CSCI1680
Network Layer: IP & Forwarding

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti

Administivia

- Snowcast: now due Thursday (9/26)
  - Updated tester on Gradescope, look for another local tester update

Administivia

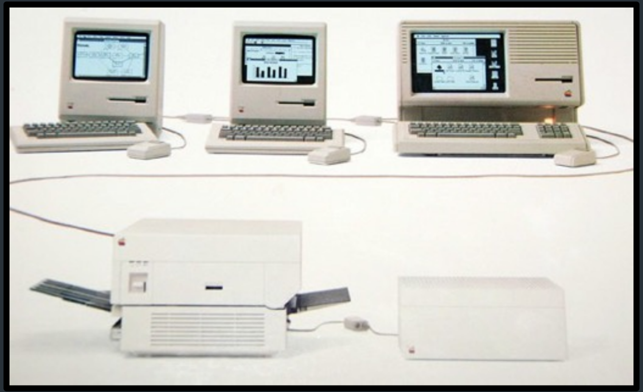
- Snowcast: now due Thursday (9/26)
  - Updated tester on Gradescope, look for another local tester update
- IP project: out Thursday
  - Team form: out TODAY, due THURSDAY 9/26
  - Gearup: THURSDAY 9/26 5-7pm CIT 368
- **HW1: out later today, due next Thurs**
  - Some practice for IP!

Today

Start of network layer

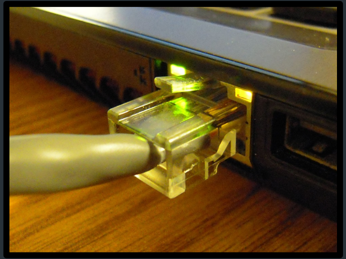
- Network layer: Internet Protocol (IP) (v4)
- Mechanics of IP forwarding
- Intro to IP project

Recap: the link layer



Goal: How to connect hosts on a "small" network

- Hosts connect to network via <u>interfaces</u>
- Every interface has a link-layer address
  - Ethernet/Wifi: MAC address (`0c:45:22:c1:be:03`)



Recap: the link layer

Goal: How to connect hosts on a "small" network

- Hosts connect to network via <u>interfaces</u>
- Every interface has a link-layer address
  - Ethernet/Wifi: MAC address (`0c:45:22:c1:be:03`)

<u>Mental model for the link layer</u>

- Every host connected to every other host (at least logically)
- Given a link-layer address, know how to reach host on your network

=> Devices: Switches, Wifi APs: forward packets between nodes on same network

Switch: network device that forwards frames (packets) between *ports*

Example: Ethernet switch MAC table

```
R6#sh mac-address-table
EHWIC: 0
Destination Address Address Type VLAN Destination Port
------------------- ------------ ---- ------------------
5c45.27e0.8383            Dynamic         1 GigabitEthernet0/1/3
7641.7b63.584a            Dynamic        20 GigabitEthernet0/1/3
5c45.27e0.8381            Dynamic        10 GigabitEthernet0/1/3
0000.5e00.0101            Dynamic        10 GigabitEthernet0/0/1
ca3f.aee3.e3e6            Dynamic        20 GigabitEthernet0/1/3
644b.f012.7f75            Dynamic        20 GigabitEthernet0/1/3
f018.9815.8eb8            Dynamic        20 GigabitEthernet0/1/3
ecb5.fa13.4677            Dynamic        20 GigabitEthernet0/0/2
a0a4.c5c2.4165            Dynamic        20 GigabitEthernet0/0/1
4c71.0c92.4f10            Dynamic        10 GigabitEthernet0/1/3
12d3.acae.bbc0            Dynamic        20 GigabitEthernet0/0/1
04d4.c448.9cf7            Dynamic        20 GigabitEthernet0/1/3
```

Example: Ethernet switch MAC table

```
R6#sh mac-address-table
EHWIC: 0
Destination Address  Address Type  VLAN  Destination Port
-------------------  ------------  ----  -----------------
5c45.27e0.8383          Dynamic         1 GigabitEthernet0/1/3
7641.7b63.584a          Dynamic        20 GigabitEthernet0/1/3
5c45.27e0.8381          Dynamic        10 GigabitEthernet0/1/3
0000.5e00.0101          Dynamic        10 GigabitEthernet0/0/1
ca3f.aee3.e3e6          Dynamic        20 GigabitEthernet0/1/3
644b.f012.7f75          Dynamic        20 GigabitEthernet0/1/3
f018.9815.8eb8          Dynamic        20 GigabitEthernet0/1/3
ecb5.fa13.4677          Dynamic        20 GigabitEthernet0/0/2
a0a4.c5c2.4165          Dynamic        20 GigabitEthernet0/0/1
4c71.0c92.4f10          Dynamic        10 GigabitEthernet0/1/3
12d3.acae.bbc0          Dynamic        20 GigabitEthernet0/0/1
04d4.c448.9cf7          Dynamic        20 GigabitEthernet0/1/3
```

Does it scale? 🤔

Why doesn't it scale?

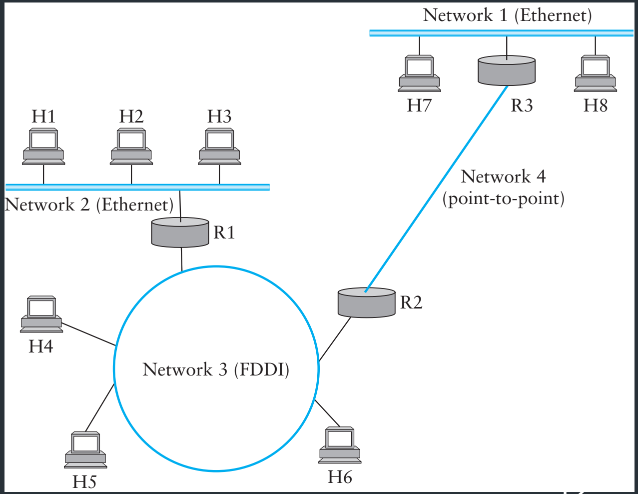
Enter: IP

Application	Service: user-facing application. Application-defined messages
Transport	Service: multiplexing applications Reliable byte stream to other node (TCP), Unreliable datagram (UDP)
Network	Service: move packets to any other node in the network Internet Protocol (IP)
Link	Service: move frames to other node across link. May add reliability, medium access control
Physical	Service: move bits to other node across link

Internet Protocol (IP) Goals

How to connect *everyone*?

- Glue lower-level networks together
- A network of networks!
- Router: device that forwards packets between *networks*

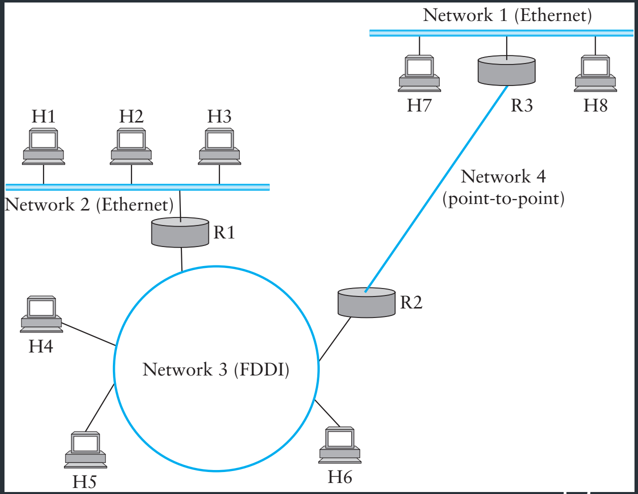


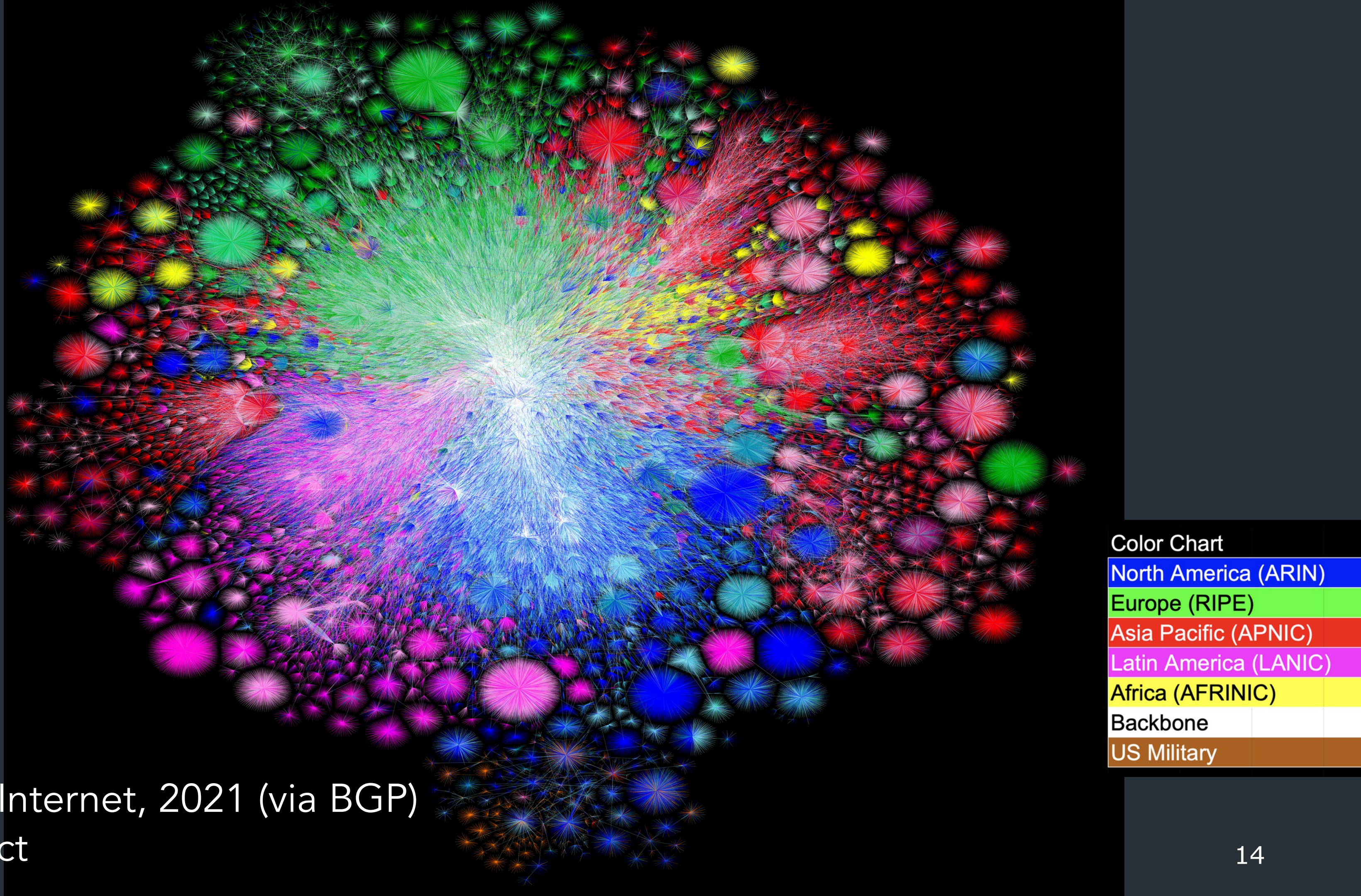
Internet Protocol (IP) Goals

How to connect *everyone*?

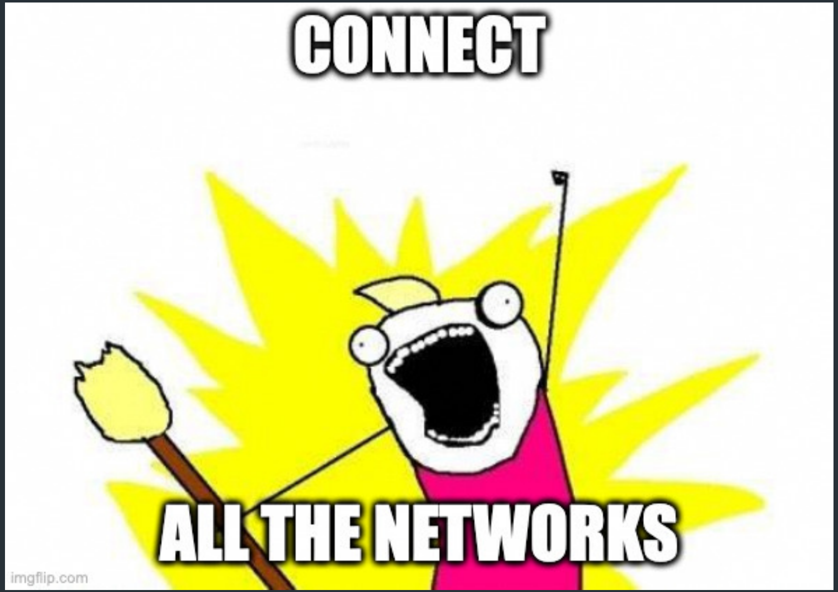
- Glue lower-level networks together
- A network of networks!
- Router: device that forwards packets between *networks*

=> Doesn't that sound like switching?



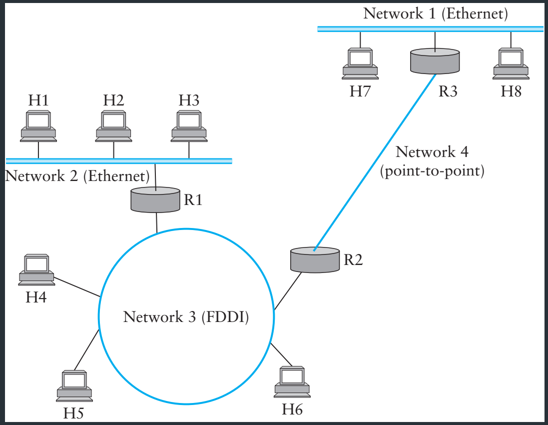


Map of the Internet, 2021 (via BGP)
OPTE project


CONNECT
ALL THE NETWORKS
imgflip.com

New Challenges

- Networks are heterogeneous (eg. Wifi vs. Ethernet)
  - Different frame formats
  - Some are more reliable than others
  - Different packet sizes/bandwidths
- Scaling: link-layer strategies don't work!



What came before the Internet?

The (landline) telephone network

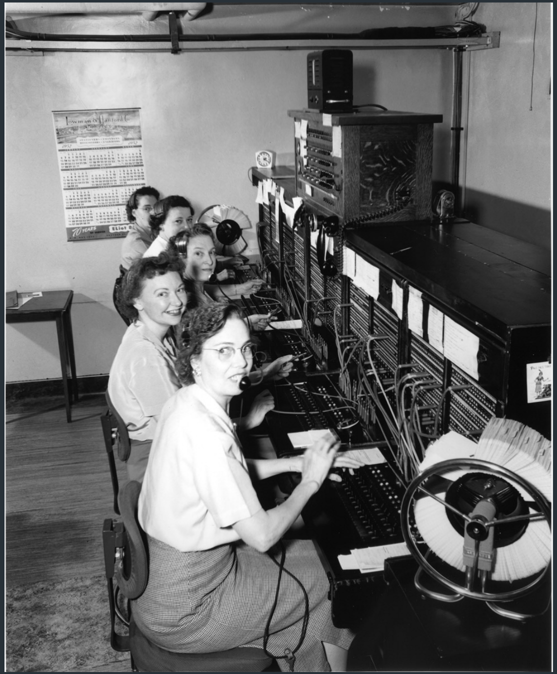
(Plain Old Telephone Service (POTS)

A Host

The (landline) telephone network

(Plain Old Telephone Service (POTS)

A Host



A large telephone exchange, 1943

Example: long distance telephone call

Dramatization from an episode of radio program "Dragnet", 1949

Early telephone systems

- Circuit switching: set up whole path for call beforehand

Does it scale?

Early Internet goals

Early Internet goals

⇒ Build system that can connect different *networks*
=> Operate over long distances
=> Managed by different entities

Early Internet goals

⇒ Build system that can connect different *networks*
=> Operate over long distances
=> Managed by different entities

Need: *devices* and *protocols* to make this work

Design questions

- How to deal with heterogeneous networks?
- How to find hosts?
- Should messages be reliable or unreliable?
- What to do when a device joins/leaves?
- …

A Bit of History

Early Packet switched networks: Arpanet's IMPs

- Late 1960's => RFC 1, 1969!
- Reliable network with many features we know today

A Bit of History

Early Packet switched networks: Arpanet's IMPs

- Late 1960's => RFC 1, 1969!
- Reliable network with many features we know today

Initial version: Network Control Program (NCP)

- Assumed IMPs were reliable

A Bit of History

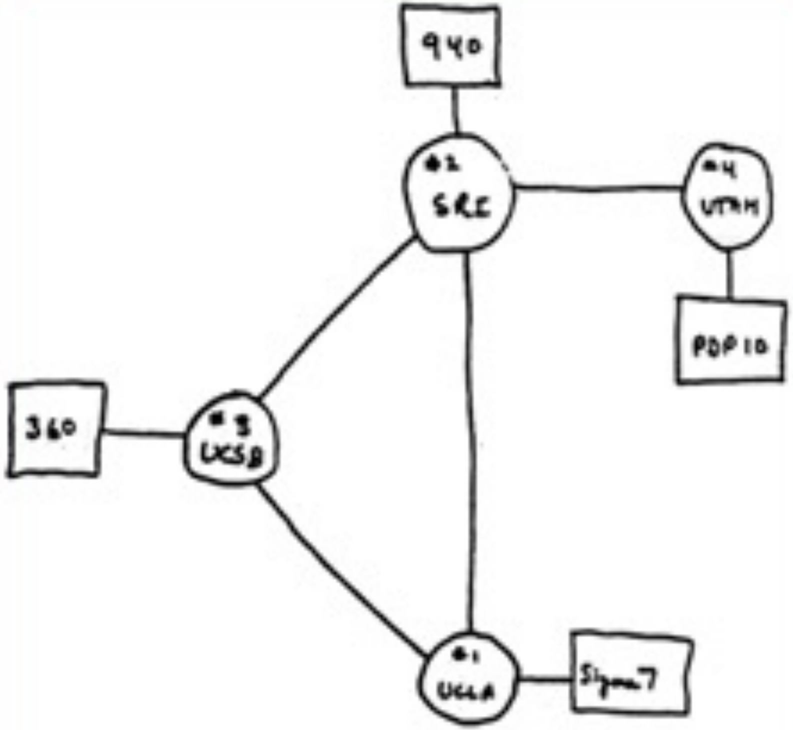
Early Packet switched networks: Arpanet's IMPs

- Late 1960's => RFC 1, 1969!
- Reliable network with many features we know today

Initial version: Network Control Program (NCP)

- Assumed IMPs were reliable

What about when network isn't reliable?



THE ARPA NETWORK
DEC 1969

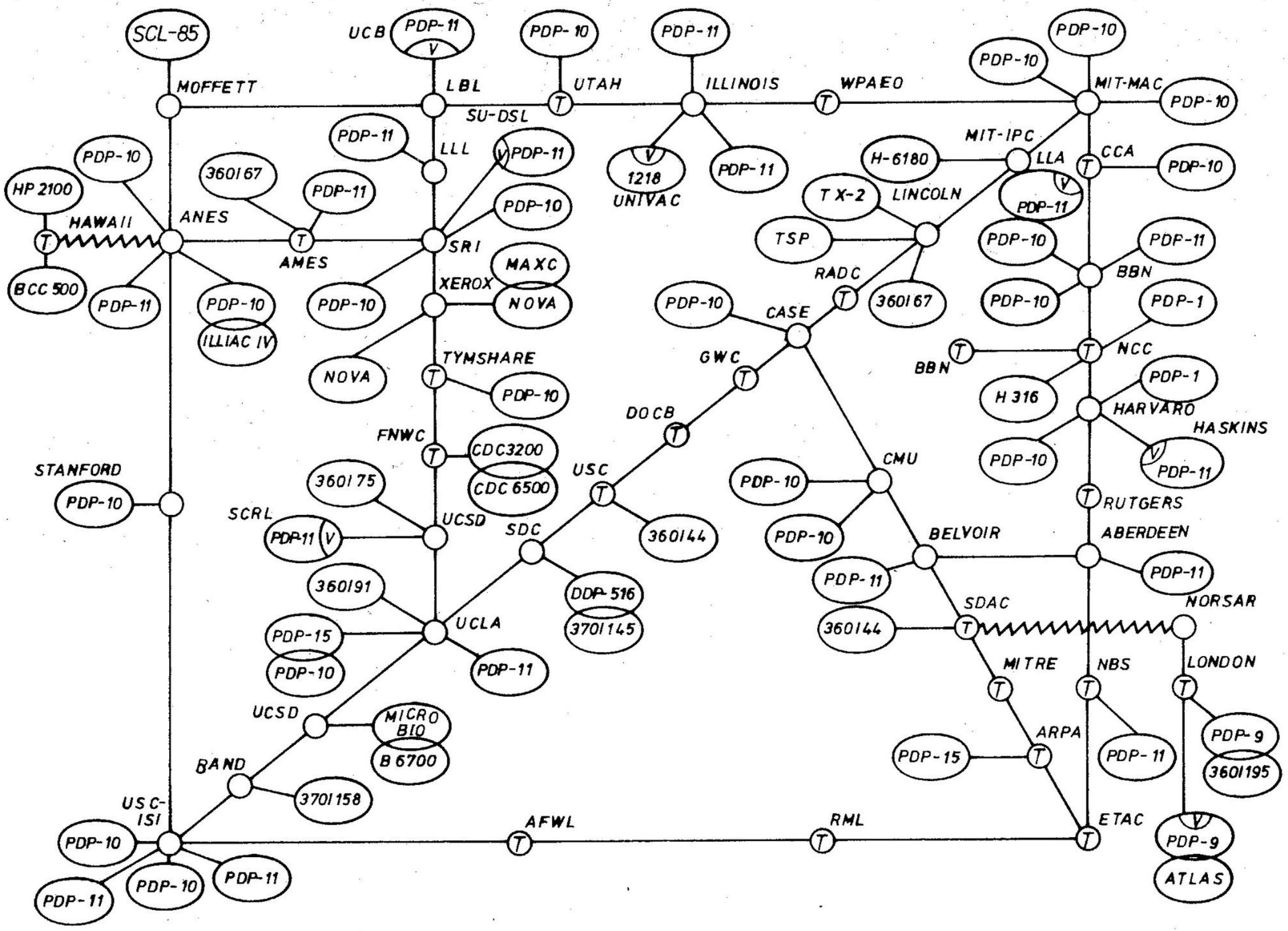


Abb. 4 ARPA NETwork, topologische Karte. Stand Juni 1974.

How to make such a protocol?

- How to deal with heterogeneous networks?
- How to find hosts?
- Should messages be reliable or unreliable?
- What to do when a device joins/leaves?
- …

Big concerns
$\Rightarrow$ Not every application needs all features
$\Rightarrow$ Can't assume much functionality from (heterogeneous link layer)

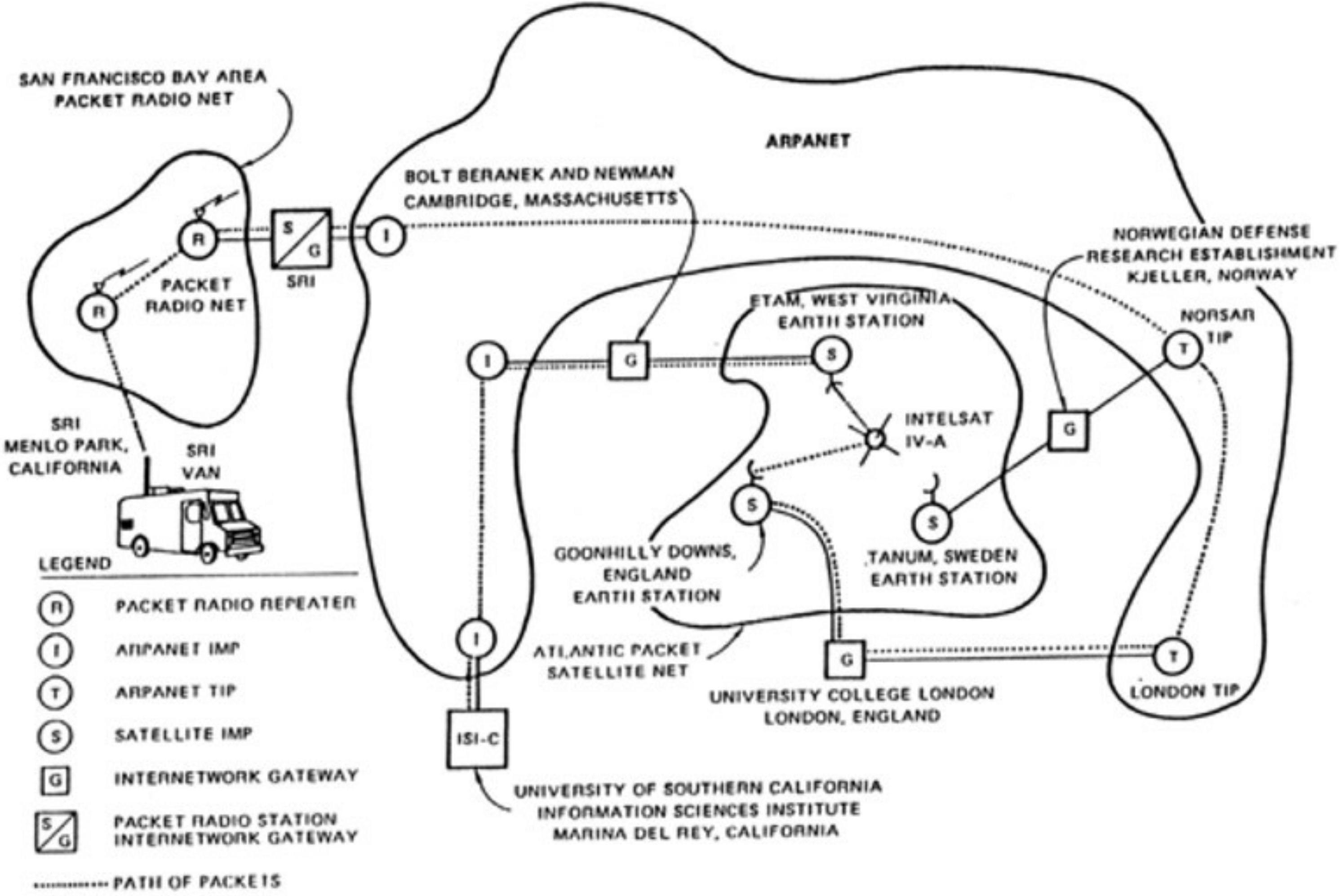
1974: TCP/IP Introduced

- Vint Cerf, Robert Kahn build protocol to replace NCP
- Initial design: single protocol providing a reliable pipe

1974: TCP/IP Introduced

- Vint Cerf, Robert Kahn build protocol to replace NCP
- Initial design: single protocol providing a reliable pipe

Eventually, separated into different protocols we know today


SAN FRANCISCO BAY AREA
PACKET RADIO NET
ARPANET
BOLT BERANEK AND NEWMAN
CAMBRIDGE, MASSACHUSETTS
SRI
PACKET
RADIO NET
NORWEGIAN DEFENSE
RESEARCH ESTABLISHMENT
KJELLER, NORWAY
ETAM, WEST VIRGINIA
EARTH STATION
NORSAR
TIP
INTELSAT
IV-A
SRI
MENLO PARK,
CALIFORNIA
SRI
VAN
GOONHILLY DOWNS,
ENGLAND
EARTH STATION
TANUM, SWEDEN
EARTH STATION
ATLANTIC PACKET
SATELLITE NET
UNIVERSITY COLLEGE LONDON
LONDON, ENGLAND
LONDON TIP
ISI-C
UNIVERSITY OF SOUTHERN CALIFORNIA
INFORMATION SCIENCES INSTITUTE
MARINA DEL REY, CALIFORNIA
LEGEND
PACKET RADIO REPEATER
ARPANET IMP
ARPANET TIP
SATELLITE IMP
INTERNETWORK GATEWAY
PACKET RADIO STATION
INTERNETWORK GATEWAY
PATH OF PACKETS

IP's Decisions

- Connectionless, packet-switched network
- "Best-effort" service

How to reach hosts?

IP's Decisions

- Connectionless, packet-switched network

  => Routers are "simple" => no connection state
- "Best-effort" service: other layers add reliability if you need it

  => Packets might be dropped, reordered, delayed, …

How to reach hosts?

- Common message format: IP header
- Every host identified by an IP address

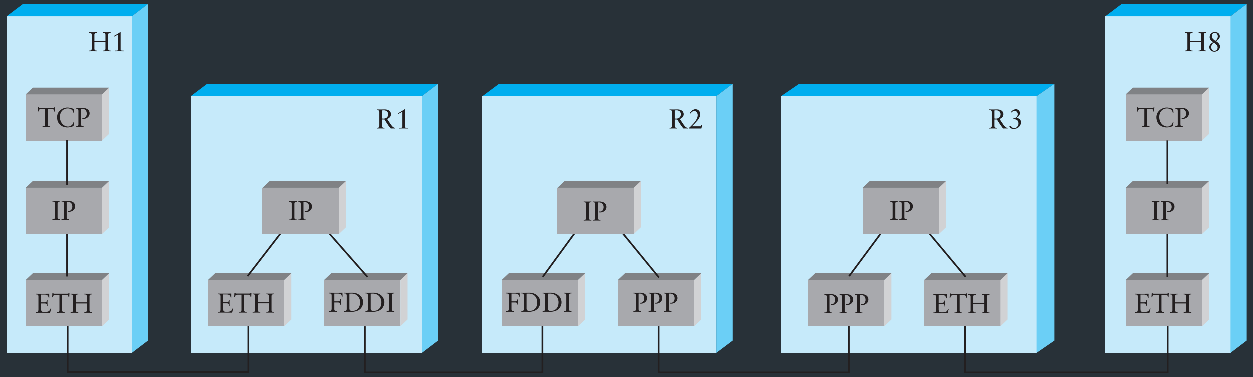
An excellent read

David D. Clark, "The design Philosophy of the DARPA Internet Protocols", 1988

Primary goal: multiplexed utilization of existing interconnected networks

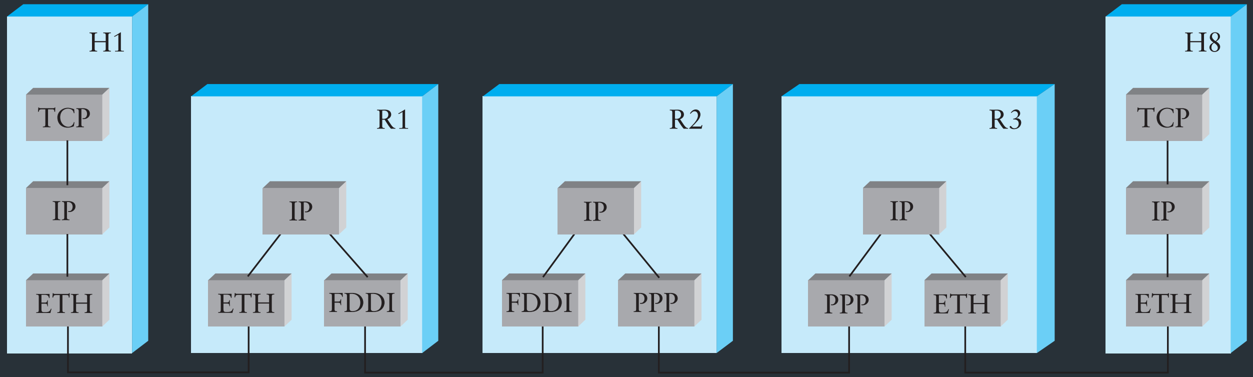
- Other goals:
  - Communication continues despite loss of networks or gateways
  - Support a variety of communication services
  - Accommodate a variety of networks
  - Permit distributed management of its resources
  - Be cost effective
  - Low effort for host attachment
  - Resources must be accountable

The Internet Protocol (IP): Runs on all hosts and routers



The Internet Protocol (IP): Runs on all hosts and routers

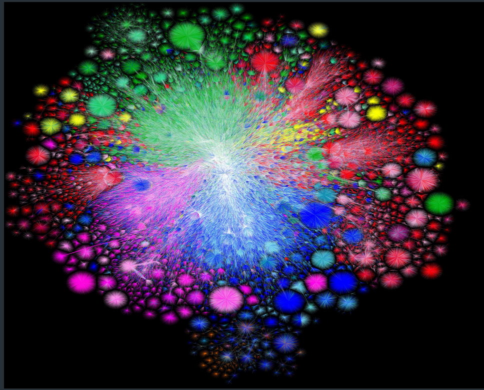
- *Addressing*: how we name nodes in an IP network
- Provides *forwarding*: how routers move packets based on the destination address
- *(later) Routing*: how routers build forwarding rules



IP Addressing

What’s an IP address

- Unique number to identify “all” hosts on the Internet
- A number with structure => the number tells the network where the host is



Example: phone numbers

+ 1 401 863 1000

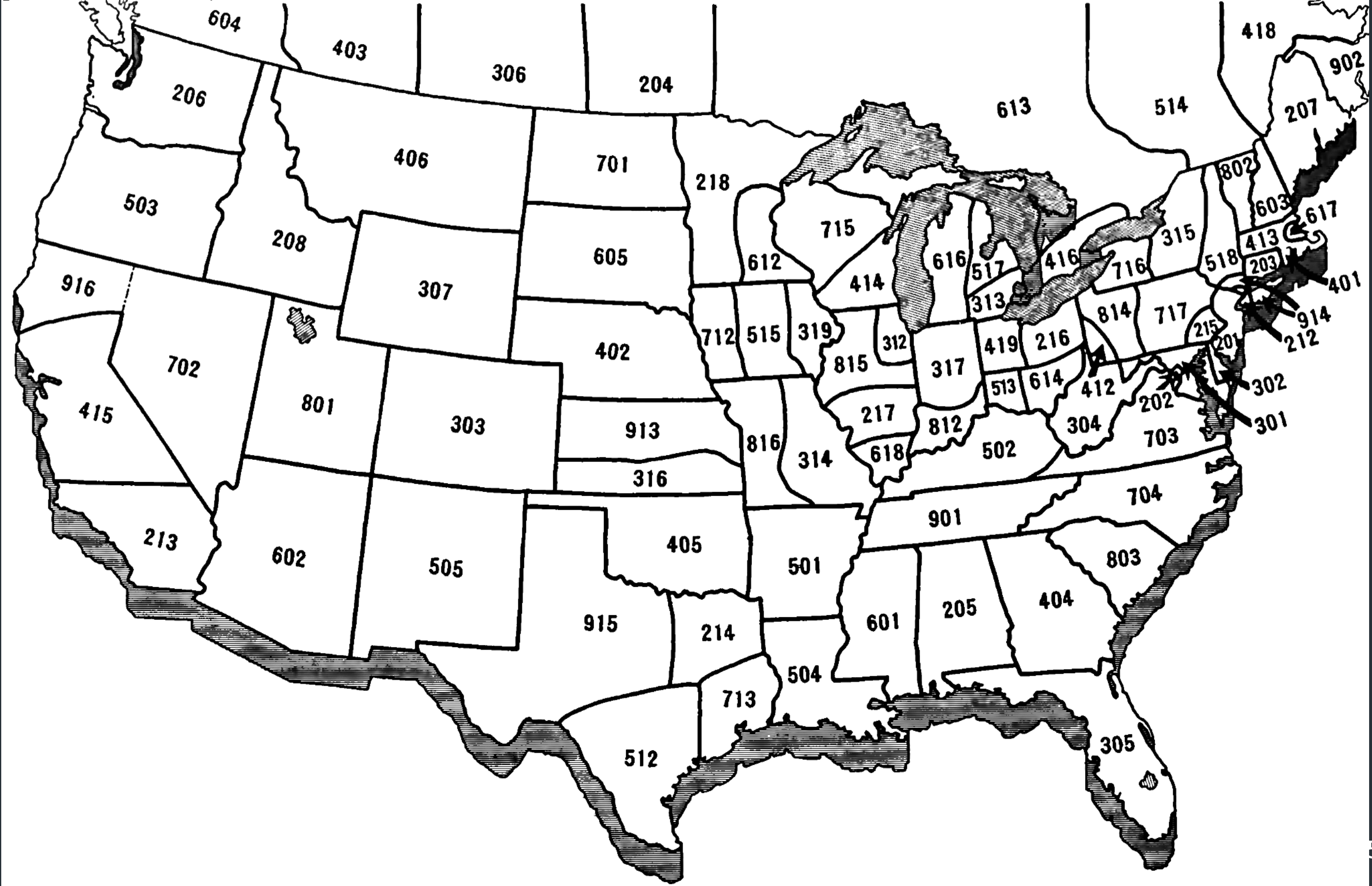
Analogy: back to phones

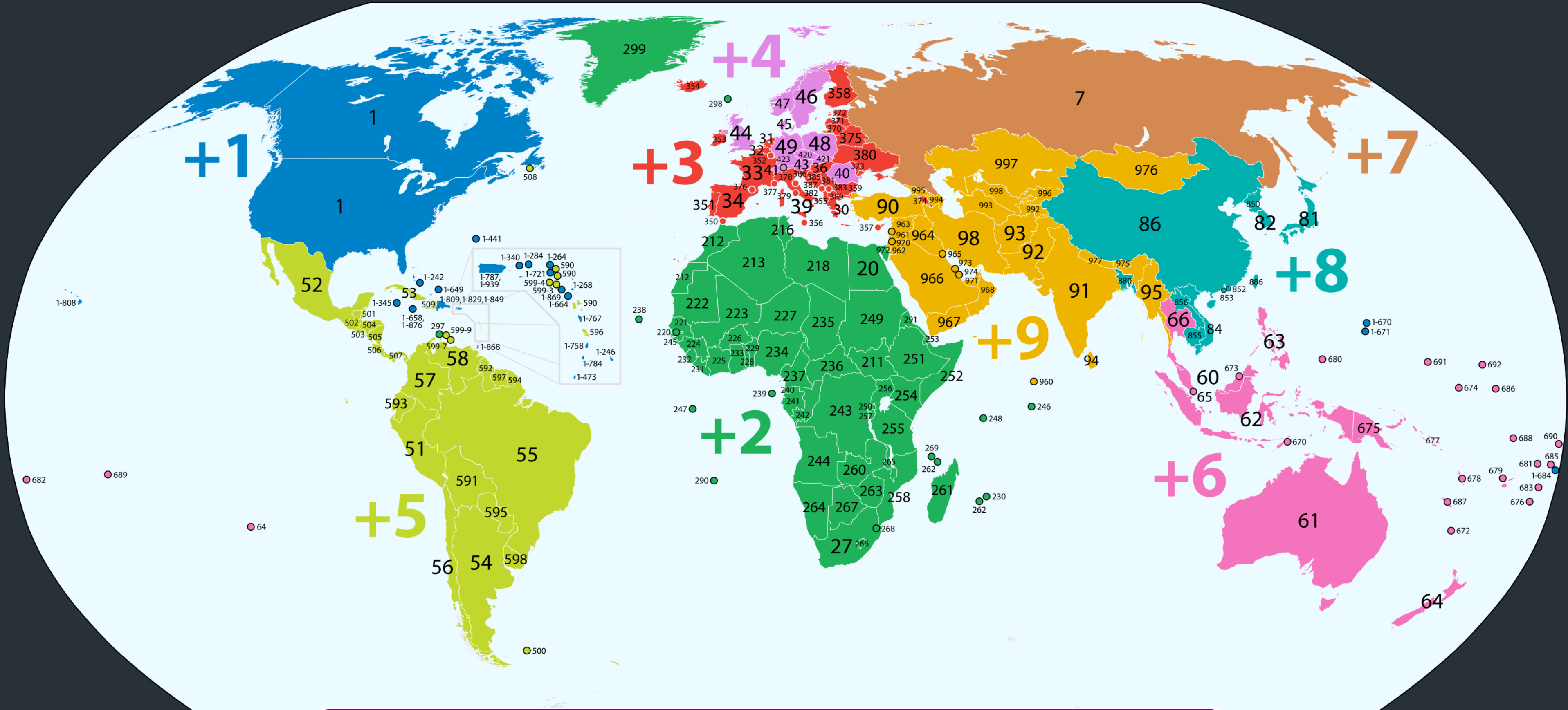
Telephone numbers have a structure to them

+ 1 401 863 1000

+1 212 555 4253

Part of the number tells where you are!
(or at least it did before cell phones)


604
403
306
204
418
902
206
613
514
207
406
701
218
802
503
603
617
208
605
715
315
413
612
616
517
416
716
518
203
401
916
307
414
313
814
717
914
402
712
515
319
312
419
216
215
212
702
815
317
201
513
614
412
302
415
801
217
202
304
303
913
812
301
816
314
703
618
502
316
704
901
213
405
602
505
501
803
404
205
601
915
214
504
713
305
512



… and not all numbers are the same length!

IP Addressing

128.148.16.7

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111

<u>Notation</u>

- Write each byte (“octet”) as a decimal number
- This is called ”dotted decimal” or ”dotted quad” notation

IP Addressing

128.148.16.7

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111

32 bits => 2^{32} possible addresses… problem?

Notation

- Write each byte ("octet") as a decimal number
- This is called "dotted decimal" or "dotted quad" notation

Conceptually: an IP address has two parts

128.148.16.7

10000000 10010100 00010000 00000111

Conceptually: an IP address has two parts

=> Network part: identifies this network to the Internet

=> Host part: identifies hosts on that network

128.148.16.7

10000000 10010100 00010000 00000111

Size of host part vs. network part can vary (more on this later)

IP Addressing

Brown owns the range:

128.148.xxx.xxx

`10000000 10010100 xxxxxxxx xxxxxxxx`

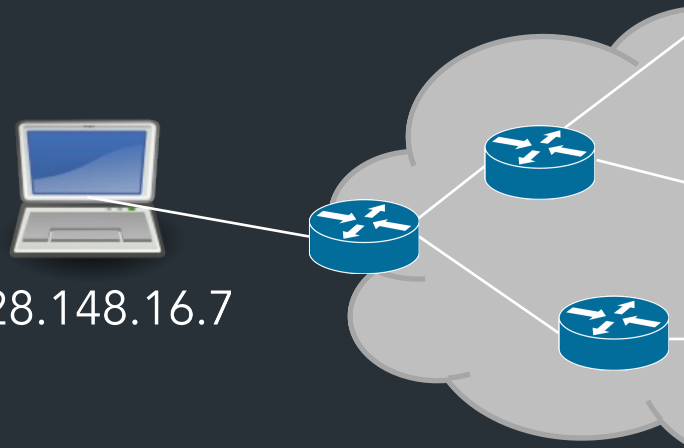
<u>Network part</u>
Identifies Brown (to the Internet)

<u>Host part</u>
Denotes individual hosts
within the Brown Network

Assigning numbers

- Networks are allocated ranges of IPs by global authority (ICANN)
  - Further subdivided by regions, ISPs, …

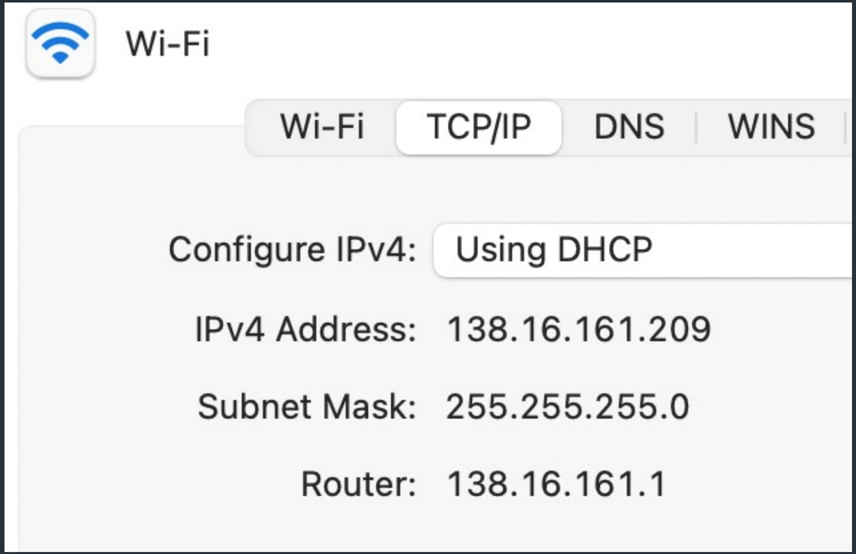
- Some IPs have special uses (eg. 127.0.0.1)



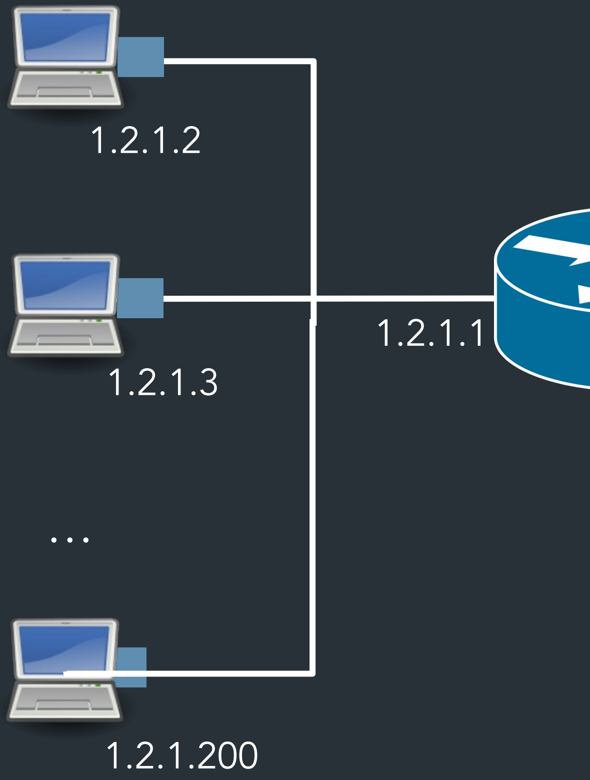
eg. Brown owns 128.148.xxx.xxx, 138.16.xxx.xxx

*ICANN (Internet Corporation for Assigned Names and Numbers)

A typical configuration



What do we do with this number?



What do we do with this number?

- Link layer: know how to communicate with devices on "your network"
- Routers: know about multiple networks => Use address to decide how to forward packets between them

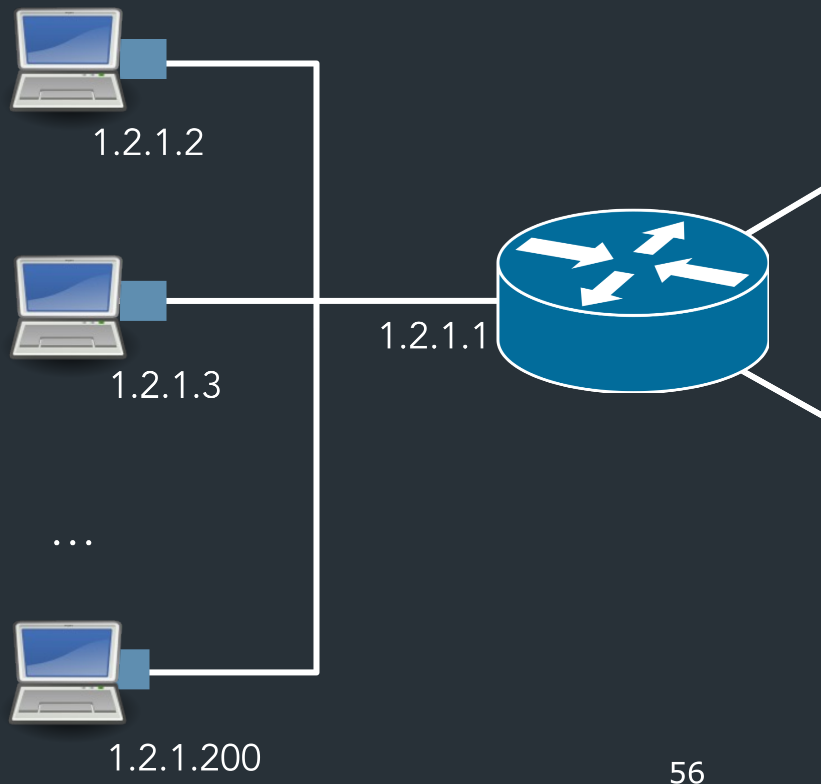
How IP forwarding works

Assume:

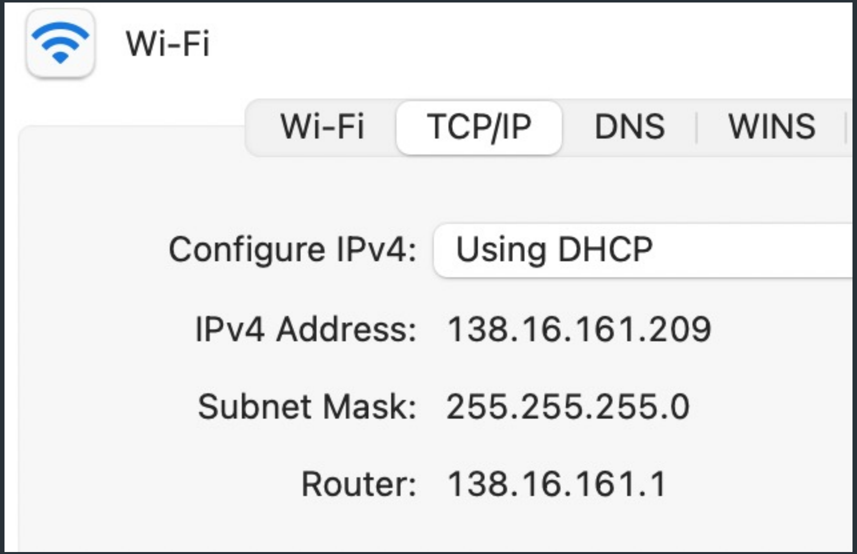
- Communicating on same network is easy—this is the link-layer's job!
- Can map IP addresses to MAC addresses (more on this later)

How to reach an address *outside* this network?

Send packets to a <u>router</u>, which <u>forwards</u> IP packets to other networks

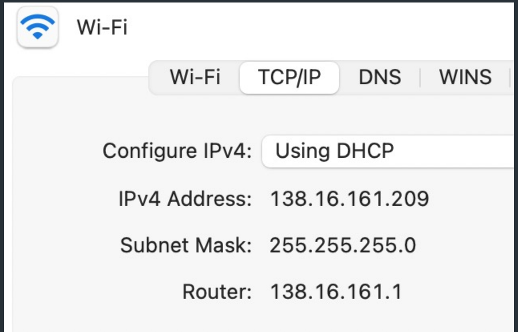


A typical configuration



What does it mean to be on the same network?

All systems with an IP address have a configuration like this



138.16.161.209

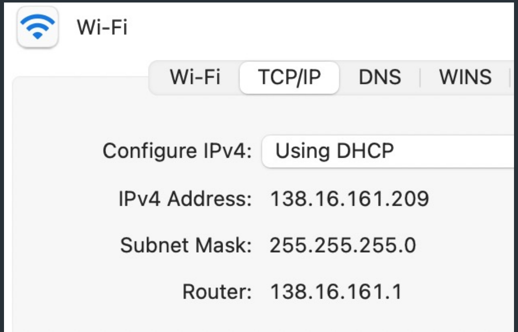
Addr: 138.16.161.209		10001010 00010000 10100001 11010001
Mask: 255.255.255.0	&	11111111 11111111 11111111 00000000
		10001010 00010000 10100001 00000000

24 bits

138.16.161.0

Can also write as 138.16.161.209/24
"Prefix notation" or "CIDR notation"

All systems with an IP address have a configuration like this



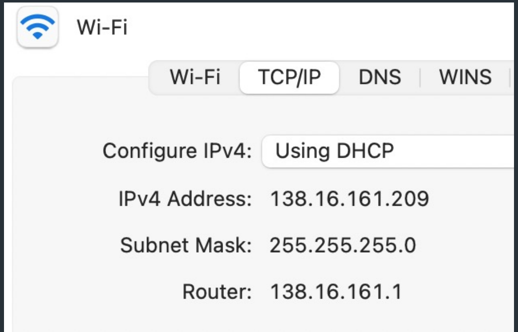
138.16.161.209

Addr:
138.16.161.209 `10001010 00010000 10100001 11010001`

Mask:
255.255.255.0 `11111111 11111111 11111111 00000000`

=> Bitmask used to “filter out” which part is for hosts on the same network

All systems with an IP address have a configuration like this



138.16.161.209

Addr:
138.16.161.209 — 10001010 00010000 10100001 11010001

Mask:
255.255.255.0 — & 11111111 11111111 11111111 00000000

10001010 00010000 10100001 00000000

24 bits

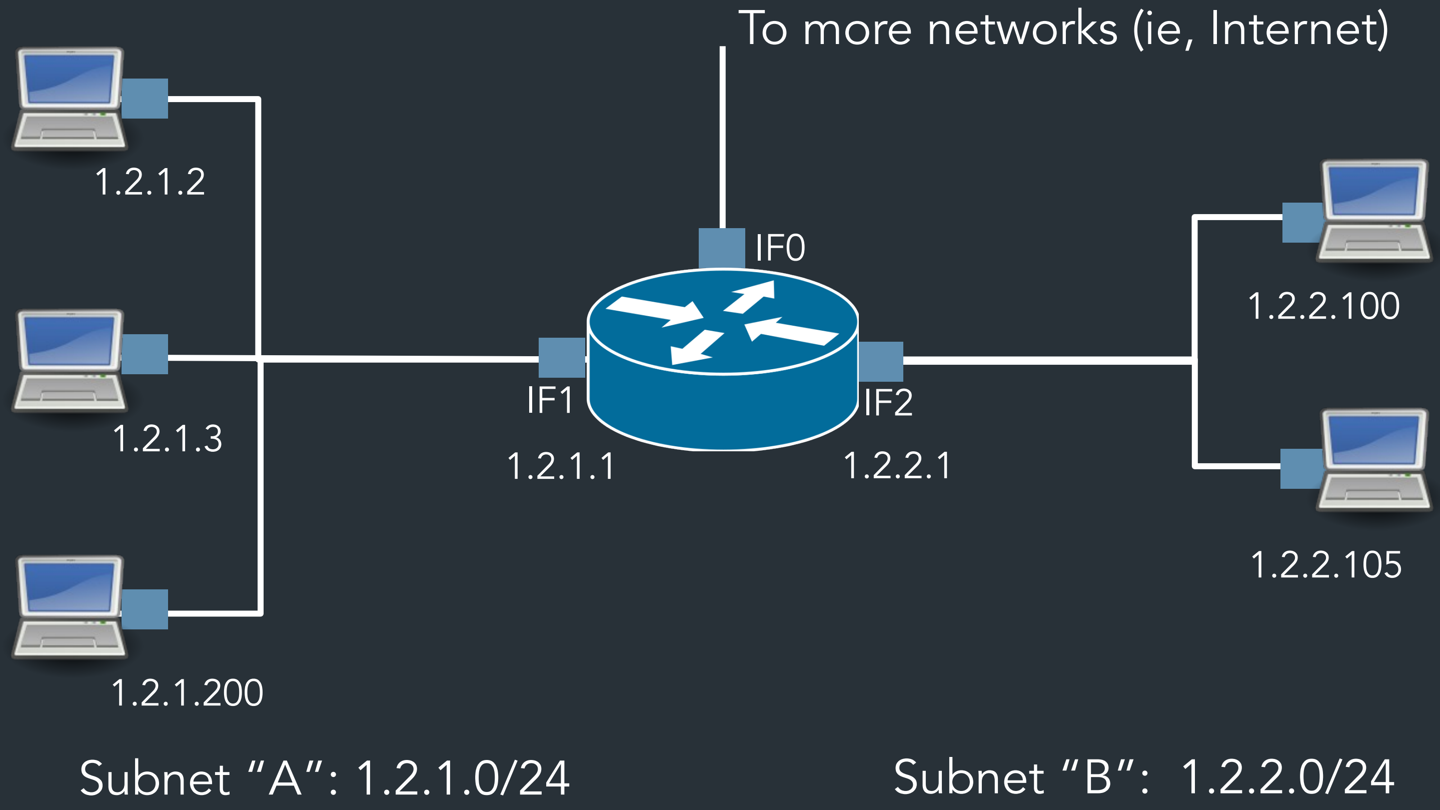
138.16.161.0

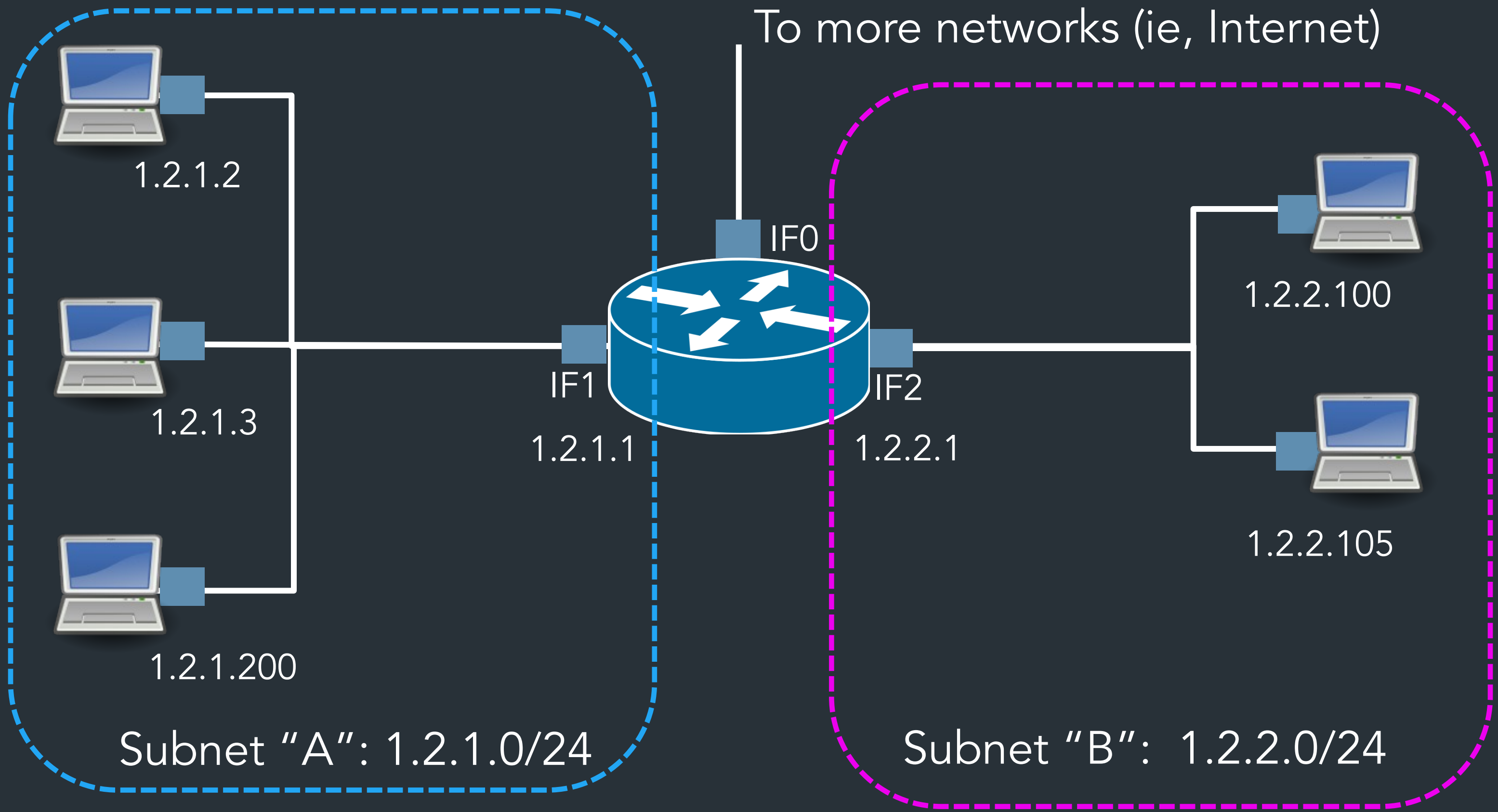
Can also write as 138.16.161.209/24
"Prefix notation" or "CIDR notation"

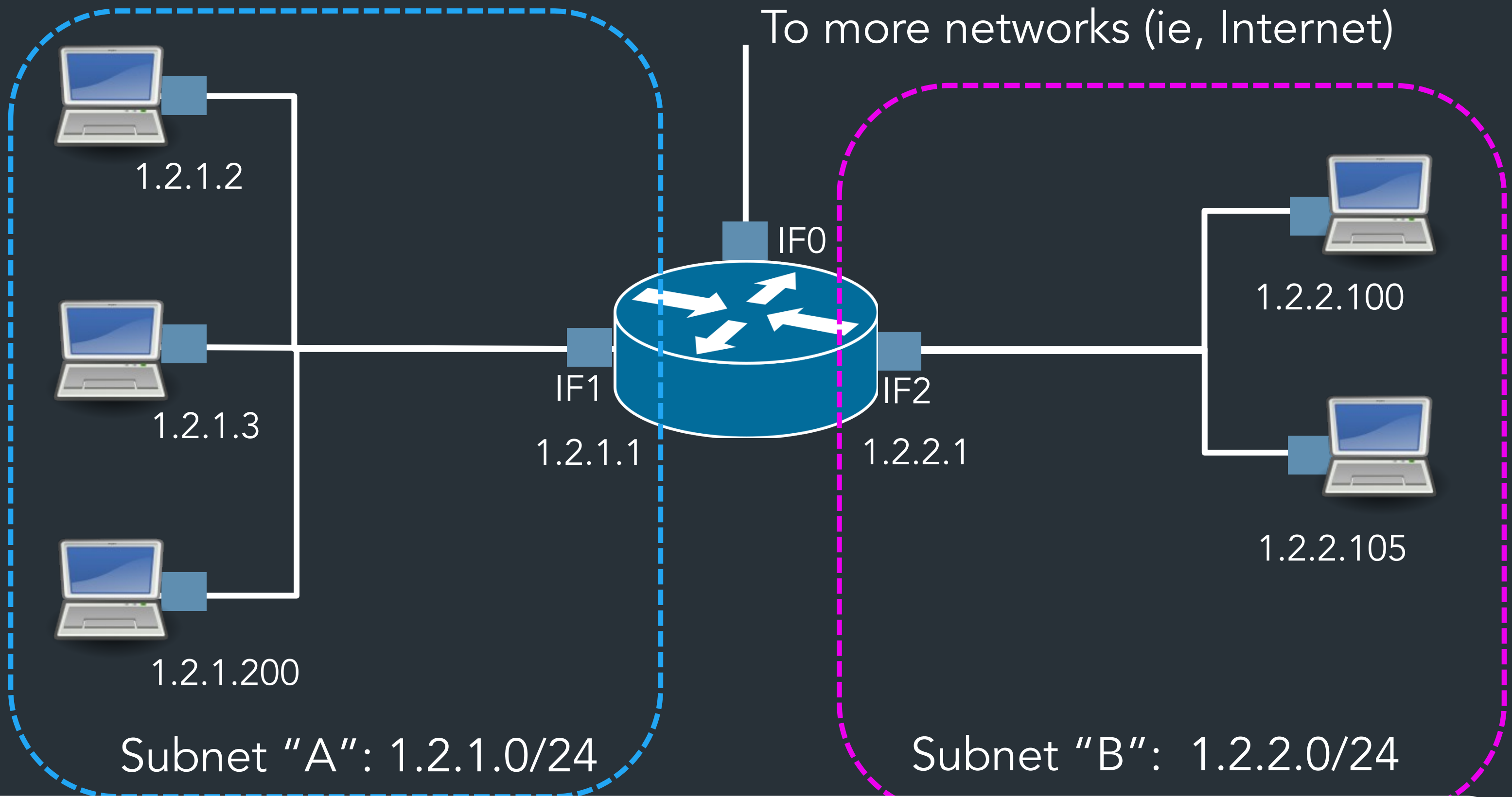
138.16.161.0/24	**10001010 00010000 10100001** xxxxxxxx
138.16.161.204	10001010 00010000 10100001 10100001
1.2.3.4	00000001 00000010 00000011 00000100

⇒ The mask can be any size 0-32
Not just checking the first three digits!

How do we move packets between networks?


To more networks (ie, Internet)
1.2.1.2
IF0
1.2.2.100
1.2.1.3
IF1
IF2
1.2.1.1
1.2.2.1
1.2.2.105
1.2.1.200
Subnet “A”: 1.2.1.0/24
Subnet “B”: 1.2.2.0/24


To more networks (ie, Internet)
IF0
1.2.1.2
1.2.1.3
1.2.1.200
IF1
1.2.1.1
IF2
1.2.2.1
1.2.2.100
1.2.2.105
Subnet "A": 1.2.1.0/24
Subnet "B": 1.2.2.0/24



The story so far:

=>Can "easily" communicate with nodes on the same network, but what about other networks?

=> Routers know about multiple networks, forward packets between them

IP Addressing

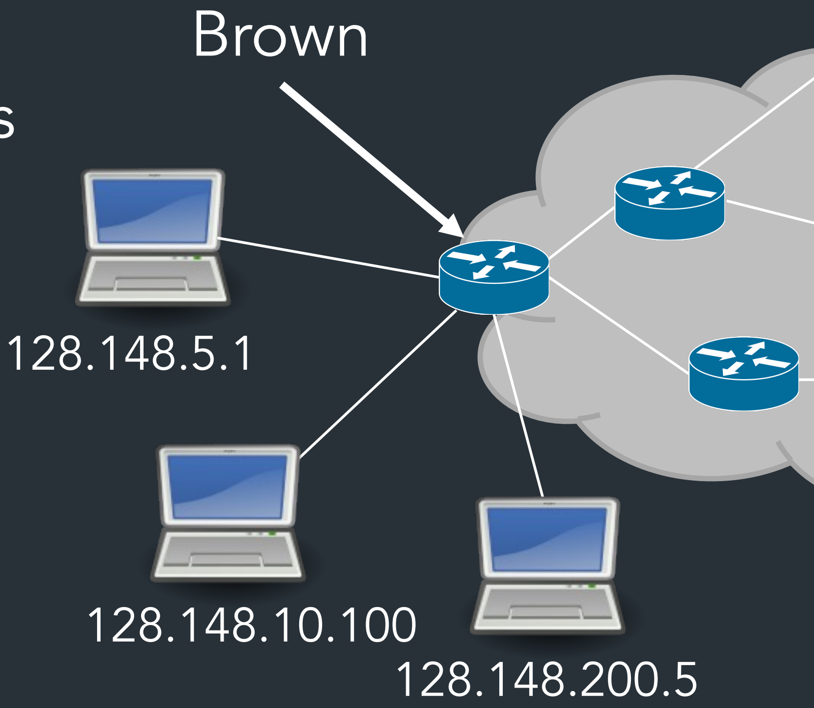
A network can designate IP addresses for its own hosts within its address range

For 128.148.xxx.xxx:

```
10000000 10010100 xxxxxxxx xxxxxxxx
```

Brown uses the the <u>prefix</u> 128.148.0.0/16

Some other ways to write this:
128.148/16
128.148.0.0 + subnet mask 255.255.0.0



Common prefixes

1.2.0.0/16	`00000001 00000010 xxxxxxxx xxxxxxxx`
8.0.0.0/8	`00001000 xxxxxxxx xxxxxxxx xxxxxxxx`
123.10.1.0/24	`01111011 00001010 00000001 xxxxxxxx`
201.112.10.200/30	`11001001 01110000 00001010 110010xx`

Example

How many addresses are in the network 1.1.0.0/20?

How do we move packets between networks?

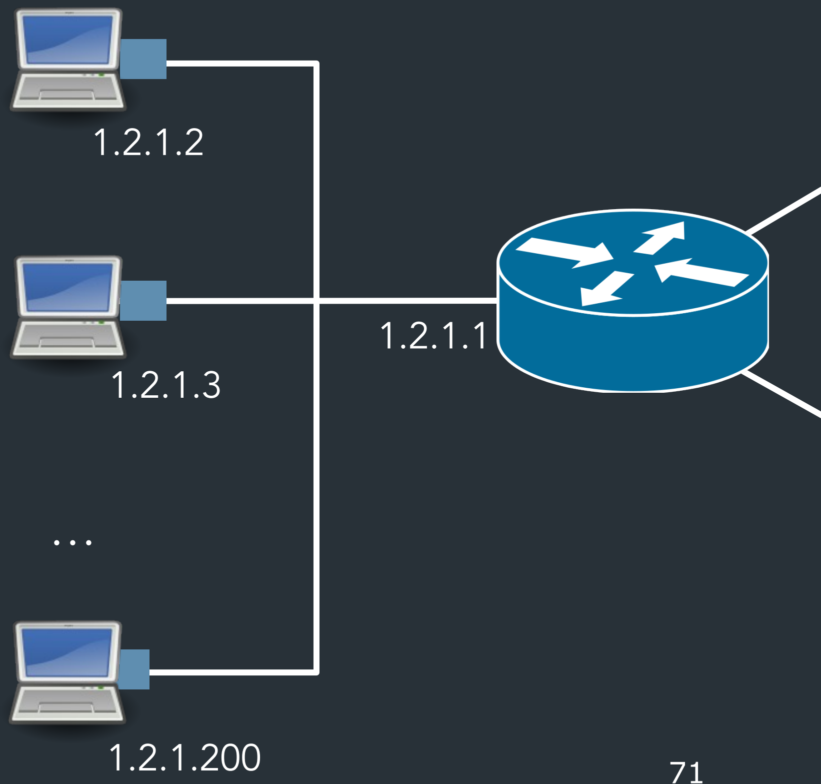
How IP forwarding works

Assume:

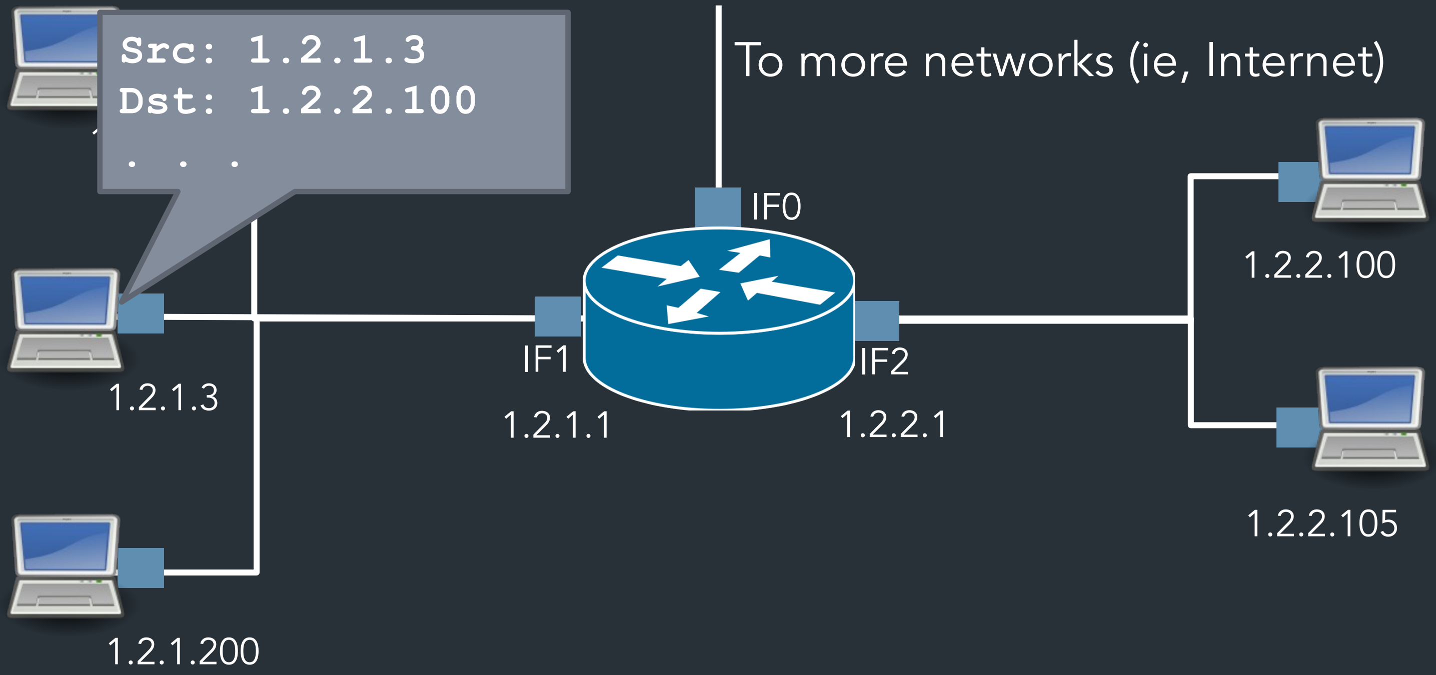
- Communicating on same network is easy—this is the link-layer's job!
- Can map IP addresses to MAC addresses (more on this later)

How to reach an address *outside* this network?

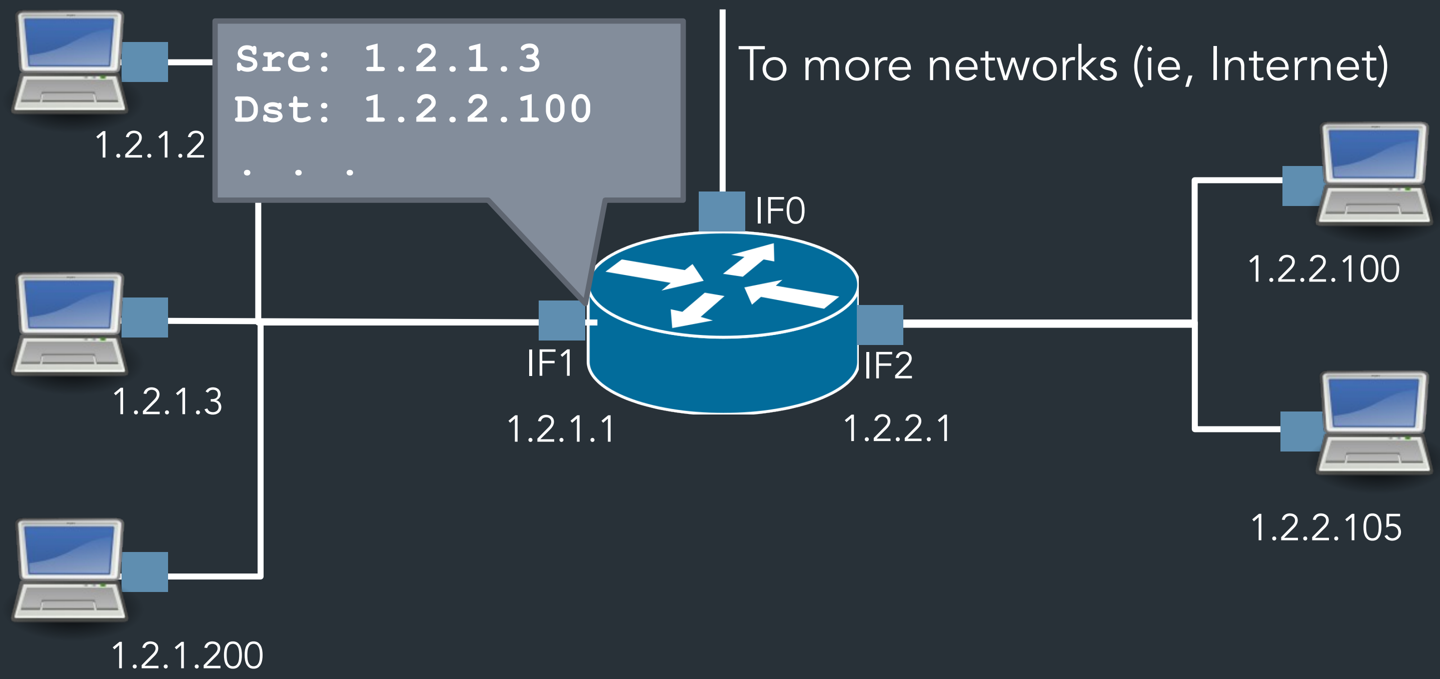
Send packets to a router, which forwards IP packets to other networks



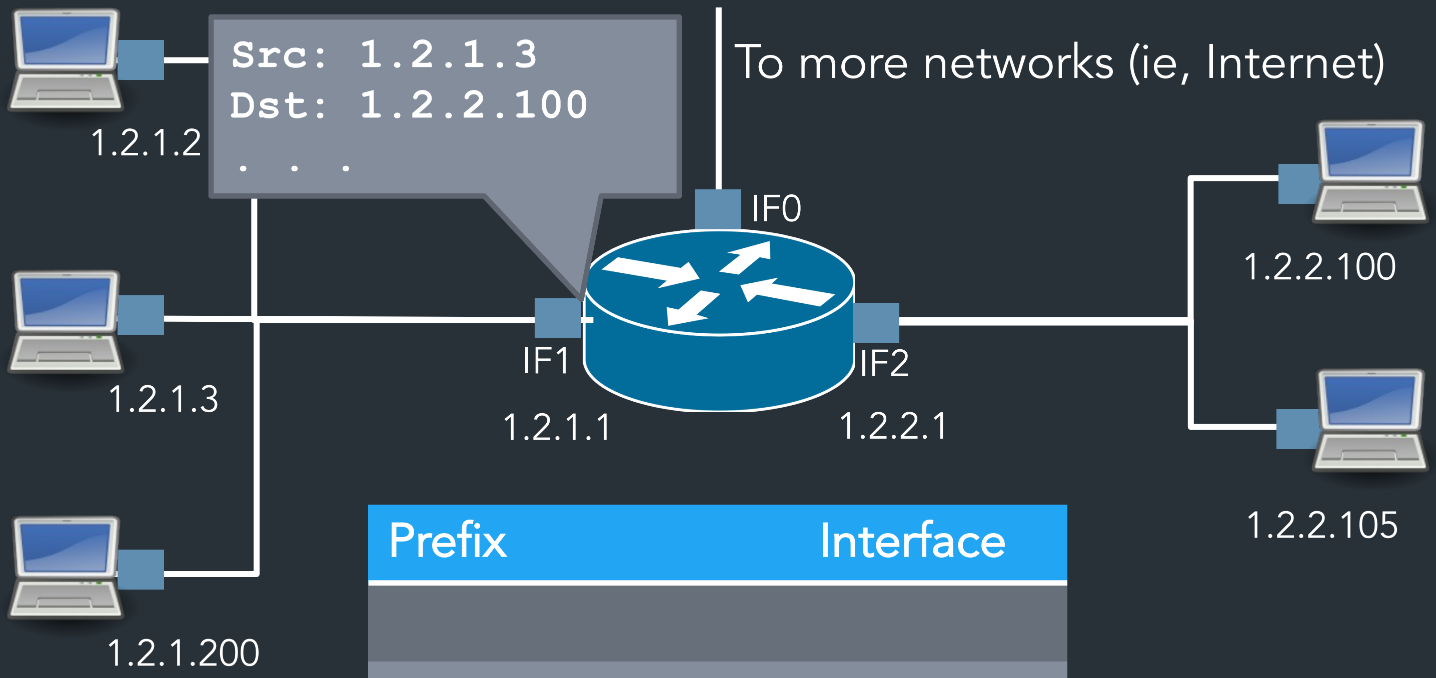
Forwarding IP packets



Forwarding IP packets

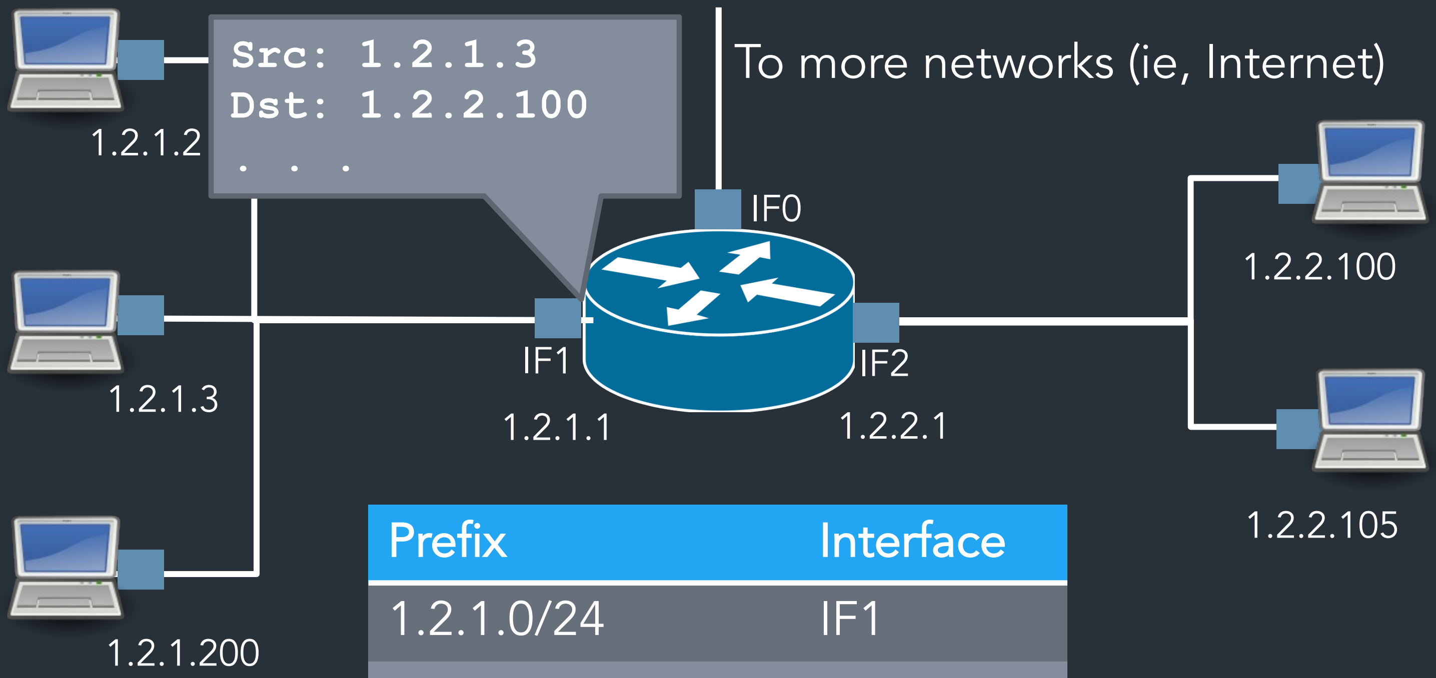


Forwarding IP packets



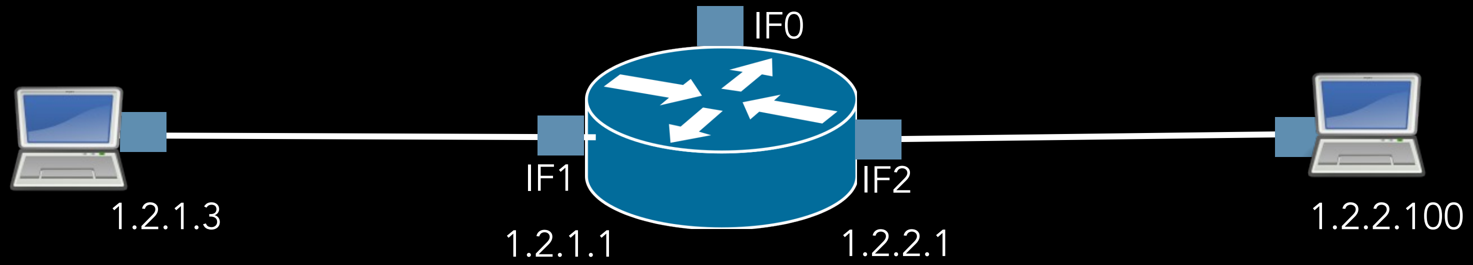
Prefix	Interface

Forwarding IP packets



Prefix	Interface
1.2.1.0/24	IF1
1.2.2.0/24	IF2
<everything else>	(IF0)

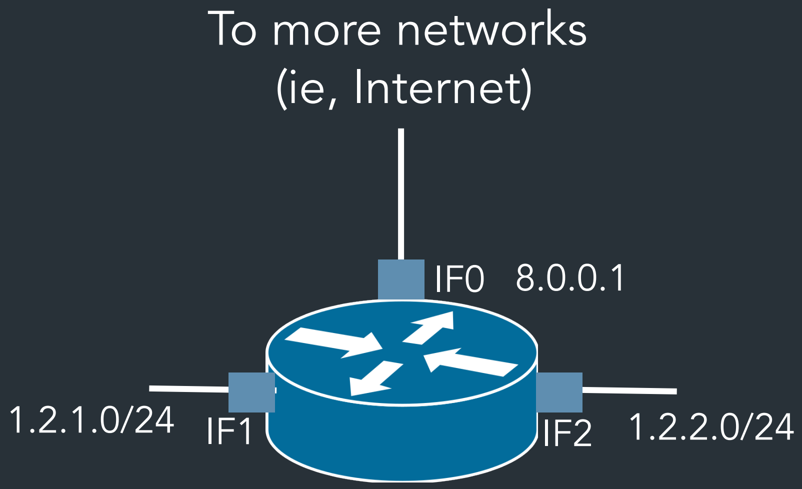
Wait, what happens at the link layer?



What about the rest?

How to reach networks that aren't directly connected?

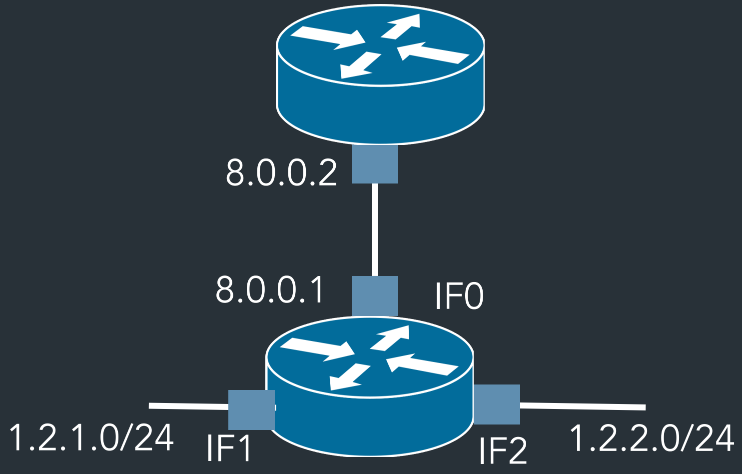
Prefix	Interface
1.2.1.0/24	IF1
1.2.2.0/24	IF2
<everything else>	IF0



What about the rest?

- Need "next hop" IP: another router that knows about other networks
  - How to reach it? Check table again!
- "Default gateway": where to send to reach anything not in the table

Prefix	IF/Next hop
1.2.1.0/24	IF1
1.2.2.0/24	IF2
8.0.0.0/30	IF0
128.148.0.0/16	1.2.1.5
Default	8.0.0.2



The forwarding table

Exploits hierarchical structure of addresses: know how to reach *networks*, not individual hosts

Prefix	IF/Next hop
1.2.1.0/24	IF1
1.2.2.0/24	IF2
8.0.0.0/30	IF0
128.148.0.0/16	1.2.1.5
Default	8.0.0.2

- Table is keyed is a network prefix, not a whole address
- Select best prefix with *longest prefix matching* (more on this later)

A forwarding table

```
# ip route
127.0.0.0/8 via 127.0.0.1 dev lo
172.17.44.0/24 dev enp7s0  proto kernel  scope link  src 172.17.44.22  metric 204
default via 172.17.44.1 dev eth0 src 172.17.44.22  metric 204
```

How do we move packets between networks?

IP forwarding

Given a packet, decide where to send it

Prefix	Interface/Next hop

A forwarding table (my laptop)

```
deemer@ceres ~ % ip route default via 10.3.128.1 dev wlp2s0
10.3.128.0/18 dev wlp2s0 proto dhcp scope link src 10.3.135.44 metric 3003
172.18.0.0/16 dev docker0 proto kernel scope link src 172.18.0.1
192.168.1.0/24 dev enp0s31f6 proto kernel scope link src 192.168.1.1
```

Forwarding: examples

Prefix	Interface/Next hop

Routing based on networks

A routing table

```
R6#sh ip ro
Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S*     0.0.0.0/0 [1/0] via 108.34.215.1
       10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C         10.1.0.0/24 is directly connected, wlan-ap0
L         10.1.0.2/32 is directly connected, wlan-ap0
O IA      10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C         10.1.48.0/24 is directly connected, Loopback0
L         10.1.48.1/32 is directly connected, Loopback0
C         10.20.30.32/31 is directly connected, Tunnel0
L         10.20.30.32/32 is directly connected, Tunnel0
       108.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C         108.34.215.0/24 is directly connected, GigabitEthernet0/0
L         108.34.215.208/32 is directly connected, GigabitEthernet0/0
       172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C         172.16.98.0/24 is directly connected, Vlan98
L         172.16.98.1/32 is directly connected, Vlan98
       172.17.0.0/16 is variably subnetted, 6 subnets, 3 masks
O IA      172.17.44.0/24 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C         172.17.48.0/24 is directly connected, Vlan20
L         172.17.48.1/32 is directly connected, Vlan20
C         172.17.49.0/25 is directly connected, Vlan50
```

A routing table

```
R6#sh ip ro
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override


Gateway of last resort is 108.34.215.1 to network 0.0.0.0


S*    0.0.0.0/0 [1/0] via 108.34.215.1
      10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C        10.1.0.0/24 is directly connected, wlan-ap0
L        10.1.0.2/32 is directly connected, wlan-ap0
O IA     10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C        10.1.48.0/24 is directly connected, Loopback0
L        10.1.48.1/32 is directly connected, Loopback0
C        10.20.30.32/31 is directly connected, Tunnel0
L        10.20.30.32/32 is directly connected, Tunnel0
```

A large table

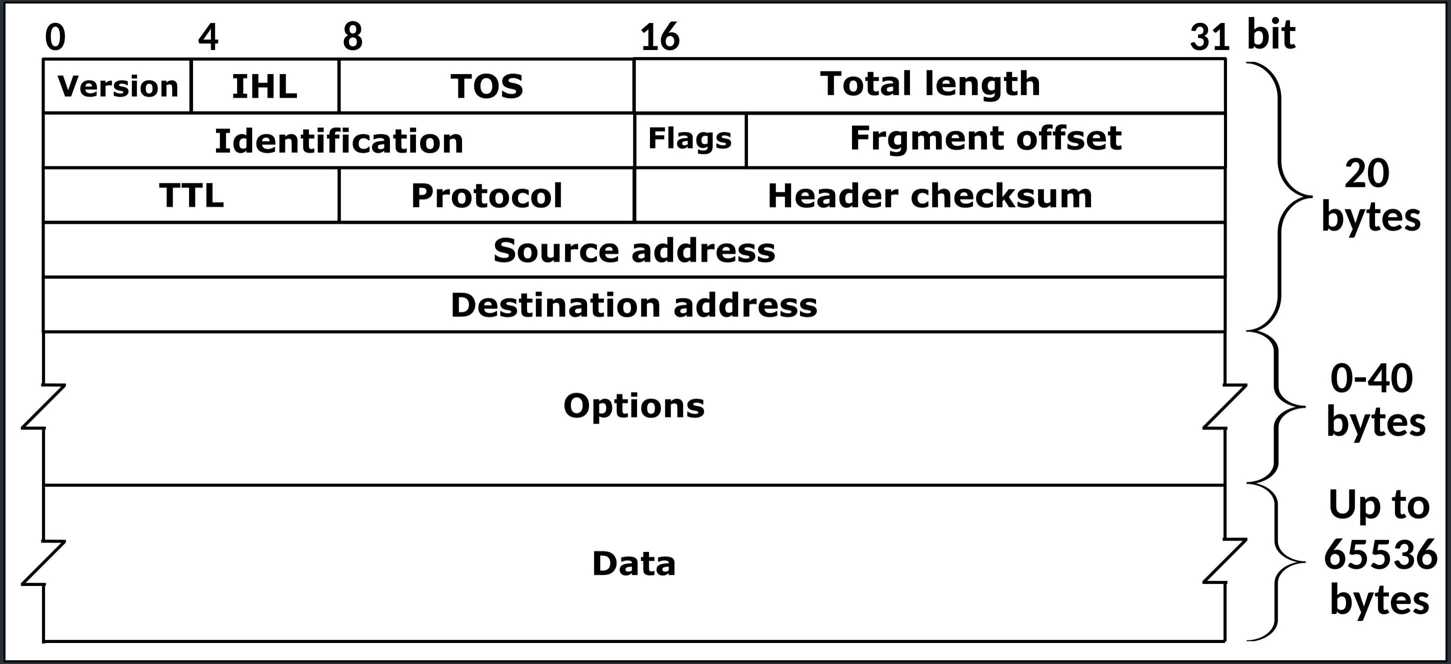
```
rviews@route-server.ip.att.net>show route table inet.0 active-path

inet.0: 866991 destinations, 13870153 routes (866991 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both


0.0.0.0/0          *[Static/5] 5w0d 19:43:09
                    > to 12.0.1.1 via em0.0
1.0.0.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 13335 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.4.0/22         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.4.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.5.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.6.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
```

How does forwarding actually work?

The IPv4 Header



Defined by RFC 791
RFC (Request for Comment): defines network standard

Most Important fields

- Version: 4 for IPv4 packets, 6 for IPv6
- Source address: where the packet came from
- Destination address: where the packet is going

(continued...)

More important fields

- TTL (time-to-live): decremented each hop
  - Can prevent forwarding loops (and do other stuff…)
- Checksum: computed over header (very weak!)
- Protocol identifier: describes what's in the packet
  - 6: TCP, 17: UDP, 1: ICMP, …
  - Defines the type of the payload

Less important fields

- Header length: in 32-bit units
  - >5 implies use of IP options
  - Almost all routers ignore IP options
- Fragmentation
  - Network can fragment a packet if next link requires a small frame
  - Most routers don't fragment (or reassemble fragments)
- We won't talk about…
  - Type of Service (TOS): basic traffic classification
  - Identifier: might have special meaning on some networks

Forwarding steps

What does a router do when it receives a packet?

Forwarding mechanics

When an IP packet arrives at a host/router:

- Is it valid? Verify checksum over *header*
- **Is it for me?** If dest IP == your address, send to OS
- If not, where should it go?
  - Consult forwarding table => find next hop
  - Decrement TTL
  - Send packet to next hop

Traceroute

- When TTL reaches 0, router may send back an error
  - ICMP TTL exceeded
- If it does, we can identify a path used by a packet!

Traceroute example

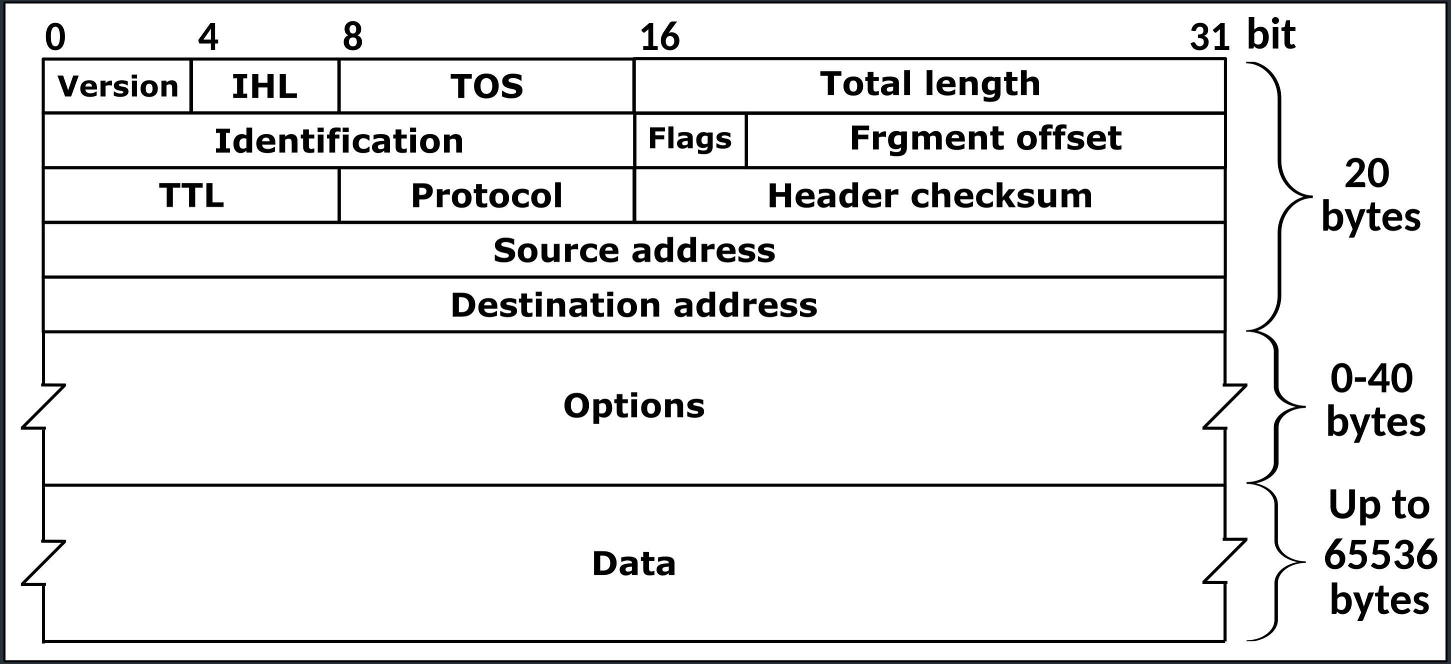
```
[deemer@Warsprite ~]$ traceroute -q 1 google.com
traceroute to google.com (142.251.40.174), 30 hops max, 60 byte packets
 1  router1-nac.linode.com (207.99.1.13)  0.621 ms
 2  if-0-1-0-0-0.gw1.cjj1.us.linode.com (173.255.239.26)  0.499 ms
 3  72.14.222.136 (72.14.222.136)  0.949 ms
 4  72.14.222.136 (72.14.222.136)  0.919 ms
 5  108.170.248.65 (108.170.248.65)  1.842 ms
 6  lga25s81-in-f14.1e100.net (142.251.40.174)  1.812 ms
```

Traceroute example

```
[deemer@Warsprite ~]$ traceroute -q 1 amazon.co.uk
traceroute to amazon.co.uk (178.236.7.220), 30 hops max, 60 byte packets
 1  router2-nac.linode.com (207.99.1.14)  0.577 ms
 2  if-11-1-0-1-0.gw2.cjj1.us.linode.com (173.255.239.16)  0.461 ms
 3  ix-et-2-0-2-0.tcore3.njy-newark.as6453.net (66.198.70.104)  1.025 ms
 4  be3294.ccr41.jfk02.atlas.cogentco.com (154.54.47.217)  2.938 ms
 5  be2317.ccr41.lon13.atlas.cogentco.com (154.54.30.186)  69.725 ms
 6  be2350.rcr21.b023101-0.lon13.atlas.cogentco.com (130.117.51.138)  69.947 ms
 7  a100-row.demarc.cogentco.com (149.11.173.122)  71.639 ms
 8  150.222.15.28 (150.222.15.28)  78.217 ms
 9  150.222.15.21 (150.222.15.21)  84.383 ms
10  *
11  150.222.15.4 (150.222.15.4)  74.529 ms
                                        . . .
30  178.236.14.162 (178.236.14.162)  83.659 ms
```

Demo: IP project

The IPv4 Header



Defined by RFC 791
RFC (Request for Comment): defines network standard

Important fields

- Version: 4 for IPv4 packets, 6 for IPv6
- Destination address: used for forwarding
- TTL (time-to-live): decremented each hop
  - Can prevent forwarding loops (and do other stuff…)
- Checksum: computed over header (very weak!)
- Protocol identifier: describes what's in the packet
  - 6: TCP, 17: UDP, 1: ICMP, …
  - Defines the type of the payload

Less important fields

- Header length: in 32-bit units
  - >5 implies use of IP options
  - Almost all routers ignore IP options
- Fragmentation
  - Network can fragment a packet if next link requires a small frame
  - Most routers don't fragment (or reassemble fragments)
- We won't talk about…
  - Type of Service (TOS): basic traffic classification
  - Identifier: might have special meaning on some networks

Forwarding mechanics

When an IP packet arrives at a host/router:

- Is it valid? Verify checksum over *header*
- **Is it for me?** If dest IP == your address, send to OS
- If not, where should it go?
  - Consult forwarding table => find next hop
  - Decrement TTL
  - Send packet to next hop

Traceroute

- When TTL reaches 0, router may send back an error
  - ICMP TTL exceeded
- If it does, we can identify a path used by a packet!

Traceroute example

```
[deemer@Warsprite ~]$ traceroute -q 1 google.com
traceroute to google.com (142.251.40.174), 30 hops max, 60 byte packets
 1  router1-nac.linode.com (207.99.1.13)  0.621 ms
 2  if-0-1-0-0-0.gw1.cjj1.us.linode.com (173.255.239.26)  0.499 ms
 3  72.14.222.136 (72.14.222.136)  0.949 ms
 4  72.14.222.136 (72.14.222.136)  0.919 ms
 5  108.170.248.65 (108.170.248.65)  1.842 ms
 6  lga25s81-in-f14.1e100.net (142.251.40.174)  1.812 ms
```

Traceroute example

```
[deemer@Warsprite ~]$ traceroute -q 1 amazon.co.uk
traceroute to amazon.co.uk (178.236.7.220), 30 hops max, 60 byte packets
 1  router2-nac.linode.com (207.99.1.14)  0.577 ms
 2  if-11-1-0-1-0.gw2.cjj1.us.linode.com (173.255.239.16)  0.461 ms
 3  ix-et-2-0-2-0.tcore3.njy-newark.as6453.net (66.198.70.104)  1.025 ms
 4  be3294.ccr41.jfk02.atlas.cogentco.com (154.54.47.217)  2.938 ms
 5  be2317.ccr41.lon13.atlas.cogentco.com (154.54.30.186)  69.725 ms
 6  be2350.rcr21.b023101-0.lon13.atlas.cogentco.com (130.117.51.138)  69.947 ms
 7  a100-row.demarc.cogentco.com (149.11.173.122)  71.639 ms
 8  150.222.15.28 (150.222.15.28)  78.217 ms
 9  150.222.15.21 (150.222.15.21)  84.383 ms
10  *
11  150.222.15.4 (150.222.15.4)  74.529 ms
                                        . . .
30  178.236.14.162 (178.236.14.162)  83.659 ms
```

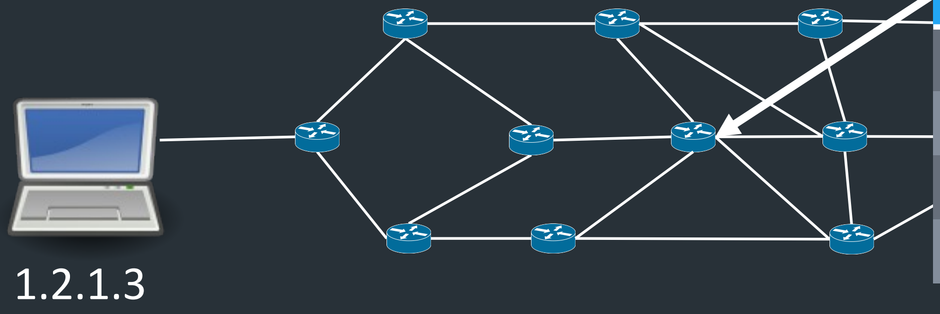
Demo: IP project

Coming up…

- ARP: Mapping IPs to MAC addresses
- How are addresses assigned?
- NAT: When it gets complicated
- Routing algorithms: how to build forwarding tables

Fill out the group preference survey for the IP project (announcement soon) by
tomorrow (Sep 30) by 11:59PM

Putting it all together…



Prefix	Interface
. . .	
1.2.0.0/16	1.2.0.1 via IF42
142.0.0/8	5.0.0.5 via IF21
. . .	

- The more connected a router becomes, the more complex its forwarding table… and the more it may change!

- <u>Routing</u> algorithms: routers exchange path information to their forwarding tables (more on this later)

Goal: find the most specific (ie, longest) prefix matching the destination

How to reach 1.2.2.100?

Prefix	Interface
1.2.1.0/24	IF1
1.2.2.0/24	IF2
0.0.0.0/0	IF0

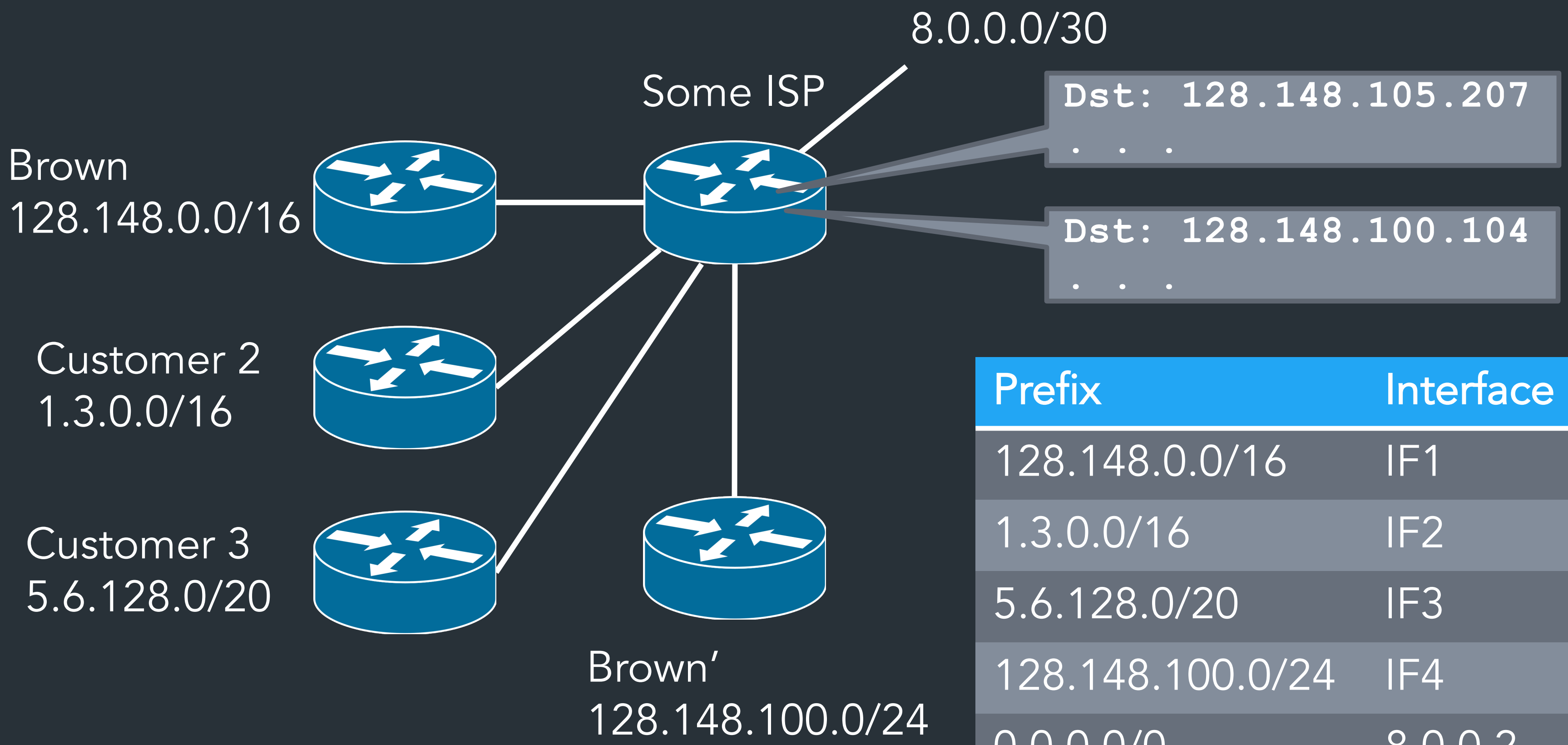
```
1.2.2.100       00000001.00000010.00000010.01100100
                               ?=
1.2.1.0/24      00000001.00000010.00000001.xxxxxxxx

1.2.2.0/24      00000001.00000010.00000010.xxxxxxxx

0.0.0.0/0       xxxxxxxx.xxxxxxxx.xxxxxxxx.xxxxxxxx
```

Output: IF2

Longest Prefix Matching (LPM): can represent entire IP space in (small) table!



Prefix	Interface
128.148.0.0/16	IF1
1.3.0.0/16	IF2
5.6.128.0/20	IF3
128.148.100.0/24	IF4
0.0.0.0/0	8.0.0.2

A routing table

```
R6#sh ip ro
Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S*     0.0.0.0/0 [1/0] via 108.34.215.1
       10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C         10.1.0.0/24 is directly connected, wlan-ap0
L         10.1.0.2/32 is directly connected, wlan-ap0
O IA      10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C         10.1.48.0/24 is directly connected, Loopback0
L         10.1.48.1/32 is directly connected, Loopback0
C         10.20.30.32/31 is directly connected, Tunnel0
L         10.20.30.32/32 is directly connected, Tunnel0
       108.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C         108.34.215.0/24 is directly connected, GigabitEthernet0/0
L         108.34.215.208/32 is directly connected, GigabitEthernet0/0
       172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C         172.16.98.0/24 is directly connected, Vlan98
L         172.16.98.1/32 is directly connected, Vlan98
       172.17.0.0/16 is variably subnetted, 6 subnets, 3 masks
O IA      172.17.44.0/24 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C         172.17.48.0/24 is directly connected, Vlan20
L         172.17.48.1/32 is directly connected, Vlan20
C         172.17.49.0/25 is directly connected, Vlan50
```

A routing table

```
R6#sh ip ro
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
       + - replicated route, % - next hop override


Gateway of last resort is 108.34.215.1 to network 0.0.0.0


S*    0.0.0.0/0 [1/0] via 108.34.215.1
      10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C        10.1.0.0/24 is directly connected, wlan-ap0
L        10.1.0.2/32 is directly connected, wlan-ap0
O IA     10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C        10.1.48.0/24 is directly connected, Loopback0
L        10.1.48.1/32 is directly connected, Loopback0
C        10.20.30.32/31 is directly connected, Tunnel0
L        10.20.30.32/32 is directly connected, Tunnel0
```

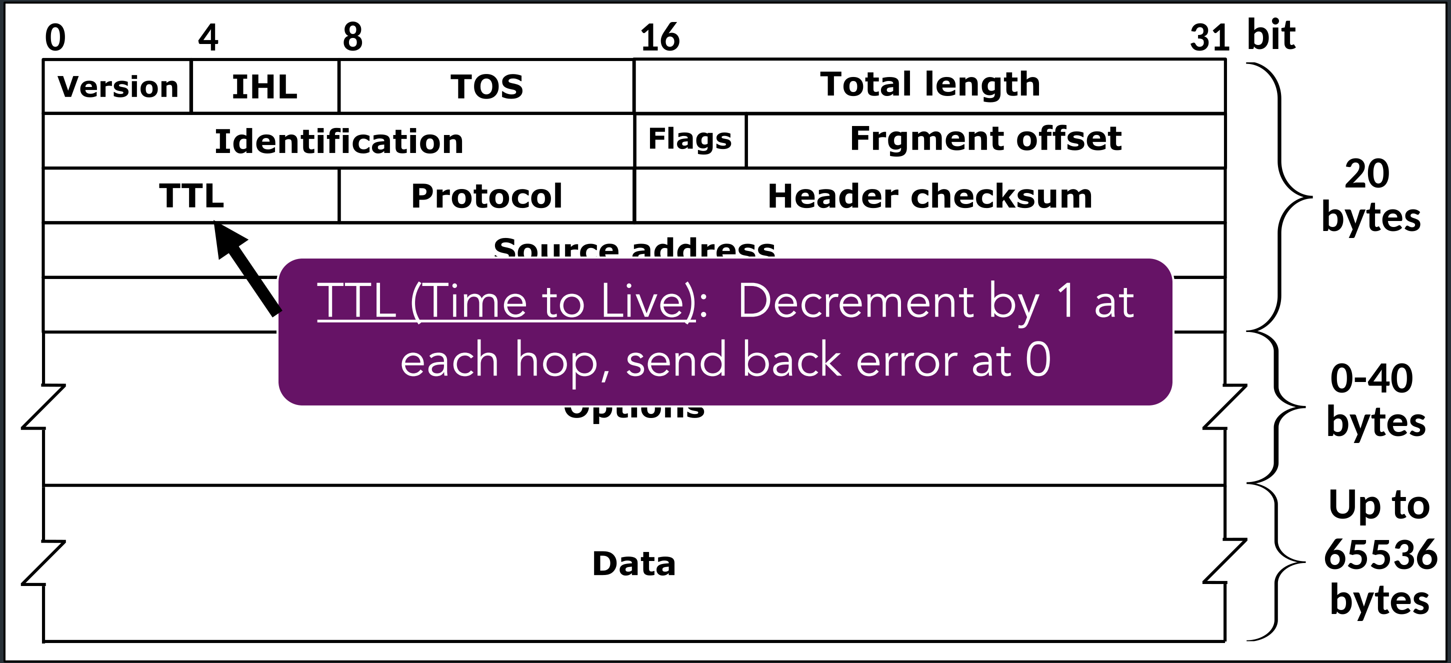
A large table

```
rviews@route-server.ip.att.net>show route table inet.0 active-path

inet.0: 866991 destinations, 13870153 routes (866991 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both


0.0.0.0/0          *[Static/5] 5w0d 19:43:09
                    > to 12.0.1.1 via em0.0
1.0.0.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 13335 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.4.0/22         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.4.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.5.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
1.0.6.0/24         *[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238
                      AS path: 7018 3356 4826 38803 I, validation-state: valid
                    > to 12.0.1.1 via em0.0
```

How to avoid loops?



TTL (Time to Live): Decrement by 1 at each hop, send back error at 0

traceroute: tool to send packets with increasing TTLs
=> can learn about network paths!