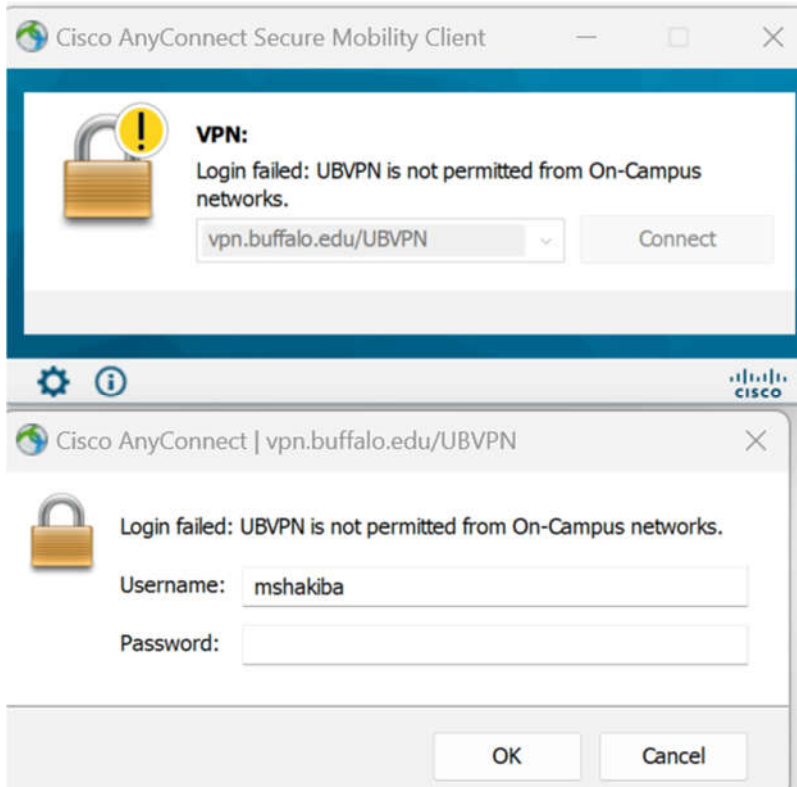
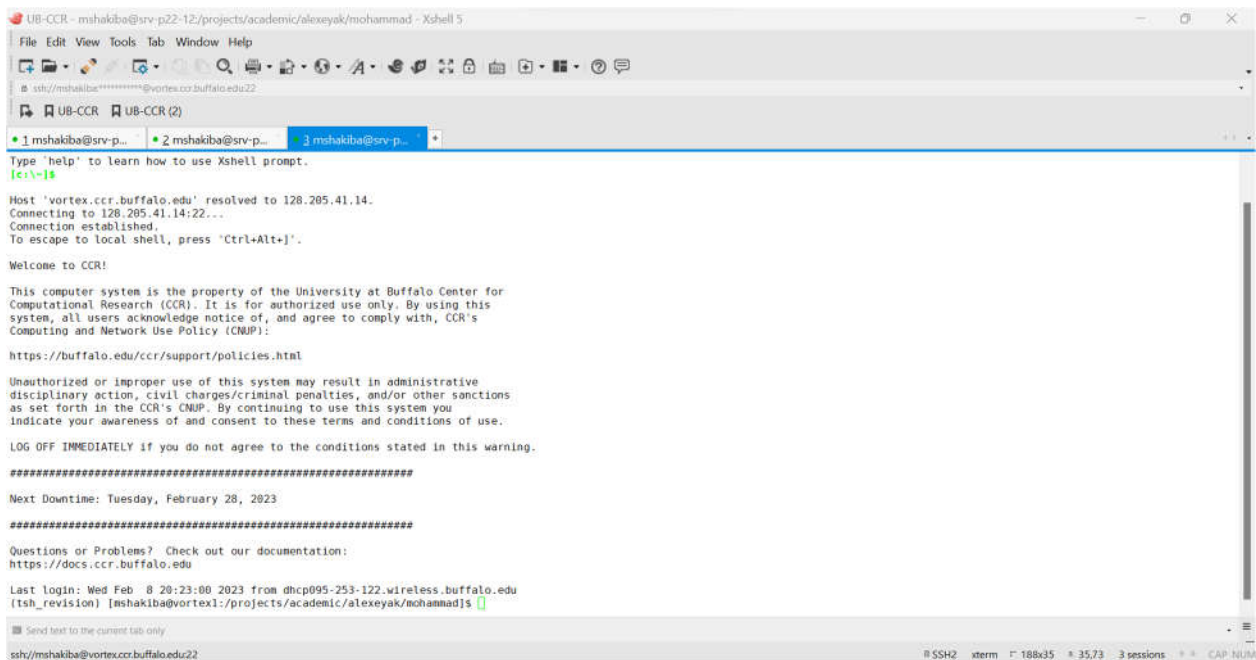


Mohammad Shakiba, 50442122, CE451 Homework 2:

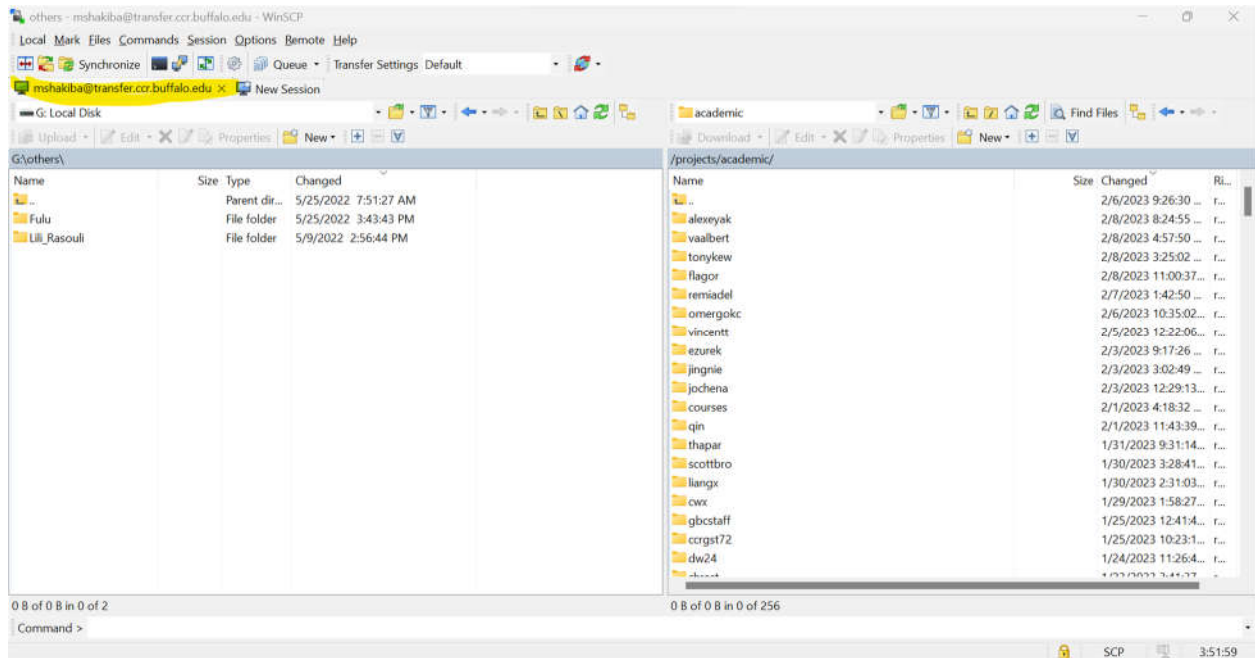
Connecting to UBVPN: It does not let me connect to UBVPN from On-Campus networks.



Connecting to CCR using XShell:

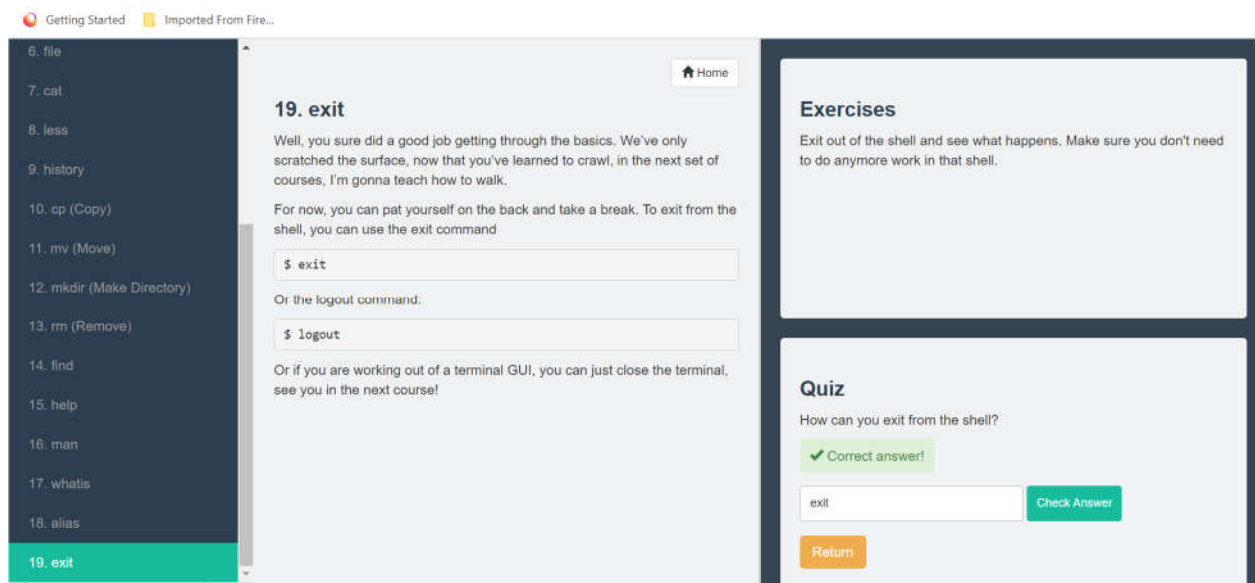


Connecting to CCR using WinSCP:



LinuxJourney did not give me a proof of completion for the courses so I only screenshotted the last part with answer. The examples are attached in the history txt file.

Command line:



Text Fu:

3. stderr (Standard Error)

4. pipe and tee

5. env (Environment)

6. cut

7. paste

8. head

9. tail

10. expand and unexpand

11. join and split

12. sort

13. tr (Translate)

14. uniq (Unique)

15. wc and nl

16. grep

16. grep

The grep command is quite possibly the most common text processing command you will use. It allows you to search files for characters that match a certain pattern. What if you wanted to know if a file existed in a certain directory or if you wanted to see if a string was found in a file? You certainly wouldn't dig through every line of text, you would use grep!

Let's use our sample.txt file as an example:

```
$ grep fox sample.txt
```

You should see that grep found fox in the sample.txt file.

You can also grep patterns that are case insensitive with the -i flag:

```
$ grep -i somepattern somefile
```

To get even more flexible with grep you can combine it with other commands with |.

```
$ env | grep -i User
```

As you can see grep is pretty versatile. You can even use regular expressions in your pattern:

```
$ ls /somedir | grep '.txt$'
```

Should return all files ending with .txt in somedir.

Exercises

You may have heard of egrep or fgrep, these are deprecated grep calls and have since been replaced by grep -E and grep -F. Read the grep manpage to learn more.

Quiz

What command do you use to find a certain pattern?

✓ Correct answer!

grep

Check Answer

Return

Permissions:

Permissions

1. File Permissions

2. Modifying Permissions

3. Ownership Permissions

4. Umask

5. Setuid

6. Setgid

7. Process Permissions

8. The Sticky Bit

Home

8. The Sticky Bit

One last special permission bit I want to talk about is the sticky bit.

This permission bit, "sticks a file/directory" this means that only the owner or the root user can delete or modify the file. This is very useful for shared directories. Take a look at the example below:

```
$ ls -ld /tmp
drwxrwxrxt 6 root root 4096 Dec 15 11:45 /tmp
```

You'll see a special permission bit at the end here t, this means everyone can add files, write files, modify files in the /tmp directory, but only root can delete the /tmp directory.

Modify sticky bit

```
$ sudo chmod +t mydir
$ sudo chmod 1755 mydir
```

The numerical representation for the sticky bit is 1

Exercises

What other files and directories do you think have a sticky bit enabled?

Quiz

What symbol represents the sticky bit?

✓ Correct answer!

t

Check Answer

Return

Processes:

Getting StartedImported From Fire...

Processes

1. ps (Processes)

2. Controlling Terminal

3. Process Details

4. Process Creation

5. Process Termination

6. Signals

7. kill (Terminate)

8. niceness

9. Process States

10. /proc filesystem

11. Job Control

```
pete@icebox ~ $ bg
[4]+  sleep 1003 &

pete@icebox ~ $ jobs

[1]    Running      sleep 1000 &
[2]    Running      sleep 1001 &
[3]-   Running      sleep 1002 &
[4]+   Running      sleep 1003 &
```

Moving a job from the background to the foreground

To move a job out of the background just specify the job ID you want. If you run fg without any options, it will bring back the most recent background job (the job with the + sign next to it)

Kill background jobs

Similar to moving jobs out of the background, you can use the same form to kill the processes by using their Job ID.

Exercises

Move some jobs between the background and the foreground

Quiz

What command is used to list background jobs?

✓ Correct answer!

The Filesystem:

93401 -rw-rw-r-- 2 pete pete 8 Jan 21 21:36 myhardlink

The Filesystem

1. Filesystem Hierarchy

2. Filesystem Types

3. Anatomy of a Disk

4. Disk Partitioning

5. Creating Filesystems

6. mount and umount

7. /etc/fstab

8. swap

9. Disk Usage

10. Filesystem Repair

11. Inodes

12. symlinks

A hardlink just creates another file with a link to the same inode. So if I modified the contents of myfile2 or myhardlink, the change would be seen on both, but if I deleted myfile2, the file would still be accessible through myhardlink. Here is where our link count in the ls command comes into play. The link count is the number of hardlinks that an inode has, when you remove a file, it will decrease that link count. The inode only gets deleted when all hardlinks to the inode have been deleted. When you create a file, it's link count is 1 because it is the only file that is pointing to that inode. Unlike symlinks, hardlinks do not span filesystems because inodes are unique to the filesystem.

Creating a symlink

To create a symbolic link, you use the ln command with -s for symbolic and you specify a target file and then a link name.

Creating a hardlink

Similar to a symlink creation, except this time you leave out the -s.

Exercises

Play around with making symlinks and hardlinks, delete a couple and see what happens.

Quiz

What is the command used to make a symlink?

✓ Correct answer!

Process Utilization:

Process Utilization

1. Tracking processes: top
2. lsof and fuser
3. Process Threads
4. CPU Monitoring
5. I/O Monitoring
6. Memory Monitoring
7. Continuous Monitoring
8. Cron Jobs

would be a good point to mention a neat tool in Linux that is used to schedule tasks using cron. There is a service that runs programs for you at whatever time you schedule. This is a really useful if you have a script you want to run once a day that needs to execute something for you.

For example, let's say I have a script located in `/home/pete/scripts/change_wallpaper`. I use this script every morning to change the picture I use for my wallpaper, but each morning I have to manually execute this script. Instead what I can do is create a cron job that executes my script through cron. I can specify the time I want this cron job to run and execute my script.

```
30 08 * * * /home/pete/scripts/change_wallpaper
```

The fields are as follows from left to right:

- Minute - (0-59)
- Hour - (0-23)
- Day of the month - (1-31)
- Month - (1-12)
- Day of the week - (0-7). 0 and 7 are denoted as Sunday

The asterisk in the field means to match every value. So in my above example, I want this to run every day in every month at 8:30am.

To create a cronjob, just edit the crontab file:

```
crontab -e
```

Exercises

Create a cronjob that you want to run at a scheduled time.

Quiz

What is the command to edit your cronjobs?

✓ Correct answer!

Check AnswerReturn