Week #8
IPv4 Addressing and Static Routing

Name – B.Pravena **Section -** B
SRN -PES2UG19CS076

Objective: To setup a network with two routers and exchange packets across routers.

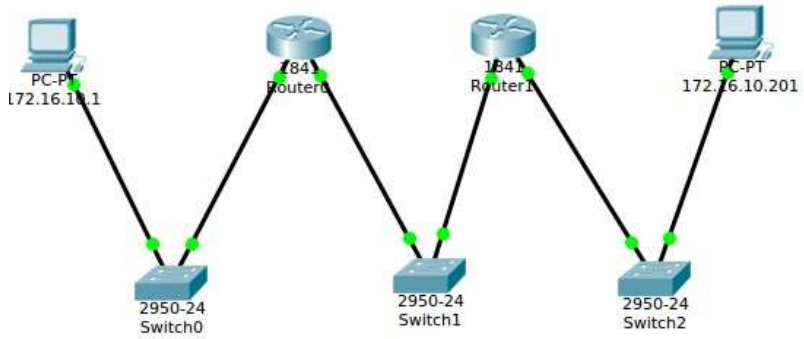
Hardware Requirements:

- Desktops/Laptops : 4
- Switch : 3
- Patch Cords (1.5m) : 6
- External NIC : 2

Software Requirements:

- Wireshark Tool
- Ubuntu Linux Operating System

Topology Description: Design a network with at least 2 router networks. Host **Ha** should be able to communicate with Host **Hd** using newly assigned addresses.



Note:

- Experiment to be accomplished in a group of 4.

- Make sure connections are flawless.
- Assign the IP address using commands or ‘Edit connections’.
- Don’t disturb existing hardware setup while setting IP address or doing experiment.
- Choose your ethernet interface according to your machine.

Task 1: Assign IP addresses to all computers A, B, C and D (Source Host Ha, Router R1, Router R2 & Destination Host Hd).

Step 1: Assign the IP address to the Ha.

$ sudo ip addr add 172.16.10.1/24 dev eth1

$ ip addr show

```
pesu@pesu-05:~$ ifconfig
enp2s8    Link encap:Ethernet  HWaddr 00:19:db:79:53:5e
          inet6 addr: fe80::246a:d5c:a6a1:7ebe/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:34212 errors:0 dropped:0 overruns:0 frame:0
          TX packets:23747 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:30461144 (30.4 MB)  TX bytes:4545860 (4.5 MB)

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:2213 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2213 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:199864 (199.8 KB)  TX bytes:199864 (199.8 KB)

pesu@pesu-05:~$ sudo ip addr add 172.16.10.1/24 dev enp2s8
[sudo] password for pesu:
```

Step 2: Assign the IP address to R1.

$ sudo ip addr add

172.16.10.201/24 dev eth1

$ sudo ip addr add

172.16.11.1/24 dev eth2

$ ip addr show

```
pesu@PESU-06:~$ ifconfig
enp2s8    Link encap:Ethernet  HWaddr 00:19:db:79:54:6d
          inet addr:172.16.10.201  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::219:dbff:fe79:546d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:83 errors:0 dropped:0 overruns:0 frame:0
          TX packets:148 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:9406 (9.4 KB)  TX bytes:16369 (16.3 KB)

enx28ee520069e0 Link encap:Ethernet  HWaddr 28:ee:52:00:69:e0
          inet addr:172.16.11.1  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::2aee:52ff:fe00:69e0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:135 errors:0 dropped:0 overruns:0 frame:0
          TX packets:122 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:13149 (13.1 KB)  TX bytes:13217 (13.2 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:89 errors:0 dropped:0 overruns:0 frame:0
          TX packets:89 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7774 (7.7 KB)  TX bytes:7774 (7.7 KB)
```

Step 3: Assign the IP address to R2.

$ sudo ip addr add 172.16.11.201/24 dev eth2

$ sudo ip addr add 172.16.12.1/24 dev eth1

$ ip addr show

```
pesu@PESU-07:~$ sudo ip addr add 172.16.11.201/24 dev enx28ee52002a4d
[sudo] password for pesu:
pesu@PESU-07:~$ sudo ip addr add 172.16.12.1/24 dev enp2s8
```

Step 4: Assign the IP address to the Hd.

$ sudo ip addr add 172.16.12.201/24 dev eth1

$ ip addr show

```
pesu@PESU-08:~$ sudo ip addr add 172.16.12.201/24 dev enp1s7
[sudo] password for pesu:
```

Note 1: The machines are physically on the same LAN, thus you may get ICMP redirect messages from other machines (in case you make some configuration mistakes). So, as a precautionary measure disable accepting the ICMP redirect packets. By default, the Ubuntu Linux enables

accepting the ICMP redirect packets. On host machines Ha and Hd, issue the following command:

$ sudo sysctl –w net.ipv4.conf.all.accept_redirects=0

```
pesu@pesu-05:~$ sudo sysctl -w net.ipv4.conf.all.accept_redirects=0
net.ipv4.conf.all.accept_redirects = 0
```

Note 2: Since machines are on same physical interface, the router is going to send ICMP redirect message disturbing the routing decision by hosts. Thus, disable sending of the ICMP redirect packets by these routers with aliased interfaces. To have precautionary measures issue below command in router machines R1 and R2.

$ sudo sysctl –w net.ipv4.conf.all.send_redirects=0

Task 2: Convert the machines B and C into routers.

Note 1: Check if IP forwarding is enabled or not. We need to query the sysctl kernel value ***net.ipv4.ip_forward*** to see if forwarding is enabled or not using sysctl:

$ sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0

Command to set the value of ***net.ipv4.ip_forward*** in R1 & R2 is given below:

At R1: **$ sudo sysctl –w net.ipv4.ip_forward=1**

At R2: **$ sudo sysctl –w net.ipv4.ip_forward=1**

```
pesu@PESU-06:~$ sudo sysctl -w net.ipv4.conf.all.send_redirects=0
net.ipv4.conf.all.send_redirects = 0
pesu@PESU-06:~$ sysctl net.ipv4.ip_forward
net.ipv4.ip_forward = 0
pesu@PESU-06:~$ sudo sysctl -w net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
pesu@PESU-06:~$
```

Task 3: Verify the connection between Ha and Hd using ping command.

Initially test the connection of systems within the same network.

At Ha: **$ ping 172.16.10.1 (Local network)**

```
pesu@pesu-05:~$ ping 172.16.10.1
PING 172.16.10.1 (172.16.10.1) 56(84) bytes of data.
64 bytes from 172.16.10.1: icmp_seq=1 ttl=64 time=0.059 ms
64 bytes from 172.16.10.1: icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from 172.16.10.1: icmp_seq=3 ttl=64 time=0.052 ms
64 bytes from 172.16.10.1: icmp_seq=4 ttl=64 time=0.053 ms
64 bytes from 172.16.10.1: icmp_seq=5 ttl=64 time=0.055 ms
64 bytes from 172.16.10.1: icmp_seq=6 ttl=64 time=0.068 ms
64 bytes from 172.16.10.1: icmp_seq=7 ttl=64 time=0.083 ms
^Z
[2]+  Stopped                 ping 172.16.10.1
```

At Hd: **$ ping 172.16.10.201(Local network)**

```
pesu@PESU-08:~$ ping 172.16.12.201
PING 172.16.12.201 (172.16.12.201) 56(84) bytes of data.
64 bytes from 172.16.12.201: icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from 172.16.12.201: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from 172.16.12.201: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 172.16.12.201: icmp_seq=4 ttl=64 time=0.054 ms
64 bytes from 172.16.12.201: icmp_seq=5 ttl=64 time=0.054 ms
64 bytes from 172.16.12.201: icmp_seq=6 ttl=64 time=0.058 ms
64 bytes from 172.16.12.201: icmp_seq=7 ttl=64 time=0.056 ms
64 bytes from 172.16.12.201: icmp_seq=8 ttl=64 time=0.055 ms
^Z
[3]+  Stopped                 ping 172.16.12.201
```

Task 4: Insert routing table entries on each system to direct ipv4 packets to ping across the networks.

At Ha: **$ sudo ip route add 172.16.12.0/24 via 172.16.10.201**

$ sudo ip route add 172.16.11.0/24 via 172.16.10.201

$ ip route show

```
pesu@pesu-05:~$ sudo ip route add 172.16.12.0/24 via 172.16.10.201
pesu@pesu-05:~$ sudo ip route add 172.16.11.0/24 via 172.16.10.201
pesu@pesu-05:~$ ip route show
169.254.0.0/16 dev enp2s8  scope link  metric 1000
172.16.10.0/24 dev enp2s8  proto kernel  scope link  src 172.16.10.1
172.16.11.0/24 via 172.16.10.201 dev enp2s8
172.16.12.0/24 via 172.16.10.201 dev enp2s8
pesu@pesu-05:~$
```

In the first host since 172.16.10.0/24 network is local network we don't need any routing table entry. We need to have routing table entries for other networks such as 172.16.11.0/24 and 172.16.12.0/24.

At R1: **$ sudo ip route add 172.16.12.0/24 via 172.16.11.201**

$ ip route show

```
pesu@PESU-06:~$ sudo ip route add 172.16.12.0/24 via 172.16.11.201
pesu@PESU-06:~$ ip route show
169.254.0.0/16 dev enp2s8  scope link  metric 1000
172.16.10.0/24 dev enp2s8  proto kernel  scope link  src 172.16.10.201
172.16.11.0/24 dev enx28ee520069e0  proto kernel  scope link  src 172.16.11.1
172.16.12.0/24 via 172.16.11.201 dev enx28ee520069e0
```

Since R1 is connected to 172.16.10.0/24 and 172.16.11.0/24 networks we need to have one routing table entry to 172.16.12.0/24.

At R2: **$ sudo ip route add 172.16.10.0/24 via 172.16.11.1**

$ ip route show

```
pesu@PESU-07:~$ sudo ip route add 172.16.10.0/24 via 172.16.11.1
pesu@PESU-07:~$ ip route show
169.254.0.0/16 dev enx28ee52002a4d  scope link  metric 1000
172.16.10.0/24 via 172.16.11.1 dev enx28ee52002a4d
172.16.11.0/24 dev enx28ee52002a4d  proto kernel  scope link  src 172.16.11.201
172.16.12.0/24 dev enp2s8  proto kernel  scope link  src 172.16.12.1
pesu@PESU-07:~$
```

At Hd: **$ sudo ip route add 172.16.10.0/24 via 172.16.12.1**

$ sudo ip route add 172.16.11.0/24 via 172.16.12.1

$ ip route show

```
pesu@PESU-08:~$ sudo ip route add 172.16.10.0/24 via 172.16.12.1
[sudo] password for pesu:
pesu@PESU-08:~$ sudo ip route add 172.16.11.0/24 via 172.16.12.1
pesu@PESU-08:~$ ip route show
169.254.0.0/16 dev enp1s7  scope link  metric 1000
172.16.10.0/24 via 172.16.12.1 dev enp1s7
172.16.11.0/24 via 172.16.12.1 dev enp1s7
172.16.12.0/24 dev enp1s7  proto kernel  scope link  src 172.16.12.201
```

Task 5: After adding routing table entries again verify the connection from Ha and Hd using ping command.

Step 1: Testing path from Ha and Hd

$ ping 172.16.12.1 and $ ping 172.16.12.201

```
pesu@pesu-05:~$ ping 172.16.12.1
PING 172.16.12.1 (172.16.12.1) 56(84) bytes of data.
64 bytes from 172.16.12.1: icmp_seq=1 ttl=63 time=0.501 ms
64 bytes from 172.16.12.1: icmp_seq=2 ttl=63 time=0.492 ms
64 bytes from 172.16.12.1: icmp_seq=3 ttl=63 time=0.484 ms
64 bytes from 172.16.12.1: icmp_seq=4 ttl=63 time=0.466 ms
64 bytes from 172.16.12.1: icmp_seq=5 ttl=63 time=0.335 ms
64 bytes from 172.16.12.1: icmp_seq=6 ttl=63 time=0.435 ms
^Z
[4]+  Stopped                 ping 172.16.12.1
pesu@pesu-05:~$ ping 172.16.12.201
PING 172.16.12.201 (172.16.12.201) 56(84) bytes of data.
64 bytes from 172.16.12.201: icmp_seq=1 ttl=62 time=0.567 ms
64 bytes from 172.16.12.201: icmp_seq=2 ttl=62 time=0.474 ms
64 bytes from 172.16.12.201: icmp_seq=3 ttl=62 time=0.587 ms
64 bytes from 172.16.12.201: icmp_seq=4 ttl=62 time=0.591 ms
^Z
[5]+  Stopped                 ping 172.16.12.201
```

Step 2: Testing path from Hd and Ha

$ ping 172.16.12.1 and $ ping 172.16.12.201

```
pesu@PESU-08:~$ ping 172.16.12.201
PING 172.16.12.201 (172.16.12.201) 56(84) bytes of data.
64 bytes from 172.16.12.201: icmp_seq=1 ttl=64 time=0.066 ms
64 bytes from 172.16.12.201: icmp_seq=2 ttl=64 time=0.055 ms
64 bytes from 172.16.12.201: icmp_seq=3 ttl=64 time=0.053 ms
64 bytes from 172.16.12.201: icmp_seq=4 ttl=64 time=0.054 ms
64 bytes from 172.16.12.201: icmp_seq=5 ttl=64 time=0.054 ms
64 bytes from 172.16.12.201: icmp_seq=6 ttl=64 time=0.058 ms
64 bytes from 172.16.12.201: icmp_seq=7 ttl=64 time=0.056 ms
64 bytes from 172.16.12.201: icmp_seq=8 ttl=64 time=0.055 ms
^Z
[3]+  Stopped                 ping 172.16.12.201
pesu@PESU-08:~$ ping 172.16.12.1
PING 172.16.12.1 (172.16.12.1) 56(84) bytes of data.
64 bytes from 172.16.12.1: icmp_seq=1 ttl=64 time=0.175 ms
64 bytes from 172.16.12.1: icmp_seq=2 ttl=64 time=0.157 ms
64 bytes from 172.16.12.1: icmp_seq=3 ttl=64 time=0.148 ms
64 bytes from 172.16.12.1: icmp_seq=4 ttl=64 time=0.150 ms
64 bytes from 172.16.12.1: icmp_seq=5 ttl=64 time=0.169 ms
64 bytes from 172.16.12.1: icmp_seq=6 ttl=64 time=0.139 ms
64 bytes from 172.16.12.1: icmp_seq=7 ttl=64 time=0.159 ms
64 bytes from 172.16.12.1: icmp_seq=8 ttl=64 time=0.146 ms
64 bytes from 172.16.12.1: icmp_seq=9 ttl=64 time=0.151 ms
```

Task 6: Check each system neighbor to verify the connection. ip neighbor provides a command line interface to [display the neighbor table (ARP cache)]

At Ha: **$ ip neigh show**

```
student@pesit-To-be-filled-by-O-E-M:~$ ip neigh  show
172.16.10.201 dev eth1 lladdr 50:e5:49:1d:4a:ad REACHABLE
student@pesit-To-be-filled-by-O-E-M:~$
```

At R1: **$ ip neigh show**

```
student@pesit-To-be-filled-by-O-E-M:~$ ip neigh show
'172.16.11.201 dev eth2 lladdr 28:10:7b:4b:15:05 STALE
172.16.10.1 dev eth1 lladdr 50:e5:49:1c:64:c0 STALE
student@pesit-To-be-filled-by-O-E-M:~$ '
```

At R2: **$ ip neigh show**

```
pesu@PESU-07:~$ ip neigh show
172.16.11.1 dev enx28ee52002a4d lladdr 28:ee:52:00:69:e0 REACHABLE
172.16.12.201 dev enp2s8 lladdr 00:1a:4d:0d:0c:91 REACHABLE
pesu@PESU-07:~$
```

At Hd: **$ ip neigh show**

```
student@pesit-To-be-filled-by-O-E-M:~$ ip neigh show
172.16.12.1 dev eth1 lladdr 50:e5:49:1c:d3:ae REACHABLE
student@pesit-To-be-filled-by-O-E-M:~$
```

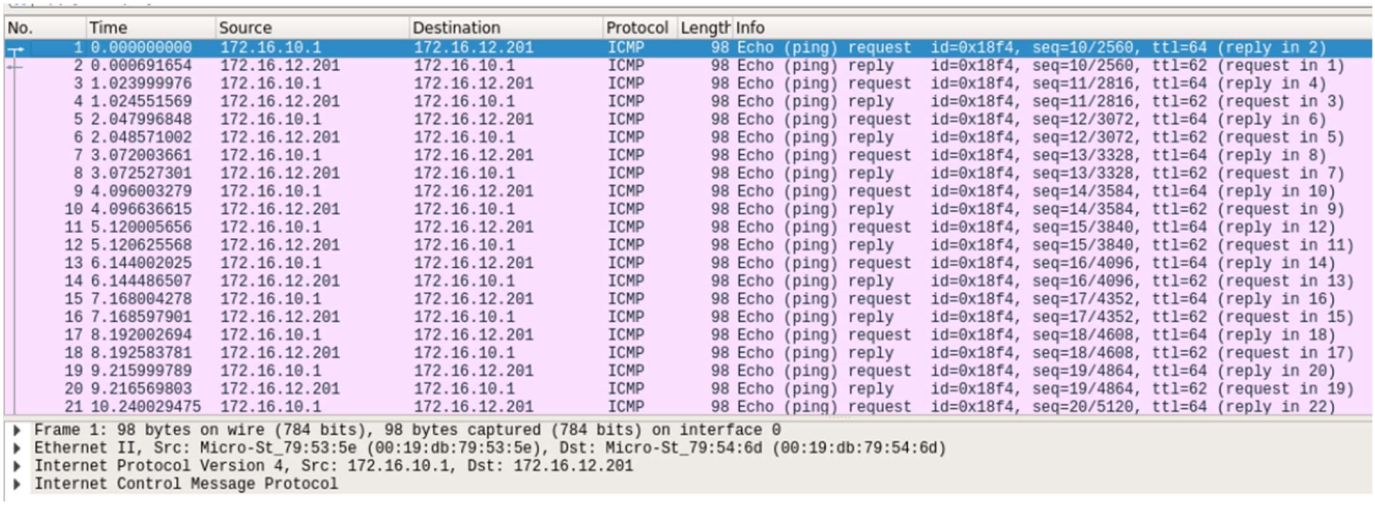
Task 7: Capture packets from Ha and Hb using Wireshark tool.

Step 1: Capture packets from Ha and Hd.

At Ha:

T1: $ sudo wireshark

T2: $ ping 172.16.12.201



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=10/2560, ttl=64 (reply in 2)
2	0.000691654	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=10/2560, ttl=62 (request in 1)
3	1.023999976	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=11/2816, ttl=64 (reply in 4)
4	1.024551569	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=11/2816, ttl=62 (request in 3)
5	2.047996848	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=12/3072, ttl=64 (reply in 6)
6	2.048571002	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=12/3072, ttl=62 (request in 5)
7	3.072003661	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=13/3328, ttl=64 (reply in 8)
8	3.072527301	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=13/3328, ttl=62 (request in 7)
9	4.096003279	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=14/3584, ttl=64 (reply in 10)
10	4.096636615	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=14/3584, ttl=62 (request in 9)
11	5.120005656	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=15/3840, ttl=64 (reply in 12)
12	5.120625568	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=15/3840, ttl=62 (request in 11)
13	6.144002025	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=16/4096, ttl=64 (reply in 14)
14	6.144486507	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=16/4096, ttl=62 (request in 13)
15	7.168004278	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=17/4352, ttl=64 (reply in 16)
16	7.168597901	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=17/4352, ttl=62 (request in 15)
17	8.192002694	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=18/4608, ttl=64 (reply in 18)
18	8.192583781	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=18/4608, ttl=62 (request in 17)
19	9.215999789	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=19/4864, ttl=64 (reply in 20)
20	9.216569803	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x18f4, seq=19/4864, ttl=62 (request in 19)
21	10.240029475	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x18f4, seq=20/5120, ttl=64 (reply in 22)

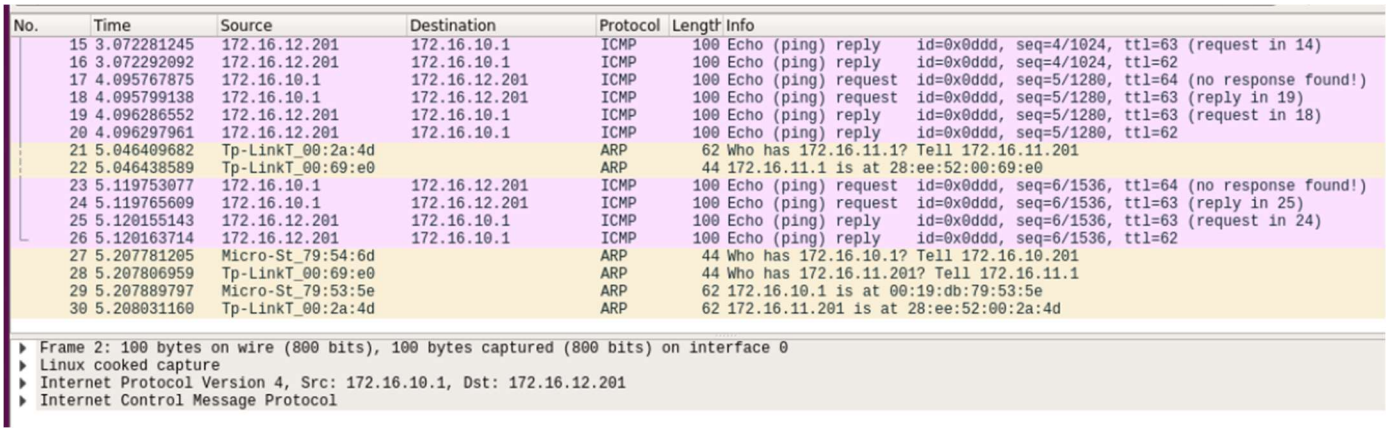
```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: Micro-St_79:53:5e (00:19:db:79:53:5e), Dst: Micro-St_79:54:6d (00:19:db:79:54:6d)
▸ Internet Protocol Version 4, Src: 172.16.10.1, Dst: 172.16.12.201
▸ Internet Control Message Protocol
```

Step 2: Capture packets from R1 using both eth1

and eth2 interfaces.

$ sudo wireshark

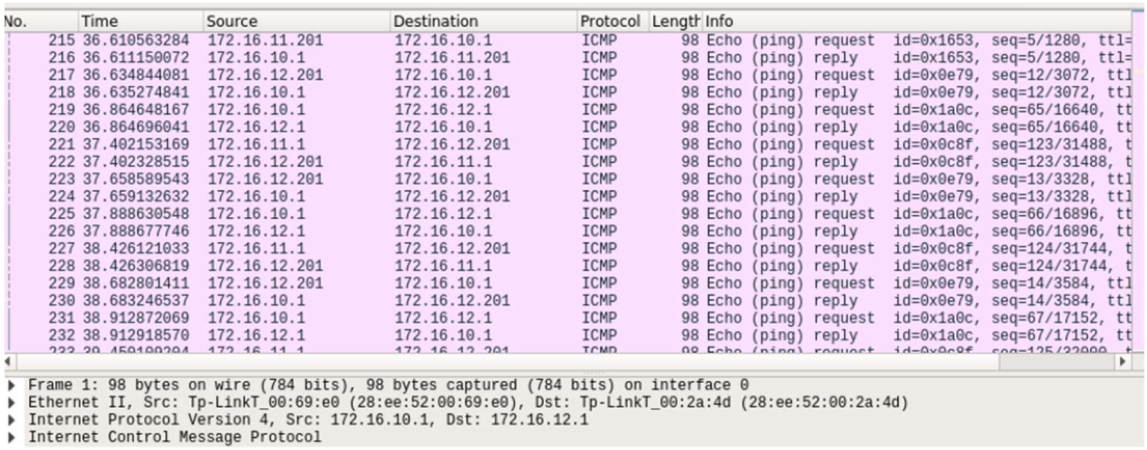
At eth1:



No.	Time	Source	Destination	Protocol	Length	Info
15	3.072281245	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=4/1024, ttl=63 (request in 14)
16	3.072292092	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=4/1024, ttl=62
17	4.095767875	172.16.10.1	172.16.12.201	ICMP	100	Echo (ping) request id=0x0ddd, seq=5/1280, ttl=64 (no response found!)
18	4.095799138	172.16.10.1	172.16.12.201	ICMP	100	Echo (ping) request id=0x0ddd, seq=5/1280, ttl=63 (reply in 19)
19	4.096286552	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=5/1280, ttl=63 (request in 18)
20	4.096297961	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=5/1280, ttl=62
21	5.046409682	Tp-LinkT_00:2a:4d		ARP	62	Who has 172.16.11.1? Tell 172.16.11.201
22	5.046438589	Tp-LinkT_00:69:e0		ARP	44	172.16.11.1 is at 28:ee:52:00:69:e0
23	5.119753077	172.16.10.1	172.16.12.201	ICMP	100	Echo (ping) request id=0x0ddd, seq=6/1536, ttl=64 (no response found!)
24	5.119765609	172.16.10.1	172.16.12.201	ICMP	100	Echo (ping) request id=0x0ddd, seq=6/1536, ttl=63 (reply in 25)
25	5.120155143	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=6/1536, ttl=63 (request in 24)
26	5.120163714	172.16.12.201	172.16.10.1	ICMP	100	Echo (ping) reply id=0x0ddd, seq=6/1536, ttl=62
27	5.207781205	Micro-St_79:54:6d		ARP	44	Who has 172.16.10.1? Tell 172.16.10.201
28	5.207806959	Tp-LinkT_00:69:e0		ARP	44	Who has 172.16.11.201? Tell 172.16.11.1
29	5.207889797	Micro-St_79:53:5e		ARP	62	172.16.10.1 is at 00:19:db:79:53:5e
30	5.208031160	Tp-LinkT_00:2a:4d		ARP	62	172.16.11.201 is at 28:ee:52:00:2a:4d

```
▸ Frame 2: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 172.16.10.1, Dst: 172.16.12.201
▸ Internet Control Message Protocol
```

At eth2:



No.	Time	Source	Destination	Protocol	Length	Info
215	36.610563284	172.16.11.201	172.16.10.1	ICMP	98	Echo (ping) request id=0x1653, seq=5/1280, ttl=
216	36.611150072	172.16.10.1	172.16.11.201	ICMP	98	Echo (ping) reply id=0x1653, seq=5/1280, ttl=
217	36.634844081	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) request id=0x0e79, seq=12/3072, ttl
218	36.635274841	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) reply id=0x0e79, seq=12/3072, ttl
219	36.864648167	172.16.10.1	172.16.12.1	ICMP	98	Echo (ping) request id=0x1a0c, seq=65/16640, tt
220	36.864696041	172.16.12.1	172.16.10.1	ICMP	98	Echo (ping) reply id=0x1a0c, seq=65/16640, tt
221	37.402153169	172.16.11.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x0c8f, seq=123/31488, t
222	37.402328515	172.16.12.201	172.16.11.1	ICMP	98	Echo (ping) reply id=0x0c8f, seq=123/31488, t
223	37.658589543	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) request id=0x0e79, seq=13/3328, ttl
224	37.659132632	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) reply id=0x0e79, seq=13/3328, ttl
225	37.888630548	172.16.10.1	172.16.12.1	ICMP	98	Echo (ping) request id=0x1a0c, seq=66/16896, tt
226	37.888677746	172.16.12.1	172.16.10.1	ICMP	98	Echo (ping) reply id=0x1a0c, seq=66/16896, tt
227	38.426121033	172.16.11.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x0c8f, seq=124/31744, t
228	38.426306819	172.16.12.201	172.16.11.1	ICMP	98	Echo (ping) reply id=0x0c8f, seq=124/31744, t
229	38.682801411	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) request id=0x0e79, seq=14/3584, ttl
230	38.683246537	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) reply id=0x0e79, seq=14/3584, ttl
231	38.912872069	172.16.10.1	172.16.12.1	ICMP	98	Echo (ping) request id=0x1a0c, seq=67/17152, tt
232	38.912918570	172.16.12.1	172.16.10.1	ICMP	98	Echo (ping) reply id=0x1a0c, seq=67/17152, tt

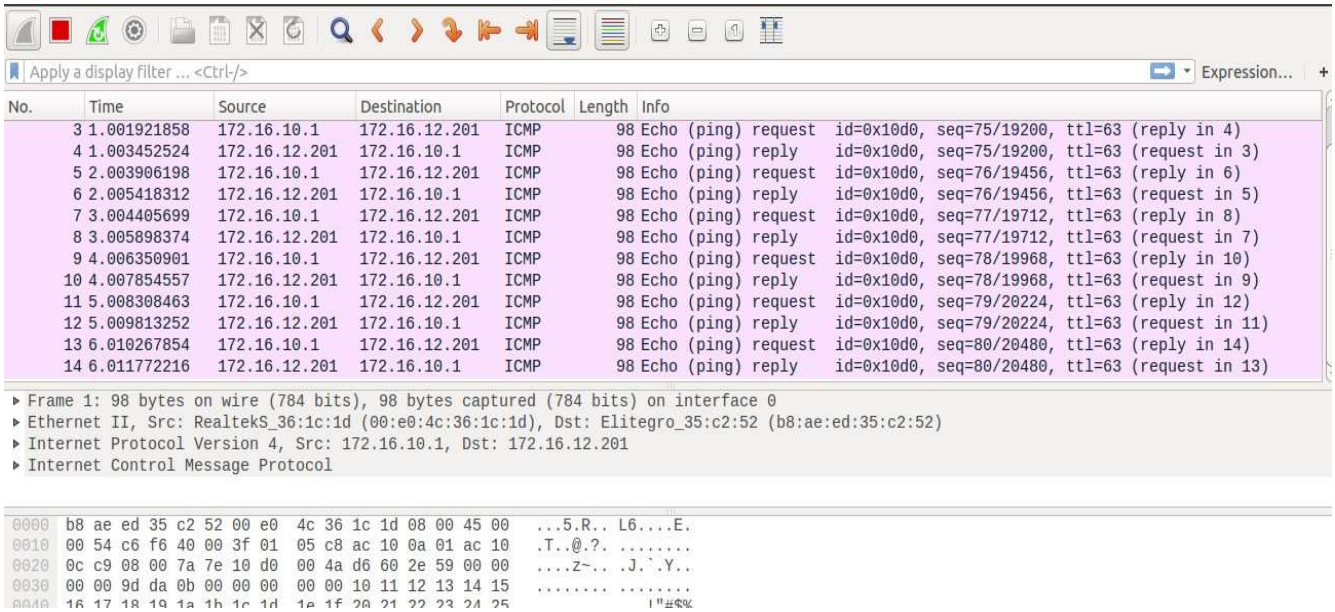
```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: Tp-LinkT_00:69:e0 (28:ee:52:00:69:e0), Dst: Tp-LinkT_00:2a:4d (28:ee:52:00:2a:4d)
▸ Internet Protocol Version 4, Src: 172.16.10.1, Dst: 172.16.12.1
▸ Internet Control Message Protocol
```

Step 3: Capture packets from R2 using both eth1 and eth2 interfaces.

$ sudo

wireshark

At eth1:



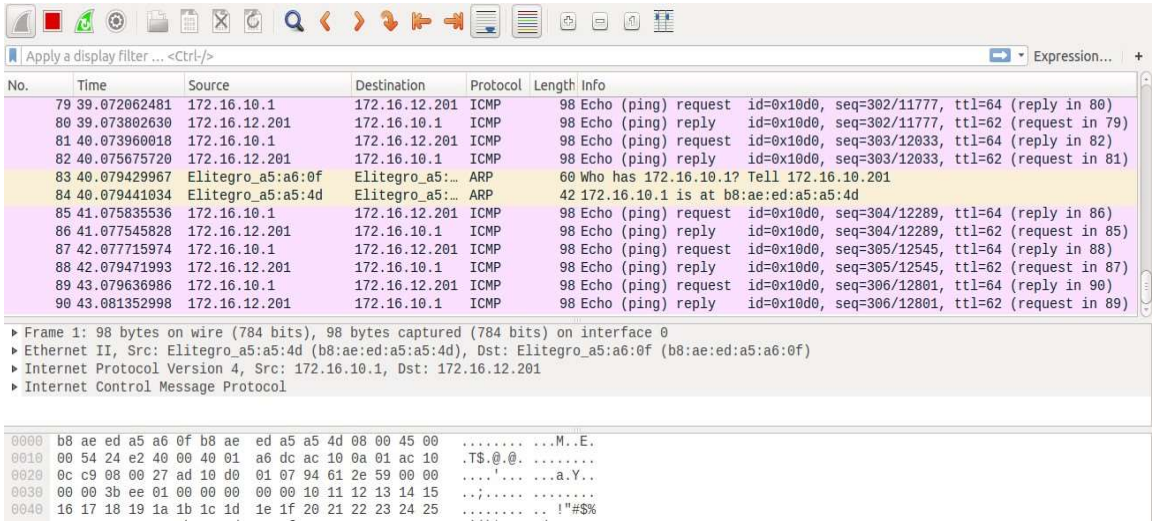
Apply a display filter ... <Ctrl-/> Expression... +

No.	Time	Source	Destination	Protocol	Length	Info
3	1.001921858	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=75/19200, ttl=63 (reply in 4)
4	1.003452524	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=75/19200, ttl=63 (request in 3)
5	2.003906198	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=76/19456, ttl=63 (reply in 6)
6	2.005418312	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=76/19456, ttl=63 (request in 5)
7	3.004405699	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=77/19712, ttl=63 (reply in 8)
8	3.005898374	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=77/19712, ttl=63 (request in 7)
9	4.006350901	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=78/19968, ttl=63 (reply in 10)
10	4.007854557	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=78/19968, ttl=63 (request in 9)
11	5.008308463	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=79/20224, ttl=63 (reply in 12)
12	5.009813252	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=79/20224, ttl=63 (request in 11)
13	6.010267854	172.16.10.1	172.16.12.201	ICMP	98	Echo (ping) request id=0x10d0, seq=80/20480, ttl=63 (reply in 14)
14	6.011772216	172.16.12.201	172.16.10.1	ICMP	98	Echo (ping) reply id=0x10d0, seq=80/20480, ttl=63 (request in 13)

```
▸ Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) on interface 0
▸ Ethernet II, Src: RealtekS_36:1c:1d (00:e0:4c:36:1c:1d), Dst: Elitegro_35:c2:52 (b8:ae:ed:35:c2:52)
▸ Internet Protocol Version 4, Src: 172.16.10.1, Dst: 172.16.12.201
▸ Internet Control Message Protocol
```

```
0000  b8 ae ed 35 c2 52 00 e0  4c 36 1c 1d 08 00 45 00   ...5.R.. L6....E.
0010  00 54 c6 f6 40 00 3f 01  05 c8 ac 10 0a 01 ac 10   .T..@.?. ........
0020  0c c9 08 00 7a 7e 10 d0  00 4a d6 60 2e 59 00 00   ....z~.. .J.`.Y..
0030  00 00 9d da 0b 00 00 00  00 00 10 11 12 13 14 15   ........ ........
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25                 !"#$%
```

At eth2:



Step 4: Capture packets from Hd and Ha.

At Hd:

T1: $ sudo wireshark