Connecting to ICS Server, Shell, Vim
CS238P – Operating Systems
fall ‘18

By
Aftab Hussain

(Adapted from Claudio A. Parra’s Slides for Fall ‘18 CS-143A)

October 5 2018
University of California, Irvine
Andromeda

- Server with Linux managed by the ICS.
- You must have an “@ics.uci.edu” account.
- Your server number is:
  \[
  \text{ser\_num} = (\text{<ucinetid>}\%75)+1
  \]
- The server you will connect is:
  \[
  \text{andromeda-<ser\_num>.ics.uci.edu}
  \]
How to connect to it

We need a secure shell client (ssh).
• In Linux: integrated.
• In Windows: integrated (win10), or Putty.
• In Mac-OS: integrated.

ssh username@andromeda-XX.ics.uci.edu
Welcome to Linux, now what?

A couple of notes:

- `/` is the root directory, everything is under it.
- `~` is your home directory, it is an alias for `/home/yourUsername/`

https://en.wikipedia.org/wiki/Home_directory
Welcome to Linux, now what?

A couple of notes:

- `/` is the root directory, everything is under it.
- `~` is your home directory, it is an alias for `/home/yourUsername/`
- `./` is the current directory.
- `../` is the parent directory.
- Passwords are invisible.
- Case matters, “A” and “a” are different.
Some commands

```
ls [options] [dir]
```

- List information about the FILEs (the current directory by default).
- `-R`: recursive.
- `-l`: long format, shows info of each file
- `-a`: show all files, including hidden files, those that start with a “.”
- `-h`: file sizes in a nice way
Some commands

```
cd [dir]
```
Change the shell working directory.

- If no directory is given, it changes to the home directory.
Some commands

```
pwd
```
Print the name of the current working directory.
Some commands

```plaintext
mkdir <dir_name>
```
Create the directory `<dir_name>`, if it does not already exist.
Some commands

```
touch -c <filename>
```

Creates a new empty file.
Some commands

`cp <source> <dest>`

Copy the source file to the destination.

• Example:

```
cp myFile.txt ./aDirectory/newFile.txt
```
Some commands

```
mv <source> <dest>
```

Move or rename the source file to the destination file.

- Example moving:
  
```sh
mv myFile.txt ./aDirectory/newFile.txt
```

- Example renaming:
  
```sh
mv myFile.txt newName.txt
```
Some commands

`cat [filename]`

Print the content of a file to standard output. If no file is given, then prints what comes from standard input (most of the cases the keyboard)
Some commands

`head <filename>`

Print the first 10 lines of the filename to standard output.

- `-n X`: prints the first X lines
Some commands

tail <op> <filename>

Print the last 10 lines of the filename to standard output.

- \texttt{-n X}: prints the last X lines
Copying things

```
scp <source> <destination>::<dir>
```

Copy files from one machine to another through ssh. Easier if you run it in your local machine

Example local to remote:
```
scp ~/localFile peter@andromeda-XX.ics.uci.edu:~/remoteFile
```

Example remote to local:
```
scp peter@andromeda-XX.ics.uci.edu:~/remoteFile ~/localFile
```
Some commands

```
grep <pattern> [file]
```

Search for `<pattern>` in the given file.

- If no file is given, and you pass `-r` as option, `grep` searches recursively in the working directory.
- Using `-e <pattern>`, `grep` interprets the pattern as an extended regular expression.
Some commands

```
find -name <filename>
<dir>
```

Find a file named `<filename>` in the directory `<dir>`

- There is a lot of other options.
Other useful tools

“>” and “<”

Redirect the standard output or input.

• Example: `ls -l > myContent.txt`
• Example: `cat < myContent.txt`

More on input redirection
https://askubuntu.com/questions/883786/how-does-input-redirection-work
Other useful tools

“>>”
Append.

• Example:

  echo "Dear diary" > log.txt
  echo "Today I..." >> log.txt
Other useful tools

Pipe, passes the output of a command to another as input.

• Example:

  `ls -R | grep "myFile"`
Some more commands

`top`
- display Linux processes

`htop`
- same as above but interactive
Some more commands

ps

display snapshot of current processes
Some more commands

```
kill <pid>
```

Kills a process
Some commands

```
rm [op] <filename>
<dir>
```

delete a file in the specified directory. If no directory is given, uses the current one.

- `-r`: recursive.
- `-d`: remove empty directories.
- `-i`: interactive, ask before each file.

DO NOT TRY THIS: `rm -rd ./`
Editors

**vim**

**NORMAL MODE**
- G go to end of file
- nG go to line number n
- v visual select
- Ctrl+O or I forward/backward
- y copy
- p paste
- u undo
- Ctrl+r redo
- dd cut line
- /<pattern> search pattern
- i go to insert mode
- : go to command mode

**COMMAND MODE**
- w write (save)
- wq save and quit
- q! quit without saving changes
- f show filename
- split horizontal split
- vsplit vertical split
- Ctrl+W <arrow> navigate windows
- copen quickfix tool
- grep <pattern> *.<filetype> clear all splitting