

CS 360 Linux Tutorial

<http://ryanstutorials.net/linuxtutorial/cheatsheet.php>

Change directory to your home directory.

Remember, ~ means your home directory

cd ~

Check to see your current working directory

pwd

You should be in /home/PUNetID

Let's see what time it is

date

Let's see the files and directories in your home directory.

ls

Let's see some more information about those files and directories

The **-l** option (dash ell) displays the long entry for the file or

directory

ls -l

Let's see who else has a home directory on this machine

the **..** (period period) means to look in the directory above the

current directory

In this case, that is the directory /home.

ls -l ..

In this view, you also see the permissions of the directories.

your home directory is marked as

drwx-----

which means **d**: directory

rwx : the owner can read, write, and execute,

--- : (middle three dashes) the group has no permissions

--- : (rightmost three dashes) the world has no permissions

These permissions allow you access to your home directory but

no one else has access. You can notice the group is listed in the

middle of the line, **users**.

Check to make sure you are still in your home directory. Using

ls does not move you even if you use the **..** option.

```
# Let's make a new directory inside of our home directory
mkdir cs360_test
```

```
# Check to make sure the new directory is listed
```

```
# Let's move into that directory.
```

```
# Change Directory
```

```
cd cs360_test
```

```
# Confirm your new current working directory
```

```
# If you ever need an explanation of a command, use the man command
```

```
# this will explain the command and list all the options available.
```

```
man ls
```

```
(then press enter at the question, q to quit)
```

```
# Let's copy some files from zeus to the local machine for the
```

```
# rest of the exercise.
```

```
# wget will retrieve a file from a web server.
```

```
wget zeus.cs.pacificu.edu/chadd/LinuxTest.tar.gz
```

```
# The fetch command is similar to wget. The curl command is
```

```
# also similar. Not all systems have all three commands.
```

```
# Copy this LinuxTest.tar.gz file to your home directory on zeus.
```

```
# we will use Secure Copy (scp) to do this.
```

```
# Note the : at the end!
```

```
scp LinuxTest.tar.gz punetid@zeus.cs.pacificu.edu:
```

```
# Connect to zeus to make sure the file transferred up correctly
```

```
ssh punetid@zeus.cs.pacificu.edu
```

```
# Make sure your prompt says zeus, otherwise you are still on your
```

```
# local machine.
```

```
# Confirm the file exists on Zeus
```

```
# The who command will show you who else is logged on to this  
# machine.
```

```
who
```

```
# Use the exit command to disconnect from zeus.
```

```
exit
```

```
# Make sure your prompt says the name of your local machine and  
# not zeus!
```

```
# tar.gz files are compressed archives.
```

```
# Uncompress and extract the files:
```

```
# tar: the command
```

```
# x: extract
```

```
# v: verbose - list to the screen the files extracted
```

```
# z: compressed (the .gz file extension requires this)
```

```
# f: file (the filename must follow directly after the f option)
```

```
tar xvzf LinuxTest.tar.gz
```

```
# Change directory to the newly created LinuxTest directory
```

```
# List all the files in that directory
```

```
# A number of text files ( .txt) should be listed.
```

```
# Let's look at one of them
```

```
# cat displays the contents of a text file to the screen
```

```
cat CS150.txt
```

```
# If you just want to see the first two lines in a file:
```

```
head -n 2 CS150.txt
```

If you just want to see the last two lines in a file:

tail -n 2 CS150.txt

A good way to edit the file is to use nano.

The menu is at the bottom of the screen. ^ means the control key.

Control-X edits nano

nano CS150.txt

pico is similar to nano.

vi is more powerful than pico or nano, but takes a while to learn.

google vi cheat sheet if you are interested in vi.

There are many words in that file. Let's count how many

lines, words, and characters are in that file. The wc command

will do that.

wc CS150.txt

How many lines are in the file?

How many characters are in the file?

You can also print just the number of lines, words, or characters

wc -l

wc -w

wc -m

What does the following do

wc *

```
# Let's check how many files contain the word data.
```

```
grep data *
```

```
# Grep explained:  
# grep : the command  
# data : the pattern to look for  
# * : the files to check (* is a wildcard meaning every file)
```

```
# Let's find all the text files (.txt) that exist in the  
# LinuxTest directory and its subdirectories
```

```
find . -name '*.txt'
```

```
# Find explained:  
# find : the command  
# . : Start looking in the current directory.  
# -name '*.txt' : look for file names that match the pattern *.txt  
# the single quotes are important!
```

```
# Let's save that list of text files into a new file:  
# the > will take the output of the command to the left, and write  
# that output  
# to the file on the right.
```

```
find . -name '*.txt' > listOfFiles.txt
```

```
# Let's make sure that new file got created
```

```
# Show the contents of the new file listOfFiles.txt
```

```
# Let's find all the text files again
```

```
# Notice that listOfFiles.txt is now listed.
```

```
# Let's save the list of text files to a new file, named  
# newListOfFiles.txt
```

```
# Let's see how the two list of text files differ.  
# The diff command compares two text files and displays the  
# difference
```

```
diff listOfFiles.txt newListOfFiles.txt
```

```
# The output contains an > which means that line exists in the  
# file listed on the left, newListOfFiles.txt, and not in the file  
# listed on the right, listOfFiles.txt.
```

```
# Let's count how many files we found:  
# The pipe, or vertical bar, takes the output from the command on the  
# left and uses that output as input to the command on the right.
```

```
find . -name '*.txt' | wc -l
```

```
# Let's list information about each of the text files:  
# the xargs command will take each of the lines from the  
# input (the command on the left) and pass that line  
# to the command on the right (in this case, ls -l)  
find . -name '*.txt' | xargs ls -l
```

```
# Let's move into the documents directory to work with those files
```

```
# Display the contents of the file preamble.txt
```

```
# Display the contents of the file declaration.txt
```

```
# This file declaration.txt is really long and scrolls by too fast.
# Let's slowly scroll through the file.
# the less command shows you a screen full of text at a time
# press space for the next screen. press enter for the next line.
# press q to quit
```

```
cat declaration.txt | less
```

```
# The more command is similar
```

```
cat declaration.txt | more
```

```
# You don't need to use cat
```

```
more declaration.txt
```

```
less declaration.txt
```

```
# Open up another terminal on the other monitor
```

```
# Run top to show all the processes currently running
```

```
top
```

```
# restrict top to show only your processes by typing u in the window
```

```
# displaying top and typing your punetid
```

```
# Press q in the window displaying top to quit top. Do not close the
```

```
# terminal.
```

```
# Type ps to see all the processes running in your current window
```

```
ps
```

```
# The sleep command will sleep for some number of seconds
```

```
sleep 3
```

```
# Putting an & at the end of a command will run the command in the
```

```
# background and allow you to type more commands.
```

```
sleep 120 &
```

```
# Check all the running processes again, you should see sleep running
```

```
# (or sleeping)
```

Open top again in the other terminal

restrict top to show only your processes

Go back to the original terminal and launch sleep again with the &
sleep 600 &

Look at top. The PID or process id is listed at the left. Sleep
should be listed.

Find the process id.

Go back to the other terminal and kill the sleep process
kill PID

Some processes don't die when you use kill alone. Sometimes you
need to do the following. 9 stands for SIGKILL:

kill -9 PID

Check out the following man page.

You are looking at the definition of kill from section 1p.

SIGKILL is listed around line 70.

man -s 1p kill

A few more commands.

Let's check how much space is left on the hard drive

You should be able to see how much space is available on /home

df -h

Another useful command is screen.

Let's first check to make sure screen is available in the lab.

The **which** command will tell you where the command is on the

file system, if it exists.

which screen

which bob # should fail and tell you in which directories which
looked for bob


```
# Screen exists, good.
# screen is especially useful if you
# need to run a long running command on a remote machine, such as
# zeus. You would log into zeus and then start screen on zeus.
screen -S test

# This creates a shell session that won't die if you get
# disconnected. Also, you can reconnect to this shell session from
# another location to check
# the progress of your work or make changes.

# In a new terminal, connect to the screen you just created:
screen -x test
ls
ls -al

# you should see the same output being shown in both terminals.
# everything is happening exactly once, the output is just
# copied to both places.

# Let's disconnect from screen in one terminal.
Control-A D

# In the other terminal, let's write a little loop
# The bash shell is powerful, you can program right in the shell!
# The following is a while loop that will sleep for 5 seconds,
# display information about the hard drive, and then display the
# date.
# The loop is equivalent to while(true) in C++ so it will
# never terminate. Later, we will use Control-C to kill the loop.

while [ true ] ; do sleep 3; df -h ; date ; done

# Let's disconnect from screen in this terminal.
Control-A D

# Take a deep breath.

# Let's reconnect:
screen -x test
```

Your previous output should show up and you should see that the
loop is still running.

Press **control-C** to kill the loop

Typing **exit** will terminate the screen.