UWS UNIVERSITY OF THE WEST of SCOTLAND

# COMP09024 Unix System Administration

## Laboratory 8: Networking

*Networking*

**Learning Outcomes**

- **Understanding the basics of Linux networking**
- **Understanding the basic commands for network examination: *`ifconfig`*, *`route`* and *`netstat`*.**

This Lab is an introduction to Unix/Linux based networking. Unix and networking are almost synonymous. Unix-like systems did networking first! ... the rest is history ... As we are using a virtual environment you will first verify your setup before starting this laboratory. Things would be much more straightforward in a Unix/Linux dedicated laboratory environment. When you are ready, an introduction to the core network commands **`ifconfig`**, **`route`** and **`netstat`** will be given. This will be followed by an exercise that explains how to set up a graphical (X-like) connection between networked systems.

## Network Confirmation

Important: Most of the networking steps have been performed and default settings applied via a configuration file.

**Question**

- **Q8.1)** Althoug in this virutal environment we only have on single machine “under the same network”, what could be the reason to allocate a random MAC address in a virtual environment of more than one virtual machine? Consider what would happen if the same value was in use by more than one system.

Make sure, that after you logon as `student`, you open a terminal session as the root user. Now in a root window type the command **`ifconfig`**.

```
[root@UWS ~]# ifconfig
```

**Questions**

- **Q8.2)** What is the current IP-address given to the system and the MAC address allocated to the Virtual Box?
- **Q8.3)** Summarize in your own words what the command **`ifconfig`** does.

Your machine’s IP address and networking should be pre-configured for eth0. You will have seen the active interface(s) when you typed the ifconfig command. Adding the –s flag will give just a summary:

```
[root@UWS ~]# ifconfig -s
```

You can also use the **`ifconfig`** and the **`route`** commands to make changes and display the networking settings.

**Question**

- **Q8.4)** How many individual systems can a subnet with netmask 255.255.240.0 accommodate?

Now let us play with the **`ping`** command, type:

```
[root@UWS ~]# ping 127.0.0.1
```

exit after 5 to 10 responses by typing the control sequence <`CTRL-C`>

Then issue the command:

```
[root@UWS ~]# !!
```

Yes, an exclamation should be typed twice!

**Question**

- **Q8.5)** What is the meaning of the double exclamation mark in Unix-like operating systems?

Note the summary output of the **`ping`** command, which is displayed after `<CTRL-C>.` Then enter as follows (substituting your own machine's IP address where appropriate):

```
[root@UWS ~]# ping IP_loc_machine
```

Again, note the summary output of the **`ping`** command, which is displayed after the interrupt `<CTRL-C>.`

**Questions**

- **Q8.6)** Explain the **`ping`** command in your own words. What do the summary entries `min/avg/max/mdev` represent?

- **Q8.7)** Explain why in both cases, the summary output is very similar, for each of the `min/avg/max` and `mdev` values.

- **Q8.8)** Given that the loopback `127.0.0.1` is your **`IP_loc_machine`** kernel interface, why are both summary outputs not identical? Give a short explanation.

Now **`ping`** a remote machine, choose an external IP, for example 8.8.8.8:

```
[root@UWS ~]# ping CHOOSE_A_REMOTE_IP
```

**Questions**

- **Q8.9)** Do you see an output? if not, review the network settings or restart the virtual machine.

- **Q8.10)** How much longer are the times now for `min/avg/max` and `mdev`?

- **Q8.11)** Why are the times longer for the remote machine? (An easy answer will do. At your leisure you may wish to reflect on a complicated answer, including the propagation of electric signals in metallic media, kernel priorities, distance to the remote machine, etc.)

The ping command tells us that we are networked and can do things over the network.

## Network Observation

Just two commands are sufficient to configure a network interface: **`ifconfig`** and **`route`**.

```
[root@UWS ~]# route -n
```

Examine the outputs and try to answer the following questions:

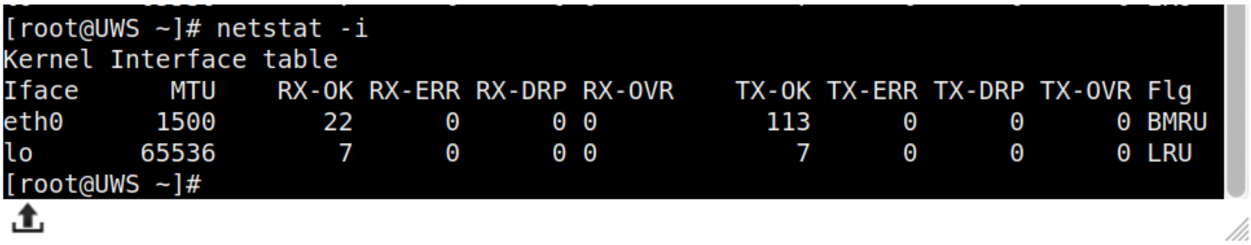
## Questions

- **Q8.12)** Summarise the **`route`** command in your own words.
- **Q8.13)** What are the Destinations, Gateways and Genmasks in the kernel?

Now it's time to introduce another very important program that allows you to monitor the performance of your local network: **`netstat`** .

The command **`netstat -i`** e.g. enables you to examine network performance statistics for the since the system was last booted. Try:

```
[root@UWS ~]# netstat -i
```



```
[root@UWS ~]# netstat -i
Kernel Interface table
Iface       MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500       22      0      0 0           113      0      0      0 BMRU
lo        65536        7      0      0 0             7      0      0      0 LRU
[root@UWS ~]#
```

The `MTU` and `Met` fields show the current `MTU` and metric values for that interface. The `RX` and `TX` columns show how many packets have been received or transmitted error-free (`RX-OK/TX-OK`) or damaged (`RX-ERR/TX-ERR`); how many were dropped (`RX-DRP/TX-DRP`); and how many were lost because of an overrun (`RX-OVR/TX-OVR`).

The last column shows the flags that have been set for this interface. These characters are one-character versions of the long flag names that are printed when you display the interface configuration with **`ifconfig`**:

| | |
|---|---|
| B | = A broadcast address has been set. |
| L | = This interface is a loopback device. |
| M | = All packets are received (promiscuous mode). |
| O | = ARP (Address resolution programme is turned off for this interface. |
| P | = This is a point-to-point connection. |
| R | = Interface is running. |
| U | = Interface is up. |

## Questions

- **Q8.14)** How many packets have been received OK and transmitted error free by your loopback interface?
- **Q8.15)** Is your ethernet a point-to-point connection?
- **Q8.16)** Are loopback and ethernet broadcasting?
- **Q8.17)** How many packages have been dropped on your system by the ethernet interface?

Now log in to your machine as **`student`** with the **`ssh`** command (that you set up in the previous laboratory).

Now that we are connected via SSH as student, log as root:

```
[student@UWS ~]# su root
passwd: qwerty
```

and type:

```
[root@UWS ~]# netstat -na
```

Try to understand this complex output! Check for the entry in the '**State**' column that says **ESTABLISHED**. (Tip: pipe with: **| grep 'ESTABLISHED'**)

**Question**

- **Q8.18)** What are the local and Foreign addresses for the **ESTABLISHED** net-connection?

A very powerful qualifier used in conjunction with **netstat** is the **statistics** one. Try this out on your own local machine:

```
[root@UWS ~]# netstat --statistics
```

Read the output and try to answer the following questions:

**Questions**

- **Q8.19)** How many **TCP** (protocol based) connections are open at the moment?
- **Q8.20)** How many segments have been received and send via **TCP/IP**?

Now continue by performing some actions while still connected via **SSH**. If you are disconnected, log in to the remote machine again and type in as root the following

```
[root@UWS ~]# cd / ; find /
```

After the command on the remote machine has finished executing (~1 minute), check for changes in the statistics output on your local machine by retyping: **netstat --statistics**

**Questions**

- **Q8.21)** Use your established knowledge to explain the command that you just issued on the remote system. What was the purpose of the semicolon?

- **Q8.22)** How many segments have been received and send via **TCP/IP** now? (I hope you see the difference.

As you should see, **netstat** gives a quick and precise overview of the connections, which are present within a Linux environment. It's a very quick way to e.g. check for nasty intruders and hung network processes. Given that it has numerous qualifiers it is very useful for investigating all kinds of network related issues.

Exit from the **SSH** tunnel and check for the existence of the **ESTABLISHED** link using the **netstat** command again:

To return to student user:

```
[root@UWS ~]# exit
```

To exit from the SSH tunnel:

```
[root@UWS ~]# exit
```

Check again the number of established connections:

```
[root@UWS ~]# netstat -na | grep ESTABLISHED
```

– END OF LAB –