

Darren Lee
Final Project
CSE150

dump

```
mininet> dump
<Host h10: h10-eth0:10.1.1.10 pid=3618>
<Host h20: h20-eth0:10.1.2.20 pid=3620>
<Host h30: h30-eth0:10.1.3.30 pid=3622>
<Host h40: h40-eth0:10.1.4.40 pid=3624>
<Host h50: h50-eth0:10.2.5.50 pid=3626>
<Host h60: h60-eth0:10.2.6.60 pid=3628>
<Host h70: h70-eth0:10.2.7.70 pid=3630>
<Host h80: h80-eth0:10.2.8.80 pid=3632>
<Host h_server: h_server-eth0:10.3.9.90 pid=3634>
<Host h_trust: h_trust-eth0:108.24.31.112 pid=3636>
<Host h_untrust: h_untrust-eth0:106.44.82.103 pid=3638>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None,s1-eth4:None,
s1-eth5:None,s1-eth6:None,s1-eth7:None pid=3643>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth3:None,s2-eth4:None pid=3646>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth3:None,s3-eth4:None pid=3649>
<OVSSwitch s4: lo:127.0.0.1,s4-eth1:None,s4-eth3:None,s4-eth4:None pid=3652>
<OVSSwitch s5: lo:127.0.0.1,s5-eth1:None,s5-eth3:None,s5-eth4:None pid=3655>
<OVSSwitch s6: lo:127.0.0.1,s6-eth1:None,s6-eth2:None pid=3658>
<RemoteController c0: 127.0.0.1:6633 pid=3612>
mininet>
```

Dump is a command to dump all the information about all nodes and shows that the devices were created successfully. It shows that all the hosts and switches are set up correctly in the mininet topology for final_skel.py. We can see the IP addresses and pids for hosts h10 through h_untrust and that the IP addresses are correct according to the lab specification. The IP address of the localhost and the ethernet are given too. Also listed are the pids for switch s1 through switch s6 and controller c0. The port for the controller is 6633.

links

```
mininet> links
s1-eth6<->h_trust-eth0 (OK OK)
s1-eth7<->h_untrust-eth0 (OK OK)
s1-eth2<->s2-eth1 (OK OK)
s1-eth3<->s3-eth1 (OK OK)
s1-eth4<->s4-eth1 (OK OK)
s1-eth5<->s5-eth1 (OK OK)
s1-eth1<->s6-eth1 (OK OK)
s2-eth3<->h10-eth0 (OK OK)
s2-eth4<->h20-eth0 (OK OK)
s3-eth3<->h30-eth0 (OK OK)
s3-eth4<->h40-eth0 (OK OK)
s4-eth3<->h50-eth0 (OK OK)
s4-eth4<->h60-eth0 (OK OK)
s5-eth3<->h70-eth0 (OK OK)
s5-eth4<->h80-eth0 (OK OK)
s6-eth2<->h_server-eth0 (OK OK)
mininet>
```

The links command shows the links in the mininet topology. This screenshot shows that the topology is correctly set up because s1 is the core switch, s2 through s5 are the floor switches, and s6 is the data center switch. Then you can see that the switches are connected to the hosts that they are supposed to be as listed in the lab specification.

ICMP protocols

pingall

```
mininet> pingall
*** Ping: testing ping reachability
h10 -> h20 h30 h40 X X X X h_server h_trust X
h20 -> h10 h30 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 X X
h_untrust -> X X X X X X X X X X
*** Results: 56% dropped (48/110 received)
mininet>
```

I used pingall to check the ICMP traffic between each host. Pings that succeed show a correct connection using ICMP. Department A is unable to send ICMP messages to Department B. Department B cannot send ICMP packets to Department A either. The trusted host can only

send ICMP messages to Department A. the untrusted host cannot send packets to anyone because it cannot send IP or ICMP packets anywhere. The external hosts outside our networks h_trust and h_untrust cannot receive a ping from each other because IP traffic is blocked. The untrusted host cannot send ICMP traffic to hosts 10 through 80. The untrusted and trusted hosts cannot send any traffic to the server. The trusted host cannot send ICMP traffic to hosts 50 through 80. Hosts 10 through 40 cannot send ICMP traffic to hosts 50 through 80.

Flow table

dpctl dump-flows

```
mininet> dpctl dump-flows
*** s1 -----
NXST_FLOW reply (xid=0x4):
  cookie=0x0, duration=22.921s, table=0, n_packets=1, n_bytes=98, idle_timeout=30, hard_timeout=60, idle_age=22, pr
os=0,icmp_type=8,icmp_code=0 actions=drop
  cookie=0x0, duration=12.965s, table=0, n_packets=1, n_bytes=98, idle_timeout=30, hard_timeout=60, idle_age=12, pr
nw_tos=0,icmp_type=8,icmp_code=0 actions=drop
  cookie=0x0, duration=17.925s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
op=1 actions=FL00D
  cookie=0x0, duration=17.918s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
op=2 actions=FL00D
  cookie=0x0, duration=7.9s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, priori
op=2 actions=FL00D
  cookie=0x0, duration=27.919s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
op=1 actions=FL00D
  cookie=0x0, duration=27.911s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
op=2 actions=FL00D
  cookie=0x0, duration=7.92s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prior
p_op=1 actions=FL00D
*** s2 -----
NXST_FLOW reply (xid=0x4):
  cookie=0x0, duration=17.929s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
op=1 actions=FL00D
  cookie=0x0, duration=17.92s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pri
op=2 actions=FL00D
  cookie=0x0, duration=7.902s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
rp_op=2 actions=FL00D
  cookie=0x0, duration=27.921s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
op=1 actions=FL00D
  cookie=0x0, duration=27.914s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
op=2 actions=FL00D
  cookie=0x0, duration=7.906s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
rp_op=1 actions=FL00D
*** s3 -----
```

```

*** s3 -----
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=17.933s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=1 actions=FL00D
 cookie=0x0, duration=17.924s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.907s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=2 actions=FL00D
 cookie=0x0, duration=27.926s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=1 actions=FL00D
 cookie=0x0, duration=27.919s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.916s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=1 actions=FL00D
*** s4 -----
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=17.938s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=1 actions=FL00D
 cookie=0x0, duration=17.929s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.911s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=2 actions=FL00D
 cookie=0x0, duration=27.929s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=1 actions=FL00D
 cookie=0x0, duration=27.923s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.913s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=1 actions=FL00D
*** s5 -----
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=17.942s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=1 actions=FL00D
 cookie=0x0, duration=17.933s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.917s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=2 actions=FL00D
 cookie=0x0, duration=27.935s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=1 actions=FL00D
 cookie=0x0, duration=27.931s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.925s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=1 actions=FL00D
*** s6 -----
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=17.946s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=1 actions=FL00D
 cookie=0x0, duration=17.942s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=17, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.92s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prior
 p_op=2 actions=FL00D
 cookie=0x0, duration=27.938s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=1 actions=FL00D
 cookie=0x0, duration=27.931s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=27, pr
 op=2 actions=FL00D
 cookie=0x0, duration=7.924s, table=0, n_packets=1, n_bytes=42, idle_timeout=30, hard_timeout=60, idle_age=7, prio
 rp_op=1 actions=FL00D
mininet>

```

After running pingall, I ran the `dpctl dump-flows` command to see the installed flow mods in the table of each switch according to the rules specified in the lab document. For example, ICMP from the hosts in Department A (Host 10, 20, 30, 40) to the hosts in Department B (Host 50, 60, 70, 80) is blocked and vice versa. These table entries were installed to the switches when the switches got the first packets. The switches don't know how to process the first packets they receive, which is why these table entries need to be installed on the switches.

IP
Iperf

```
mininet> iperf h10 h40
*** Iperf: testing TCP bandwidth between h10 and h40
*** Results: ['29.5 Gbits/sec', '29.5 Gbits/sec']
mininet> iperf h20 h50
*** Iperf: testing TCP bandwidth between h20 and h50
*** Results: ['30.3 Gbits/sec', '30.3 Gbits/sec']
mininet> iperf h20 h80
*** Iperf: testing TCP bandwidth between h20 and h80
*** Results: ['34.5 Gbits/sec', '34.6 Gbits/sec']
mininet> iperf h20 h_trust
*** Iperf: testing TCP bandwidth between h20 and h_trust
*** Results: ['39.0 Gbits/sec', '39.0 Gbits/sec']
mininet> iperf h40 h_trust
*** Iperf: testing TCP bandwidth between h40 and h_trust
*** Results: ['43.3 Gbits/sec', '43.4 Gbits/sec']
mininet> iperf h50 h_trust
*** Iperf: testing TCP bandwidth between h50 and h_trust
*** Results: ['40.6 Gbits/sec', '41.2 Gbits/sec']
mininet> █
```

I used iperf to check the IP protocol access that was denied by the pingall command due to ICMP traffic control. Iperf can create a connection between Department A and B nodes. The trusted host can send TCP packets to both departments rather than just department A. Iperf uses TCP and TCP packets are allowed through the firewall in the channels shown above, so iperf succeeds.

```
mininet> iperf
*** Iperf: testing TCP bandwidth between h10 and h_untrust
*** Results: ['46.8 Gbits/sec', '46.8 Gbits/sec']
mininet> iperf h20 h_untrust
*** Iperf: testing TCP bandwidth between h20 and h_untrust
*** Results: ['33.2 Gbits/sec', '33.5 Gbits/sec']
mininet> iperf h50 h_untrust
*** Iperf: testing TCP bandwidth between h50 and h_untrust
*** Results: ['38.3 Gbits/sec', '38.4 Gbits/sec']
mininet> iperf h30 h_untrust
*** Iperf: testing TCP bandwidth between h30 and h_untrust
*** Results: ['40.2 Gbits/sec', '40.2 Gbits/sec']
mininet> iperf h70 h_untrust
*** Iperf: testing TCP bandwidth between h70 and h_untrust
*** Results: ['45.6 Gbits/sec', '45.7 Gbits/sec']
mininet> iperf h_trust h_untrust
*** Iperf: testing TCP bandwidth between h_trust and h_untrust
*** Results: ['50.8 Gbits/sec', '50.8 Gbits/sec']
mininet> iperf h_server h_untrust
```

The untrusted host is allowed to send TCP traffic to the hosts even though ICMP traffic from the untrusted host to the other hosts is blocked by the firewall.