

# Mininet Project

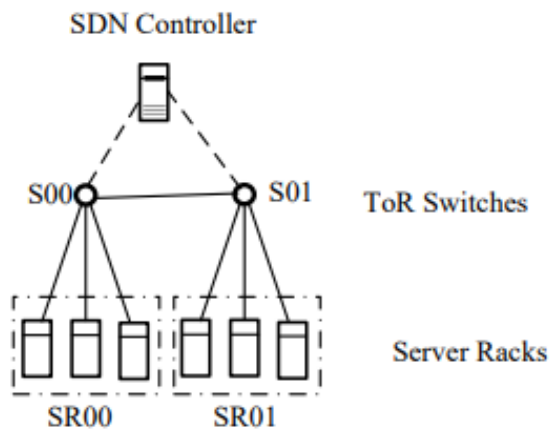
## EE 295 Project

Rishabh Uppal

SJSU ID - 015276573

## Simulation 1

### Topology



CUSTOM TOPOLOGY FOR SIMULATION 1

```
from mininet.topo import Topo
```

```

class MyTopo( Topo ):
    "Simple topology example."

    def build( self ):
        "Create custom topo."

        # Add hosts [server rack]
        h1 = self.addHost( 'h1' )
        h2 = self.addHost( 'h2' )
        h3 = self.addHost( 'h3' )
        h4 = self.addHost( 'h4' )
        h5 = self.addHost( 'h5' )
        h6 = self.addHost( 'h6' )

        # Add switches
        S00 = self.addSwitch( 'S00' )
        S01 = self.addSwitch( 'S01' )

        # Add links
        self.addLink( h1, S00 )
        self.addLink( h2, S00 )
        self.addLink( h3, S00 )
        self.addLink( h4, S01 )
        self.addLink( h5, S01 )
        self.addLink( h6, S01 )
        self.addLink( S00, S01 )

topos = { 'mytopo': ( lambda: MyTopo() ) }

```

Custom code written in python language via gedit text editor.

- Initialized the number of host [total of 6] where **SR00** contains [ h1-h3 ] and **SR01** contains [h4-h6] .
- Initialized the number of switches S00 and S01 which are **Tor Switches**.
- Initialized the links between the host and switches.

## Basic Setup of the mininet by introducing custom topology with a default controller

```

sudo mn --custom sim1.py [filename of custom topology] --topo mytopo [topology name]

```

```
rishabh@rishabh-VirtualBox: ~/Desktop
rishabh@rishabh-VirtualBox:~/Desktop$ sudo mn --custom sim1.py --topo mytopo
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6
*** Adding switches:
S00 S01
*** Adding links:
(S00, S01) (h1, S00) (h2, S00) (h3, S00) (h4, S01) (h5, S01) (h6, S01)
*** Configuring hosts
h1 h2 h3 h4 h5 h6
*** Starting controller
c0
*** Starting 2 switches
S00 S01 ...
*** Starting CLI:
mininet> nodes
available nodes are:
S00 S01 c0 h1 h2 h3 h4 h5 h6
mininet>
```

code

nodes

## 1) Conduct a test using **Pingall** command

```
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>
```

None of the packets dropped. means the connection is stable

- Calculating the **Average round trip time (RTT)**

```
command - pingallfull
```

```
rishabh@rishabh-VirtualBox: ~/Desktop
mininet> pingallfull
*** 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:
h1->h2: 1/1, rtt min/avg/max/mdev 0.915/0.915/0.915/0.000 ms
h1->h3: 1/1, rtt min/avg/max/mdev 0.261/0.261/0.261/0.000 ms
h1->h4: 1/1, rtt min/avg/max/mdev 0.338/0.338/0.338/0.000 ms
h1->h5: 1/1, rtt min/avg/max/mdev 0.813/0.813/0.813/0.000 ms
h1->h6: 1/1, rtt min/avg/max/mdev 0.874/0.874/0.874/0.000 ms
h2->h1: 1/1, rtt min/avg/max/mdev 0.201/0.201/0.201/0.000 ms
h2->h3: 1/1, rtt min/avg/max/mdev 0.343/0.343/0.343/0.000 ms
h2->h4: 1/1, rtt min/avg/max/mdev 0.850/0.850/0.850/0.000 ms
h2->h5: 1/1, rtt min/avg/max/mdev 0.295/0.295/0.295/0.000 ms
h2->h6: 1/1, rtt min/avg/max/mdev 0.172/0.172/0.172/0.000 ms
h3->h1: 1/1, rtt min/avg/max/mdev 0.189/0.189/0.189/0.000 ms
h3->h2: 1/1, rtt min/avg/max/mdev 0.124/0.124/0.124/0.000 ms
h3->h4: 1/1, rtt min/avg/max/mdev 1.148/1.148/1.148/0.000 ms
h3->h5: 1/1, rtt min/avg/max/mdev 0.285/0.285/0.285/0.000 ms
h3->h6: 1/1, rtt min/avg/max/mdev 1.109/1.109/1.109/0.000 ms
h4->h1: 1/1, rtt min/avg/max/mdev 0.633/0.633/0.633/0.000 ms
h4->h2: 1/1, rtt min/avg/max/mdev 0.679/0.679/0.679/0.000 ms
h4->h3: 1/1, rtt min/avg/max/mdev 0.684/0.684/0.684/0.000 ms
h4->h5: 1/1, rtt min/avg/max/mdev 4.133/4.133/4.133/0.000 ms
h4->h6: 1/1, rtt min/avg/max/mdev 0.259/0.259/0.259/0.000 ms
h5->h1: 1/1, rtt min/avg/max/mdev 0.777/0.777/0.777/0.000 ms
h5->h2: 1/1, rtt min/avg/max/mdev 0.404/0.404/0.404/0.000 ms
```

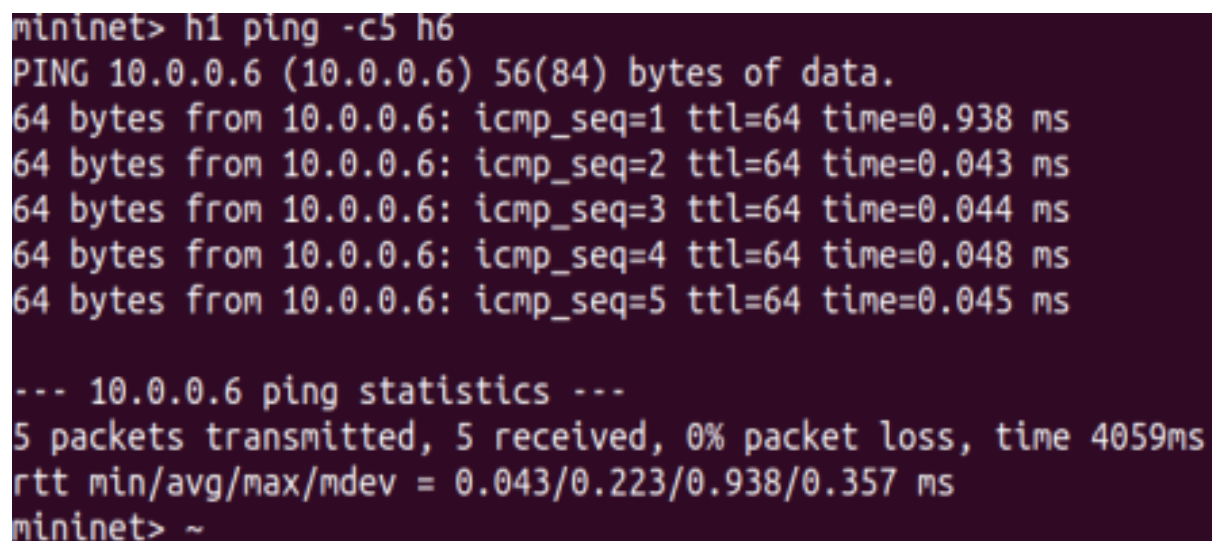
Average value of RTT is calculated to be ~0.63 msecs

—Continue snapshot

```
rishabh@rishabh-VirtualBox: ~/Desktop
h1->h2: 1/1, rtt min/avg/max/mdev 0.915/0.915/0.915/0.000 ms
h1->h3: 1/1, rtt min/avg/max/mdev 0.261/0.261/0.261/0.000 ms
h1->h4: 1/1, rtt min/avg/max/mdev 0.338/0.338/0.338/0.000 ms
h1->h5: 1/1, rtt min/avg/max/mdev 0.813/0.813/0.813/0.000 ms
h1->h6: 1/1, rtt min/avg/max/mdev 0.874/0.874/0.874/0.000 ms
h2->h1: 1/1, rtt min/avg/max/mdev 0.201/0.201/0.201/0.000 ms
h2->h3: 1/1, rtt min/avg/max/mdev 0.343/0.343/0.343/0.000 ms
h2->h4: 1/1, rtt min/avg/max/mdev 0.850/0.850/0.850/0.000 ms
h2->h5: 1/1, rtt min/avg/max/mdev 0.295/0.295/0.295/0.000 ms
h2->h6: 1/1, rtt min/avg/max/mdev 0.172/0.172/0.172/0.000 ms
h3->h1: 1/1, rtt min/avg/max/mdev 0.189/0.189/0.189/0.000 ms
h3->h2: 1/1, rtt min/avg/max/mdev 0.124/0.124/0.124/0.000 ms
h3->h4: 1/1, rtt min/avg/max/mdev 1.148/1.148/1.148/0.000 ms
h3->h5: 1/1, rtt min/avg/max/mdev 0.285/0.285/0.285/0.000 ms
h3->h6: 1/1, rtt min/avg/max/mdev 1.109/1.109/1.109/0.000 ms
h4->h1: 1/1, rtt min/avg/max/mdev 0.633/0.633/0.633/0.000 ms
h4->h2: 1/1, rtt min/avg/max/mdev 0.679/0.679/0.679/0.000 ms
h4->h3: 1/1, rtt min/avg/max/mdev 0.684/0.684/0.684/0.000 ms
h4->h5: 1/1, rtt min/avg/max/mdev 4.133/4.133/4.133/0.000 ms
h4->h6: 1/1, rtt min/avg/max/mdev 0.259/0.259/0.259/0.000 ms
h5->h1: 1/1, rtt min/avg/max/mdev 0.777/0.777/0.777/0.000 ms
h5->h2: 1/1, rtt min/avg/max/mdev 0.404/0.404/0.404/0.000 ms
h5->h3: 1/1, rtt min/avg/max/mdev 8.277/8.277/8.277/0.000 ms
h5->h4: 1/1, rtt min/avg/max/mdev 0.280/0.280/0.280/0.000 ms
h5->h6: 1/1, rtt min/avg/max/mdev 0.600/0.600/0.600/0.000 ms
h6->h1: 1/1, rtt min/avg/max/mdev 0.373/0.373/0.373/0.000 ms
h6->h2: 1/1, rtt min/avg/max/mdev 0.398/0.398/0.398/0.000 ms
h6->h3: 1/1, rtt min/avg/max/mdev 0.428/0.428/0.428/0.000 ms
h6->h4: 1/1, rtt min/avg/max/mdev 0.302/0.302/0.302/0.000 ms
h6->h5: 1/1, rtt min/avg/max/mdev 0.158/0.158/0.158/0.000 ms
mininet>
```

## 2) Comment specifically on an RTT over a connection from a server in S00 rack to a server in S01 rack.

```
Command - h1 [host1] ping -c5 [5 packets] h6 [host 6]
```



```
mininet> h1 ping -c5 h6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=0.938 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=0.043 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=0.044 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=0.048 ms
64 bytes from 10.0.0.6: icmp_seq=5 ttl=64 time=0.045 ms

--- 10.0.0.6 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4059ms
rtt min/avg/max/mdev = 0.043/0.223/0.938/0.357 ms
mininet> ~
```

Pinging from Host1 [S00] to Host 6 [S01], the output of RTT is 0.223 msec

## 3) Experience *iperf* a server in S00 rack to a server in S01 rack

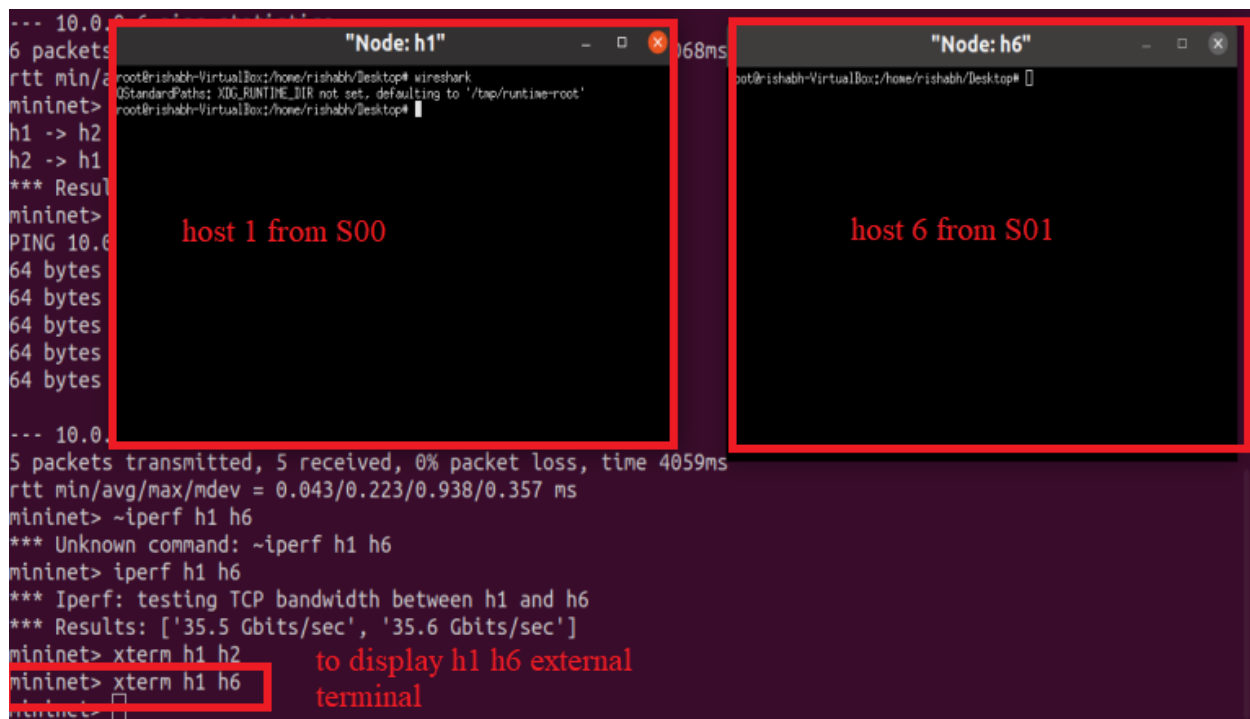
```
Command - iperf [ N/W performance measurement tool] h1 [host1] h6 [host6]
```

```
mininet> iperf h1 h6
*** Iperf: testing TCP bandwidth between h1 and h6
*** Results: ['35.5 Gbits/sec', '35.6 Gbits/sec']
mininet>
```

Its an TCP connection and the result is 35.5 Gbits/sec

#### 4) Wireshark on a server in S00 rack to a server in S01 rack

Command - `xterm h1` [ creates new terminal for host1 CLI ]



Wireshark can be launched for Host 1 to analysis the incoming or outgoing traffic

-Running **dump** to identify IP address of each host and switch

```
mininet> dump mac
<Host h1: h1-eth0:10.0.0.1 pid=9661>
<Host h2: h2-eth0:10.0.0.2 pid=9663>
<Host h3: h3-eth0:10.0.0.3 pid=9665>
<Host h4: h4-eth0:10.0.0.4 pid=9667>
<Host h5: h5-eth0:10.0.0.5 pid=9669>
<Host h6: h6-eth0:10.0.0.6 pid=9671>
<OVSSwitch S00: lo:127.0.0.1,S00-eth1:None,S00-eth2:None,S00-eth3:None,S00-eth4:None pid=9676>
<OVSSwitch S01: lo:127.0.0.1,S01-eth1:None,S01-eth2:None,S01-eth3:None,S01-eth4:None pid=9679>
<Controller c0: 127.0.0.1:6653 pid=9654>
```

**dump** command lets us know the IP address assigned to each host

### -Pinging Host6 [S01] from Host1 [S00]

Command - h1 [host1] ping -c1 [only 1 packet] h6 [host 6]

```
mininet> h1 ping -c1 h6
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=2.47 ms

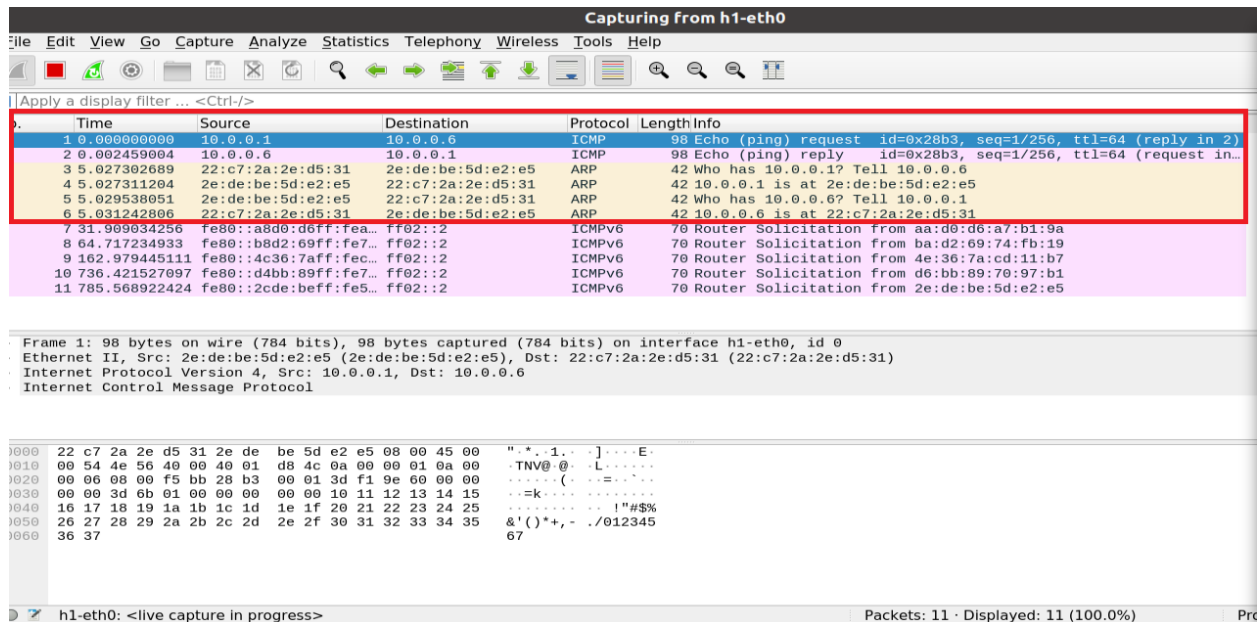
--- 10.0.0.6 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time
0ms
rtt min/avg/max/mdev = 2.474/2.474/2.474/0.000 ms
mininet>
```

host1 =10.0.0.1    host6 =10.0.0.6 | RTT is observed to be 2.474 ms

### Running **wireshark** on host1 terminal

Command - wireshark





## Wireshark

Wireshark is a packet sniffer and analysis tool. It captures network traffic on the local network and stores that data for offline analysis.

### Wireshark Analysis

- 1) Source [host1] sends and **ICMP** packet as a request in search for Destination [host6]
- 2) Source [ host6] replied to the **ICMP** packet to Destination [host1]

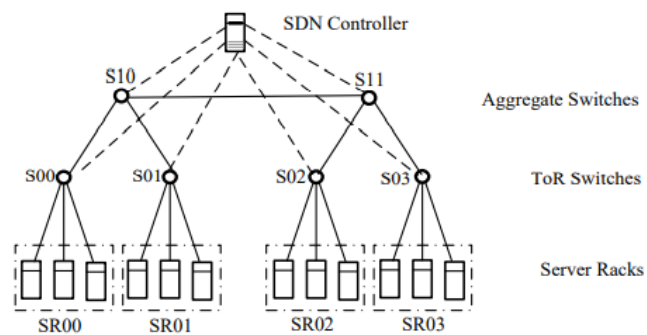
The **Internet Control Message Protocol (ICMP)** is a protocol that devices within a network use to communicate problems with data transmission.

Example (3) (4) (5) (6) all show the **ARP** handshake.

**Address Resolution Protocol (ARP)** is a procedure for mapping a dynamic Internet Protocol address (IP address) to a permanent physical machine address in a local area network (LAN).

## Simulation 2

### Topology



```
CUSTOM TOPOLOGY FOR SIMULATION 2

from mininet.topo import Topo

class MyTopo( Topo ):
    "Simple topology example."

    def build( self ):
        "Create custom topo."

        # Add hosts [Server Racks]
        h1 = self.addHost( 'h1' )
        h2 = self.addHost( 'h2' )
        h3 = self.addHost( 'h3' )
        h4 = self.addHost( 'h4' )
        h5 = self.addHost( 'h5' )
        h6 = self.addHost( 'h6' )
        h7 = self.addHost( 'h7' )
        h8 = self.addHost( 'h8' )
        h9 = self.addHost( 'h9' )
        h10 = self.addHost( 'h10' )
        h11= self.addHost( 'h11' )
        h12= self.addHost( 'h12' )

        #Add switches [ToR switches]
        S00 = self.addSwitch( 'S00' )
        S01 = self.addSwitch( 'S01' )
```

```

S02 = self.addSwitch( 'S02' )
S03 = self.addSwitch( 'S03' )

#Add switches [Aggregate switch]
S10 = self.addSwitch( 'S10' )
S11 = self.addSwitch( 'S11' )

# Add links
self.addLink( h1, S00 )
self.addLink( h2, S00 )
self.addLink( h3, S00 )

self.addLink(h4, S01 )
self.addLink(h5, S01 )
self.addLink(h6, S01 )

self.addLink( h7, S02 )
self.addLink( h8, S02 )
self.addLink( h9, S02 )

self.addLink(h10, S03 )
self.addLink(h11, S03 )
self.addLink(h12, S03 )

self.addLink( S10, S00 )
self.addLink( S10, S01 )

self.addLink( S11, S02 )
self.addLink( S11, S03 )

self.addLink( S10, S11 )

topos = { 'mytopo': ( lambda: MyTopo() ) }

```

Custom code written in python language via gedit text editor.

- Initialized the number of host [total of 12] where **SR00** contains [ h1-h3 ], **SR01** contains [ h4-h6 ], **SR02** contains [ h7-h9 ], **SR03** contains [ h10-h12 ].
- Initialized **Tor Switches** from S00-S03.
- Initialized **Aggregate Switches** - S10 & S11.
- Initialized the links between the host and switches.

## Basic Setup of the mininet by introducing custom topology with a default controller

```
rishabh@rishabh-VirtualBox: ~/Desktop
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
*** Adding switches:
S00 S01 S02 S03 S10 S11
*** Adding links:
(S10, S00) (S10, S01) (S10, S11) (S11, S02) (S11, S03) (h1, S00) (h2, S00) (h3, S00) (h4, S01) (h5, S01) (h6, S01)
(h7, S02) (h8, S02) (h9, S02) (h10, S03) (h11, S03) (h12, S03)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
*** Starting controller
c0
*** Starting 6 switches
S00 S01 S02 S03 S10 S11 ...
*** Starting CLI:
mininet> nodes
available nodes are:
S00 S01 S02 S03 S10 S11 c0 h1 h10 h11 h12 h2 h3 h4 h5 h6 h7 h8 h9
mininet>
```

### 1) Conduct a test using Pingall command

Command - pingall

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10 h11 h12
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10 h11 h12
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10 h11 h12
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10 h11 h12
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10 h11 h12
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10 h11 h12
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10 h11 h12
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h11 h12
h11 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h12
h12 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11
*** Results: 0% dropped (132/132 received)
mininet>
```

None of the packets dropped means the connection is stable

- Calculating the **Average round trip time (RTT)**

Command - pingallfull

```
mininet> pingallfull
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10 h11 h12
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10 h11 h12
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10 h11 h12
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10 h11 h12
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10 h11 h12
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10 h11 h12
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10 h11 h12
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h11 h12
h11 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h12
h12 -> h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11
*** Results:
h1->h2: 1/1, rtt min/avg/max/mdev 1.648/1.648/1.648/0.000 ms
h1->h3: 1/1, rtt min/avg/max/mdev 1.962/1.962/1.962/0.000 ms
h1->h4: 1/1, rtt min/avg/max/mdev 9.222/9.222/9.222/0.000 ms
h1->h5: 1/1, rtt min/avg/max/mdev 4.954/4.954/4.954/0.000 ms
h1->h6: 1/1, rtt min/avg/max/mdev 24.081/24.081/24.081/0.000 ms
h1->h7: 1/1, rtt min/avg/max/mdev 3.720/3.720/3.720/0.000 ms
h1->h8: 1/1, rtt min/avg/max/mdev 3.132/3.132/3.132/0.000 ms
h1->h9: 1/1, rtt min/avg/max/mdev 3.080/3.080/3.080/0.000 ms
h1->h10: 1/1, rtt min/avg/max/mdev 2.994/2.994/2.994/0.000 ms
h1->h11: 1/1, rtt min/avg/max/mdev 5.188/5.188/5.188/0.000 ms
h1->h12: 1/1, rtt min/avg/max/mdev 11.421/11.421/11.421/0.000 ms
h2->h1: 1/1, rtt min/avg/max/mdev 1.658/1.658/1.658/0.000 ms
h2->h3: 1/1, rtt min/avg/max/mdev 1.535/1.535/1.535/0.000 ms
h2->h4: 1/1, rtt min/avg/max/mdev 3.680/3.680/3.680/0.000 ms
h2->h5: 1/1, rtt min/avg/max/mdev 13.620/13.620/13.620/0.000 ms
h2->h6: 1/1, rtt min/avg/max/mdev 7.188/7.188/7.188/0.000 ms
h2->h7: 1/1, rtt min/avg/max/mdev 5.610/5.610/5.610/0.000 ms
```

Average value of RTT is calculated to be ~0.63 msecs

**2) Comment specifically on an RTT over a connection from a server in S00 rack to a server in S03 rack.**

Command - h1 [host1] ping -c5 [5 packets] h12 [host 12]

```
mininet> h1 ping -c5 h12
PING 10.0.0.12 (10.0.0.12) 56(84) bytes of data.
64 bytes from 10.0.0.12: icmp_seq=1 ttl=64 time=9.22 ms
64 bytes from 10.0.0.12: icmp_seq=2 ttl=64 time=2.06 ms
64 bytes from 10.0.0.12: icmp_seq=3 ttl=64 time=0.506 ms
64 bytes from 10.0.0.12: icmp_seq=4 ttl=64 time=0.049 ms
64 bytes from 10.0.0.12: icmp_seq=5 ttl=64 time=0.069 ms

--- 10.0.0.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4036ms
rtt min/avg/max/mdev = 0.049/2.379/9.215/3.495 ms
```

Ping from Host1 [S00] to Host 12 [S03], the output of RTT is 2.379 msecs

### 3) Experience *iperf* a server in S00 rack to a server in S03 rack

Command - `iperf [ N/W performance measurement tool] h1 [host1] h12 [host12]`

```
mininet> iperf h1 h12
*** Iperf: testing TCP bandwidth between h1 and h12
*** Results: ['25.8 Gbits/sec', '26.8 Gbits/sec']
mininet>
```

Its an TCP connection and the result is 25.8 Gbits/sec [ the bandwidth has dropped compared to simulation 1]

### 4) Wireshark on a server in S00 rack to a server in S03 rack

-Running dump to identify IP address of each host and switch

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=10671>
<Host h2: h2-eth0:10.0.0.2 pid=10673>
<Host h3: h3-eth0:10.0.0.3 pid=10675>
<Host h4: h4-eth0:10.0.0.4 pid=10677>
<Host h5: h5-eth0:10.0.0.5 pid=10679>
<Host h6: h6-eth0:10.0.0.6 pid=10681>
<Host h7: h7-eth0:10.0.0.7 pid=10683>
<Host h8: h8-eth0:10.0.0.8 pid=10685>
<Host h9: h9-eth0:10.0.0.9 pid=10687>
<Host h10: h10-eth0:10.0.0.10 pid=10689>
<Host h11: h11-eth0:10.0.0.11 pid=10691>
<Host h12: h12-eth0:10.0.0.12 pid=10693>
<OVSSwitch S00: lo:127.0.0.1,S00-eth1:None,S00-eth2:None,S00-eth3:None,S00-eth4:None pid=10698>
<OVSSwitch S01: lo:127.0.0.1,S01-eth1:None,S01-eth2:None,S01-eth3:None,S01-eth4:None pid=10701>
<OVSSwitch S02: lo:127.0.0.1,S02-eth1:None,S02-eth2:None,S02-eth3:None,S02-eth4:None pid=10704>
<OVSSwitch S03: lo:127.0.0.1,S03-eth1:None,S03-eth2:None,S03-eth3:None,S03-eth4:None pid=10707>
<OVSSwitch S10: lo:127.0.0.1,S10-eth1:None,S10-eth2:None,S10-eth3:None pid=10710>
<OVSSwitch S11: lo:127.0.0.1,S11-eth1:None,S11-eth2:None,S11-eth3:None pid=10713>
<Controller c0: 127.0.0.1:6653 pid=10664>
mininet>
```

## -Pinging Host12 from Host1

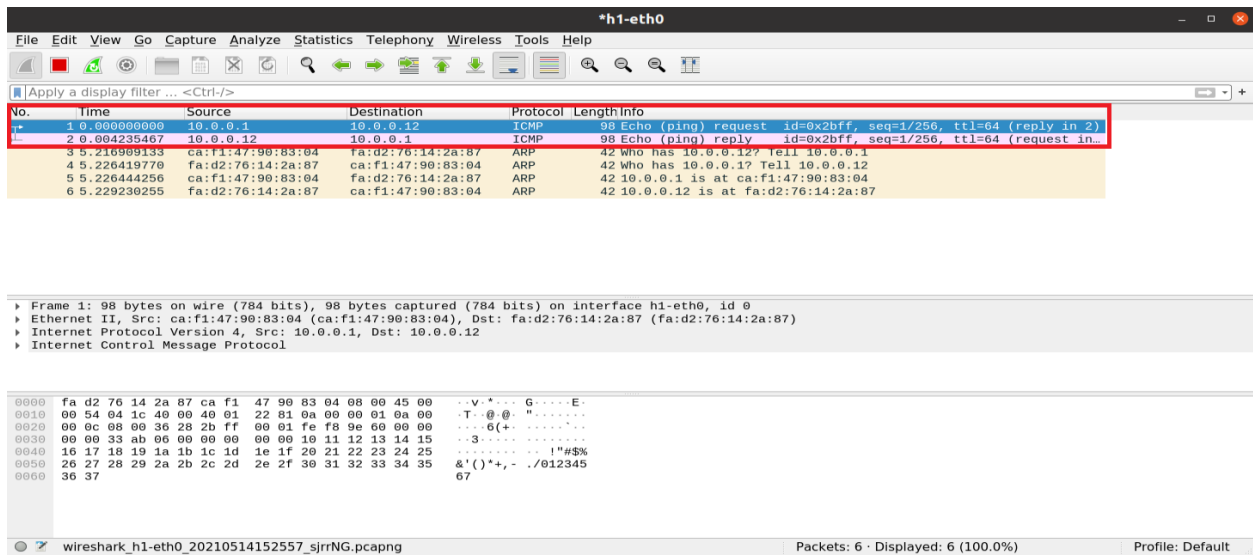
```
mininet> h1 ping -c1 h12
PING 10.0.0.12 (10.0.0.12) 56(84) bytes of data.
64 bytes from 10.0.0.12: icmp_seq=1 ttl=64 time=4.25 ms

--- 10.0.0.12 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 4.251/4.251/4.251/0.000 ms
mininet>
```

host 1 =10.0.0.1 host12 = 10.0.0.12 | RTT is observed to be 4.251 msec

## Running wireshark on host1 terminal

Command - wireshark



## Wireshark

Wireshark is a packet sniffer and analysis tool. It captures network traffic on the local network and stores that data for offline analysis.

## Wireshark Analysis

1) Source [host1] sends and **ICMP** packet as a request in search for Destination [host12]

2) Source [ host12] replied to the **ICMP** packet to Destination [host1]

The **Internet Control Message Protocol (ICMP)** is a protocol that devices within a network use to communicate problems with data transmission.

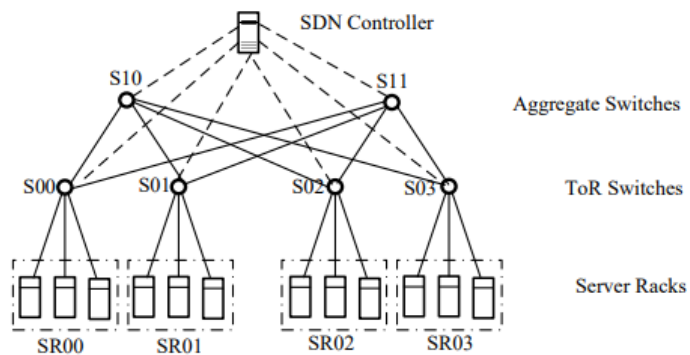
Example (3) (4) (5) (6) all show the **ARP** handshake.

**Address Resolution Protocol (ARP)** is a procedure for mapping a dynamic Internet Protocol address (IP address) to a permanent physical machine address in a local area network (LAN).



## Simulation 3

### Topology



```
CUSTOM TOPOLOGY FOR SIMULATION 3

from mininet.topo import Topo

class MyTopo( Topo ):
    "Simple topology example."

    def build( self ):
        "Create custom topo."

        # Add hosts [Server Racks]
        h1 = self.addHost( 'h1' )
        h2 = self.addHost( 'h2' )
        h3 = self.addHost( 'h3' )
        h4 = self.addHost( 'h4' )
        h5 = self.addHost( 'h5' )
        h6 = self.addHost( 'h6' )
        h7 = self.addHost( 'h7' )
        h8 = self.addHost( 'h8' )
        h9 = self.addHost( 'h9' )
        h10 = self.addHost( 'h10' )
        h11= self.addHost( 'h11' )
        h12= self.addHost( 'h12' )
```

```

#Add switches [ToR switches]
S00 = self.addSwitch( 'S00' )
S01 = self.addSwitch( 'S01' )
S02 = self.addSwitch( 'S02' )
S03 = self.addSwitch( 'S03' )

#Add switches [Aggregate switch]
S10 = self.addSwitch( 'S10' )
S11 = self.addSwitch( 'S11' )

# Add links
self.addLink( h1, S00 )
self.addLink( h2, S00 )
self.addLink( h3, S00 )

self.addLink(h4, S01 )
self.addLink(h5, S01 )
self.addLink(h6, S01 )

self.addLink( h7, S02 )
self.addLink( h8, S02 )
self.addLink( h9, S02 )

self.addLink(h10, S03 )
self.addLink(h11, S03 )
self.addLink(h12, S03 )

self.addLink( S10, S00 )
self.addLink( S10, S01 )
self.addLink( S10, S02 )
self.addLink( S10, S03 )

self.addLink( S11, S00 )
self.addLink( S11, S01 )
self.addLink( S11, S02 )
self.addLink( S11, S03 )

topos = { 'mytopo': ( lambda: MyTopo() ) }

```

Custom code written in python language via gedit text editor.

- Initialized the number of host [total of 12] where **SR00** contains [ h1-h3 ], **SR01** contains [ h4-h6 ], **SR02** contains [ h7-h9 ], **SR03** contains [ h10-h12 ].
- Initialized **Tor Switches** from S00-S03.

- Initialized **Aggregate Switches** - S10 & S11.
- Initialized the links between the host and switches but there are **no links** between two switches.

## SETUP

### 1) Initializing POX controller.

```
rishabh@rishabh-VirtualBox:~$ cd pox
rishabh@rishabh-VirtualBox:~/pox$ ./pox.py forwarding.l2_learning
POX 0.7.0 (gar) / Copyright 2011-2020 James McCauley, et al.
WARNING:version:Support for Python 3 is experimental.
INFO:core:POX 0.7.0 (gar) is up.
```

POX controller comes by default with the mininet installer.

Pox controller is found to be by default installed in the root directory.

**POX** is a Python based open source OpenFlow/Software Defined Networking (SDN) Controller. POX is used for faster development and prototyping of new network applications.

### 2) Running up the topology

```
Command- sudo mn -custom sim3.py [filename] --topo mytopo [topology name]
```

```
rishabh@rishabh-VirtualBox: ~/Desktop
rishabh@rishabh-VirtualBox:~/Desktop$ sudo python ./router3.py
*** Adding controller
*** Add switches
*** Add hosts
*** Add links
*** Starting network
*** Configuring hosts
h3 h2 h6 h5 h7 h9 h10 h8 h4 h12 h11 h1
*** Starting controllers
*** Starting switches
*** Post configure switches and hosts
*** Starting CLI:
mininet> pingall
```

## 1) Conduct a test using Pingall command

Command - pingall

```
mininet> pingall
*** Ping: testing ping reachability
h3 -> h2 h6 h5 h7 h9 h10 h8 h4 h12 h11 h1
h2 -> h3 h6 h5 h7 h9 h10 h8 h4 h12 h11 h1
h6 -> h3 h2 h5 h7 h9 h10 h8 h4 h12 h11 h1
h5 -> h3 h2 h6 h7 h9 h10 h8 h4 h12 h11 h1
h7 -> h3 h2 h6 h5 h9 h10 h8 h4 h12 h11 h1
h9 -> h3 h2 h6 h5 h7 h10 h8 h4 h12 h11 h1
h10 -> h3 h2 h6 h5 h7 h9 h8 h4 h12 h11 h1
h8 -> h3 h2 h6 h5 h7 h9 h10 h4 h12 h11 h1
h4 -> h3 h2 h6 h5 h7 h9 h10 h8 h12 h11 h1
h12 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h11 h1
h11 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h12 h1
h1 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h12 h11
*** Results: 0% dropped (132/132 received)
mininet> 
```

- Calculating the **Average round trip time (RTT)**

Command- pingallfull

```

mininet> pingallfull
*** Ping: testing ping reachability
h3 -> h2 h6 h5 h7 h9 h10 h8 h4 h12 h11 h1
h2 -> h3 h6 h5 h7 h9 h10 h8 h4 h12 h11 h1
h6 -> h3 h2 h5 h7 h9 h10 h8 h4 h12 h11 h1
h5 -> h3 h2 h6 h7 h9 h10 h8 h4 h12 h11 h1
h7 -> h3 h2 h6 h5 h9 h10 h8 h4 h12 h11 h1
h9 -> h3 h2 h6 h5 h7 h10 h8 h4 h12 h11 h1
h10 -> h3 h2 h6 h5 h7 h9 h8 h4 h12 h11 h1
h8 -> h3 h2 h6 h5 h7 h9 h10 h4 h12 h11 h1
h4 -> h3 h2 h6 h5 h7 h9 h10 h8 h12 h11 h1
h12 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h11 h1
h11 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h12 h1
h1 -> h3 h2 h6 h5 h7 h9 h10 h8 h4 h12 h11
*** Results:
h3->h2: 1/1, rtt min/avg/max/mdev 2.683/2.683/2.683/0.000 ms
h3->h6: 1/1, rtt min/avg/max/mdev 6.433/6.433/6.433/0.000 ms
h3->h5: 1/1, rtt min/avg/max/mdev 5.763/5.763/5.763/0.000 ms
h3->h7: 1/1, rtt min/avg/max/mdev 18.758/18.758/18.758/0.000 ms
h3->h9: 1/1, rtt min/avg/max/mdev 15.546/15.546/15.546/0.000 ms
h3->h10: 1/1, rtt min/avg/max/mdev 12.047/12.047/12.047/0.000 ms
h3->h8: 1/1, rtt min/avg/max/mdev 18.130/18.130/18.130/0.000 ms
h3->h4: 1/1, rtt min/avg/max/mdev 7.218/7.218/7.218/0.000 ms
h3->h12: 1/1, rtt min/avg/max/mdev 15.559/15.559/15.559/0.000 ms
h3->h11: 1/1, rtt min/avg/max/mdev 11.848/11.848/11.848/0.000 ms
h3->h1: 1/1, rtt min/avg/max/mdev 2.962/2.962/2.962/0.000 ms
h2->h3: 1/1, rtt min/avg/max/mdev 3.937/3.937/3.937/0.000 ms
h2->h6: 1/1, rtt min/avg/max/mdev 7.125/7.125/7.125/0.000 ms
h2->h5: 1/1, rtt min/avg/max/mdev 10.181/10.181/10.181/0.000 ms
h2->h7: 1/1, rtt min/avg/max/mdev 13.585/13.585/13.585/0.000 ms
h2->h9: 1/1, rtt min/avg/max/mdev 18.834/18.834/18.834/0.000 ms

```

**2) Comment specifically on an RTT over a connection from a server in S00 rack to a server in S03 rack.**

```
Command- h1 [host1] ping -c5 [5packets] h12 [host12]
```

```

mininet> h1 ping -c2 h12
PING 10.0.0.12 (10.0.0.12) 56(84) bytes of data.
64 bytes from 10.0.0.12: icmp_seq=1 ttl=64 time=8.77 ms
64 bytes from 10.0.0.12: icmp_seq=2 ttl=64 time=0.527 ms

--- 10.0.0.12 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 0.527/4.647/8.768/4.120 ms
mininet>

```

### 3) Experience *iperf* a server in S00 rack to a server in S03 rack

Command - `iperf [N/W performance measurement tool] h1 [host1] h12 [host12]`

```
mininet> iperf h1 h12
*** Iperf: testing TCP bandwidth between h1 and h12
*** Results: ['34.4 Gbits/sec', '34.4 Gbits/sec']
mininet>
```

### 4) Wireshark on a server in S00 rack to a server in S03 rack

-Running dump to identify IP address of each host and switch

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=10671>
<Host h2: h2-eth0:10.0.0.2 pid=10673>
<Host h3: h3-eth0:10.0.0.3 pid=10675>
<Host h4: h4-eth0:10.0.0.4 pid=10677>
<Host h5: h5-eth0:10.0.0.5 pid=10679>
<Host h6: h6-eth0:10.0.0.6 pid=10681>
<Host h7: h7-eth0:10.0.0.7 pid=10683>
<Host h8: h8-eth0:10.0.0.8 pid=10685>
<Host h9: h9-eth0:10.0.0.9 pid=10687>
<Host h10: h10-eth0:10.0.0.10 pid=10689>
<Host h11: h11-eth0:10.0.0.11 pid=10691>
<Host h12: h12-eth0:10.0.0.12 pid=10693>
<OVSSwitch S00: lo:127.0.0.1,S00-eth1:None,S00-eth2:None,S00-eth3:None,S00-eth4:None pid=10698>
<OVSSwitch S01: lo:127.0.0.1,S01-eth1:None,S01-eth2:None,S01-eth3:None,S01-eth4:None pid=10701>
<OVSSwitch S02: lo:127.0.0.1,S02-eth1:None,S02-eth2:None,S02-eth3:None,S02-eth4:None pid=10704>
<OVSSwitch S03: lo:127.0.0.1,S03-eth1:None,S03-eth2:None,S03-eth3:None,S03-eth4:None pid=10707>
<OVSSwitch S10: lo:127.0.0.1,S10-eth1:None,S10-eth2:None,S10-eth3:None pid=10710>
<OVSSwitch S11: lo:127.0.0.1,S11-eth1:None,S11-eth2:None,S11-eth3:None pid=10713>
<Controller c0: 127.0.0.1:6653 pid=10664>
mininet>
```

-Pinging Host12 from Host1

```

mininet> xterm h1
mininet> h1 ping -c5 h12
PING 10.0.0.12 (10.0.0.12) 56(84) bytes of data.
64 bytes from 10.0.0.12: icmp_seq=1 ttl=64 time=14.4 ms
64 bytes from 10.0.0.12: icmp_seq=2 ttl=64 time=0.513 ms
64 bytes from 10.0.0.12: icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from 10.0.0.12: icmp_seq=4 ttl=64 time=0.056 ms
64 bytes from 10.0.0.12: icmp_seq=5 ttl=64 time=0.052 ms

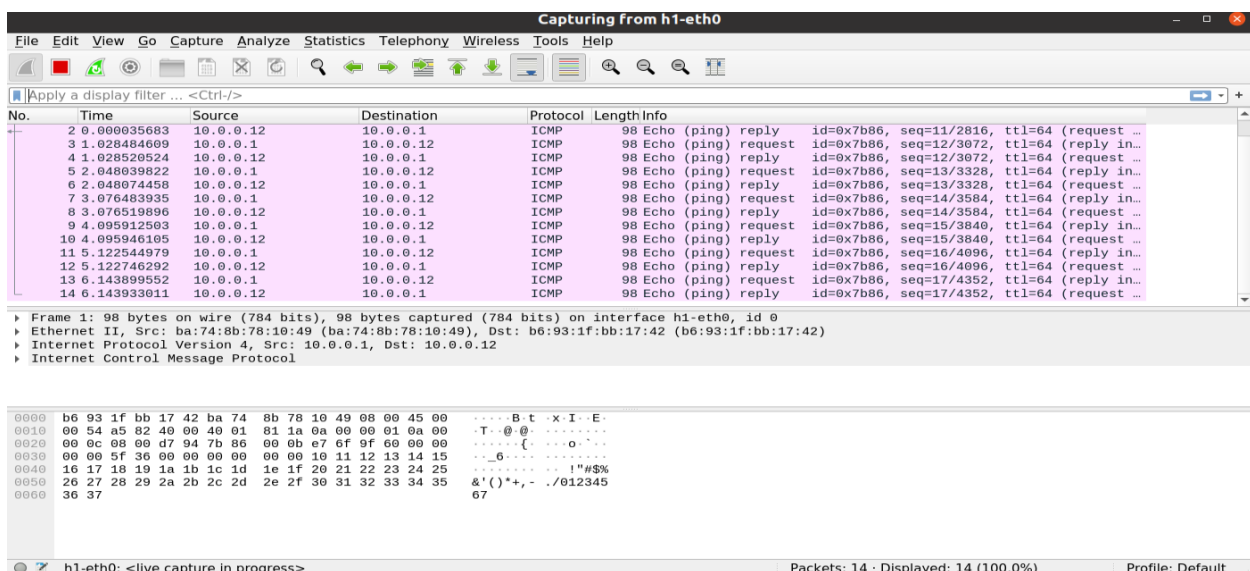
--- 10.0.0.12 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4040ms
rtt min/avg/max/mdev = 0.052/3.008/14.356/5.676 ms
mininet>

```

host 1 = 10.0.0.1 host12 = 10.0.0.12 | RTT is observed to be 4.251 msec

## Running wireshark on host1 terminal

Command - wireshark



## Wireshark

Wireshark is a packet sniffer and analysis tool. It captures network traffic on the local network and stores that data for offline analysis.

## Wireshark Analysis

- 1) Source [host1] sends and **ICMP** packet as a request in search for Destination [host12]
- 2) Source [ host12] replied to the **ICMP** packet to Destination [host1]

The **Internet Control Message Protocol (ICMP)** is a protocol that devices within a network use to communicate problems with data transmission.