Hitchhiker's Guide
to
Linux

David Money Harris
Revised 2 February 2020

The VLSI design tools at Harvey Mudd College are hosted on a Linux server named *tera*. This document introduces you to connecting with the server, getting around Linux, and starting the tools.

Connecting to Tera

The usual way to connect to tera is from your Windows, Mac, or Linux-based computer using the X11 display protocol.
From Windows, follow the instructions in *X Forwarding on Windows.* (The required programs are already installed (but not configured) on the ECF computers).

MobaXTerm (https://mobaxterm.mobatek.net/) is a useful collection of Windows utilities, including X forwarding, remote file access, and an SSH client.

If you have a Mac, install XQuartz, a Mac implementation of the X Window System, used for X11 forwarding (https://www.xquartz.org). After install, you'll need to log out and back in to your Mac.

If you're using Linux, the X Window System is most likely built in. If not, you likely possess the debugging skills to start X forwarding yourself (or to install an X server).

If you are on Windows, start Xming and then start Git Bash (or another bash shell). If you are on MacOS, start XQuartz and open a new terminal. On Linux, just open a new terminal.

IMPORTANT: If you are using Git Bash for the first time on your personal computer or on a lab computer, it may be necessary to follow the directions in the Configuring your Bash shell section *X Forwarding on Windows.*

At the terminal, type `ssh -Y -C username@tera.eng.hmc.edu`, where you supply your username (usually first initial and last name, such as bkeller for Ben Keller). Tera will prompt you for your password.

Change your password the first time you log in, by typing `passwd` and following the prompt.

You will need some files for the various labs. They are located in `/courses/cmosvlsi/20`

Getting around Linux

Tera runs Linux CentOS 6.6, which is the old, stable, and preferred release for the CAD tools. If you aren't already familiar with Linux, follow this tutorial to learn your away around.

You can create, view, and manipulate files by typing commands into the terminal. Type

```
ls
```

to **li**s**t** the contents of your directory. The directory will initially be empty. To create a new subdirectory, type

```
mkdir IC_CAD
```

when you ls again, you'll see

```
IC_CAD
```

To create a new file, type

```
gedit foo
```

The GNOME text editor opens up. Type "Hello" into the text editor, then save and quit. Now,

Typing

```
ls -l
```

gives you a longer listing showing more information about the files:

```
-rw-rw----  1 bkeller users    6 Jan 17 12:54  foo drwxr-
xr-x  2 bkeller users 4096 Jan 17 12:44  IC_CAD
```

The owner is **bkeller**, who is part of the group **users** (you should see your user name instead) If a d appears instead of the first `-`, it indicates that a file is a directory. It is followed by three triples of letters. The first wxr means that the user can **w**rite, **r**ead, and e**x**ecute the file (for directories, execute means to enter the directory). The second means that members of the same group can read and execute but not write the file. The third means that others not in the same group can also read and execute but not write. The file size is 4096 bytes, which is standard for a directory. The IC_CAD directory was created on January 17 of this year at 12:44. foo is not a directory and is only 6 bytes long (five characters plus the carriage return at the end). It is readable and writable only by bkeller and other group members. In Linux, file names beginning with a dot are normally not displayed when you type ls. Usually, configuration files are named this way, so they don't clutter things up. To display everything including the dot files, type

```
ls -al
```

Some dot files have already been created in your directory. For example, .bash_profile is a configuration file invoked on log-in that sets up the paths to find various programs.

To **p**rint your current **w**orking **d**irectory, type

```
pwd
```

and you'll see

```
/home/bkeller
```

To **c**hange **d**irectory into the new directory you created, type

```
cd IC_CAD
```

Now if you type `pwd`, you'll see

```
/home/bkeller/IC_CAD
```

To go back up to your home directory (`/home/bkeller`), type

```
cd ..
```

Where `..` indicates the directory above where you currently are. (Note that . refers to the directory where you currently are.) Alternatively, you could have typed

```
cd /home/bkeller
```

or

```
cd /home
cd bkeller
```

or

```
cd ~bkeller
```

where `~bkeller` is a shortcut for the home directory of that user (i.e. `/home/bkeller`)

or just

```
cd
```

because `cd` returns you to your home directory from wherever you are.

To view the contents of a text file, you can open it in an editor such as gedit. However, for short files, it is often more convenient to use the `more` command.

```
more foo
```

To **m**o**v**e or rename a file, type

```
mv foo foobar
```

To **c**o**p**y a file, type

```
cp foobar fooseball
```

To **r**e**m**ove a file, type

```
rm foobar
```

Then type `ls` to see

```
charlie fooseball IC_CAD
```

Linux supports symbolic **l**i**n**k**s**, in which one file is really just a pointer to a second. If you change the second file, the first will change. For example

```
ln –s fooseball foolink
```

Then when you ls –l, you'll see what foolink links to. If you were to use gedit to change fooseball to say "Goodbye", then typed more foolink, you'd see Goodbye. Thus, symbolic links are better than copying when you want two files to remain the same.

To **ch**ange the permissions on a file, use `chmod`. `chmod` takes three octal numbers, indicating the permissions for the user, the group, and for others. Each number should be the sum of three bit fields: 4 for read, 2 for write, and 1 for execute. For example,

```
chmod 644 fooseball
```

changes the file to be readable and writable by the user and read-only for everyone else. Then

you could move `fooseball` into `IC_CAD` and rename it back to `foo`

```
mv fooseball IC_CAD/foo
```

To get rid of the entire `IC_CAD` folder and its contents, use the recursive option on `rm`:

```
rm –r IC_CAD
```

Use this command with care! Open `gedit` again by typing

```
gedit
```

While it is running, the terminal is tied up and can't be used. To fix this, press ctrl-Z in the terminal to put `gedit` to sleep. Now `gedit` becomes unresponsive. Then type

```
bg
```

in the terminal to run gedit in the **b**ack**g**round. Now, you can use both gedit and the terminal. To find out what **p**rocesse**s** you have running, type

```
ps
```

And you'll see something like

```
    PID TTY          TIME CMD
29382 pts/1    00:00:00 bash
29905 pts/1    00:00:00 gedit
29949 pts/1    00:00:00 ps
```

bash is the "**B**ourne-Again **S**hell," part of the terminal that interprets the commands you have been typing. Gedit is running in the background. ps is the command you just invoked. Notice that each command has a process ID (PID). If a program locks up, you can kill it based on its PID. For example

```
kill 29905
```

kills the gedit session. You would lose any unsaved work, so don't use this unless you have to.

Instead of putting gedit to sleep and restarting it in the background with separate commands, you could have typed

```
gedit &
```

to invoke it in the background in the first place.

Transferring Files to and from Tera

There are many tools to transfer files between servers, or between your computer and Tera. Two common tools are SCP and SMB. These utilities are widely documented online, but will be described briefly below. With these utilities, you can copy files between the systems. This is particularly helpful to copy over images from tera for a report that you are writing on your PC.

SCP, Secure File Copy, is a utility that allows you to transfer files and directories over SSH. To send a file from the system you are logged in on to another system, use the command

```
scp file host:path
```

Or to transfer a directory, use the command:
`scp -r directory host:path`

For example, you can transfer some file data.txt to /proj on Tera with the command:

```
scp data.txt <username>@tera.eng.hmc.edu:/proj
```

To transfer a file or directory from a remote system to the system you are logged in on, you can switch the order of the arguments to scp. For example, to transfer data.txt from Tera to your current directory, you can use:

```
scp <username>@tera.eng.hmc.edu:/proj/data.txt .
```

SMB, a networking suite known on Windows as Samba, allows you access your home directory on Tera as a local directory. On a Windows PC, you can access a Samba share through File Explorer → Add a Network Location. On MacOS, you can access a Samba share through Finder → Go → Connect to Server. Due to the transient nature of the Harvey Mudd network architecture, you may need to connect to the Claremont Colleges VPN before you are able to connect over SMB.

Another way to move files between your computer and tera is through a **f**ile **t**ransport **p**rotocol (FPT) client. WinSCP and FileZilla are good FPT programs for Windows. To get more information about a command, use `man`:

```
man rm
```

SSH Keys

If you plan on connecting to Tera from the same computer multiple time, you can setup ssh keys to make it remember your computer and not require a password.

To do this, first generate a ssh keypair from your computer (if you haven't already) by typing
`ssh-keygen`
and accepting all the default options (this is cryptographically secure without an additional password)

Then, install them on tera by typing
`ssh-copy-id <username>@tera.eng.hmc.edu`
Hopefully you should now be able to ssh to Tera from your computer without typing your password!

Moving to the Next Level

The commands so far are adequate to get you started. However, Linux is full of other commands that you will find useful as you become more sophisticated and work with a greater assortment of files.

`!!`: repeat the last command
`!word`: repeat the last command that started with word

`*`: matches any file name
Ex: `rm *.txt` removes all files ending with the suffix .txt (dangerous!)

`grep`: global regular expression print

Ex: `grep Hello *` prints the lines of all files that contain the word “Hello”

`|`: pipe the output of one command to another
Ex: `grep Hello * | more` displays the results of the grep with more

Other commands

`okular <file>`	read a PDF or PS file
`ps2pdf`	convert a Postscript (.ps or .eps) file to PDF
`xterm &`	create another terminal window
`exit`	closes a terminal
`top`	display a list of the processes using the most resources

A longer list of commands is located at http://www.ss64.com/bash/