Excercise 1

Open Two SSH windows

You will need two windows open to your VM for this lab. Go ahead and open the second one now.

Screen

Screen can be started by typing

screen

in a terminal. Press Enter after reading the introductory text.

Virtual terminals in Screen can be manipulated by pressing the Ctrl+A key combination, and subsequently pressing a key to execute one of the commands given below:

```
c creates a new virtual console

n switches to the next available virtual console

p switches back to the previous virtual console

" lists all available virtual consoles and their assigned numbers
hitting a number key brings the corresponding virtual console to the foreground

Esc lets you scroll back and forth in your terminal output

d detatches the current screen sessions and brings you back to the normal terminal
```

When a Screen session is detached, the processes that were running inside it aren't stopped. You can re-attach a detached session by typing

screen -r

in a terminal

Become root

All of the actions in this exercise are done as "root", so if you are not root already type the following in **both** windows:

```
mininet@mininet-vm:~$ sudo bash
root@mininet-vm:~#
```

HTTP with Mininet and OVS-OFCTL

Ensure that no other controller is present

> Note that 'controller' is a simple OpenFlow reference controller implementation in linux. We want to ensure that this is not running to interfere with adding flows manually.

```
root@mininet-vm:~# killall controller
controller: no process found
root@mininet-vm:~#
```

Clear all mininet components

```
root@mininet-vm:~# mn -c
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd
ovs-controller udpbwtest mnexec ivs 2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd
ovs-controller udpbwtest mnexec ivs 2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
***  Removing OVS datapathsovs-vsctl --timeout=1 list-br
ovs-vsctl del-br s1
ovs-vsctl del-br s2
ovs-vsctl del-br s3
ovs-vsctl del-br s4
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '(\w+-eth\w+)'
*** Cleanup complete.
root@mininet-vm:~#
```

Start mininet with a single switch topology and three hosts without a controller

```
sudo mn --topo=single,3 --mac --switch ovsk --controller=none
```

Try these useful mininet commands

mininet command help

```
mininet> help
```

Topology information

```
mininet> net
```

IP address information

```
mininet> dump
```

List of nodes and switches

```
mininet> nodes
```

Ping from each host to all other hosts

```
mininet> pingall
```

Try these ovs-ofctl commands. (Use sh to execute a shell command from in mininet.)

ovs-ofctl command help

```
mininet> sh ovs-ofctl --help | less
```

Shows switch information including OpenFlow Feature support, port type, port speed and mac addresses.

```
mininet> sh ovs-ofctl show s1
```

Shows active flow rules on the switch

```
mininet> sh ovs-ofctl dump-flows s1
```

Show port traffic and error statistics

```
mininet> sh ovs-ofctl dump-ports s1
```

Add Flows to the switch

```
mininet> sh ovs-ofctl add-flow s1
arp,idle_timeout=180,priority=500,actions=output:1,2,3
mininet> sh ovs-ofctl add-flow s1 tcp,tp_dst=80,nw_dst=10.0.0.3,actions=output:3
mininet> sh ovs-ofctl add-flow s1 ip,nw_dst=10.0.0.1,actions=output:1
```

`

Status Check

```
mininet> sh ovs-ofctl dump-flows s1
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=47.91s, table=0, n_packets=0, n_bytes=0, idle_age=47,
ip,nw_dst=10.0.0.1 actions=output:1
 cookie=0x0, duration=73.85s, table=0, n_packets=18, n_bytes=756, idle_timeout=180,
idle_age=1, priority=500,arp actions=output:1,output:2,output:3
 cookie=0x0, duration=62.666s, table=0, n_packets=0, n_bytes=0, idle_age=62,
tcp,nw_dst=10.0.0.3,tp_dst=80 actions=output:3
```

Start a web server on h3

```
mininet> h3 python -m SimpleHTTPServer 80 &
```

Retrieve the web page from h1

```
mininet> h1 wget http://10.0.0.3
--2015-10-12 02:00:26--  http://10.0.0.3/
Connecting to 10.0.0.3:80... connected.
```

```
HTTP request sent, awaiting response... 200 OK
Length: 530 [text/html]
Saving to: ‘index.html.1’

     0K                                                       100% 49.5M=0s

2015-10-12 02:00:26 (49.5 MB/s) - ‘index.html.1’ saved [530/530]
```

Can you ping?

```
mininet> pingall
```

Why not?

Why might you use the following flow rules instead?

```
mininet> sh ovs-ofctl add-flow s1 arp,in_port=1,actions=output:3
mininet> sh ovs-ofctl add-flow s1 arp,in_port=3,actions=output:1
```