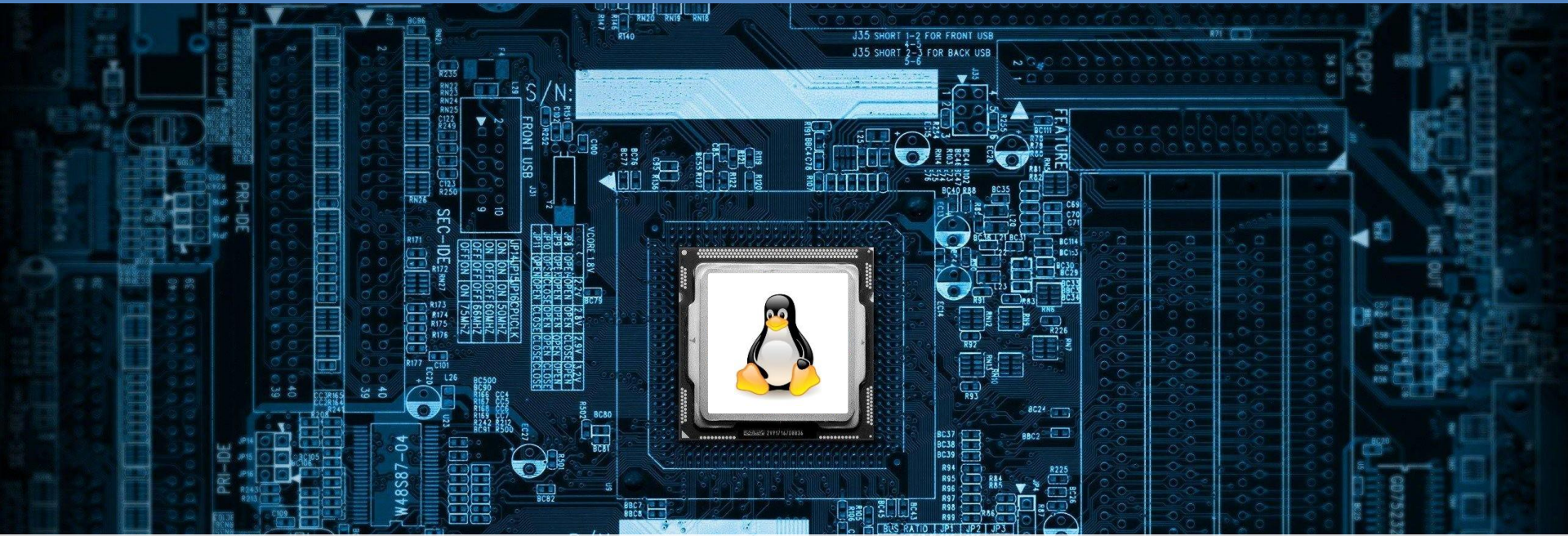

ECE 650 – LINUX & GIT TUTORIAL

Jeff Luo

PhD Student working with Prof. Dietl

Connecting to ecelinux from home

- Open up an SSH client, eg PuTTY (http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html)
- Connect to `ecelinux4.uwaterloo.ca` with your Quest username and password
- Once connected, type `ssh -X eceLinuxN` to connect to one of the ecelinux work machines to do work (N = 1 to 11)

- If at any point it asks you something about RSA key fingerprints or authenticity, just accept them.

Basic Linux commands

Command	What it does
`pwd`	prints out the path to the current working directory you are in
`cd` *folder*	will change the current working directory to *folder*
`ls`	lists all the folders and files in your current working directory
`ls -la`	also lists all hidden files and directories
`ls` *folder*	same command but will list the contents of *folder*
`mkdir` *foldername*	creates a folder called *foldername* in your current directory
`rmdir` *foldername*	deletes the folder called *foldername*
`touch` *filename*	creates a blank file called *filename*
`rm` *filename*	removes the file called *filename*
`rm -r` *folder*	deletes *folder*, and all files and subfolders within *folder*

Basic Linux commands

Command	What it does
`man` *`program`*	display the help page for a particular command or program
`more` *`filename`*	display the contents of the file called *filename*, with controls to advance the contents page by page
`less` *`filename`*	display the last page of content of the file called *filename*, useful for inspecting log files for the most recent events

There’s many more Linux commands, look them up on Google

File Editing

- There are various ways to create and edit files in Linux, but most commonly these will be text files
- Most popular command line text editors: `emacs`, `vi`, and `vim`
- Most popular graphical text editors: `gedit`
  - In `ecelinux` you won't be able to get a graphical user interface running, in your own Linux computers you can use graphical text editor tools

Vim (Vi Improved)

- Has 4 modes, normal, insert, visual, and ex mode, which you must switch between
  - Vim always launches in normal mode (for viewing)
- Basic tutorial: http://vim.wikia.com/wiki/Tutorial
- Advanced commands (search & replace, etc):
  - https://danielmiessler.com/study/vim/
  - http://derekwyatt.org/vim/tutorials/novice/
- To edit a file: type `vim filename` in command line to launch vim with that file opened
  - Press i to enter insert mode, start editing
- To save the file, from insert mode, press esc, then type :w and press enter
  - Press i again to go back to insert mode
- To quit the program, from insert mode press esc then type :q
  - Together: :wq will save and quit

Linux Shortcuts and Environment Variables

- ~ is a shortcut to your home directory (eg `cd ~`), on `ecelinux` this is `/home/`*`yourUWID`*
- You can create your own custom shortcuts (called environment variables) by editing the file `~/.bash_profile`
- Eg add this to `.bash_profile` then save:
  - `export ECE650=~/myCourses/ECE/650`
  - `export A1=$ECE650/assignment1`
- Of course, you'll also have to create these folders using `mkdir`
- Run command `source .bash_profile` to add the environment variables to memory
- To see that it is in memory, use echo command: `echo $ECE650`
- You can then use `cd $A1` to jump right to `/home/uwID/myCourses/ECE/650/assignment1` folder

Source Code Version Control Systems

- Software systems to help you manage and share source code and projects
  - Versions and version history
  - Team editing
- Two most popular systems: `Git`, `Hg`
- Two most popular hosting sites: Github.com (works with Git only), Bitbucket.org (works with both Git and Hg)
  - Create an account and a git project on one of these!
- `ecelinux` only has `Git` installed

Git System

- Keeps track of incremental updates to your source code and project via a series of **Commits**
  - repeat(make some file edits and changes, commit)
  - You can backtrack your commits to go to an earlier version of your project, and go forward again as well
- Keeps track of development work in parallel across team members using **Branches**
  - From some commit, we **Fork** into two branches, each branch then advances with its own series of commits, and some time later we **Merge** the branches
  - Each branch typically owned and worked on by 1 person
  - Merge will combine the work across team members

Basic Git project management

- To clone a project (including your own) from the web, use `git clone` command:
  - `git clone https://github.com/youraccount/yourproject.git`
- Committing:
  - Add all relevant files you want to the commit using `git add`
    - `git add /path/to/filename`
  - Then create a commit:
    - `git commit`
    - It will prompt you with a screen asking you to enter a commit message, the message describes the overall changes you did in this commit (eg added display functionality, changed x, removed y…)
  - To upload the commit to the server, use `git push`
- Full tutorial: http://rogerdudler.github.io/git-guide/