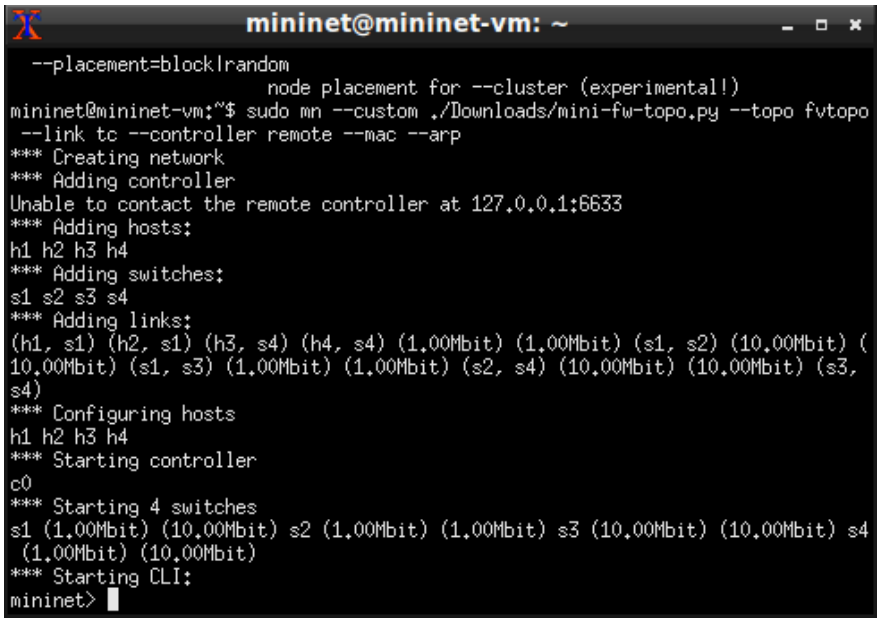
Exercise 7 – Mininet & FlowVisor

1. Create your own FlowVisor topology (50P)

(50P) Using the Mininet Python API, create the FlowVisor WAN topology (which you may know from earlier exercises) in a file **mini-fw-topo.py**:
After you have defined it start your topology:
`$ sudo mn --custom mini-fw-topo.py --topo <YOURTOPOCLASS> --link tc --controller remote --mac --arp`
In the Mininet console, make sure that your topology is properly connected.
The network is created as below, and the code is in the attachment with the name Exercise7-mini-fw-topo.py.



2. Slice the Network (100P)

Now, slice your network so that it supports the following slices:
In short, this slice arrangement allows traffic to be sent from h1 to h3 and h2 to h4 (and vice-versa) only, even though the topology itself would allow sending traffic between arbitrary pairs of hosts.
For slicing a network with FlowVisor in general, you need to take the following steps.
First, make sure you set up the flowvisor package correctly in Exercise 5. Then, start flowvisor in a new terminal:

```
$ sudo /etc/init.d/flowvisor start
```

We have to enable topology control for flowvisor as well:

```
$ fvctl –f /dev/null set-config –-enable-topo-ctrl
```

Similar to **ovs-ofctl**, **fvctl** is the control channel that we will use for flowvisor. The option **–f** refers to the flowvisor password file. Since we have set the password to be empty, it we can hand it **/dev/null**. This part will be present in all the following fvctl calls.
Restart flowvisor:

```
$ sudo /etc/init.d/flowvisor restart
```

Now, have a look at the FlowVisor configuration:

```
$ fvctl –f /dev/null get-config
```

This also has the purpose of making sure that flowvisor is actually running and that all the switches have indeed a connection to flowvisor. The configuration should show this.

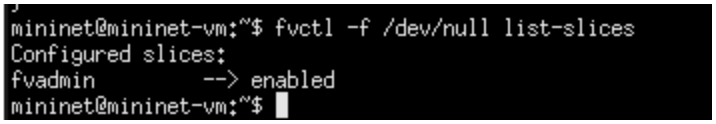
a. (5P) Which part of the configuration file tells you that all four switches have connected to flowvisor?

This part means 4 switches have been connected.

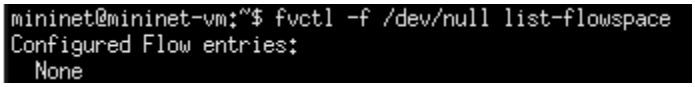
In the lecture, you also got a brief overview over the major flowvisor commands. Now, make use of these commands to

b. (5P) List the currently existing slices.

Only one slice called fvadmin



c. (5P) List the currently existing flowspaces.



d. (5P) List the currently connected switches.

```
mininet@mininet-vm:~$ fvctl -f /dev/null list-datapaths
Connected switches:
  1 : 00:00:00:00:00:00:00:01
  2 : 00:00:00:00:00:00:00:02
  3 : 00:00:00:00:00:00:00:03
  4 : 00:00:00:00:00:00:00:04
```

e. (5P) List the currently existing links.

Links between switches.

```
mininet@mininet-vm:~$ fvctl -f /dev/null list-links
[
  {
    "dstDPID": "00:00:00:00:00:00:00:02",
    "dstPort": "1",
    "srcDPID": "00:00:00:00:00:00:00:01",
    "srcPort": "1"
  },
  {
    "dstDPID": "00:00:00:00:00:00:00:01",
    "dstPort": "1",
    "srcDPID": "00:00:00:00:00:00:00:02",
    "srcPort": "1"
  },
  {
    "dstDPID": "00:00:00:00:00:00:00:04",
    "dstPort": "1",
    "srcDPID": "00:00:00:00:00:00:00:02",
    "srcPort": "2"
```

Afterwards, proceed with slicing your topology:

f. (10P) Create the appropriate slices.

Created two slices, one is upper, one is lower.

```
mininet@mininet-vm:~$ fvctl -f /dev/null list-slices
Configured slices:
fvadmin         --> enabled
upper           --> enabled
lower           --> enabled
```

g. (40P) Create the appropriate flowspaces.

```
mininet@mininet-vm:~$ fvctl -f /dev/null list-flowspace
Configured Flow entries:
{"force-enqueue": -1, "name": "switch1-port3", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:01", "id": 1, "match": {"wildcards": 4194302, "in_port": 3}}
{"force-enqueue": -1, "name": "switch1-port1", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:01", "id": 2, "match": {"wildcards": 4194302, "in_port": 1}}
{"force-enqueue": -1, "name": "switch2-port1", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:02", "id": 3, "match": {"wildcards": 4194302, "in_port": 1}}
{"force-enqueue": -1, "name": "switch2-port2", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:02", "id": 4, "match": {"wildcards": 4194302, "in_port": 2}}
{"force-enqueue": -1, "name": "switch4-port1", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:04", "id": 5, "match": {"wildcards": 4194302, "in_port": 1}}
{"force-enqueue": -1, "name": "switch4-port3", "slice-action": [{"slice-name": "
upper", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:04", "id": 6, "match": {"wildcards": 4194302, "in_port": 3}}
{"force-enqueue": -1, "name": "switch1-port4", "slice-action": [{"slice-name": "
lower", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:01", "id": 7, "match": {"wildcards": 4194302, "in_port": 4}}
{"force-enqueue": -1, "name": "switch1-port2", "slice-action": [{"slice-name": "
lower", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:01", "id": 8, "match": {"wildcards": 4194302, "in_port": 2}}
{"force-enqueue": -1, "name": "switch3-port1", "slice-action": [{"slice-name": "
lower", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:03", "id": 9, "match": {"wildcards": 4194302, "in_port": 1}}
{"force-enqueue": -1, "name": "switch3-port2", "slice-action": [{"slice-name": "
lower", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
00:00:03", "id": 10, "match": {"wildcards": 4194302, "in_port": 2}}
{"force-enqueue": -1, "name": "switch4-port2", "slice-action": [{"slice-name": "
lower", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:
```

h. (10P) Connect an instance of the POX controller to each of your slices

Upper slice:

```
mininet@mininet-vm:~/pox$ sudo ./pox.py openflow.of_01 --port=10001 forwarding.l2_pairs
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:forwarding.l2_pairs:Pair-Learning switch running.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[00-00-00-00-00-04 1] connected
INFO:openflow.of_01:[00-00-00-00-00-02 2] connected
INFO:openflow.of_01:[00-00-00-00-00-01 3] connected
```

Lower slice:

```
mininet@mininet-vm:~/pox$ sudo ./pox.py openflow.of_01 --port=10002 forwarding.l
2_pairs
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:forwarding.l2_pairs:Pair-Learning switch running.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[00-00-00-00-00-04 1] connected
INFO:openflow.of_01:[00-00-00-00-00-03 2] connected
INFO:openflow.of_01:[00-00-00-00-00-01 3] connected
```

i. (10P) In Mininet, verify that your slicing works properly, i.e., h1 can reach h3 but not h2 and h4, and h2 can reach h4, but not h1 and h3.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X h3 X
h2 -> X X h4
h3 -> h1 X X
h4 -> X h2 X
*** Results: 66% dropped (4/12 received)
```

SUBMISSION: Provide screenshots of the output of list-slices, list-flowspace, the controllers and your mininet console with the relevant information after you completed all steps.

3. Another Topology (50P)

Also do the slicing for the topology we saw in Exercise 3 (shown in the figure below).

First clean up the upper and lower slice. Then create new slice video and non-video. Add flowspaces, connect to the controllers.

The executing script is in the attachment called exercise7-video.

start with (switch s1, port 3) that connects s1 to h1. First, create two high priority flowspaces that match all traffic with source or destination port of 9999 at that port, and assign it to the video slice. We have arbitrarily chosen a high priority value of 100. Next, create a low-priority flowspace that matches all traffic at that port, and assign it to the non-video slice. his results in all the video traffic at (switch s1, port 3) being handled by the video slice, and all the non-video traffic at being handled by the non-video slice. Similarly, create flowspaces to segregate video and non-video traffic at (switch s1, port 4). Similarly, create flowspaces for the other two edge ports (at switch s4). Create flowspaces for (switch s4, port 3). Create flowspaces for (switch s4, port 4). We are now just left with the internal ports. Creating flowspaces for them is simple, as traffic at each internal port is exclusively handled by one slice. Create flowspaces for the internal ports just the way you did part 2 of this exercise.

This is the flowspaces(part of screen shot):

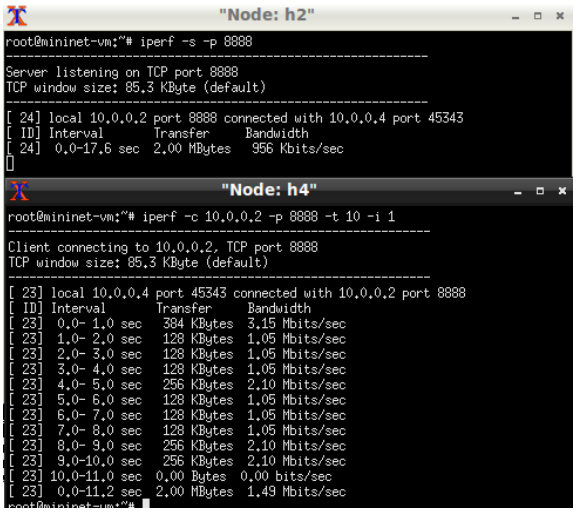
```
mininet@mininet-vm:~$ fvctl -f /dev/null list-flowspace
Configured Flow entries:
{"force-enqueue": -1, "name": "s1-p3-nv", "slice-action": [{"slice-name": "non-v
ideo", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:0
0:00:01", "id": 41, "match": {"wildcards": 4194302, "in_port": 3}}
{"force-enqueue": -1, "name": "s1-p4-nv", "slice-action": [{"slice-name": "non-v
ideo", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:0
0:00:01", "id": 42, "match": {"wildcards": 4194302, "in_port": 4}}
{"force-enqueue": -1, "name": "s4-p3-nv", "slice-action": [{"slice-name": "non-v
ideo", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:0
0:00:04", "id": 43, "match": {"wildcards": 4194302, "in_port": 3}}
{"force-enqueue": -1, "name": "s4-p4-nv", "slice-action": [{"slice-name": "non-v
ideo", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:00:00:00:00:0
0:00:04", "id": 44, "match": {"wildcards": 4194302, "in_port": 4}}
{"force-enqueue": -1, "name": "dpid1-port3-video-src", "slice-action": [{"slice-
name": "video", "permission": 7}], "queues": [], "priority": 100, "dpid": "00:00
:00:00:00:00:00:01", "id": 51, "match": {"wildcards": 4194190, "tp_src": 9999, "
dl_type": 2048, "nw_proto": 6, "in_port": 3}}
{"force-enqueue": -1, "name": "dpid1-port3-video-dst", "slice-action": [{"slice-
name": "video", "permission": 7}], "queues": [], "priority": 100, "dpid": "00:00
:00:00:00:00:00:01", "id": 52, "match": {"wildcards": 4194126, "dl_type": 2048,
"nw_proto": 6, "in_port": 3, "tp_dst": 9999}}
{"force-enqueue": -1, "name": "dpid1-port3-non-video", "slice-action": [{"slice-
name": "non-video", "permission": 7}], "queues": [], "priority": 1, "dpid": "00:
00:00:00:00:00:00:01", "id": 53, "match": {"wildcards": 4194302, "in_port": 3}}
{"force-enqueue": -1, "name": "dpid1-port4-video-src", "slice-action": [{"slice-
name": "video", "permission": 7}], "queues": [], "priority": 100, "dpid": "00:00
:00:00:00:00:00:01", "id": 54, "match": {"wildcards": 4194190, "tp_src": 9999, "
dl_type": 2048, "nw_proto": 6, "in_port": 4}}
{"force-enqueue": -1, "name": "dpid1-port4-video-dst", "slice-action": [{"slice-
name": "video", "permission": 7}], "queues": [], "priority": 100, "dpid": "00:00
:00:00:00:00:00:01", "id": 55, "match": {"wildcards": 4194126, "dl_type": 2048,
"nw_proto": 6, "in_port": 4, "tp_dst": 9999}}
```

Test on h1 and h3:

```
^Croot@mininet-vm:~# iperf -s -p 1234
------------------------------------------------------------
Server listening on TCP port 1234
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 24] local 10.0.0.1 port 1234 connected with 10.0.0.3 port 32817
[ ID] Interval       Transfer     Bandwidth
[ 24]  0.0-11.9 sec  12.2 MBytes  8.63 Mbits/sec
```

```
------------------------------------------------------------
Client connecting to 10.0.0.1, TCP port 1234
TCP window size: 85.3 KByte (default)
------------------------------------------------------------
[ 23] local 10.0.0.3 port 32817 connected with 10.0.0.1 port 1234
[ ID] Interval       Transfer     Bandwidth
[ 23]  0.0- 1.0 sec  1.62 MBytes  13.6 Mbits/sec
[ 23]  1.0- 2.0 sec  1.00 MBytes  8.39 Mbits/sec
[ 23]  2.0- 3.0 sec  1.38 MBytes  11.5 Mbits/sec
[ 23]  3.0- 4.0 sec  1.00 MBytes  8.39 Mbits/sec
[ 23]  4.0- 5.0 sec  1.12 MBytes  9.44 Mbits/sec
[ 23]  5.0- 6.0 sec  1.25 MBytes  10.5 Mbits/sec
[ 23]  6.0- 7.0 sec   896 KBytes  7.34 Mbits/sec
[ 23]  7.0- 8.0 sec  1.62 MBytes  13.6 Mbits/sec
[ 23]  8.0- 9.0 sec  1.00 MBytes  8.39 Mbits/sec
[ 23]  9.0-10.0 sec  1.25 MBytes  10.5 Mbits/sec
[ 23]  0.0-10.5 sec  12.2 MBytes  9.78 Mbits/sec
```

Test on h2 and h4:


"Node: h2"
root@mininet-vm:~# iperf -s -p 8888
Server listening on TCP port 8888
TCP window size: 85.3 KByte (default)
[24] local 10.0.0.2 port 8888 connected with 10.0.0.4 port 45343
[ID] Interval Transfer Bandwidth
[24] 0.0-17.6 sec 2.00 MBytes 956 Kbits/sec
"Node: h4"
root@mininet-vm:~# iperf -c 10.0.0.2 -p 8888 -t 10 -i 1
Client connecting to 10.0.0.2, TCP port 8888
TCP window size: 85.3 KByte (default)
[23] local 10.0.0.4 port 45343 connected with 10.0.0.2 port 8888
[ID] Interval Transfer Bandwidth
[23] 0.0- 1.0 sec 384 KBytes 3.15 Mbits/sec
[23] 1.0- 2.0 sec 128 KBytes 1.05 Mbits/sec
[23] 2.0- 3.0 sec 128 KBytes 1.05 Mbits/sec
[23] 3.0- 4.0 sec 128 KBytes 1.05 Mbits/sec
[23] 4.0- 5.0 sec 256 KBytes 2.10 Mbits/sec
[23] 5.0- 6.0 sec 128 KBytes 1.05 Mbits/sec
[23] 6.0- 7.0 sec 128 KBytes 1.05 Mbits/sec
[23] 7.0- 8.0 sec 128 KBytes 1.05 Mbits/sec
[23] 8.0- 9.0 sec 256 KBytes 2.10 Mbits/sec
[23] 9.0-10.0 sec 256 KBytes 2.10 Mbits/sec
[23] 10.0-11.0 sec 0.00 Bytes 0.00 bits/sec
[23] 0.0-11.2 sec 2.00 MBytes 1.49 Mbits/sec