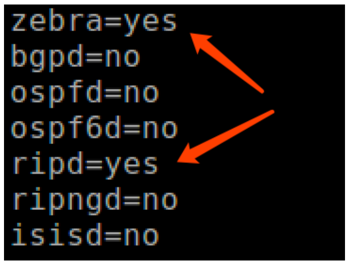
Table of Contents

B1 1
B2 2
B3 8

B1

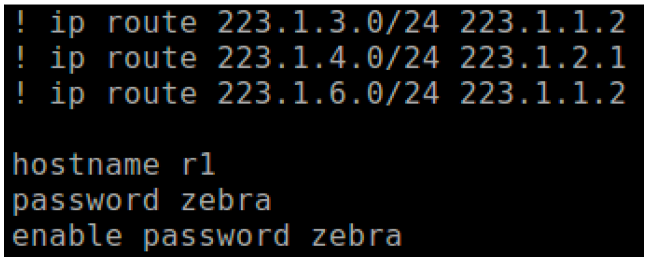
(a) the set of commands you used for the configuration in correct order
I will take router 1 (r1) as an example.

For *r1/daemon*, I turn on *zebra* and *ripd*



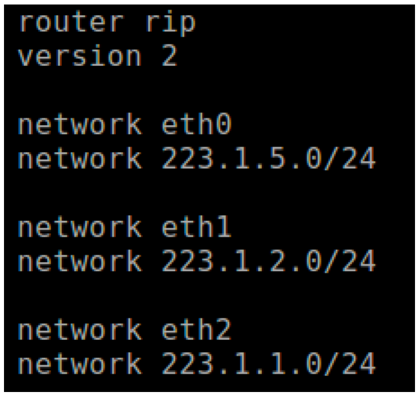
```
zebra=yes
bgpd=no
ospfd=no
ospf6d=no
ripd=yes
ripngd=no
isisd=no
```

For *r1/zebra.conf*, I comment out the static routing code (of course you can delete them).



```
! ip route 223.1.3.0/24 223.1.1.2
! ip route 223.1.4.0/24 223.1.2.1
! ip route 223.1.6.0/24 223.1.1.2

hostname r1
password zebra
enable password zebra
```

For *r1/ripd.conf*, the commands are as follows:



```
router rip
version 2

network eth0
network 223.1.5.0/24

network eth1
network 223.1.2.0/24

network eth2
network 223.1.1.0/24
```

(b) explanation for each command

Rip portal requires interface information maintained by zebra daemon. So, running *zebra* is mandatory to run *ripd*. So, in *r1/daemons*, we need to set *zebra=yes & ripd=yes*.

In *r1/zebra.conf*, we also *set hostname* and *set password*, so that we can telnet quagga.

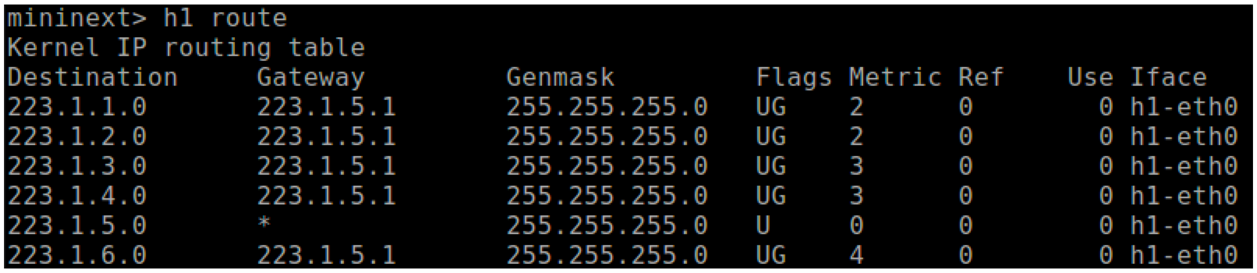
In r1/ripd.conf, I used 4 different commands, explained as follows:

1. **router rip** command is necessary to enable RIP.
2. Disabling RIPv1 by specifying **version 2** is strongly encouraged.
3. **network** *network* command set the RIP enable interface by *network*. The interfaces which have addresses matching with *network* are enabled.
4. **network** *ifname* command set a RIP enabled interface by *ifname*. Both the sending and receiving of RIP packets will be enabled on the port specified in the *network ifname* command.

B2

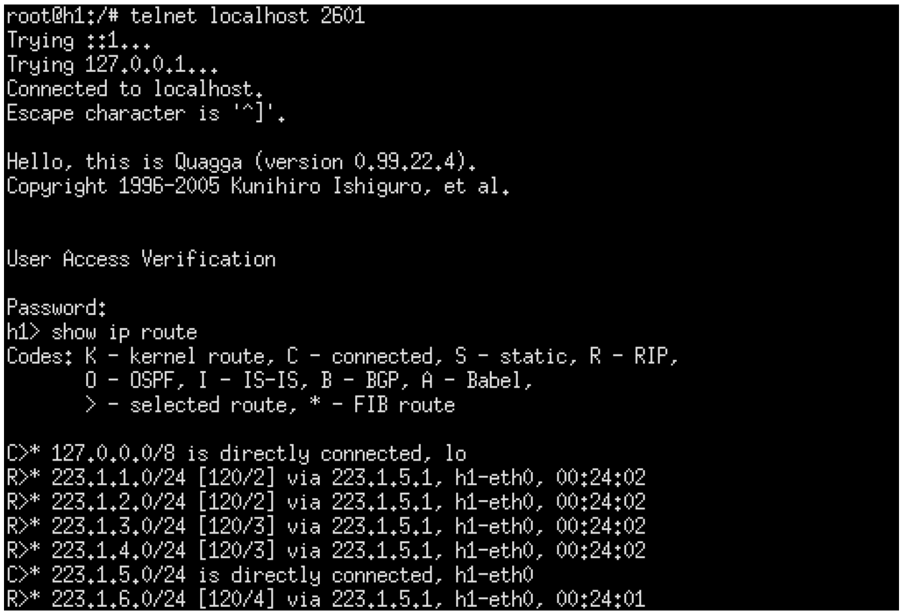
(a) The routing tables at each node (both the kernel and Quagga routing table)

h1-kernal:



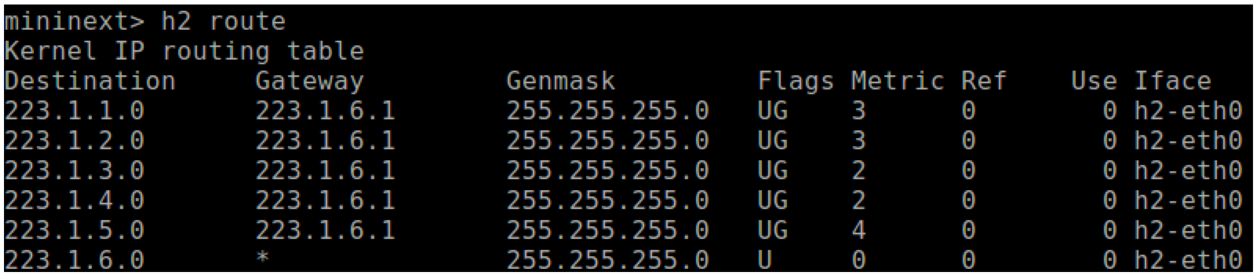
```
mininext> h1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       223.1.5.1       255.255.255.0   UG    2      0        0 h1-eth0
223.1.2.0       223.1.5.1       255.255.255.0   UG    2      0        0 h1-eth0
223.1.3.0       223.1.5.1       255.255.255.0   UG    3      0        0 h1-eth0
223.1.4.0       223.1.5.1       255.255.255.0   UG    3      0        0 h1-eth0
223.1.5.0       *               255.255.255.0   U     0      0        0 h1-eth0
223.1.6.0       223.1.5.1       255.255.255.0   UG    4      0        0 h1-eth0
```

h1-quagga:



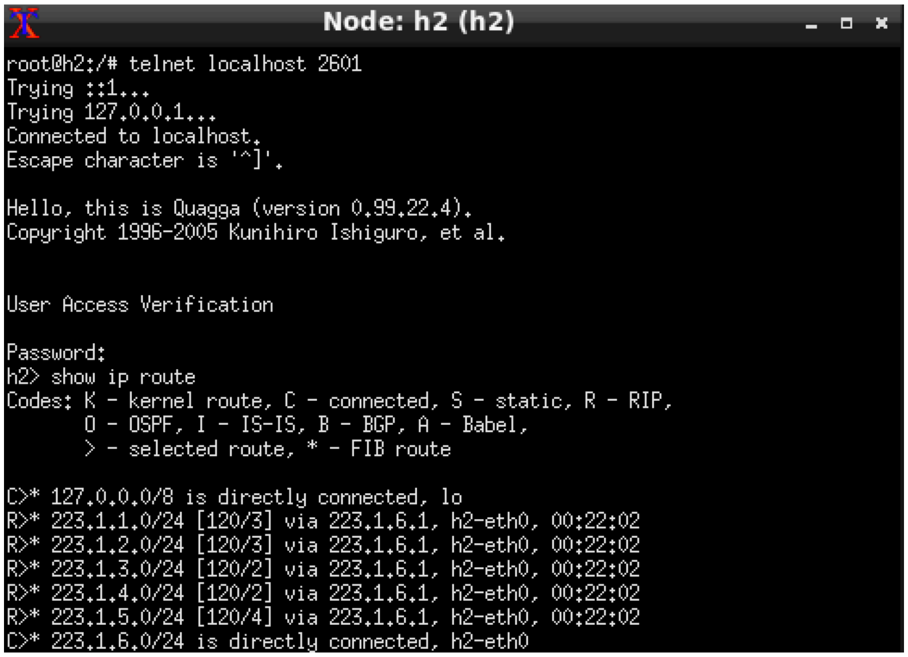
```
root@h1:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
h1> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
R>* 223.1.1.0/24 [120/2] via 223.1.5.1, h1-eth0, 00:24:02
R>* 223.1.2.0/24 [120/2] via 223.1.5.1, h1-eth0, 00:24:02
R>* 223.1.3.0/24 [120/3] via 223.1.5.1, h1-eth0, 00:24:02
R>* 223.1.4.0/24 [120/3] via 223.1.5.1, h1-eth0, 00:24:02
C>* 223.1.5.0/24 is directly connected, h1-eth0
R>* 223.1.6.0/24 [120/4] via 223.1.5.1, h1-eth0, 00:24:01
```

h2-kernal:



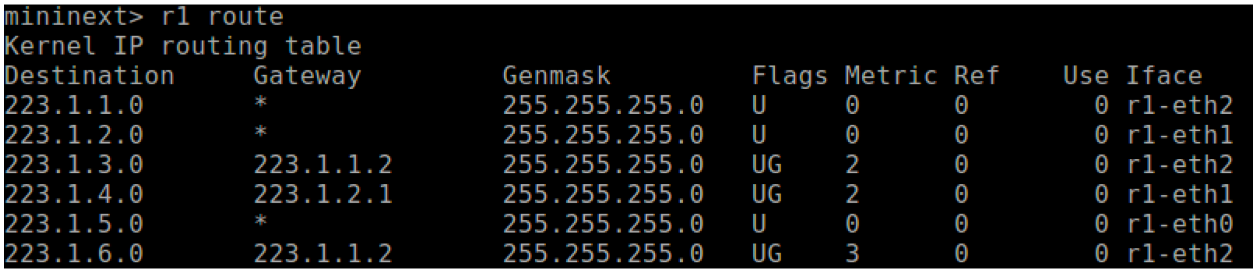
```
mininext> h2 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       223.1.6.1       255.255.255.0   UG    3      0        0 h2-eth0
223.1.2.0       223.1.6.1       255.255.255.0   UG    3      0        0 h2-eth0
223.1.3.0       223.1.6.1       255.255.255.0   UG    2      0        0 h2-eth0
223.1.4.0       223.1.6.1       255.255.255.0   UG    2      0        0 h2-eth0
223.1.5.0       223.1.6.1       255.255.255.0   UG    4      0        0 h2-eth0
223.1.6.0       *               255.255.255.0   U     0      0        0 h2-eth0
```

h2-quagga:



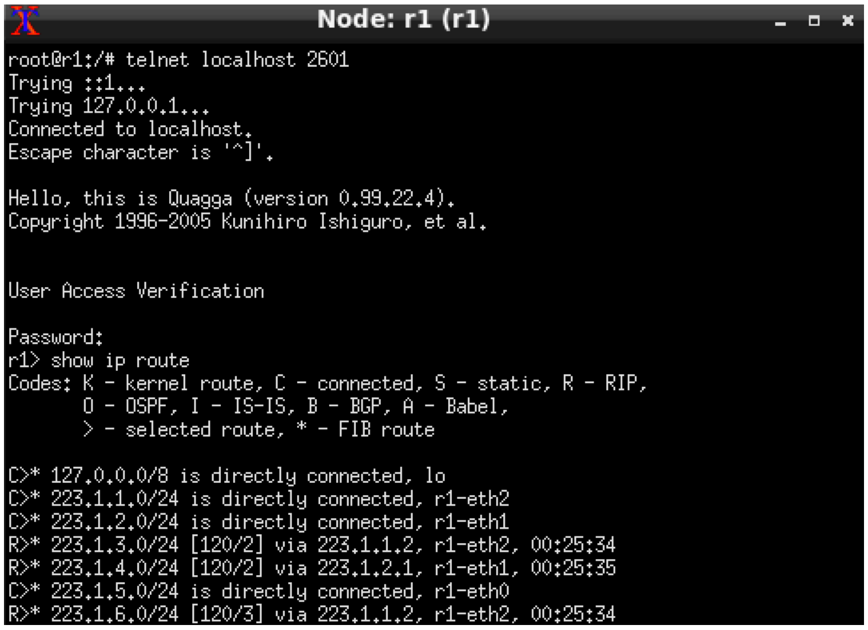
```
root@h2:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
h2> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
R>* 223.1.1.0/24 [120/3] via 223.1.6.1, h2-eth0, 00:22:02
R>* 223.1.2.0/24 [120/3] via 223.1.6.1, h2-eth0, 00:22:02
R>* 223.1.3.0/24 [120/2] via 223.1.6.1, h2-eth0, 00:22:02
R>* 223.1.4.0/24 [120/2] via 223.1.6.1, h2-eth0, 00:22:02
R>* 223.1.5.0/24 [120/4] via 223.1.6.1, h2-eth0, 00:22:02
C>* 223.1.6.0/24 is directly connected, h2-eth0
```

r1-kernal:



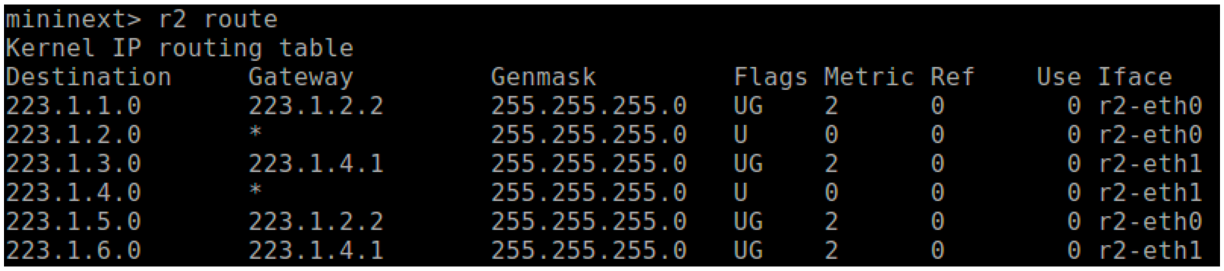
```
mininext> r1 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       *               255.255.255.0   U     0      0        0 r1-eth2
223.1.2.0       *               255.255.255.0   U     0      0        0 r1-eth1
223.1.3.0       223.1.1.2       255.255.255.0   UG    2      0        0 r1-eth2
223.1.4.0       223.1.2.1       255.255.255.0   UG    2      0        0 r1-eth1
223.1.5.0       *               255.255.255.0   U     0      0        0 r1-eth0
223.1.6.0       223.1.1.2       255.255.255.0   UG    3      0        0 r1-eth2
```

r1-quagga:



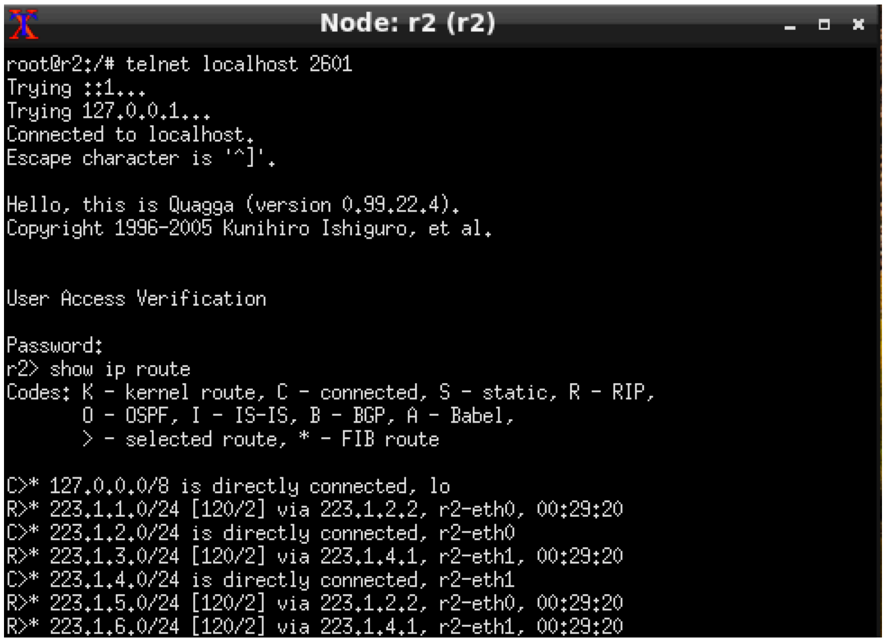
```
Node: r1 (r1)
root@r1:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
r1> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
C>* 223.1.1.0/24 is directly connected, r1-eth2
C>* 223.1.2.0/24 is directly connected, r1-eth1
R>* 223.1.3.0/24 [120/2] via 223.1.1.2, r1-eth2, 00:25:34
R>* 223.1.4.0/24 [120/2] via 223.1.2.1, r1-eth1, 00:25:35
C>* 223.1.5.0/24 is directly connected, r1-eth0
R>* 223.1.6.0/24 [120/3] via 223.1.1.2, r1-eth2, 00:25:34
```

r2-kernal:



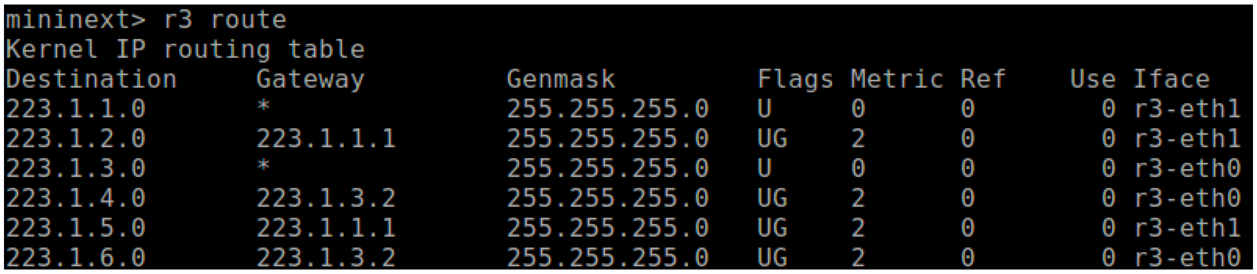
```
mininext> r2 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       223.1.2.2       255.255.255.0   UG    2      0        0 r2-eth0
223.1.2.0       *               255.255.255.0   U     0      0        0 r2-eth0
223.1.3.0       223.1.4.1       255.255.255.0   UG    2      0        0 r2-eth1
223.1.4.0       *               255.255.255.0   U     0      0        0 r2-eth1
223.1.5.0       223.1.2.2       255.255.255.0   UG    2      0        0 r2-eth0
223.1.6.0       223.1.4.1       255.255.255.0   UG    2      0        0 r2-eth1
```

r2-quagga:



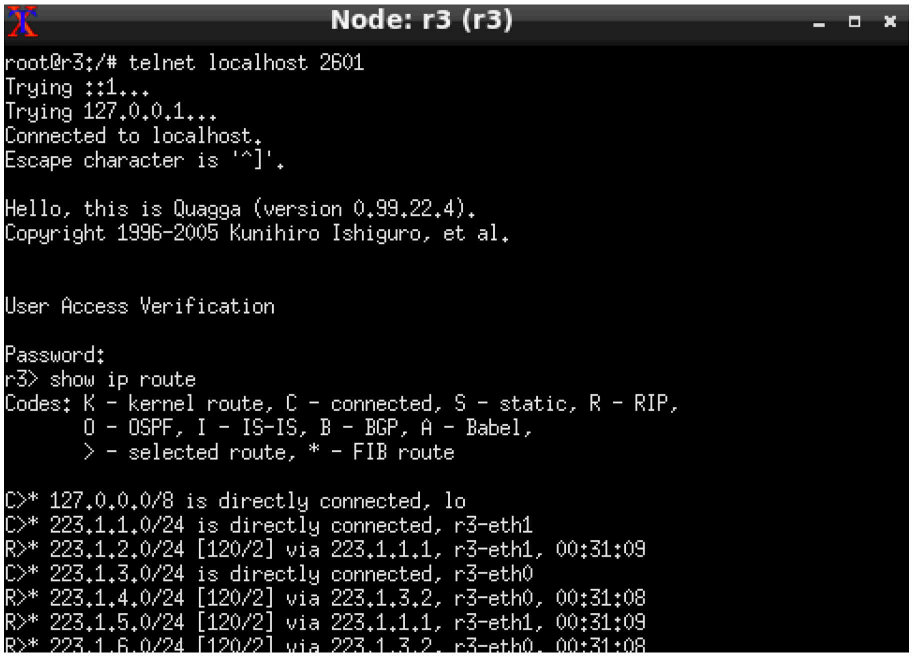
```
Node: r2 (r2)
root@r2:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
r2> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
R>* 223.1.1.0/24 [120/2] via 223.1.2.2, r2-eth0, 00:29:20
C>* 223.1.2.0/24 is directly connected, r2-eth0
R>* 223.1.3.0/24 [120/2] via 223.1.4.1, r2-eth1, 00:29:20
C>* 223.1.4.0/24 is directly connected, r2-eth1
R>* 223.1.5.0/24 [120/2] via 223.1.2.2, r2-eth0, 00:29:20
R>* 223.1.6.0/24 [120/2] via 223.1.4.1, r2-eth1, 00:29:20
```

r3-kernal:



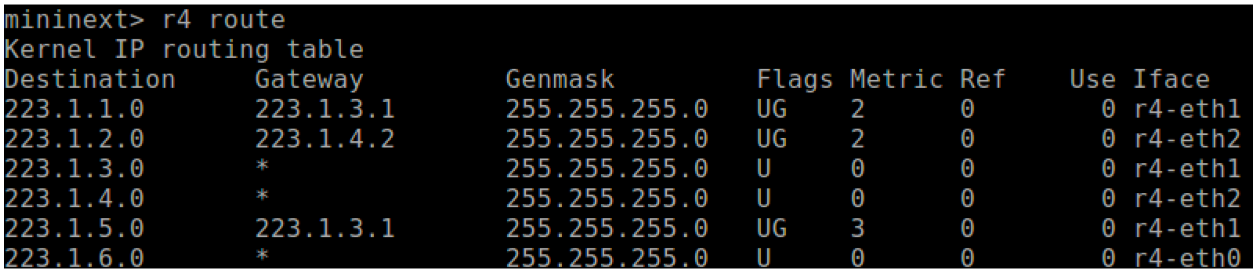
```
mininext> r3 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       *               255.255.255.0   U     0      0        0 r3-eth1
223.1.2.0       223.1.1.1       255.255.255.0   UG    2      0        0 r3-eth1
223.1.3.0       *               255.255.255.0   U     0      0        0 r3-eth0
223.1.4.0       223.1.3.2       255.255.255.0   UG    2      0        0 r3-eth0
223.1.5.0       223.1.1.1       255.255.255.0   UG    2      0        0 r3-eth1
223.1.6.0       223.1.3.2       255.255.255.0   UG    2      0        0 r3-eth0
```

r3-quagga:



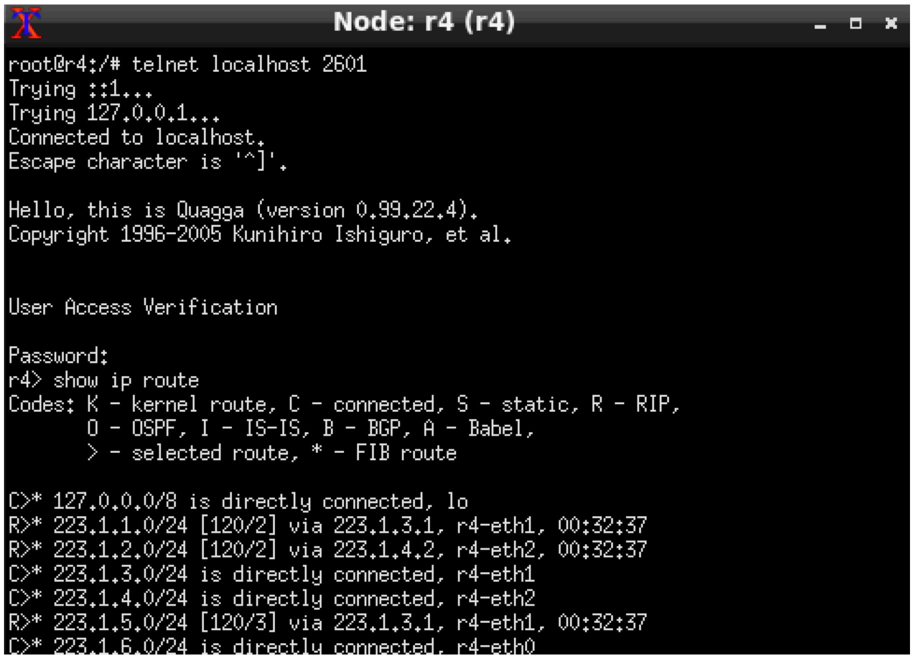
```
Node: r3 (r3)
root@r3:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
r3> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
C>* 223.1.1.0/24 is directly connected, r3-eth1
R>* 223.1.2.0/24 [120/2] via 223.1.1.1, r3-eth1, 00:31:09
C>* 223.1.3.0/24 is directly connected, r3-eth0
R>* 223.1.4.0/24 [120/2] via 223.1.3.2, r3-eth0, 00:31:08
R>* 223.1.5.0/24 [120/2] via 223.1.1.1, r3-eth1, 00:31:09
R>* 223.1.6.0/24 [120/2] via 223.1.3.2, r3-eth0, 00:31:08
```

r4-kernal:



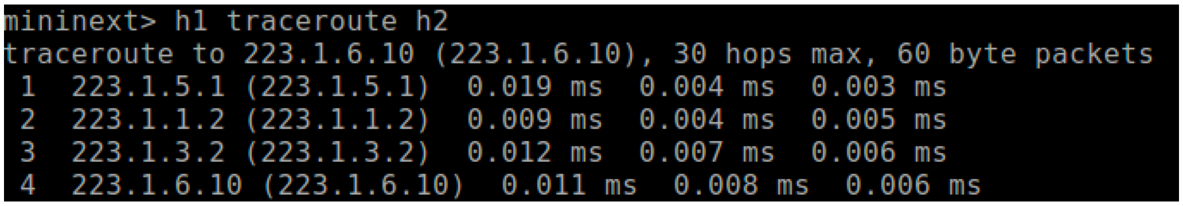
```
mininext> r4 route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
223.1.1.0       223.1.3.1       255.255.255.0   UG    2      0        0 r4-eth1
223.1.2.0       223.1.4.2       255.255.255.0   UG    2      0        0 r4-eth2
223.1.3.0       *               255.255.255.0   U     0      0        0 r4-eth1
223.1.4.0       *               255.255.255.0   U     0      0        0 r4-eth2
223.1.5.0       223.1.3.1       255.255.255.0   UG    3      0        0 r4-eth1
223.1.6.0       *               255.255.255.0   U     0      0        0 r4-eth0
```

r4-quagga:

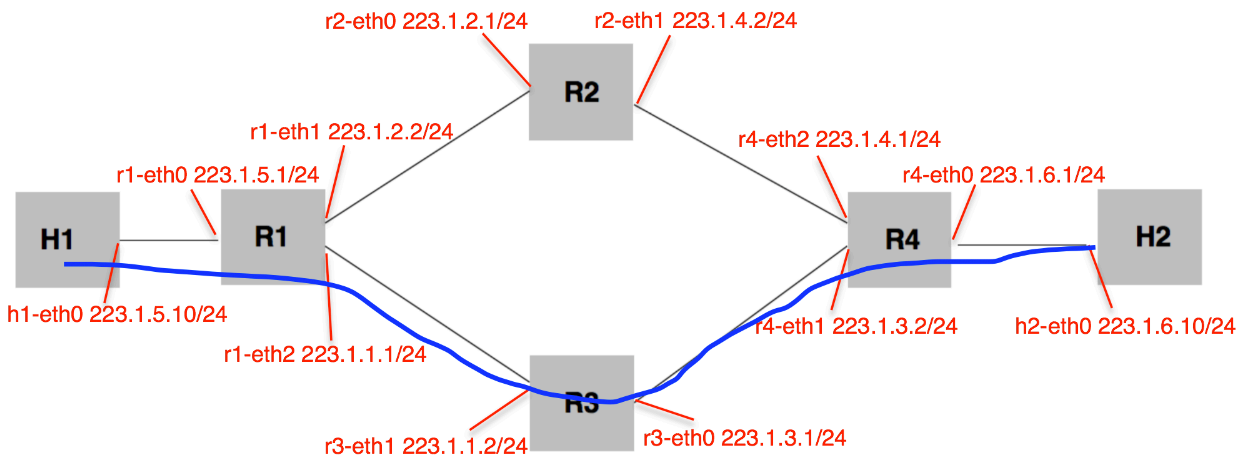


```
root@r4:/# telnet localhost 2601
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.

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


User Access Verification

Password:
r4> show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP,
       O - OSPF, I - IS-IS, B - BGP, A - Babel,
       > - selected route, * - FIB route

C>* 127.0.0.0/8 is directly connected, lo
R>* 223.1.1.0/24 [120/2] via 223.1.3.1, r4-eth1, 00:32:37
R>* 223.1.2.0/24 [120/2] via 223.1.4.2, r4-eth2, 00:32:37
C>* 223.1.3.0/24 is directly connected, r4-eth1
C>* 223.1.4.0/24 is directly connected, r4-eth2
R>* 223.1.5.0/24 [120/3] via 223.1.3.1, r4-eth1, 00:32:37
C>* 223.1.6.0/24 is directly connected, r4-eth0
```

(b) The traceroute output that gives the path between nodes h1 and h2.

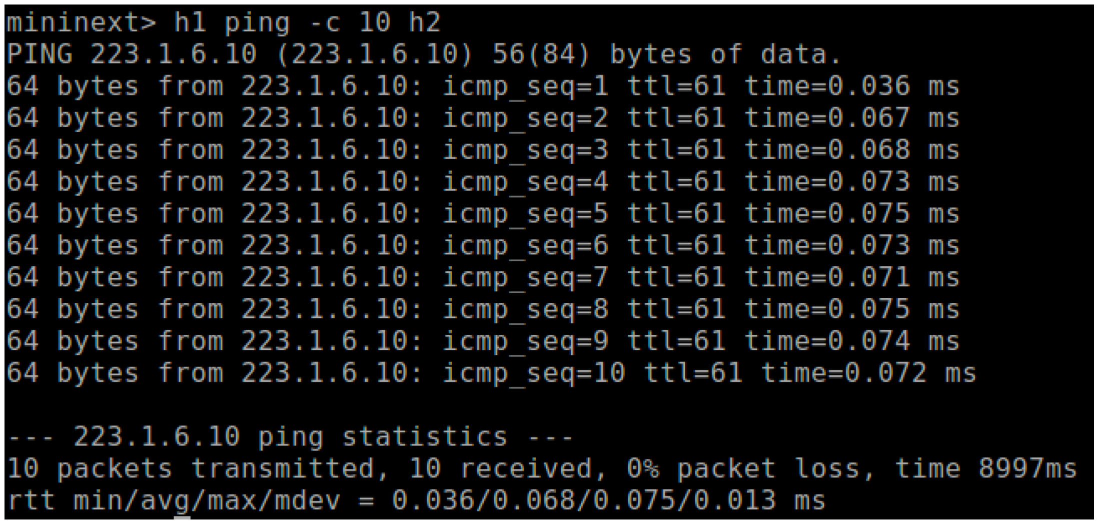


```
mininext> h1 traceroute h2
traceroute to 223.1.6.10 (223.1.6.10), 30 hops max, 60 byte packets
 1  223.1.5.1 (223.1.5.1)  0.019 ms  0.004 ms  0.003 ms
 2  223.1.1.2 (223.1.1.2)  0.009 ms  0.004 ms  0.005 ms
 3  223.1.3.2 (223.1.3.2)  0.012 ms  0.007 ms  0.006 ms
 4  223.1.6.10 (223.1.6.10)  0.011 ms  0.008 ms  0.006 ms
```



(c) The time takes for the ping

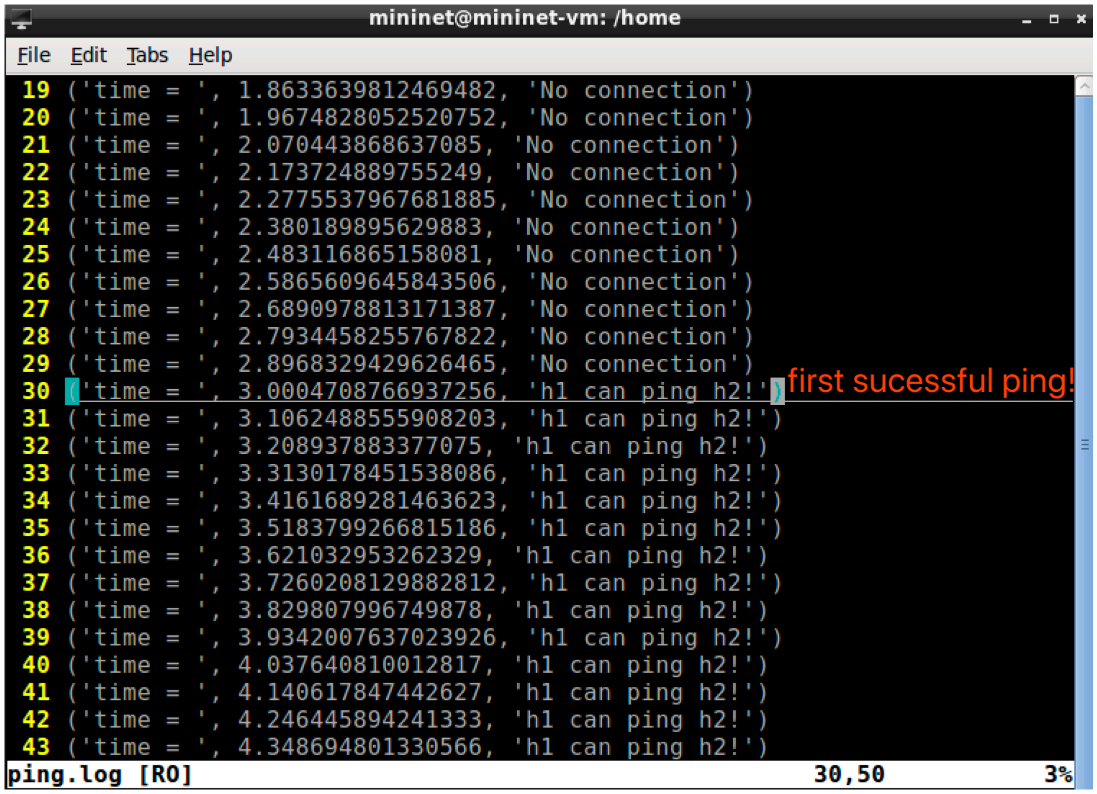
The ping time average is **0.068** seconds. Standard deviation is 0.013.



```
mininext> h1 ping -c 10 h2
PING 223.1.6.10 (223.1.6.10) 56(84) bytes of data.
64 bytes from 223.1.6.10: icmp_seq=1 ttl=61 time=0.036 ms
64 bytes from 223.1.6.10: icmp_seq=2 ttl=61 time=0.067 ms
64 bytes from 223.1.6.10: icmp_seq=3 ttl=61 time=0.068 ms
64 bytes from 223.1.6.10: icmp_seq=4 ttl=61 time=0.073 ms
64 bytes from 223.1.6.10: icmp_seq=5 ttl=61 time=0.075 ms
64 bytes from 223.1.6.10: icmp_seq=6 ttl=61 time=0.073 ms
64 bytes from 223.1.6.10: icmp_seq=7 ttl=61 time=0.071 ms
64 bytes from 223.1.6.10: icmp_seq=8 ttl=61 time=0.075 ms
64 bytes from 223.1.6.10: icmp_seq=9 ttl=61 time=0.074 ms
64 bytes from 223.1.6.10: icmp_seq=10 ttl=61 time=0.072 ms

--- 223.1.6.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 8997ms
rtt min/avg/max/mdev = 0.036/0.068/0.075/0.013 ms
```

(d) The convergence time

According to the following ping.log, the convergence time = **3.0 seconds**. Sometimes it can take only 2 seconds, and sometimes over 4 seconds in different experiments.

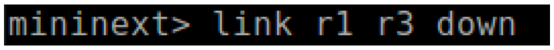


```
('time = ', 1.8633639812469482, 'No connection')
('time = ', 1.9674828052520752, 'No connection')
('time = ', 2.070443868637085, 'No connection')
('time = ', 2.173724889755249, 'No connection')
('time = ', 2.2775537967681885, 'No connection')
('time = ', 2.380189895629883, 'No connection')
('time = ', 2.483116865158081, 'No connection')
('time = ', 2.5865609645843506, 'No connection')
('time = ', 2.6890978813171387, 'No connection')
('time = ', 2.7934458255767822, 'No connection')
('time = ', 2.8968329429626465, 'No connection')
('time = ', 3.0004708766937256, 'h1 can ping h2!')
('time = ', 3.1062488555908203, 'h1 can ping h2!')
('time = ', 3.208937883377075, 'h1 can ping h2!')
('time = ', 3.3130178451538086, 'h1 can ping h2!')
('time = ', 3.4161689281463623, 'h1 can ping h2!')
('time = ', 3.5183799266815186, 'h1 can ping h2!')
('time = ', 3.621032953262329, 'h1 can ping h2!')
('time = ', 3.7260208129882812, 'h1 can ping h2!')
('time = ', 3.829807996749878, 'h1 can ping h2!')
('time = ', 3.9342007637023926, 'h1 can ping h2!')
('time = ', 4.037640810012817, 'h1 can ping h2!')
('time = ', 4.140617847442627, 'h1 can ping h2!')
('time = ', 4.246445894241333, 'h1 can ping h2!')
('time = ', 4.348694801330566, 'h1 can ping h2!')
ping.log [RO]                                         30,50          3%
```

B3

Bring down r1-r3, estimate the time from when the link went down to when the connectivity was reestablished.

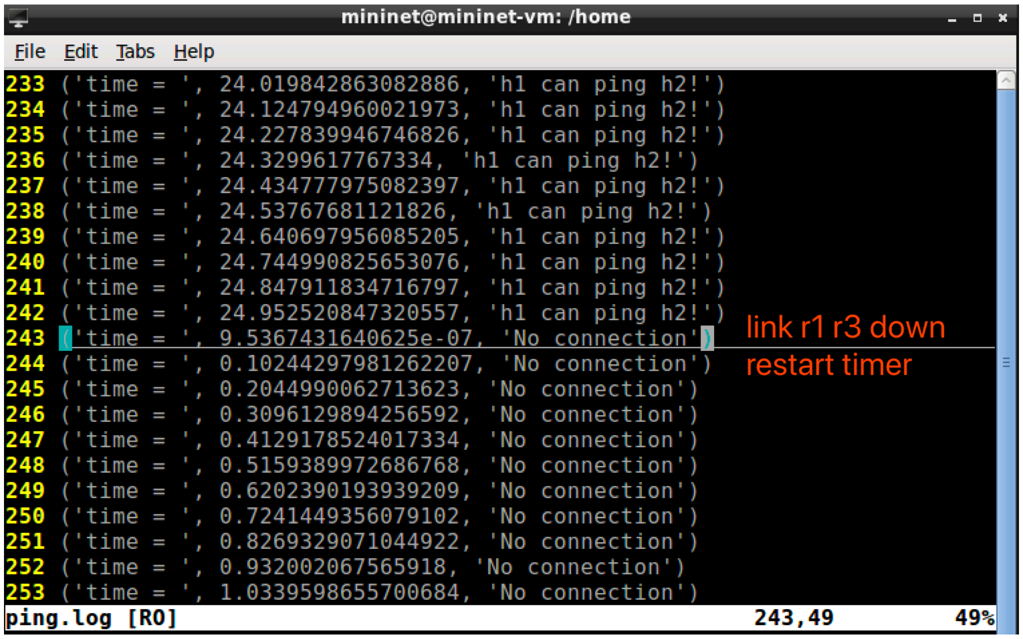
(a) how you got the link to go down.



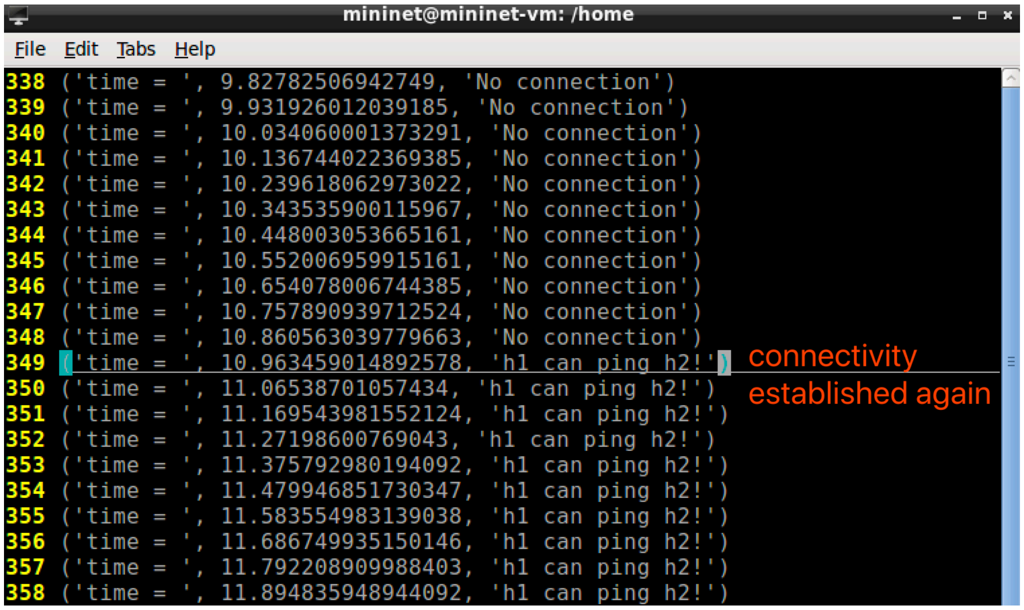
```
mininext> link r1 r3 down
```

(b) the time it takes for connectivity to be established.

Time = **10.9 seconds**

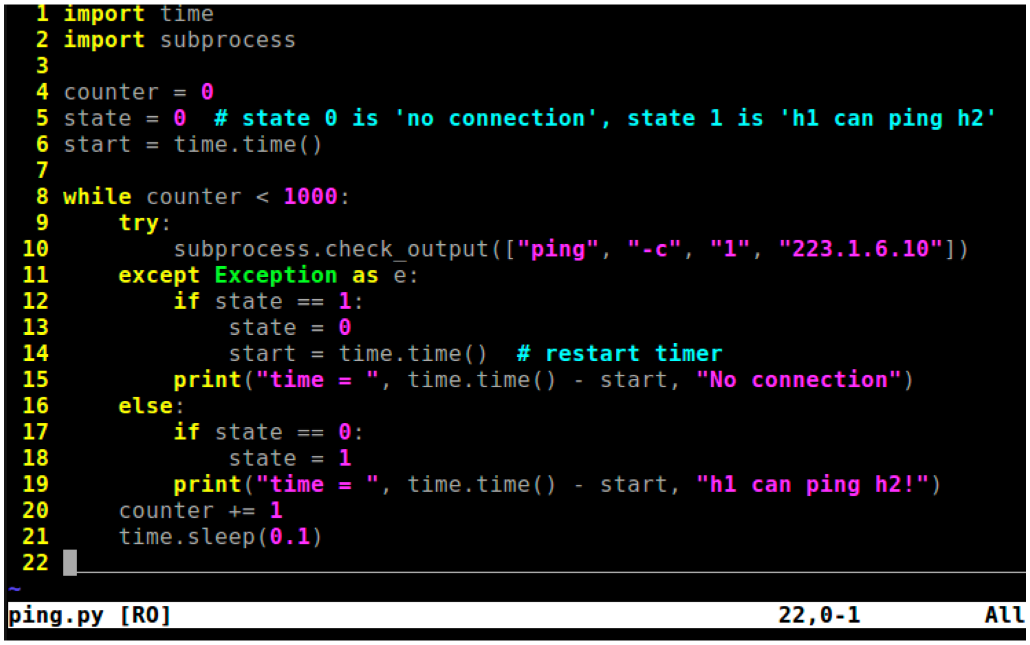


```
('time = ', 24.019842863082886, 'h1 can ping h2!')
('time = ', 24.124794960021973, 'h1 can ping h2!')
('time = ', 24.227839946746826, 'h1 can ping h2!')
('time = ', 24.3299617767334, 'h1 can ping h2!')
('time = ', 24.434777975082397, 'h1 can ping h2!')
('time = ', 24.53767681121826, 'h1 can ping h2!')
('time = ', 24.640697956085205, 'h1 can ping h2!')
('time = ', 24.744990825653076, 'h1 can ping h2!')
('time = ', 24.847911834716797, 'h1 can ping h2!')
('time = ', 24.952520847320557, 'h1 can ping h2!')
('time = ', 9.5367431640625e-07, 'No connection')
('time = ', 0.10244297981262207, 'No connection')
('time = ', 0.2044990062713623, 'No connection')
('time = ', 0.3096129894256592, 'No connection')
('time = ', 0.4129178524017334, 'No connection')
('time = ', 0.5159389972686768, 'No connection')
('time = ', 0.6202390193939209, 'No connection')
('time = ', 0.7241449356079102, 'No connection')
('time = ', 0.8269329071044922, 'No connection')
('time = ', 0.932002067565918, 'No connection')
('time = ', 1.0339598655700684, 'No connection')
ping.log [RO]                                          243,49         49%
```



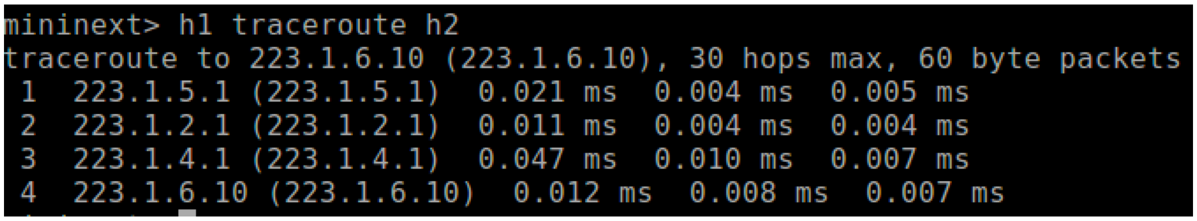
```
('time = ', 9.82782506942749, 'No connection')
('time = ', 9.931926012039185, 'No connection')
('time = ', 10.034060001373291, 'No connection')
('time = ', 10.136744022369385, 'No connection')
('time = ', 10.239618062973022, 'No connection')
('time = ', 10.343535900115967, 'No connection')
('time = ', 10.448003053665161, 'No connection')
('time = ', 10.552006959915161, 'No connection')
('time = ', 10.654078006744385, 'No connection')
('time = ', 10.757890939712524, 'No connection')
('time = ', 10.860563039779663, 'No connection')
('time = ', 10.963459014892578, 'h1 can ping h2!')
('time = ', 11.06538701057434, 'h1 can ping h2!')
('time = ', 11.169543981552124, 'h1 can ping h2!')
('time = ', 11.27198600769043, 'h1 can ping h2!')
('time = ', 11.375792980194092, 'h1 can ping h2!')
('time = ', 11.479946851730347, 'h1 can ping h2!')
('time = ', 11.583554983139038, 'h1 can ping h2!')
('time = ', 11.686749935150146, 'h1 can ping h2!')
('time = ', 11.792208909988403, 'h1 can ping h2!')
('time = ', 11.894835948944092, 'h1 can ping h2!')
```

The following is the ping.py python program that runs at host h1. Host h1 will ping h2 every *0.1 seconds*, and the output is redirected to file *ping.log*

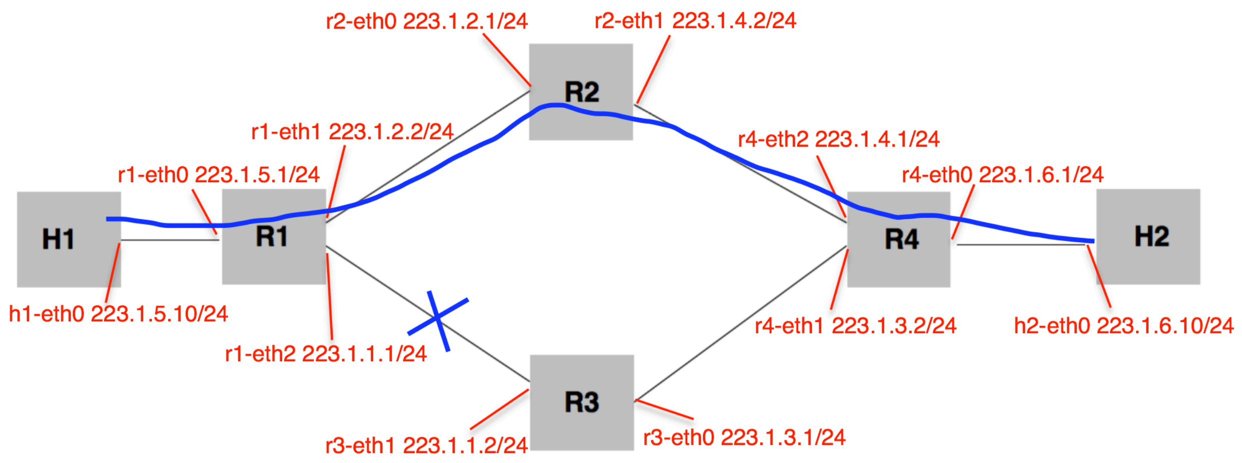


```
import time
import subprocess

counter = 0
state = 0  # state 0 is 'no connection', state 1 is 'h1 can ping h2'
start = time.time()

while counter < 1000:
    try:
        subprocess.check_output(["ping", "-c", "1", "223.1.6.10"])
    except Exception as e:
        if state == 1:
            state = 0
            start = time.time()  # restart timer
        print("time = ", time.time() - start, "No connection")
    else:
        if state == 0:
            state = 1
        print("time = ", time.time() - start, "h1 can ping h2!")
    counter += 1
    time.sleep(0.1)

~
ping.py [RO]                                              22,0-1          All
```

(c) provide the traceroute output that gives the new path between nodes h1 and h2.



```
mininext> h1 traceroute h2
traceroute to 223.1.6.10 (223.1.6.10), 30 hops max, 60 byte packets
 1  223.1.5.1 (223.1.5.1)  0.021 ms  0.004 ms  0.005 ms
 2  223.1.2.1 (223.1.2.1)  0.011 ms  0.004 ms  0.004 ms
 3  223.1.4.1 (223.1.4.1)  0.047 ms  0.010 ms  0.007 ms
 4  223.1.6.10 (223.1.6.10)  0.012 ms  0.008 ms  0.007 ms
```



Note: link **r1-r3** is down, so the new route goes through **r1-r2**.