[544] Networking

Tyler Caraza-Harter

Learning Objectives

- explain how MAC addresses, IP addresses, and port numbers provide addressing, used to facilitate communicate between processes on different machines

- select IP addresses correctly for binding and using in a browser to achieve connection in the context of a server running behind a NAT

- identify the port number being used by a process

- select between transport methods (TCP and UDP) based on the functionality needed (on top of IP functionality)

Outline

Networks

Internets and "The Internet"

Transport Protocols

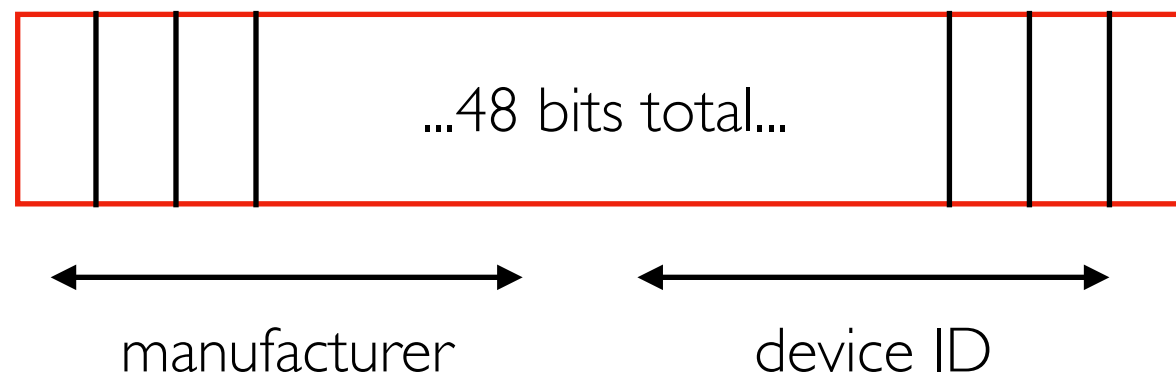# Network Interface Controllers and MAC Addresses

NICs can connect a computer to different physical mediums, such as:
- Ethernet (wired)
- Wi-Fi (wireless)

Every NIC in the world has a unique MAC (media access control) address
- 28 trillion possible addrs
- some devices randomly change their MAC addr for privacy

MAC address:

...48 bits total...

manufacturer   device ID

`ifconfig` (Interface Config)

interface

MAC address

```
trh@instance-1:~$ ifconfig
ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.128.0.36  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::4001:aff:fe80:24  prefixlen 64  scopeid 0x20<link>
        ether 42:01:0a:80:00:24  txqueuelen 1000  (Ethernet)
        RX packets 2332795  bytes 6456770667 (6.4 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1994116  bytes 718670305 (718.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 407321  bytes 417582056 (417.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 407321  bytes 417582056 (417.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

`sudo apt install net-tools`

Virtual Interfaces

```
trh@instance-1:~$ ifconfig
ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.128.0.36  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::4001:aff:fe80:24  prefixlen 64  scopeid 0x20<link>
        ether 42:01:0a:80:00:24  txqueuelen 1000  (Ethernet)
        RX packets 2332795  bytes 6456770667 (6.4 GB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1994116  bytes 718670305 (718.6 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 407321  bytes 417582056 (417.5 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 407321  bytes 417582056 (417.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

loopback (lo) device a virtual interface (not actual hardware)
connecting to a mini network containing just your computer

Google Console: Adding Interfaces (NICs)

Create Instance > Advanced Options > Networking

Network interfaces ❓

Network interface is permanent

| default default (10.128.0.0/20) | ⌄ |

| other-net subnet (10.0.0.0/24) | ⌄ |

ADD NETWORK INTERFACE

Virtual Machine Summary

| ☐ | ✅ | instance-2 | us-central1-a | | 10.128.0.37 (nic0) 10.0.0.2 (nic1) | 34.29.220.248 (nic0) 35.202.74.234 (nic1) |

Google Console: Adding Interfaces

Create Instance > Advanced Options > Networking

Network interfaces ❷

```
trh@instance-2:~$ ifconfig
ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.128.0.37  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::4001:aff:fe80:25  prefixlen 64  scopeid 0x20<link>
        ether 42:01:0a:80:00:25  txqueuelen 1000  (Ethernet)
        RX packets 637  bytes 546000 (546.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 612  bytes 97265 (97.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.0.0.2  netmask 255.255.255.255  broadcast 0.0.0.0
        inet6 fe80::4001:aff:fe00:2  prefixlen 64  scopeid 0x20<link>
        ether 42:01:0a:00:00:02  txqueuelen 1000  (Ethernet)
        RX packets 51  bytes 9955 (9.9 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 61  bytes 6834 (6.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 120  bytes 13534 (13.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 120  bytes 13534 (13.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```
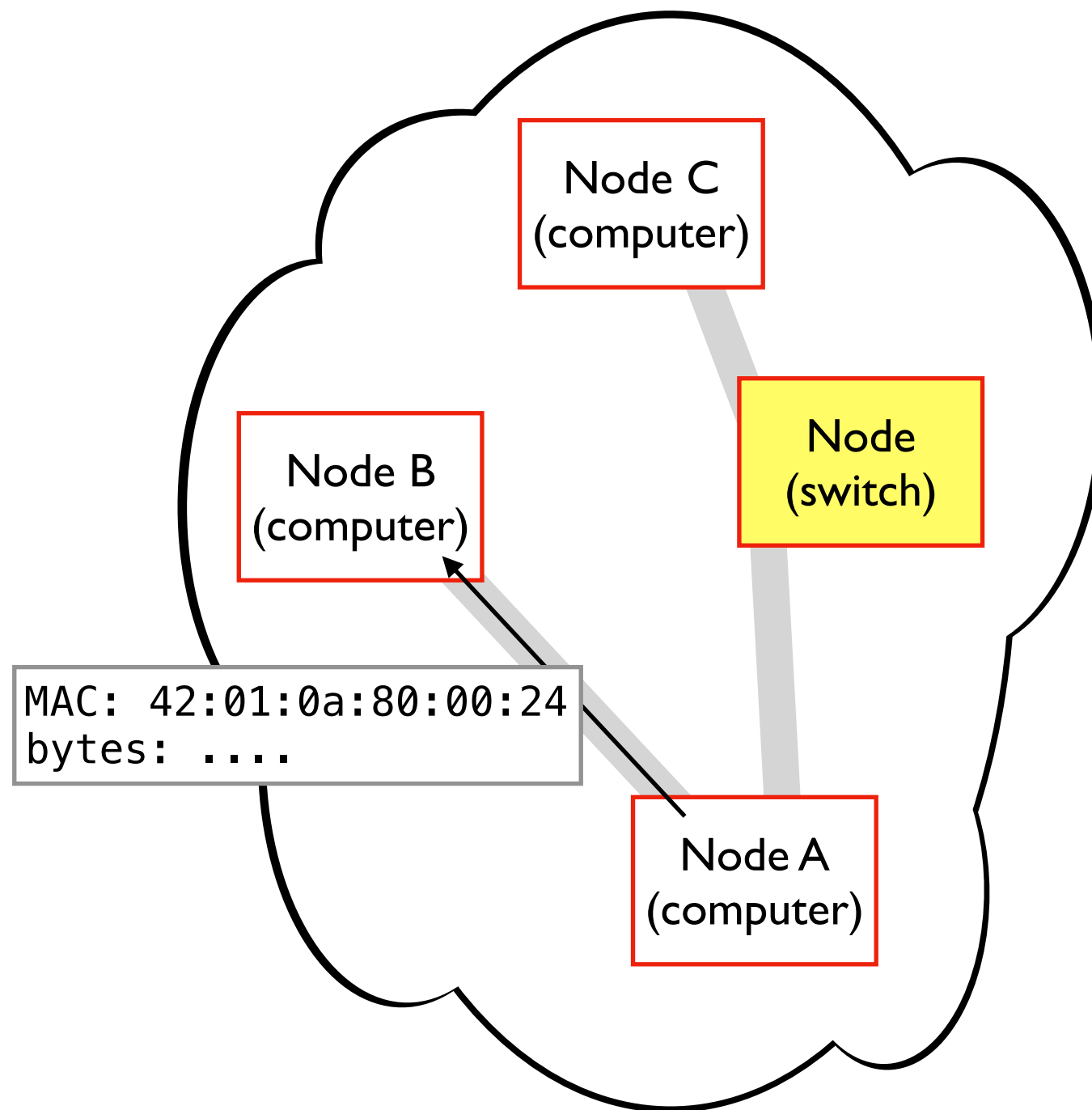
7   34.29.220.248

(nic0)

35.202.74.234

(nic1)

Networks

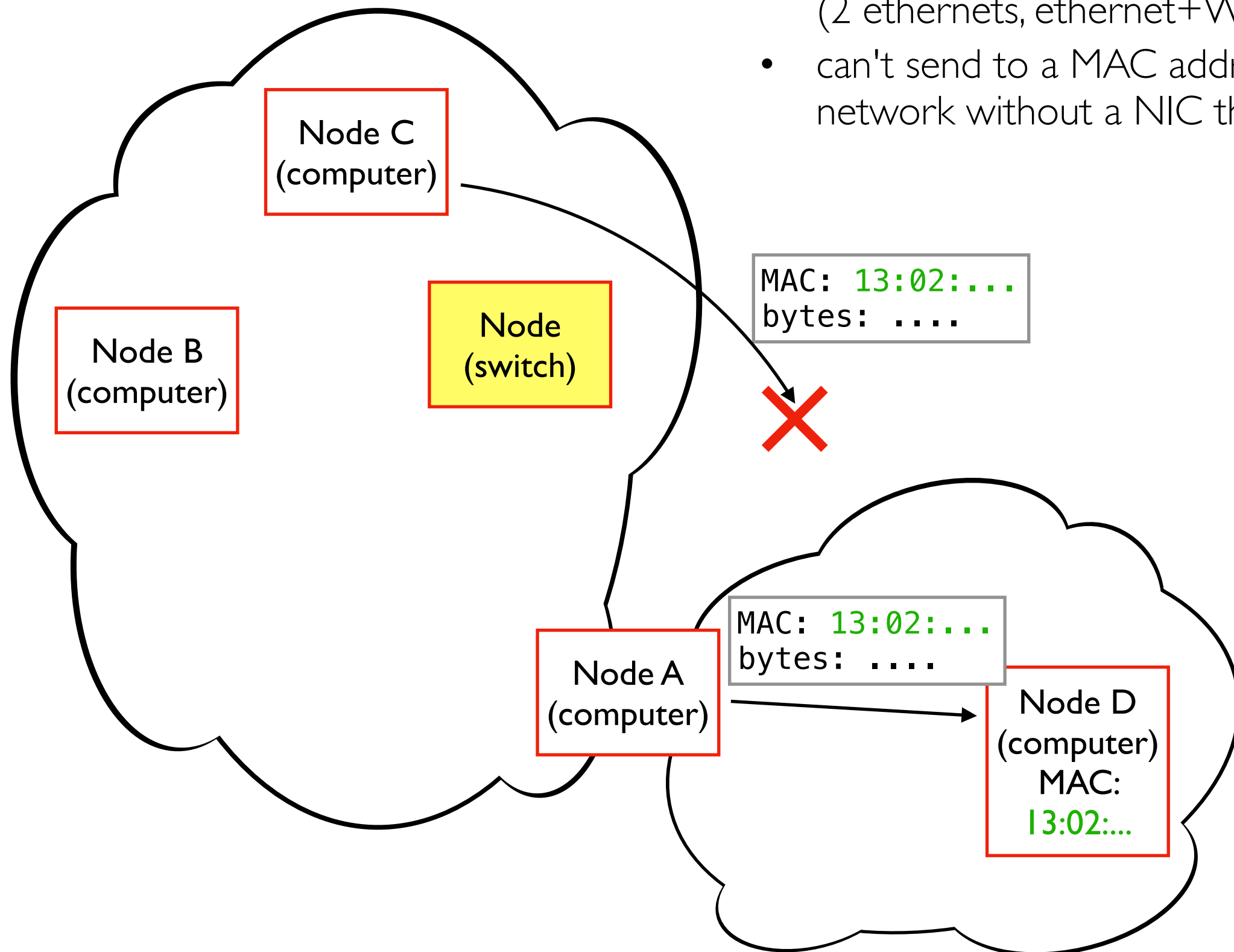A network has nodes that send bytes to other nodes **by MAC address**

- **nodes**: computer, switch, etc
- direct, or **forwarded by** switches
- whole network uses same physical tech (Wi-Fi, Ethernet, etc)

Node C
(computer)

Node
(switch)

Node B
(computer)

```
MAC: 42:01:0a:80:00:24
bytes: ....
```

Node A
(computer)

Networks

Computers can have multiple NICs
- can be on multiple networks (2 ethernets, ethernet+Wi-Fi, etc)
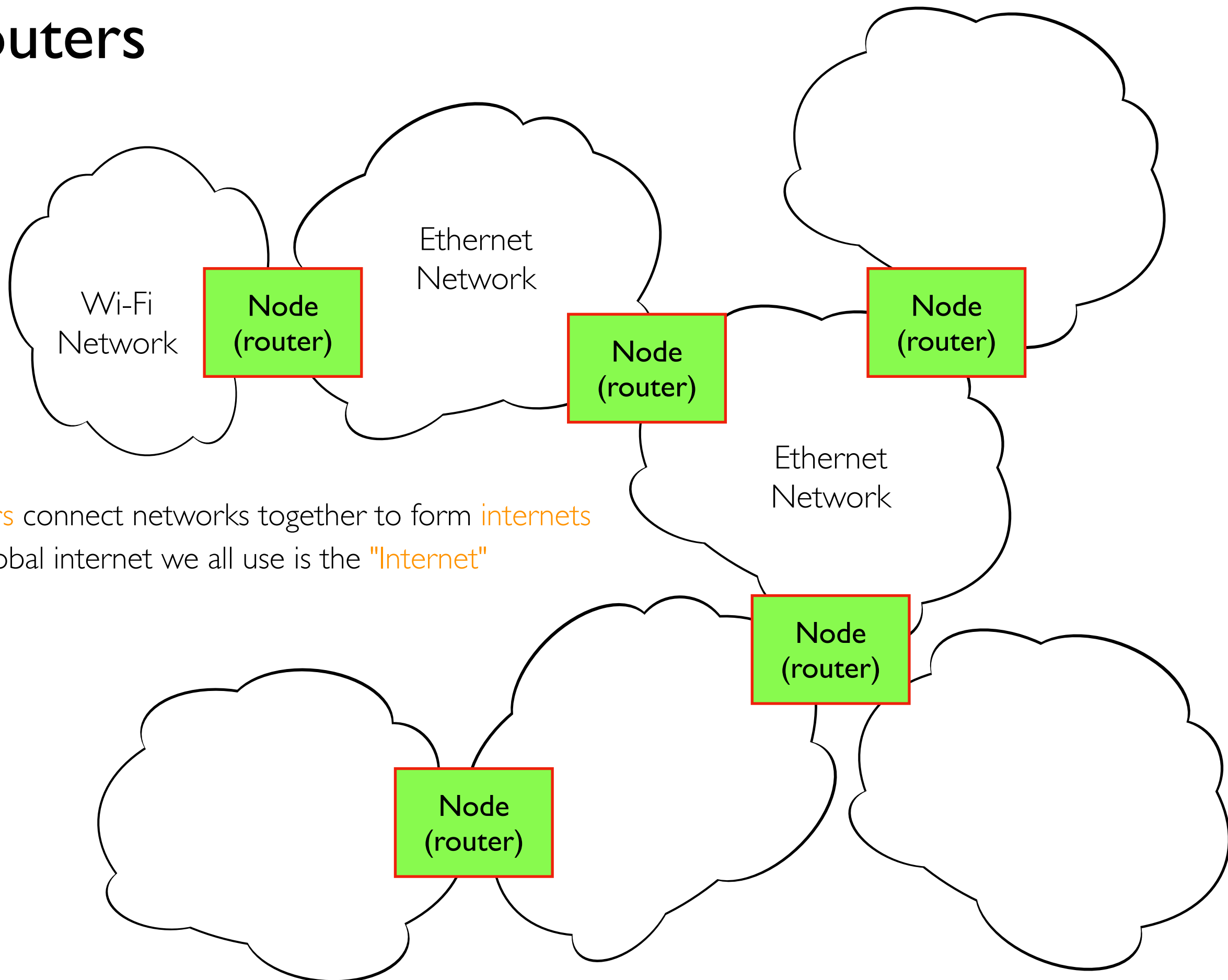- can't send to a MAC addr in another network without a NIC there

Node C (computer)

Node B (computer)

Node (switch)

```
MAC: 13:02:...
bytes: ....
```

❌

```
MAC: 13:02:...
bytes: ....
```

Node A (computer)

Node D (computer) MAC: 13:02:...

Outline

Networks

Internets and "The Internet"

Transport Protocols

Routers

Wi-Fi Network

Ethernet Network

Node (router)

Node (router)

Node (router)

Ethernet Network

Node (router)

Node (router)

- routers connect networks together to form internets
- the global internet we all use is the "Internet"

Packet Forwarding

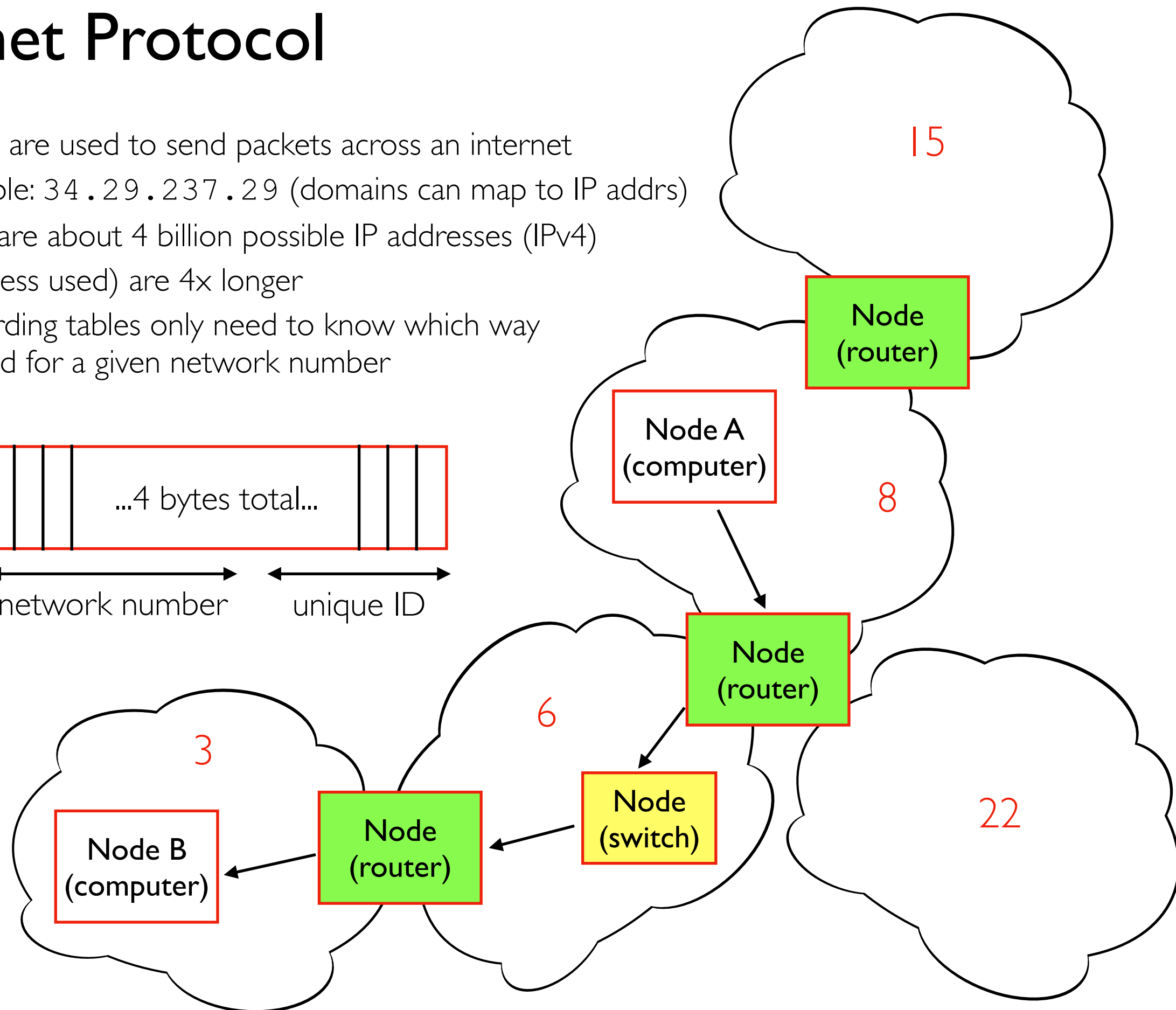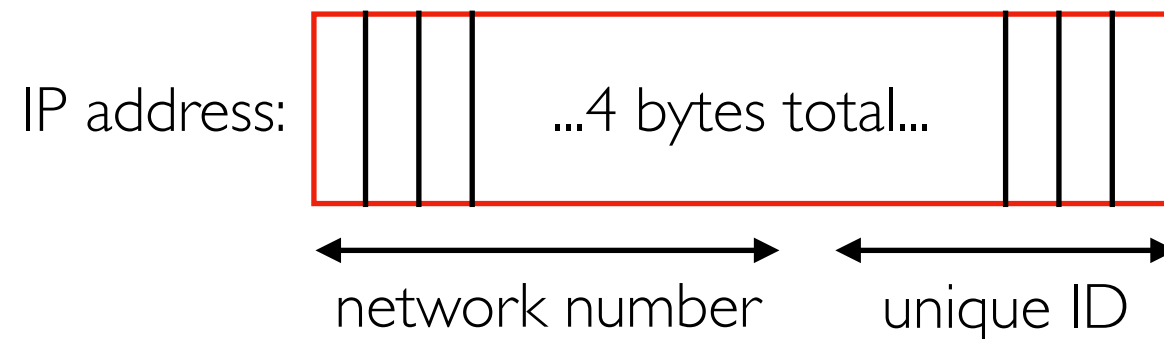Packets (some bytes with an address and other info) can be forwarded along a path from point A to point B

- routers contain forwarding tables that help them decide which direction to send along a packet
- those tables would be too big if a router had to know where every MAC address existed in the Internet

Node
(router)

Node A
(computer)

Node
(router)

Node
(switch)

Node
(router)

Node B
(computer)
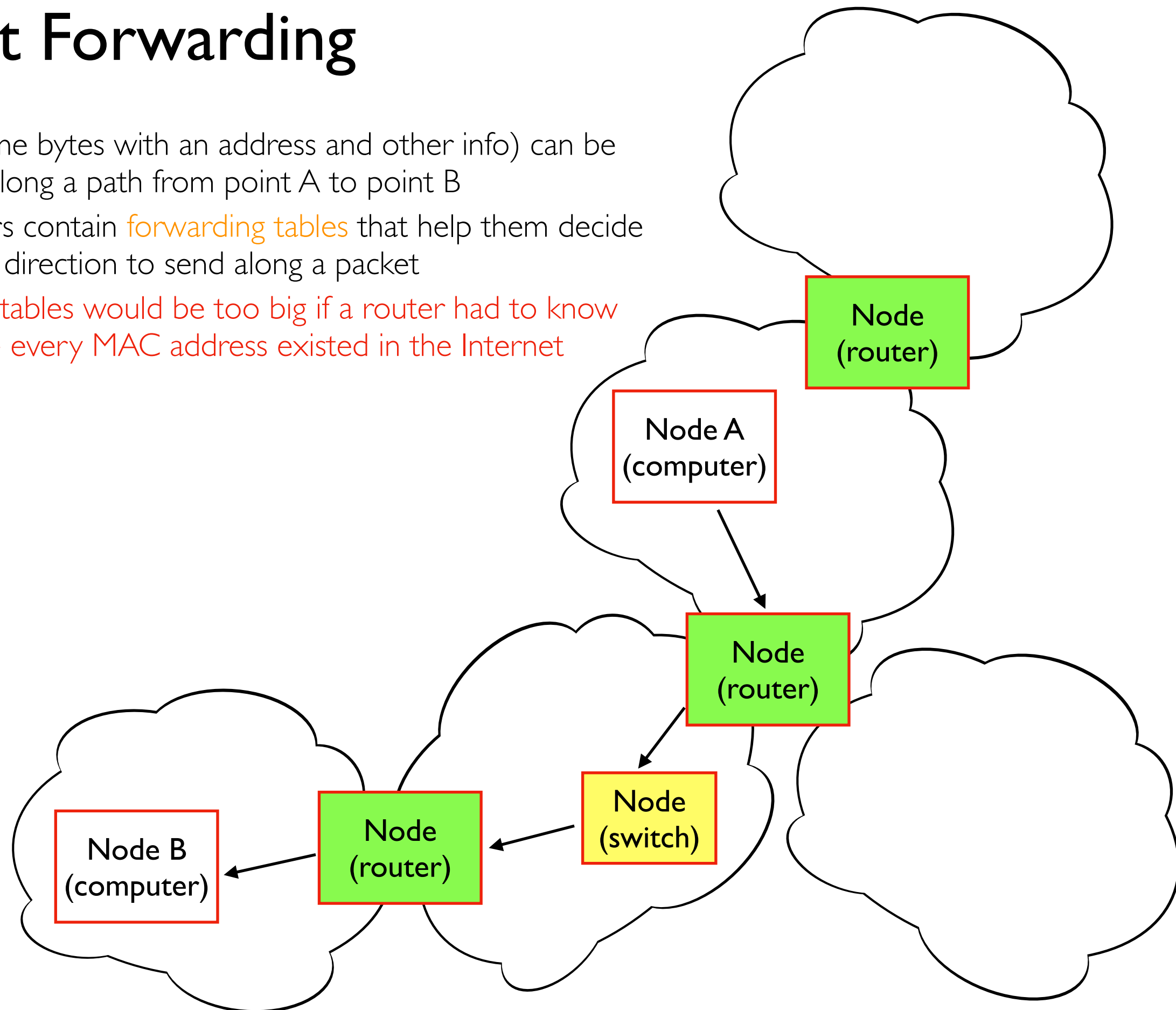
Internet Protocol

IP addresses are used to send packets across an internet

- example: 34.29.237.29 (domains can map to IP addrs)
- there are about 4 billion possible IP addresses (IPv4)
- IPv6 (less used) are 4x longer
- forwarding tables only need to know which way to send for a given network number

IP address:

...4 bytes total...

← network number →   ← unique ID →

15

Node (router)

Node A (computer)

8

Node (router)

6

3

Node B (computer)   Node (router)   Node (switch)

22

Listening on an Interface

```
trh@instance-2:~$ ifconfig
ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
      inet 10.0.1.2   netmask 255.255.255.255   broadcast 0.0.0.0
      inet6 fe80
      ether 42:0                                          <link>
      ...
```

`python3 -m http.server --bind 10.0.1.2`

```
ens5: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1430
      inet 10.0.3.2   netmask 255.255.255.255   broadcast 10.0.3.2
      inet6 fe80
      ether 42:0                                          <link>
      ...
```

`python3 -m http.server --bind 10.0.3.2`

```
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
      inet 127.0.0.1   netmask 255.0.0.0
      inet6 ::1
      loop  txqu
      ...
```

`python3 -m http.server --bind 127.0.0.1`

all of them: `python3 -m http.server --bind 0.0.0.0`
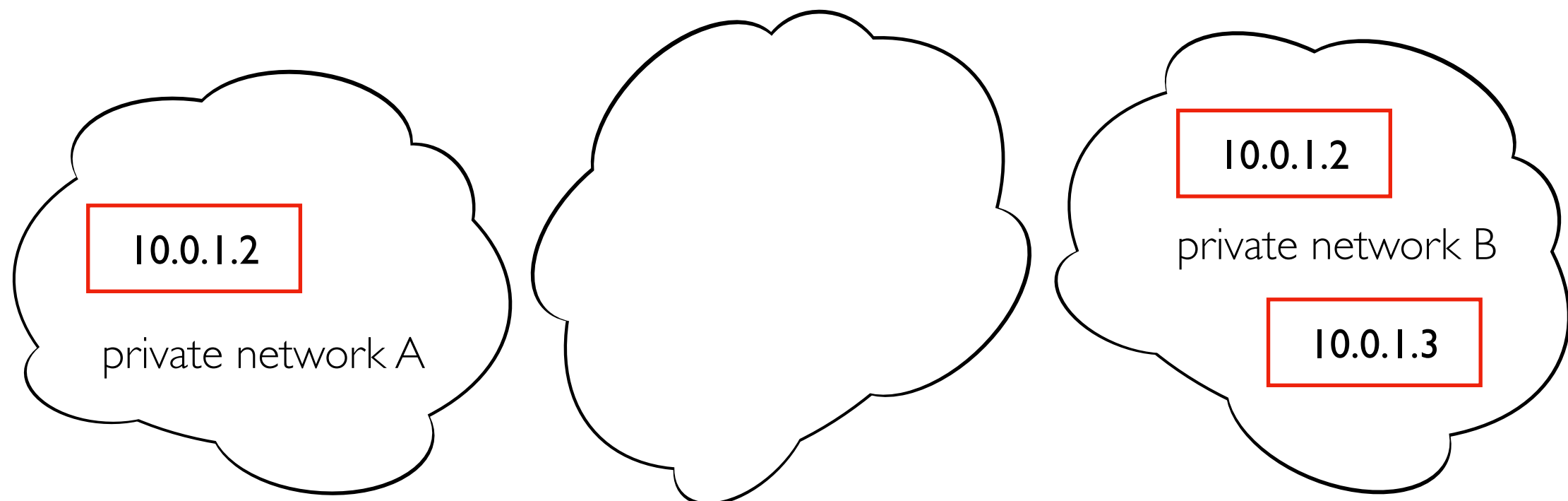
Private Networks

Challenges
- we don't have enough IPv4 addresses
- we don't want every machine to be able to receive packets from anywhere

Private ranges:
- `192.168.0.0 to 192.168.255.255`
- `172.16.0.0 to 172.31.255.255`
- `10.0.0.0 to 10.255.255.255`

these can be divided into "sub networks" (subnets) to create different networks in a bigger org

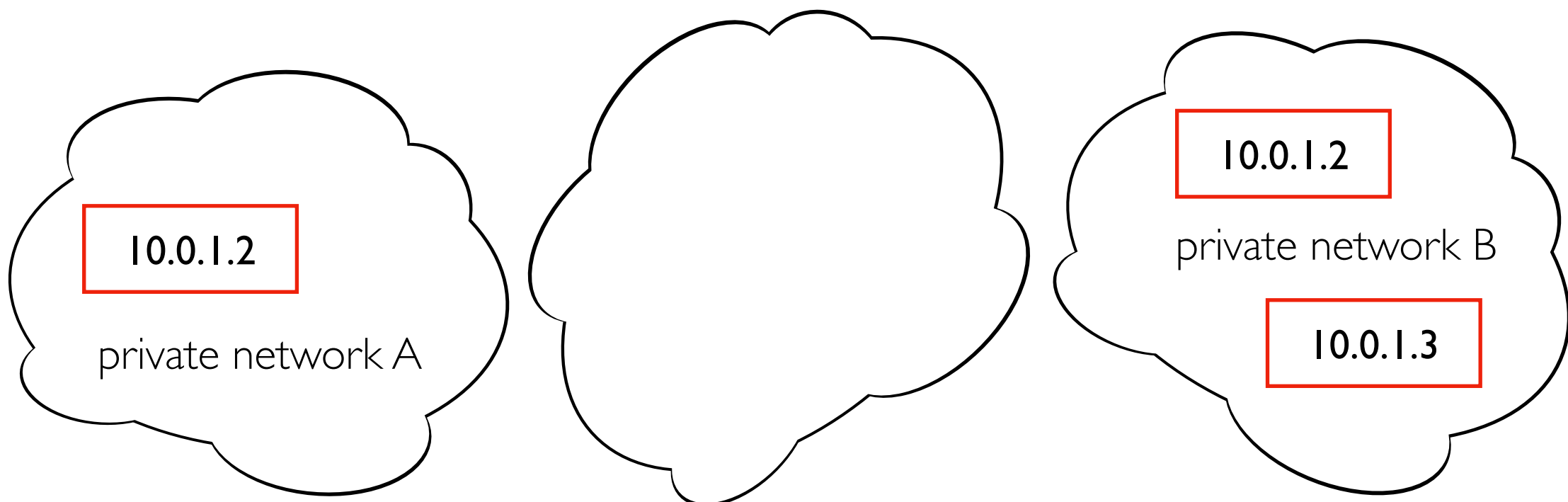Private networks allow duplicates and unreachable machines

10.0.1.2

private network A

10.0.1.2

private network B

10.0.1.3

Private Networks

```
trh@instance-2:~$ ifconfig
ens4: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1460
        inet 10.0.1.2
        inet6 fe80::40    python3 -m http.server --bind 10.0.1.2    nk>
        ether 42:01:0a:00:01:02  txqueuelen 1000  (Ethernet)
        ...
```

Private ranges:

- 192.168.0.0 to 192.168.255.255
- 172.16.0.0 to 172.31.255.255
- 10.0.0.0 to 10.255.255.255
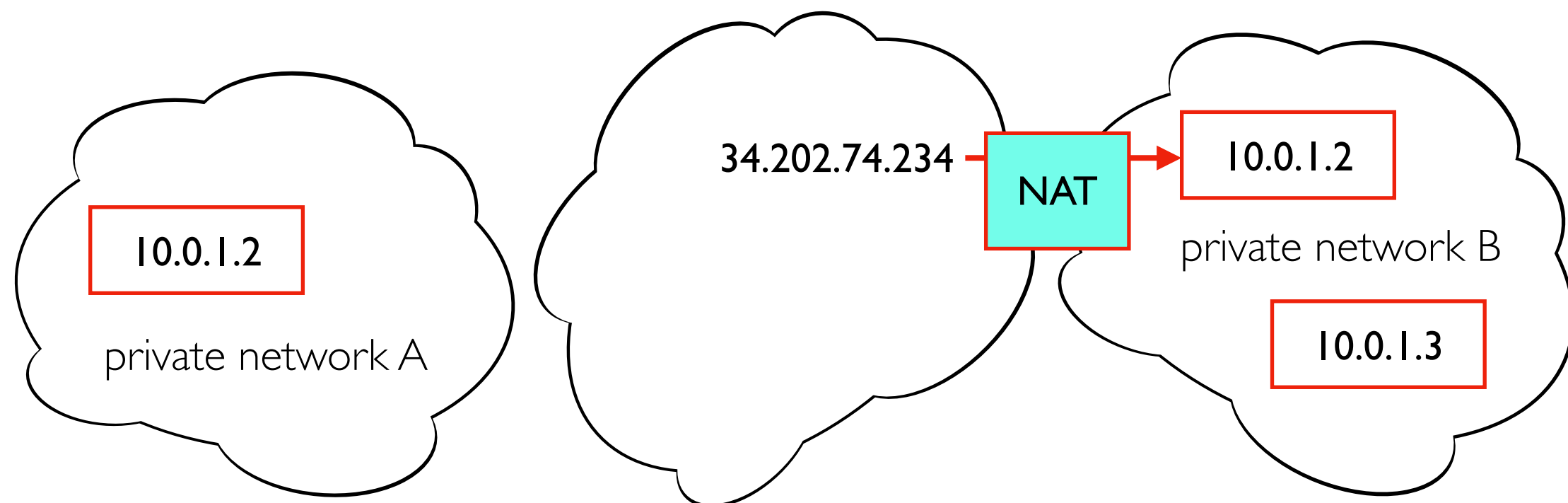
http://10.0.1.2:...
won't work in web browser!

Private networks allow duplicates and unreachable machines

Network Address Translation

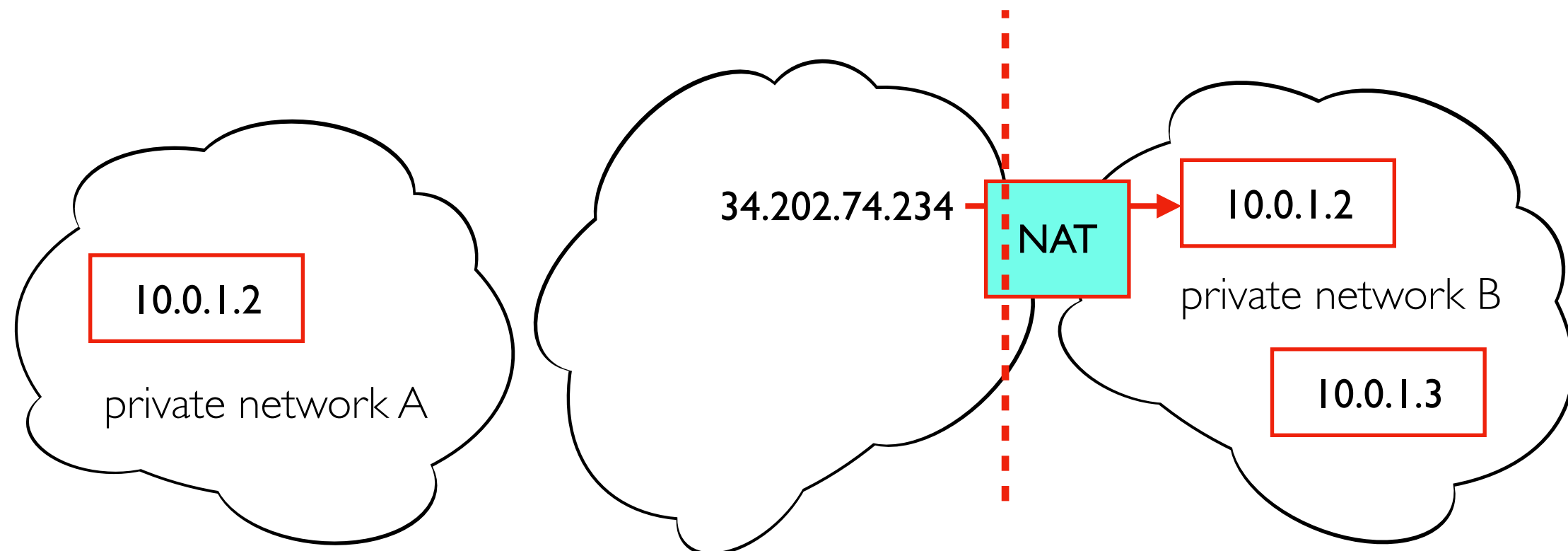|	Status	Name ↑	Internal IP	External IP
☐	✅	instance-1	10.128.0.36 (nic0)	34.29.237.29 (nic0)
☐	✅	instance-2	10.0.1.2 (nic0) 10.0.3.2 (nic1)	35.202.74.234 (nic0) 34.29.220.248 (nic1)

34.202.74.234 → NAT → 10.0.1.2

private network A : 10.0.1.2

private network B : 10.0.1.2 , 10.0.1.3

Network Address Translation

|	Status	Name ↑	Internal IP	External IP
☐	✅	instance-1	10.128.0.36 (nic0)	34.29.237.29 (nic0)
☐	✅	instance-2	10.0.1.2 (nic0) 10.0.3.2 (nic1)	35.202.74.234 (nic0) 34.29.220.248 (nic1)

a firewall can limit what
traffic is allowed

34.202.74.234 → NAT → 10.0.1.2

private network A

10.0.1.2

private network B

10.0.1.3

Network Address Translation

	Status	Name ↑	Internal IP	External IP
☐	✅	instance-1	10.128.0.36 (nic0)	34.29.237.29 (nic0)
☐	✅	instance-2	10.0.1.2 (nic0) 10.0.3.2 (nic1)	35.202.74.234 (nic0) 34.29.220.248 (nic1)

Browser

By default, the external IPs are "ephemeral" (change upon reboot). You can rent "static" IPs that don't change.

A Not Secure | 35.202.74.234:8000

Apps   DS DataStax Spark C...   544

Directory listing for /
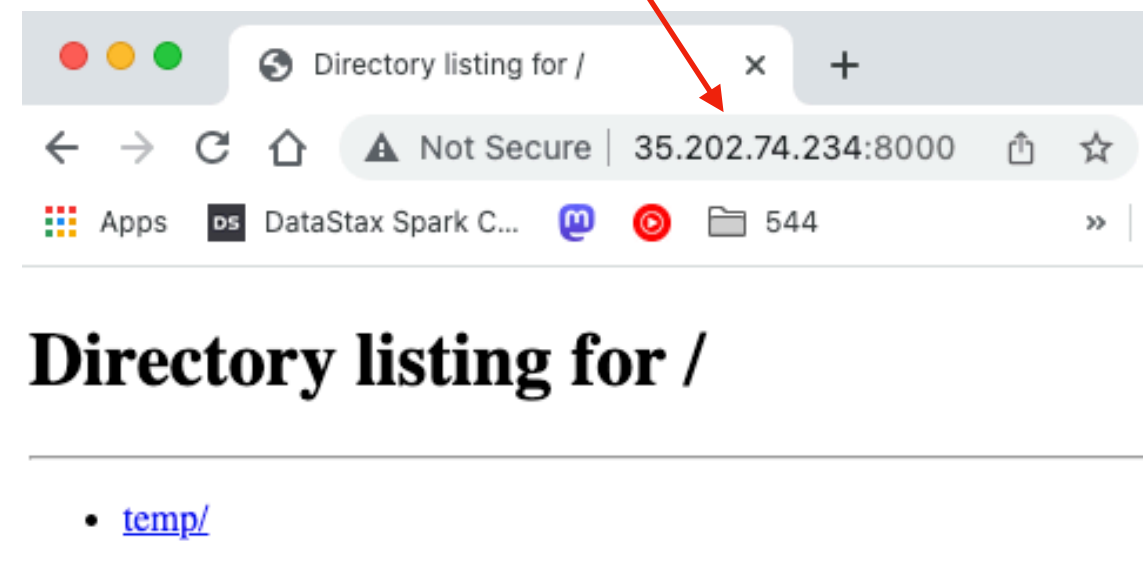
- temp/

Server

```
trh@instance-2:~/temp$ python3 -m http.server --bind 10.0.1.2
Serving HTTP on 10.0.1.2 port 8000 (http://10.0.1.2:8000/) ...
72.33.0.184 - - [10/Feb/2023 21:12:53] "GET / HTTP/1.1" 200 -
...
```

Outline

Networks

Internets and "The Internet"

Transport Protocols

Port Numbers

Computers might be running multiple processes using the network

- IP address => which NIC?

- Port number => which process?

```
trh@instance-2:~$ python3 -m http.server --directory=A --bind 10.0.1.2 8000 &
[1] 13502
Serving HTTP on 10.0.1.2 port 8000 (http://10.0.1.2:8000/) ...

trh@instance-2:~$ python3 -m http.server --directory=B --bind 10.0.1.2 9000 &
[2] 13503
Serving HTTP on 10.0.1.2 port 9000 (http://10.0.1.2:9000/) ...
```

http://35.202.74.234:8000/ ➡ first server

http://35.202.74.234:9000/ ➡ second server
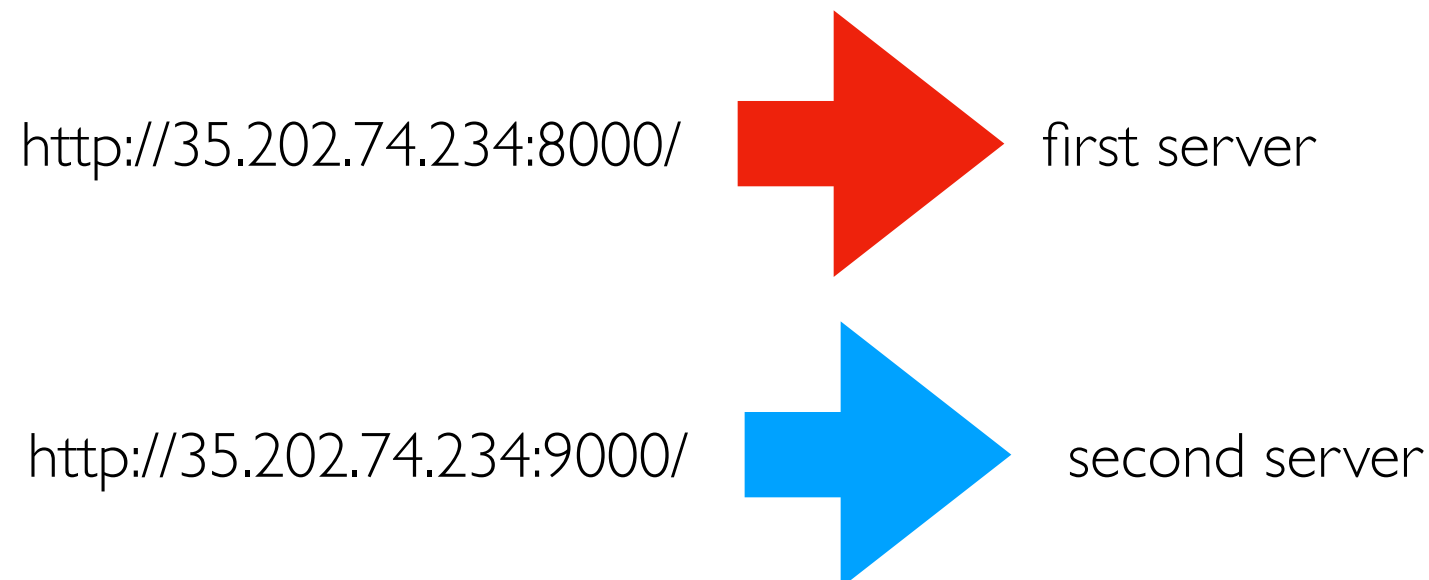
TopHat

42:01:0a:80:00:25 is an example of what?

Transport Protocols

Most common
- UDP (User Datagram Protocol)
- TCP (Transmission Control Protocol)

BOTH build on IP networking and BOTH provide port numbers

```
trh@instance-2:~/temp$ sudo lsof -i tcp -P
COMMAND      PID NODE NAME
...
sshd         863 TCP *:22 (LISTEN)
sshd         863 TCP *:22 (LISTEN)
...
python3    13607 TCP instance-2...internal:8000 (LISTEN)
python3    13608 TCP instance-2...internal:9000 (LISTEN)
```
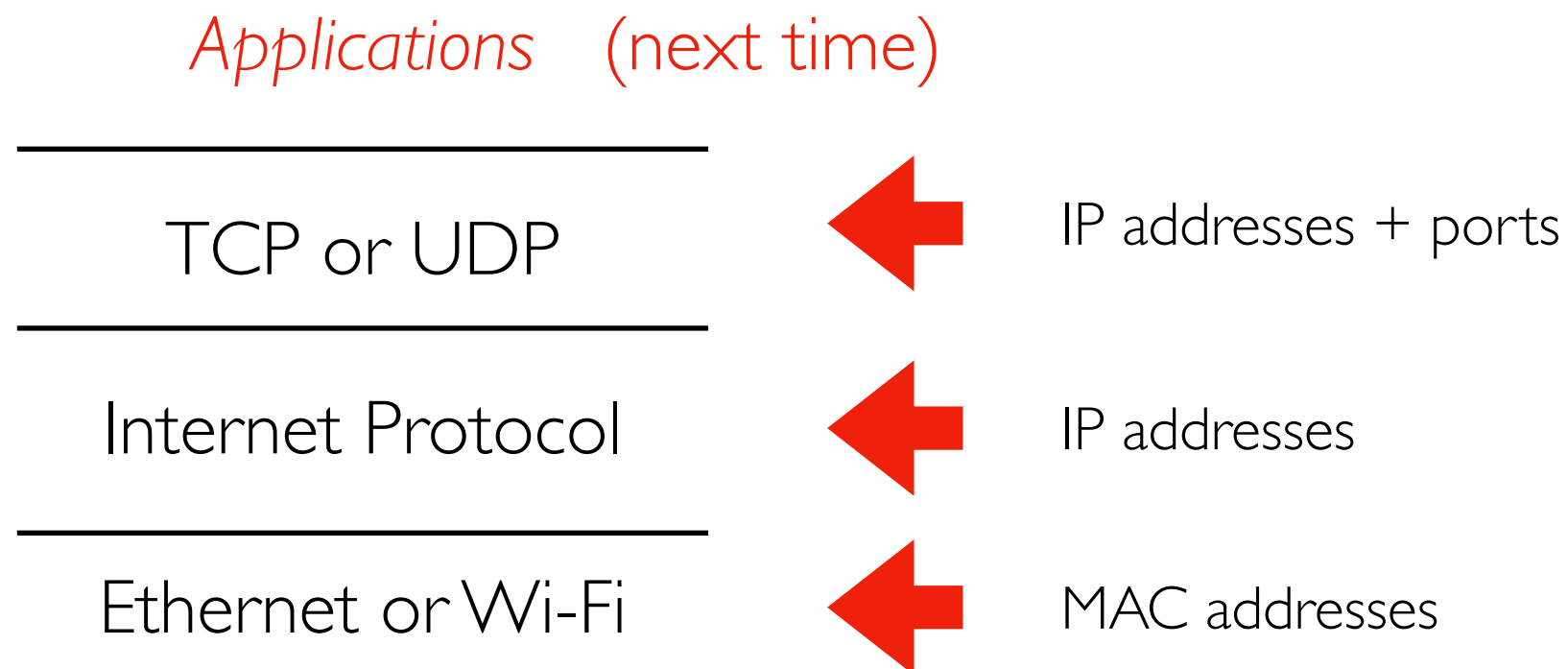
Reliability: UDP vs. TCP

Packets may be
- dropped
- reordered
- split

TCP saves+reassembles packets in order to provide original message (when possible). For packet drops, it retries. We'll mostly use TCP.

UDP doesn't do this extra work. Why ever use UDP?

Network Stack: Common Implementations

Applications   (next time)

_____

TCP or UDP ⬅ IP addresses + ports

_____

Internet Protocol ⬅ IP addresses

_____

Ethernet or Wi-Fi ⬅ MAC addresses

Network applications (like most complex systems) are not built as one single system. Layers are built upon other layers to provide additional functionality.