

# CG2271 Real Time Operating Systems

## Lab 6 Supplement - Introduction to UNIX

### 1. SoC Account

First and foremost, you'll need an SoC Account (previously known as SoC UNIX accounts) in order to access the `sunfire` UNIX server. Most of you should already have one. In case you don't, here are the steps to create one:

- a) Logon to this website using your NUSNET id and password:  
<https://mysoc.nus.edu.sg/~newacct>
- b) Read through the user-agreement and make sure you understand the obligations.
- c) If applicable, assign your own username. Your username should be between 5-8 characters and must be formed from your name. You may also use your NUSNETid.  
[Note : Your username, once chosen, shall be permanent. You will not be allowed to change it under any circumstances whatsoever]
- d) Type in your new password (twice).
- e) Submit your application.
- f) If the server returns with the success screen, your application is successful. Your account will be available typically within a few seconds but on busy days it may take a few minutes.

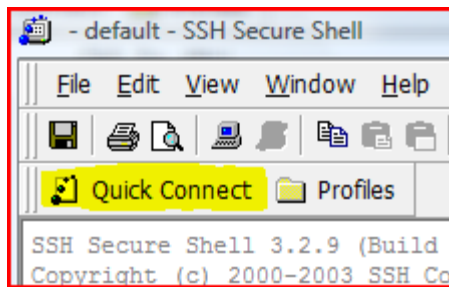
### 2. Logging in to `sunfire` :

The SoC Account is what is used to log into `sunfire`, or other UNIX machines in SoC. One way of accessing `sunfire` is to start a remote shell (also known as command prompt in Windows) where you can perform tasks and execute programs on `sunfire` without having to be physically at the machine. Almost all systems in SoC should have the Secure Shell (SSH) Client installed on them. For your Windows machines, you can download and install the SSH client from <https://docs.comp.nus.edu.sg/sites/default/files/SSHSecureShellClient-3.2.9.exe> Once you have the client installed, follow these steps to access `sunfire` :

- a) Run the client program by double clicking the icon:



- b) Click Quick Connect:



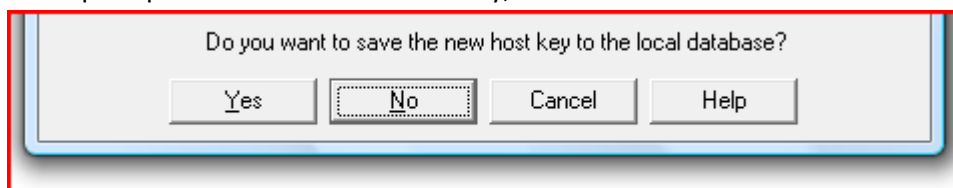
- c) Use the following credentials:

Host Name: sunfire.comp.nus.edu.sg

User Name: <your SoC account userID>

Port Number: 22

- d) When prompted to save the new host key, click No.



- e) When prompted for your password, enter your SoC Account password.

- f) You should see a prompt similar to:

```
userid@sunfire [23:32:10] ~ $
```

That means you're logged in. By default you're placed in your home directory on log in.

### 3. Common UNIX commands:

At the prompt, you should be able to enter commands or run programs. We'll now look at some of the common commands that can be used on UNIX.

- a) `echo`

The `echo` command simply "prints" whatever parameters you type after the command.

```
userid@sunfire [23:41:11] ~ $ echo "2271 is great"
2271 is great
```

The redirection operator, `>`, can be used to redirect the output to a file.

```
userid@sunfire [23:44:49] ~ $ echo "2271 is great" >
2271.txt
```

The above command would have written the string "2271 is great" to the file "2271.txt".

- b) `cat`

The `cat` command displays the contents of a file. It actually concatenates all the files listed as parameters and displays them.

```
userid@sunfire [23:48:32] ~ $ cat 2271.txt
2271 is great
```

c) `ls`

The `ls` command displays the contents of the current directory or the directory passed as parameter.

```
uesrid@sunfire [23:53:10] ~ $ ls
2271.txt  test1      test2
```

d) `man`

The `man` command is your friend. It displays the manual page associated with the commandname you pass to it as a parameter.

```
userid@sunfire [23:57:10] ~ $ man ls
Reformatting page. Please Wait... done
```

```
User Commands ls(1)
```

NAME

```
ls - list contents of directory
```

SYNOPSIS

```
/usr/bin/ls [-aAbcCdEfFghHilLmnopqrRstuvVx1@] [file]...
```

```
/usr/xpg4/bin/ls [-aAbcCdEfFghHilLmnopqrRstuvVx1@]
[file]...
```

```
/usr/xpg6/bin/ls [-aAbcCdEfFghHilLmnopqrRstuvVx1@]
[file]...
```

DESCRIPTION

```
For each file that is a directory, ls lists the contents of
the directory. For each file that is an ordinary file, ls
repeats its name and any other information requested. The
output is sorted alphabetically by default. When no argument
is given, the current directory (.) is listed. When several
arguments are given, the arguments are first sorted
--More-- (2%)
```

It will list all possible options associated with the command along with a description as well. You can and should use it to learn about all commands listed in this lab manual as well as other commands you might come to know of.

e) `cd`, `mkdir`, `rmdir` - Directory related commands

**cd** command is used to change the working directory.

```
userid@sunfire [00:04:02] ~ $ cd ~/2271
userid@sunfire [00:04:13] ~/2271 $
```

[Note: ~ refers to the home directory]

**mkdir** command is used to create a new directory

```
userid@sunfire [00:04:13] ~/2271 $ mkdir test
userid@sunfire [00:06:05] ~/2271 $ ls
2271.txt  test      test1      test2
```

**rmdir** command is used to delete a directory

```
uesrid@sunfire [00:06:10] ~/2271 $ rmdir test
userid@sunfire [00:07:47] ~/2271 $ ls
2271.txt  test1      test2
```

f) **cp, mv, rm** - File manipulation commands

**cp** command is used to make a copy of a file

```
uesrid@sunfire [00:07:48] ~/2271 $ cp 2271.txt 2271-copy.txt
uesrid@sunfire [00:09:52] ~/2271 $ ls
2271-copy.txt  2271.txt      test1          test2
```

**mv** command is used to move a file

```
userid@sunfire [00:09:53] ~/2271 $ mv 2271.txt test1/
userid@sunfire [00:11:56] ~/2271 $ ls ./test1
2271.txt
```

**rm** command is used to delete a file

```
userid@sunfire [00:12:06] ~/2271 $ rm 2271-copy.txt
rm: remove 2271-copy.txt (yes/no)? y
userid@sunfire [00:13:13] ~/2271 $ ls
test1  test2
```

g) **lpr, lpq, lprm, lpstat** - Printing related commands

You can get more details on these commands at:

<https://docs.comp.nus.edu.sg/node/1583>

h) `exit`, `logout`

The `exit` or `logout` command will terminate the shell and log you out.

i) Other commands you might want to look up:

`history`, `pwd`, `chmod`, `less`, `who`, `finger`, `quota`, `ps`, `kill`,  
`jobs`, `fg`, `bg`

A nice list of useful UNIX utilities can be obtained at:

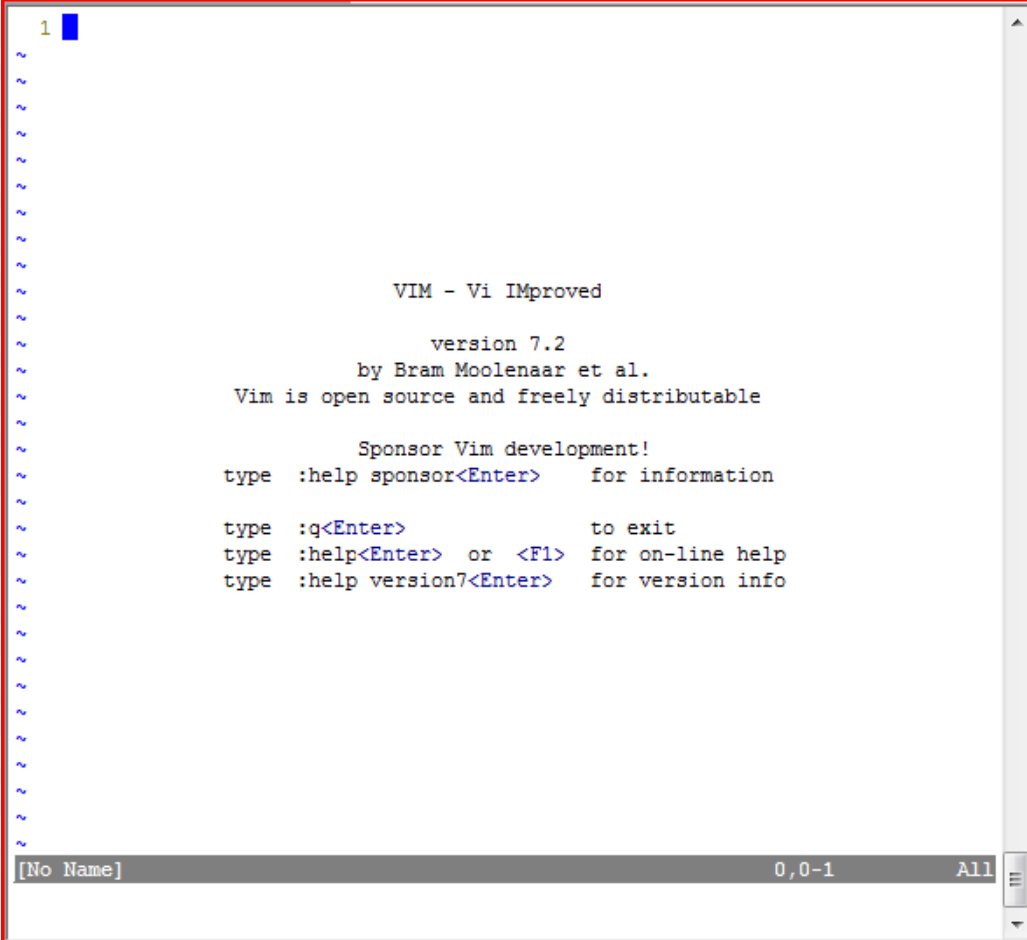
[http://en.wikipedia.org/wiki/List\\_of\\_Unix\\_utilities](http://en.wikipedia.org/wiki/List_of_Unix_utilities)

#### 4. Writing a program

There are a good few text editors available on the UNIX environment. E.g. `pico`, `nano`, `emacs`, `vim`, etc.

One of the more powerful and commonly used options is `vim`. Here we run you through the creation of a simple C file using `vim`.

a) Run `vim` by typing `vim` at the shell prompt. You should see a window like this:

A screenshot of the Vim text editor window. The window has a title bar with "[No Name]", "0,0-1", and "All". The main area shows the Vim startup screen with the following text: "1" at the top left, followed by a series of tilde characters (~) forming a border. Inside the border, the text reads: "VIM - Vi IMproved", "version 7.2", "by Bram Moolenaar et al.", "Vim is open source and freely distributable", "Sponsor Vim development!", "type :help sponsor<Enter> for information", "type :q<Enter> to exit", "type :help<Enter> or <F1> for on-line help", "type :help version7<Enter> for version info". The bottom status bar shows "[No Name]", "0,0-1", and "All".

b) In `vim`, there are three modes of operation:

i. Command mode

- ii. Insert mode
- iii. Visual mode [kind of a submode of command mode]

The <Esc> key will bring you from any mode to the Command mode, which is also the default mode when you start up vim.

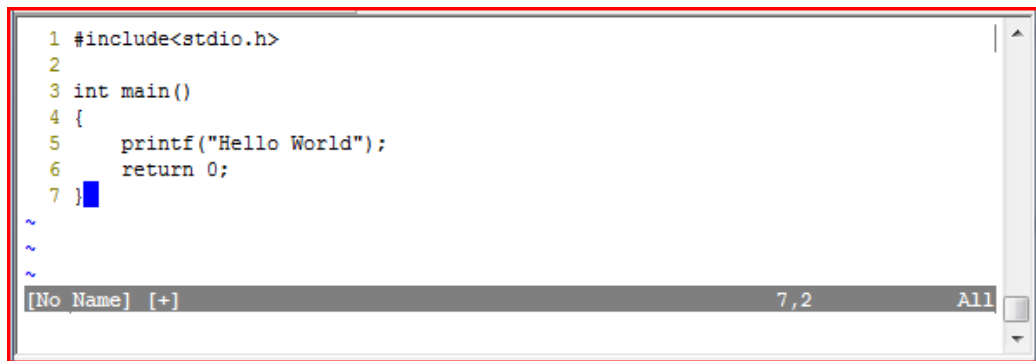
Pressing <i> in Command mode will bring you to the Insert mode.

Pressing <v> in Command mode will bring you to the visual mode.

- c) Now, having started up vim, we want to start typing our code. For this purpose, we have to first enter the Insert mode by pressing <i>

In insert mode, the shell window now is as good as your Notepad except for the mouse controls. You can type your code and use the arrow keys for movement.

Type out the following piece of code:



```
1 #include<stdio.h>
2
3 int main()
4 {
5     printf("Hello World");
6     return 0;
7 }
```

- d) Having typed the code, in order to write this text to a file, you first need to get back to the Command mode by pressing <Esc>.

Now, type the following:

**:w test.c**

[Take note of the colon at the start]

The above command essentially says write the text to a file named test.c

At any point to save the file again (equivalent to Ctrl+S in say, Notepad), get into Command mode and type:

**:w**

In order to quit vim, use the command:

**:q**

If you want to force quit without applying changes, use:

**:q!**

If you want to save the file and quit, use:

**:wq**

- e) When in command mode, you can use the following commands:

Movement:

- i. **h** - Move left

- ii. **l** - Move right
- iii. **j** - Move up
- iv. **k** - Move down

#### Positioning

- i. **o** - Beginning of line
- ii. **\$** - End of line
- iii. **w** - Jump to next word
- iv. **b** - Jump to word beginning
- v. **e** - Jump to word end

#### Editing

- i. **x** - Delete character at cursor
- ii. **y** - Copy the text starting at cursor until position determined by next command [e.g. **yw** would copy starting from current cursor position to the end of the current word]
- iii. **p** - Paste previously deleted or copied text
- iv. **r** - Replace character at current cursor position
- i. **d** - Delete text starting at cursor until position determined by next command [e.g. **dw** would delete starting from current cursor position to the end of the current word]
- ii. **dd** - Delete current line
- iii. **yy** - Copy current line

- f) For more information on using vim to its full potential, you can visit:  
[http://www.linuxconfig.org/Vim\\_Tutorial](http://www.linuxconfig.org/Vim_Tutorial)

## 5. Compiling and running a C program

We shall use the GNU C/C++ compiler, **gcc**. Below are instructions on how to compile and run the C program you wrote in step 4.

- a) Type the following command:

```
userid@sunfire [01:14:01] ~/2271 $ gcc -Wall test.c -o test.o
```

In the above command, you have invoked the gcc compiler to compile the file test.c

Two of the most commonly used options have been used here:

- Wall: turns on all the most commonly-used compiler warnings [Recommended]
- o <filename> : used to specify the output file [without this option, the default name of the output file is a.out]

- b) Having compiled the C file using the command above, you should find a new file named `test.o` in your current directory, which is a binary executable [Note: the file extension does not matter]. In order to run this file, simply enter the name of the file and hit <Enter>.

```
userid@sunfire [01:25:39] ~/2271 $ test.o
Hello World
userid@sunfire [01:26:13] ~/2271 $
```

- c) To look at other available options with `gcc`, simply use `man gcc`.

## 6. File transfer and remote drive mapping

If you wish to use a windows based IDE for writing code you're free to do so. You will however have to transfer the file to your UNIX directory in order to compile it and execute it on the UNIX server.

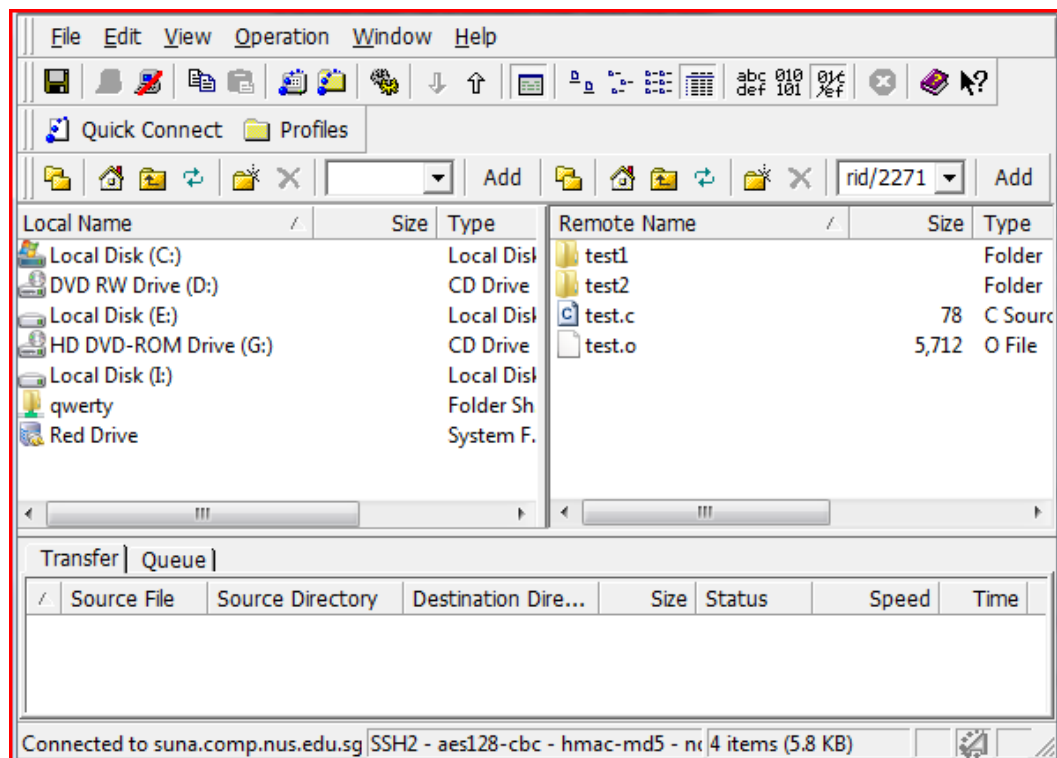
- a) File transfer

File transfer can be done using the Secure File Transfer Client that is installed along with the SSH client. To open up the File Transfer Client click the following highlighted icon in your SSH client window:



It should open up a window that looks like this:





Now you can move files back and forth between the server and the computer you're currently using.

b) Samba

You could use samba to map your home directory on sunfire to a network drive on your Windows PC. Details on how to do this can be obtained at:

<https://docs.comp.nus.edu.sg/node/1663>

[Note: You will need to be on the school network for this. If outside campus, you'll have to vpn into the NUS network before being able to use this service]

c) Red Drive

Another alternative to mapping the server directory to a local drive is via this 3<sup>rd</sup> party software called Red Drive.

[Disclaimer: We do not certify the safety of this software. We cannot be held responsible for any damage the Red Drive software might cause to your computer]

The development of this software has been stopped for a while. But the last stable release seems to do the job just fine. The advantage here is that you can access files on the server through the mapped drive without having to use vpn even when outside the school network.

If you're interested, head over to <http://www.jscape.com/reddrive/>, grab the installer and install it on your PC.

To map the server directory,

- i. Go to My Computer
- ii. Right click Red Drive

- iii. Click New Connection
- iv. Choose SFTP for the protocol
- v. Enter the same credentials you use to login to `sunfire` using the SSH client
- vi. Press OK

Double clicking on Red Drive should lead you to the mapped drive now.