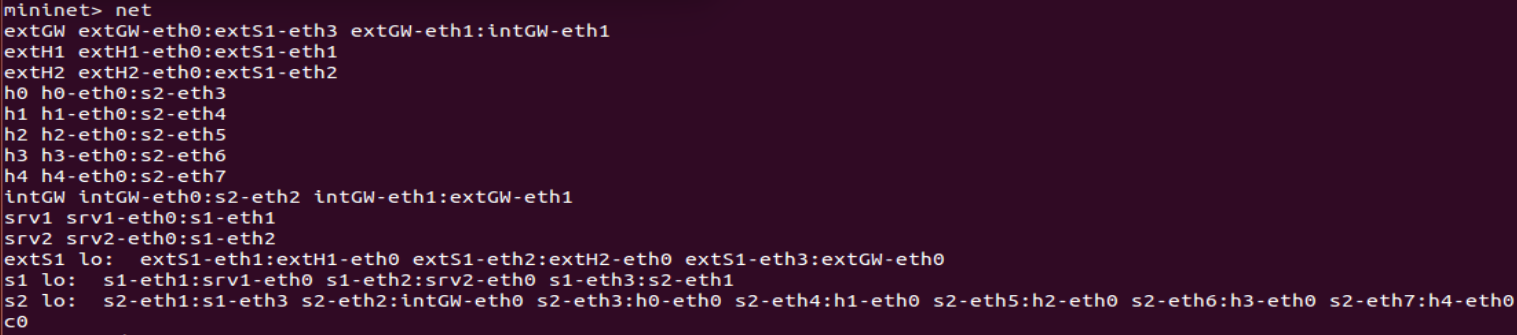
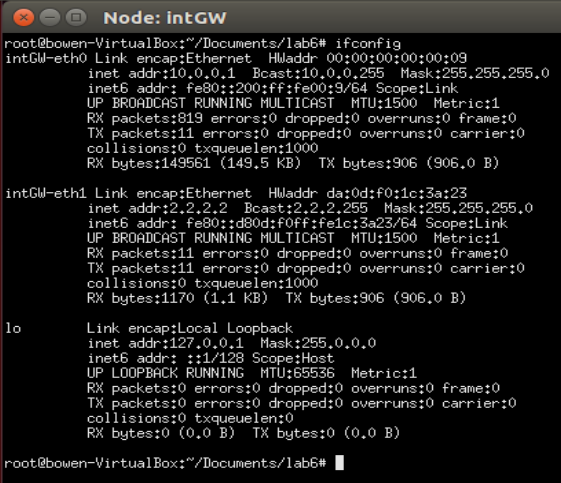
**50.012 Networks Lab 6**

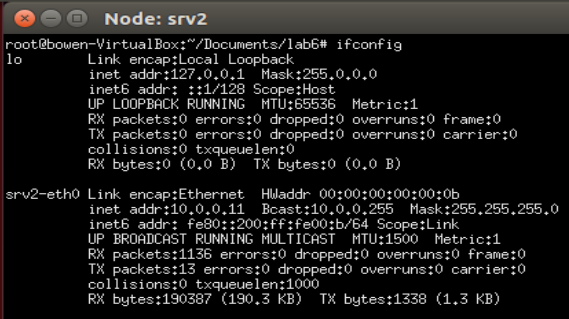
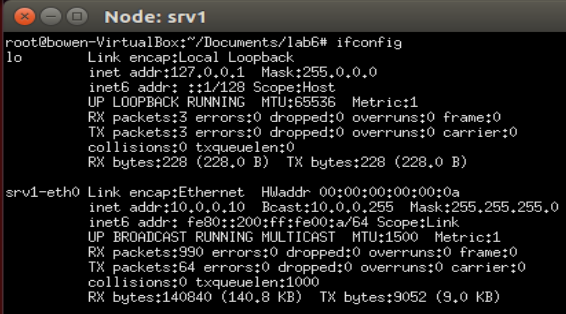
**Q1:**





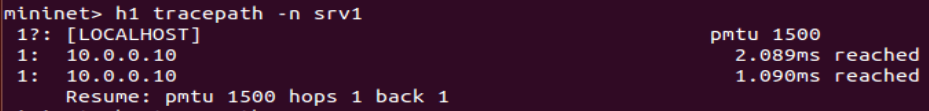
From above, the internal gateway router’s interface intGW-eth0 is connected to switch s2’s interface s2-eth2. From intGW’s ifconfig output and from the interface intGW-eth0 section, the IP subnet chosen for the hosts is 10.0.0.0/24.

**Q2:**



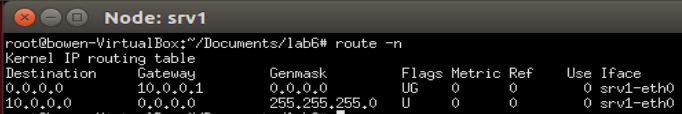
Yes, servers srv1 and srv2 are in the same subnet. Their IP addresses are 10.0.0.10 and 10.0.0.11 respectively, and their IP masks are both 255.255.255.0, meaning they are both in the subnet 10.0.0.0/24.

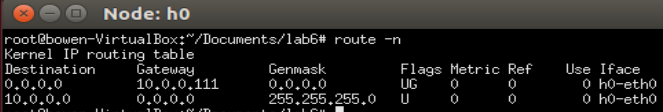
**Q3:**



The tracepath command only traces router hops and does not take into consideration link-layer switches.

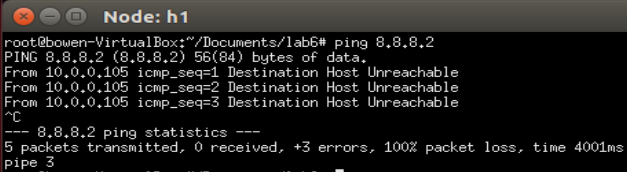
**Q4:**





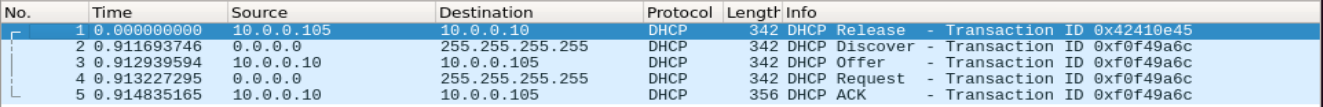
The gateway for srv1 and srv2 is 10.0.0.1, and the gateway for h0 to h4 is 10.0.0.111.

**Q5:**



No, h1 cannot ping test.net (8.8.8.2). The gateway IP specified for h1 (as well as h0, h2, h3, h4) is wrong, since 10.0.0.111 does not exist as an IP address for any interface in the network.

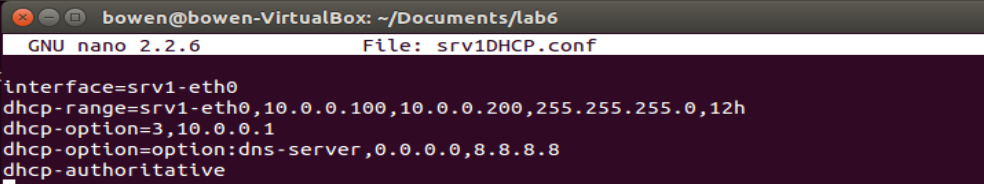
**Q6:**



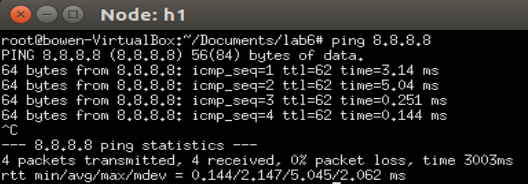
Yes, a DHCP server is running in the local network, on srv1. This can be seen from the Wireshark screenshot, which is captured during the time period where the commands h1 dhclient -r h1-eth0 and h1 dhclient h1-eth0 are executed. The DHCP ACK packets sent to h1 come from 10.0.0.10, which is the IP address of srv1.

**Q7:**

Yes, the third line ‘dhcp-option=3,10.0.0.111’ has a wrong gateway IP specified. 10.0.0.111 should be changed to 10.0.0.1. After modifying, the conf file is as shown:



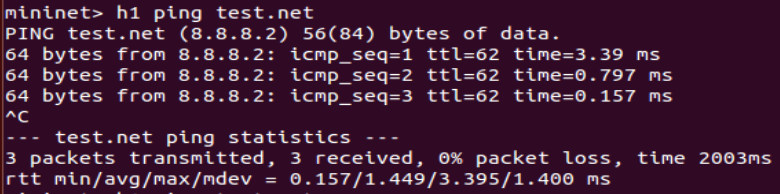
**Q8:**



Yes, h1 can reach Google (8.8.8.8) now.

**Q9:**

Before the changes were applied, h1 could ping test.net.

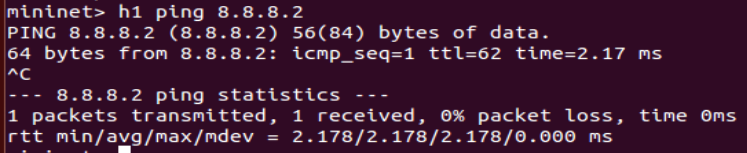


This is because when mininet is started at the very beginning, dnsmasq configures /etc/resolv.conf to have the DNS server at 8.8.8.8, so h1 could ping test.net.

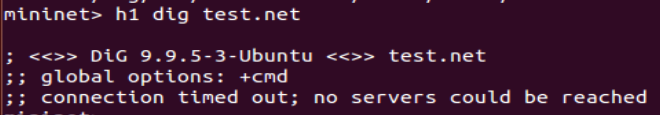
After the changes were applied, h1 could not ping test.net.

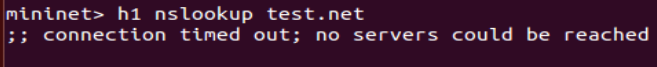


The hostname test.net cannot be resolved into its IP address (8.8.8.2). The issue is confirmed to be a DNS issue because h1 can ping 8.8.8.2



and that dig and nslookup cannot resolve its hostname.



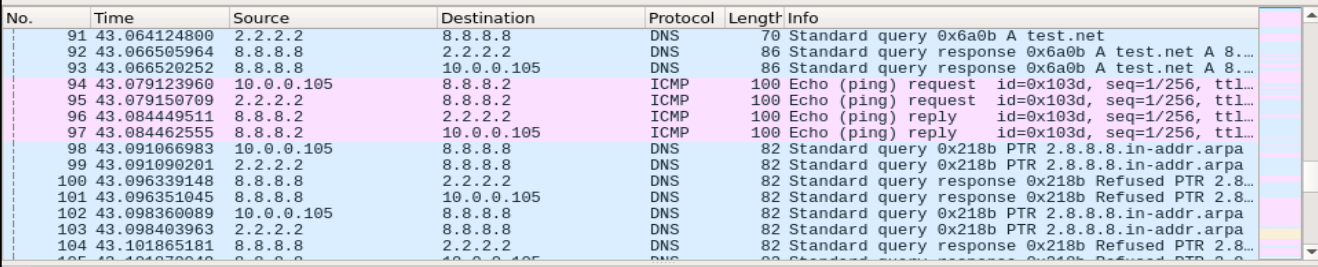


This happens because when restart the network manager service, the contents of the /etc/resolv.conf file will be automatically written back to 127.0.0.1, and overwrite whatever we wrote to it when the network manager service was stopped.

The IP of test.net is 8.8.8.2.

**Q10:**

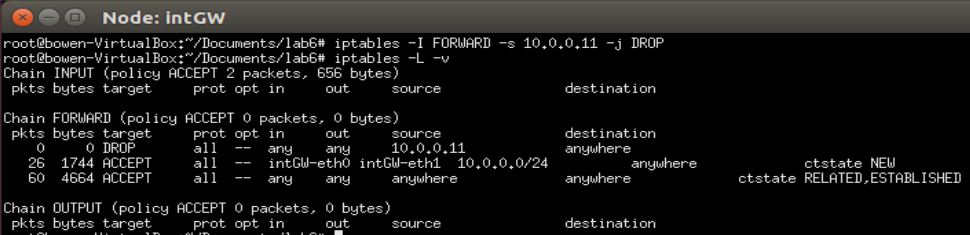
This node that provides NAT for the hosts with private IP addresses is intGW.

This is a Wireshark screenshot in intGW when the command h1 ping 8.8.8.8 is executed.  


In the enableNAT(net, ‘intGW’) function in net.py, IP masquerading and forwarding is set up in intGW. The results can be seen in the Wireshark screenshot above, where packets coming from 10.0.0.x will have their source IP translated by NAT from 10.0.0.x to 2.2.2.2 and packets coming from 8.8.8.x will have their destination IP translated by NAT from 2.2.2.2 to 10.0.0.x.

**Q11:**

To block all traffic from srv2 specifically to intGW, the command executed on intGW is iptables -I FORWARD -s 10.0.0.11 -j DROP



After this is done, srv2 can no longer ping 8.8.8.8.

