

Express Introduction to Linux (A Quick and Short Introduction)

Ritu Arora

Email: rauta@tacc.utexas.edu

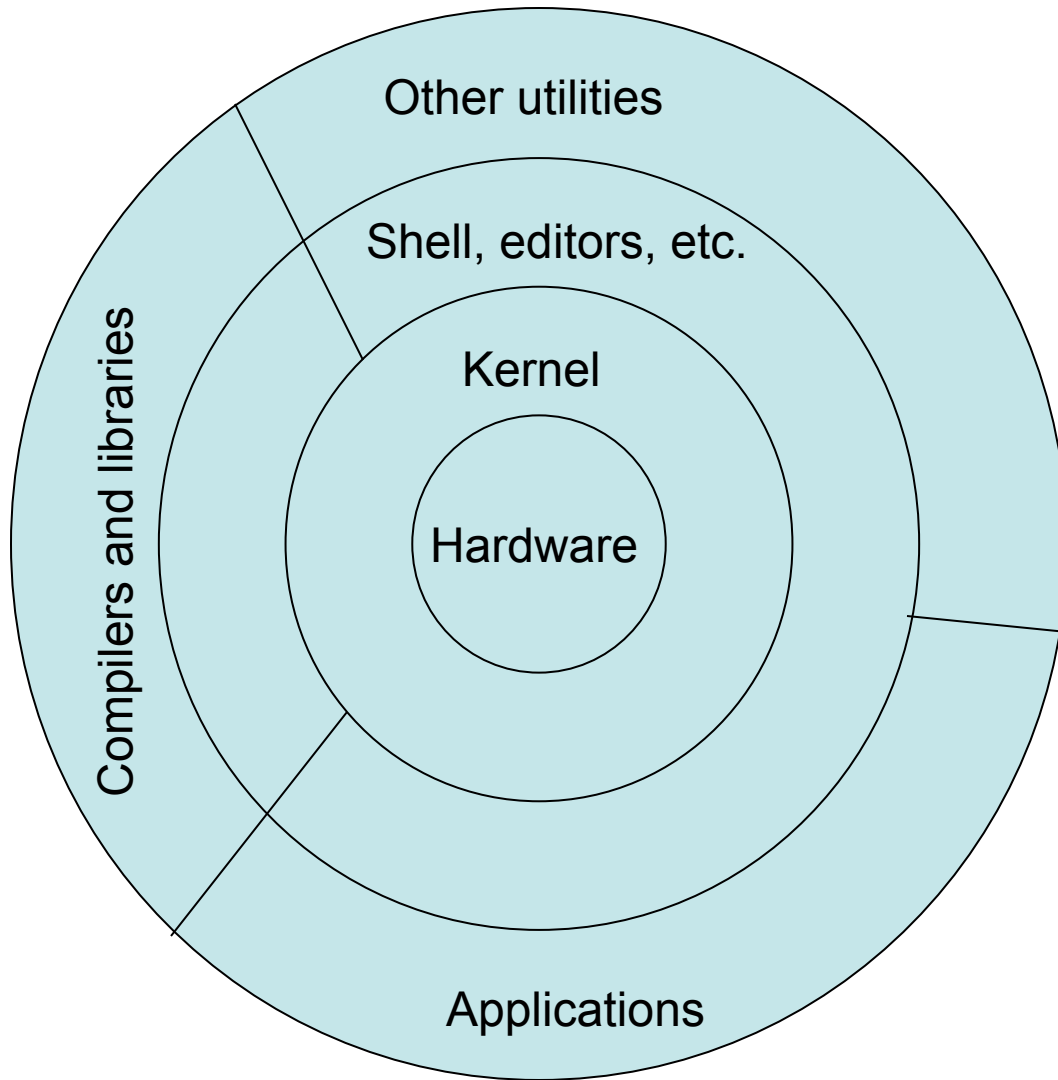
Overview

- Definition of Operating System (slide 3)
- Access to Linux (slide 7-14)
- Basic Commands (slide 16-27)
- Other Useful Commands (slides 29 -40)

What is an Operating System (OS)?

- Software interface between the user and the computer hardware
- Controls the execution of other programs
- Responsible for managing multiple computer resources (CPU, memory, disk, display, keyboard, etc.)
- Examples of OS: Windows, Unix/Linux, OS X

How does the Linux OS work?



- Linux has a kernel and one or more shells
- The shell is the command line interface through which the user interacts with the OS. Most commonly used shell is "bash"
- The kernel sits on top of the hardware and is the core of the OS; it receives tasks from the shell and performs them

Linux File System

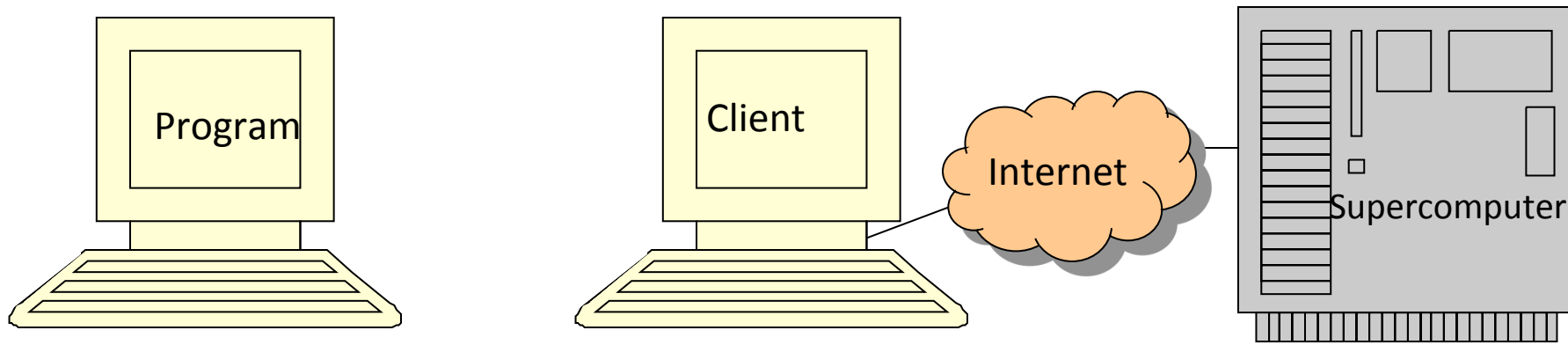
- A file is a basic unit of storage and should have a name associated with it – please note – Linux is case-sensitive
- Files are organized into directories and sub-directories
- A directory in Linux is a special file
 - A directory is similar to a “Folder” in Windows OS
- In Linux, paths begin at the root directory which is the top-level of the file system and is represented as a forward slash (/)
 - Relative path versus absolute path
- Forward slash is used to separate directory and file names

Overview

- Definition of Operating System
- **Access to Linux**
- Basic Commands
- Other Useful Commands

Local Access vs. Remote Access

- Local (Desktop/Laptop)
- Remote (Servers)



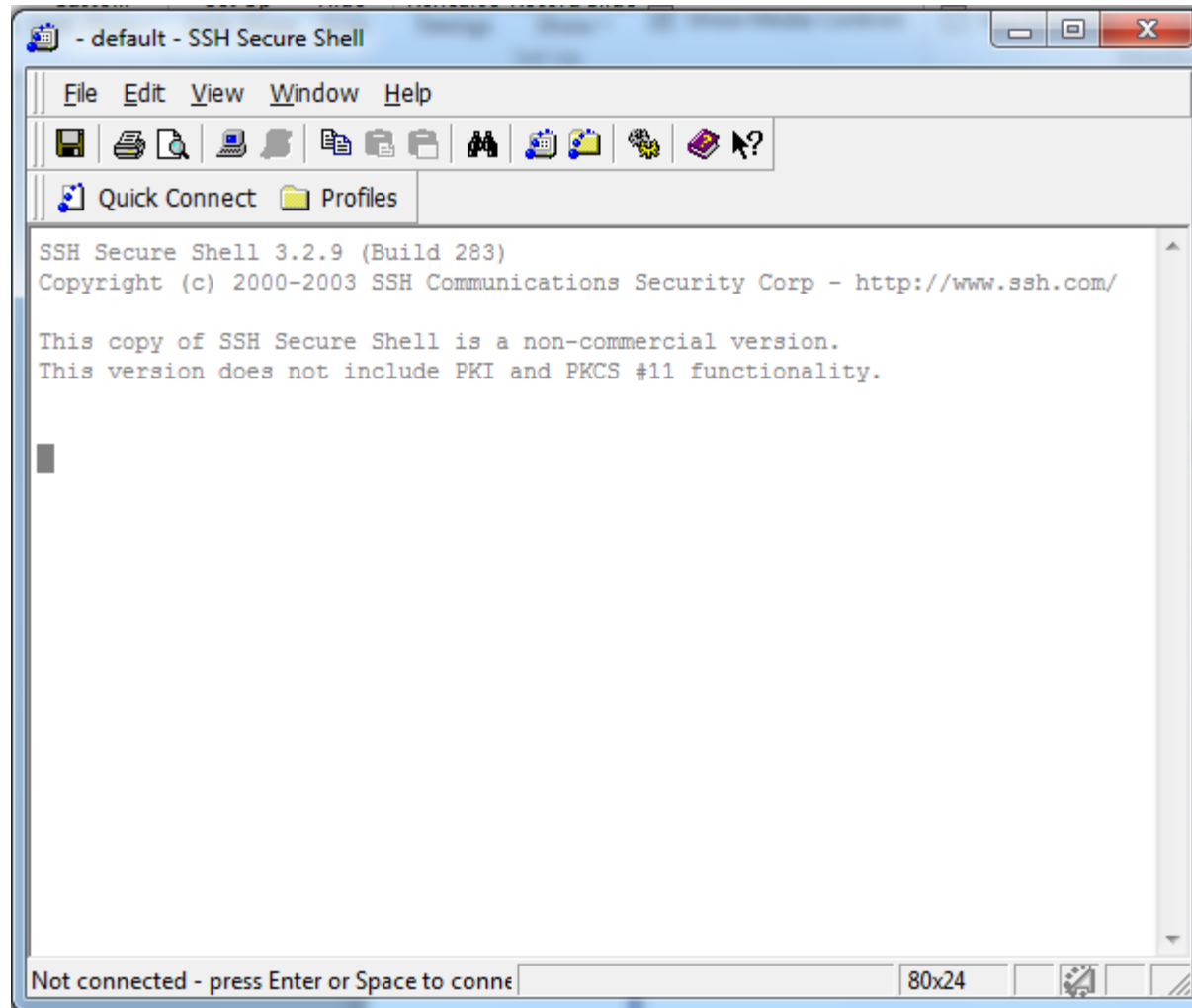
Install Linux locally on your computer, or access a Linux system remotely

Access to Linux from a Windows Computer

- For local access to Linux
 - Install Linux on a USB stick: <http://www.pendrivelinux.com/>
 - <http://www.pendrivelinux.com/all-in-one-usb-dsl/>
 - <http://www.ubuntu.com/download/desktop/create-a-usb-stick-on-windows>
 - Use Cygwin/VM Ware (runs as a windows process)
- For remote access to a Linux system, use client programs (like SSH Secure Shell Client or Putty) on a Windows computer
 - SSH Secure Shell Client (has a convenient way to transfer small files)
<https://shareware.unc.edu/>
 - PuTTY
<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
- Video showing the usage of SSH secure shell client
<https://www.youtube.com/watch?v=cigMNqXIkRE>

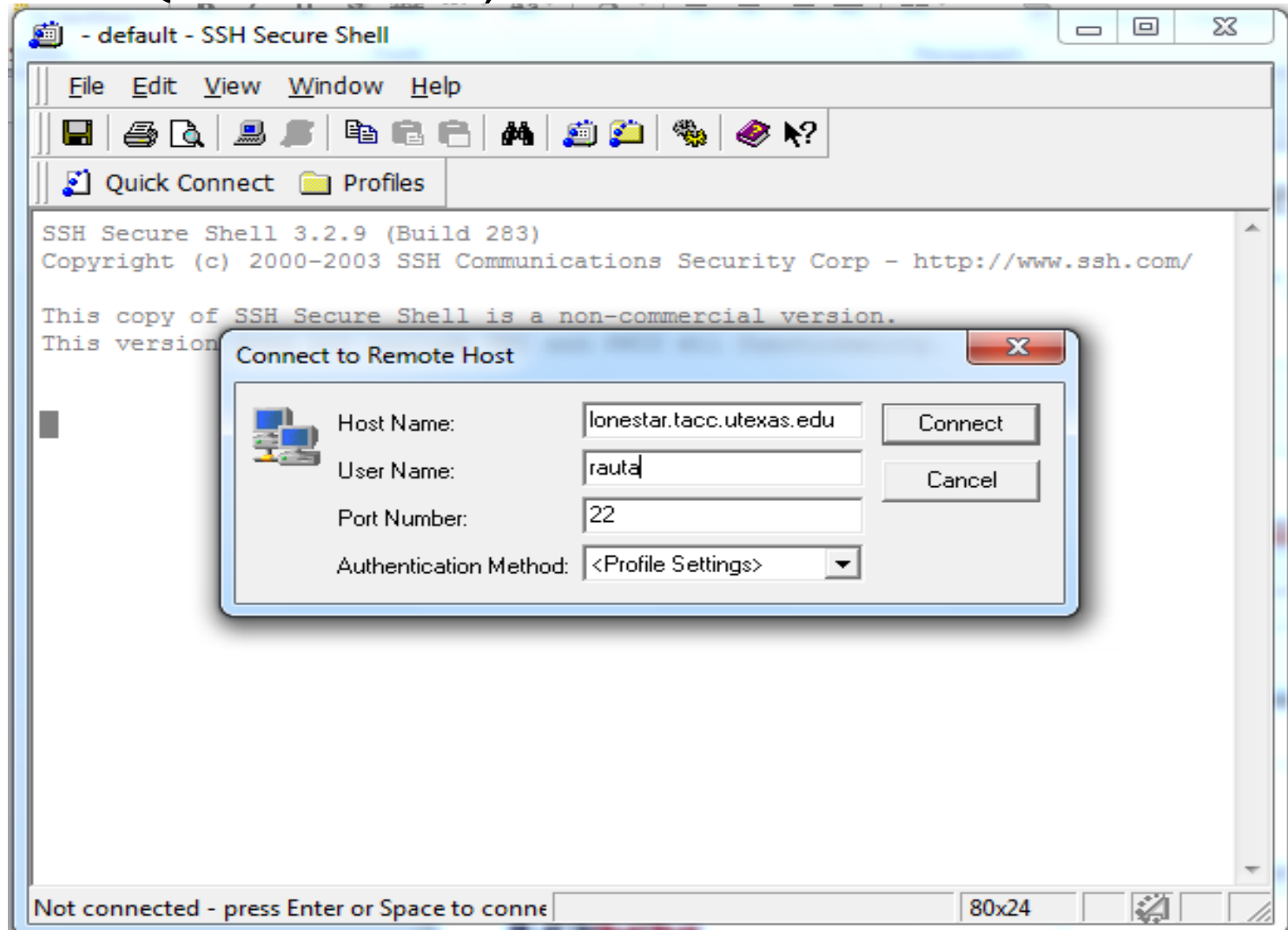
Using SSH Secure Shell Client - Step 1

- On Windows, double click on the SSH Secure Shell Client, the following window will appear



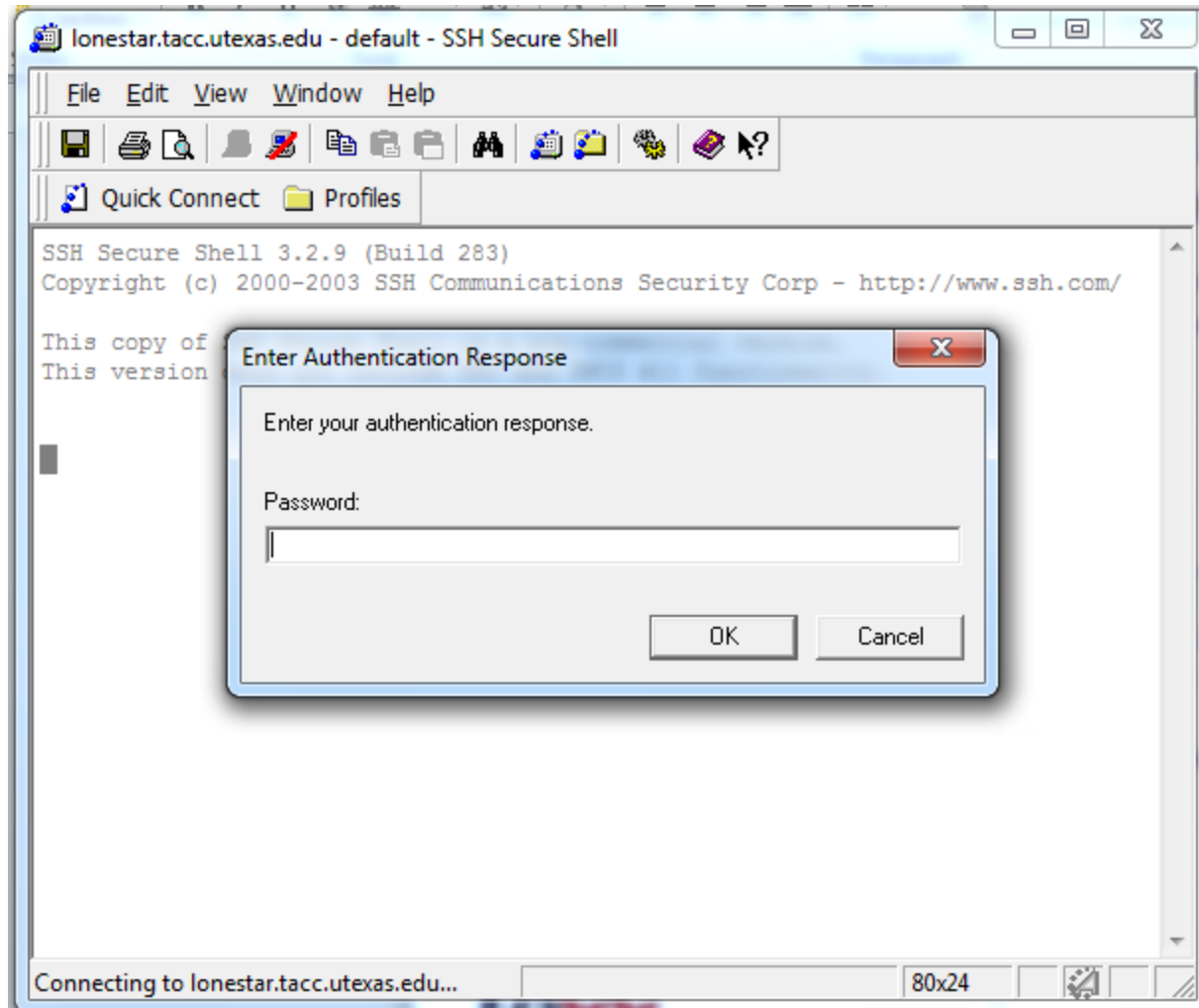
Using SSH Secure Shell Client - Step 2

- Click on “Quick Connect”, enter “Host Name” and “Username”



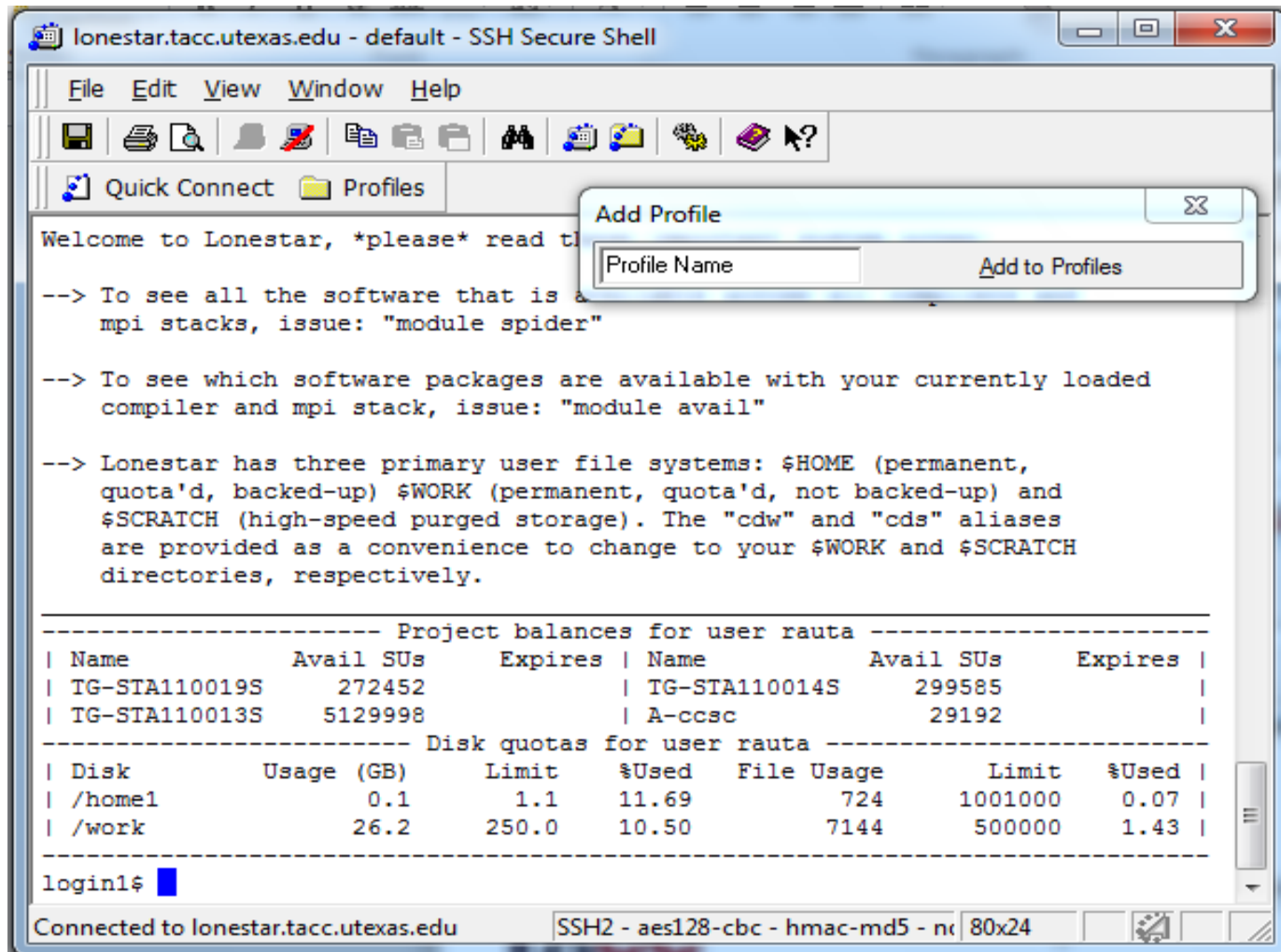
Using SSH Secure Shell Client - Step 3

- Click on “Quick Connect”, enter “Host Name”, “Username”, click “Connect”, enter password, click on “OK” for “Enter Authentication...”



Using SSH Secure Shell Client - Step 4

- Enter commands at the command prompt



lonestar.tacc.utexas.edu - default - SSH Secure Shell

File Edit View Window Help

Quick Connect Profiles

Welcome to Lonestar, *please* read the following instructions:

--> To see all the software that is available with your currently loaded compiler and mpi stacks, issue: "module spider"

--> To see which software packages are available with your currently loaded compiler and mpi stack, issue: "module avail"

--> Lonestar has three primary user file systems: \$HOME (permanent, quota'd, backed-up) \$WORK (permanent, quota'd, not backed-up) and \$SCRATCH (high-speed purged storage). The "cdw" and "cds" aliases are provided as a convenience to change to your \$WORK and \$SCRATCH directories, respectively.

----- Project balances for user rauta -----

Name	Avail SUs	Expires	Name	Avail SUs	Expires
TG-STA110019S	272452		TG-STA110014S	299585	
TG-STA110013S	5129998		A-ccsc	29192	

----- Disk quotas for user rauta -----

Disk	Usage (GB)	Limit	%Used	File Usage	Limit	%Used
/home1	0.1	1.1	11.69	724	1001000	0.07
/work	26.2	250.0	10.50	7144	500000	1.43

login1\$

Connected to lonestar.tacc.utexas.edu SSH2 - aes128-cbc - hmac-md5 - n 80x24

For Mac Users

- You can have remote access to servers through your “Terminal” application (Look under “Applications” -> “Utilities”)
- After opening the terminal, type the SSH command below (replace `username` with your actual username) – you will be prompted for password after that

```
staff$ ssh username@stampede.tacc.utexas.edu
```

Interacting with the Shell

- Type a command (**ls**) at the prompt (**login3\$**) and press ENTER
Example: **login3\$ ls**
- Shell starts a new process for executing the requested command, the new process executes the command and the shell displays any output generated by the command
- When the process completes, the shell displays the prompt and is ready to take the next command
- Specific information is passed to the command via more arguments
- The shell is killed by “**exit**” or **CTRL-D**

login3\$ exit

logout



Overview

- Definition of Operating System
- Access to Linux
- **Basic Commands**
- Other Useful Commands

Basic Commands (1)

- To print the name of the current/working directory, use the `pwd` command

```
login4$ pwd  
/home1/01698/rauta
```

- To make a new directory, use the `mkdir` command

```
login4$ mkdir ssc222
```

- To change your working directory, use the `cd` command

```
login4$ cd ssc222
```


Basic Commands (2)

- To create a new file use the `vi` command (see cheat-sheet)
`login4$ vi test.txt`
 - Press **i** to start **inserting** text
 - Type some text: `Hello Class 222`
 - To **save and quit**, press “ Esc ” key, and enter **:wq!**
(press the enter key after typing **:wq!**)
 - To **quit without saving**, press “ Esc ” key if in insert mode, and enter “ **:q!** ”
- To display the contents of the file, use the `cat` (short for concatenation) command

```
login4$ cat test.txt
```

Basic Commands (3)

- To list the contents of a directory, use the `ls` command

```
login4$ ls
```

- To see all files and directories, including hidden ones use the `-a` flag with the `ls` command. Hidden files have a “.” in front of them

```
login4$ ls -a
```

Note: your current working directory can be checked by using the `pwd` command.

Basic Commands (4)

- To copy contents of one file to another, use the `cp` command

```
login4$ cp test.txt copytest.txt
```

```
login4$ cp test.txt test3.txt
```

One more example:

```
login4$ mkdir junk
```

```
login4$ cp test.txt ./junk/test2.txt
```

Relative path example



(The command above copies a file to the sub-directory `junk`)

```
login4$ cd junk
```

```
login4$ ls
```

```
login4$ cd ..
```

- To go a level up from the current working directory

```
login4$ cd ..
```

Exercise -1 (Part A)

- Run the following commands to make a directory:

```
login1$ mkdir ssc229
```

```
login1$ cd ssc229
```

- Create a file using vi command in `ssc229` (see slide 17)

```
login1$ vi test.txt
```

- Run the following commands in the `ssc229` directory

```
login1$ cp test.txt test2.txt
```

```
login1$ mkdir junk
```

```
login1$ mkdir junk2
```

```
login1$ cp test2.txt ./junk/test2.txt
```

```
login1$ cp test2.txt ./junk2/test2.txt
```

```
login1$ ls
```

Exercise -1 (Part B)

- Run the following commands starting from the `ssc229` directory that you created in Part A of Exercise-1

```
login1$ ls  
login1$ cd junk  
login1$ ls  
login1$ cd ..  
login1$ cd junk2  
login1$ ls  
login1$ cd ..  
login1$ ls  
login1$ cp test.txt test3.txt
```

Basic Commands (5)

- To remove a file, use the `rm` command

```
login4$ rm test2.txt
```

- To remove a directory, use the “ `-r` ” option with the `rm` command

```
login4$ rm -r junk2
```

- You can also use the `rmdir` command to remove an empty directory

```
login4$ rmdir junk2
```

Note: `rmdir` command does not have `-r` option

Basic Commands (6)

- A file can be renamed by moving it. The same can be achieved by using the `mv` command

```
login4$ mv test3.txt newtest3.txt
```

- Use the `man` command to get more information about a command – it is like using help in Windows

```
login4$ man rmdir
```

- Use the `diff` command to see the differences in two files

```
login4$ diff test.txt newtest3.txt
```

Basic Commands (7)

- Previously executed commands in a shell can be viewed by using the `history` command. For example:

```
login4$ history
```

```
1  man ls
```

```
2  ls -ltr
```

```
3  ls -l -t -r
```

```
4  ls -ltr
```

```
5  history
```


Basic Commands (8)

- If the contents to display are more than one page, you could use the `more/less` command for paging through text a screenful at a time

```
login4$ more test.txt
```

```
login4$ less test.txt
```

(`less` allows both fwd and bwd movement)

Basic Commands (9)

Creating a tarball

- TAR (Tape Archive) command bundles files and sub-directories together and creates an archive (known as tar file or tarball)
- To create a tarball of all the files and sub-directories in the directory ssc229 that you created in Exercise 1, use **c** flag:

```
tar -cvf mytar.tar *
```

- To extract the contents of a tar file use **x** flag:

```
login1$ tar -xvf mytar.tar
```

Basic Commands (10)

Creating a Compressed tarball

- To compress the tar file as it is being created use **z** flag with **c** flag :

```
login1$ tar -cvzf mytar.tar.gz *
```

- To extract the contents of a compressed tar file use **x** flag:

```
login1$ tar -xvf mytar.tar.gz
```

Note: the **c**, **v**, and **f** flags mean create a new archive, be verbose so that the files being archived are listed, and write the archive to a file.

Overview

- Definition of Operating System
- Access to Linux
- Basic Commands
- Other Useful Commands

Check Username and Group

- Three types of users: owner or user, group, all others
- To check the login name use the command **whoami** or **echo \$USER**
- To check the groups you are a member of use the command **groups**
 - Members of a Linux group can share files with each other
 - Each account is assigned a primary group, and an account can be a member of multiple groups
- To check your user id, or group id use the command **id**

File Permissions (1)

- Users typically perform the following operations on files:
 - Read files (using `more`, `cat`, *etc.*)
 - Write files (using `>`, `vi`, *etc.*)
 - Execute commands in a file (executables, *etc.*)
- Each file has three permissions – read, write and execute (`rwX`)
- Person creating the file is the owner or user and can modify permissions as desired
 - Owner can modify permissions on files to grant or revoke access to other users

File Permissions (2)

- To check the file permissions use the `-l` flag with the `ls` command

```
login4$ ls -l
total 24
drwx----- 2 rauta G-25072 4096 Jan 17 14:07 junk
drwx----- 2 rauta G-25072 4096 Jan 17 14:15 junk2
-rw----- 1 rauta G-25072   65 Jan 17 13:59 test.txt
```

File Permissions (3)

- `chmod` command is used to change permissions on a file
- To add specific permission use `chmod +`
 - To add write permission to all users use:
`chmod a+w filename`
 - To add read permission to only the users in your group use:
`chmod g+r filename`
 - To make a file executable and runnable by any user
`chmod a+x myfile`
- To remove specific permission use `chmod -`
- Add and remove permissions can be combined in a single step
 - **`chmod u+x,g+r,o-rwx filename`**

Note: u = user or owner, g = group, o = other

File Permissions (4)

- Instead of using alphabets `u`, `g`, `o` for user, group, and others we can use numbers to specify file permissions

`rx` = 111 = 7

`rw` = 110 = 6

`r-x` = 101 = 5

`rw-` = 100 = 4

`-wx` = 011 = 3

`-w-` = 010 = 2

`--x` = 001 = 1

`---` = 000 = 0

- `chmod go+rx filename = chmod 755 filename`
(assuming the user already has the `r`, `w`, and `x` permissions.)

Directory Permissions

- To check the contents of a file with `ls` command, you would need read permission
- To add or remove files in a directory, you would need write and execute permission
- To change to a directory or to go through its contents, you would need execute permission
- To list files in a directory using `ls -l` command you would need read and execute permissions

Redirecting Output

- By default, the output is displayed on the screen
- “>” symbol can be used to redirect the output to a file or a utility (e.g., `ls`). Example:
`ls -ltr > myContent`
- The “|” symbol is used to connect the output of one process to the input of another process

`ls -l | wc -l`

`wc` counts the number of lines

Other Directives

- “<” symbol is used for input redirection

```
mail -s "SSC 222/292" rauta@tacc.utexas.edu < test.txt
```

- “>>” symbol is used for appending output to a file

```
login4$ cat test3.txt >> test.txt
```

- “;” is used to execute multiple commands in one step

```
login4$ clear;date
```

Adding Content to a File

- You can add content to a file as follows

```
login4$ cat > test.txt
```

This is what I am entering from the console
CTRL-D

```
login4$ cat test.txt
```

This is what I am entering from the console

- You can append content to a file as follows

```
login4$ cat >> test.txt
```

Appending more lines
CTRL-D

Editing in Unix

- Text-mode editors that do not require an X-server to be running on your PC
 - pico is easiest editor to learn
 - emacs is most powerful editor and has a built-in tutorial
 - vi is present on essentially all Unix systems
 - GNU nano was supposed to be a free replacement of the pico editor
- If you have an X-server running
 - textedit
 - xedit

Data Transfer Using **scp** or WinSCP

- If your local computer is a Mac or a Linux laptop, you can use the `scp` commands to transfer data to and from a remote resource like Stampede

```
localhost% scp filename  
username@stampede.tacc.utexas.edu: /path/to/project/  
directory
```

- If you are using a Windows computer, you can download and use the WinSCP application (GUI-based), or download and use Cygwin (command-line based, can run the aforementioned commands)
 - For small amounts of data, you may also use the “File Transfer Window” available in the SSH client – drag and drop the files across the local laptop and a remote resource

Process and Process Control

- `ps` – display process information on the system
- `kill pid` – terminates the process id
- `^C` (CTRL+c) terminates the running program

```
login4$ ps
```

PID	TTY	TIME	CMD
20482	pts/32	00:00:00	bash
21035	pts/32	00:00:00	ps

References

- <http://www.cs.jhu.edu/~joanne/unixRC.pdf>
- <http://www.digilife.be/quickreferences/qrc/vi%20reference%20card.pdf>
- http://www.tacc.utexas.edu/documents/13601/118360/LinuxIntro_HPC_09+11+2011_hliu.pdf