Kathará

kathara lab

rip with FRRouting

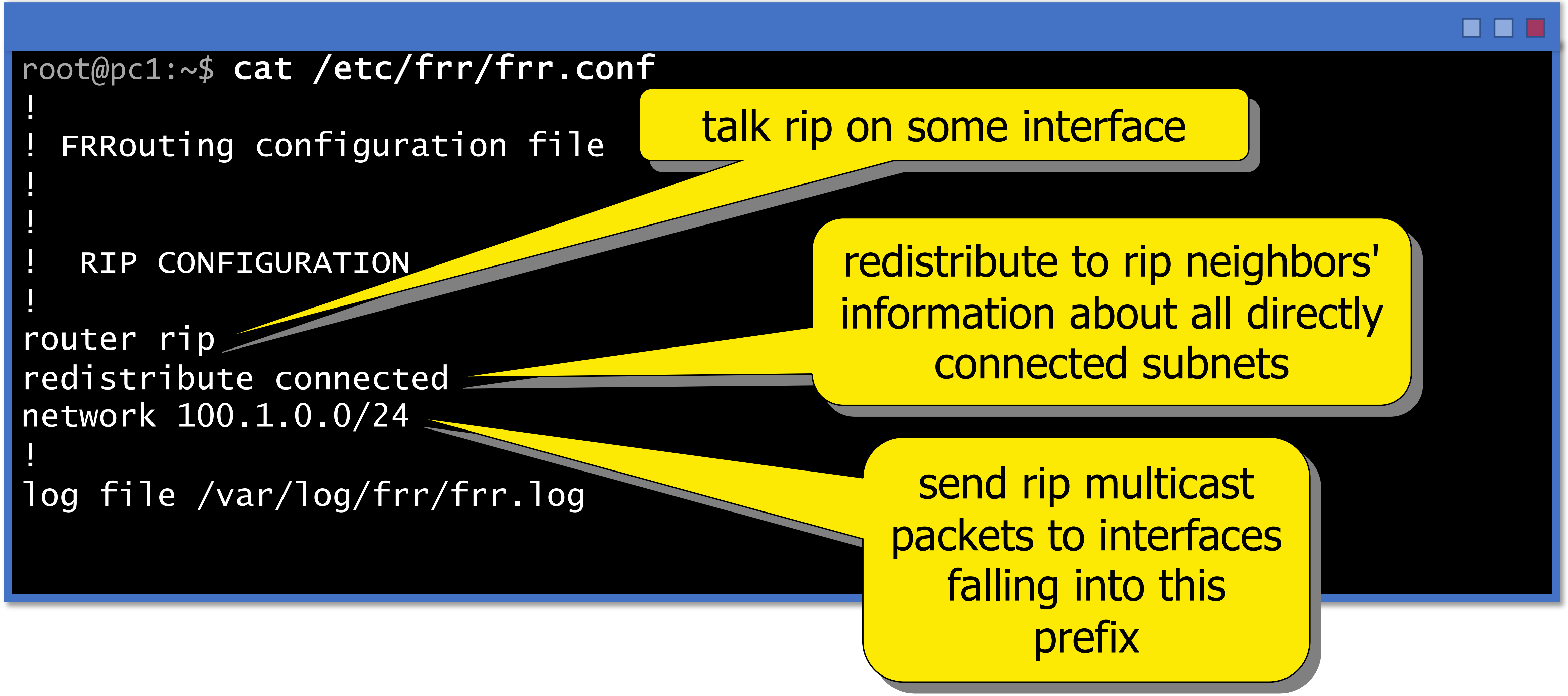
Version	1.1
Author(s)	G. Di Battista, M. Patrignani, M. Pizzonia, L. Ariemma, M. Scazzariello, T. Caiazzi
E-mail	contact@kathara.org
Web	http://www.kathara.org/
Description	experiences with the ripv2 distance vector routing protocol – derives from kathara rip lab ver. 1.2 which, in turns, derives from netkit rip lab ver. 2.4

copyright notice

the RIP protocol

- RIP is a routing protocol
- RIPS is a distance-vector routing protocol
  - approach: send all your information to a few
  - update your routing information based on what you hear
- in this lab we will see an example of RIPv2 protocol on the FRRouting Suite

sample `frr.conf` configuration file



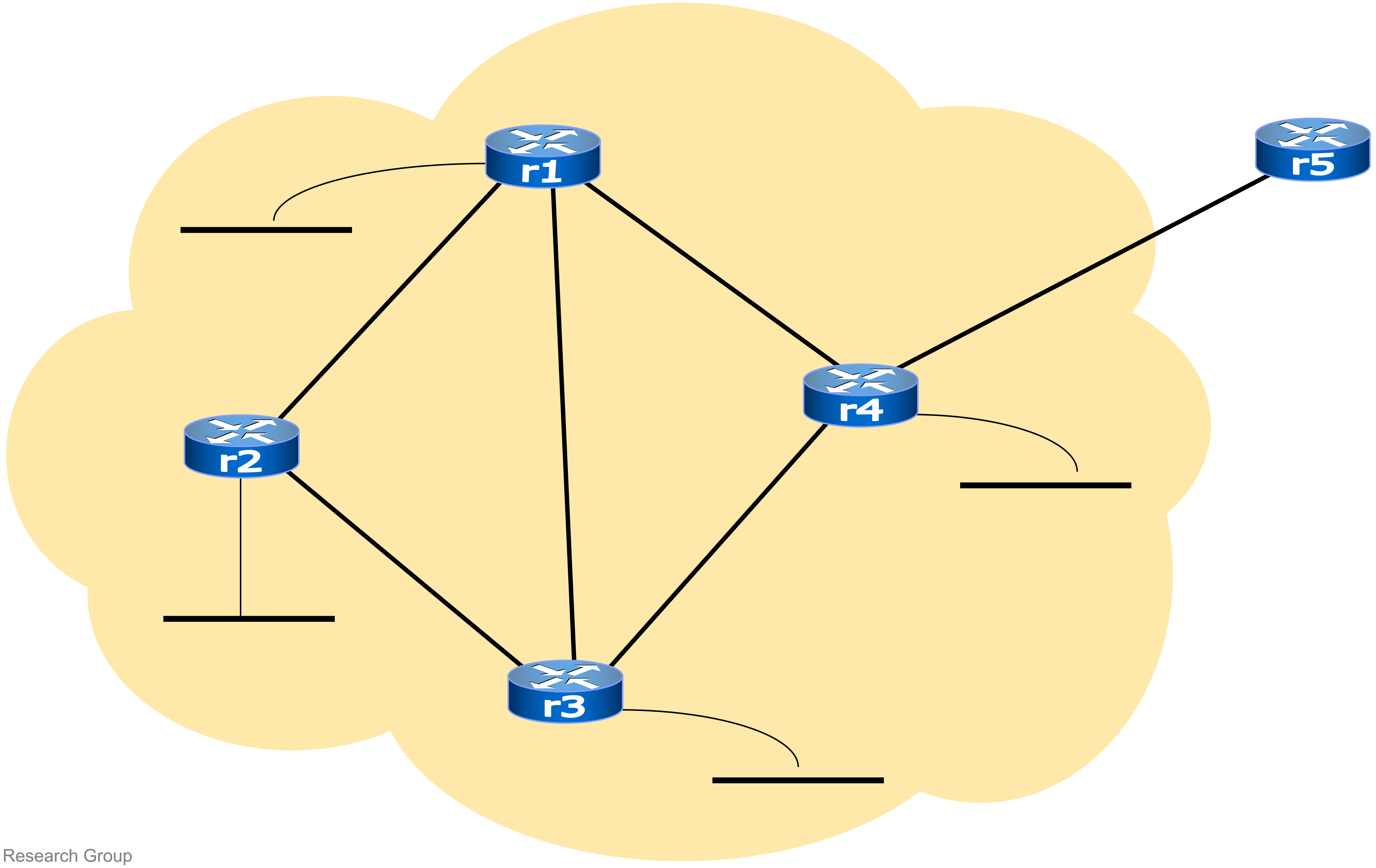
```
root@pc1:~$ cat /etc/frr/frr.conf
!
! FRRouting configuration file
!
!
!  RIP CONFIGURATION
!
router rip
redistribute connected
network 100.1.0.0/24
!
log file /var/log/frr/frr.log
```

talk rip on some interface
redistribute to rip neighbors' information about all directly connected subnets
send rip multicast packets to interfaces falling into this prefix

about `redistribute connected`

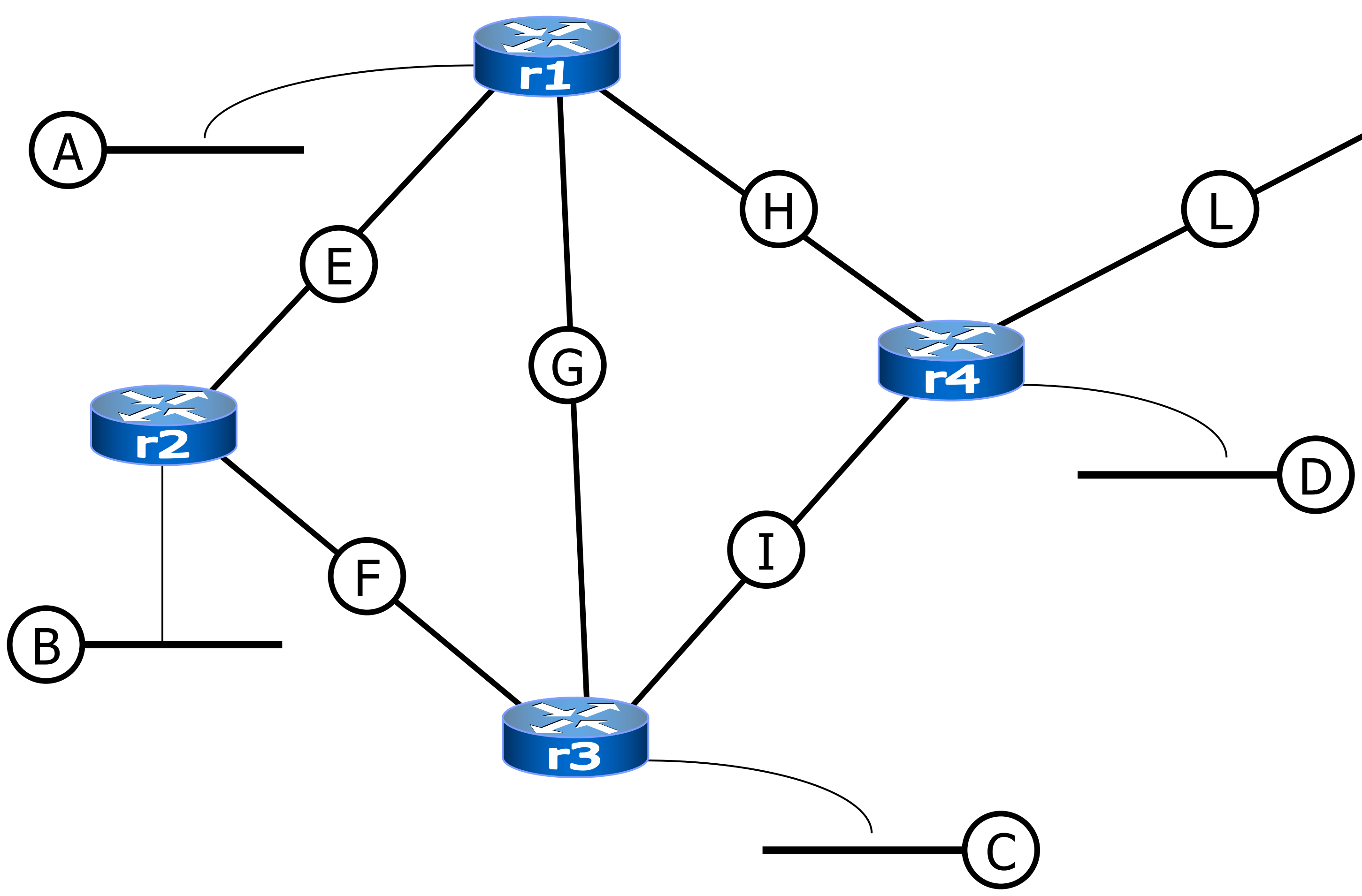
- by default (i.e., without further configuration) RIP already propagates information about all directly connected subnets attached to RIP-speaking interfaces
- **`redistribute connected`** forces RIP to propagate information about all connected subnets
- the semantic of **`redistribute connected`** applies to all routing protocols
- the default behavior does not
  - some protocols (e.g., bgp) are lazier, and do not propagate anything unless explicitly told to do so

small network connected to Internet



last update: Oct 2023

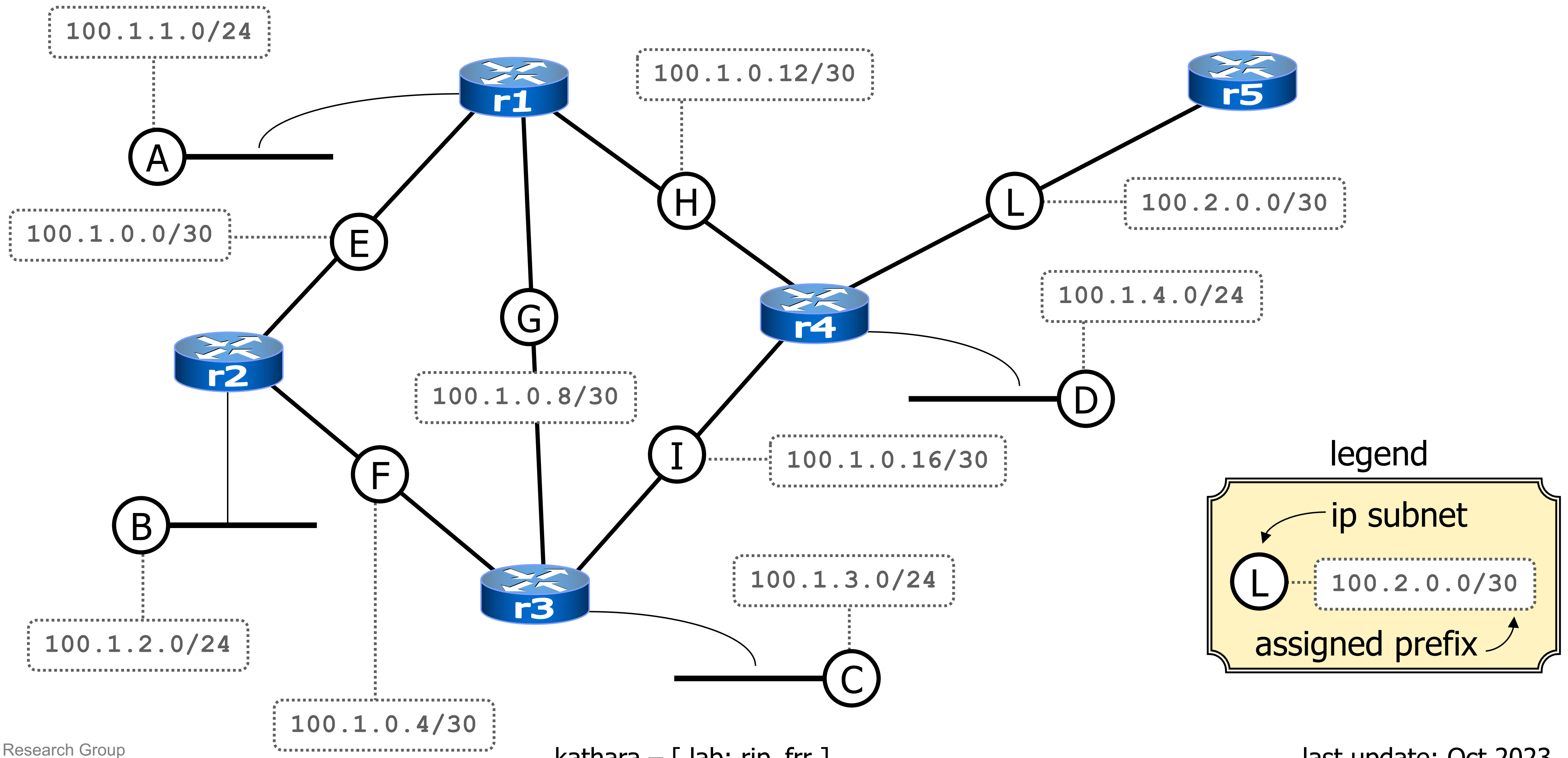
the involved ip subnets



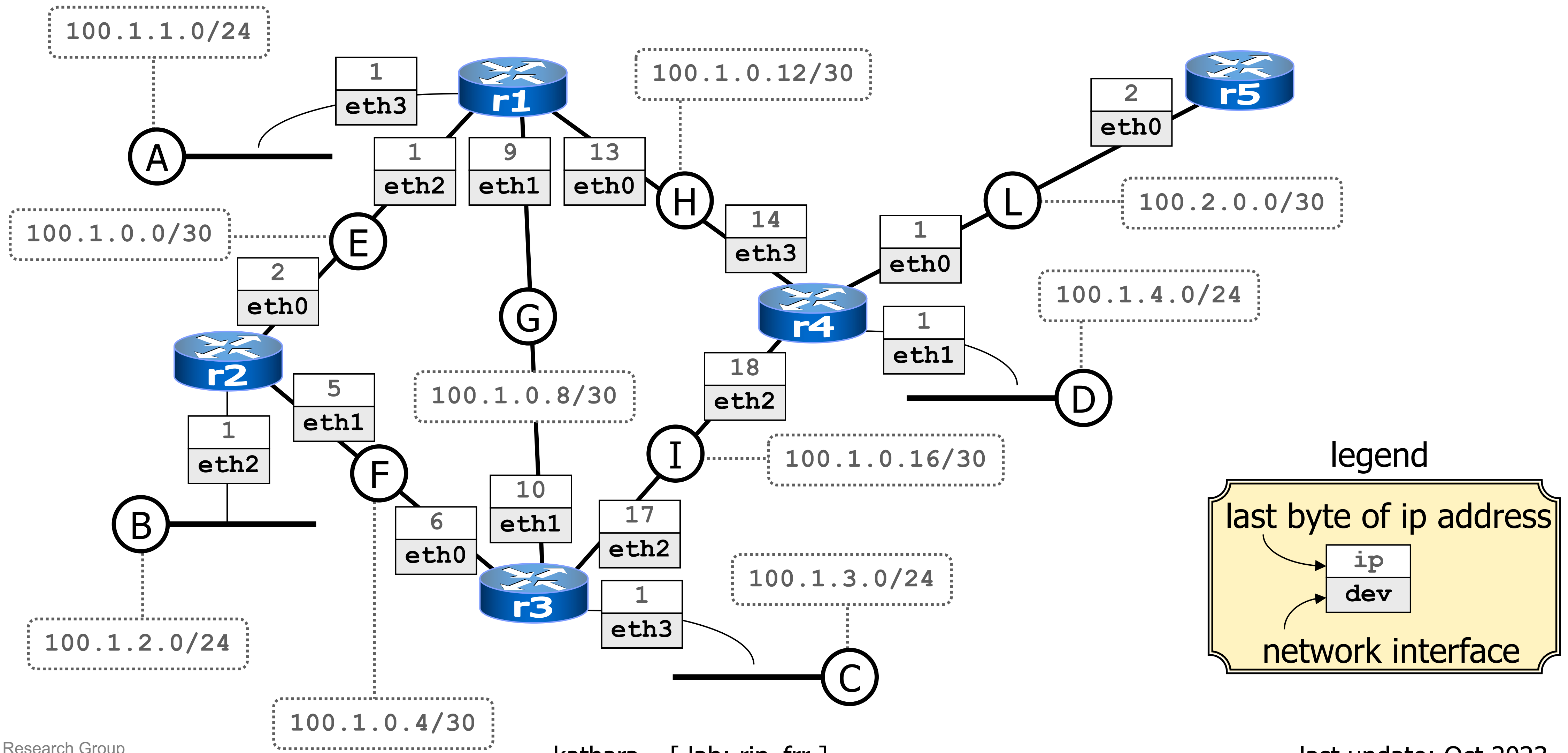
legend

each circle ○ represents a subnet

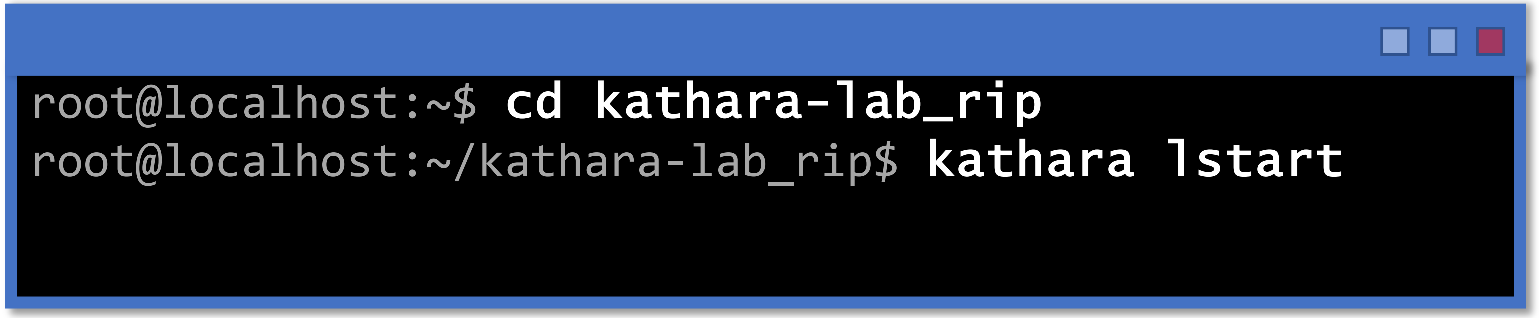
assigning ip numbers to subnets



assigning ip numbers to interfaces



launching the lab

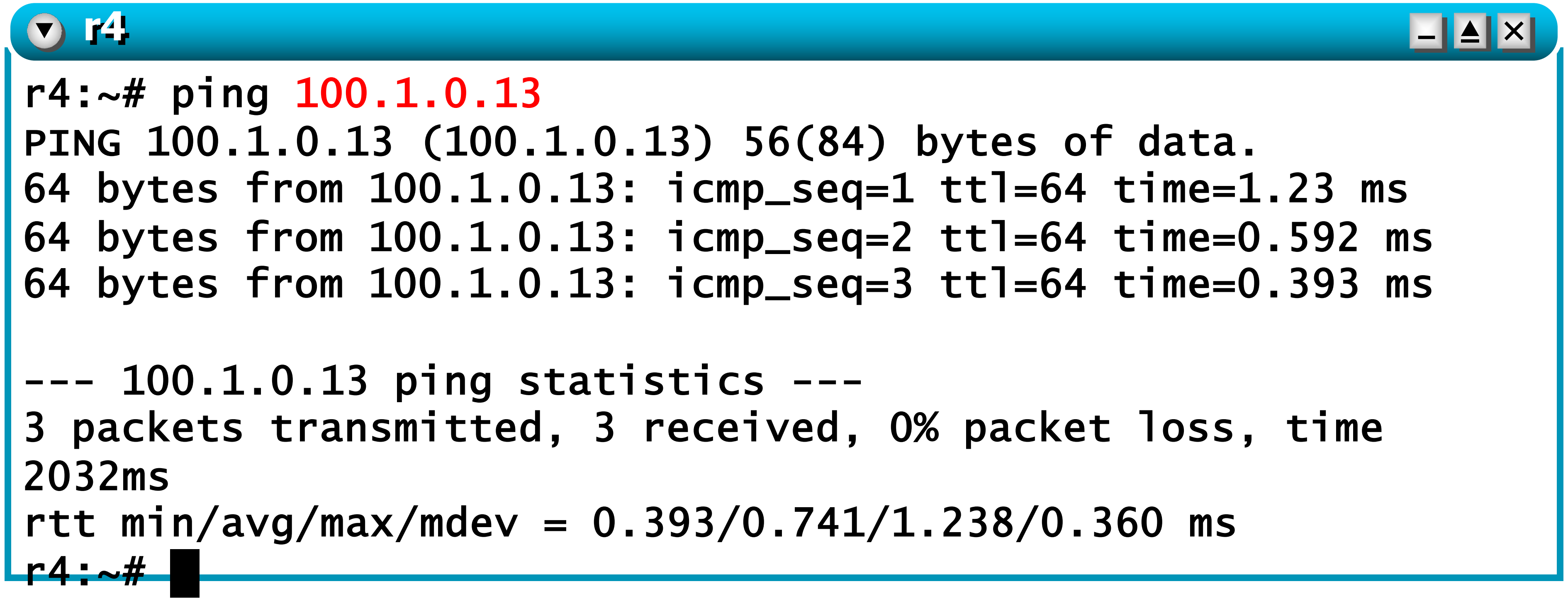


```
root@localhost:~$ cd kathara-lab_rip
root@localhost:~/kathara-lab_rip$ kathara lstart
```

- the lab configuration is such that the frr routing daemon is not automatically started

checking connectivity

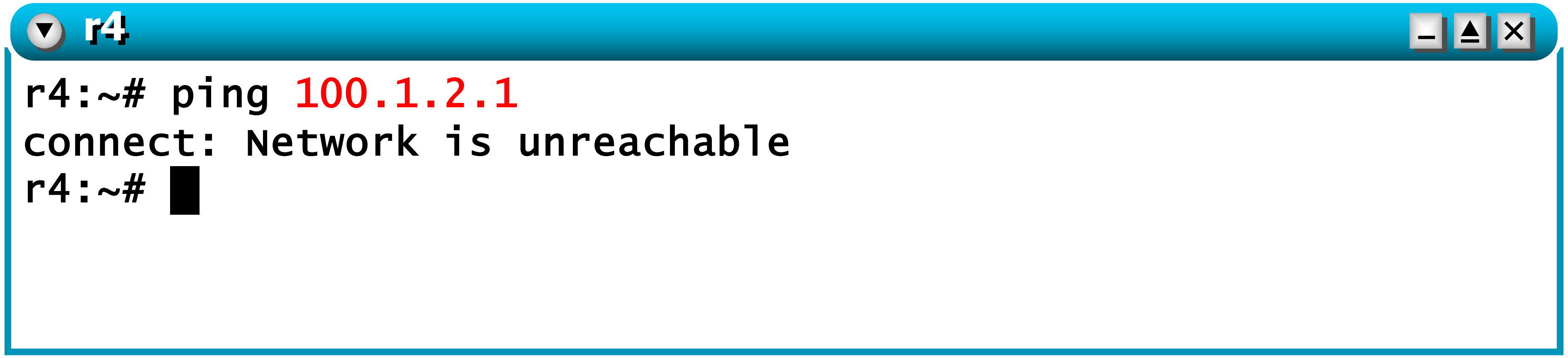
- towards a directly connected destination



```
r4:~# ping 100.1.0.13
PING 100.1.0.13 (100.1.0.13) 56(84) bytes of data.
64 bytes from 100.1.0.13: icmp_seq=1 ttl=64 time=1.23 ms
64 bytes from 100.1.0.13: icmp_seq=2 ttl=64 time=0.592 ms
64 bytes from 100.1.0.13: icmp_seq=3 ttl=64 time=0.393 ms

--- 100.1.0.13 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time
2032ms
rtt min/avg/max/mdev = 0.393/0.741/1.238/0.360 ms
r4:~#
```

checking connectivity

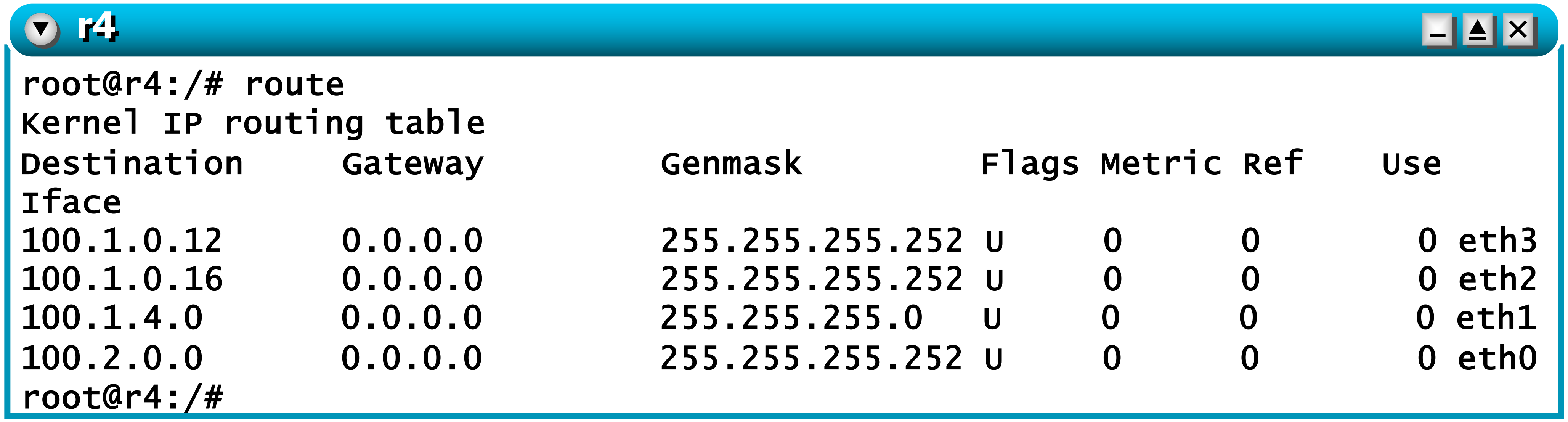
- towards a remote destination



```
r4:~# ping 100.1.2.1
connect: Network is unreachable
r4:~#
```

- what’s going on?

examining the kernel routing table

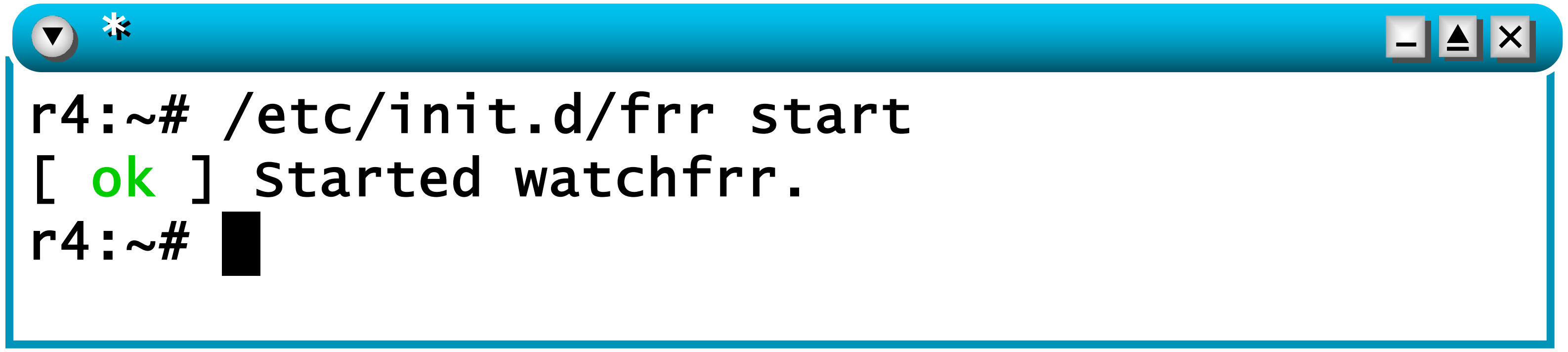


```
root@r4:/# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
100.1.0.12      0.0.0.0         255.255.255.252 U     0      0        0 eth3
100.1.0.16      0.0.0.0         255.255.255.252 U     0      0        0 eth2
100.1.4.0       0.0.0.0         255.255.255.0   U     0      0        0 eth1
100.2.0.0       0.0.0.0         255.255.255.252 U     0      0        0 eth0
root@r4:/#
```

- since no routing daemon is currently running, only directly connected destinations are known to the router

starting the routing daemons

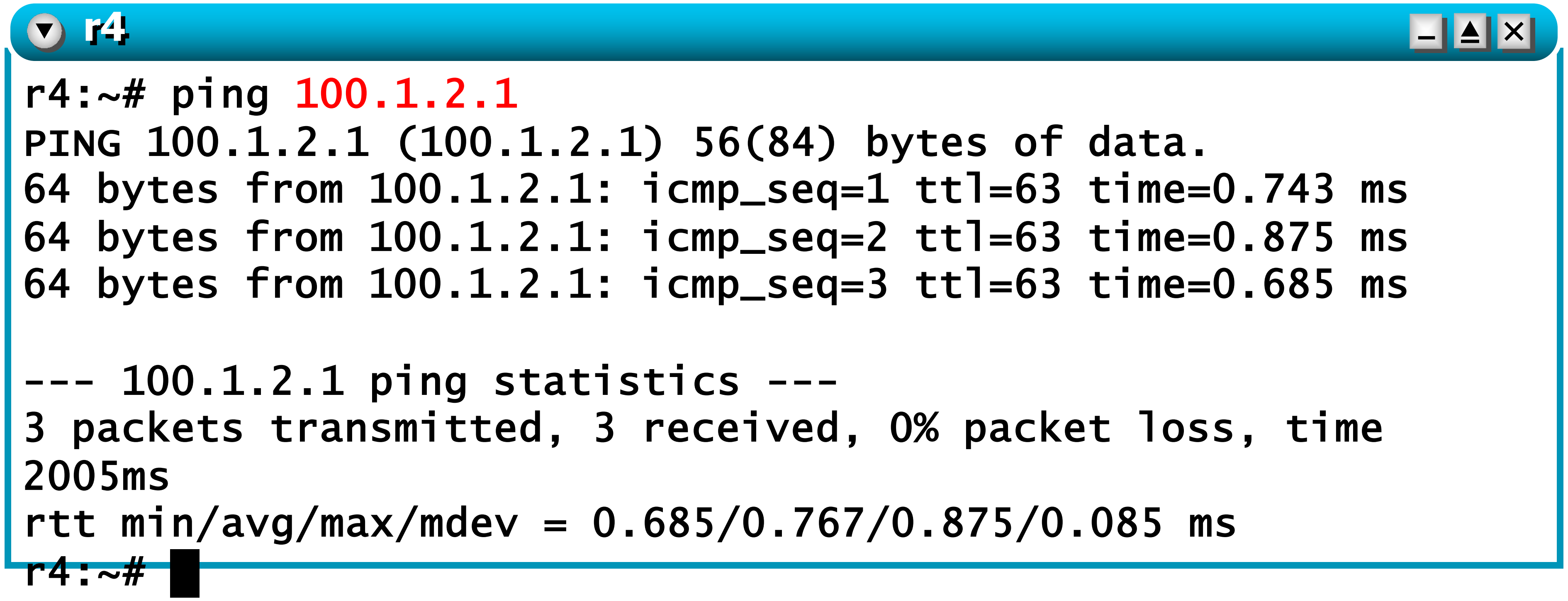
- on each router (but **`r5`**) issue the following command:



```
r4:~# /etc/init.d/frr start
[ ok ] Started watchfrr.
r4:~# █
```

checking connectivity (again)

- towards a remote destination



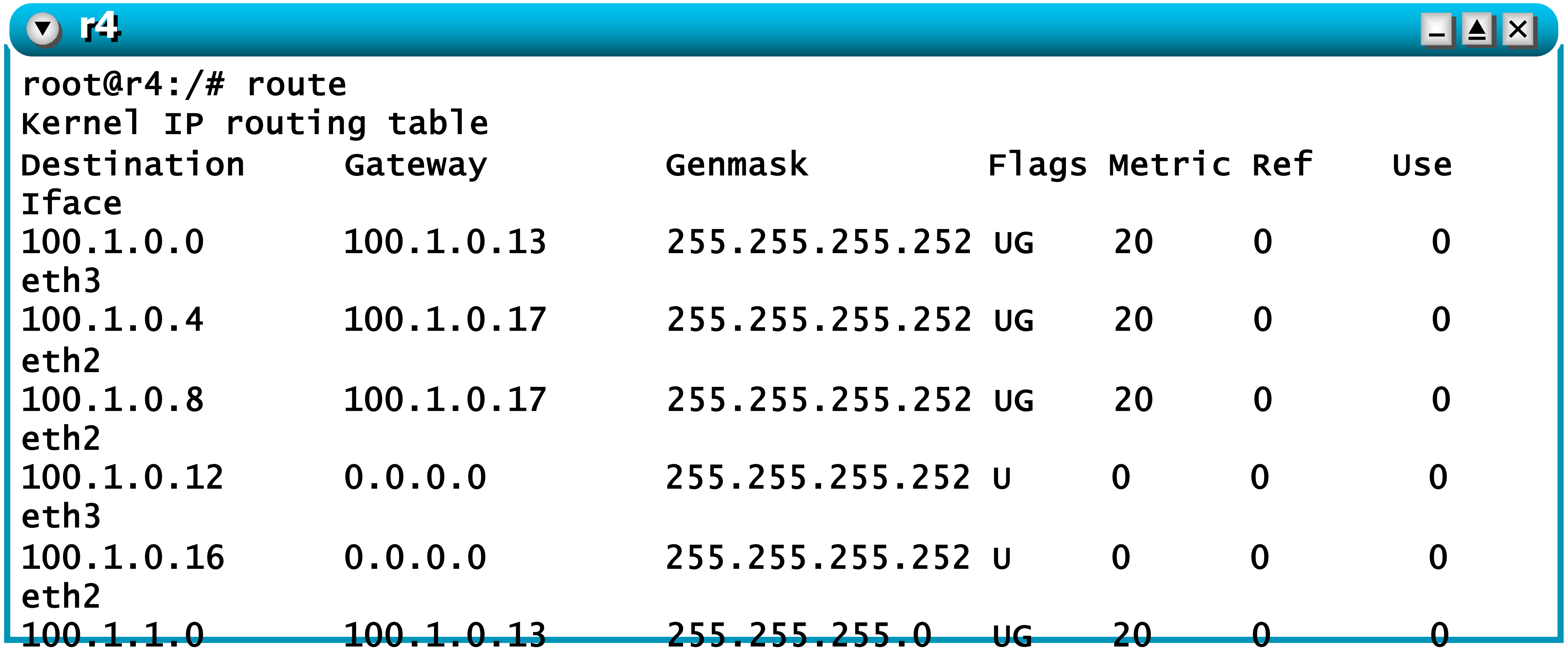
r4

```
r4:~# ping 100.1.2.1
PING 100.1.2.1 (100.1.2.1) 56(84) bytes of data.
64 bytes from 100.1.2.1: icmp_seq=1 ttl=63 time=0.743 ms
64 bytes from 100.1.2.1: icmp_seq=2 ttl=63 time=0.875 ms
64 bytes from 100.1.2.1: icmp_seq=3 ttl=63 time=0.685 ms

--- 100.1.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time
2005ms
rtt min/avg/max/mdev = 0.685/0.767/0.875/0.085 ms
r4:~#
```

- after a while, all remote destinations are reachable

checking the routing table

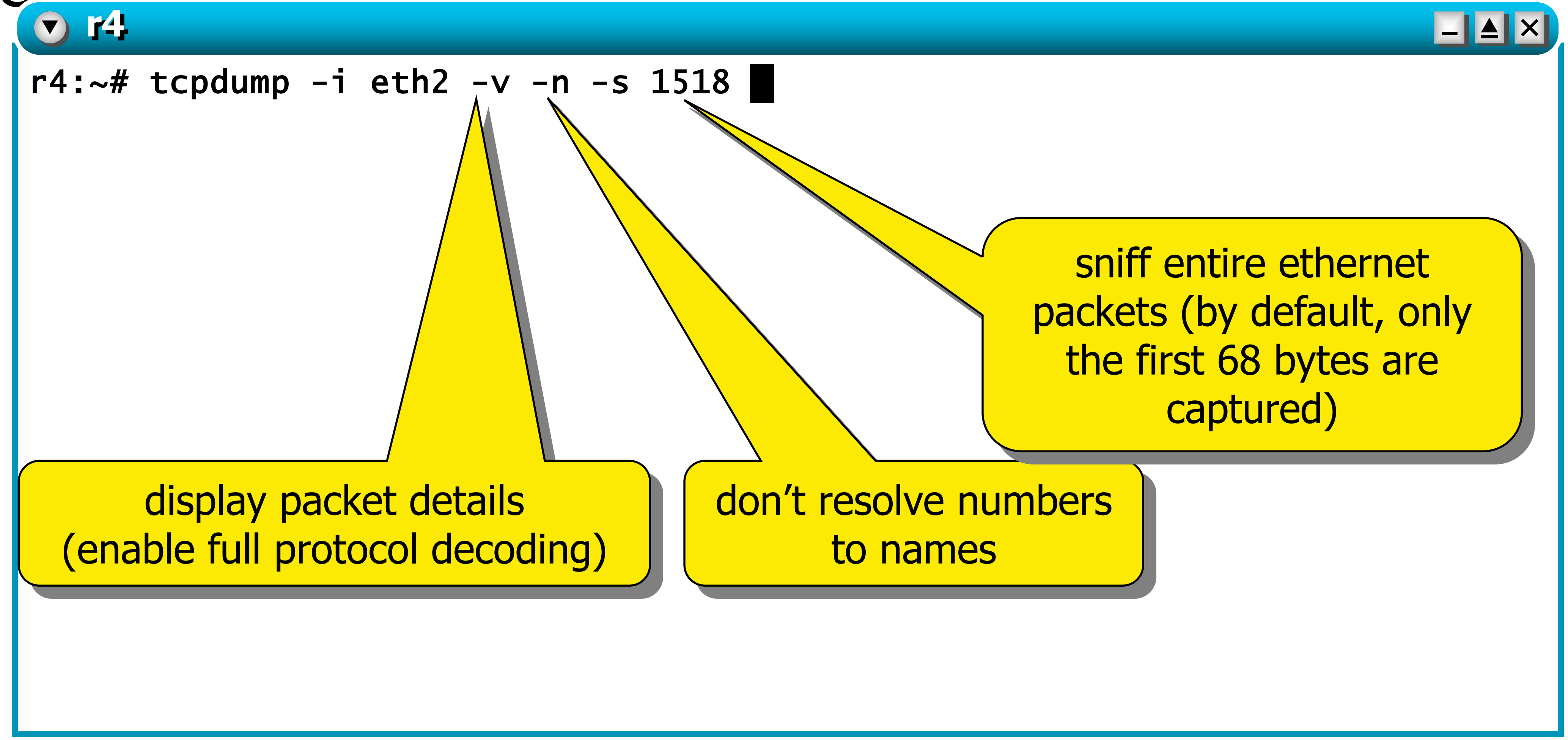
- the routing table is now updated


r4

```
root@r4:/# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
100.1.0.0       100.1.0.13      255.255.255.252 UG    20     0        0
eth3
100.1.0.4       100.1.0.17      255.255.255.252 UG    20     0        0
eth2
100.1.0.8       100.1.0.17      255.255.255.252 UG    20     0        0
eth2
100.1.0.12      0.0.0.0         255.255.255.252 U     0      0        0
eth3
100.1.0.16      0.0.0.0         255.255.255.252 U     0      0        0
eth2
100.1.1.0       100.1.0.13      255.255.255.0   UG    20     0        0
eth3
100.1.2.0       100.1.0.17      255.255.255.0   UG    20     0        0
eth2
```

a look at ripv2 packets

- let's sniff ripv2 packets



r4

```
r4:~# tcpdump -i eth2 -v -n -s 1518
```

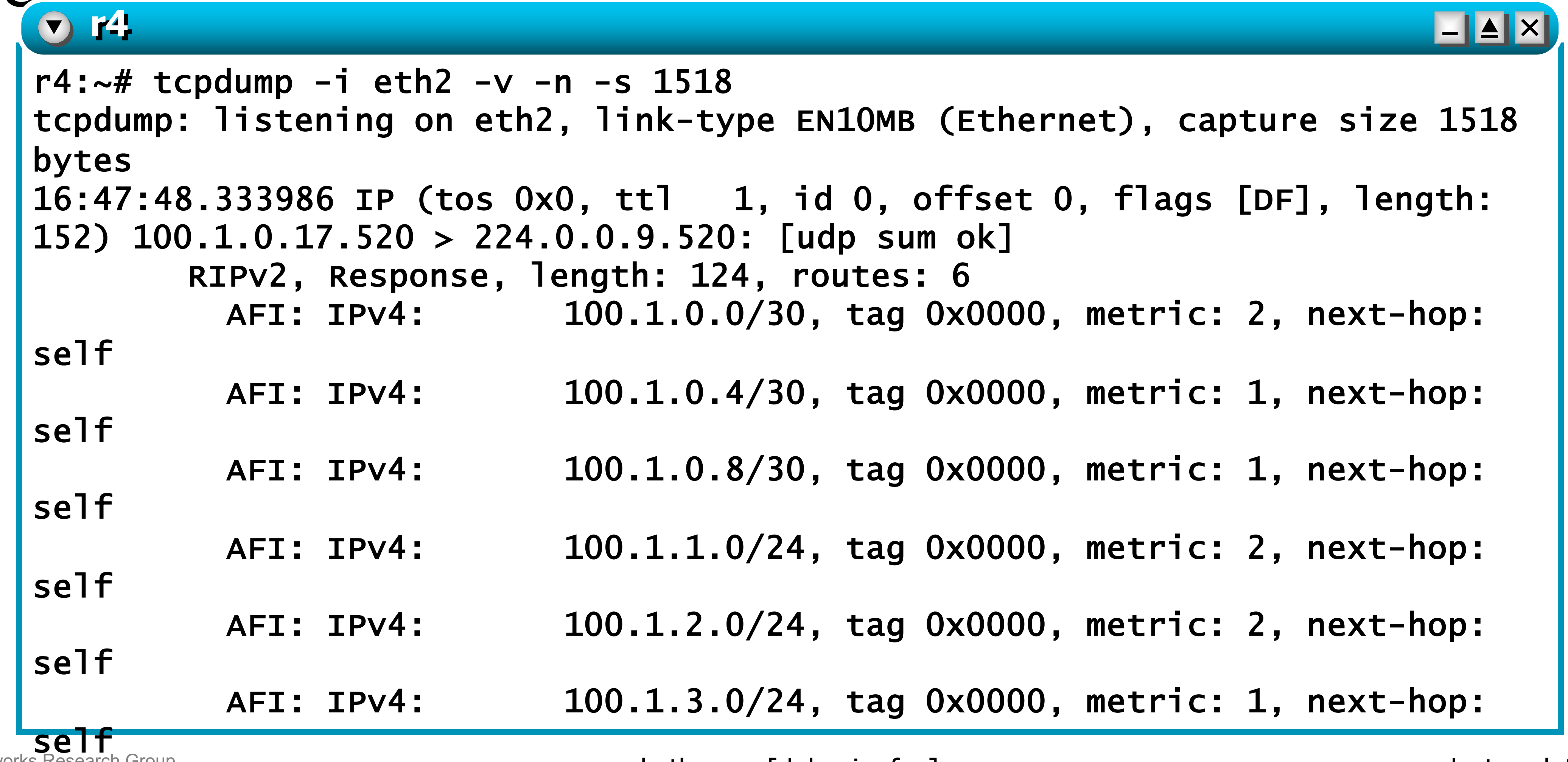
display packet details (enable full protocol decoding)

don't resolve numbers to names

sniff entire ethernet packets (by default, only the first 68 bytes are captured)

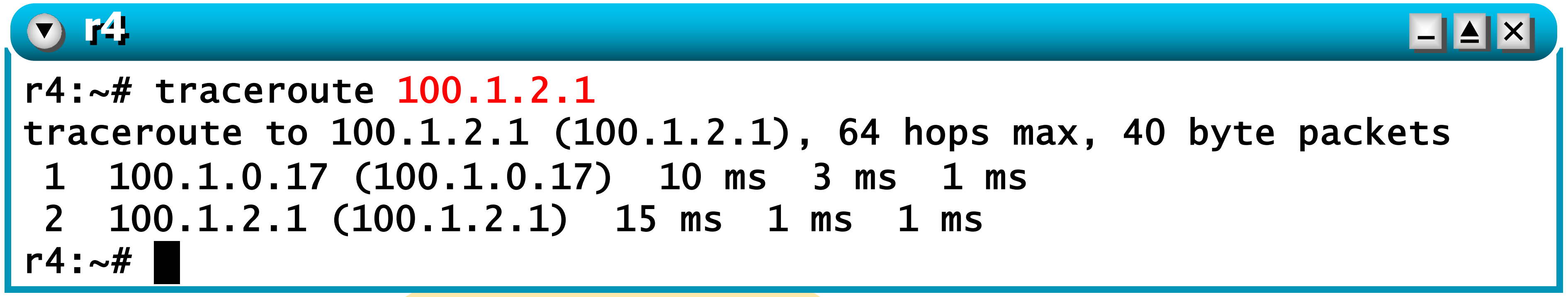
a look at ripv2 packets

- let's sniff ripv2 packets

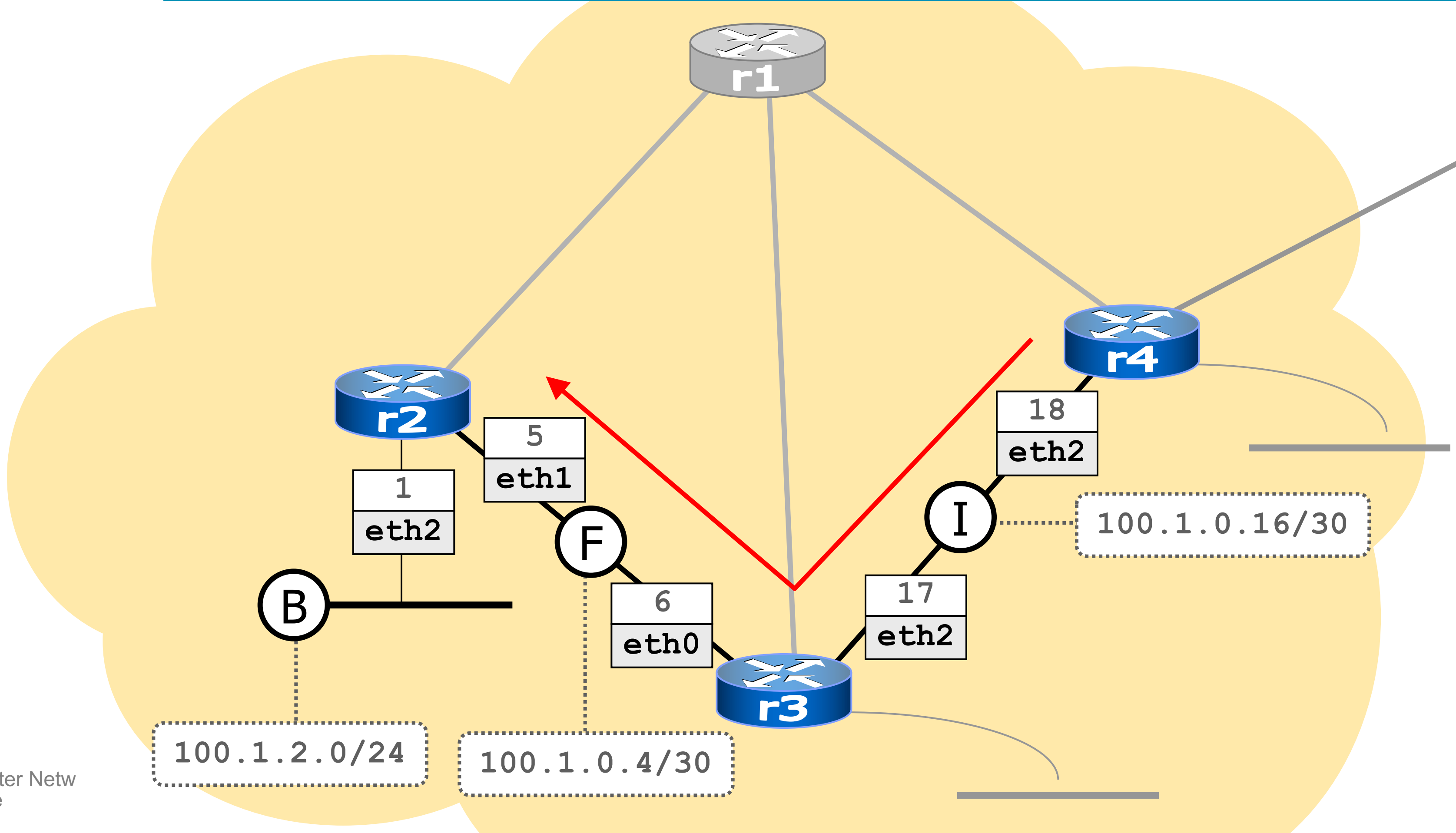


```
r4:~# tcpdump -i eth2 -v -n -s 1518
tcpdump: listening on eth2, link-type EN10MB (Ethernet), capture size 1518
bytes
16:47:48.333986 IP (tos 0x0, ttl   1, id 0, offset 0, flags [DF], length:
152) 100.1.0.17.520 > 224.0.0.9.520: [udp sum ok]
        RIPv2, Response, length: 124, routes: 6
          AFI: IPv4:       100.1.0.0/30, tag 0x0000, metric: 2, next-hop:
self
          AFI: IPv4:       100.1.0.4/30, tag 0x0000, metric: 1, next-hop:
self
          AFI: IPv4:       100.1.0.8/30, tag 0x0000, metric: 1, next-hop:
self
          AFI: IPv4:       100.1.1.0/24, tag 0x0000, metric: 2, next-hop:
self
          AFI: IPv4:       100.1.2.0/24, tag 0x0000, metric: 2, next-hop:
self
          AFI: IPv4:       100.1.3.0/24, tag 0x0000, metric: 1, next-hop:
self
```

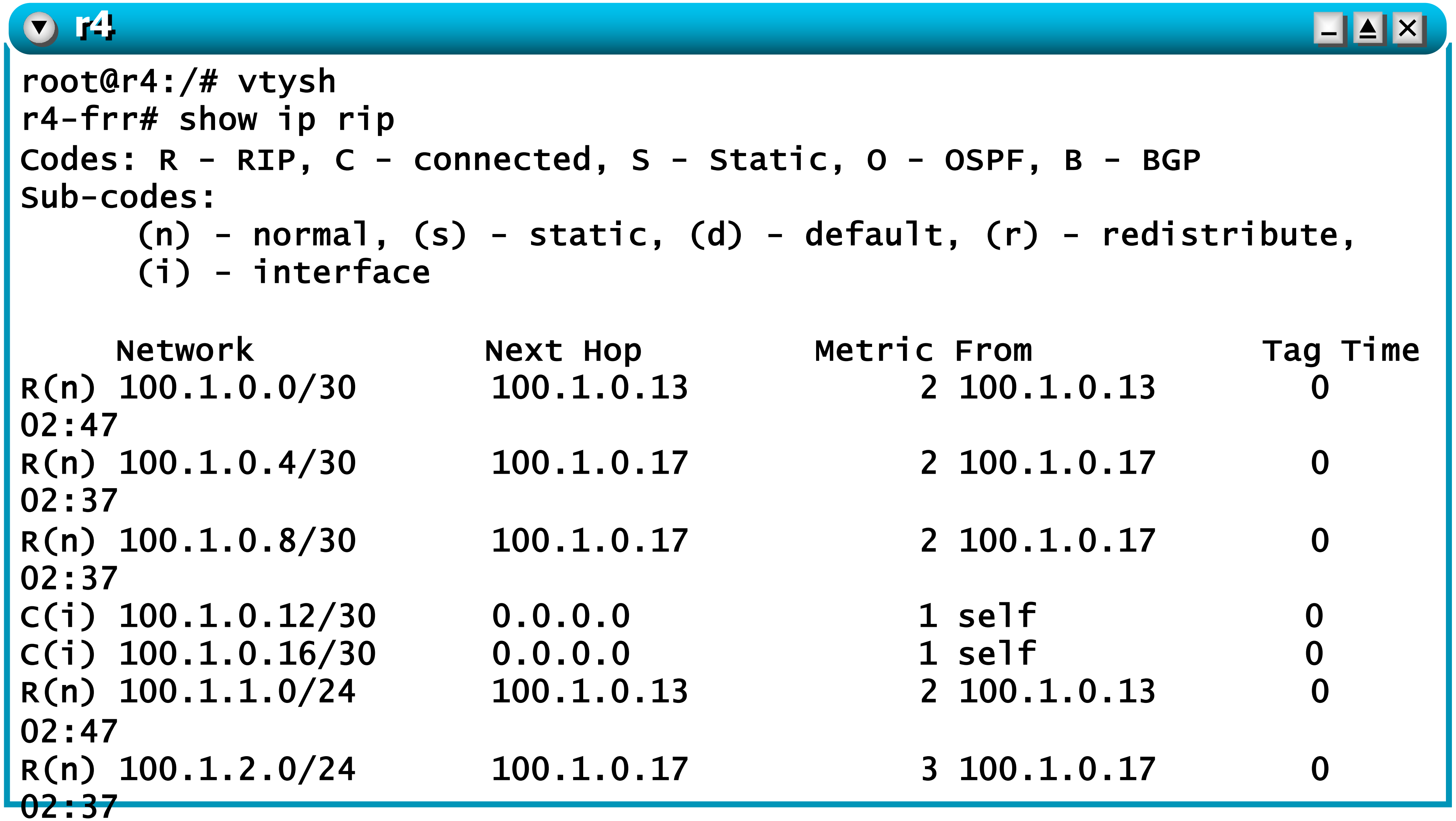
a traceroute



```
r4:~# traceroute 100.1.2.1
traceroute to 100.1.2.1 (100.1.2.1), 64 hops max, 40 byte packets
 1  100.1.0.17 (100.1.0.17)  10 ms  3 ms  1 ms
 2  100.1.2.1 (100.1.2.1)  15 ms  1 ms  1 ms
r4:~# 
```



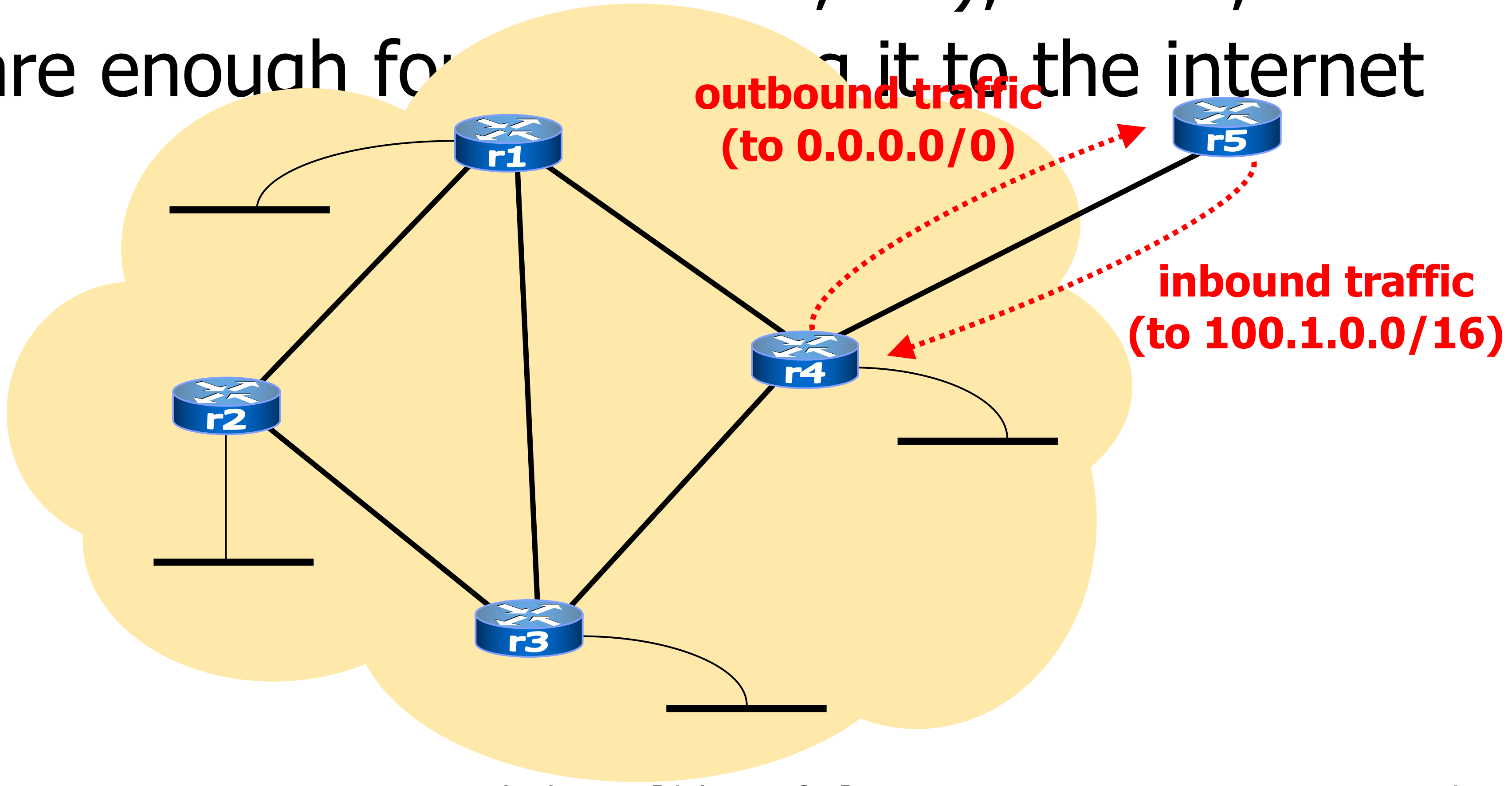
inspecting the rip routing table


r4

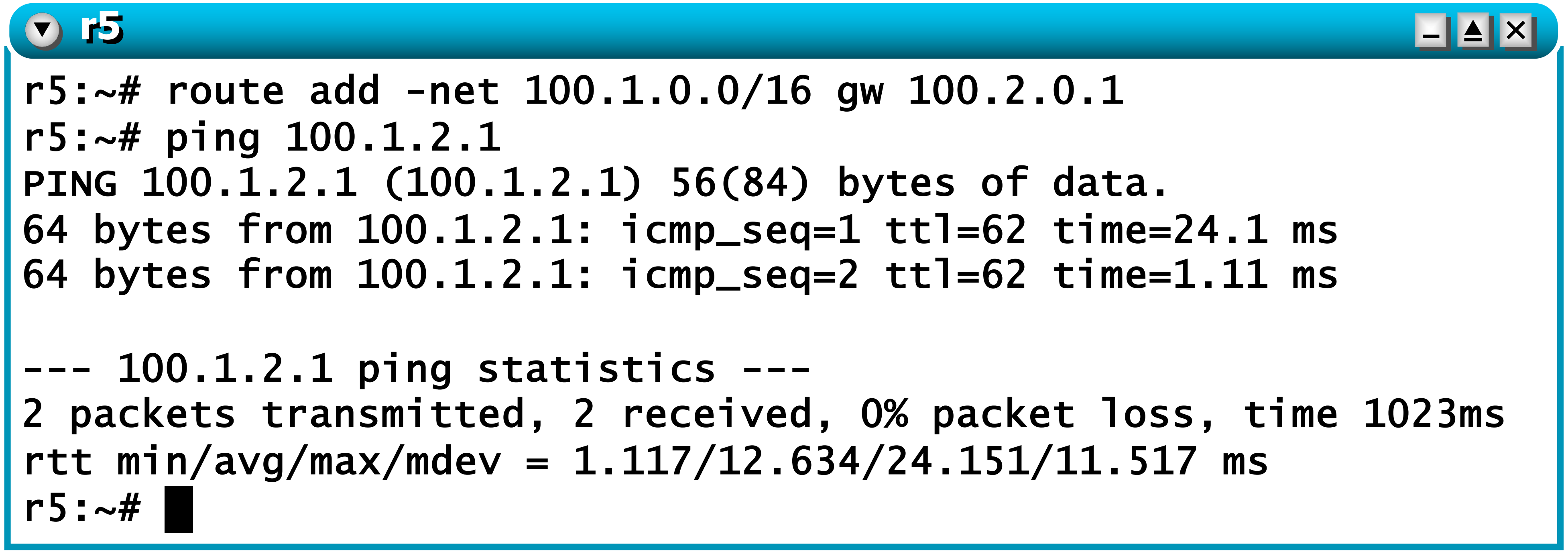
```
root@r4:/# vtysh
r4-frr# show ip rip
Codes: R - RIP, C - connected, S - Static, O - OSPF, B - BGP
Sub-codes:
      (n) - normal, (s) - static, (d) - default, (r) - redistribute,
      (i) - interface

     Network            Next Hop         Metric From            Tag Time
R(n) 100.1.0.0/30       100.1.0.13            2 100.1.0.13        0
02:47
R(n) 100.1.0.4/30       100.1.0.17            2 100.1.0.17        0
02:37
R(n) 100.1.0.8/30       100.1.0.17            2 100.1.0.17        0
02:37
C(i) 100.1.0.12/30      0.0.0.0               1 self              0
C(i) 100.1.0.16/30      0.0.0.0               1 self              0
R(n) 100.1.1.0/24       100.1.0.13            2 100.1.0.13        0
02:47
R(n) 100.1.2.0/24       100.1.0.17            3 100.1.0.17        0
02:37
R(n) 100.1.3.0/24       100.1.0.17            2 100.1.0.17        0
02:37
```

static routing

- our network is a stub network (i.e., it has just one connection to an external router, `r5`); hence, static routes are enough fo g it to the internet

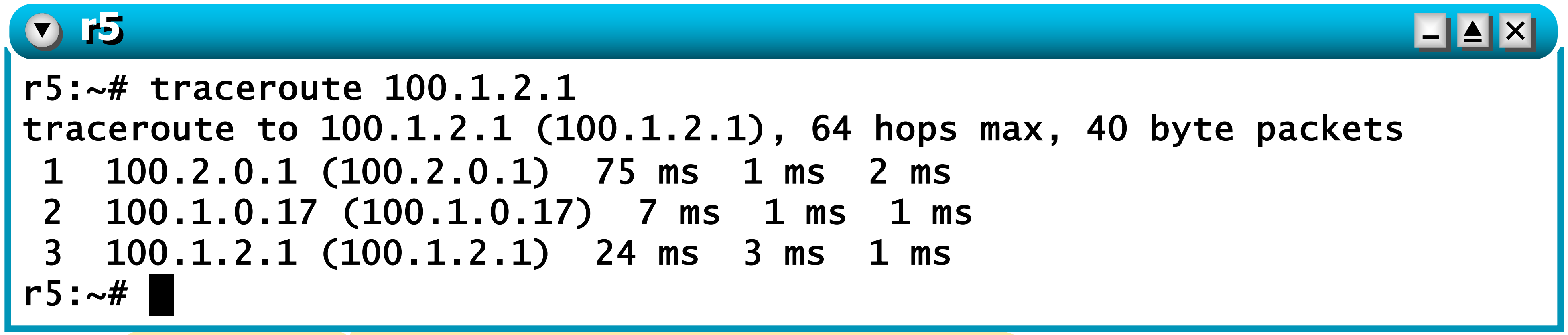


adding a static route to r5

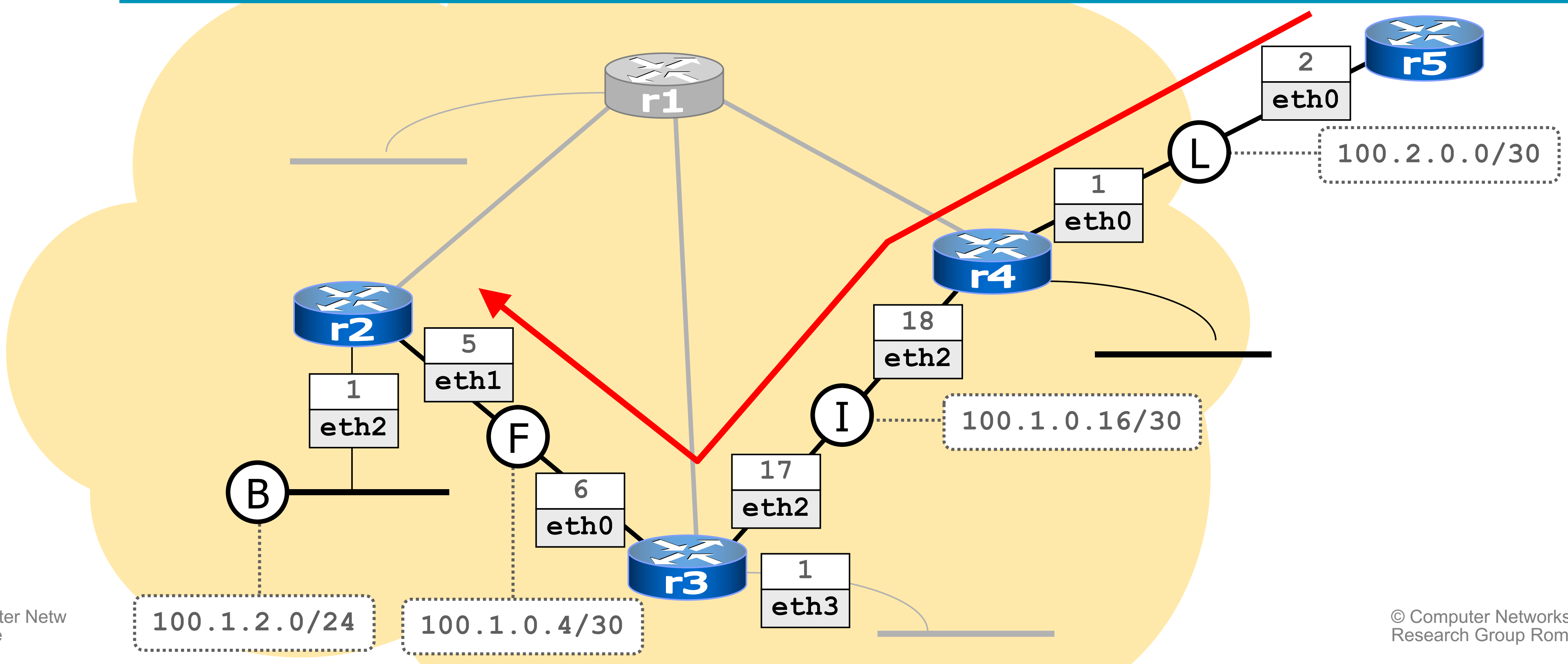

r5

```
r5:~# route add -net 100.1.0.0/16 gw 100.2.0.1
r5:~# ping 100.1.2.1
PING 100.1.2.1 (100.1.2.1) 56(84) bytes of data.
64 bytes from 100.1.2.1: icmp_seq=1 ttl=62 time=24.1 ms
64 bytes from 100.1.2.1: icmp_seq=2 ttl=62 time=1.11 ms

--- 100.1.2.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1023ms
rtt min/avg/max/mdev = 1.117/12.634/24.151/11.517 ms
r5:~# █
```

checking connectivity

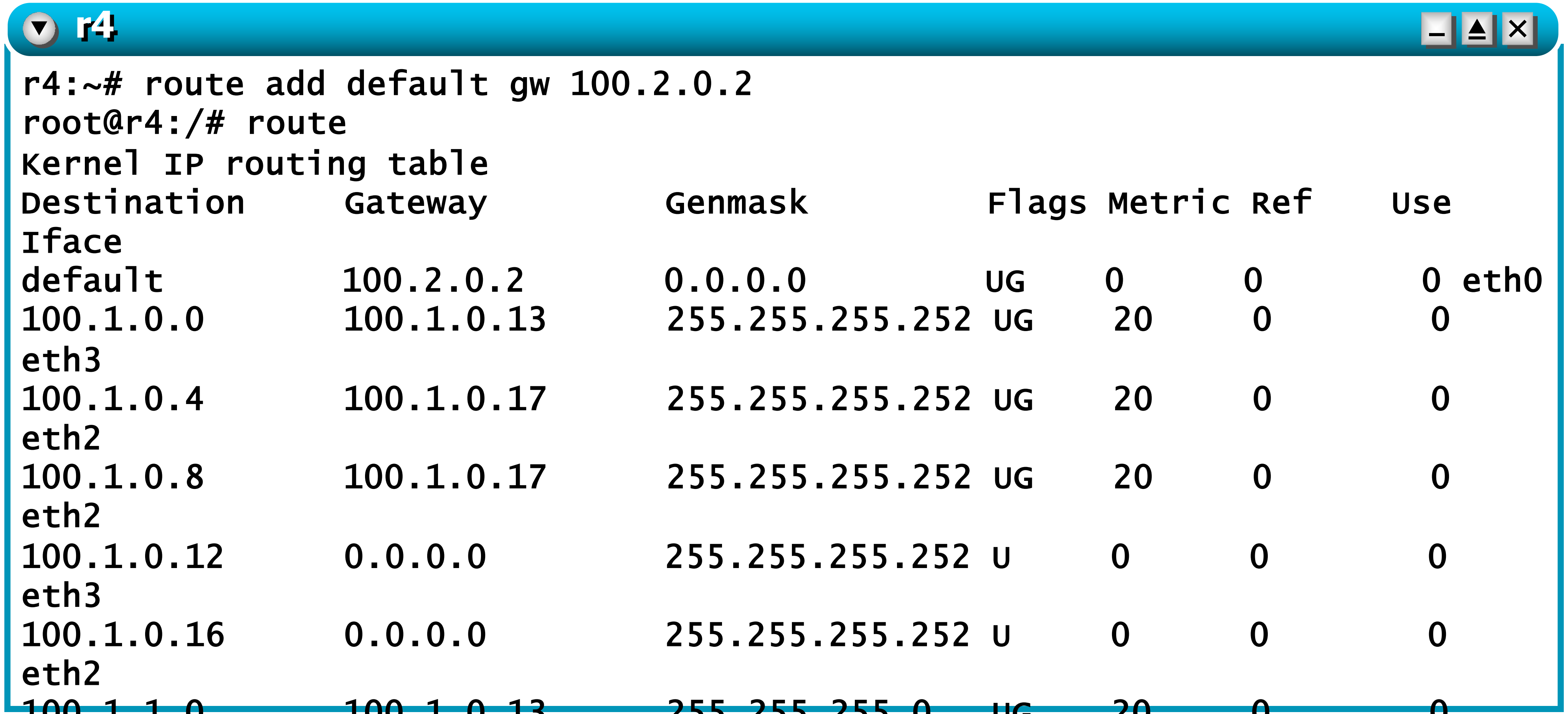


```
r5:~# traceroute 100.1.2.1
traceroute to 100.1.2.1 (100.1.2.1), 64 hops max, 40 byte packets
 1  100.2.0.1 (100.2.0.1)  75 ms  1 ms  2 ms
 2  100.1.0.17 (100.1.0.17)  7 ms  1 ms  1 ms
 3  100.1.2.1 (100.1.2.1)  24 ms  3 ms  1 ms
r5:~# █
```



configuring r4

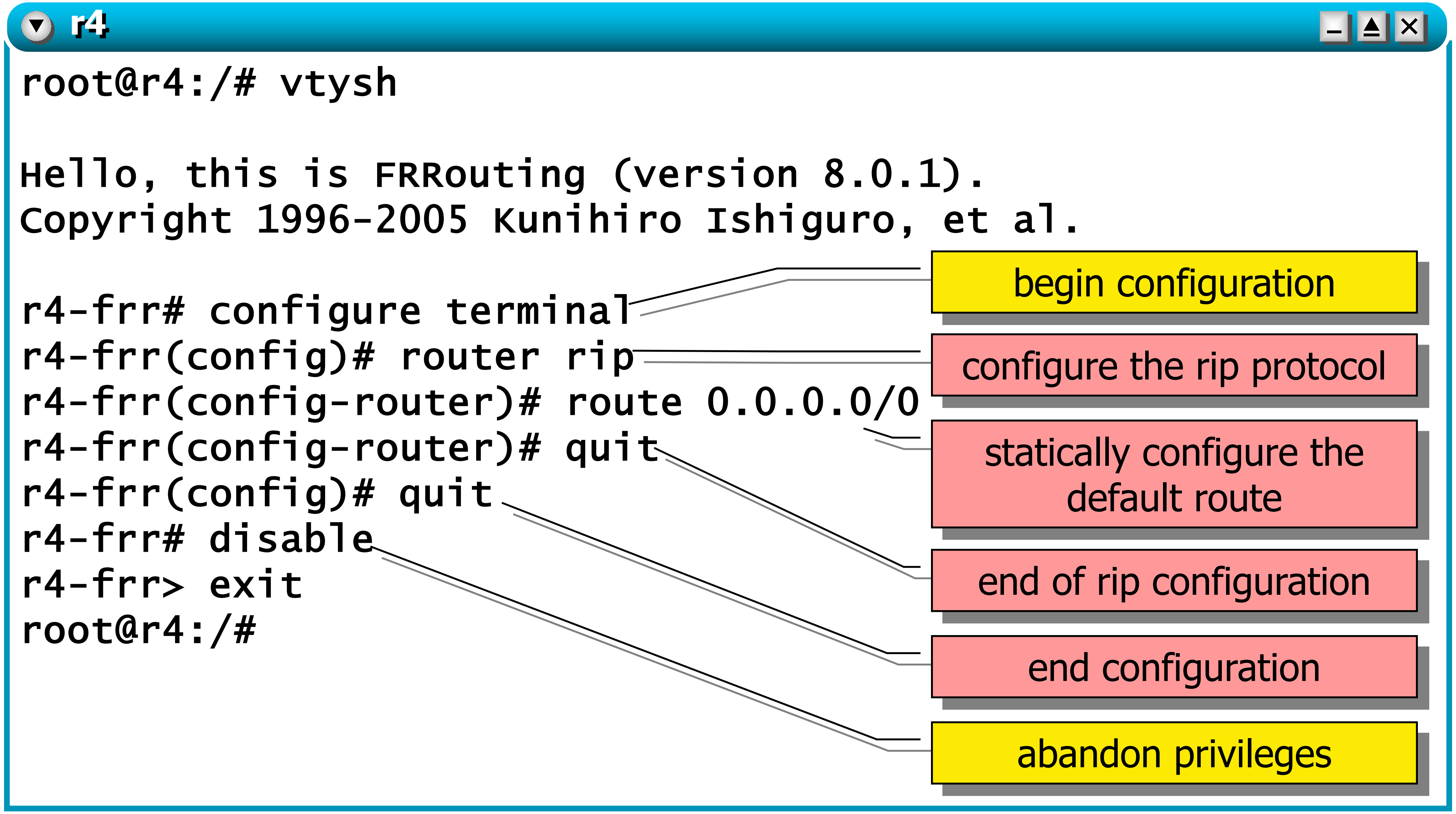
- step 1: configuring the default route



```
r4:~# route add default gw 100.2.0.2
root@r4:/# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
default         100.2.0.2       0.0.0.0         UG    0      0        0 eth0
100.1.0.0       100.1.0.13      255.255.255.252 UG    20     0        0
eth3
100.1.0.4       100.1.0.17      255.255.255.252 UG    20     0        0
eth2
100.1.0.8       100.1.0.17      255.255.255.252 UG    20     0        0
eth2
100.1.0.12      0.0.0.0         255.255.255.252 U     0      0        0
eth3
100.1.0.16      0.0.0.0         255.255.255.252 U     0      0        0
eth2
100.1.1.0       100.1.0.13      255.255.255.0   UG    20     0        0
eth3
100.1.2.0       100.1.0.17      255.255.255.0   UG    20     0        0
```

configuring **r4**

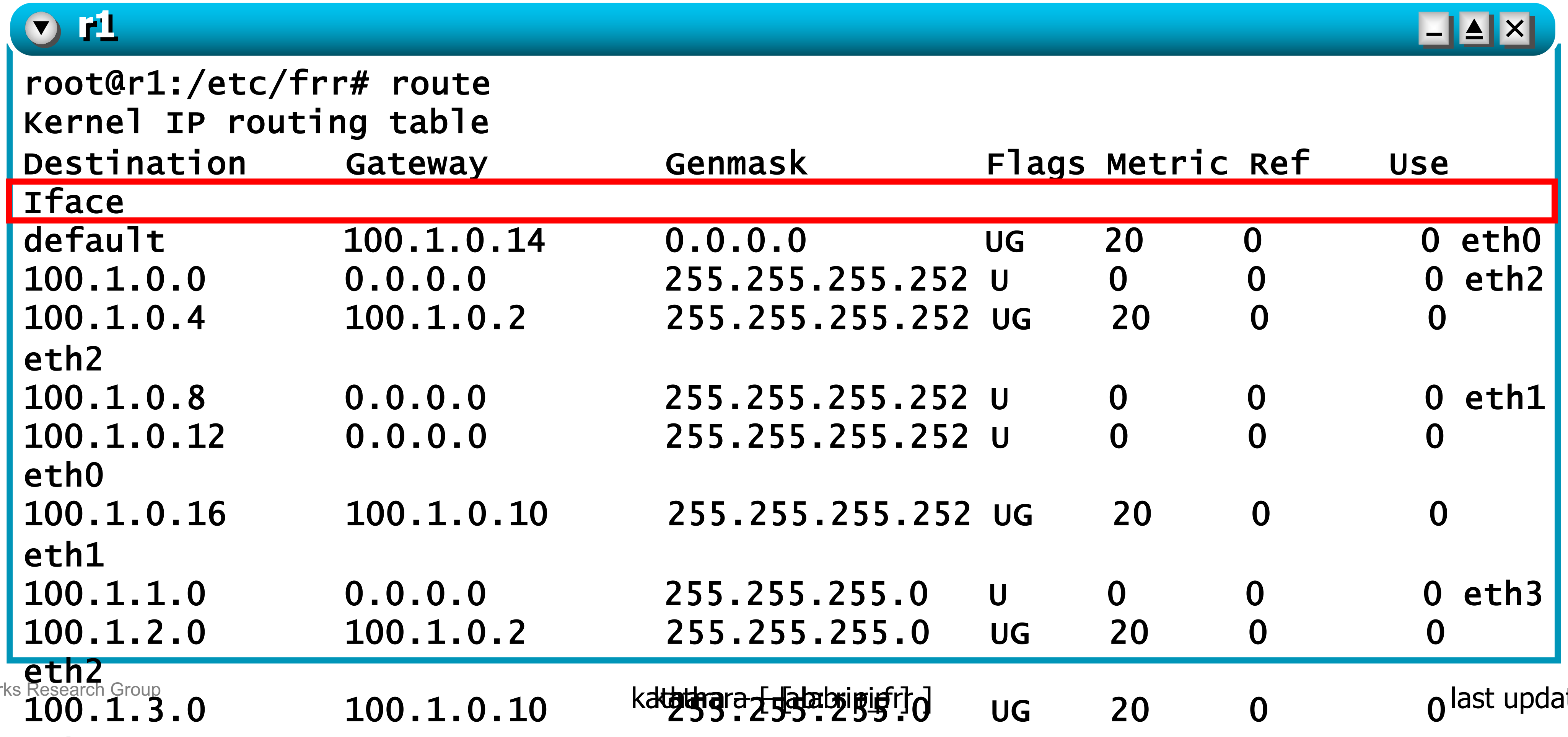
- step 2: propagating the default route into rip



```
root@r4:/# vtysh

Hello, this is FRRouting (version 8.0.1).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

r4-frr# configure terminal
r4-frr(config)# router rip
r4-frr(config-router)# route 0.0.0.0/0
r4-frr(config-router)# quit
r4-frr(config)# quit
r4-frr# disable
r4-frr> exit
root@r4:/#
```

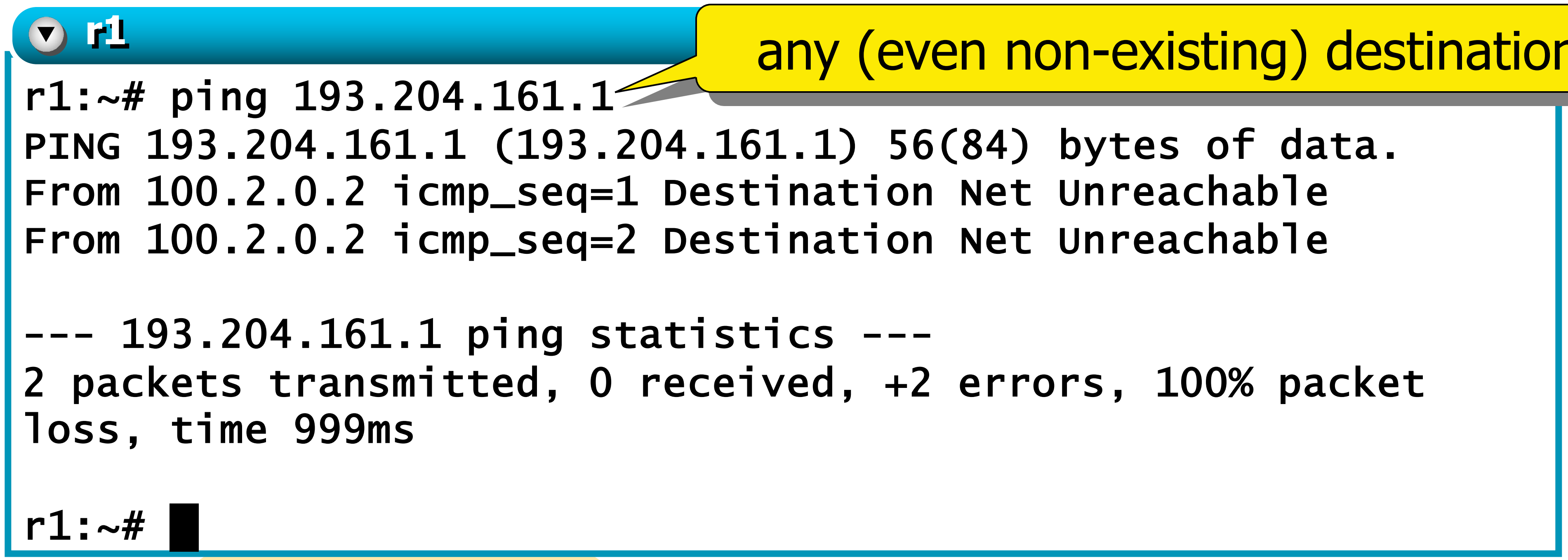
the default route

- after a while, the default route has been injected (via rip) into the network



```
root@r1:/etc/frr# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
default         100.1.0.14      0.0.0.0         UG    20     0        0 eth0
100.1.0.0       0.0.0.0         255.255.255.252 U     0      0        0 eth2
100.1.0.4       100.1.0.2       255.255.255.252 UG    20     0        0
eth2
100.1.0.8       0.0.0.0         255.255.255.252 U     0      0        0 eth1
100.1.0.12      0.0.0.0         255.255.255.252 U     0      0        0
eth0
100.1.0.16      100.1.0.10      255.255.255.252 UG    20     0        0
eth1
100.1.1.0       0.0.0.0         255.255.255.0   U     0      0        0 eth3
100.1.2.0       100.1.0.2       255.255.255.0   UG    20     0        0
eth2
100.1.3.0       100.1.0.10      255.255.255.0   UG    20     0        0
```

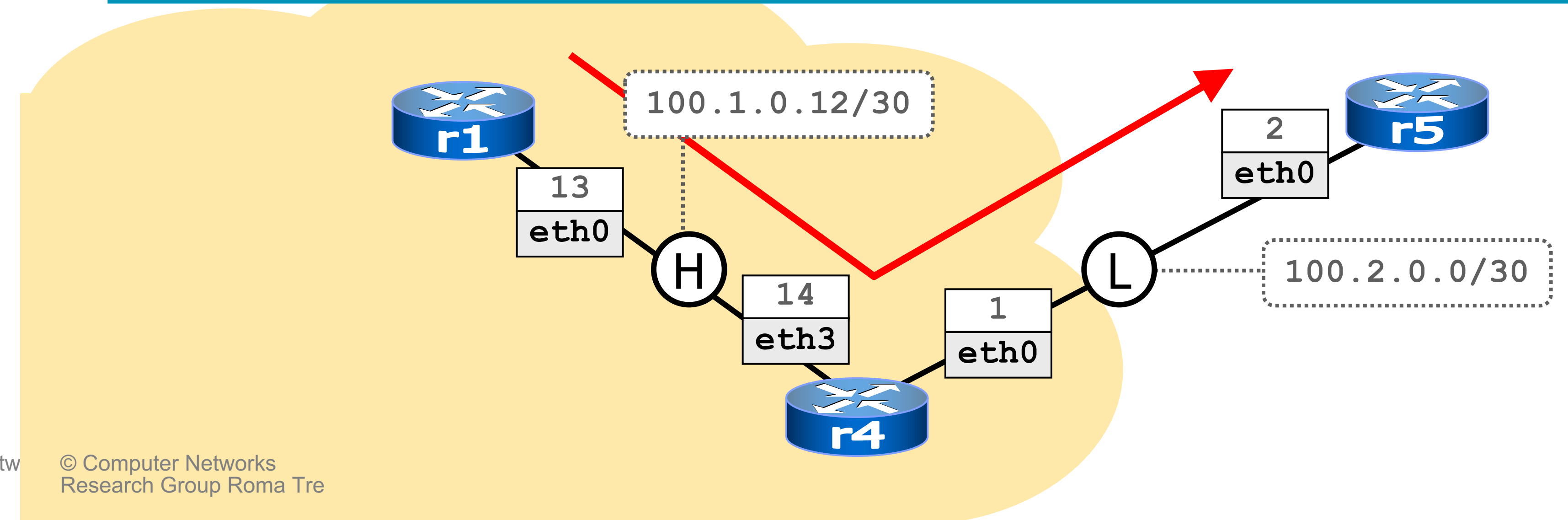
checking connectivity



r1

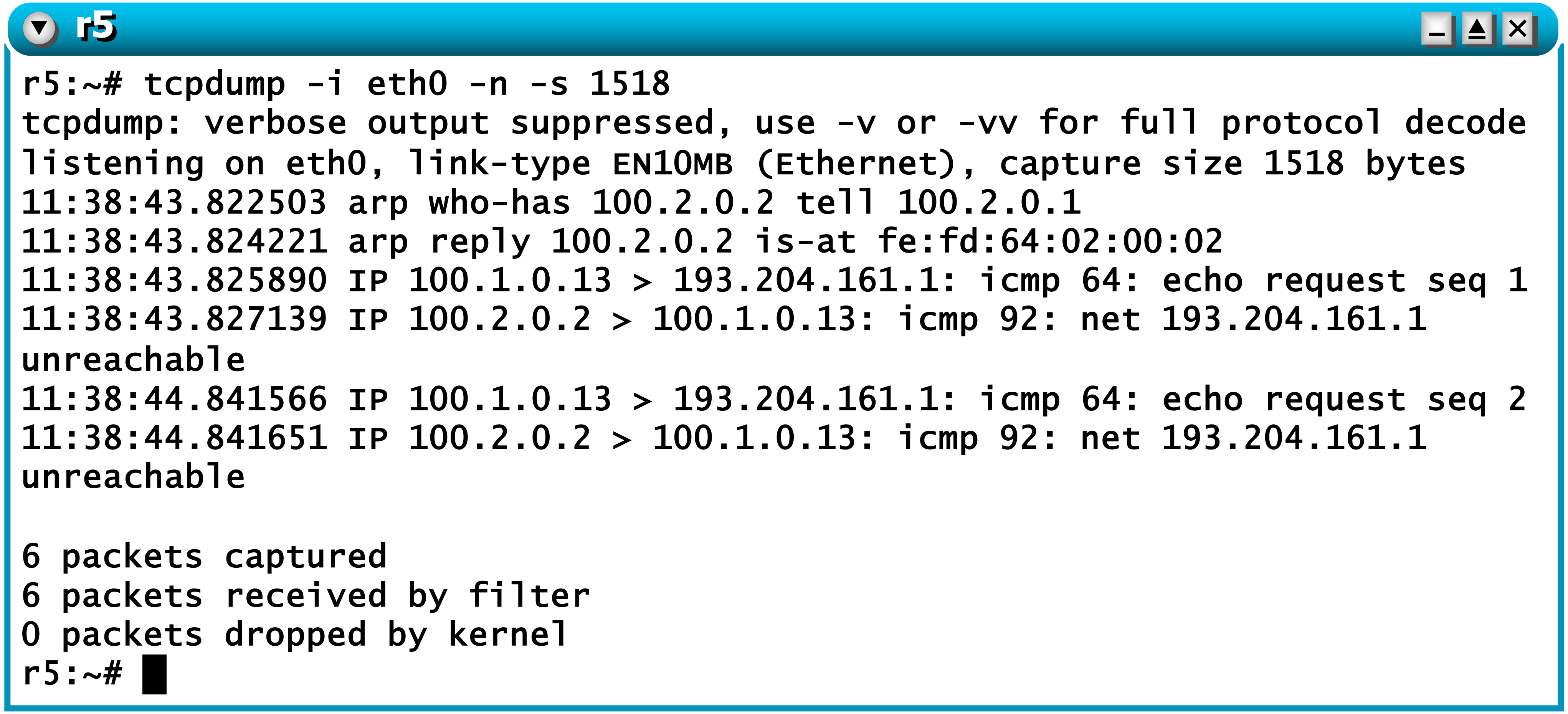
any (even non-existing) destination

```
r1:~# ping 193.204.161.1
PING 193.204.161.1 (193.204.161.1) 56(84) bytes of data.
From 100.2.0.2 icmp_seq=1 Destination Net Unreachable
From 100.2.0.2 icmp_seq=2 Destination Net Unreachable

--- 193.204.161.1 ping statistics ---
2 packets transmitted, 0 received, +2 errors, 100% packet
loss, time 999ms

r1:~#
```



checking connectivity

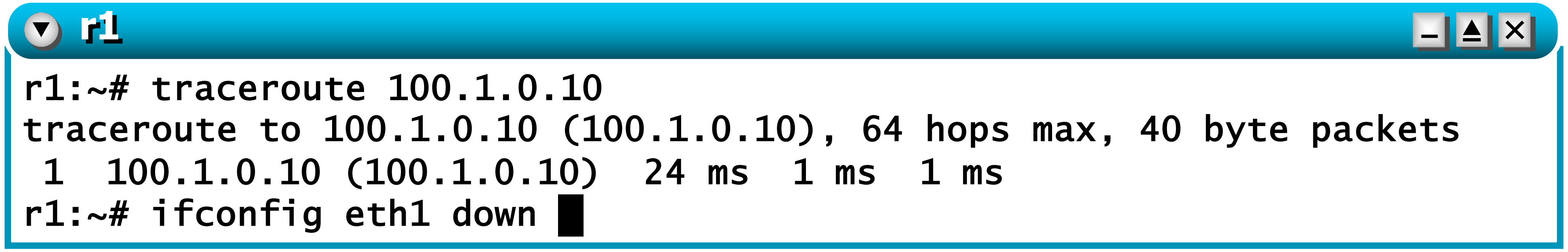
- **`r5`** is actually receiving echo request packets



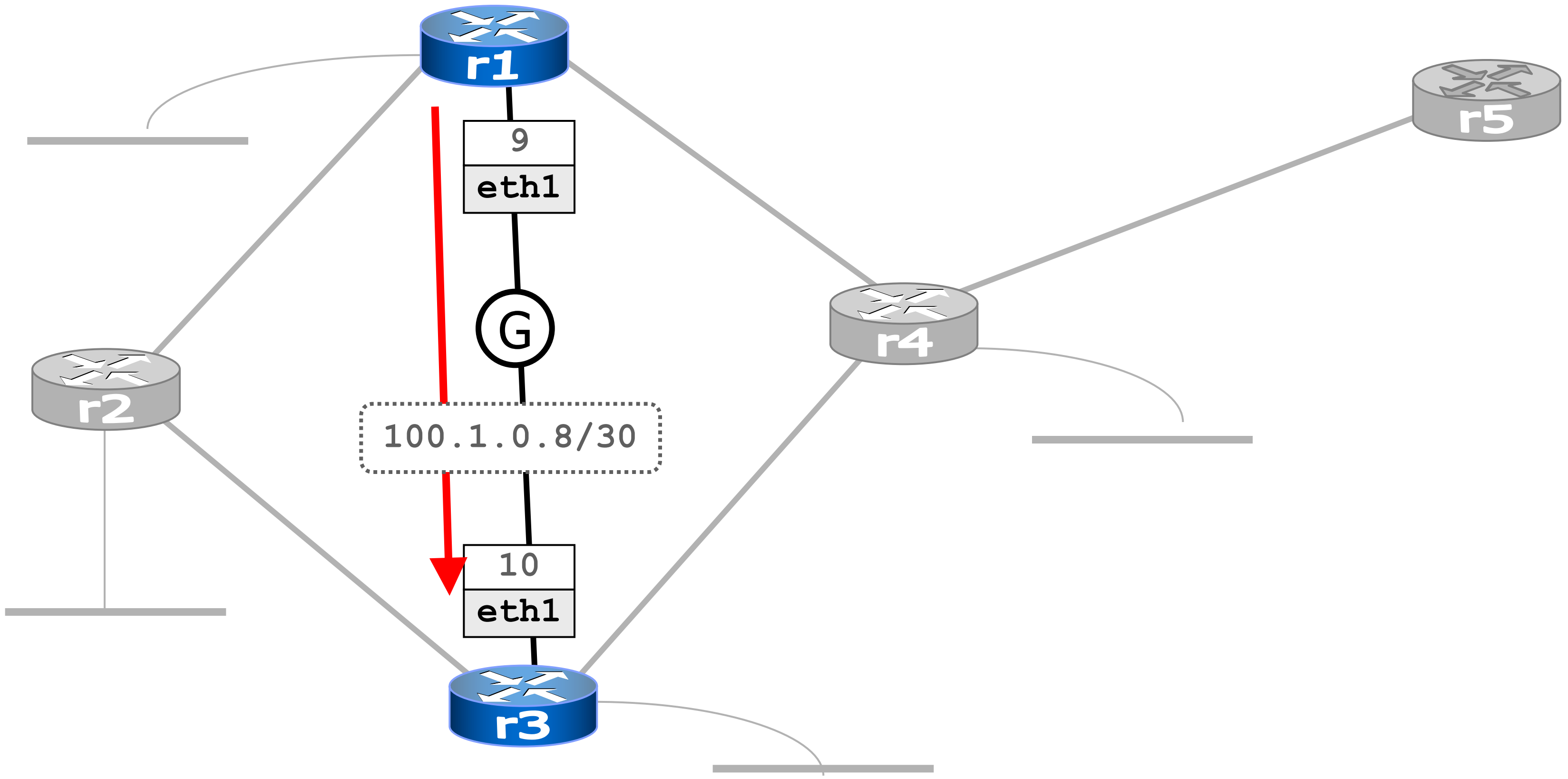
r5

```
r5:~# tcpdump -i eth0 -n -s 1518
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 1518 bytes
11:38:43.822503 arp who-has 100.2.0.2 tell 100.2.0.1
11:38:43.824221 arp reply 100.2.0.2 is-at fe:fd:64:02:00:02
11:38:43.825890 IP 100.1.0.13 > 193.204.161.1: icmp 64: echo request seq 1
11:38:43.827139 IP 100.2.0.2 > 100.1.0.13: icmp 92: net 193.204.161.1
unreachable
11:38:44.841566 IP 100.1.0.13 > 193.204.161.1: icmp 64: echo request seq 2
11:38:44.841651 IP 100.2.0.2 > 100.1.0.13: icmp 92: net 193.204.161.1
unreachable

6 packets captured
6 packets received by filter
0 packets dropped by kernel
r5:~# █
```

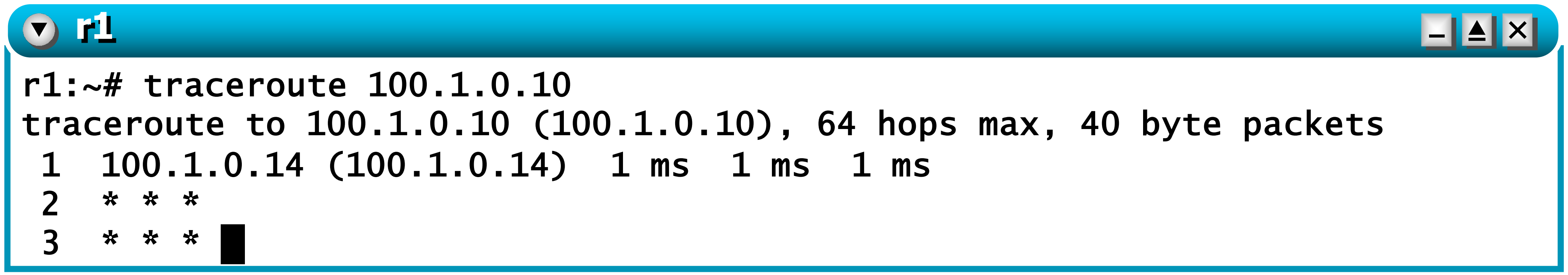
shutting down an interface



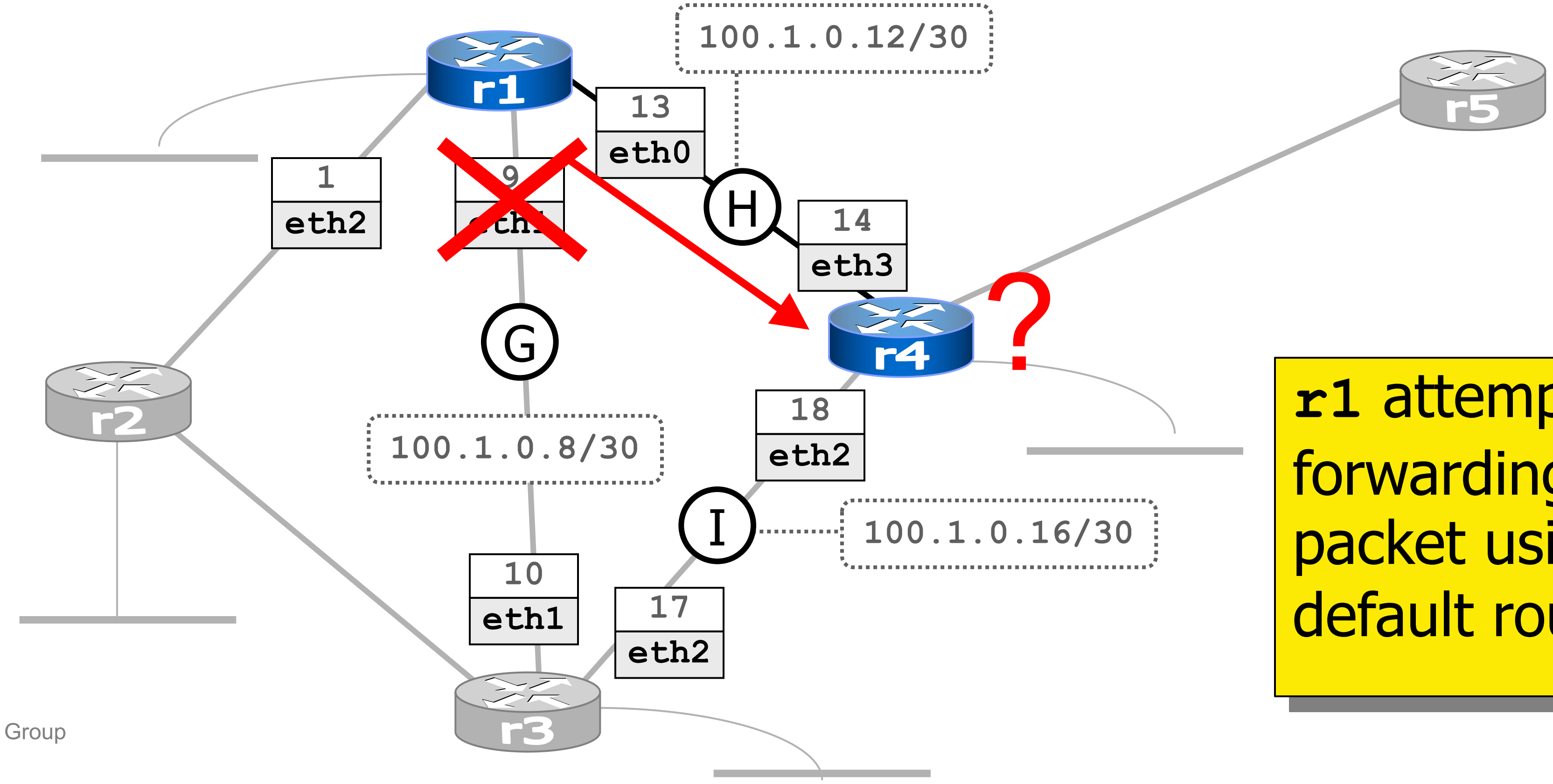
```
r1:~# traceroute 100.1.0.10
traceroute to 100.1.0.10 (100.1.0.10), 64 hops max, 40 byte packets
 1  100.1.0.10 (100.1.0.10)  24 ms  1 ms  1 ms
r1:~# ifconfig eth1 down
```



shutting down an interface

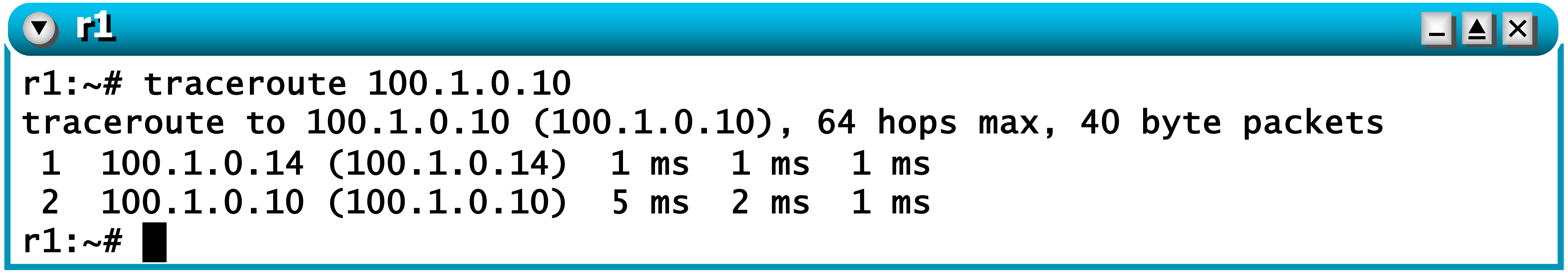


```
r1:~# traceroute 100.1.0.10
traceroute to 100.1.0.10 (100.1.0.10), 64 hops max, 40 byte packets
 1  100.1.0.14 (100.1.0.14)  1 ms  1 ms  1 ms
 2  * * *
 3  * * *
```

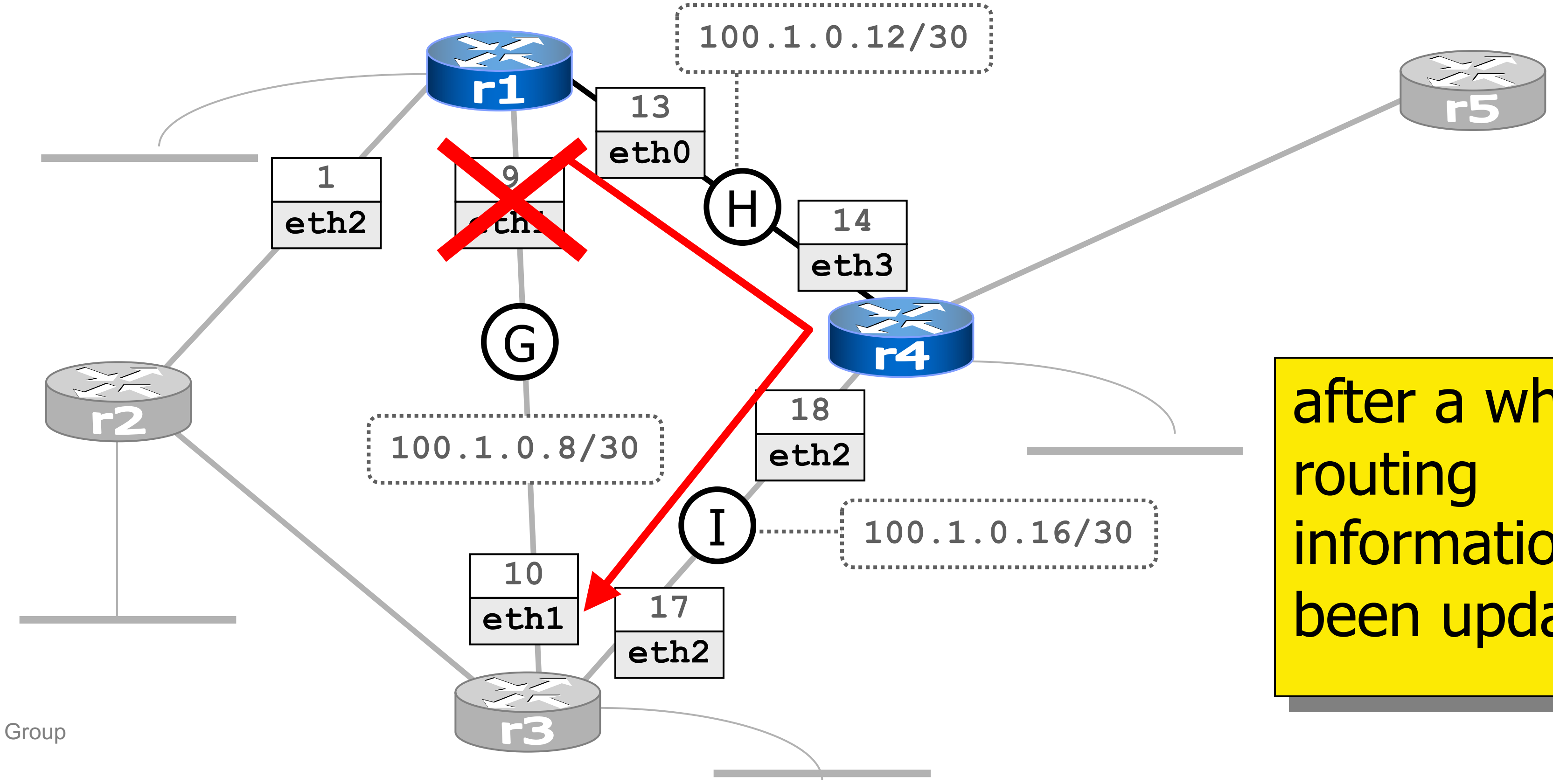


r1 attempts forwarding the packet using the default route

shutting down an interface



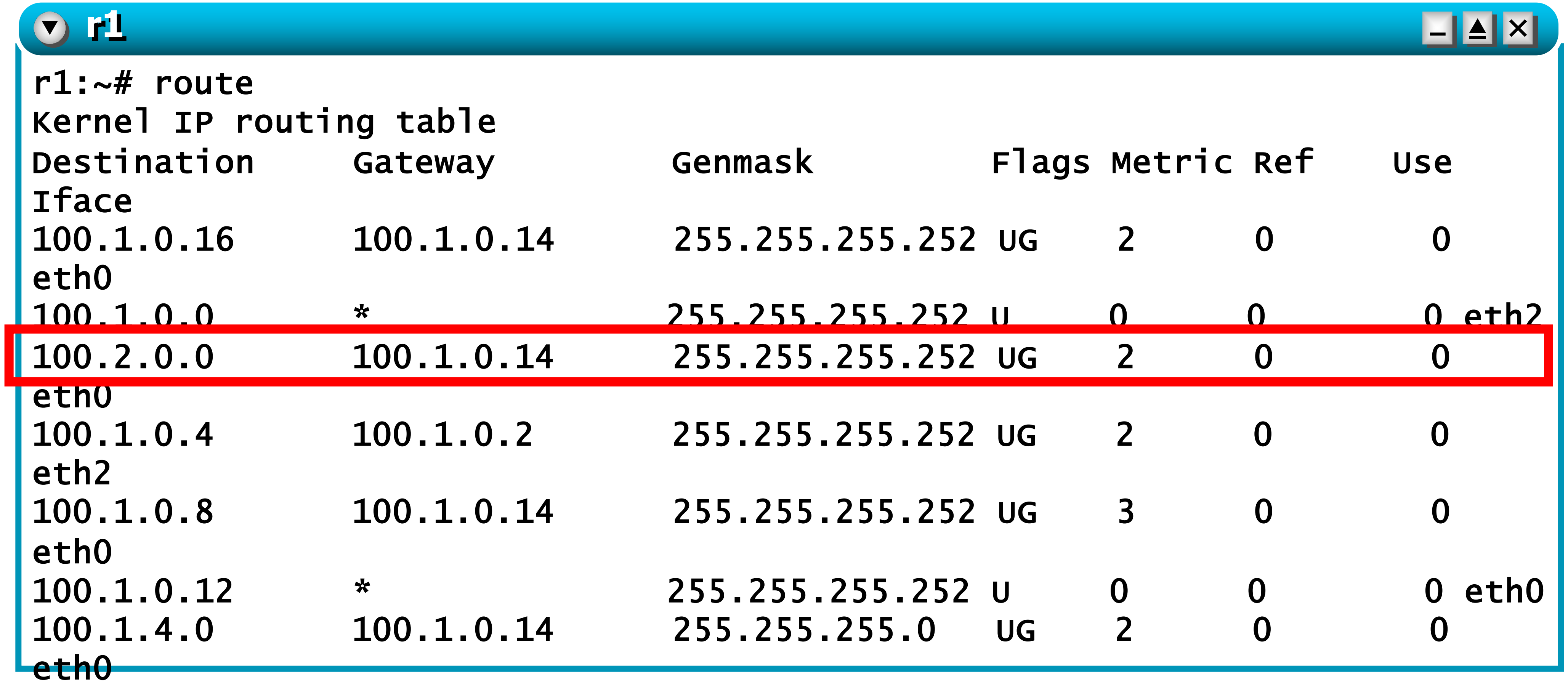
```
r1:~# traceroute 100.1.0.10
traceroute to 100.1.0.10 (100.1.0.10), 64 hops max, 40 byte packets
 1  100.1.0.14 (100.1.0.14)  1 ms  1 ms  1 ms
 2  100.1.0.10 (100.1.0.10)  5 ms  2 ms  1 ms
r1:~#
```



after a while, routing information has been updated

shutting down an interface

- **r1**'s routing table has been updated



```
r1:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use
Iface
100.1.0.16      100.1.0.14      255.255.255.252 UG    2      0        0
eth0
100.1.0.0       *               255.255.255.252 U     0      0        0 eth2
100.2.0.0       100.1.0.14      255.255.255.252 UG    2      0        0
eth0
100.1.0.4       100.1.0.2       255.255.255.252 UG    2      0        0
eth2
100.1.0.8       100.1.0.14      255.255.255.252 UG    3      0        0
eth0
100.1.0.12      *               255.255.255.252 U     0      0        0 eth0
100.1.4.0       100.1.0.14      255.255.255.0   UG    2      0        0
eth0
100.1.2.0       100.1.0.2       255.255.255.0   UG    2      0        0
eth2
100.1.3.0       100.1.0.14      255.255.255.0   UG    3      0        0
```