CSE 461: Computer networks

Spring 2021

Ratul Mahajan

IP Addresses

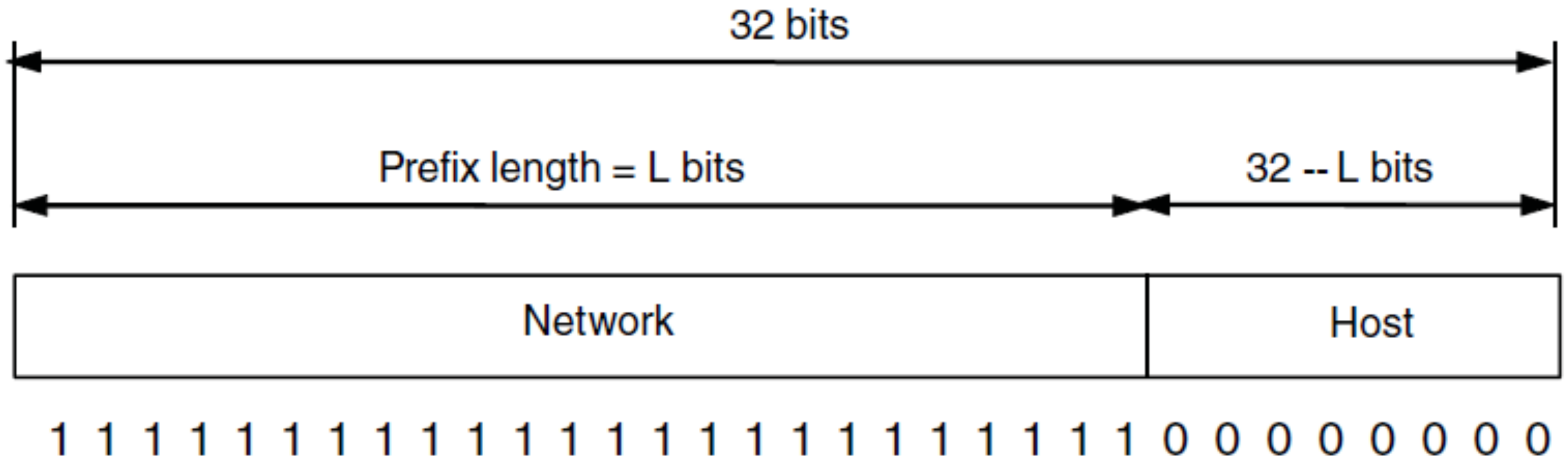
- IPv4 uses 32-bit addresses
  - Later we'll see IPv6, which uses 128-bit addresses
- Written in "dotted quad" notation
  - Four 8-bit numbers separated by dots

$$\overbrace{\texttt{aaaaaaaa}}^{\text{8 bits}}\overbrace{\texttt{bbbbbbbb}}^{\text{8 bits}}\overbrace{\texttt{cccccccc}}^{\text{8 bits}}\overbrace{\texttt{dddddddd}}^{\text{8 bits}} \leftrightarrow \texttt{A.B.C.D}$$

$$\texttt{00010010}\,|\,\texttt{00011111}\,|\,\texttt{00000000}\,|\,\texttt{00000001} \leftrightarrow ??$$

IP Prefixes

- Addresses are allocated in blocks called <u>prefixes</u>
  - Addresses in an L-bit prefix have the same top L bits
  - There are 2^{32-L} addresses aligned on 2^{32-L} boundary



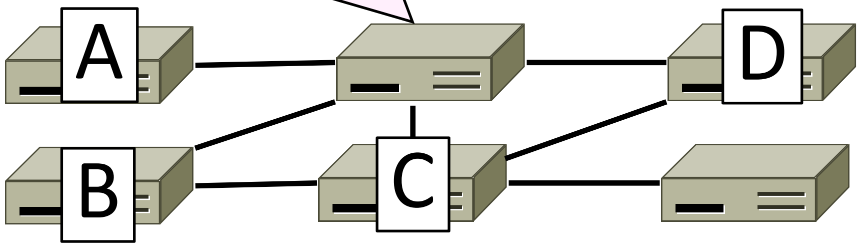
IP Prefixes (2)

- Written in “IP address/length” notation
  - Address is lowest address in the prefix, length is prefix bits
  - E.g., 128.13.0.0/16 is 128.13.0.0 to 128.13.255.255
  - So a /24 (“slash 24”) is 256 addresses and /32 is 1 address

IP Forwarding

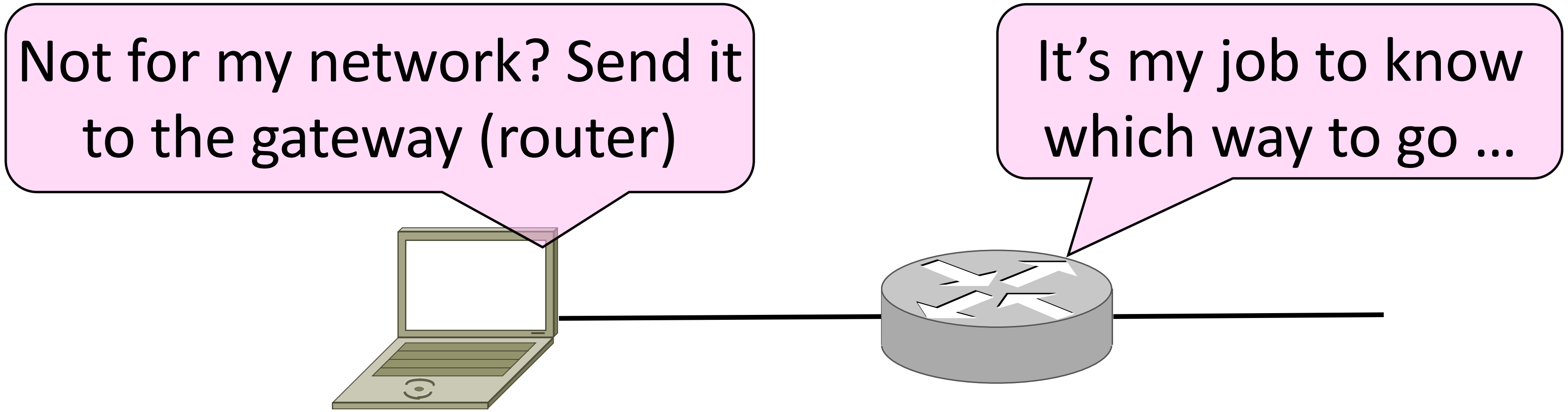
- Nodes use a table that lists the next hop for prefixes
- Lookup the destination address's prefix in the table

Prefix	Next Hop
102.24.0.0/19	D
192.24.12.0/22	B



Host/Router Distinction

- In the Internet:
  - Routers do the routing, know way to all destinations
  - Hosts send remote traffic (out of prefix) to nearest router



Host Networking

- Consists of 4 pieces of data:
  - IP Address
  - Subnet Mask
    - Defines local addresses
  - Gateway
    - Who (local) to send non-local packets to for routing
  - DNS Server (Later)

Host Forwarding Table

Prefix	Next Hop
My network prefix	Send on local link
Default (0.0.0.0/0)	Send to my router

```
[Ratuls-MacBook-Pro:19wi ratul$ netstat -r -f inet | grep 192
default            192.168.88.1       UGSc          85       30     en0
192.168.88         link#10            UCS            0        0     en0       !
192.168.88.1/32    link#10            UCS            2        0     en0       !
192.168.88.14/32   link#10            UCS            0        0     en0       !
```

Issues?

- Where does this break down?

Bootstrapping (DHCP)

Finding Link nodes (ARP)

Really big packets (Fragmentation)

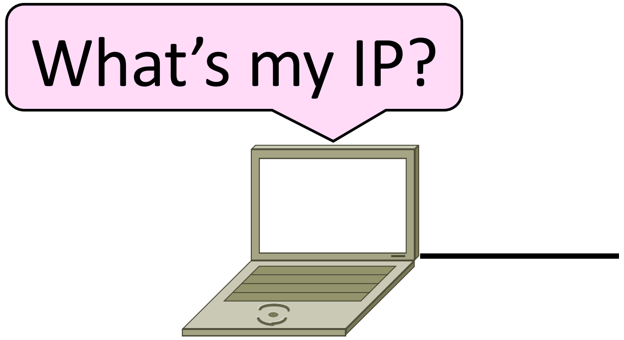
Errors in the network (ICMP)

Running out of addresses (IPv6, NAT)

Dynamic Host Configuration Protocol (DHCP)

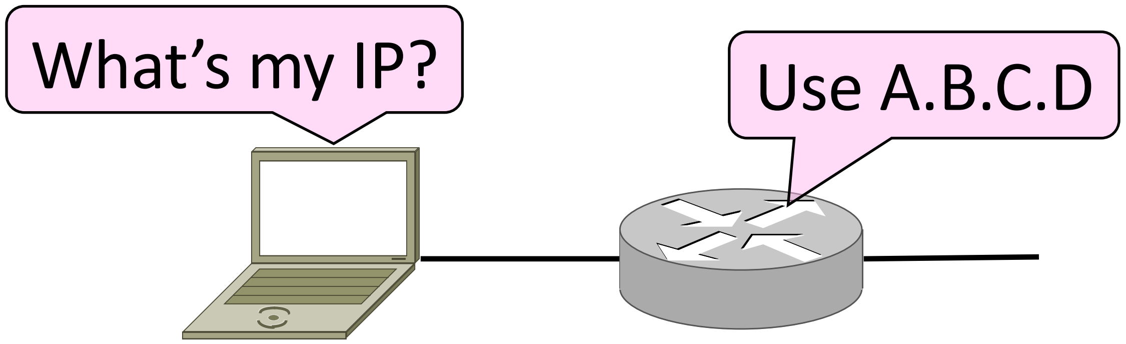
Bootstrapping

- Problem:
  - A node wakes up for the first time …
  - What is its IP address? What’s the IP address of its router?
  - At least Ethernet address is on NIC



Bootstrapping (2)

1. Manual configuration (old days)
   - Can’t be factory set, depends on use
2. DHCP: Automatically configure addresses



DHCP

- DHCP (Dynamic Host Configuration Protocol), from 1993, widely used
- It <u>leases</u> IP address to nodes
- Provides other parameters too
  - Network prefix
  - Address of local router
  - DNS server, time server, etc.

DHCP Protocol Stack

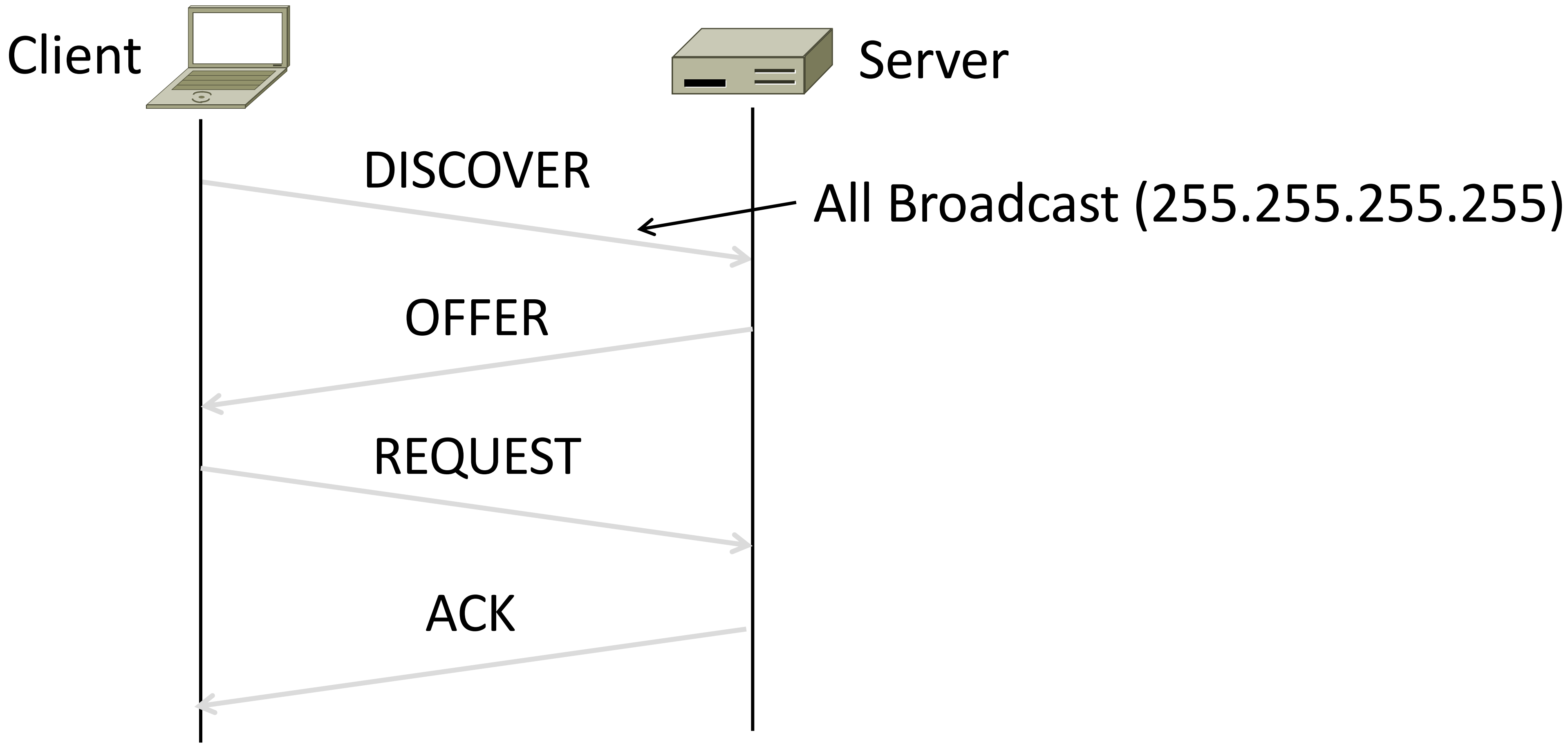
- DHCP is a client-server application
  - Uses UDP ports 67, 68

DHCP
UDP
IP
Ethernet

DHCP Addressing

- Bootstrap issue:
  - How does node send a message to DHCP server before it is configured?
- Answer:
  - Node sends broadcast messages that delivered to all nodes on the link-level network
  - Broadcast address is all 1s
  - IP (32 bit): 255.255.255.255
  - Ethernet (48 bit): ff:ff:ff:ff:ff:ff

DHCP Messages



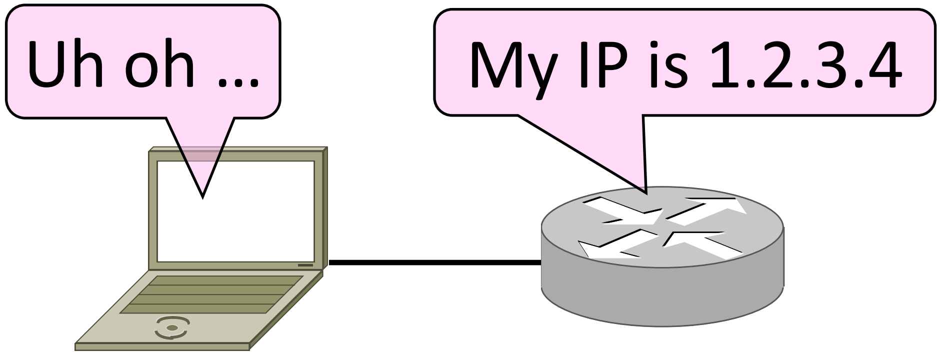
DHCP Messages (2)

- To renew an existing lease, an abbreviated sequence is used:
  - REQUEST, followed by ACK

Address Resolution Protocol (ARP)

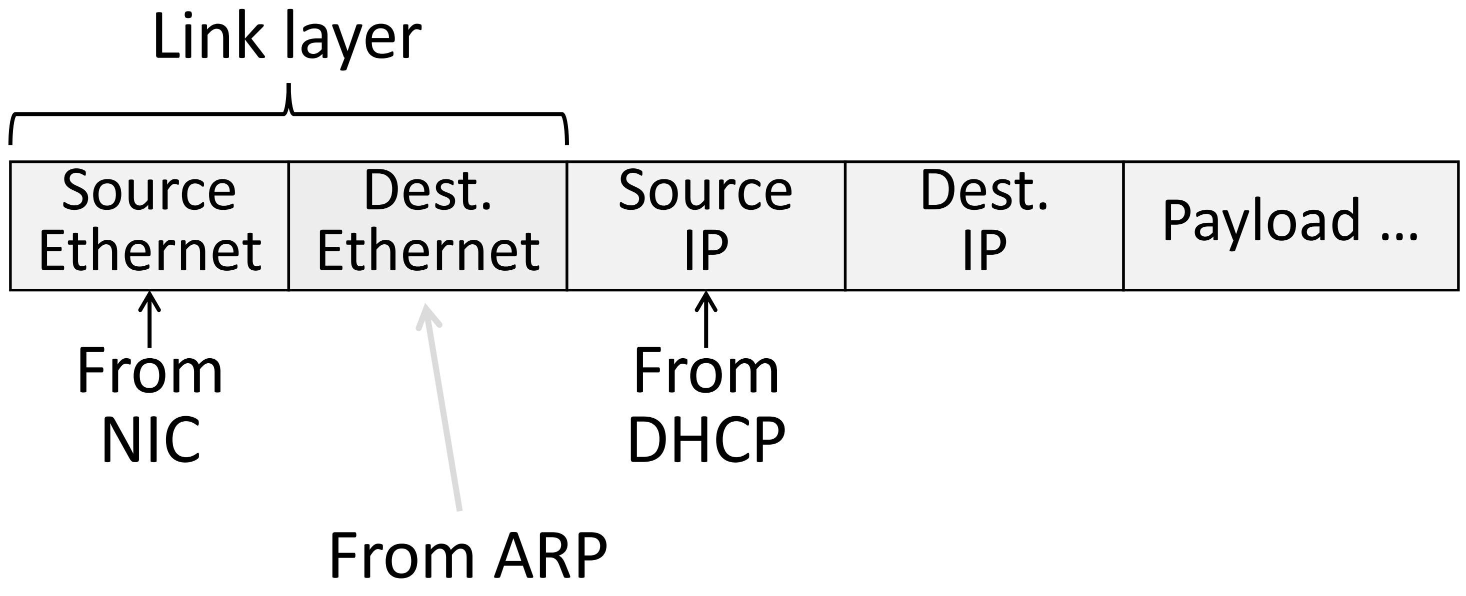
Sending an IP Packet

- Problem:
  - A node needs Link layer addresses to send a frame over the local link
  - How does it get the destination link address from a destination IP address?



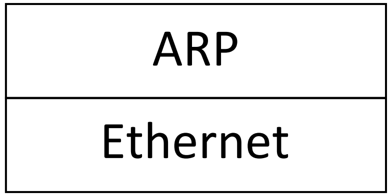
ARP (Address Resolution Protocol)

- Node uses to map a local IP address to its Link layer addresses

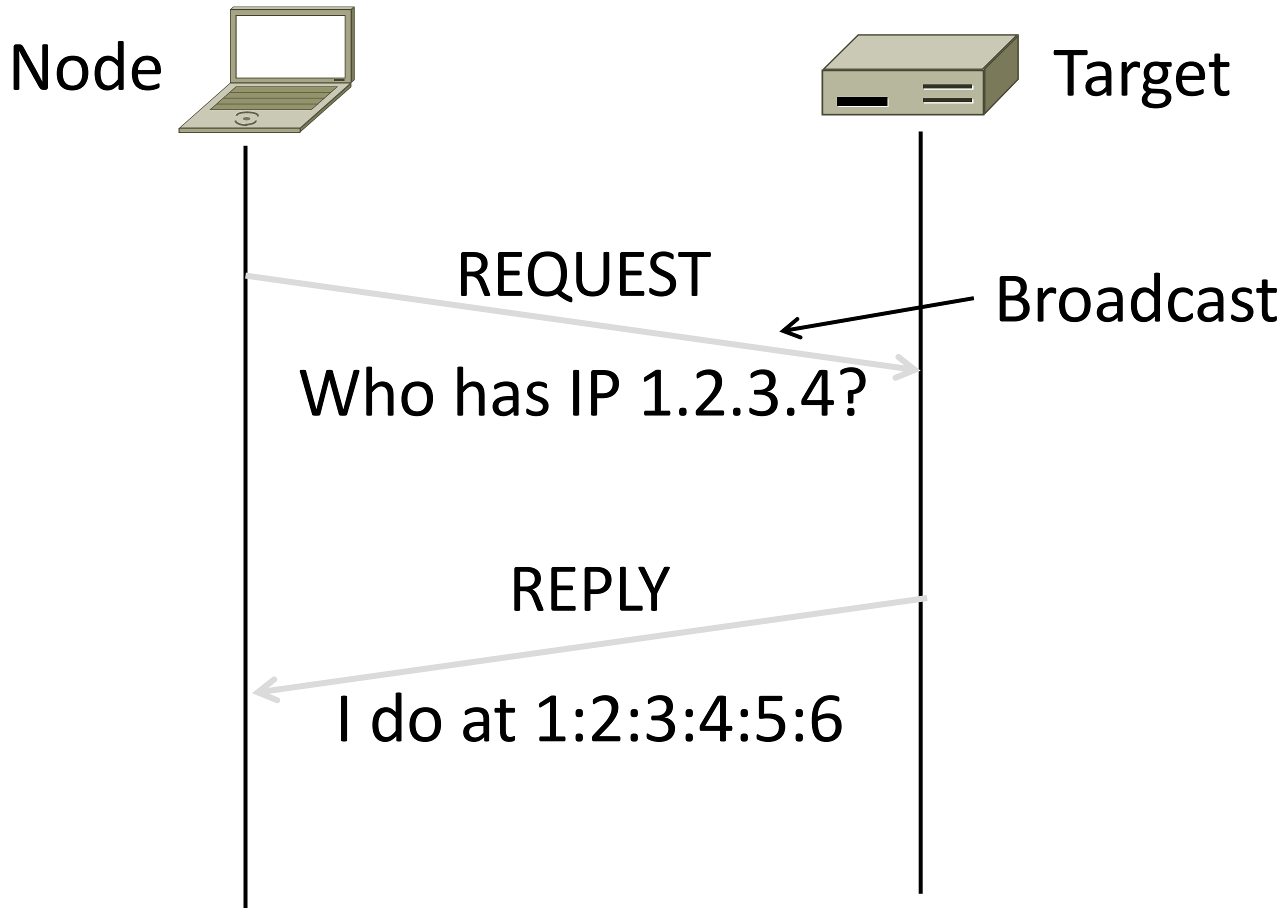


ARP Protocol Stack

- ARP sits right on top of link layer
  - No servers, just asks node with target IP to identify itself
  - Uses broadcast to reach all nodes



ARP Messages



[root@host ~]# tcpdump -lni any arp & (sleep 1; arp -d 10.0.0.254; ping -c1 -n 10.0.0.254)

listening on any, link-type LINUX_SLL (Linux cooked), capture size 96 bytes

17:58:02.155495 arp who-has 10.2.1.224 tell 10.2.1.253

17:58:02.317444 arp who-has 10.0.0.96 tell 10.0.0.253

17:58:02.370446 arp who-has 10.3.1.12 tell 10.3.1.61

ARP Table

```
[Ratuls-MacBook-Pro:19wi ratul$ arp -a | grep 192
? (192.168.88.1) at e4:8d:8c:54:0:52 on en0 ifscope [ethernet]
```

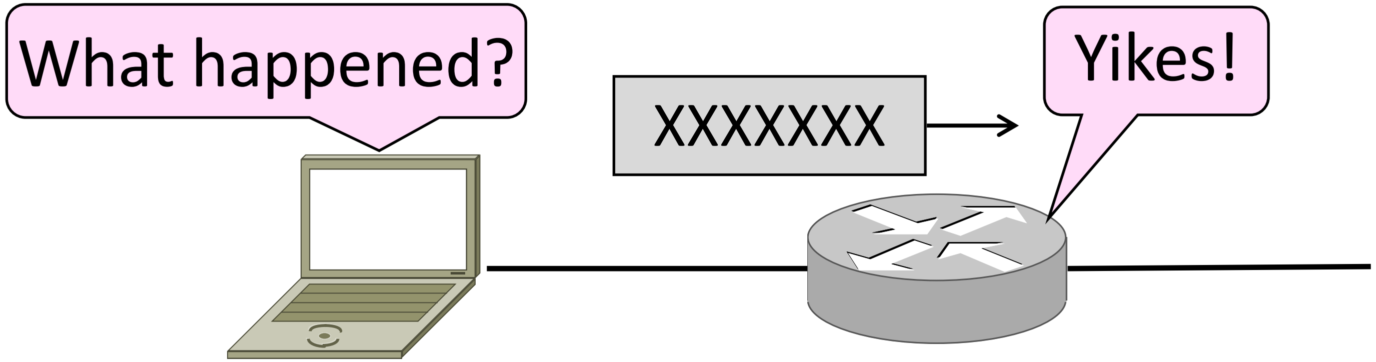
Discovery Protocols

- There are more of them!
  - Help nodes find each other and services
  - E.g., Zeroconf, Bonjour
- Often involve broadcast
  - Since nodes aren’t introduced
  - Very handy glue

Internet Control Message Protocol (ICMP)

Topic

- Problem: What happens when something goes wrong during forwarding?
  - Need to be able to find the problem



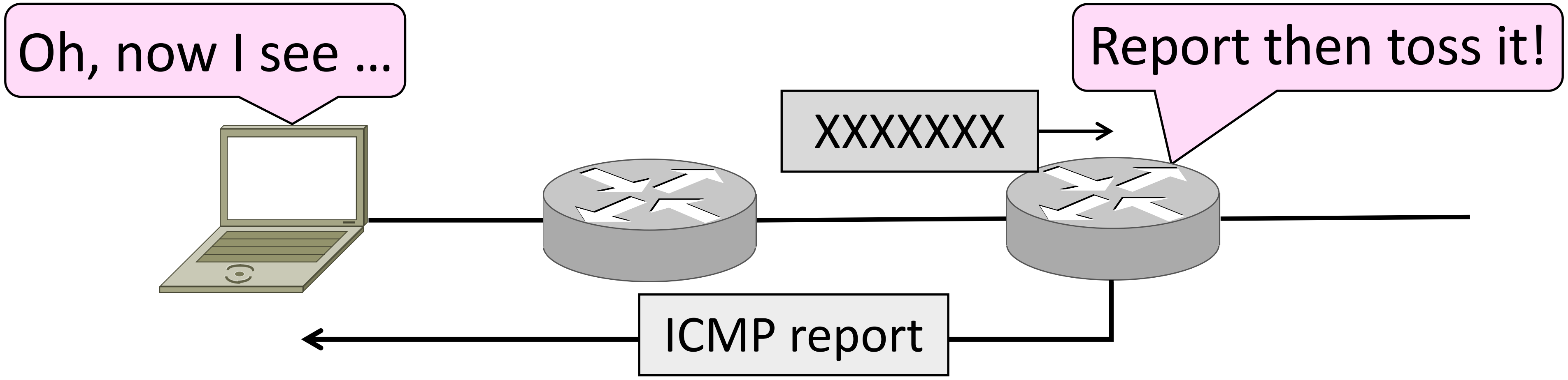
Internet Control Message Protocol

- ICMP is a companion protocol to IP
  - They are implemented together
  - Sits on top of IP (IP Protocol=1)

- Provides error report and testing
  - Error is at router while forwarding
  - Also testing that hosts can use

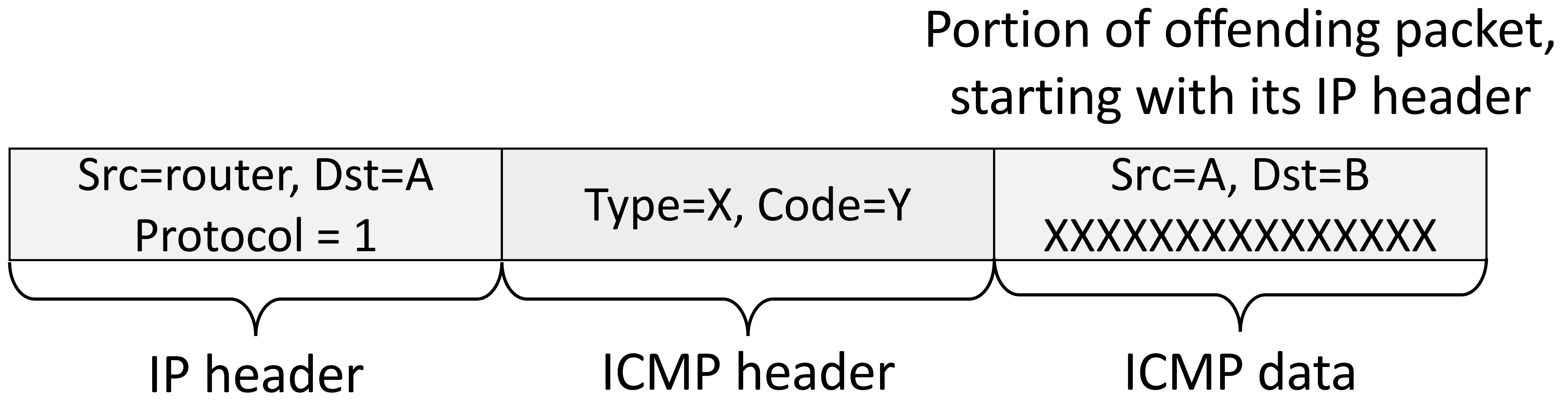
ICMP Errors

- When router encounters an error while forwarding:
  - It sends an ICMP error report back to the IP source
  - It discards the problematic packet; host needs to rectify



ICMP Message Format (2)

- Each ICMP message has a Type, Code, and Checksum
- Often carry the start of the offending packet as payload
- Each message is carried in an IP packet



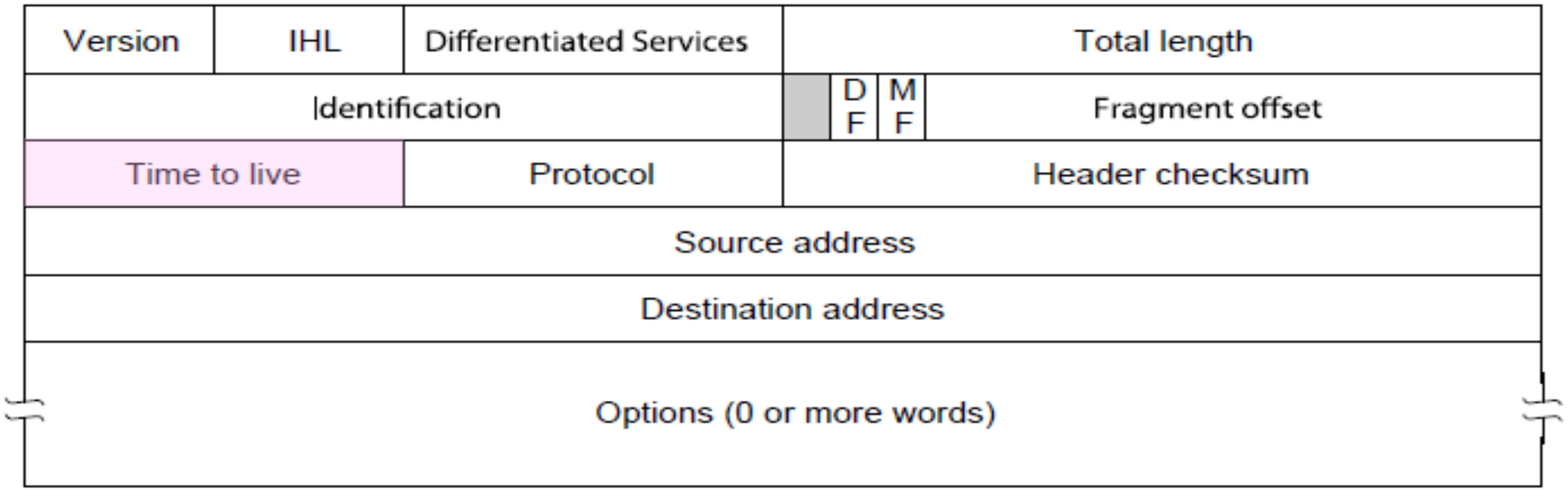
Example ICMP Messages

Name	Type / Code	Usage
Dest. Unreachable (Net or Host)	3 / 0 or 1	Lack of connectivity
Dest. Unreachable (Fragment)	3 / 4	Path MTU Discovery
Time Exceeded (Transit)	11 / 0	Traceroute
Echo Request or Reply	8 or 0 / 0	Ping

Testing, not a forwarding error: Host sends Echo Request, and destination responds with an Echo Reply

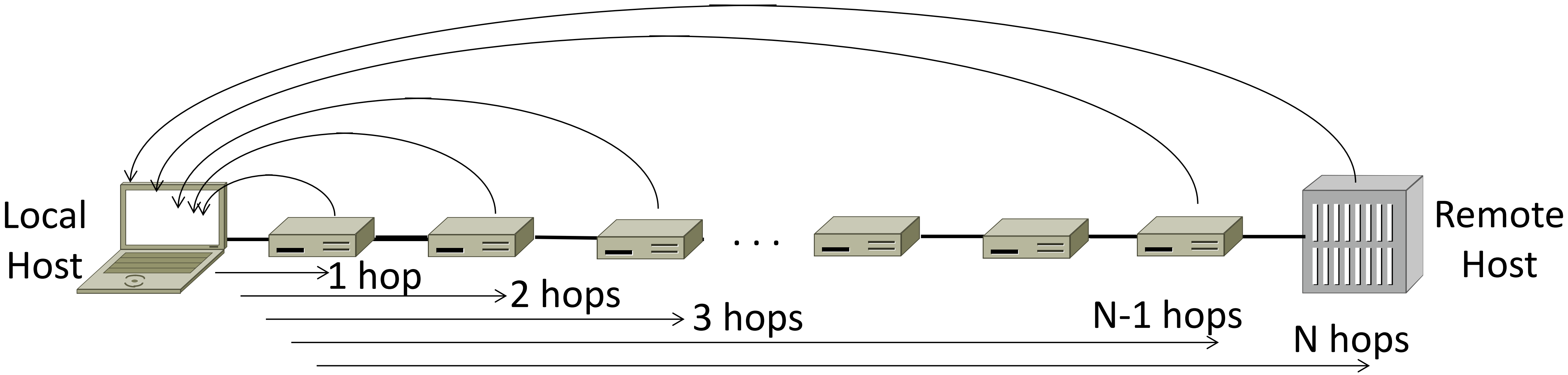
Traceroute

- IP header contains TTL (Time to live) field
  - Decremented every router hop, with ICMP error at zero
  - Protects against forwarding loops



Traceroute (2)

- Traceroute repurposes TTL and ICMP functionality
  - Sends probe packets increasing TTL starting from 1
  - ICMP errors identify routers on the path



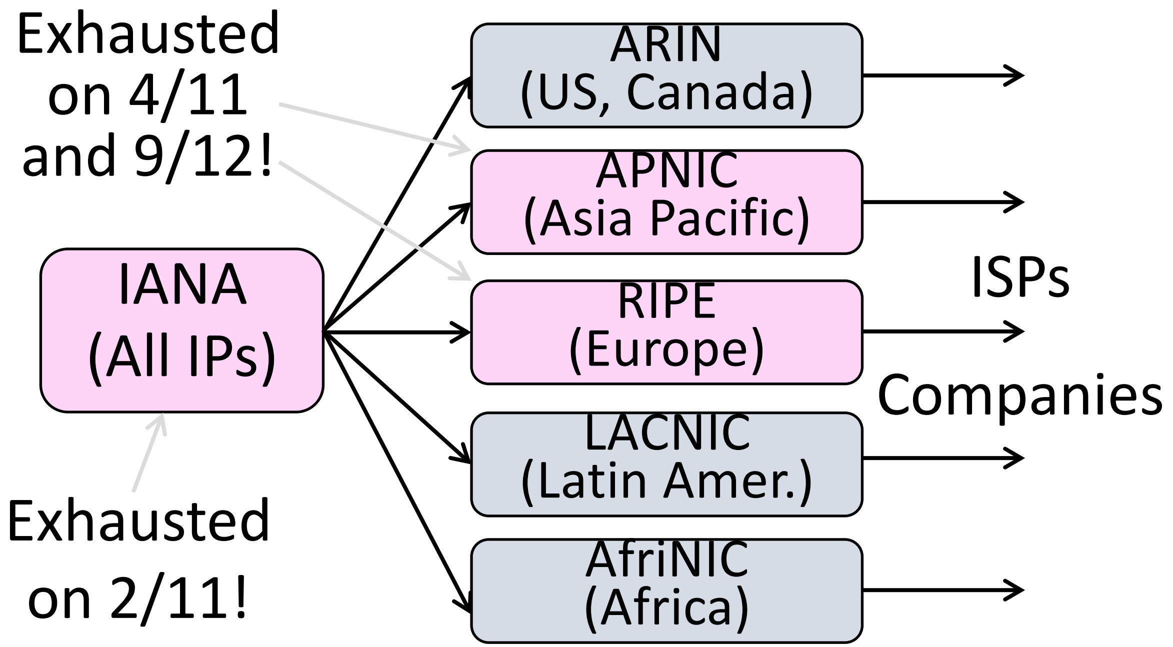
Network Address Translation (NAT)

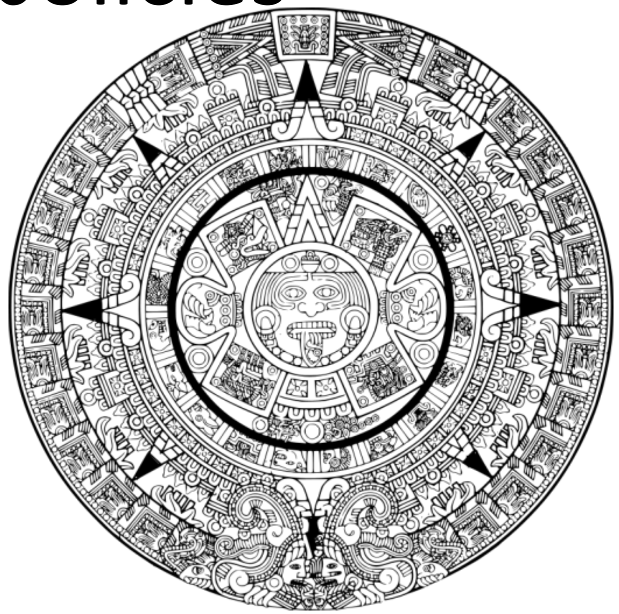
Problem: Internet Growth

- Today, Internet connects
  - 4B people
  - ~50B devices
- And we're using 32-bit addresses!
  - ~2B unique addresses

The End of New IPv4 Addresses

- Now running on leftover blocks held by the regional registries; much tighter allocation policies





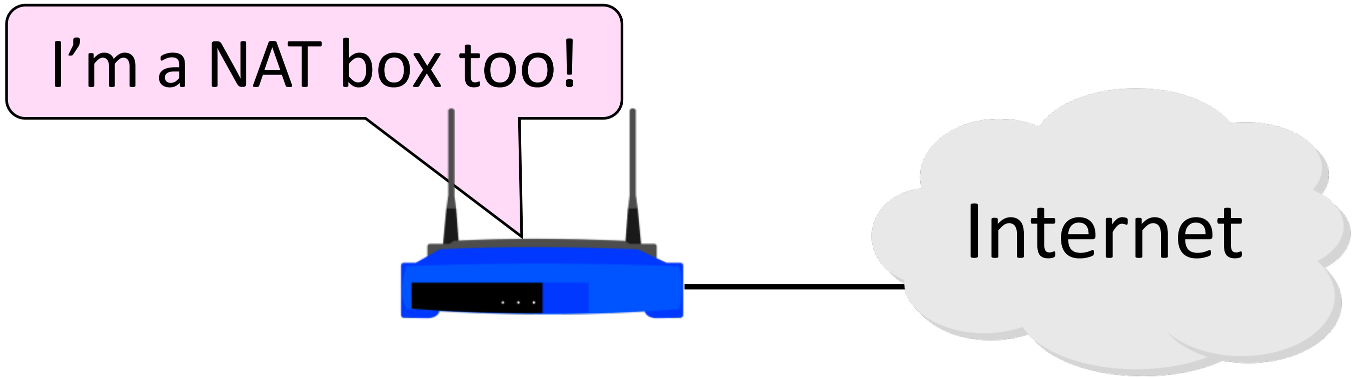
End of the world ? 12/21/12?

A market for IPv4 addresses

https://ipv4marketgroup.com/ipv4-price-trends/

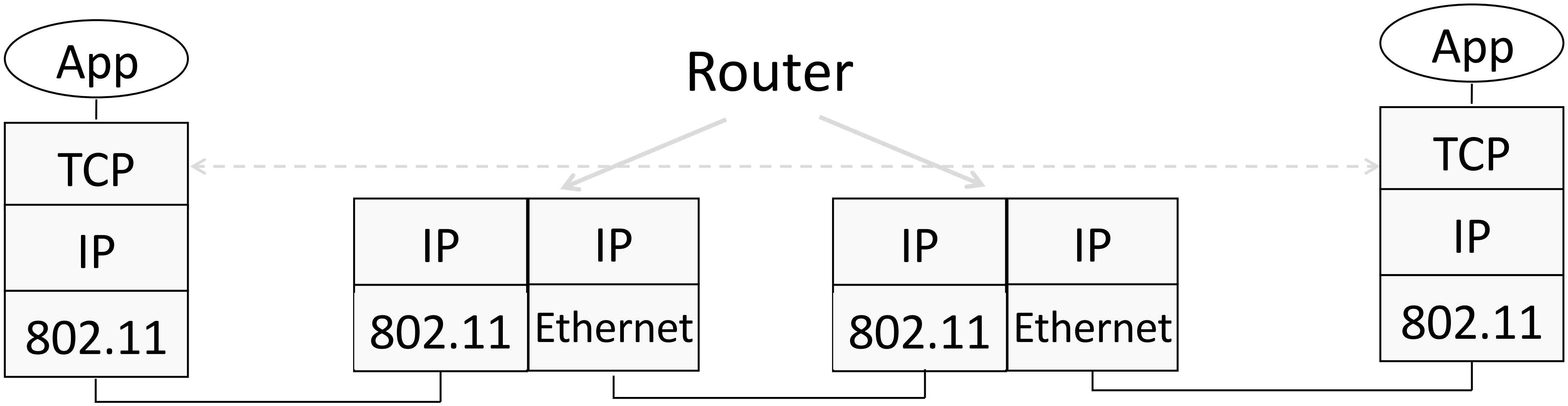
Solution 1: Network Address Translation (NAT)

- Basic idea: Map many "Private" IP addresses to one "Public" IP.
- Allocate IPs for private use (192.168.x, 10.x)



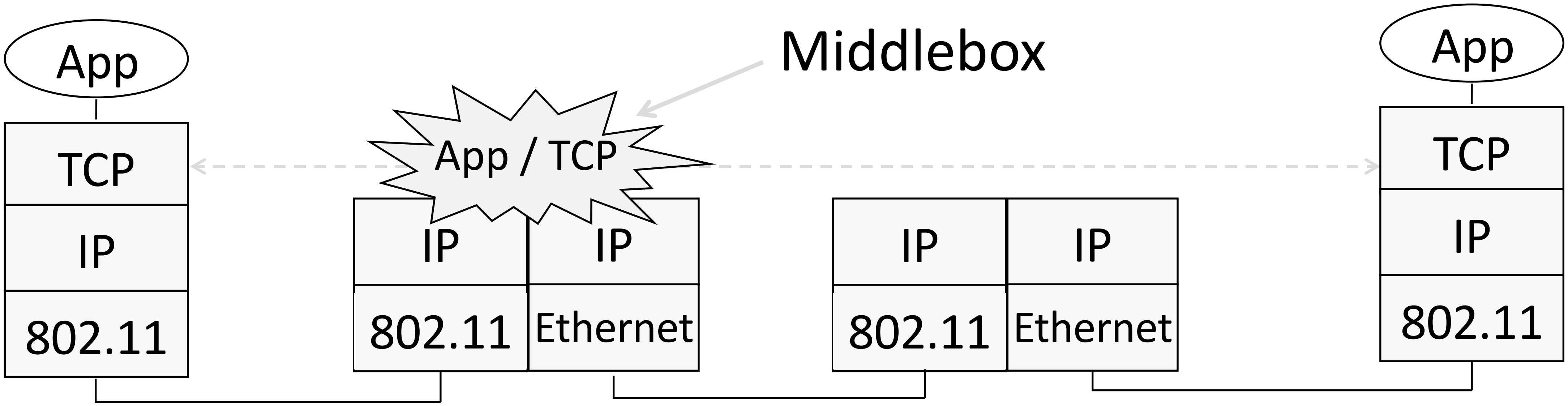
Layering Review

- Remember how layering is meant to work?
  - “Routers don’t look beyond the IP header.” Well …



Middleboxes

- Sit "inside the network" but perform "more than IP" processing on packets to add new functionality
  - NAT box, Firewall / Intrusion Detection System



Middleboxes (2)

- Advantages
  - A possible rapid deployment path when no other option
  - Control over many hosts (IT)
- Disadvantages
  - Breaking layering interferes with connectivity
    - strange side effects
  - Poor vantage point for many tasks

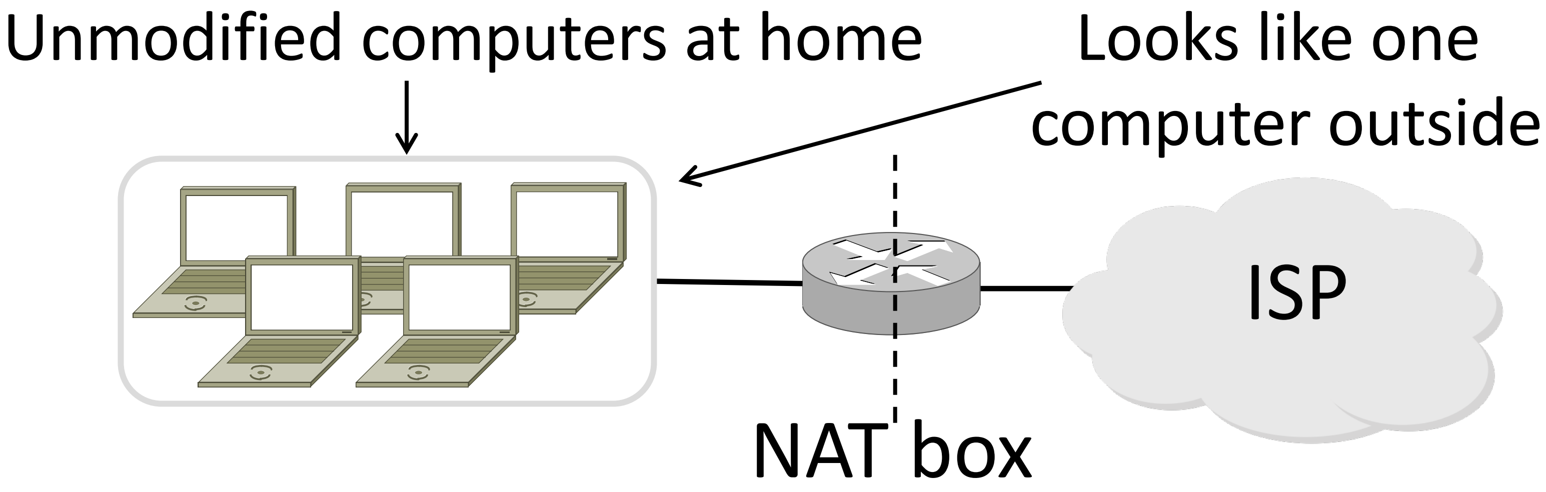
NAT (Network Address Translation) Box

- NAT box maps an internal IP to an external IP
  - Many internal hosts connected using few external addresses
  - Middlebox that “translates addresses”

- Motivated by IP address scarcity
  - Controversial at first, now accepted

NAT (2)

- Common scenario:
  - Home computers use “private” IP addresses
  - NAT (in AP/firewall) connects home to ISP using a single external IP address



How NAT Works

- Keeps an internal/external translation table
  - Typically uses IP address + TCP port
  - This is address and port translation

What host thinks	What ISP thinks
Internal IP:port	**External IP : port**
192.168.1.12 : 5523	44.25.80.3 : 1500
192.168.1.13 : 1234	44.25.80.3 : 1501
192.168.2.20 : 1234	44.25.80.3 : 1502

- Need ports to make mapping 1-1 since there are fewer external IPs

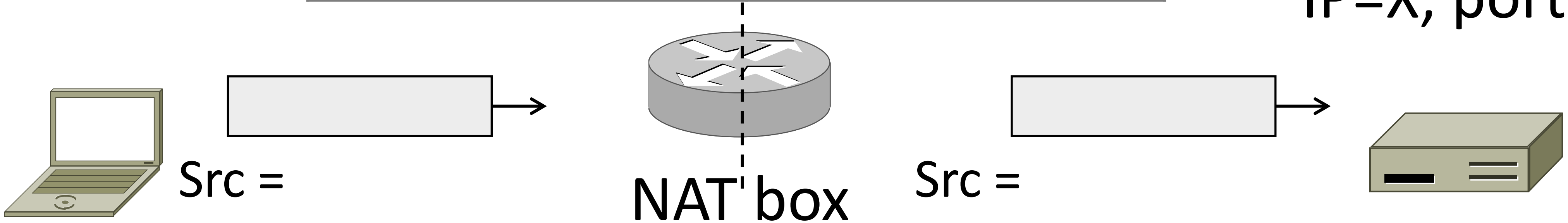
How NAT Works (2)

- Internal → External:
  - Look up and rewrite Source IP/port

Internal source

Internal IP:port	External IP : port
192.168.1.12 : 5523	44.25.80.3 : 1500

External destination IP=X, port=Y



Src =

Dst =

NAT box

Src =

Dst =

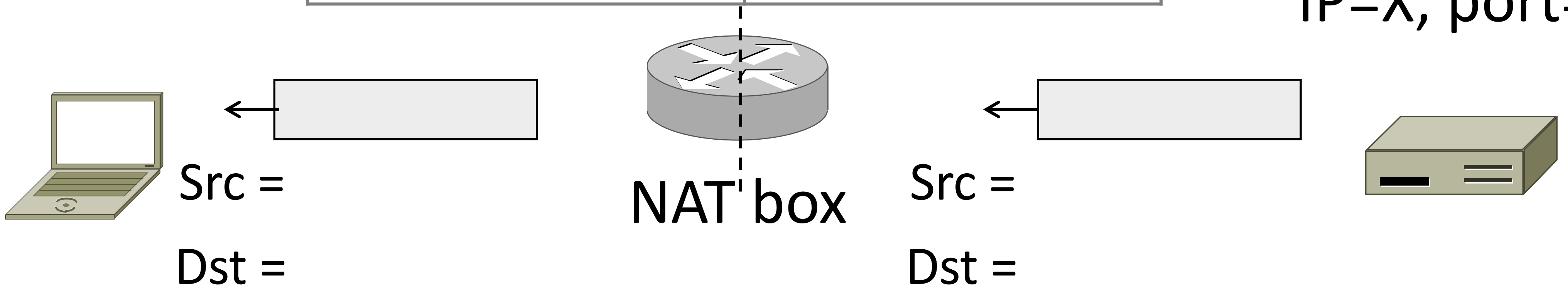
How NAT Works (3)

- External → Internal
  - Look up and rewrite Destination IP/port

Internal destination

Internal IP:port	External IP : port
192.168.1.12 : 5523	44.25.80.3 : 1500

External source IP=X, port=Y



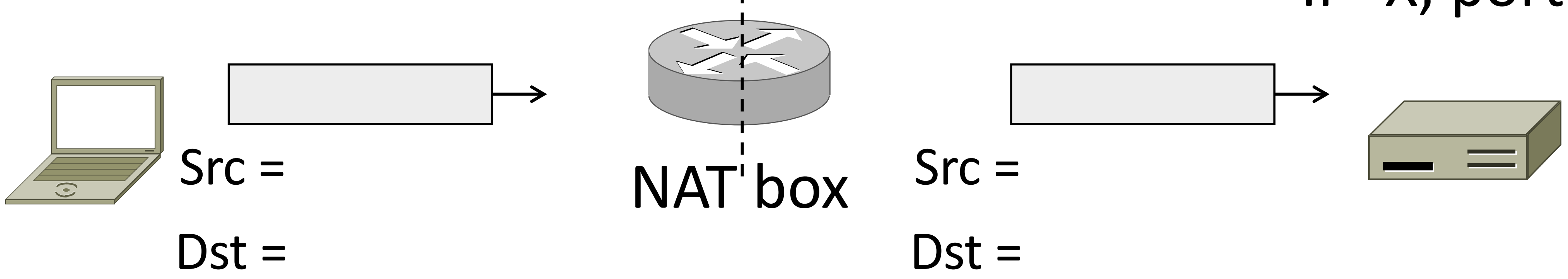
How NAT Works (4)

- Need to enter translations in the table for it to work
  - Create external name when host makes a TCP connection

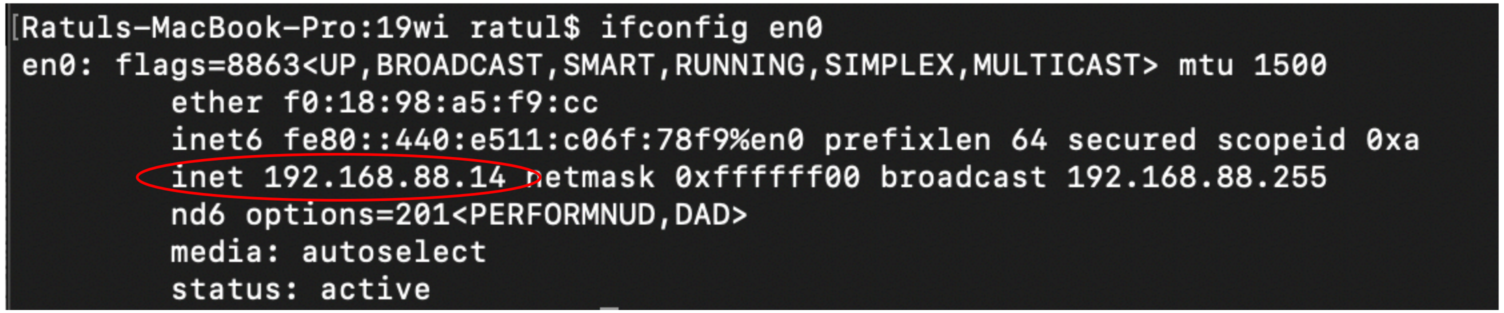
Internal source

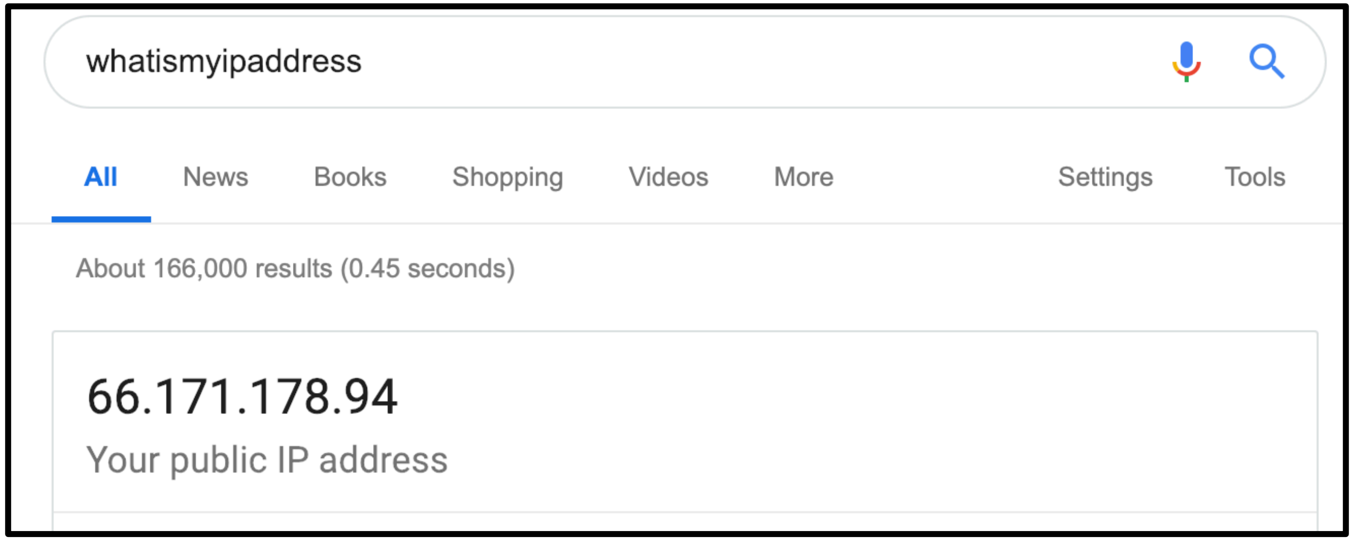
Internal IP:port	External IP : port
192.168.1.12 : 5523	

External destination
IP=X, port=Y



NAT in action





NAT Downsides

- Connectivity has been broken!
  - Can only send incoming packets after an outgoing connection is set up
  - Difficult to run servers or peer-to-peer apps (Skype)
- Doesn’t work if return traffic by passes the NAT
- Breaks apps that expose their IP addresses (FTP)

NAT Upsides

- Relieves much IP address pressure
  - Many home hosts behind NATs
- Easy to deploy
  - Rapidly, and by you alone
- Useful functionality
  - Firewall, helps with privacy
- Kinks will get worked out eventually
  - “NAT Traversal” for incoming traffic