Practical Session One – Practical Networking Tools

In this practical class you will learn a little bit about the most common tools used for administration in a computer network. Unfortunately, in this class, we are limited by two things: Firstly we have only a small amount of time to cover a very, very complex subject; and secondly, we only have access to a regular users account on this computer system. Many of the more interesting things that you might choose to do with networks require *superuser* access to the computers which we do not have on this system. In this class I will take you through some of the basic commands which are used on a unix system for investigating networking. For the most part, in this class, we will be able to look but not touch — that is, we will not actually be able to affect the network a great deal. (This is a good thing, imagine if just anyone who showed up in classrooms could change how the network was operating.)

Begin by logging into the tower system from the sgi machine. All your commands should be executed from tower. (It might, sometimes, be interesting to try the commands from the sgi machine instead to see if it looks different.) Work through this sheet at your own pace. Please ask questions (though I may not be able to answer them all).

The Basics — Am I Connected to a Network?

The first commands we will look at are `ifconfig` and `ping`. `ifconfig` means interface configuration. It checks what networks are connected to the computer. Before you use the command, type `man ifconfig` to look at the manual for the command. Manual pages are often confusing but you should get into the habit of looking at them for basic information about a command. Don't worry too much that the majority of what you read won't make sense. Scroll down to the DESCRIPTION part of the page and read a few paragraphs to get an idea of what the command should do.

`ifconfig -a` will give you some basic information about what the computer is directly connected to. You will find that two interfaces should be set up, a loopback interface `lo0` and a 100Mb ethernet interface `hme0`. The loopback interface is simply a test device which is a loop back to the same computer (note the IP address 127.0.0.1 — that always means *my machine*, your machine will usually have a real IP address and, in addition, the address 127.0.0.1). [A well-known geek joke is to convince particularly stupid teenage hackers to launch an attack on the machine with the IP 127.0.0.1]

We could try to use ifconfig to remove an interface `ifconfig down hme0` but, if you try this you will find you don't have permission. Perhaps this is as well. (If you had succeeded you, and everyone else, would have lost contact with tower.) Note the netmask and the IP address for tower. (Run the same command on the SGI machine and see how the answer differs). (mtu is the maximum transfer unit, the largest size a packet can have). From the netmask we can determine what other machines on our subnet. We don't have much permission to actually change things with ifconfig so let us move on.

The next command is `ping` which can be thought of as being a little like sonar in a submarine film. It is used for seeing if machines on a network can be reached by us. As usual do `man ping` and take a look at what you think the command will do. A slight problem with it is that we can't actually get out of campus with it. From your brief look at the `ifconfig` command you should be able to work out the IP numbers of some machines which would be reachable. Try guessing some IP numbers and pinging the machines. To ping a machine type `ping` and its IP number or its name for example:

`ping xyzzy` will ping my machine xyzzy.

`ping -s xyzzy` will produce something a little more interesting. (You will have to use `Ctrl-C` to stop this command.)

The campus ethernet is fast enough that you will find almost no delays to anywhere on campus if all is working well. We can also try various other things for example `ping -s -I 5 [number]` will ping a machine with a particular IP number at intervals of five seconds. We can also try altering the time to live of packets (that is the number of hops that they can travel). This doesn't matter too much on campus since every machine is relatively close but try reducing it to 1. `ping -s -t 1 xyzzy`. Because we are restricted to campus this is not so interesting. The campus network is fast and reliable (usually). Load this `http://www.fifi.org/services/ping` into your web browser. This will allow you to ping sites all over the world. Try various web sites you know (here we have the opposite problem, we can't ping INTO campus so campus sites will not produce interesting information). Play with this script and get an idea of the separation of sites in the internet. (The site you are connecting to is in America, in San Fransisco so try somewhere distant from there like `www.yahoo.co.jp`. You may be able to get a good idea of how distant various places are from San Fransisco in network terms. (If you can't think of web sites, use `www.google.com` to find some).

`http://ftp.arl.mil/~mike/ping.html` gives some information about how the ping command was created.

Digging Deeper — networks and routes

The next command we will look at is `netstat` which, logically, shows the status of the network. Netstat can do an awful lot. So much so that most of the time you will want to pipe the output through the less command (use space for more information, b for back and q for quit).

`netstat -an | less` will give an awful lot of information about what is connecting to us (the n part means show IP as numbers — try the same command without). [notice that IP addresses seem to have FIVE digits, the fifth is a port number]. We can see all the connections being made to tower — there are likely to be an awful lot.

`netstat -rn | less` gives the routing table — that is, where do packets go if they want to get to a certain group of destinations (remember when we were discussing routing, here it is working in practice). You will notice that almost all of the routes are to local destinations (on the campus subnet — if that isn't clear look at the same thing without the n flag). There is only really one way *out* of the campus network so that's why most of the table is dealing with local routing.

`netstat -s | less` shows some statistics about bytes received and transmitted. Look particularly at the TCP statistics. How many bytes of data have been sent? How many have been received? (Much of the information here will mean little to you I am afraid).

We can modify these various command with the `-P` command to just look at certain protocols. For example

`netstat -s -P tcp | less` just looks at TCP stats. (Find information about incoming and outgoing ACKnowledgements. Is more data coming into the university or going out? Why might that be? Is more data TCP or UDP?)

`netstat -a -P udp | less` just looks at UDP connections.

Use the manual page to work out what netstat' information means (yes, it is hard to understand, but you should be able to pick up a lot of what is going on).

`dig` is used to look up IP addresses. Unfortunately, no man page is installed for dig but `dig -h | less` provides some information. All we will really use dig for here is to find IP addresses or to find names from IP addresses.

`dig xyzzy.york.ac.uk` finds the IP address of my machine. The additional information lists the authorities which have provided the information and their IP addresses. If we want to go the other way round then we can do:

`dig -x 144.32.100.24` which will get us back a human readable address from the IP address.

Try the dig command for a few web addresses that you know. There are other things we can do with dig but we don't have time to consider them here.

Some files of use

Finally, not very exciting, but we should take a look at some of the files which are important to network configuration. Let's take a look at some of these system files `/etc/hosts` (use `less /etc/hosts` to view it)will show a list of the most important machines connected to tower. This list is only there to provide IP addresses for these commonly used machines.

`/etc/resolv.conf` points at the nameservers — that is, the machines which are responsible for translating human readable addresses `www.amazon.com` into IP addresses `207.171.184.18`.

`/etc/services` lists ports and protocols for various programs that the machine runs. For example, the line:

```
ftp 21/tcp
```

means that ftp (file transfer program) connects to port 21 using the TCP protocol. What other services can you recognise?

Traceroute

`traceroute` is a routine for working out how we get between one IP address and another. Remember from lectures how this works? It pings machines in turn with increasing TTLs. (`http://www.freesoft.org/CIE/Topics/54.htm` is a reminder if you have forgotten.)

`traceroute xyzzy.york.ac.uk` shows how to get from tower to my computer. Not very interesting.

`traceroute www.amazon.com` is a bit more interesting a journey. Note that stars in the output indicate packets which are lost (often rejected by the computer which we are pinging).

Typically three packets are sent by traceroute but we can change this with the q flag:

`traceroute -q 1 news.bbc.co.uk` sends only one packet (note that the bbc is much closer to us than amazon in america)

Find `http://www.fifi.org/services/traceroute` in your web browser. Let's try finding the route from San Fransico to some other places. Try some sites at random and find out about how they are reached.

We can learn a lot from traceroute — how reliable are travel times? Think about the scales of times involved. Try a site in australia (google will find you one). You will find that the timescales go up considerably on some jumps. You will also find that the time to get a packet around the world is considerable in human terms (0.3 seconds approximately).

There are plenty more tools we could use but in such a short time there is not time to consider them all.