

Lab 1

2. [30 pts] Save a screenshot of dump and pingall output. Explain what is being shown in the screenshot.

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=1826>
<Host h2: h2-eth0:10.0.0.2 pid=1830>
<Host h3: h3-eth0:10.0.0.3 pid=1832>
<Host h4: h4-eth0:10.0.0.4 pid=1834>
<Host h5: h5-eth0:10.0.0.5 pid=1836>
<Host h6: h6-eth0:10.0.0.6 pid=1838>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=1843>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=1846>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None,s3-eth4:None
pid=1849>
<Controller c0: 127.0.0.1:6633 pid=1819>

mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6
h2 -> h1 h3 h4 h5 h6
h3 -> h1 h2 h4 h5 h6
h4 -> h1 h2 h3 h5 h6
h5 -> h1 h2 h3 h4 h6
h6 -> h1 h2 h3 h4 h5
*** Results: 0% dropped (30/30 received)
mininet>
```

Dump - shows all information about the nodes. It shows the connections between hosts nodes and switches. It also shows the ip addresses that have to do with the nodes themselves. Like I said it dumps all the information about the nodes.

pingall - pings all the connections between nodes. This shows if the connections are done sufficiently and in my case no packets were dropped which reflects the connections are sufficient.

3. [10 pts] Run the iperf command as well, and screenshot the output, how fast is the connect?

```
mininet> iperf
*** Iperf: testing TCP bandwidth between h1 and h6
*** Results: ['31.5 Gbits/sec', '31.5 Gbits/sec']
mininet>
```

The connect is 31.5 gbits a second

4. Run wireshark, and using the display filter, filter for “of”. Note: When you run wireshark you should do so as “sudo wireshark”. When you choose an interface to capture on, you should select “any”.

a. [20 pts] Run ping from a host to any other host using hX ping -c 5 hY. How many of _packet_in messages show up? Take a screenshot of your results.

6498	3166.736501	127.0.0.1	127.0.0.1	OF 1.0	76 of echo reply
6501	3167.478373	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6502	3167.478605	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out
6508	3167.481077	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6509	3167.481354	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out
6516	3167.481602	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6517	3167.481838	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out
6523	3167.482046	10.0.0.2	10.0.0.1	OF 1.0	184 of packet in
6524	3167.482334	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6527	3167.482682	10.0.0.2	10.0.0.1	OF 1.0	184 of packet in
6528	3167.483080	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6531	3167.483321	10.0.0.2	10.0.0.1	OF 1.0	184 of packet in
6532	3167.483772	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6538	3168.480010	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6539	3168.480282	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6543	3168.480793	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6544	3168.481026	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6548	3168.481278	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6549	3168.481438	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6595	3172.497115	02:84:5c:bd:c7:18	d6:b6:f8:9d:f2:f4	OF 1.0	128 of packet in
6596	3172.497542	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6600	3172.497890	02:84:5c:bd:c7:18	d6:b6:f8:9d:f2:f4	OF 1.0	128 of packet in
6601	3172.498294	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6605	3172.498558	02:84:5c:bd:c7:18	d6:b6:f8:9d:f2:f4	OF 1.0	128 of packet in
6606	3172.498874	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6610	3172.499162	d6:b6:f8:9d:f2:f4	02:84:5c:bd:c7:18	OF 1.0	128 of packet in
6611	3172.499444	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6614	3172.499746	d6:b6:f8:9d:f2:f4	02:84:5c:bd:c7:18	OF 1.0	128 of packet in
6615	3172.500046	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add
6618	3172.502799	d6:b6:f8:9d:f2:f4	02:84:5c:bd:c7:18	OF 1.0	128 of packet in
6619	3172.503046	127.0.0.1	127.0.0.1	OF 1.0	148 of flow add

```
mininet> h1 ping -c 5 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=5.67 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=2.34 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.861 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=0.041 ms
64 bytes from 10.0.0.2: icmp_seq=5 ttl=64 time=0.074 ms

--- 10.0.0.2 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 0.041/1.800/5.679/2.111 ms
mininet>
```

There were 15 packets of _packet_in

b. [20 pts] What is the source and destination IP addresses for these entries? Find another packet that matches the “of” filter with the OpenFlow typefield set to OFPT_PACKET_OUT. What is the source and destination IP address for this entry? Take screenshots showing your results.

Source and destination for of_packet_in =
Source - 10.0.0.2

Destination - 10.0.0.2, 10.0.0.1

Source and destination for of_packet_of
127.0.0.1

6502	3167.478605	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out
6508	3167.481077	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6509	3167.481354	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out
6516	3167.481602	10.0.0.1	10.0.0.2	OF 1.0	184 of packet in
6517	3167.481838	127.0.0.1	127.0.0.1	OF 1.0	92 of packet out

c. [20 pts] Replace the display filter for “of” to “icmp && not of”. Run pingall again, how many entries are generated in wireshark? What types of icmp entries show up? Take a screenshot of your results.

260	4.743786000	10.0.0.1	10.0.0.6	ICMP	100 Echo (ping) reply	id=0x0e0d, seq=1/256, ttl=64 (request in 259)
261	4.743887000	10.0.0.1	10.0.0.6	ICMP	100 Echo (ping) reply	id=0x0e0d, seq=1/256, ttl=64
262	4.743888000	10.0.0.1	10.0.0.6	ICMP	100 Echo (ping) reply	id=0x0e0d, seq=1/256, ttl=64
263	4.743826000	10.0.0.1	10.0.0.6	ICMP	100 Echo (ping) repl	
264	4.745278000	10.0.0.6	10.0.0.2	ICMP	100 Echo (ping) requ	
265	4.745389000	10.0.0.6	10.0.0.2	ICMP	100 Echo (ping) requ	
266	4.745393000	10.0.0.6	10.0.0.2	ICMP	100 Echo (ping) requ	
267	4.745419000	10.0.0.6	10.0.0.2	ICMP	100 Echo (ping) requ	
268	4.745427000	10.0.0.2	10.0.0.6	ICMP	100 Echo (ping) repl	
269	4.745448000	10.0.0.2	10.0.0.6	ICMP	100 Echo (ping) repl	*** Results: 0% dropped (30/30 received)
270	4.745448000	10.0.0.2	10.0.0.6	ICMP	100 Echo (ping) repl	mininet> pingall
271	4.745467000	10.0.0.2	10.0.0.6	ICMP	100 Echo (ping) repl	*** Ping: testing ping reachability
272	4.746877000	10.0.0.6	10.0.0.3	ICMP	100 Echo (ping) requ	h1 -> h2 h3 h4 h5 h6
274	4.746984000	10.0.0.6	10.0.0.3	ICMP	100 Echo (ping) requ	h2 -> h1 h2 h3 h4 h5 h6
275	4.746986000	10.0.0.6	10.0.0.3	ICMP	100 Echo (ping) requ	h3 -> h1 h2 h3 h4 h5 h6
276	4.747030000	10.0.0.6	10.0.0.3	ICMP	100 Echo (ping) requ	h4 -> h1 h2 h3 h4 h5 h6
277	4.747037000	10.0.0.3	10.0.0.6	ICMP	100 Echo (ping) repl	h5 -> h1 h2 h3 h4 h5 h6
278	4.747058000	10.0.0.3	10.0.0.6	ICMP	100 Echo (ping) repl	h6 -> h1 h2 h3 h4 h5 h6
279	4.747058000	10.0.0.3	10.0.0.6	ICMP	100 Echo (ping) repl	*** Results: 0% dropped (30/30 received)
280	4.747077000	10.0.0.3	10.0.0.6	ICMP	100 Echo (ping) repl	mininet> pingall
281	4.750500000	10.0.0.6	10.0.0.4	ICMP	100 Echo (ping) requ	*** Ping: testing ping reachability
282	4.750646000	10.0.0.6	10.0.0.4	ICMP	100 Echo (ping) requ	h1 -> h2 h3 h4 h5 h6
283	4.750649000	10.0.0.6	10.0.0.4	ICMP	100 Echo (ping) requ	h2 -> h1 h2 h3 h4 h5 h6
284	4.750677000	10.0.0.6	10.0.0.4	ICMP	100 Echo (ping) requ	h3 -> h1 h2 h3 h4 h5 h6
285	4.750686000	10.0.0.4	10.0.0.6	ICMP	100 Echo (ping) requ	h4 -> h1 h2 h3 h4 h5 h6
286	4.750761000	10.0.0.4	10.0.0.6	ICMP	100 Echo (ping) repl	h5 -> h1 h2 h3 h4 h5 h6
287	4.750762000	10.0.0.4	10.0.0.6	ICMP	100 Echo (ping) repl	h6 -> h1 h2 h3 h4 h5 h6
288	4.750783000	10.0.0.4	10.0.0.6	ICMP	100 Echo (ping) repl	*** Results: 0% dropped (30/30 received)
289	4.753060000	10.0.0.6	10.0.0.5	ICMP	100 Echo (ping) request	mininet>
290	4.753221000	10.0.0.6	10.0.0.5	ICMP	100 Echo (ping) request	id=0x0e11, seq=1/256, ttl=64
291	4.753232000	10.0.0.5	10.0.0.6	ICMP	100 Echo (ping) reply	id=0x0e11, seq=1/256, ttl=64 (reply in 291)
292	4.753261000	10.0.0.5	10.0.0.6	ICMP	100 Echo (ping) reply	id=0x0e11, seq=1/256, ttl=64 (request in 290)

292 entries are made with pingall

Types: Echo (ping) reply

Echo (ping) request