0f976:~ $ cd               Returns to home directory
pakala@uga-2f0f976:~ $ cd ~pakala       Returns to home directory/pakala
pakala@uga-2f0f976:~ $ cd $HOME         Environment Variable/home dir
```

- **pwd** :present working directory

```
pakala@uga-2f0f976:~ $ pwd
/home/pakala
```

List Directory (ls)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **ls**
  - lists files and directories that exist in the current location
    - Note: we cannot differentiate between files and directories

```
pakala@uga-2f0f976:~ $ ls
e_coli_data.fq  hello.sh  sample_script  sub.sh  Suchi_Scripts
```

- **ls –l**
  - shows file permissions, owner of file, group, file size, modified date and time, and differentiates between file or directory name.

```
pakala@uga-2f0f976:~ $ ls -l
total 584496
-rw-r--r-- 1 pakala abclab 1610499990 Mar  6 09:46 e_coli_data.fq
-rwxr----- 1 pakala abclab        136 Feb 21 15:22 sample_script
-rw-r--r-- 1 pakala abclab        284 Mar  6 09:50 sub.sh
drwxr-xr-x 2 pakala abclab          2 Feb 22 12:07 Suchi_Scripts
```

List Directory (ls)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **ls -a**
  - Lists hidden files. They start with '.'
  - These are files containing profiles and other settings that should not be altered unless necessary, and hence are "hidden"

```
pakala@uga-2f0f976:~ $ ls -a
.                    .bash_history.n609          .emacs        .mozilla
..                   .bash_history.sapelo2-sub1  .emacs.d      .oracle_jre_usage
.bash_history        .bash_history.sapelo2-sub2  .felix        sample_script
.bash_history.n201   .bash_logout                .fontconfig   .ssh
.bash_history.n204   .bash_profile               .gnome2       sub.sh
.bash_history.n206   .bashrc                     hello.sh      Suchi_Scripts
.bash_history.n210   .beast                      .java         .swp
.bash_history.n227   .cache                      .ldaprc       .viminfo
.bash_history.n233   .config                     .lmod.d
.bash_history.n234   e_coli_data.fq              .matlab
```

List Directory (ls)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **ls -lh**
  - shows sizes in human readable format

```
pakala@uga-2f0f976:~ $ ls -lh
total 571M
-rw-r--r-- 1 pakala gclab 1.5G Mar  6 09:46 e_coli_data.fq
-rwxr----- 1 pakala gclab  136 Feb 21 15:22 sample_script
-rw-r--r-- 1 pakala gclab  284 Mar  6 09:50 sub.sh
drwxr-xr-x 2 pakala gclab    2 Feb 22 12:07 Suchi_Scripts
```

- **ls -lS**
  - Displays file size in order

```
pakala@uga-2f0f976:~ $ ls -lS
total 584496
-rw-r--r-- 1 pakala gclab 1610499990 Mar  6 09:46 e_coli_data.fq
-rw-r--r-- 1 pakala gclab        284 Mar  6 09:50 sub.sh
-rwxr----- 1 pakala gclab        136 Feb 21 15:22 sample_script
drwxr-xr-x 2 pakala gclab          2 Feb 22 12:07 Suchi_Scripts
```

- Files
- Permissions
- Creation, Deletion, Copy and Move Commands

Files And Processes

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- File
  - Collection of data
  - Location of a file – Path
  - Can be created using text editors (nano, vi, etc)

- Process
  - Any program that is run
  - Unique process identifier - PID
  - For example: “ps” command which lists all processes

```
pakala@uga-2f0f976:~ $ ps
  PID TTY          TIME CMD
21505 pts/225  00:00:00 bash
24908 pts/225  00:00:00 ps
```

Files And File Names

- File
  - Basic unit of storage for data
  - May contain any characters
  - File names are always **case sensitive**
  - You should **avoid spaces, quotes, and parenthesis**
  - File names can be long, up to 255 characters

- Directory
  - Special type of file
  - Holds information about other files
  - Present working directory (pwd)

```
pakala@uga-2f0f976:~ $ pwd
/home/pakala
```

File Permissions

- Multi-user environment
- File permissions are used to protect users and system files.
- The types of permissions a file can have are:

Read Permissions	Write Permissions	Execute Permissions
r	w	x

- Files and directories have three levels of permissions:
  - **User**
  - **Group**
  - **World**

User (owner)	Group	Others (everyone else)
rwx	rwx	rwx

File Permissions

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

File Type: - Regular file(d for Directory)

Group and world have read and execute permissions

Group

File size

Date modified

File name

```
-rwxr-xr-x    1    pakala   gclab    284k        Feb 22 12:07   hello.sh
```

User has read, write, and execute permissions on this file

# of hard links

User

Changing File Permissions

- **chmod** command to change permissions of a file.
  - Symbolic mode:
    - Syntax: chmod [references][operator][modes]
    - References – "**u**" for user, "**g**" for group, "**o**" others
    - "**a**" for all three types
    - The operator – "**+**" to add and "**-**" to remove

```
>> Default settings when file was created:
pakala@uga-2f0f976:~ $ ls -l
-rw-r--r-- 1 pakala gclab   24 Feb 15 10:35 sample_script
```

```
>> Adding x(excute) permission for the user:
$ chmod u+x sample_script
-rwxr--r-- 1 pakala gclab   24 Feb 15 10:45 sample_script
```

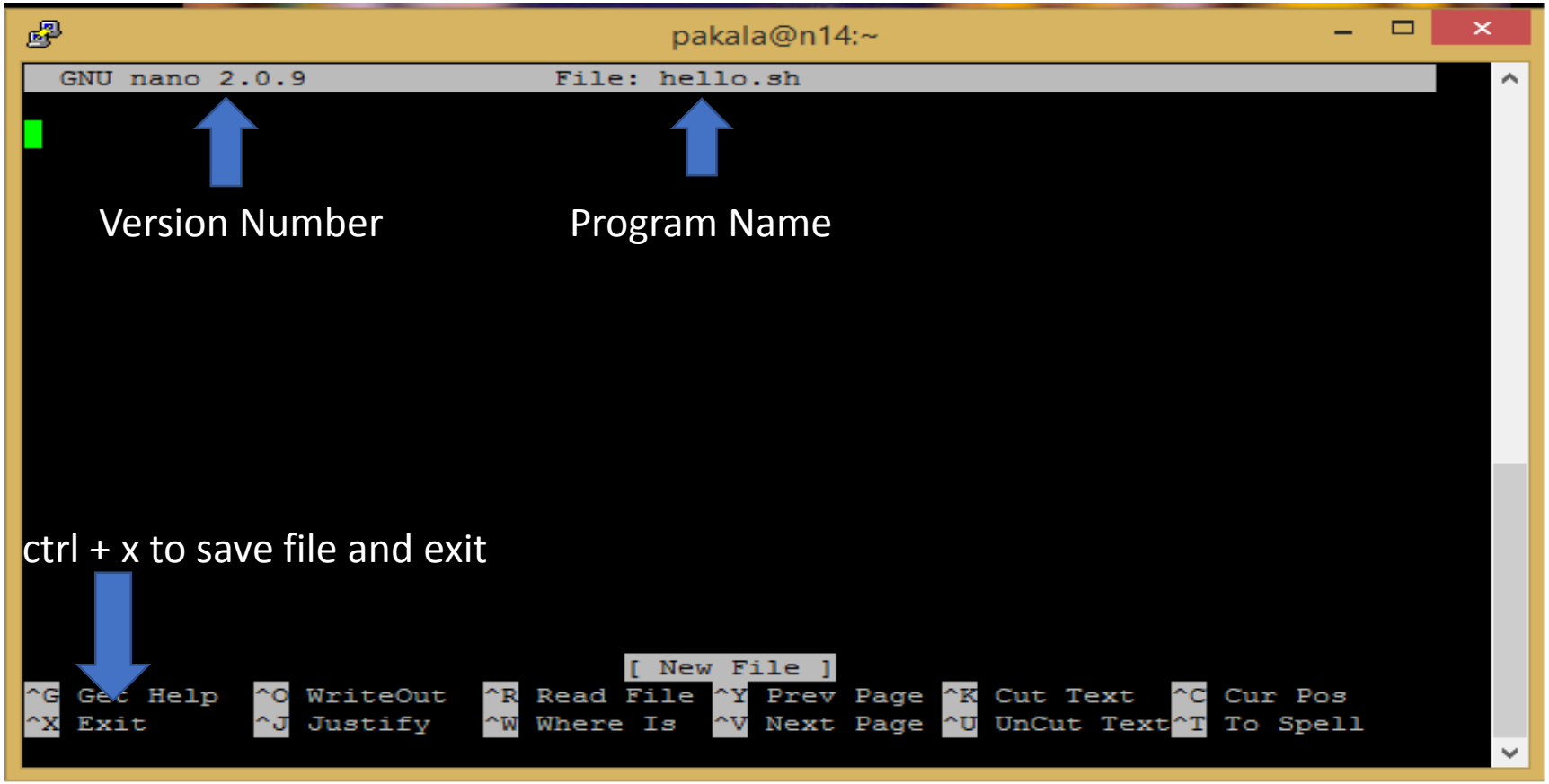
```
>> Removing r(read) permission for others:
$ chmod o-r sample_script
-rwxr----- 1 pakala gclab   24 Feb 15 10:50 sample_script
```

Creating and Editing Files

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Creating and editing files using a text editor
- The most widely used editors available on sapelo are vim, **nano**, etc

```
$ nano hello.sh
```



Creating and Deleting Directories

- **mkdir** creates a directory

```
$ mkdir testdir
```

- Creating directories and subdirectories in one step

```
$ mkdir -p <dirname>/<subdirname>
```

- **rmdir r**emoves an empty directory

```
$ rmdir testdir
```

- Remove directories

```
$ rm -ri <directoryname>        → Interactive Mode
```

- Removing Files

```
$ rm -i <filename>              → Interactive Mode
```

Remove Files (rm)

- **rm** removes files

```
$rm -i /home/pakala/sample_script
```

- **Other options:**

```
option      description
Remove (unlink) the FILE(s)
rm -f        ignore nonexistent files, never prompt
rm -i        prompt before every removal
rm -r, -R    remove directories and their contents recursively
rm -v        explain what is being done
```

- With the **-r** or **-R** option
  - Removes entire directories recursively and permanently!!!
- **rm -r *** option
  - Removes all of the files and subdirectories (**not recommended**)
- To remove an empty directory, use **rmdir**

Copy Files (cp)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **cp** copies files or directories.
- To copy a file from /home/pakala/sample_script to /home/pakala/Suchi_Scripts

```
$ cp -i /home/pakala/sample_script /home/pakala/Suchi_Scripts
```

- **Other Options**:

```
option        description
cp -a         archive files
cp -f         force copy by removing the destination file if needed
cp -i         interactive - ask before overwrite
cp -n         no file overwrite
cp -R         recursive copy (including hidden files)
cp -u         update - copy when source is newer than destination
cp -v         verbose - print informative messages
```

Move Files (mv)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **mv** moves a file to another location.
- For example, to move a file from /lustre1/pakala/AF293.fs to /lustre1/pakala/Sample_Data

```
$ mv -i AF293.fs /lustre1/pakala/Sample_Data
```

- Can also be used to **rename** a file in the same directory.
- For example, to rename *myFile* to *myFileNew*:

```
$ mv myFile myFileNew
```

- **Other options:**

```
option        description
mv -f         force move by overwriting destination file without prompt
mv -i         interactive prompt before overwrite
mv -u         update - move when source is newer than destination
mv -v         verbose - print source and destination files
man mv        help manual
```

Summary of Common Linux Commands

- **cd** : Change your current working directory
- **pwd** : Print absolute path of your current working directory
- **ls** : List the files that exist in the current directory
- **mv** : Moves a file to another location.
- **cp** : Copies files or directories
- **mkdir** : Create a directory
- **rmdir** : Delete an empty directory
- **rm –r** : Delete a nonempty directory and its contents

More Linux Commands

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- **file <filename>** : Display file type of file with name
- **cat textfile** : Throws content of text file on the screen
- **more <filename>** : Output the contents of a file
- **less <filename>** : Output the contents of a file
- **man <command>** : Read man pages command
- **dos2unix :** convert DOS/Windows file to Linux format
- **mac2unix:** convert mac file to Linux format
- **exit or logout**: leave the session

File Viewing

- **file** – determine the type of a file

```
pakala@uga-2f0f976:~ $ file Linux_Scripts/                    directory
Linux_Scripts/: directory
pakala@uga-2f0f976:~ $ file e_coli_data.fq                    ASCII text
e_coli_data.fq: ASCII text
pakala@uga-2f0f976:~ $ file hello.sh                          Shell Script
hello.sh: Bourne-Again shell script text executable
```

- **cat**
  - cat is a standard Linux utility that concatenates
  - Prints the content of a file to standard output

```
pakala@uga-2f0f976:~ $ cat temp.txt
Hello!!!!!
Welcome to Linux world!
```

File Viewing

- **more**
  - view text files - one page at a time, scroll down only
  - spacebar to scroll down

```
pakala@uga-2f0f976:~ $ more testfile
```

- **less**
  - view text files, one page at a time, scroll up and down
  - space bar to scroll down
  - key **b** to scroll up, Key **q** to quit

```
pakala@uga-2f0f976:~ $ less testfile
```

Manual Pages (man)

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Linux includes a built in manual for nearly all commands.
- Example: **man rm** (remove)

```
$ man rm
RM(1)                            User Commands                          RM(1)

NAME
    rm - remove files or directories

SYNOPSIS
    rm [OPTION]... FILE...

DESCRIPTION
    This  manual page documents the GNU version of rm.  rm removes each
    specified file. By default, it does not remove directories.

     If the -I or --interactive=once option is given,  and  there  are  more
     than  three  files  or  the  -r,  -R, or --recursive are given, then rm
     prompts the user for whether to proceed with the entire operation.
```

```
OPTIONS
    Remove (unlink) the FILE(s).

       -f, --force
              ignore nonexistent files, never prompt

       -i     prompt before every removal

       -r, -R, --recursive
              remove directories and their contents recursively
```

File Conversion

- **dos2unix** : Convert DOS/Windows file to Linux format
  - Example: dos2unix file1
  - Removes DOS/Windows line endings in file1

```
$ dos2unix file1
```

- **mac2unix** : Convert Mac file to Linux format
  - Example: mac2unix file1
  - Removes Mac line endings in file1

```
$ mac2unix file2
```

- Shell Scripting
- Script Execution

Shell Scripting

- Shell Script - series of commands written in plain text file

- Why to write Shell Script?
  - To automate tasks that should be run daily
  - Build “pipelines” of commands and other programs to run
  - Serve as automatic documentation
  - Useful to create our own commands
  - Save lots of time

Example Script


```
#!/bin/bash

# rsync using variables

SOURCEDIR=/home/pakala/Linux_Scripts
DESTDIR=/lustre1/pakala/backup_files/

rsync -avh $SOURCEDIR $DESTDIR

# compressing directory

compress=Linux_Scripts_$(date +%Y%m%d).tar.gz
tar -czf $compress /home/pakala

# Simple if/else statement, checking if the directory exists or not

directory="./Suchi_Scripts"

if [ -d $directory ]; then
        echo "Directory exists"
else
        echo "Directory does not exist"
fi
```

Variables in Shell

- What is a **"variable"**?
  - A character string to which we assign a value
  - Value could be a number, text, filename or any other type of data
  - Pointer to the actual data

- **There are two types of variables:**
  - System variables
  - User defined variables

- **System variables**
  - Created and maintained by Linux
  - Defined in CAPITAL LETTERS, user can reset their default values

System Variables

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

System Variable	Meaning	Example Value
HOME	User's home directory	/home/pakala
PATH	Path to binaries	/usr/bin:/sbin:/bin:/usr/sbin
PWD	Current working directory	/home/pakala
SHELL	Path to default shell	/bin/bash
USER	User who is currently logged in	pakala
TERM	Login terminal type of user	xterm
LD_LIBRARY_PATH	Shared library search path	

```
pakala@uga-2f0f976:~ $ echo $SHELL
/bin/bash
```

```
pakala@uga-2f0f976:~ $ echo $HOME
/home/pakala
```

User Defined Variables

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- Created and maintained by user, defined in lower letters
- Syntax: **variable name=value**
- Rules for naming variable name
  - Don't put spaces on either side of the equal sign
  - Variables are **case sensitive**
  - Do not use ?,* etc, to name your variable names
- To print or access user defined variables
  - Syntax: **$variable name**

```
$ no=10
$ echo $no            #will print 10
$ no =25              #no spaces on either side of equal sign
-bash: no: command not found
$ No=11
$ echo $No            #case sensitive, will print 11
```

Example Script – breaking it down

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

```
#!/bin/bash                                  Location of shell to use

# rsync using variables                      Comment line

SOURCEDIR=/home/pakala/Linux_Scripts
DESTDIR=/lustre1/pakala/backup_files/

rsync -avh $SOURCEDIR $DESTDIR               Actual command to run

# compressing directory

compress=Linux_Scripts_$(date +%Y%m%d).tar.gz
tar -czf $compress /home/pakala

# Simple if/else statement, checking if the directory exists or not

directory="./Suchi_Scripts"

if [ -d $directory ]; then
        echo "Directory exists"
else
        echo "Directory does not exists"
fi
```

Run Shell Script

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

```
$ chmod u+x sample_script.sh
```

Adding execute permission for User

```
$ ./sample_script.sh
```

Running the script

```
sending incremental file list
created directory /lustre1/pakala/backup_files
Linux_Scripts/
Linux_Scripts/.swp
Linux_Scripts/car.sh
Linux_Scripts/file2
Linux_Scripts/file2.sh
Linux_Scripts/first.sh
Linux_Scripts/forloop.sh
Linux_Scripts/sample
Linux_Scripts/sampledata.sh
Linux_Scripts/samplescript.sh
Linux_Scripts/test1.sh
Linux_Scripts/whileloop.sh
Linux_Scripts/sample1/

sent 14.68K bytes  received 229 bytes  29.82K bytes/sec
total size is 13.89K  speedup is 0.93
tar: Removing leading `/' from member names
tar: /home/pakala: file changed as we read it
Directory exists
```

.bashrc

Georgia Advanced Computing Resource Center
UNIVERSITY OF GEORGIA

- .bashrc is a shell script that Bash runs whenever it is started interactively.
  - Think about all the startup programs that run when you start Windows
- It initializes an interactive shell session. You can put any command in this file that you would type at the command prompt
- A common thing to put in .bashrc are aliases that you want to always be available

```
# .bashrc
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
# User specific aliases and functions
export PATH=/home/pakala/bin:$PATH

alias ls='ls --color=auto -l'
alias p="pwd"
```

THANK YOU ☺