

ASTR4004/ASTR8004

Astronomical Computing

Lecture 03

Christoph Federrath

2 August 2019

Remote computing

1 Setting up VPN (Virtual Private Network)

1. Most of the MSO servers and also some ANU services are only accessible from *inside* the ANU network. This means that they normally can't be accessed and used when you are at home. However, there is a service called 'VPN', which allows you to setup a secure connection, such that it looks to the ANU servers as if you were actually inside the Uni network, although you might be at home or even overseas.
2. Download the VPN server for your computer from the ANU VPN website. This is currently located here: <https://services.anu.edu.au/information-technology/login-access/virtual-private-network-0>
3. Follow the instructions on that web page to install the VPN. This will be useful for later also, because you can then connect to MSO servers and ANU services from outside the University network.
4. Now try to connect to the VPN, but as user name, use '[your_uni_id]@rsaa' or '[your_uni_id]@student' (staff: '[your_uni_id]@anubasic').
5. However, if we only want to connect to a server at RSAA, we will see later that there is another way (§5), which does *not* require the VPN.

2 Customising the Bash shell environment

1. First, lets change the Bash console prompt. Add to your `.bashrc` in your `$HOME` directory the line:
`PS1='\u@\h:\w>'`
2. Now make sure that the following line is also present in your `.bashrc`:
`test -s ~/.alias && . ~/.alias`
This ensures that the content of `.alias` is read and added to the Bash environment. Can you explain the syntax and what's going on here in this one-liner?

3. We can now modify (or create, if it does not exist yet) the `.alias` and add some useful shell aliases that allow you to access commands more quickly. A common command would be listing an extended, more detailed version of `ls`, containing the file sizes, last modification date of the files, and file permissions and ownership. Please add the following lines to `.alias`:

```
alias ls='ls -G $OPTIONS'
alias ll='ls -Glh $OPTIONS'
```

The first one should enable colourised output via the `-G` option, which applies to Mac OS (while the option `--color=auto` applies to Unix OS). Use `man ls` to see the manual page for the specific version and implementation of `ls` installed on the system that you are running `ls`.

Note that `$OPTIONS` is there to take additional options. Basically, we have now overloaded the `ls` command with a colourised version. The second line defines a new command alias called `ll`, which essentially calls `ls` with the additional options `-l` and `-h` for long-listing (more details) and human-readable file size output, respectively.

Try and understand the listing of files and directories via `ll`. When you run this from your home directory, what do the different columns mean, in particular the first column, which is encoding the file/directory permissions?

4. Now lets add a quick alias for your favourite editor, so you can quickly launch it from the shell, for example `emacs`:

```
alias e='emacs $OPTIONS'
```

Thus next time, you can very quickly open emacs by just typing `e -nw [file]`. Note that the `-nw` option refers to 'no window', meaning that emacs will be started directly in shell mode instead of graphical-user-interface (GUI) mode. This is very useful if we want to modify a file on a remote computer, where we do not have a fast network connection for graphical interactions. The text-based editor mode is much faster and is usually sufficient for most tasks. If you don't already have an efficient editor that you know how to use remotely, I recommend to learn how to use `emacs` and common short-cuts by searching the manual pages of `emacs` on the internet.

5. Make sure to restart Bash (or open a new Bash shell) so that your changes to `.bashrc` and `.alias` will take effect. Another way of activating the changes is to call `source .bashrc`.

3 Connecting to another computer using the shell

1. Connecting to another computer (also called remote host) in the local network or sometimes even the internet is easy with Bash. First, let's use the main program to do this, which is `ssh` ('Secure Shell'). Type

```
> ssh [your_mso_username]@malice.anu.edu.au
```

This connects you to the host `malice.anu.edu.au` at MSO.

2. You may find that your shell prompt on malice is different from the one we just set up on your local computer. This is because every host has its own Bash environment. Follow the procedures above that we used to set up our local environment via the files `.bashrc` and `.alias` to generate the same basic Bash environment on your remote account at RSAA/MSO. After making those modifications, log out (using command `exit`), and then

login again. You should now have the updated environment on `malice.anu.edu.au`, and in fact on any other MSO server, because your home directory is mounted on every server; in other words you could also connect to `motley.anu.edu.au` or `avatar.anu.edu.au`. In other words, your home space is shared by all servers at RSAA/MSO, so you don't have to set this up again when you connect to a different server at MSO, because each server at RSAA/MSO will read the same files and start up your Bash environment in the same (or at least similar) way, because your `$HOME` will be mounted on every MSO server. So you also have access to all your files on every MSO server.

3. Make sure you have a window manager installed (on Mac OS, you can install XQuartz from <https://www.xquartz.org>). Now lets try to plot something with `gnuplot`:

```
> gnuplot
> plot sin(x)
```

The first command starts gnuplot, a handy program to plot data in ASCII text files or analytic functions real quick. In gnuplot, `plot sin(x)` should plot a sin function. However, if you strictly followed this guide, you will get no plot, but instead an error message 'gnuplot: unable to open display, X11 aborted.' or something like that. The problem is that we only connected to `malice.anu.edu.au` in text/shell mode, but in order to bring up a plot window that shows the plot, we have to ssh-connect in *graphics mode*—we have to enable what's called 'X11 forwarding'.

4. In order to achieve X11 forwarding, we have to logout and reconnect using the ssh option `-X` or `-Y`, in order to enable X11 forwarding. Lets try:

```
> ssh -Y [your_mso_username]@malice.anu.edu.au
```

...and then try to plot the sin again with gnuplot. This should now bring up a new window with the `sin(x)` plot.

4 Customising ssh connections

1. Now we know how to connect to remote servers/hosts. However, the process and command for the connection is a bit lengthy, so in order to copy files and to connect more easily without having to specify your user name and the full remote hostname all the time, there is a neat way to customise ssh connections. On your local computer, change into your home directory: `> cd`

Then, change into the directory `.ssh/`. Note that this is a hidden directory (which is why it starts with a dot), so `ls` would normally not show it, however, if you do `ls -a`, then all files and directories are shown, including hidden ones. In case this does not exist, create the directory with `> mkdir ~/.ssh/`.

2. So, now in directory `.ssh/`, see if you have a file called `config`. If not, then simply create it (or if it exists), modify (creation and modification is the same command if you are using emacs):

```
> e -nw config
```

Now lets add the following lines:

```
Host malice
Hostname malice.anu.edu.au
User [your_mso_username]
```

These three lines define a new host name alias called `malice`, which will make it a lot easier for you to connect to malice. You don't even have to remember your (possibly a

bit cryptic) user name, because you'll see that this is automatically used now every time you connect or copy files to/from MSO servers.

3. Save and close `config` and now try to do the following:

```
> ssh -Y malice
```

This should connect you to malice directly. Note that we do not have to specify your user name or the full remote host name anymore; just `malice`.

4. Finally, define an alias called 'malice' in your local `.alias` with the ssh command above, including X11 forwarding to `malice`. This will allow you to simply type: `> malice` to start a connection to `malice.anu.edu.au` with your MSO user name from your local computer.

5 Setting up an `ssh` tunnel

1. If you are connected to the VPN (see Section 1 above), you can directly connect to `malice.anu.edu.au`. However, if you are outside the ANU network and you cannot use the VPN client for some reason, there is still another possibility to connect to the MSO servers. This requires you to first connect to a specific server at MSO that is accessible from the outside world. This server is called '`msossh1.anu.edu.au`'.

2. In order to connect to `malice.anu.edu.au`, we simply connect to `msossh1.anu.edu.au` first and then `ssh` from `msossh1.anu.edu.au` to `malice.anu.edu.au`. But in order to make our life easier, especially when copying files between remote computers, we can set up an `ssh` tunnel (in order to avoid the two-step process of first connecting/copying to `msossh1.anu.edu.au` and then to `malice.anu.edu.au`). Do this by adding/modifying the following lines in your `.ssh/config`:

```
Host malice
```

```
Hostname malice.anu.edu.au
```

```
User [your_mso_username]
```

```
ProxyCommand ssh -q -a -Y [your_mso_username]@msossh1.anu.edu.au nc %h %p
```

Save and close `.ssh/config`.

3. Now you should be able to connect to `malice.anu.edu.au` by using `> malice` directly from your local computer. It will probably ask for your password twice; the first time when it connects to `msossh1.anu.edu.au` and the second time when it connects from there to `malice.anu.edu.au`. So what this effectively does is that we tunnel through `msossh1` to `malice`.

6 Setting up private and public key pairs to connect to remote hosts without using a password

In order to establish password-less connections to remote hosts, you can follow the instructions, e.g., on this webpage: http://www.linuxproblem.org/art_9.html. Note however, that this is at your own risk. It may be easier for hackers to get access to your account *and automatically to all servers where you copied your public key to for password-less access*. In any case, if you create a public/private key pair, never give your *private key* to anyone and make sure it cannot

be read by others than you (strictly set user-only permissions via [chmod](#)) in your `~/.ssh/` directory.

7 Mounting remote file systems

You can use [sshfs](#) to mount a remote file system on your computer. Learn how to set that up on your computer by searching the internet.

CAUTION: be careful with copying-and-pasting code or shell commands from this PDF document into the shell or into another document (e.g., a script opened in an editor). You may not be getting what you are seeing on the screen, for example, depending on the exact editor, shell, and PDF viewer, the special characters `'(, ')`, `' ' '`, `'%'` can all cause problems when performing copy-paste operations. This is because of potential differences in character encodings between the PDF document and the shell or script editor.

In general, you can use [man](#) [\[command\]](#) to bring up the manual page of a program/command to learn about possible command-line options for shell program(s), or search the internet for answers/solutions; especially if you get unexpected errors or warnings. Usually, someone else with a similar or the same configuration of your computer/script/shell/code will have encountered a similar problem/issue and you primarily have to learn how to search for the right answer on the internet.