

1. First we run the pingall command. This is to ensure that all ICMP packets are going through to the correct port and none are breaking the rules set at the table. For reference this is the logic I followed:

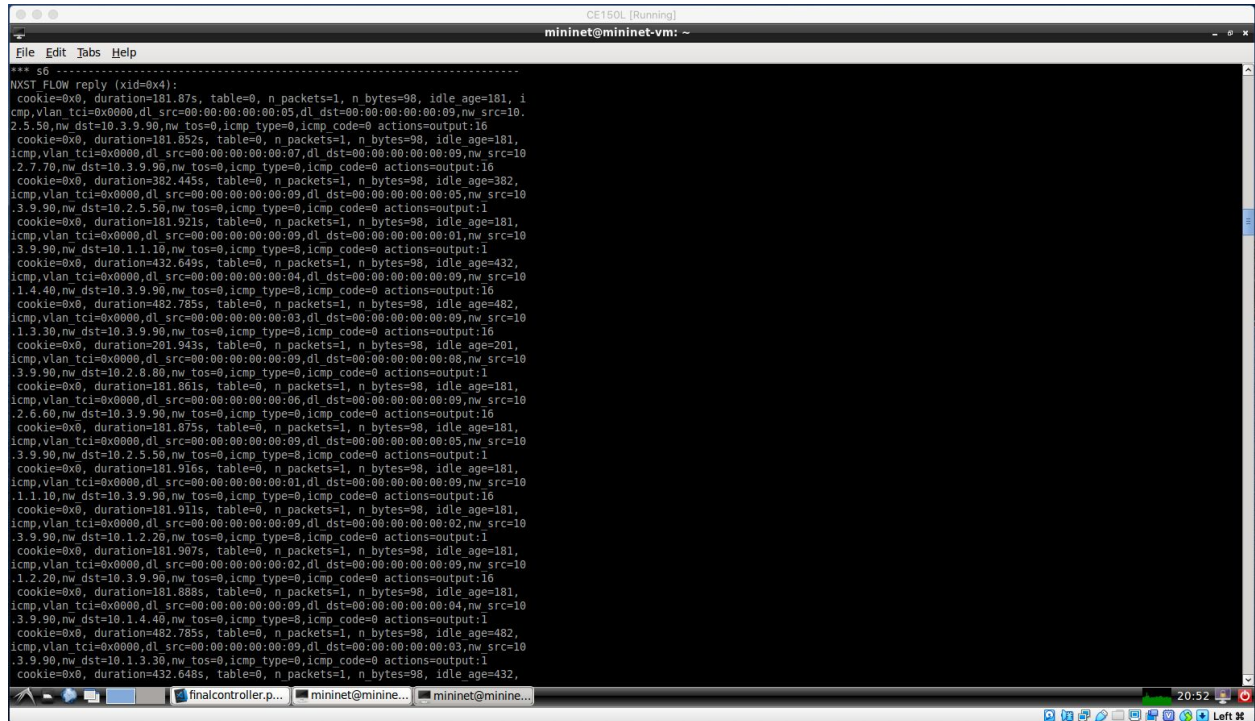
Device	Mininet Name	IP Address	Description
Floor 1 Hosts	h10, h20, h30, h40	10.1.1.10/24 10.1.2.20/24 10.1.3.30/24 10.1.4.40/24	Computers on floor 1 of the Department A in the company.
Floor 2 Hosts	h50, h60, h70, h80	10.2.5.50/24 10.2.6.60/24 10.2.7.70/24 10.2.8.80/24	Computers on floor 2 of the Department B in the company.
Trusted Host	h_trust	108.24.31.112/24	A trusted computer outside our network. This host is owned by certified employee from Department A.
Untrusted Host	h_untrust	106.44.82.103/24	An untrusted computer outside our network. We treat this computer as a potential hacker.
Server	h_server	10.3.9.90/24	A server used by our internal or trusted hosts.

- a. These are the results of the pingall command: this should show you that all of the logic is correct.

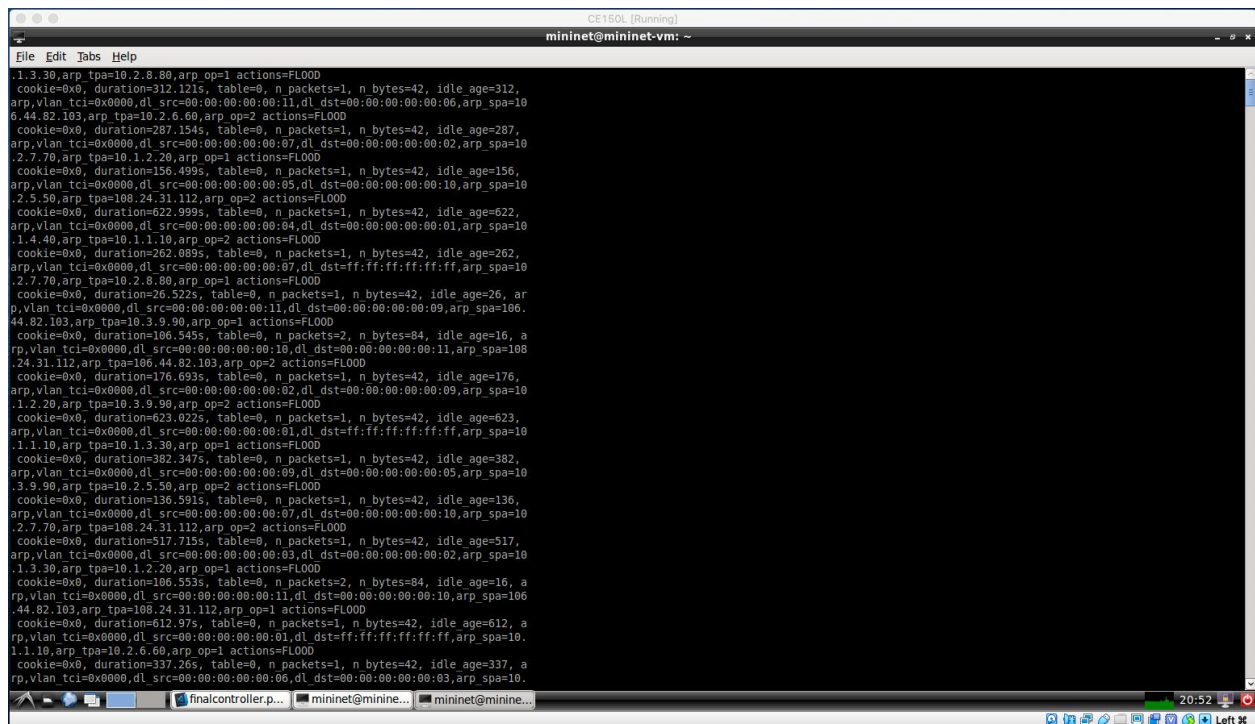
```

mininet@mininet-vm:~$ sudo python ~/final.py
mininet> pingall
*** Ping: testing ping reachability
h10 -> h20 h30 h40 X X X X h_server h_trust X
h20 -> h10 h20 h40 X X X X h_server h_trust X
h30 -> h10 h20 h40 X X X X h_server h_trust X
h40 -> h10 h20 h30 X X X X h_server h_trust X
h50 -> X X X X h60 h70 h80 h_server X X
h60 -> X X X X h50 h70 h80 h_server X X
h70 -> X X X X h50 h60 h80 h_server X X
h80 -> X X X X h50 h60 h70 h_server X X
h_server -> h10 h20 h30 h40 h50 h60 h70 h80 X X
h_trust -> h10 h20 h30 h40 X X X X h_untrust
h_untrust -> X X X X X X X X h_trust
*** Results: 54% dropped (50/110 received)
mininet>
  
```

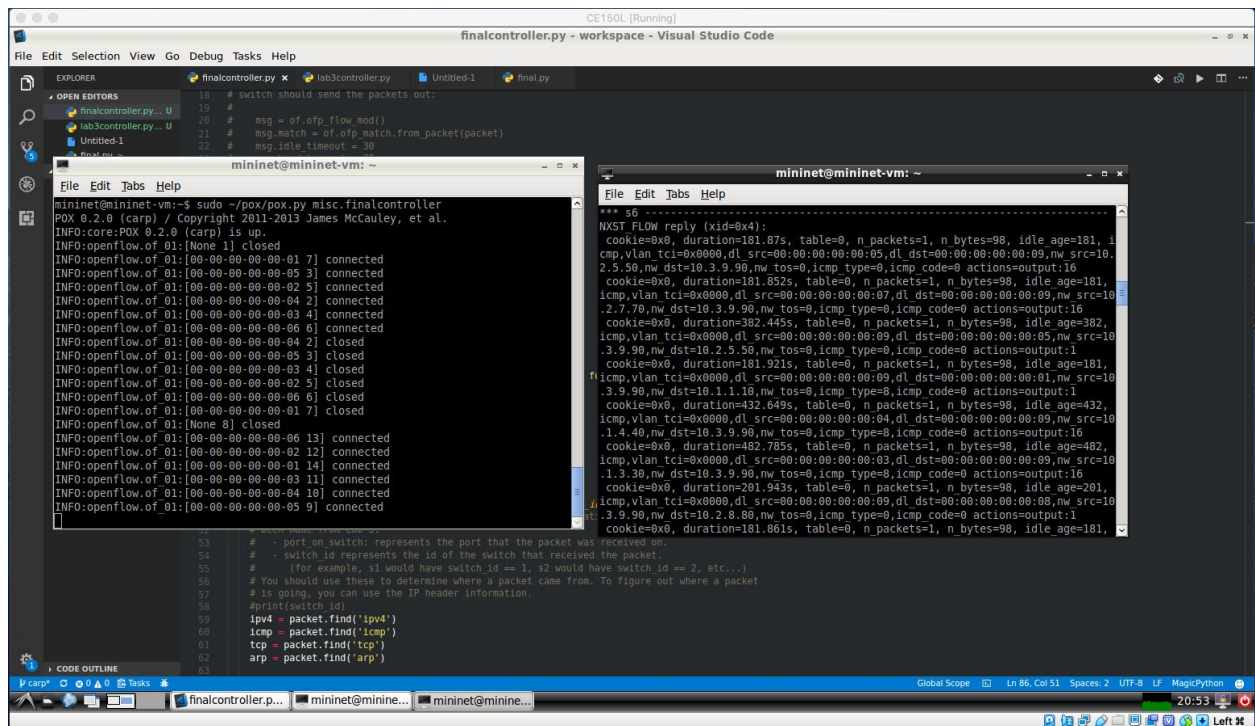
- Next is the dpctl dump-flows. This should show you exactly which ports the packets are traveling through. Also should tell you how each type of packet is moving. ARP is flooded while all IPV4 packets (ICMP and TCP) are being directed through to specific ports. I have provided multiple screenshots of this to show you what a switches' dump flow looks like. I could not provide the entire output because it was so large that it had gotten cut off. However, if you were to test it yourself, you would get the correct output.



```
CE150L [Running]
mininet@mininet-vm: ~
File Edit Tabs Help
*** 56 ***
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=181.87s, table=0, n packets=1, n bytes=98, idle age=181, i
cmp,vlan tci=0x0000,d1 src=00:00:00:00:00:05,d1 dst=00:00:00:00:00:09,nw src=10
2.5.50,nw dst=10.3.9.90,nw tos=0,icmp type=0,icmp code=0 actions=output:16
 cookie=0x0, duration=181.85s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:07,d1 dst=00:00:00:00:00:09,nw src=10
2.7.70,nw dst=10.3.9.90,nw tos=0,icmp type=8,icmp code=0 actions=output:16
 cookie=0x0, duration=382.44s, table=0, n packets=1, n bytes=98, idle age=382,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:05,nw src=10
3.9.90,nw dst=10.2.5.50,nw tos=0,icmp type=0,icmp code=0 actions=output:1
 cookie=0x0, duration=181.92s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:01,nw src=10
3.9.90,nw dst=10.1.1.10,nw tos=0,icmp type=8,icmp code=0 actions=output:1
 cookie=0x0, duration=432.64s, table=0, n packets=1, n bytes=98, idle age=432,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:04,d1 dst=00:00:00:00:00:09,nw src=10
1.4.40,nw dst=10.3.9.90,nw tos=0,icmp type=8,icmp code=0 actions=output:16
 cookie=0x0, duration=482.78s, table=0, n packets=1, n bytes=98, idle age=482,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:03,d1 dst=00:00:00:00:00:09,nw src=10
1.3.30,nw dst=10.3.9.90,nw tos=0,icmp type=8,icmp code=0 actions=output:16
 cookie=0x0, duration=482.78s, table=0, n packets=1, n bytes=98, idle age=482,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:08,nw src=10
3.9.90,nw dst=10.2.8.80,nw tos=0,icmp type=0,icmp code=0 actions=output:1
 cookie=0x0, duration=181.86s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:06,d1 dst=00:00:00:00:00:09,nw src=10
2.6.60,nw dst=10.3.9.90,nw tos=0,icmp type=0,icmp code=0 actions=output:16
 cookie=0x0, duration=181.87s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:05,nw src=10
3.9.90,nw dst=10.2.5.50,nw tos=0,icmp type=8,icmp code=0 actions=output:1
 cookie=0x0, duration=181.91s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:01,d1 dst=00:00:00:00:00:09,nw src=10
1.1.10,nw dst=10.3.9.90,nw tos=0,icmp type=0,icmp code=0 actions=output:16
 cookie=0x0, duration=181.91s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:02,nw src=10
3.9.90,nw dst=10.1.2.20,nw tos=0,icmp type=0,icmp code=0 actions=output:1
 cookie=0x0, duration=181.90s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:02,d1 dst=00:00:00:00:00:09,nw src=10
1.2.20,nw dst=10.3.9.90,nw tos=0,icmp type=0,icmp code=0 actions=output:16
 cookie=0x0, duration=181.88s, table=0, n packets=1, n bytes=98, idle age=181,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:04,nw src=10
3.9.90,nw dst=10.1.4.40,nw tos=0,icmp type=8,icmp code=0 actions=output:1
 cookie=0x0, duration=482.78s, table=0, n packets=1, n bytes=98, idle age=482,
icmp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:03,nw src=10
3.9.90,nw dst=10.1.3.30,nw tos=0,icmp type=0,icmp code=0 actions=output:1
 cookie=0x0, duration=432.64s, table=0, n packets=1, n bytes=98, idle age=432,
```



```
CE150L [Running]
mininet@mininet-vm: ~
File Edit Tabs Help
1.3.30,arp tpa=10.2.8.80,arp op=1 actions=FL000
 cookie=0x0, duration=312.12s, table=0, n packets=1, n bytes=42, idle age=312,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:11,d1 dst=00:00:00:00:00:06,arp spa=10
6.44.82.103,arp tpa=10.2.6.60,arp op=2 actions=FL000
 cookie=0x0, duration=287.15s, table=0, n packets=1, n bytes=42, idle age=287,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:07,d1 dst=00:00:00:00:00:02,arp spa=10
2.7.70,arp tpa=10.1.2.20,arp op=1 actions=FL000
 cookie=0x0, duration=156.49s, table=0, n packets=1, n bytes=42, idle age=156,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:05,d1 dst=00:00:00:00:00:10,arp spa=10
2.5.50,arp tpa=108.24.31.112,arp op=2 actions=FL000
 cookie=0x0, duration=622.99s, table=0, n packets=1, n bytes=42, idle age=622,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:04,d1 dst=00:00:00:00:00:01,arp spa=10
1.4.40,arp tpa=10.1.1.10,arp op=2 actions=FL000
 cookie=0x0, duration=262.89s, table=0, n packets=1, n bytes=42, idle age=262,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:07,d1 dst=ff:ff:ff:ff:ff:ff,arp spa=10
2.7.70,arp tpa=10.2.8.80,arp op=1 actions=FL000
 cookie=0x0, duration=26.52s, table=0, n packets=1, n bytes=42, idle age=26, ar
p,vlan tci=0x0000,d1 src=00:00:00:00:00:11,d1 dst=00:00:00:00:00:09,arp spa=106
44.82.103,arp tpa=10.3.9.90,arp op=1 actions=FL000
 cookie=0x0, duration=186.54s, table=0, n packets=2, n bytes=84, idle age=16, a
rp,vlan tci=0x0000,d1 src=00:00:00:00:00:10,d1 dst=00:00:00:00:00:11,arp spa=108
24.31.112,arp tpa=108.44.82.103,arp op=2 actions=FL000
 cookie=0x0, duration=176.69s, table=0, n packets=1, n bytes=42, idle age=176,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:02,d1 dst=00:00:00:00:00:09,arp spa=10
1.2.20,arp tpa=10.3.9.90,arp op=2 actions=FL000
 cookie=0x0, duration=623.02s, table=0, n packets=1, n bytes=42, idle age=623,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:01,d1 dst=ff:ff:ff:ff:ff:ff,arp spa=10
1.1.10,arp tpa=10.1.3.30,arp op=1 actions=FL000
 cookie=0x0, duration=382.34s, table=0, n packets=1, n bytes=42, idle age=382,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:09,d1 dst=00:00:00:00:00:05,arp spa=10
3.9.90,arp tpa=10.2.5.50,arp op=2 actions=FL000
 cookie=0x0, duration=136.59s, table=0, n packets=1, n bytes=42, idle age=136,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:07,d1 dst=00:00:00:00:00:10,arp spa=10
2.7.70,arp tpa=108.24.31.112,arp op=2 actions=FL000
 cookie=0x0, duration=517.71s, table=0, n packets=1, n bytes=42, idle age=517,
arp,vlan tci=0x0000,d1 src=00:00:00:00:00:03,d1 dst=00:00:00:00:00:02,arp spa=10
1.3.30,arp tpa=10.1.2.20,arp op=1 actions=FL000
 cookie=0x0, duration=186.55s, table=0, n packets=2, n bytes=84, idle age=16, a
rp,vlan tci=0x0000,d1 src=00:00:00:00:00:11,d1 dst=00:00:00:00:00:10,arp spa=106
44.82.103,arp tpa=108.24.31.112,arp op=1 actions=FL000
 cookie=0x0, duration=612.97s, table=0, n packets=1, n bytes=42, idle age=612, a
rp,vlan tci=0x0000,d1 src=00:00:00:00:00:01,d1 dst=ff:ff:ff:ff:ff:ff,arp spa=10
1.1.10,arp tpa=10.2.6.60,arp op=1 actions=FL000
 cookie=0x0, duration=337.26s, table=0, n packets=1, n bytes=42, idle age=337, a
rp,vlan tci=0x0000,d1 src=00:00:00:00:00:06,d1 dst=00:00:00:00:00:03,arp spa=10
```



- Finally it is the Iperf command. This should show the TCP bandwidth showing that the computers are connected and able to communicate with each other, just not able to send packets due to the firewall. I conducted the iperf command with h10 - all the hosts. This would be the demonstration for department A hosts. I did the iperf command with h5 - all the hosts. This would be my demonstration for department B hosts. Finally I do the TCP bandwidth tests from h_trust to all hosts. This would be my demonstration for h_trust and h_untrust. All of the hosts for department A and department B have TCP bandwidth with every single host in the network. The server however, can only have bandwidth between departments A and B, not with any of the remote hosts. This is shown in the screenshots below. The remote hosts have TCP bandwidth with every host in the network BESIDES the server. This would be correct logic.


```
CE150L [Running]
finalcontroller.py - workspace - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER
finalcontroller.py x lab3controller.py Untitled-1 final.py

OPEN EDITORS
finalcontroller.py... U
lab3controller.py... U
Untitled-1
final.py

mininet@mininet-vm: ~
File Edit Tabs Help

mininet@mininet-vm:~$ sudo ~/pox/pox.py misc.finalcontroller
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[None 1] closed
INFO:openflow.of_01:[00:00:00:00:00:01 7] connected
INFO:openflow.of_01:[00:00:00:00:00:05 3] connected
INFO:openflow.of_01:[00:00:00:00:00:02 5] connected
INFO:openflow.of_01:[00:00:00:00:00:04 2] connected
INFO:openflow.of_01:[00:00:00:00:00:03 4] connected
INFO:openflow.of_01:[00:00:00:00:00:06 6] connected
INFO:openflow.of_01:[00:00:00:00:00:04 2] closed
INFO:openflow.of_01:[00:00:00:00:00:05 3] closed
INFO:openflow.of_01:[00:00:00:00:00:03 4] closed
INFO:openflow.of_01:[00:00:00:00:00:02 5] closed
INFO:openflow.of_01:[00:00:00:00:00:06 6] closed
INFO:openflow.of_01:[00:00:00:00:00:01 7] closed
INFO:openflow.of_01:[None 0] closed
INFO:openflow.of_01:[00:00:00:00:00:06 13] connected
INFO:openflow.of_01:[00:00:00:00:00:02 12] connected
INFO:openflow.of_01:[00:00:00:00:00:01 14] connected
INFO:openflow.of_01:[00:00:00:00:00:03 11] connected
INFO:openflow.of_01:[00:00:00:00:00:04 10] connected
INFO:openflow.of_01:[00:00:00:00:00:05 9] connected

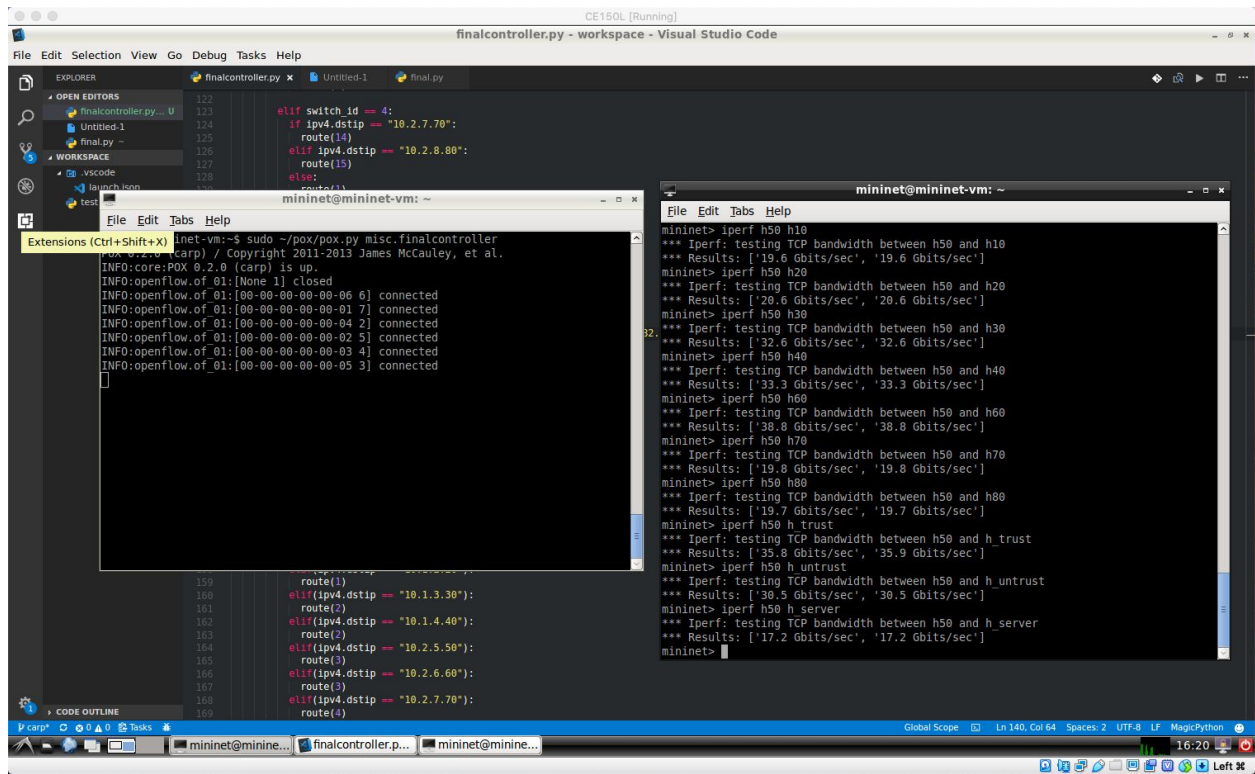
53 # - port on switch: represents the port that the packet was received on.
54 # - switch_id represents the id of the switch that received the packet.
55 # (for example, s1 would have switch_id == 1, s2 would have switch_id == 2, etc...)
56 # You should use these to determine where a packet came from. To figure out where a packet
57 # is going, you can use the IP header information.
58 #print(switch_id)
59 ipv4 = packet.find('ipv4')
60 icmp = packet.find('icmp')
61 tcp = packet.find('tcp')
62 arp = packet.find('arp')
63

mininet@mininet-vm: ~
File Edit Tabs Help

*** Iperf: testing TCP bandwidth between h10 and h40
*** Results: ['20.7 Gbits/sec', '20.7 Gbits/sec']
mininet> iperf h10 h50
*** Iperf: testing TCP bandwidth between h10 and h50
*** Results: ['13.9 Gbits/sec', '13.9 Gbits/sec']
mininet> iperf h10 h60
*** Iperf: testing TCP bandwidth between h10 and h60
*** Results: ['15.8 Gbits/sec', '15.9 Gbits/sec']
mininet> iperf h10 h70
*** Iperf: testing TCP bandwidth between h10 and h70
*** Results: ['20.1 Gbits/sec', '20.1 Gbits/sec']
mininet> iperf h10 h80
*** Iperf: testing TCP bandwidth between h10 and h80
*** Results: ['9.90 Gbits/sec', '9.91 Gbits/sec']
mininet> iperf h10 h_server
*** Iperf: testing TCP bandwidth between h10 and h_server
*** Results: ['21.4 Gbits/sec', '21.4 Gbits/sec']
mininet> iperf h10 h_trust
*** Iperf: testing TCP bandwidth between h10 and h_trust
*** Results: ['15.7 Gbits/sec', '15.8 Gbits/sec']
mininet> iperf h10 h_untrust
*** Iperf: testing TCP bandwidth between h10 and h_untrust
*** Results: ['22.7 Gbits/sec', '22.7 Gbits/sec']
mininet>

mininet@mininet-vm: ~
File Edit Tabs Help

*** Results: ['20.1 Gbits/sec', '20.1 Gbits/sec']
mininet> iperf h10 h80
*** Iperf: testing TCP bandwidth between h10 and h80
*** Results: ['9.90 Gbits/sec', '9.91 Gbits/sec']
mininet> iperf h10 h_server
*** Iperf: testing TCP bandwidth between h10 and h_server
*** Results: ['21.4 Gbits/sec', '21.4 Gbits/sec']
mininet> iperf h10 h_trust
*** Iperf: testing TCP bandwidth between h10 and h_trust
*** Results: ['15.7 Gbits/sec', '15.8 Gbits/sec']
mininet> iperf h10 h_untrust
*** Iperf: testing TCP bandwidth between h10 and h_untrust
*** Results: ['22.7 Gbits/sec', '22.7 Gbits/sec']
mininet> iperf h50 h10
*** Iperf: testing TCP bandwidth between h50 and h10
*** Results: ['19.7 Gbits/sec', '19.7 Gbits/sec']
mininet> iperf h50 h20
*** Iperf: testing TCP bandwidth between h50 and h20
*** Results: ['16.6 Gbits/sec', '16.6 Gbits/sec']
mininet> iperf h50 h30
*** Iperf: testing TCP bandwidth between h50 and h30
*** Results: ['15.4 Gbits/sec', '15.4 Gbits/sec']
mininet> iperf h50 h40
*** Iperf: testing TCP bandwidth between h50 and h40
*** Results: ['21.4 Gbits/sec', '21.5 Gbits/sec']
mininet> iperf h50 h60
*** Iperf: testing TCP bandwidth between h50 and h60
*** Results: ['23.9 Gbits/sec', '23.9 Gbits/sec']
mininet> iperf h50 h70
*** Iperf: testing TCP bandwidth between h50 and h70
*** Results: ['21.4 Gbits/sec', '21.4 Gbits/sec']
mininet> iperf h50 h80
*** Iperf: testing TCP bandwidth between h50 and h80
*** Results: ['22.2 Gbits/sec', '22.3 Gbits/sec']
mininet> iperf h50 h_server
*** Iperf: testing TCP bandwidth between h50 and h_server
*** Results: ['19.4 Gbits/sec', '19.5 Gbits/sec']
mininet> iperf h50 h_trust
*** Iperf: testing TCP bandwidth between h50 and h_trust
*** Results: ['11.6 Gbits/sec', '11.6 Gbits/sec']
mininet> iperf h50 h_untrust
*** Iperf: testing TCP bandwidth between h50 and h_untrust
*** Results: ['18.2 Gbits/sec', '18.2 Gbits/sec']
mininet>
```



```
finalcontroller.py - workspace - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
  finalcontroller.py
  Untitled-1
  final.py
  WORKSPACE
  .vscode
  launch.json
  test

mininet@mininet-vm: ~
File Edit Tabs Help
mininet@mininet-vm:~$ sudo ~/pox/pox.py misc.finalcontroller
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[None 1] closed
INFO:openflow.of_01:[00-00-00-00-00-06 6] connected
INFO:openflow.of_01:[00-00-00-00-00-01 7] connected
INFO:openflow.of_01:[00-00-00-00-00-04 2] connected
INFO:openflow.of_01:[00-00-00-00-00-02 5] connected
INFO:openflow.of_01:[00-00-00-00-00-03 4] connected
INFO:openflow.of_01:[00-00-00-00-00-05 3] connected
in server

159 route(1)
160 elif(ipv4.dstip == "10.1.3.30"):
161     route(2)
162 elif(ipv4.dstip == "10.1.4.40"):
163     route(2)
164 elif(ipv4.dstip == "10.2.5.50"):
165     route(3)
166 elif(ipv4.dstip == "10.2.6.60"):
167     route(3)
168 elif(ipv4.dstip == "10.2.7.70"):
169     route(4)

mininet@mininet-vm: ~
File Edit Tabs Help
mininet@mininet-vm:~$ iperf h trust h30
*** Iperf: testing TCP bandwidth between h trust and h30
*** Results: ['20.9 Gbits/sec', '21.0 Gbits/sec']
mininet> iperf h trust h40
*** Iperf: testing TCP bandwidth between h trust and h40
*** Results: ['23.5 Gbits/sec', '23.5 Gbits/sec']
mininet> iperf h trust h50
*** Iperf: testing TCP bandwidth between h trust and h50
*** Results: ['32.8 Gbits/sec', '32.9 Gbits/sec']
mininet> iperf h trust h60
*** Iperf: testing TCP bandwidth between h trust and h60
*** Results: ['34.0 Gbits/sec', '34.0 Gbits/sec']
mininet> iperf h trust h70
*** Iperf: testing TCP bandwidth between h trust and h70
*** Results: ['33.0 Gbits/sec', '33.0 Gbits/sec']
mininet> iperf h trust h80
*** Iperf: testing TCP bandwidth between h trust and h80
*** Results: ['31.0 Gbits/sec', '31.7 Gbits/sec']
mininet> iperf h trust h_untrust
*** Iperf: testing TCP bandwidth between h trust and h_untrust
*** Results: ['35.0 Gbits/sec', '35.0 Gbits/sec']
mininet> iperf h trust h_server
*** Iperf: testing TCP bandwidth between h trust and h_server
^C
Interrupt
mininet> iperf h10 h trust
*** Iperf: testing TCP bandwidth between h10 and h trust
*** Results: ['32.1 Gbits/sec', '32.2 Gbits/sec']
mininet> iperf h10 h_untrust
*** Iperf: testing TCP bandwidth between h10 and h_untrust
*** Results: ['35.5 Gbits/sec', '35.6 Gbits/sec']
mininet>
```

```
finalcontroller.py - workspace - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
  finalcontroller.py
  Untitled-1
  final.py
  WORKSPACE
  .vscode
  launch.json
  test

mininet@mininet-vm: ~
File Edit Tabs Help
mininet@mininet-vm:~$ sudo ~/pox/pox.py misc.finalcontroller
POX 0.2.0 (carp) / Copyright 2011-2013 James McCauley, et al.
INFO:core:POX 0.2.0 (carp) is up.
INFO:openflow.of_01:[None 1] closed
INFO:openflow.of_01:[00-00-00-00-00-06 6] connected
INFO:openflow.of_01:[00-00-00-00-00-01 7] connected
INFO:openflow.of_01:[00-00-00-00-00-04 2] connected
INFO:openflow.of_01:[00-00-00-00-00-02 5] connected
INFO:openflow.of_01:[00-00-00-00-00-03 4] connected
INFO:openflow.of_01:[00-00-00-00-00-05 3] connected
in server

159 route(1)
160 elif(ipv4.dstip == "10.1.3.30"):
161     route(2)
162 elif(ipv4.dstip == "10.1.4.40"):
163     route(2)
164 elif(ipv4.dstip == "10.2.5.50"):
165     route(3)
166 elif(ipv4.dstip == "10.2.6.60"):
167     route(3)
168 elif(ipv4.dstip == "10.2.7.70"):
169     route(4)

mininet@mininet-vm: ~
File Edit Tabs Help
mininet> iperf h trust h10
*** Iperf: testing TCP bandwidth between h trust and h10
*** Results: ['21.0 Gbits/sec', '21.0 Gbits/sec']
mininet> iperf h trust h20
*** Iperf: testing TCP bandwidth between h trust and h20
*** Results: ['22.1 Gbits/sec', '22.1 Gbits/sec']
mininet> iperf h trust h30
*** Iperf: testing TCP bandwidth between h trust and h30
*** Results: ['20.9 Gbits/sec', '21.0 Gbits/sec']
mininet> iperf h trust h40
*** Iperf: testing TCP bandwidth between h trust and h40
*** Results: ['23.5 Gbits/sec', '23.5 Gbits/sec']
mininet> iperf h trust h50
*** Iperf: testing TCP bandwidth between h trust and h50
*** Results: ['32.8 Gbits/sec', '32.9 Gbits/sec']
mininet> iperf h trust h60
*** Iperf: testing TCP bandwidth between h trust and h60
*** Results: ['34.0 Gbits/sec', '34.0 Gbits/sec']
mininet> iperf h trust h70
*** Iperf: testing TCP bandwidth between h trust and h70
*** Results: ['33.0 Gbits/sec', '33.0 Gbits/sec']
mininet> iperf h trust h80
*** Iperf: testing TCP bandwidth between h trust and h80
*** Results: ['31.0 Gbits/sec', '31.7 Gbits/sec']
mininet> iperf h trust h_untrust
*** Iperf: testing TCP bandwidth between h trust and h_untrust
*** Results: ['35.0 Gbits/sec', '35.0 Gbits/sec']
mininet> iperf h trust h_server
*** Iperf: testing TCP bandwidth between h trust and h_server
^C
Interrupt
```

You are more than welcome to test any of my code and outputs. I did not include everything because the outputs were too large or because they were redundant. Please test accordingly.