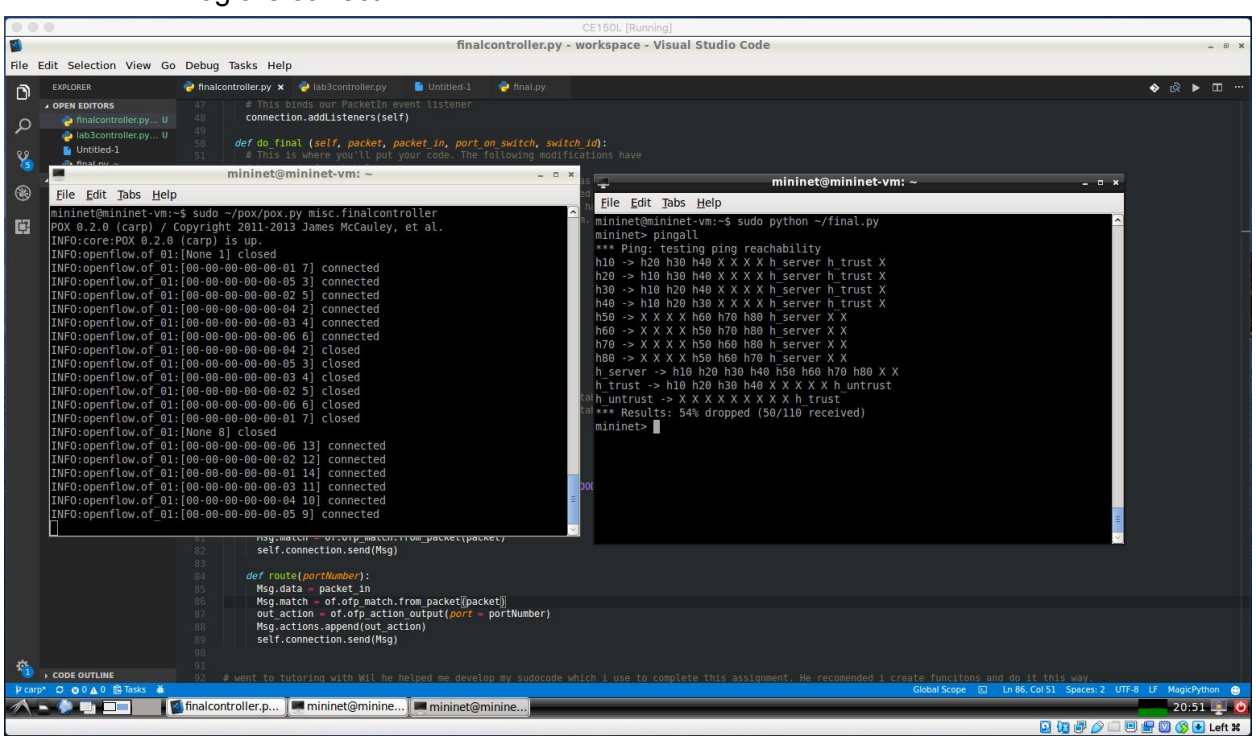
Tyler Soriano
CSE 150
Final Project

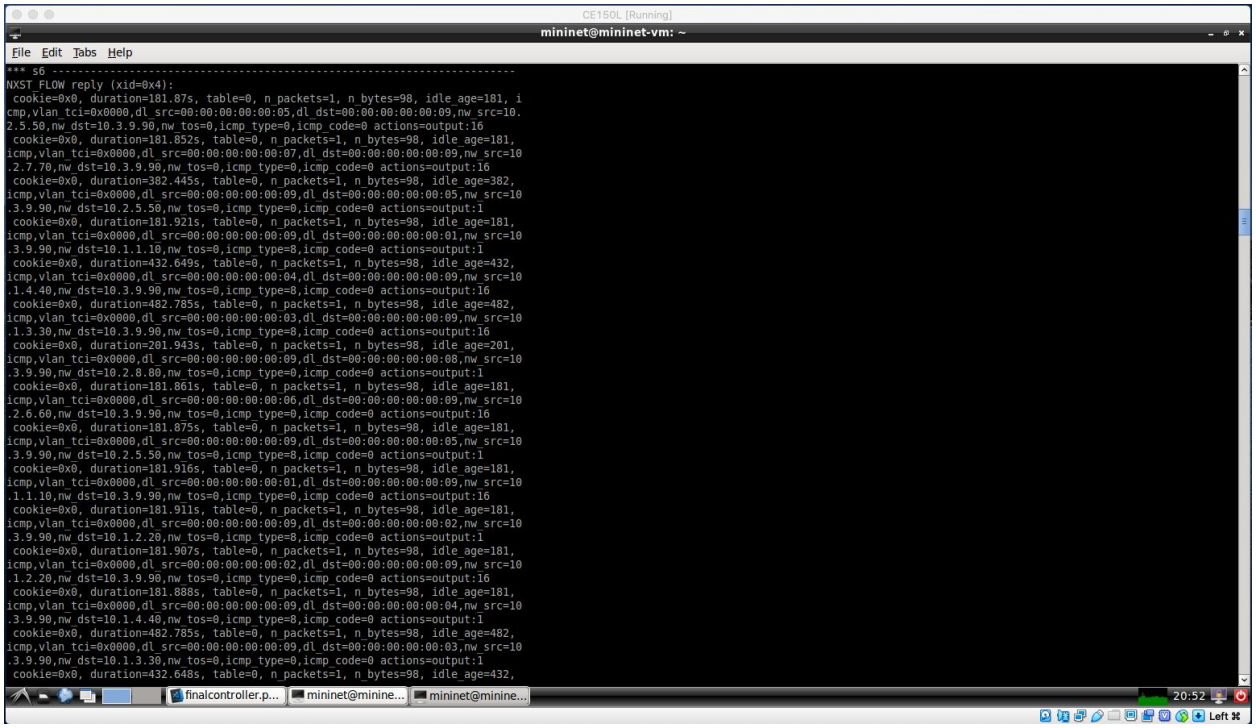
1. First we run the pingall command. This is to ensure that all ICMP packets are going through to the correct port and none are breaking the rules set at the table. For reference this is the logic I followed:

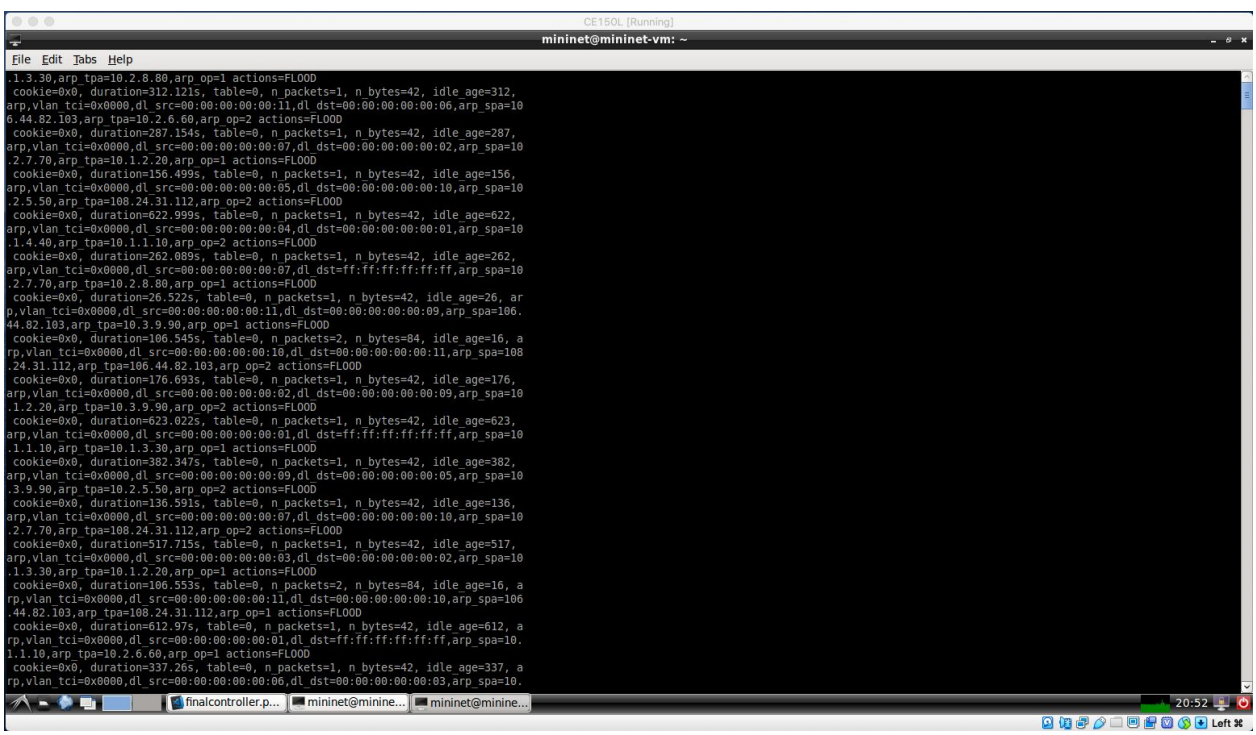
Device	Mininet Name	IP Address	Description
Floor 1 Hosts	h10, h20, h30, h40	10.1.1.10/24 10.1.2.20/24 10.1.3.30/24 10.1.4.40/24	Computers on floor 1 of the Department A in the company.
Floor 2 Hosts	h50, h60, h70, h80	10.2.5.50/24 10.2.6.60/24 10.2.7.70/24 10.2.8.80/24	Computers on floor 2 of the Department 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.

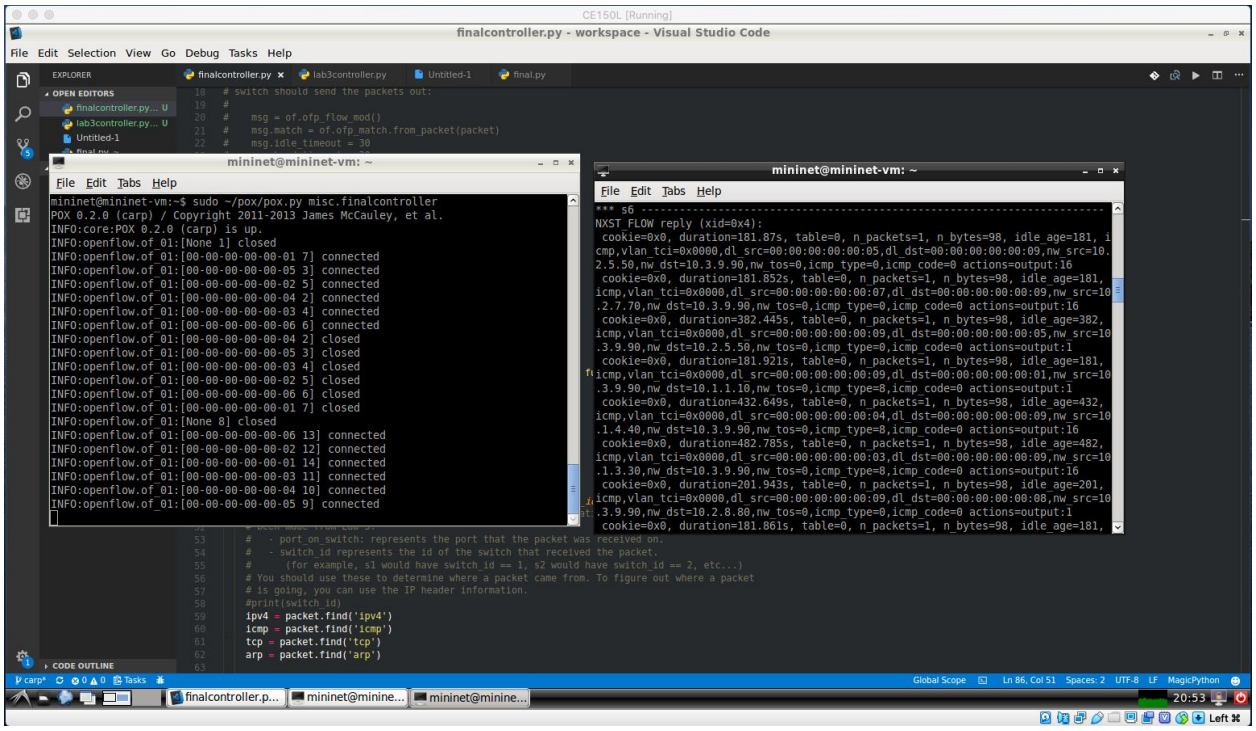
   a. These are the results of the pingall command: this should show you that all of the logic is correct.



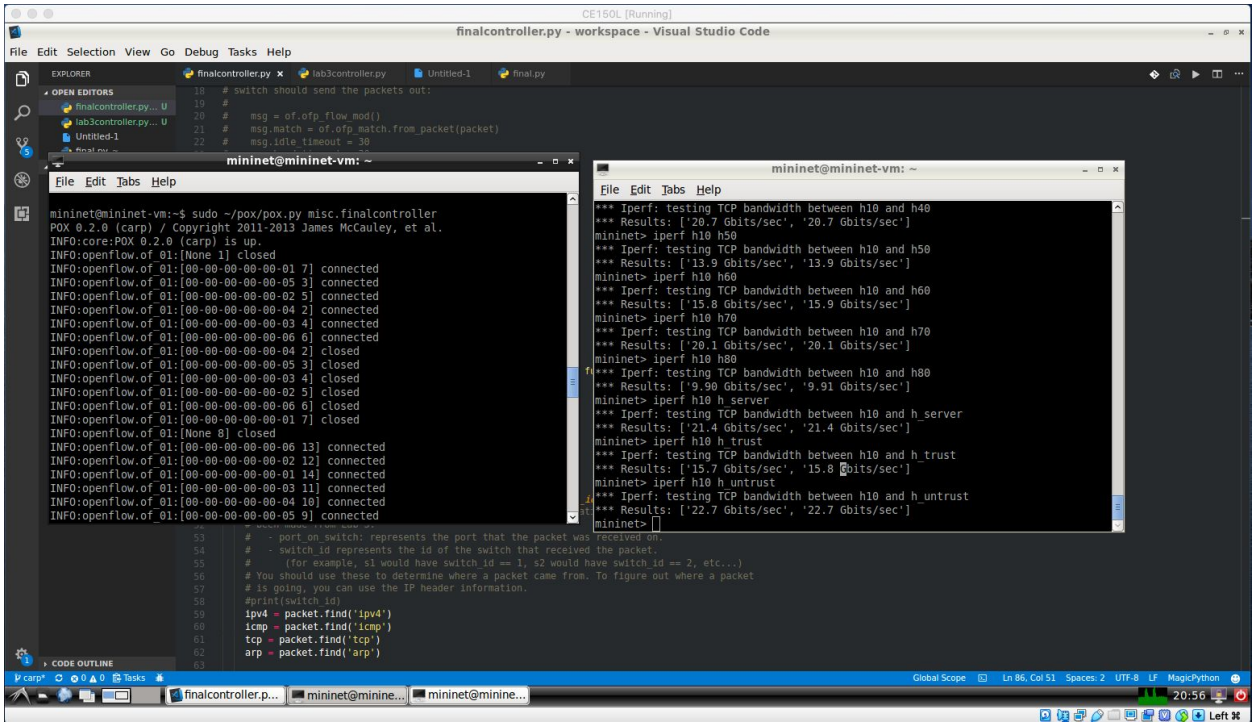
2. Next is the dpctl dump-flows. This should show you exactly which ports the packets are traveling through. Also should tell you how each type of packet is moving. ARP is flooded while all IPV4 packets (ICMP and TCP) are being directed through to specific ports. I have provided multiple screenshots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

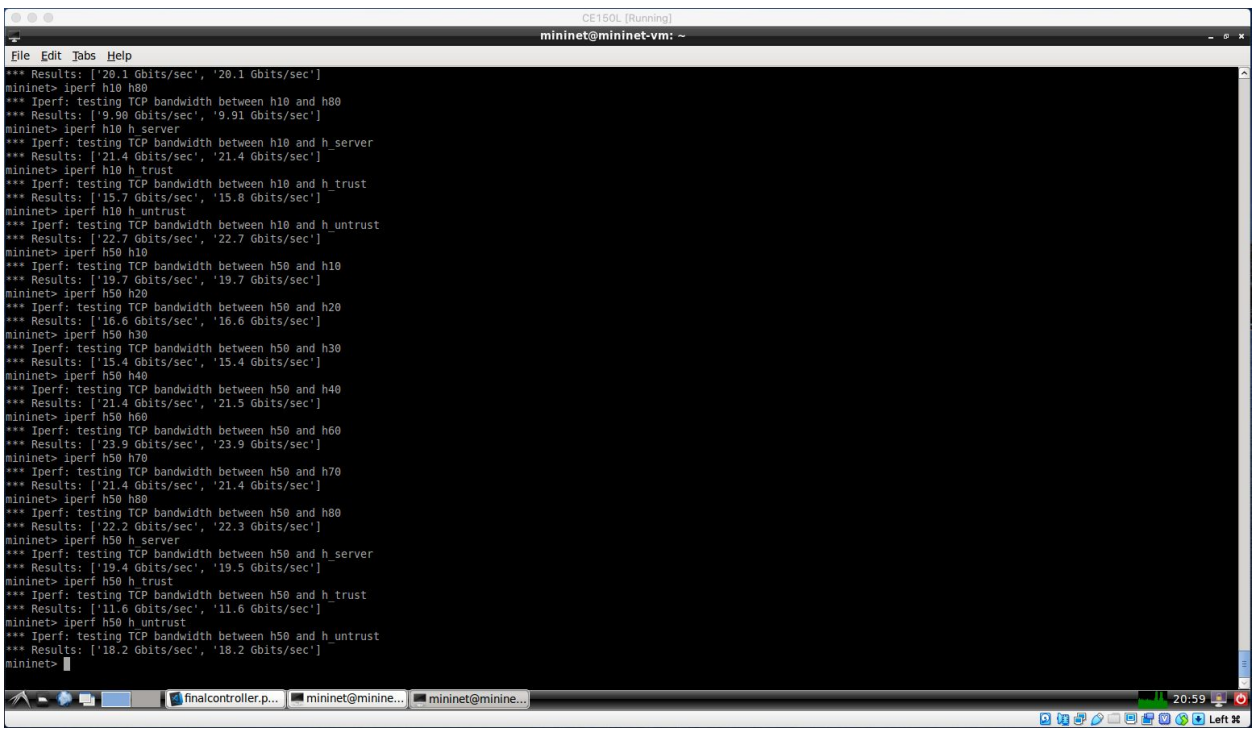


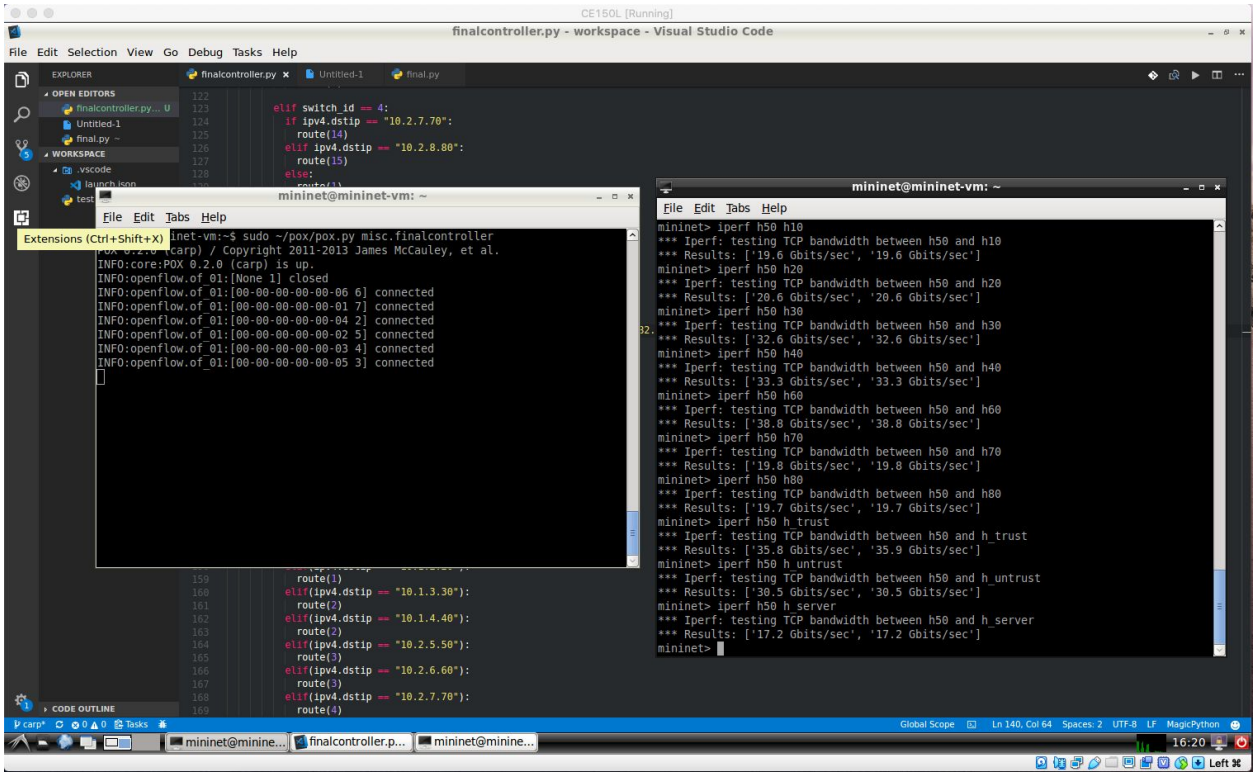


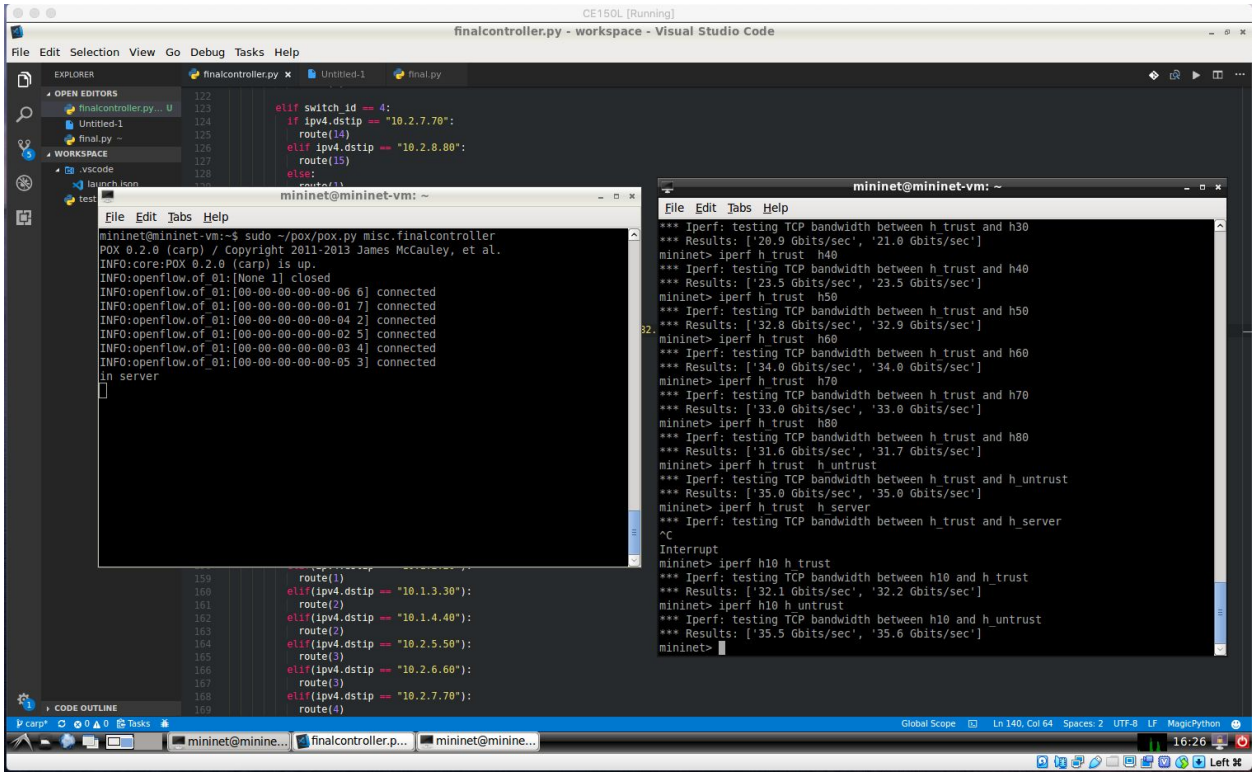


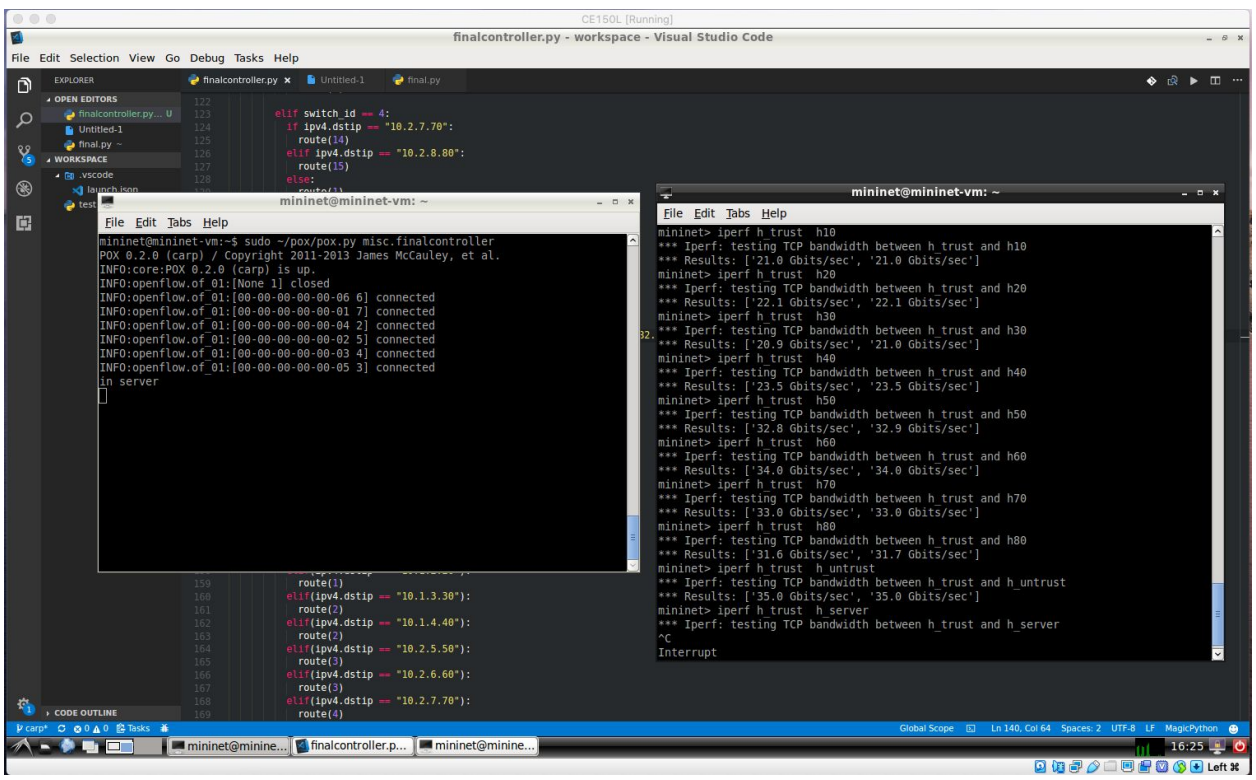
3. Finally it is the Iperf command. This should show the TCP bandwidth showing that the computers are connected and able to communicate with each other, just not able to send packets due to the firewall. I conducted the iperf command with h10 - all the hosts. This would be the demonstration for department 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
OPEN EDITORS
finalcontroller.py... U
lab3controller.py... U
Untitled-1
finalcontroller.py
lab3controller.py
Untitled-1
final.py
switch should send the packets out:
#
msg = of.ofp_flow_mod()
msg.match = of.ofp_match.from_packet(packet)
msg.idle_timeout = 30
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 8] 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
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>
- port_on_switch: represents the port that the packet was received on.
- switch_id represents the id of the switch that received the packet.
(for example, s1 would have switch_id == 1, s2 would have switch_id == 2, etc...)
You should use these to determine where a packet came from. To figure out where a packet
is going, you can use the IP header information.
#print(switch_id)
ipv4 = packet.find('ipv4')
icmp = packet.find('icmp')
tcp = packet.find('tcp')
arp = packet.find('arp')
CODE OUTLINE
carp*
Global Scope
Ln 86, Col 51
Spaces: 2
UTF-8
LF
MagicPython
finalcontroller.p...
mininet@minine...
mininet@minine...
20:56
Left ⌘


CE150L [Running]
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.p...
mininet@minine...
mininet@minine...
20:59
Left ⌘


CE150L [Running]
finalcontroller.py - workspace - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
finalcontroller.py... U
Untitled-1
final.py ~
WORKSPACE
.vscode
launch.json
test
Extensions (Ctrl+Shift+X)
CODE OUTLINE
finalcontroller.py
Untitled-1
final.py
elif switch_id == 4:
if ipv4.dstip == "10.2.7.70":
route(14)
elif ipv4.dstip == "10.2.8.80":
route(15)
else:
route(1)
elif(ipv4.dstip == "10.1.3.30"):
route(2)
elif(ipv4.dstip == "10.1.4.40"):
route(2)
elif(ipv4.dstip == "10.2.5.50"):
route(3)
elif(ipv4.dstip == "10.2.6.60"):
route(3)
elif(ipv4.dstip == "10.2.7.70"):
route(4)
mininet@mininet-vm: ~
File Edit Tabs Help
inet-vm:~$ sudo ~/pox/pox.py misc.finalcontroller
(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
mininet@mininet-vm: ~
File Edit Tabs Help
mininet> iperf h50 h10
*** Iperf: testing TCP bandwidth between h50 and h10
*** Results: ['19.6 Gbits/sec', '19.6 Gbits/sec']
mininet> iperf h50 h20
*** Iperf: testing TCP bandwidth between h50 and h20
*** Results: ['20.6 Gbits/sec', '20.6 Gbits/sec']
mininet> iperf h50 h30
*** Iperf: testing TCP bandwidth between h50 and h30
*** Results: ['32.6 Gbits/sec', '32.6 Gbits/sec']
mininet> iperf h50 h40
*** Iperf: testing TCP bandwidth between h50 and h40
*** Results: ['33.3 Gbits/sec', '33.3 Gbits/sec']
mininet> iperf h50 h60
*** Iperf: testing TCP bandwidth between h50 and h60
*** Results: ['38.8 Gbits/sec', '38.8 Gbits/sec']
mininet> iperf h50 h70
*** Iperf: testing TCP bandwidth between h50 and h70
*** Results: ['19.8 Gbits/sec', '19.8 Gbits/sec']
mininet> iperf h50 h80
*** Iperf: testing TCP bandwidth between h50 and h80
*** Results: ['19.7 Gbits/sec', '19.7 Gbits/sec']
mininet> iperf h50 h_trust
*** Iperf: testing TCP bandwidth between h50 and h_trust
*** Results: ['35.8 Gbits/sec', '35.9 Gbits/sec']
mininet> iperf h50 h_untrust
*** Iperf: testing TCP bandwidth between h50 and h_untrust
*** Results: ['30.5 Gbits/sec', '30.5 Gbits/sec']
mininet> iperf h50 h_server
*** Iperf: testing TCP bandwidth between h50 and h_server
*** Results: ['17.2 Gbits/sec', '17.2 Gbits/sec']
mininet>
carp* 0 0 Tasks
Global Scope Ln 140, Col 64 Spaces: 2 UTF-8 LF MagicPython
mininet@minine... finalcontroller.p... mininet@minine...
16:20
Left ⌘





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.