Research Computing

Philip Blakely

Laboratory for Scientific Computing, University of Cambridge

Part I

Linux networking

Secure connections

- Assuming you have a Linux/Unix-like terminal (WSL/Mac OS X), then:
- To log in to another computer over the Internet, use:

  ```
  $ ssh pmb39@apollo.lsc.phy.private.cam.ac.uk
  pmb39@apollo's password:  [Enter password.]
  pmb39@apollo $
  ```

  and you will get a terminal that is running on the remote machine (assuming you have appropriate permissions).
- Note your username may not be the same on the remote machine as on your local machine (hence specifying `pmb39`).
- Here you can run any programs that are installed on that computer.
- Any files you create will be on that computer only (unless any directories are somehow shared with others).

Password

- When logging into a new system, you should always change your initial password.
- (It might have been stored/intercepted by a lazy/nefarious sys-admin.)
- To change your password:

  ```
  $ passwd
  Enter login(LDAP) password:
  New password:
  Re-enter new password:
  ```

- Make sure you use a strong password ($\geqslant 12$ characters, upper-case, lower-case, numbers, punctuation). The sys-admin may enforce some constraints automatically.
- Either memorise the password or store it in your password-manager.

Running multiple remote terminals

- What happens if your ssh-connection goes down (wireless disconnects, you close your laptop)?
- The answer is a multiplexer such as `screen`, `tmux`, or `byobu`.
- You can open multiple terminals on a remote machine (e.g. one to compile, one to edit source-code, one to run simulations)
- This turns one remote terminal (over one ssh connection) into multiple that you can switch between with keyboard shortcuts.
- Helpfully, the multiplexer continues running even if the `ssh` connection fails, and you can reconnect to the session later.
- I use this to keep work open between working in the office versus working from home.

Running multiple remote terminals

To start a multiplexing session: `screen`

Then `screen` commands:

- Ctrl-a Ctrl-c: Create a new terminal.
- Ctrl-a 0: Switch to terminal 0 (or 1-9).
- Ctrl-a Ctrl-d: Detach from the screen session
- Ctrl-a k: Kill current terminal
- Ctrl-a ?: Screen Help

The shortcuts for `tmux` are more-or-less the same, except with Ctrl-b instead of Ctrl-a.

Running multiple remote terminals ctd

To reconnect to an existing screen when you've logged back in:

```
 $ screen -list

There are screens on:

1552000.Lecture_Slides (24/08/23 08:31:17) (Attached)
57458.Assignments_Lectures (11/08/23 15:42:13) (Attached)
54334.AMReX-work (11/08/23 15:40:28) (Attached)

3 Sockets in /run/screen/S-pmb39.

$ screen -x -r 1552000
```

Or omit the screen ID if there's only one running.

X-forwarding

- If you try to run a GUI program via ssh:

```
pmb39@h2g2 $ ssh pmb39@apollo
pmb39@apollo $ gedit
Cannot open display:
pmb39@apollo $
```

- By default `ssh` only opens a text-terminal. It does not forward GUI information from the remote machine. Instead, use:

```
ssh -X pmb39@apollo
```

- However, the GUI of remotely running programs will run slowly as all the pixel-information must be transferred to your screen across the Internet.
- Remote Desktops are an alternative, but outside the scope of this lecture.

Not X-forwarding

- Avoid needing to use `ssh -X` by learning how to use command-line based programs such as `emacs/vim` or ones that run in client-server mode such as VSCode.
- `ssh -X` may need more work if you're using Windows or Mac.
- For Windows: MobaXTerm `https://mobaxterm.mobatek.net/`

Secure copy

- The `cp` command only works for files directly accessible on the same computer, i.e. anywhere you can `cd` to in one terminal.
- To copy files from/to remote machines, use:

  ```
  pmblakely@h2g2 $ scp ./settingsFile pmb39@apollo:~/
  pmblakely@h2g2 $ scp -r -C pmb39@apollo:~/outputData ./
  ```

  to copy a settings file across, and then recursively (`-r`) with compression (`-C`) copy the output directory back.
- One of the computers involved in the copy must be the one you are currently logged into (or you need the `-3` option).
- It is usually better to run `scp` on your local machine, but make sure you get the source (first) and destination (second) files the right way round.
- It is unlikely that you can ssh from `apollo` to your laptop, so `scp` must be run on your laptop.

Synchronizing folders

- If your connection is liable to drop out while copying, or you need to incrementally re-synchronize a particular folder between two machines, use `rsync`.
- This uses heuristics (file-size, modification-time) to see if files have changed since last run and only transfers modified files.
- `rsync -a ./MyCode/ pmb39@apollo:~/MyCode/`

  will update files in `MyCode` on `apollo` to match those on my local machine.
- `-a` means "archive" mode, which is probably what you want for synchronizing directories. See `man rsync` for many more options.
- Make sure your folder-names end with `/` If not, `rsync` will probably not do what you want (e.g. create a new folder inside the one you want to transfer to).
- A better way to synchronize code across computers is to use `git`.

ssh keys

- You can set up ssh-keys to make logging into machines more secure.

```
$ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key
(/home/pmblakely/.ssh/id_rsa):
Enter passphrase:   My Secret Phase
Enter same passphrase again:   My Secret Phase
Your identification has been saved in
/home/pmblakely/.ssh/id_rsa
Your public key has been saved in
/home/pmblakely/.ssh/id_rsa.pub
```

- This stores a private/public key-pair (RSA algorithm) in two files.

ssh keys ctd

- You can now copy the (public) key to another machine:
  `ssh-copy-id pmb39@apollo`
- When you attempt to `ssh` to that machine again, you will be logged in automatically as the remote machine checks that you have the private key corresponding to the public key on that machine.
- If the private key (on your local machine) is compromised (e.g. your laptop is stolen), the attacker then has access to the remote machine. This is why we used a passphrase to encrypt the ssh-key.
- The commands `scp` and `rsync` use `ssh` to make the connection and transfer data.
- Other programs may also use `ssh` to connect to remote machines, e.g. VSCode, VisIt, etc.

Compression

- If transferring large amounts of data, even a 1GiB connection may not be fast enough, so you need to compress it.
- You can use `tar` (short for tape-archive) as:

  ```
  tar -cjf MyCode.tar.bz2 ./AllMyCode ./AllMySettings
  ```

  to create (`-c`) a bzipped (`-j`) tar-file called (`-f`) MyCode.tar.bz2 from the folders `AllMyCode` and `AllMySettings`.
- Alternative compression formats include `gzip`:

  ```
  tar -czf MyCode.tar.gz ./AllMyCode
  ```
- (bzip usually produces a smaller file)

Compression

- To decompress (eXtract) files:
  ```
  tar -xf MultiPhysics_Source_Code.tar.gz
  ```
- To list what is in the file first:
  ```
  tar -tf MultiPhysics_Source_Code.tar.gz
  ```
- You can pass `-j` or `-z` explicitly, but `tar` can probably work it out by itself.
- Alternatively `zip` and `unzip` handle `.zip` files (originally from PKWARE, most common on Windows):
  ```
  zip -r MyCode.zip ./MyCode/
  unzip MyCode.zip
  ```

Stopping and pausing programs

- If you want to kill a program running in a terminal, press Ctrl-C.
- This will not work if the programmer has disabled it (fairly unlikely).
- To pause a running program, press Ctrl-Z.
- To allow a paused to continue to run in the background, while you continue to use the terminal, type `bg`.
- If you need to put a program in the foreground again, type `fg`.
- To run a program and immediately background it, run it as:
  ```
  xclock &
  ```
- To ensure a program does not stop when your ssh-connection ends:
  ```
  nohup myLongCode &
  ```
  which stops the program from responding to the Hang-UP (HUP) signal.

Machine characteristics

- If the sys-admin has not documented their network properly, you can still find out what hardware machines have.
- `cat /proc/cpuinfo` identifies the CPU(s) in the computer.
- `free -m` gives the free memory (RAM) in units of MB.
- `df -h` gives the disk-space available on physical disks.
- Remember that RAM (Random Access Memory) is the temporary storage for currently running programs and their data. The disk (HDD or SSD) is permanent storage.
- Also try `lshw` for more hardware characteristics.
- If it has an NVIDIA GPU, `nvidia-smi` gives GPU-usage details.

CSC Network Specifics

Different Linux networks have their own ways of organising things, and the following are specific to the CSC Network.

More information can be found at `www-internal.lsc.phy.cam.ac.uk`

Please read this before coming to ask for help.

Home directories

- Your home directory is of the form `/home/raid/pmb39`.
- It is shared across all CSC machines.
- You initially have 2GB of disk-space, and it is backed up daily.
- This should be used for code, settings files, your projects, and similar things that would cause major problems if lost.
- It should not be used for large amounts of data output that can easily be regenerated.
- Large amounts of data can be stored in the `/local/data/public` folders on all computers.
- These are mounted on all other machines as `/data/apollo` for example.
- Use `df -h` to check disk usage, or the `quota-local` command.

CSC computers

- `http://www-internal.lsc.phy.cam.ac.uk/systems.shtml` for a list of computers
- This includes details such as RAM, disk-space, Processor, etc.
- If you need to find a free computer, use `http://www-internal.lsc.phy.cam.ac.uk/mrtg` to see current processor usage.
- You can also use `top` and `htop` to see what processes are running on a machine.
- To see who is logged in, use `who` or `last`.
- There is no job-queueing or enforced-limiting on these machines, so be careful you don't hog/overuse the machines.
- For long running jobs, you can reduce their priority by prefixing the command with `nice -19`.

CSC desktops

- As well as the servers, most of the desktops you see in the CSC areas of Maxwell are on the CSC network.
- You can log in to these with the same CSC password.
- Do not disconnect the desktops from the wired network, or switch them off.
- Someone may be running a simulation on them and be slightly annoyed...

VPNs

- You may have noticed that the full-name of `apollo` contains `private`. In Cambridge, this means it is not visible from outside the Cambridge network (CUDN).
- If you are outside the CUDN (e.g. in private accommodation), you need to use a VPN to reach CSC machines: `https://help.uis.cam.ac.uk/service/network-services/remote-access/uis-vpn`
- For anyone with a CSC laptop, run: `setup_vpn` to set up a VPN connection.
- Other departments/groups/systems (e.g. DAMTP, Physics TCM, CSD3) may be accessible without a VPN, or have a world-accessible `ssh`-gateway you can hop through.

Laptops

- If you want your own laptop to connect to the wired network, please contact `it.helpdesk@phy.cam.ac.uk`. They will need the ethernet MAC address from it, as well as the laptop make/model, and operating system.
- Use the `ifconfig` command to find the MAC address.
- You can just use WiFi (eduroam or UniOfCam) but that will be slower.
- If you are using a laptop, the data on it is not automatically backed-up. Either back it up yourself (e.g. to Google-Drive, Microsoft One Drive) perhaps using `rclone`, or connect to your CSC home directory using SAMBA.
- Details of how to connect are at

`http://www-internal.lsc.phy.cam.ac.uk/network_files.shtml`

Printing in CSC

- Use PaperCut Print Deploy Client (probably on the Acessories menu of a CSC laptop/desktop).
- Log in with your UIS/Raven credentials.
- Install the Maxwell_FindMe printer.
- Print from your favourite application to `Maxwell_FindMe`.
- At a Maxwell printer, present your University card, and you can release your print-jobs.
- Or, use `https://managedprint.uis.private.cam.ac.uk`

Installing new software

- If the software you need is not available already, you can:
  - Ask for it to be installed as an Ubuntu package.
  - Ask for it to be compiled from source.
  - Compile and install it yourself in `/local/data/public/pmb39/bin` for example.
- The same applies if you need a newer version than is available by default.
- Some extra software is in `/lsc/opt` already, such as newer versions of `gcc`, CUDA compilers, Intel compilers, and visualisation software VisIt.