**Gede Ria Ghosalya**
**1001841**

*50.012 Networks – Lab 5*

## 3.1 Warming Up

*What is the local network chosen for the host?*

```
mininet> h2 ifconfig
h2-eth0   Link encap:Ethernet  HWaddr 00:00:00:00:00:06
          inet addr:10.0.0.106  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::200:ff:fe00:6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:114 errors:0 dropped:0 overruns:0 frame:0
          TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15423 (15.4 KB)  TX bytes:1848 (1.8 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Ans: 10.0.0.0/24

*Are the two servers* srv1 *and* srv2 in the same subnet?

```
mininet> srv1 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG    0      0        0 srv1-eth0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 srv1-eth0
mininet> srv2 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG    0      0        0 srv2-eth0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 srv2-eth0
```

Since both servers have the same IP after mask (subnet address), they are in the same subnet.

*Test if you can* tracepath *from* h1 *to* srv1.

```
mininet> h1 tracepath 10.0.0.10 -n
 1?: [LOCALHOST]                                         pmtu 1500
 1:  10.0.0.10                                             1.257ms reached
 1:  10.0.0.10                                             1.148ms reached
     Resume: pmtu 1500 hops 1 back 1
```

We are unable to observe the switch; the only machine visible through tracepath is the server.

*What is the gateway for all the devices?*

```
mininet> h2 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.111      0.0.0.0         UG    0      0        0 h2-eth0
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 h2-eth0
mininet> srv1 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.0.0.1        0.0.0.0         UG    0      0        0 srv1-eth0
10.0.0.0        0.0.0.0         255.0.0.0       U     0      0        0 srv1-eth0
```

Host gateway: 10.0.0.111
Server gateway: 10.0.0.1

*Can you ping the server* nils.net (8.8.8.2) *from* h1? *If not, any idea what is going wrong?*

```
mininet> h1 ping 8.8.8.2
PING 8.8.8.2 (8.8.8.2) 56(84) bytes of data.
From 10.0.0.105 icmp_seq=1 Destination Host Unreachable
From 10.0.0.105 icmp_seq=2 Destination Host Unreachable
From 10.0.0.105 icmp_seq=3 Destination Host Unreachable
```

H1 is unable to ping nils.net (8.8.8.2). No idea~

*Is a DHCP server running in the local network? On which machine?*

| Source | Destination | Protocol |
|---|---|---|
| 0.0.0.0 | 255.255.255.255 | DHCP |
| 10.0.0.10 | 10.0.0.105 | DHCP |
| fe80::74c6:a1ff:fe1 | ff02::fb | MDNS |

```
mininet> h1 dhclient h1-eth0
RTNETLINK answers: File exists
mininet> h1 ifconfig
h1-eth0   Link encap:Ethernet  HWaddr 00
          inet addr:10.0.0.105  Bcast:10
          inet6 addr: fe80::200:ff:fe00:
          UP BROADCAST RUNNING MULTICAST
```

Yes, the DHCP is running on local network, which is on srv1.

## 4.1 Changing DHCP Configuration

*Improve the configuration of the DHCP server.*

```
1 interface=srv1-eth0
2 dhcp-range=srv1-eth0,10.0.0.100,10.0.0.200,255.255.255.0,12h
3 dhcp-option=3,10.0.0.111
4 dhcp-option=option:dns-server,0.0.0.0,8.8.8.8
5 dhcp-authoritative
```

On line 3, this configuration will has 10.0.0.111 as its router (option 3). However, we dont have a device with this address. Neither srv1 nor h1 can ping to this address.

```
mininet> srv1 ping 10.0.0.111
PING 10.0.0.111 (10.0.0.111) 56(84) bytes of data.
From 10.0.0.10 icmp_seq=1 Destination Host Unreachable
From 10.0.0.10 icmp_seq=2 Destination Host Unreachable
From 10.0.0.10 icmp_seq=3 Destination Host Unreachable
^C
--- 10.0.0.111 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 3999ms
pipe 3
mininet> h1 ping 10.0.0.111
PING 10.0.0.111 (10.0.0.111) 56(84) bytes of data.
From 10.0.0.105 icmp_seq=1 Destination Host Unreachable
From 10.0.0.105 icmp_seq=2 Destination Host Unreachable
From 10.0.0.105 icmp_seq=3 Destination Host Unreachable
^C
--- 10.0.0.111 ping statistics ---
4 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2999ms
pipe 4
```

However, we do know that 10.0.0.1 exists as a router. So we can improve the configuration by changing the router.

```
1 interface=srv1-eth0
2 dhcp-range=srv1-eth0,10.0.0.100,10.0.0.200,255.255.255.0,12h
3 dhcp-option=3,10.0.0.1
4 dhcp-option=option:dns-server,0.0.0.0,8.8.8.8
5 dhcp-authoritative
```

Now h1 is able to ping 8.8.8.8

```
*** Starting CLI:
mininet> h1 ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=62 time=5.84 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=62 time=0.289 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=62 time=0.061 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=62 time=0.059 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=62 time=0.058 ms
```

## 4.2 DNS

*On* h1, *try to ping* nils.net. *Can you reach it? Why?*

```
mininet> h1 ping nils.net
PING nils.net (8.8.8.2) 56(84) bytes of data.
64 bytes from 8.8.8.2: icmp_seq=1 ttl=62 time=1.36 ms
64 bytes from 8.8.8.2: icmp_seq=2 ttl=62 time=0.284 ms
64 bytes from 8.8.8.2: icmp_seq=3 ttl=62 time=0.053 ms
64 bytes from 8.8.8.2: icmp_seq=4 ttl=62 time=0.065 ms
64 bytes from 8.8.8.2: icmp_seq=5 ttl=62 time=0.056 ms
```

Yes, h1 was able to ping nils.net, with the ip (8.8.8.2).

```
mininet> h1 dig nils.net

; <<>> DiG 9.10.3-P4-Ubuntu <<>> nils.net
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 46891
;; flags: qr aa rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

;; QUESTION SECTION:
;nils.net.                      IN      A

;; ANSWER SECTION:
nils.net.               0       IN      A       8.8.8.2

;; Query time: 2 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Oct 13 11:19:21 SGT 2017
;; MSG SIZE  rcvd: 42
```

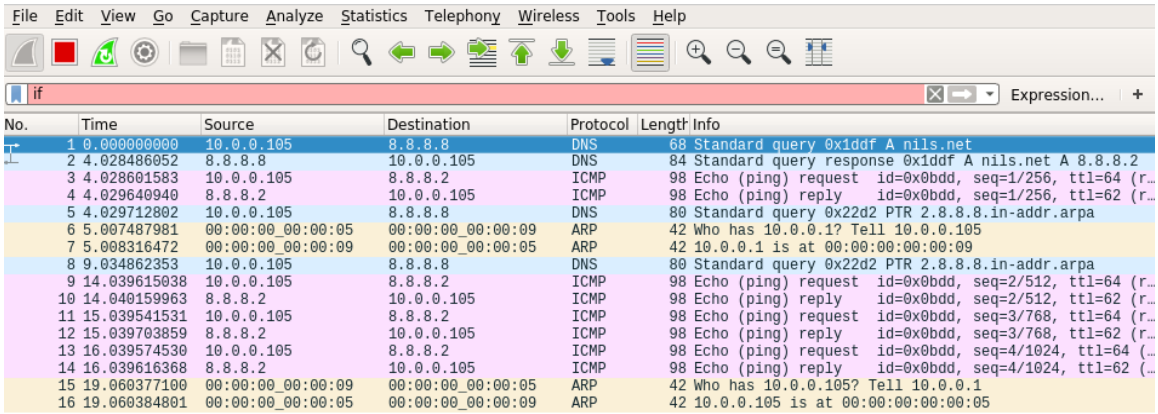
Upon using dig, it is discovered that the server 8.8.8.8 recognizes the address “nils.net” and return the IP address (8.8.8.2). As we know that h1 is able to reach 8.8.8.8, this is why h1 can connect to nils.net and recognize its IP address.

## 4.3 Observing NAT in action

*In the provided setup, one node provides NAT for the hosts with private IP address. Which node is this?*

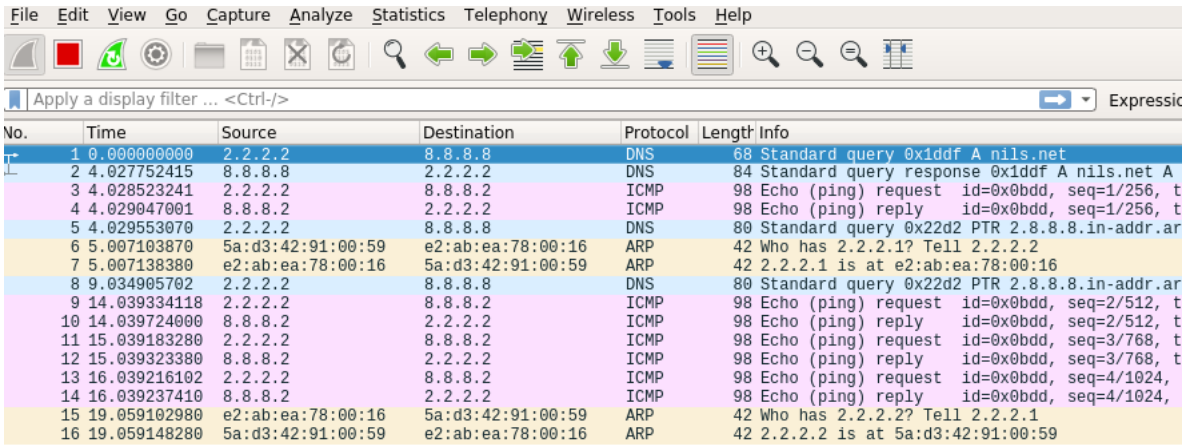
The node providing NAT is intGW.

Below is 2 screen capture from wireshark, for h1 and intGW.



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 10.0.0.105 | 8.8.8.8 | DNS | 68 | Standard query 0x1ddf A nils.net |
| 2 | 4.028486052 | 8.8.8.8 | 10.0.0.105 | DNS | 84 | Standard query response 0x1ddf A nils.net A 8.8.8.2 |
| 3 | 4.028601583 | 10.0.0.105 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=1/256, ttl=64 (r… |
| 4 | 4.029640940 | 8.8.8.2 | 10.0.0.105 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=1/256, ttl=62 (r… |
| 5 | 4.029712802 | 10.0.0.105 | 8.8.8.8 | DNS | 80 | Standard query 0x22d2 PTR 2.8.8.8.in-addr.arpa |
| 6 | 5.007487981 | 00:00:00_00:00:05 | 00:00:00_00:00:09 | ARP | 42 | Who has 10.0.0.1? Tell 10.0.0.105 |
| 7 | 5.008316472 | 00:00:00_00:00:09 | 00:00:00_00:00:05 | ARP | 42 | 10.0.0.1 is at 00:00:00:00:00:09 |
| 8 | 9.034862353 | 10.0.0.105 | 8.8.8.8 | DNS | 80 | Standard query 0x22d2 PTR 2.8.8.8.in-addr.arpa |
| 9 | 14.039615038 | 10.0.0.105 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=2/512, ttl=64 (r… |
| 10 | 14.040159963 | 8.8.8.2 | 10.0.0.105 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=2/512, ttl=62 (r… |
| 11 | 15.039541531 | 10.0.0.105 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=3/768, ttl=64 (r… |
| 12 | 15.039703859 | 8.8.8.2 | 10.0.0.105 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=3/768, ttl=62 (r… |
| 13 | 16.039574530 | 10.0.0.105 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=4/1024, ttl=64 (… |
| 14 | 16.039616368 | 8.8.8.2 | 10.0.0.105 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=4/1024, ttl=62 (… |
| 15 | 19.060377100 | 00:00:00_00:00:09 | 00:00:00_00:00:05 | ARP | 42 | Who has 10.0.0.105? Tell 10.0.0.1 |
| 16 | 19.060384801 | 00:00:00_00:00:05 | 00:00:00_00:00:09 | ARP | 42 | 10.0.0.105 is at 00:00:00:00:00:05 |

*Figure 4.3.1 wireshark screen capture from h1*



| No. | Time | Source | Destination | Protocol | Length | Info |
|---|---|---|---|---|---|---|
| 1 | 0.000000000 | 2.2.2.2 | 8.8.8.8 | DNS | 68 | Standard query 0x1ddf A nils.net |
| 2 | 4.027752415 | 8.8.8.8 | 2.2.2.2 | DNS | 84 | Standard query response 0x1ddf A nils.net A |
| 3 | 4.028523241 | 2.2.2.2 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=1/256, t |
| 4 | 4.029047001 | 8.8.8.2 | 2.2.2.2 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=1/256, t |
| 5 | 4.029553070 | 2.2.2.2 | 8.8.8.8 | DNS | 80 | Standard query 0x22d2 PTR 2.8.8.8.in-addr.ar |
| 6 | 5.007103870 | 5a:d3:42:91:00:59 | e2:ab:ea:78:00:16 | ARP | 42 | Who has 2.2.2.1? Tell 2.2.2.2 |
| 7 | 5.007138380 | e2:ab:ea:78:00:16 | 5a:d3:42:91:00:59 | ARP | 42 | 2.2.2.1 is at e2:ab:ea:78:00:16 |
| 8 | 9.034905702 | 2.2.2.2 | 8.8.8.8 | DNS | 80 | Standard query 0x22d2 PTR 2.8.8.8.in-addr.ar |
| 9 | 14.039334118 | 2.2.2.2 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=2/512, t |
| 10 | 14.039724000 | 8.8.8.2 | 2.2.2.2 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=2/512, t |
| 11 | 15.039183280 | 2.2.2.2 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=3/768, t |
| 12 | 15.039323380 | 8.8.8.2 | 2.2.2.2 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=3/768, t |
| 13 | 16.039216102 | 2.2.2.2 | 8.8.8.2 | ICMP | 98 | Echo (ping) request id=0x0bdd, seq=4/1024, |
| 14 | 16.039237410 | 8.8.8.2 | 2.2.2.2 | ICMP | 98 | Echo (ping) reply id=0x0bdd, seq=4/1024, |
| 15 | 19.059102980 | e2:ab:ea:78:00:16 | 5a:d3:42:91:00:59 | ARP | 42 | Who has 2.2.2.2? Tell 2.2.2.1 |
| 16 | 19.059148280 | 5a:d3:42:91:00:59 | e2:ab:ea:78:00:16 | ARP | 42 | 2.2.2.2 is at 5a:d3:42:91:00:59 |

*Figure 4.3.2 wireshark screen capture from intGW*

It can be seen that the request sent from h1 (10.0.0.105) to nils.net is reflected as request from intGW (2.2.2.2) to nils.net.

Opening net.py, the *enableNAT* function is written as follows.

```
def enableNAT(net,hostn):
    # this assumes that internal interface is eth0, external interface is eth1, and the network is 10.0.0.0/24
    host = net.getNodeByName(hostn)
    host.cmd("iptables -A FORWARD -o %s-eth1 -i %s-eth0 -s 10.0.0.0/24 -m conntrack --ctstate NEW -j ACCEPT"%(hostn,hostn))
    host.cmd("iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT")
    host.cmd("iptables -t nat -F POSTROUTING")
    host.cmd("iptables -t nat -A POSTROUTING -o %s-eth1 -j MASQUERADE"%hostn)
```

From here we can see that the address range for NAT translation is 10.0.0.0/24 ( or 10.0.0.0 10.0.0.255)

4.4 Simple Firewalling

Initially, srv2 ( 10.0.0.11) is able to ping both intGW (10.0.0.1) and outside network (8.8.8.8) as shown below.

```
oot@desktop:~/gede/networks_lab/lab5# ping 8.8.8.8
ING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
4 bytes from 8.8.8.8: icmp_seq=1 ttl=62 time=3.02 ms
4 bytes from 8.8.8.8: icmp_seq=2 ttl=62 time=0.251 ms
C
-- 8.8.8.8 ping statistics ---
 packets transmitted, 2 received, 0% packet loss, time 1001ms
tt min/avg/max/mdev = 0.251/1.639/3.027/1.388 ms
oot@desktop:~/gede/networks_lab/lab5# ping 10.0.0.1
ING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
4 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=1.33 ms
4 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.244 ms
C
-- 10.0.0.1 ping statistics ---
 packets transmitted, 2 received, 0% packet loss, time 1001ms
tt min/avg/max/mdev = 0.244/0.789/1.334/0.545 ms
```

However, we are able to block srv2 by adding REJECT rule to iptables, as follows.

```
root@desktop:~/gede/networks_lab/lab5# iptables -I INPUT -s 10.0.0.11 -j REJECT
root@desktop:~/gede/networks_lab/lab5# iptables -I OUTPUT -s 10.0.0.11 -j REJECT
root@desktop:~/gede/networks_lab/lab5# iptables -I FORWARD -s 10.0.0.11 -j REJECT
root@desktop:~/gede/networks_lab/lab5#
```

Note that FORWARD rule should also be added to prevent forward connection from srv2 to outside network, such as 8.8.8.8 node. After the above rule is added, we can see that srv2 is no longer able to ping neither intGW nor 8.8.8.8 node.

```
root@desktop:~/gede/networks_lab/lab5# ping 10.0.0.1
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Port Unreachable
From 10.0.0.1 icmp_seq=2 Destination Port Unreachable
^C
--- 10.0.0.1 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 999ms

root@desktop:~/gede/networks_lab/lab5# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Port Unreachable
From 10.0.0.1 icmp_seq=2 Destination Port Unreachable
^C
--- 8.8.8.8 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet loss, time 999ms
```