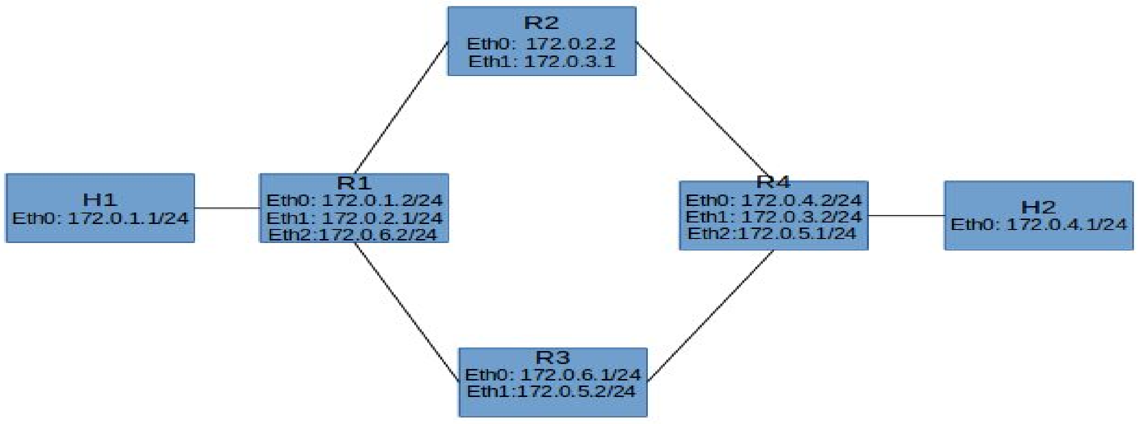
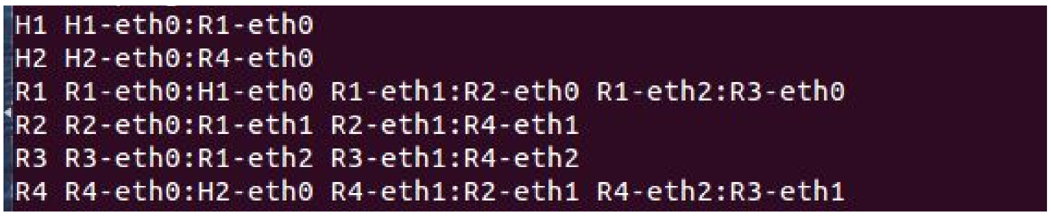
Part A

A1:

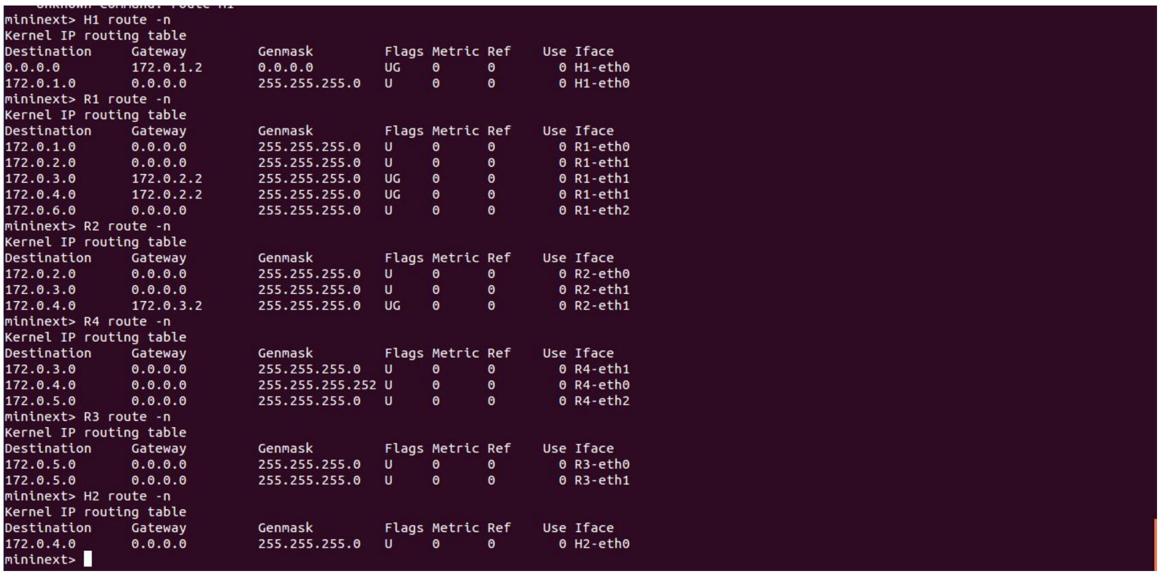


Connection Details:



```
H1 H1-eth0:R1-eth0
H2 H2-eth0:R4-eth0
R1 R1-eth0:H1-eth0 R1-eth1:R2-eth0 R1-eth2:R3-eth0
R2 R2-eth0:R1-eth1 R2-eth1:R4-eth1
R3 R3-eth0:R1-eth2 R3-eth1:R4-eth2
R4 R4-eth0:H2-eth0 R4-eth1:R2-eth1 R4-eth2:R3-eth1
```

A2:the routing tables at all nodes (as a screen capture) and explain what you did to configure them correctly.

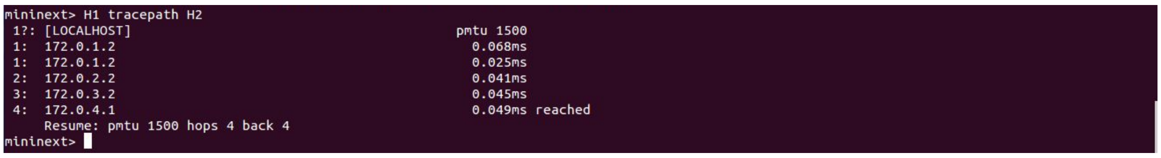


```
mininext> H1 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.0.1.2       0.0.0.0         UG    0      0        0 H1-eth0
172.0.1.0       0.0.0.0         255.255.255.0   U     0      0        0 H1-eth0
mininext> R1 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       0.0.0.0         255.255.255.0   U     0      0        0 R1-eth0
172.0.2.0       0.0.0.0         255.255.255.0   U     0      0        0 R1-eth1
172.0.3.0       172.0.2.2       255.255.255.0   UG    0      0        0 R1-eth1
172.0.4.0       172.0.2.2       255.255.255.0   UG    0      0        0 R1-eth1
172.0.6.0       0.0.0.0         255.255.255.0   U     0      0        0 R1-eth2
mininext> R2 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.2.0       0.0.0.0         255.255.255.0   U     0      0        0 R2-eth0
172.0.3.0       0.0.0.0         255.255.255.0   U     0      0        0 R2-eth1
172.0.4.0       172.0.3.2       255.255.255.0   UG    0      0        0 R2-eth1
mininext> R4 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.3.0       0.0.0.0         255.255.255.0   U     0      0        0 R4-eth1
172.0.4.0       0.0.0.0         255.255.255.252 U     0      0        0 R4-eth0
172.0.5.0       0.0.0.0         255.255.255.0   U     0      0        0 R4-eth2
mininext> R3 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.5.0       0.0.0.0         255.255.255.0   U     0      0        0 R3-eth0
172.0.5.0       0.0.0.0         255.255.255.0   U     0      0        0 R3-eth1
mininext> H2 route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.4.0       0.0.0.0         255.255.255.0   U     0      0        0 H2-eth0
mininext> 
```

The above image shows the routing table at each node.

Firstly I added the ip forwarding enable signal in the start.py file :
        net.get("R1").cmd("sysctl -w net.ipv4.ip_forward=1")
To configure the routes at each node I used the command
        ip route add “destination IP address” “subnet” via “IP address” dev “interface”
Next step which was used was to enable the Nating functionality at the nodes in the path and setting some permissions which was done using the below mentioned commands :
        net.get("R1").cmd("iptables -t nat -A POSTROUTING -o R1-eth1 -j MASQUERADE")
        net.get("R1").cmd("iptables -A FORWARD -i R1-eth1 -o R1-eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT")
        net.get("R1").cmd("iptables -A FORWARD -i R1-eth0 -o R1-eth1 -j ACCEPT")

(b) provide the trace - route output that gives the path between nodes H1 & H2.


```
mininext> H1 tracepath H2
 1?: [LOCALHOST]                                     pmtu 1500
 1:  172.0.1.2                                         0.068ms
 1:  172.0.1.2                                         0.025ms
 2:  172.0.2.2                                         0.041ms
 3:  172.0.3.2                                         0.045ms
 4:  172.0.4.1                                         0.049ms reached
     Resume: pmtu 1500 hops 4 back 4
mininext>
```

Part B:

B1: First remove the static routes. In this part you configure R1, R2, R3 and R4 as a RIP router. You need to configure Quagga ripd daemon in order to construct the routes. You will have to change the myfolders/config/ files to do so. Submit: (a) the set of commands you used for the configuration in the correct order. (b) Explanation of each command.

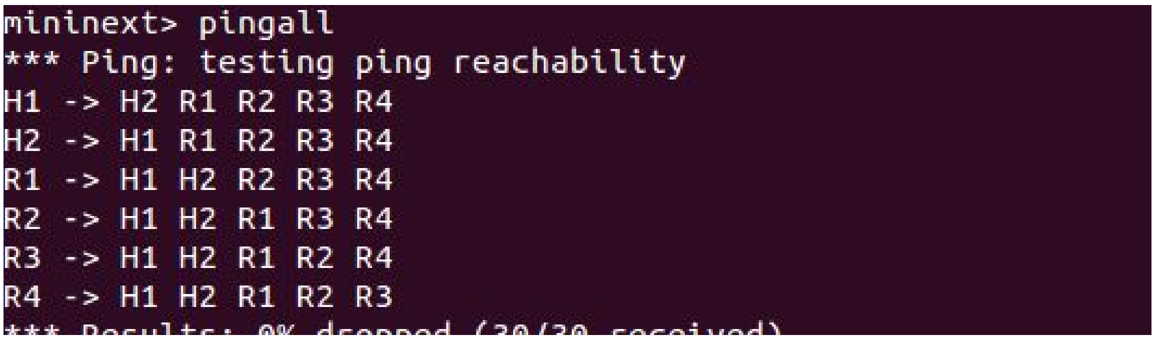
Solution:
First step included installing quagga.
Next, I changed the daemon file in each router to enable zebra and ripd and disabled bgpd.
Next, I added the subnets for each hosts in the ripd.conf file of their corresponding folders.
After doing the necessary changes i opened another SSH terminal to start quagga services using “sudo services quagga start” command.
Next i run the start.py file.
After running the file, pingall command is used to verify all to all connectivity.

The commands used are :

ripd = yes , zebra = yes , and bpgd = false ; basically this command is used to enable ripd and zebra daemon and disable bpgd daemon.

In the ripd.conf files I just added the host names and then added the subnet ips in the corresponding ripd files.

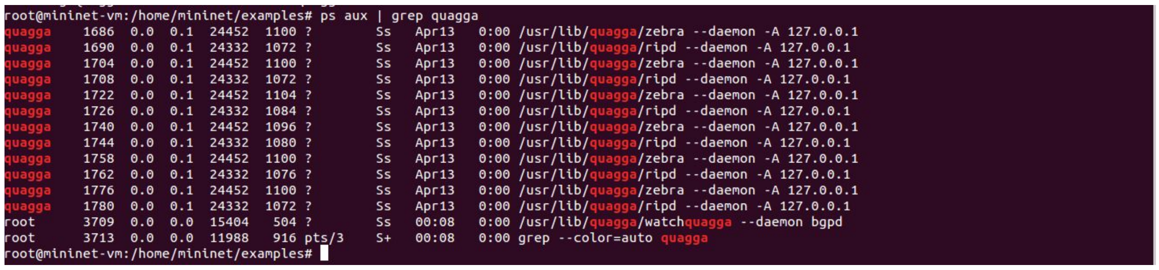
Sudo service quagga start command begins the quagga service and helps to dynamically set the routes for each node using rip protocol.



```
mininext> pingall
*** Ping: testing ping reachability
H1 -> H2 R1 R2 R3 R4
H2 -> H1 R1 R2 R3 R4
R1 -> H1 H2 R2 R3 R4
R2 -> H1 H2 R1 R3 R4
R3 -> H1 H2 R1 R2 R4
R4 -> H1 H2 R1 R2 R3
```

B2: Run the RIP daemon on each router and host. Estimate the convergence time: this is the time it takes for H1 to be able to ping H2. Submit: (a) the routing tables at each node (both the kernel and the Quagga routing table), (b) the traceroute output that gives the path between nodes H1 & H2, and (c) the time taken for the ping, and (d) the convergence time

Ans:

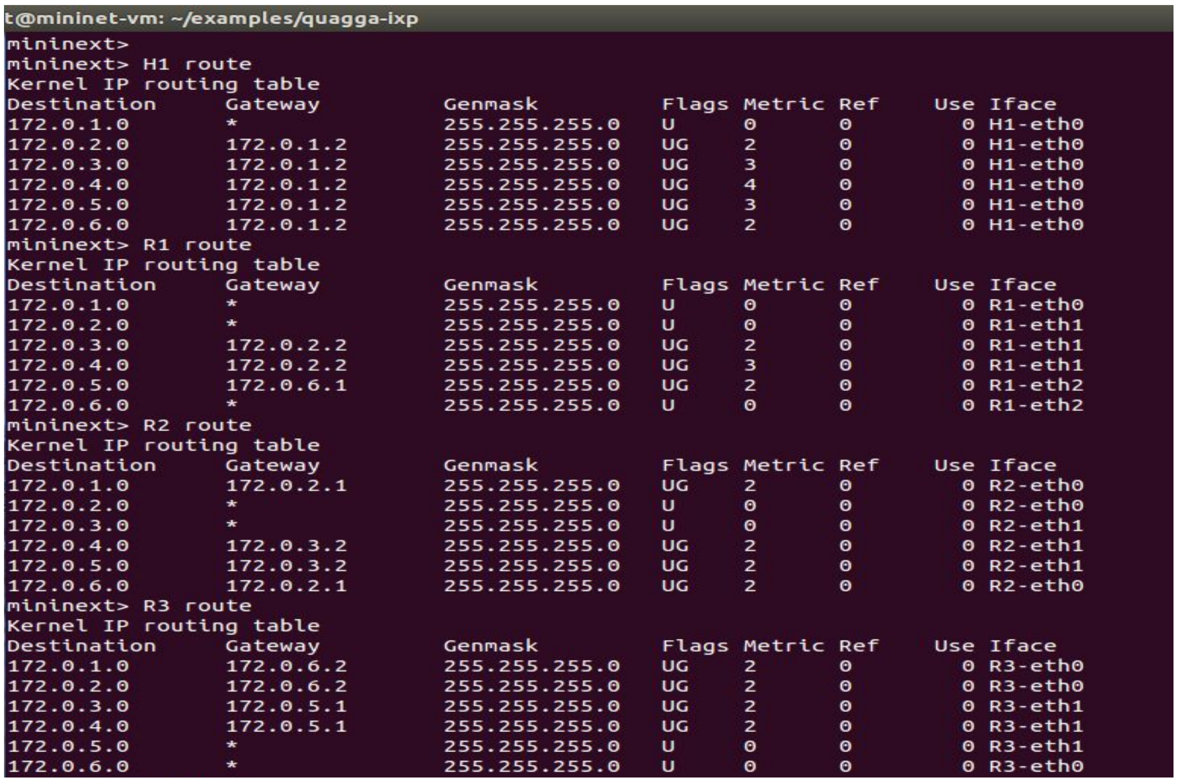


```
root@mininet-vm:/home/mininet/examples# ps aux | grep quagga
quagga    1686  0.0  0.1  24452  1100 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1690  0.0  0.1  24332  1072 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
quagga    1704  0.0  0.1  24452  1100 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1708  0.0  0.1  24332  1072 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
quagga    1722  0.0  0.1  24452  1104 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1726  0.0  0.1  24332  1084 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
quagga    1740  0.0  0.1  24452  1096 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1744  0.0  0.1  24332  1080 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
quagga    1758  0.0  0.1  24452  1100 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1762  0.0  0.1  24332  1076 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
quagga    1776  0.0  0.1  24452  1100 ?        Ss   Apr13   0:00 /usr/lib/quagga/zebra --daemon -A 127.0.0.1
quagga    1780  0.0  0.1  24332  1072 ?        Ss   Apr13   0:00 /usr/lib/quagga/ripd --daemon -A 127.0.0.1
root      3709  0.0  0.0  15404   504 ?        Ss   00:08   0:00 /usr/lib/quagga/watchquagga --daemon bgpd
root      3713  0.0  0.0  11988   916 pts/3    S+   00:08   0:00 grep --color=auto quagga
root@mininet-vm:/home/mininet/examples#
```

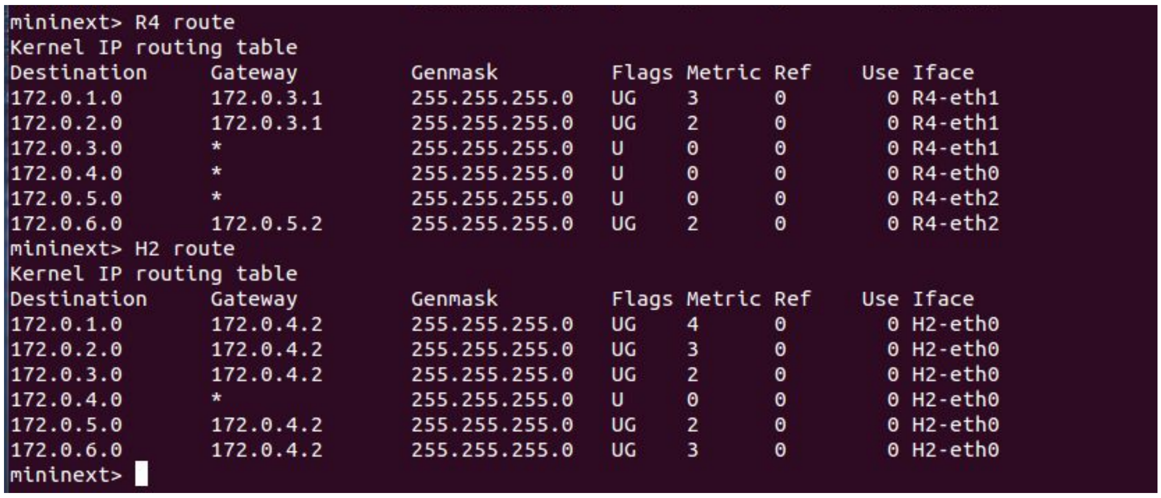
The above figure shows the daemons running.

Routing tables at Nodes:

Linux Kernel output:-

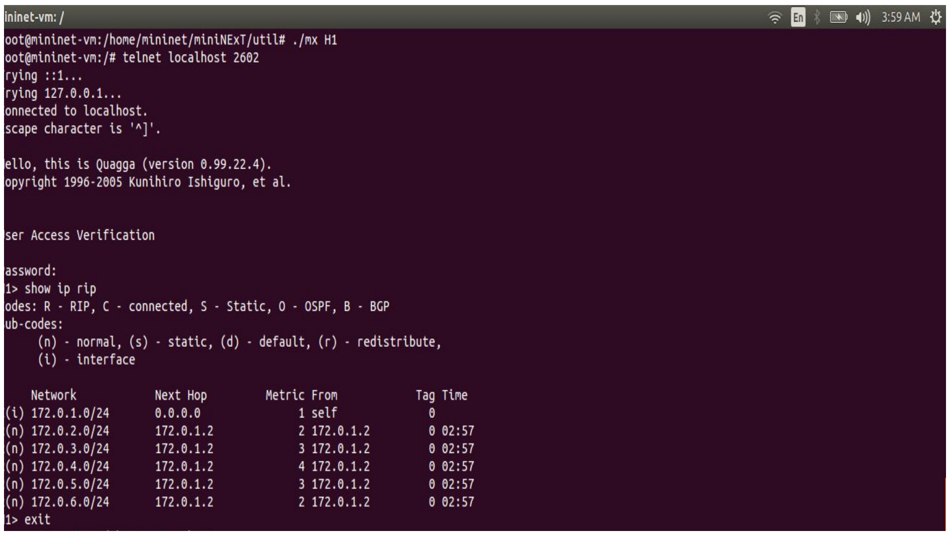


```
t@mininet-vm: ~/examples/quagga-ixp
mininext>
mininext> H1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       *               255.255.255.0   U     0      0        0 H1-eth0
172.0.2.0       172.0.1.2       255.255.255.0   UG    2      0        0 H1-eth0
172.0.3.0       172.0.1.2       255.255.255.0   UG    3      0        0 H1-eth0
172.0.4.0       172.0.1.2       255.255.255.0   UG    4      0        0 H1-eth0
172.0.5.0       172.0.1.2       255.255.255.0   UG    3      0        0 H1-eth0
172.0.6.0       172.0.1.2       255.255.255.0   UG    2      0        0 H1-eth0
mininext> R1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       *               255.255.255.0   U     0      0        0 R1-eth0
172.0.2.0       *               255.255.255.0   U     0      0        0 R1-eth1
172.0.3.0       172.0.2.2       255.255.255.0   UG    2      0        0 R1-eth1
172.0.4.0       172.0.2.2       255.255.255.0   UG    3      0        0 R1-eth1
172.0.5.0       172.0.6.1       255.255.255.0   UG    2      0        0 R1-eth2
172.0.6.0       *               255.255.255.0   U     0      0        0 R1-eth2
mininext> R2 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       172.0.2.1       255.255.255.0   UG    2      0        0 R2-eth0
172.0.2.0       *               255.255.255.0   U     0      0        0 R2-eth0
172.0.3.0       *               255.255.255.0   U     0      0        0 R2-eth1
172.0.4.0       172.0.3.2       255.255.255.0   UG    2      0        0 R2-eth1
172.0.5.0       172.0.3.2       255.255.255.0   UG    2      0        0 R2-eth1
172.0.6.0       172.0.2.1       255.255.255.0   UG    2      0        0 R2-eth0
mininext> R3 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       172.0.6.2       255.255.255.0   UG    2      0        0 R3-eth0
172.0.2.0       172.0.6.2       255.255.255.0   UG    2      0        0 R3-eth0
172.0.3.0       172.0.5.1       255.255.255.0   UG    2      0        0 R3-eth1
172.0.4.0       172.0.5.1       255.255.255.0   UG    2      0        0 R3-eth1
172.0.5.0       *               255.255.255.0   U     0      0        0 R3-eth1
172.0.6.0       *               255.255.255.0   U     0      0        0 R3-eth0
```

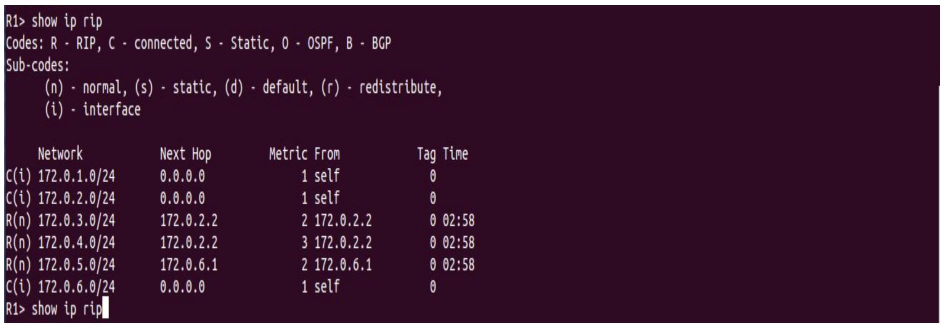


```
mininext> R4 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       172.0.3.1       255.255.255.0   UG    3      0        0 R4-eth1
172.0.2.0       172.0.3.1       255.255.255.0   UG    2      0        0 R4-eth1
172.0.3.0       *               255.255.255.0   U     0      0        0 R4-eth1
172.0.4.0       *               255.255.255.0   U     0      0        0 R4-eth0
172.0.5.0       *               255.255.255.0   U     0      0        0 R4-eth2
172.0.6.0       172.0.5.2       255.255.255.0   UG    2      0        0 R4-eth2
mininext> H2 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
172.0.1.0       172.0.4.2       255.255.255.0   UG    4      0        0 H2-eth0
172.0.2.0       172.0.4.2       255.255.255.0   UG    3      0        0 H2-eth0
172.0.3.0       172.0.4.2       255.255.255.0   UG    2      0        0 H2-eth0
172.0.4.0       *               255.255.255.0   U     0      0        0 H2-eth0
172.0.5.0       172.0.4.2       255.255.255.0   UG    2      0        0 H2-eth0
172.0.6.0       172.0.4.2       255.255.255.0   UG    3      0        0 H2-eth0
mininext>
```

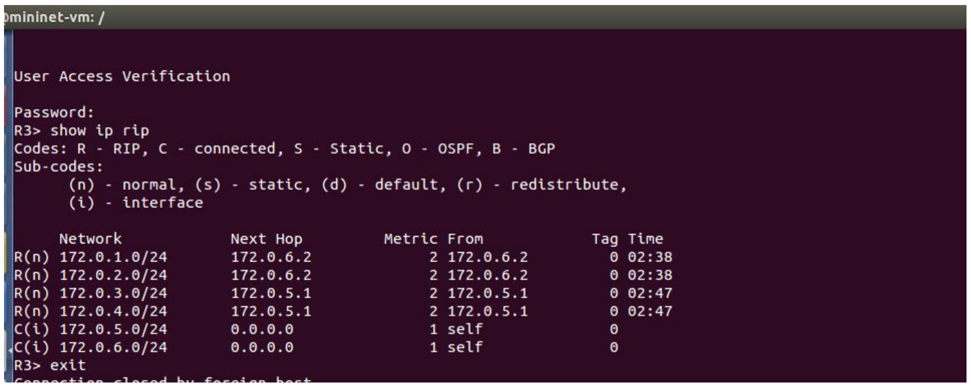
Quagga Output :



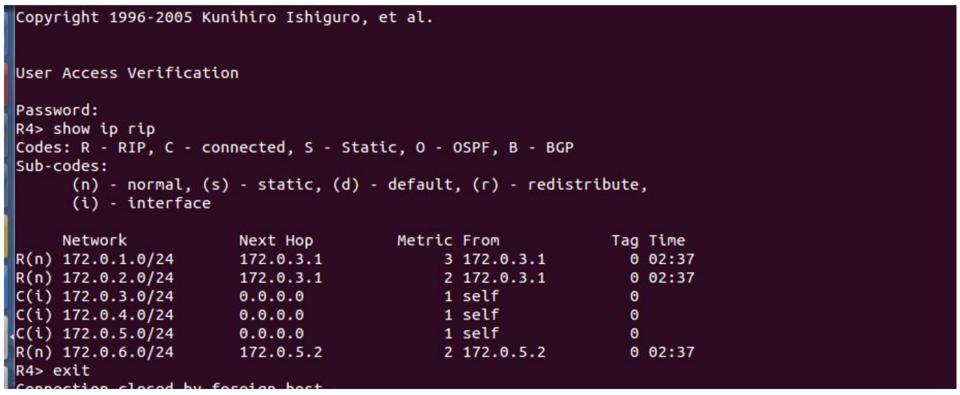
```
ininet-vm: /                                                          3:59 AM
oot@mininet-vm:/home/mininet/miniNExT/util# ./mx H1
oot@mininet-vm:/# telnet localhost 2602
rying ::1...
rying 127.0.0.1...
onnected to localhost.
scape character is '^]'.

ello, this is Quagga (version 0.99.22.4).
opyright 1996-2005 Kunihiro Ishiguro, et al.


ser Access Verification

assword:
1> show ip rip
odes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
ub-codes:
     (n) - normal, (s) - static, (d) - default, (r) - redistribute,
     (i) - interface

    Network          Next Hop        Metric From            Tag Time
(i) 172.0.1.0/24     0.0.0.0              1 self              0
(n) 172.0.2.0/24     172.0.1.2            2 172.0.1.2         0 02:57
(n) 172.0.3.0/24     172.0.1.2            3 172.0.1.2         0 02:57
(n) 172.0.4.0/24     172.0.1.2            4 172.0.1.2         0 02:57
(n) 172.0.5.0/24     172.0.1.2            3 172.0.1.2         0 02:57
(n) 172.0.6.0/24     172.0.1.2            2 172.0.1.2         0 02:57
1> exit
```



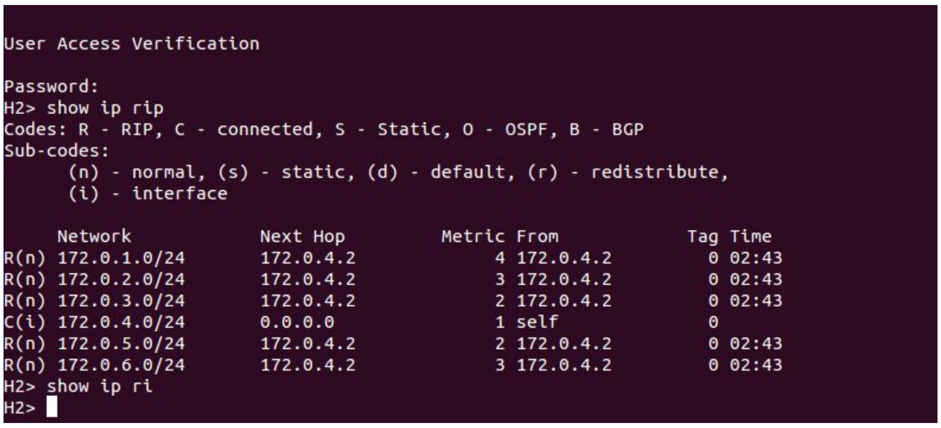
```
R1> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network          Next Hop        Metric From            Tag Time
C(i) 172.0.1.0/24     0.0.0.0              1 self              0
C(i) 172.0.2.0/24     0.0.0.0              1 self              0
R(n) 172.0.3.0/24     172.0.2.2            2 172.0.2.2         0 02:58
R(n) 172.0.4.0/24     172.0.2.2            3 172.0.2.2         0 02:58
R(n) 172.0.5.0/24     172.0.6.1            2 172.0.6.1         0 02:58
C(i) 172.0.6.0/24     0.0.0.0              1 self              0
R1> show ip rip
```



```
mininet-vm: /


User Access Verification

Password:
R3> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network          Next Hop        Metric From            Tag Time
R(n) 172.0.1.0/24     172.0.6.2            2 172.0.6.2         0 02:38
R(n) 172.0.2.0/24     172.0.6.2            2 172.0.6.2         0 02:38
R(n) 172.0.3.0/24     172.0.5.1            2 172.0.5.1         0 02:47
R(n) 172.0.4.0/24     172.0.5.1            2 172.0.5.1         0 02:47
C(i) 172.0.5.0/24     0.0.0.0              1 self              0
C(i) 172.0.6.0/24     0.0.0.0              1 self              0
R3> exit
Connection closed by foreign host
```

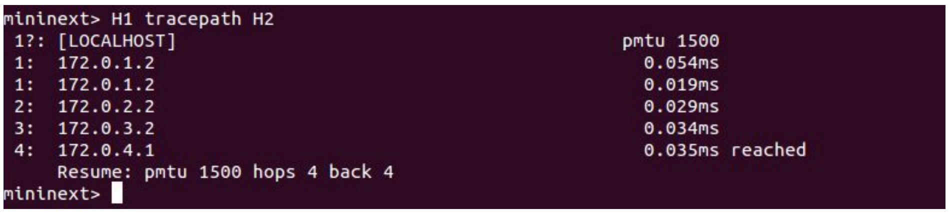


```
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
R4> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network          Next Hop        Metric From            Tag Time
R(n) 172.0.1.0/24     172.0.3.1            3 172.0.3.1         0 02:37
R(n) 172.0.2.0/24     172.0.3.1            2 172.0.3.1         0 02:37
C(i) 172.0.3.0/24     0.0.0.0              1 self              0
C(i) 172.0.4.0/24     0.0.0.0              1 self              0
C(i) 172.0.5.0/24     0.0.0.0              1 self              0
R(n) 172.0.6.0/24     172.0.5.2            2 172.0.5.2         0 02:37
R4> exit
Connection closed by foreign host
```



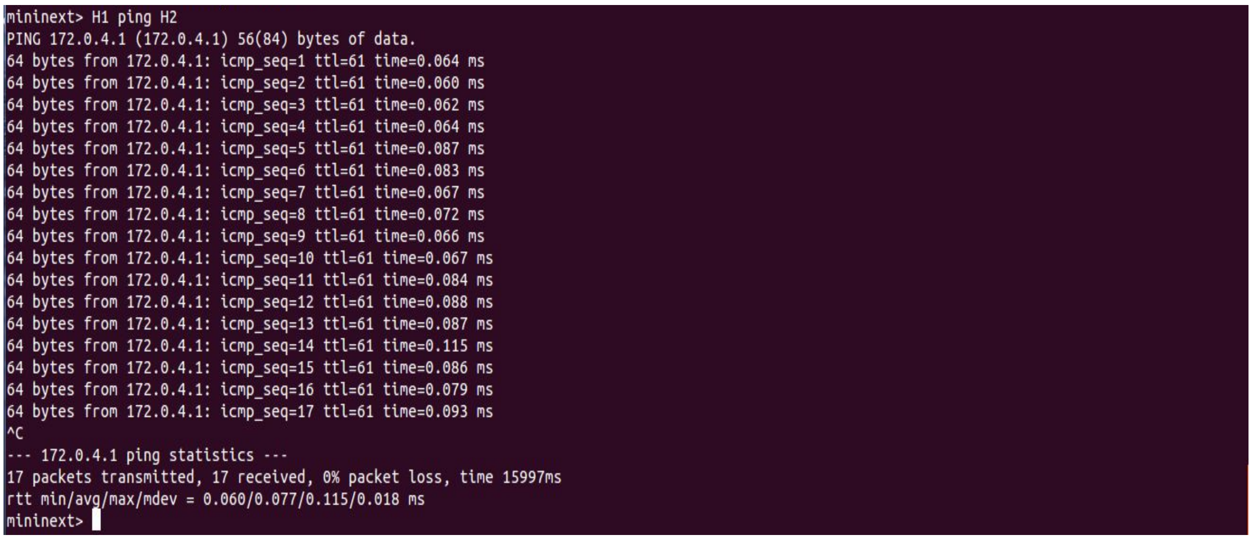
```
User Access Verification

Password:
H2> show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
R(n) 172.0.1.0/24       172.0.4.2             4 172.0.4.2         0 02:43
R(n) 172.0.2.0/24       172.0.4.2             3 172.0.4.2         0 02:43
R(n) 172.0.3.0/24       172.0.4.2             2 172.0.4.2         0 02:43
C(i) 172.0.4.0/24       0.0.0.0               1 self              0
R(n) 172.0.5.0/24       172.0.4.2             2 172.0.4.2         0 02:43
R(n) 172.0.6.0/24       172.0.4.2             3 172.0.4.2         0 02:43
H2> show ip ri
H2>
```

Trace route from H1 to H2



```
mininext> H1 tracepath H2
 1?: [LOCALHOST]                                         pmtu 1500
 1:  172.0.1.2                                             0.054ms
 1:  172.0.1.2                                             0.019ms
 2:  172.0.2.2                                             0.029ms
 3:  172.0.3.2                                             0.034ms
 4:  172.0.4.1                                             0.035ms reached
     Resume: pmtu 1500 hops 4 back 4
mininext>
```

Ping response H1 and H2 : The average ping response time is 67 ms.



```
mininext> H1 ping H2
PING 172.0.4.1 (172.0.4.1) 56(84) bytes of data.
64 bytes from 172.0.4.1: icmp_seq=1 ttl=61 time=0.064 ms
64 bytes from 172.0.4.1: icmp_seq=2 ttl=61 time=0.060 ms
64 bytes from 172.0.4.1: icmp_seq=3 ttl=61 time=0.062 ms
64 bytes from 172.0.4.1: icmp_seq=4 ttl=61 time=0.064 ms
64 bytes from 172.0.4.1: icmp_seq=5 ttl=61 time=0.087 ms
64 bytes from 172.0.4.1: icmp_seq=6 ttl=61 time=0.083 ms
64 bytes from 172.0.4.1: icmp_seq=7 ttl=61 time=0.067 ms
64 bytes from 172.0.4.1: icmp_seq=8 ttl=61 time=0.072 ms
64 bytes from 172.0.4.1: icmp_seq=9 ttl=61 time=0.066 ms
64 bytes from 172.0.4.1: icmp_seq=10 ttl=61 time=0.067 ms
64 bytes from 172.0.4.1: icmp_seq=11 ttl=61 time=0.084 ms
64 bytes from 172.0.4.1: icmp_seq=12 ttl=61 time=0.088 ms
64 bytes from 172.0.4.1: icmp_seq=13 ttl=61 time=0.087 ms
64 bytes from 172.0.4.1: icmp_seq=14 ttl=61 time=0.115 ms
64 bytes from 172.0.4.1: icmp_seq=15 ttl=61 time=0.086 ms
64 bytes from 172.0.4.1: icmp_seq=16 ttl=61 time=0.079 ms
64 bytes from 172.0.4.1: icmp_seq=17 ttl=61 time=0.093 ms
^C
--- 172.0.4.1 ping statistics ---
17 packets transmitted, 17 received, 0% packet loss, time 15997ms
rtt min/avg/max/mdev = 0.060/0.077/0.115/0.018 ms
mininext>
```

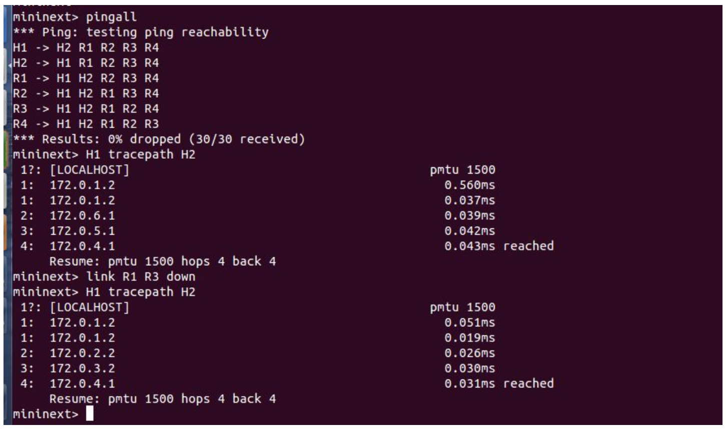
The convergence time of packet from H1 to H2 is 72 msec. (This is calculated by considering the mean response time obtained from the ping response.)

B3:Bring down R1-R2 or R1-R3 (whichever is on your current path from H1 to H2). Estimate the time from when the link went down to when the connectivity was reestablished. Submit: (a) how you got the link to go down, (b) the time it takes for connectivity to be established, (c) provide the traceroute output that gives the new path between nodes H1 & H2.

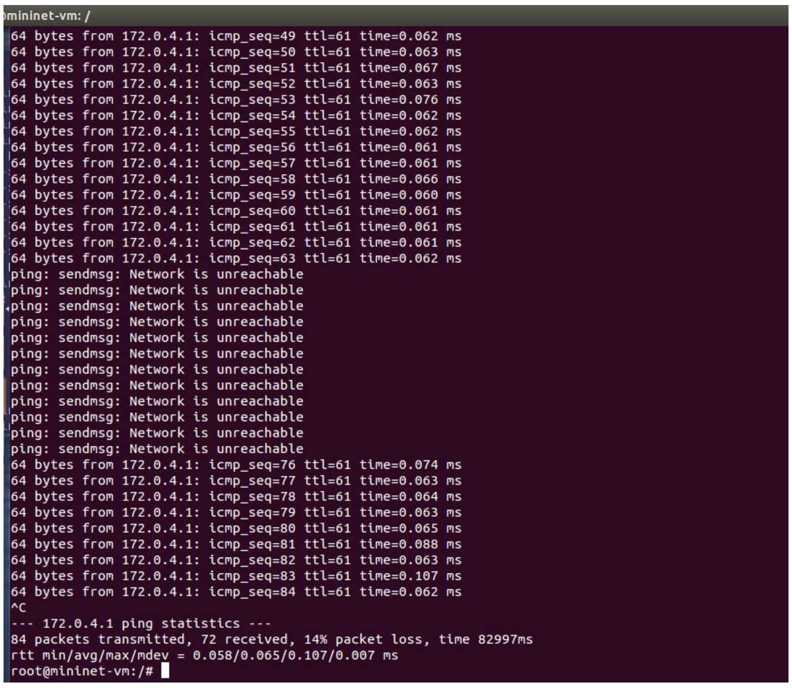
Ans: My initial path between H1 and H2 is via R1-R3 link which i understood by running "H1 tracepath H2" command on one terminal(Terminal 1). In another terminal(Terminal 2) i logged in to H1 terminal and started to ping the H2 ip. Then on Terminal 1 I did the link down using the command link "H1 H3 down".The network was unreachable for some time and the connectivity was restored using another path R1-R2.

I used a stop-watch to calculate the time it required to re-establish the connection which is 12.69 secs

Below is the CLI logs:



```
mininext> pingall
*** Ping: testing ping reachability
H1 -> H2 R1 R2 R3 R4
H2 -> H1 R1 R2 R3 R4
R1 -> H1 H2 R2 R3 R4
R2 -> H1 H2 R1 R3 R4
R3 -> H1 H2 R1 R2 R4
R4 -> H1 H2 R1 R2 R3
*** Results: 0% dropped (30/30 received)
mininext> H1 tracepath H2
 1?: [LOCALHOST]                                         pmtu 1500
 1:  172.0.1.2                                             0.560ms
 1:  172.0.1.2                                             0.037ms
 2:  172.0.6.1                                             0.039ms
 3:  172.0.5.1                                             0.042ms
 4:  172.0.4.1                                             0.043ms reached
     Resume: pmtu 1500 hops 4 back 4
mininext> link R1 R3 down
mininext> H1 tracepath H2
 1?: [LOCALHOST]                                         pmtu 1500
 1:  172.0.1.2                                             0.051ms
 1:  172.0.1.2                                             0.019ms
 2:  172.0.2.2                                             0.026ms
 3:  172.0.3.2                                             0.030ms
 4:  172.0.4.1                                             0.031ms reached
     Resume: pmtu 1500 hops 4 back 4
mininext> 
```



```
mininet-vm: /
64 bytes from 172.0.4.1: icmp_seq=49 ttl=61 time=0.062 ms
64 bytes from 172.0.4.1: icmp_seq=50 ttl=61 time=0.063 ms
64 bytes from 172.0.4.1: icmp_seq=51 ttl=61 time=0.067 ms
64 bytes from 172.0.4.1: icmp_seq=52 ttl=61 time=0.063 ms
64 bytes from 172.0.4.1: icmp_seq=53 ttl=61 time=0.076 ms
64 bytes from 172.0.4.1: icmp_seq=54 ttl=61 time=0.062 ms
64 bytes from 172.0.4.1: icmp_seq=55 ttl=61 time=0.062 ms
64 bytes from 172.0.4.1: icmp_seq=56 ttl=61 time=0.061 ms
64 bytes from 172.0.4.1: icmp_seq=57 ttl=61 time=0.061 ms
64 bytes from 172.0.4.1: icmp_seq=58 ttl=61 time=0.066 ms
64 bytes from 172.0.4.1: icmp_seq=59 ttl=61 time=0.060 ms
64 bytes from 172.0.4.1: icmp_seq=60 ttl=61 time=0.061 ms
64 bytes from 172.0.4.1: icmp_seq=61 ttl=61 time=0.061 ms
64 bytes from 172.0.4.1: icmp_seq=62 ttl=61 time=0.061 ms
64 bytes from 172.0.4.1: icmp_seq=63 ttl=61 time=0.062 ms
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
ping: sendmsg: Network is unreachable
64 bytes from 172.0.4.1: icmp_seq=76 ttl=61 time=0.074 ms
64 bytes from 172.0.4.1: icmp_seq=77 ttl=61 time=0.063 ms
64 bytes from 172.0.4.1: icmp_seq=78 ttl=61 time=0.064 ms
64 bytes from 172.0.4.1: icmp_seq=79 ttl=61 time=0.063 ms
64 bytes from 172.0.4.1: icmp_seq=80 ttl=61 time=0.065 ms
64 bytes from 172.0.4.1: icmp_seq=81 ttl=61 time=0.088 ms
64 bytes from 172.0.4.1: icmp_seq=82 ttl=61 time=0.063 ms
64 bytes from 172.0.4.1: icmp_seq=83 ttl=61 time=0.107 ms
64 bytes from 172.0.4.1: icmp_seq=84 ttl=61 time=0.062 ms
^C
--- 172.0.4.1 ping statistics ---
84 packets transmitted, 72 received, 14% packet loss, time 82997ms
rtt min/avg/max/mdev = 0.058/0.065/0.107/0.007 ms
root@mininet-vm:/# 
```

Part C:

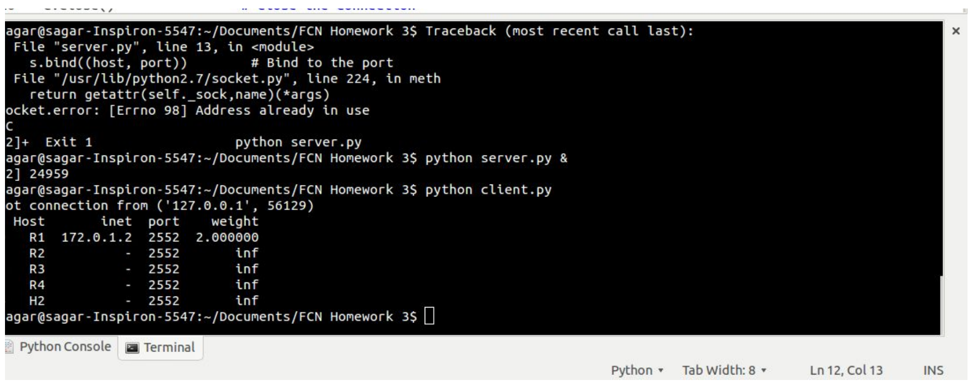
Initially each node will have a dictionary containing information about all other nodes. Initially the cost to each of the nodes other than the neighbors will be marked as infinity.Then i create two threads running between the node and its neighbors which will periodically send distance information to its neighbors in a broadcast manner and every time it evaluates its own dictionary information and checks if it could update any other information onto it.

It uses the following logic
If the distance information of any node is less than the existing value it will update to its new distance information.Otherwise no changes will be done.

As the bellman - ford algorithm implements the information at each node will be updated in a timely fashion and accordingly other information to all other nodes will be updated to get the information.

Implementation:

Each node has the information about its neighbors in a CSV file with information of other nodes , IP address, portnumber, weights to the neighbors. The image shows the information shared over socket between H1 and R1. In similar way each node will broadcast the information to its neighbors and routing table using distance vector will get updated on each node.This is updated using the conditions discussed above.



```
agar@sagar-Inspiron-5547:~/Documents/FCN Homework 3$ Traceback (most recent call last):
 File "server.py", line 13, in <module>
   s.bind((host, port))        # Bind to the port
 File "/usr/lib/python2.7/socket.py", line 224, in meth
   return getattr(self._sock,name)(*args)
ocket.error: [Errno 98] Address already in use
C
2]+  Exit 1                 python server.py
agar@sagar-Inspiron-5547:~/Documents/FCN Homework 3$ python server.py &
2] 24959
agar@sagar-Inspiron-5547:~/Documents/FCN Homework 3$ python client.py
ot connection from ('127.0.0.1', 56129)
 Host       inet  port    weight
   R1  172.0.1.2  2552  2.000000
   R2          -  2552       inf
   R3          -  2552       inf
   R4          -  2552       inf
   H2          -  2552       inf
agar@sagar-Inspiron-5547:~/Documents/FCN Homework 3$
```