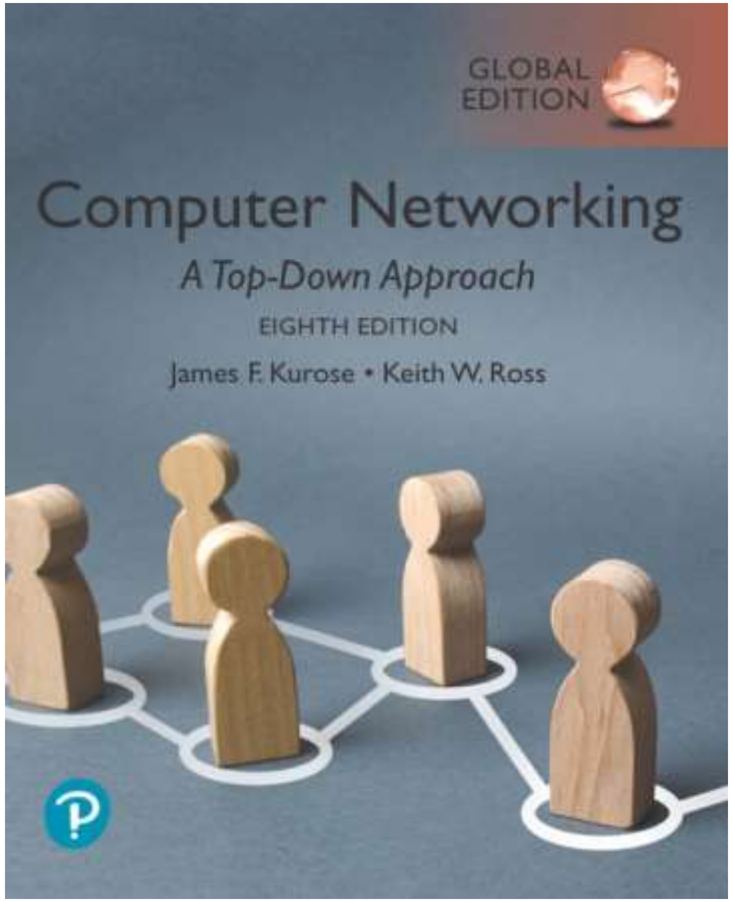
Chapter 4
Network Layer: Data Plane

Ren Ping Liu
renping.liu@uts.edu.au
adapted from textbook slides by JFK/KWR

8 April 2024



Computer Networking: A Top-Down Approach
8th Edition, Global Edition
Jim Kurose, Keith Ross
Copyright © 2022 Pearson Education Ltd

Network layer: "data plane" roadmap

4.1 Network layer: overview
- data plane
- control plane

4.2 What's inside a router
- input ports, switching, output ports
- buffer management, scheduling

4.3 IP: the Internet Protocol
- datagram format
- addressing
- network address translation
- IPv6

~~4.4 Generalized Forwarding, SDN~~
- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

IP addresses: how to get one?

That's actually two questions:

1. Q: How does a *host* get IP address within its network (host part of address)?
2. Q: How does a *network* get IP address for itself (network part of address)

IP addresses: how to get one?

That’s actually **two** questions:

1. Q: How does a *host* get IP address within its network (host part of address)?

- Manually config by sysadmin in config file
  - Windows: control-panel→network→configuration→TCP/IP
  - Apple: system preferences→networks→advanced →TCP/IP

- DHCP: **D**ynamic **H**ost **C**onfiguration **P**rotocol:
  - dynamically get address from as server
  - “plug-and-play”

DHCP: Dynamic Host Configuration Protocol

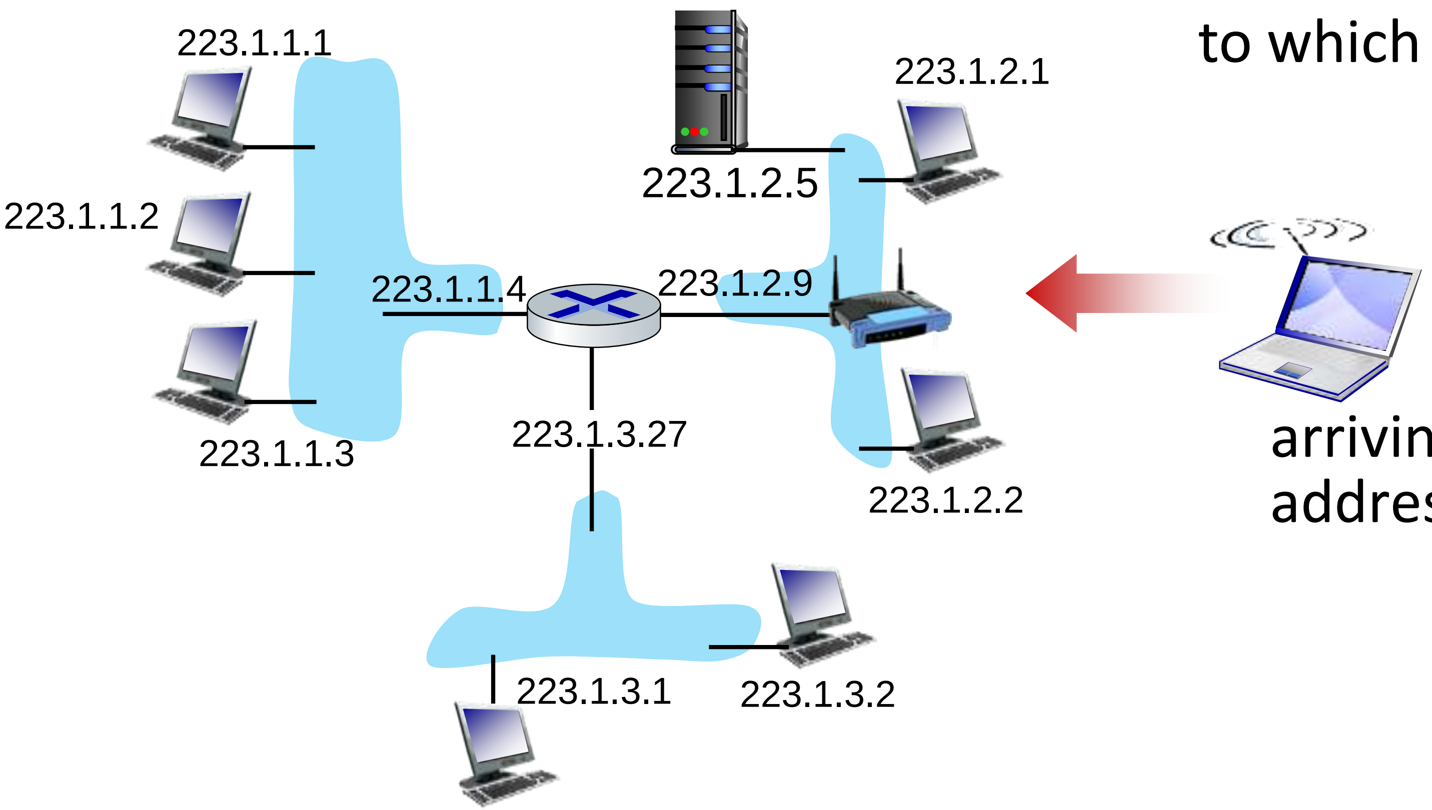
goal: host *dynamically* obtains IP address from network server when it "joins" network

- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/on)
- support for mobile users who join/leave network

DHCP overview:

- host broadcasts DHCP discover msg [optional]
- DHCP server responds with DHCP offer msg [optional]
- host requests IP address: DHCP request msg
- DHCP server sends address: DHCP ack msg

DHCP client-server scenario

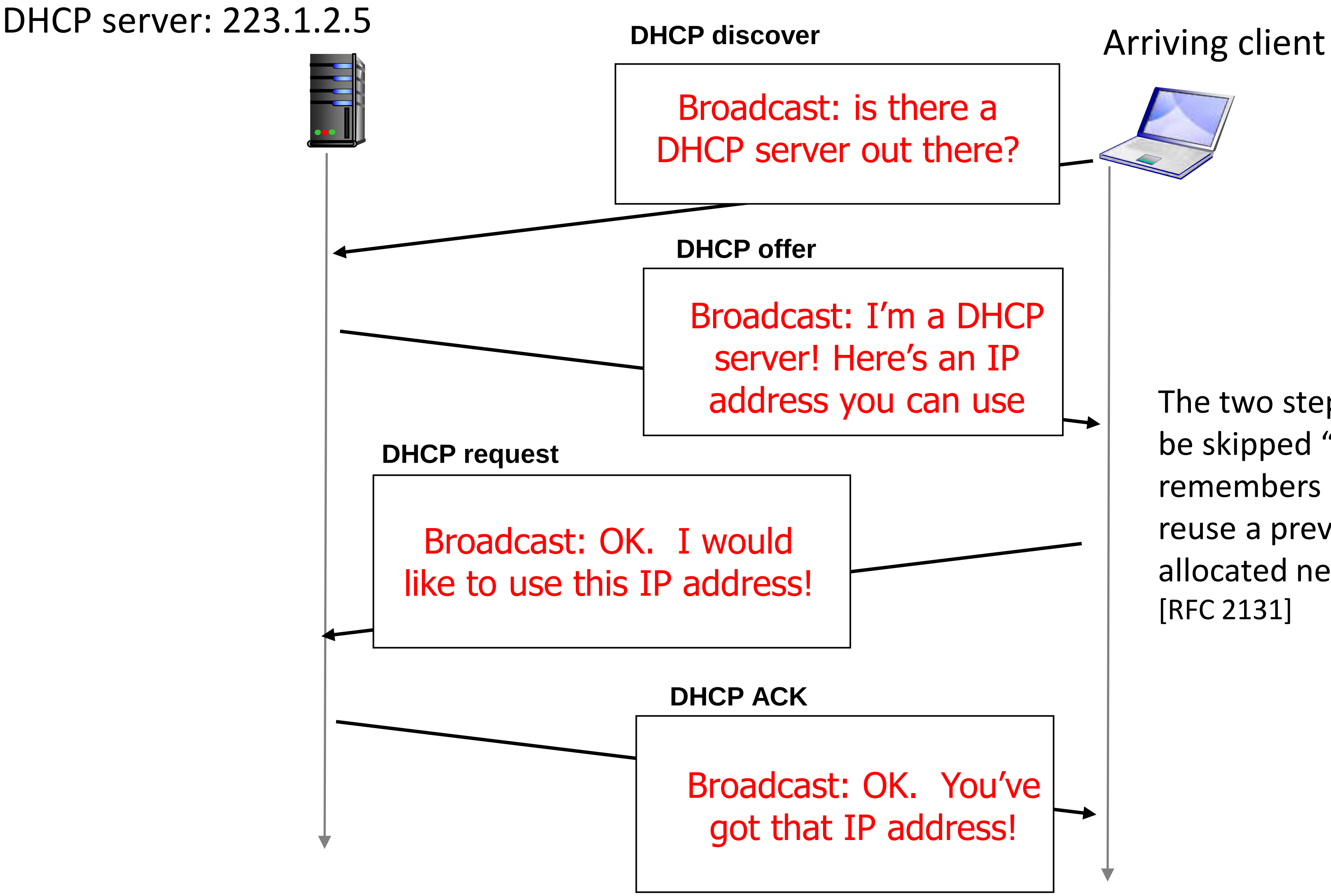


Typically, DHCP server will be co-located in router, serving all subnets to which router is attached

arriving DHCP client needs address in this network

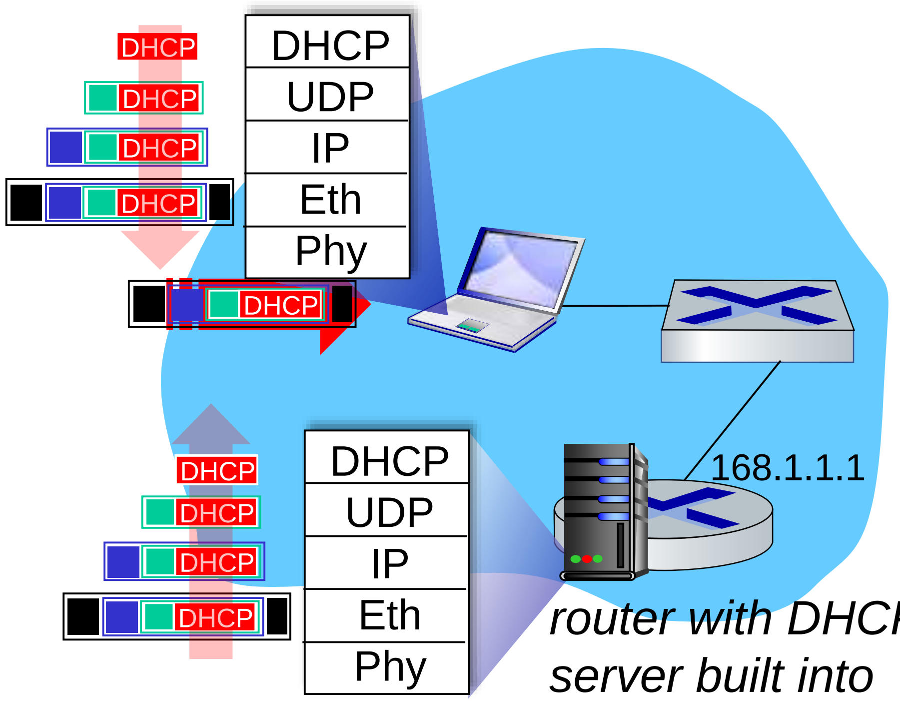
DHCP client-server scenario

255.255.255.255 (Broadcast Address)
= FF.FF.FF.FF
= 11111111.11111111.11111111.11111111



The two steps above can be skipped "if a client remembers and wishes to reuse a previously allocated network address" [RFC 2131]

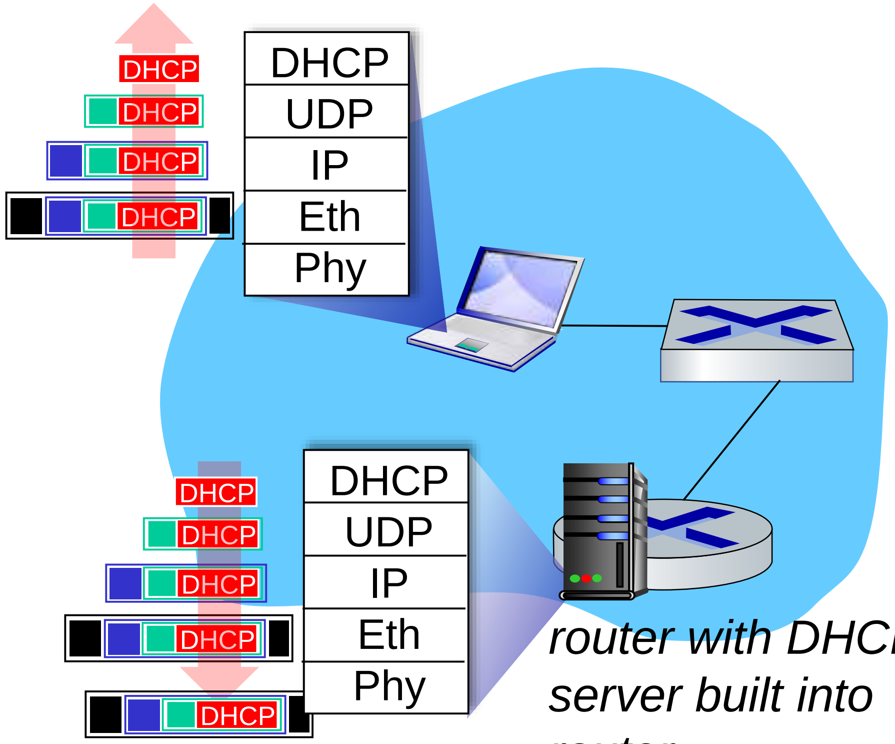
DHCP: example



router with DHCP server built into router

- Connecting laptop will use DHCP to get IP address, address of first-hop router, address of DNS server.
- DHCP REQUEST message encapsulated in UDP, encapsulated in IP, encapsulated in Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demux'ed to IP demux'ed, UDP demux'ed to DHCP

DHCP: example



router with DHCP server built into router

- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulated DHCP server reply forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DNS server, IP address of its first-hop router

DHCP: more than IP addresses

DHCP can return more than just allocated IP address on subnet:

- network mask
  - indicating network versus host portion of address
  - so given a dest IP address, I know if it is in my local area network or outside
- address of first-hop router for client
  - to access the wide Internet
- name and IP address of DNS sever
  - to resolve www.amazon.com.au into 13.224.170.88

IP addresses: how to get one?

That's actually *two* questions:

1. Q: How does a *host* get IP address within its network (host part of address)?

2. *Q:* How does a *network* get IP address for itself (network part of address)

A: gets allocated from its provider ISP's address space

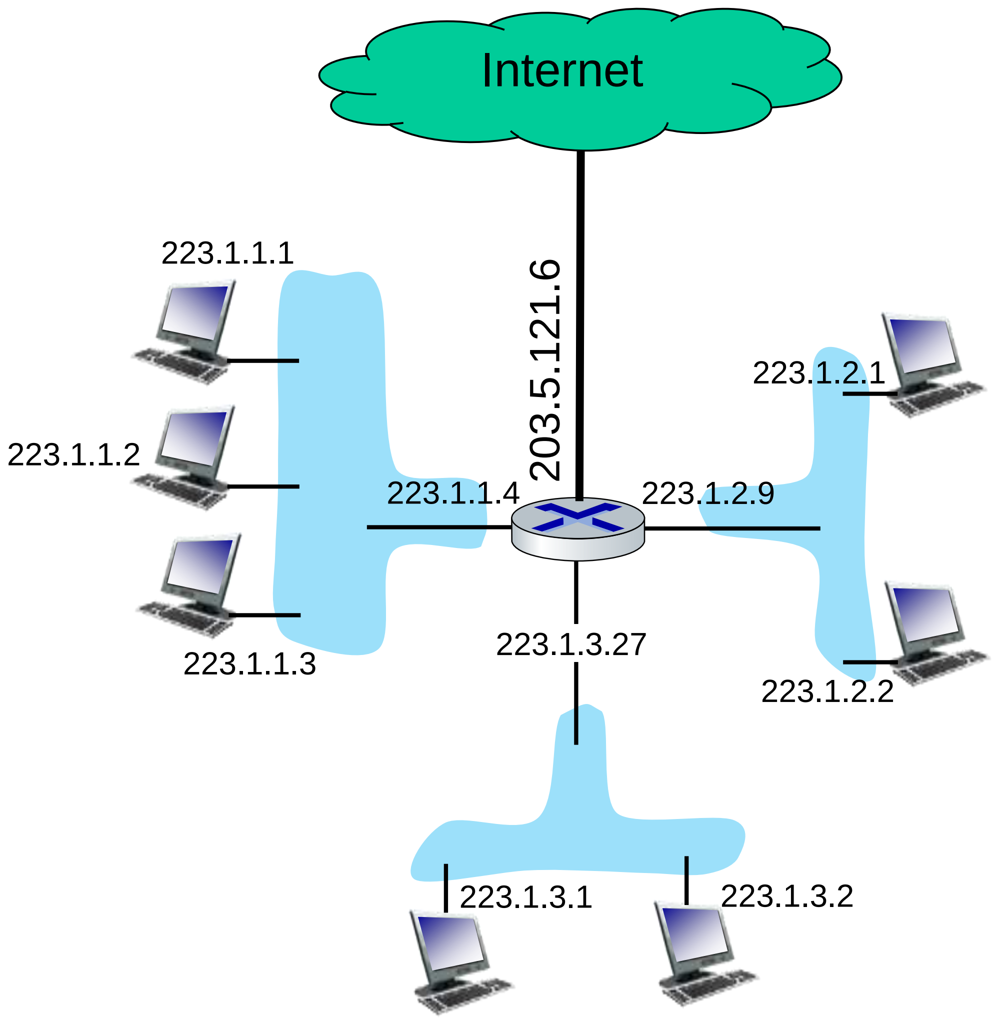
Q: how does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/

- allocates IP addresses, through 5 regional registries (RRs)

IP routing: forwarding

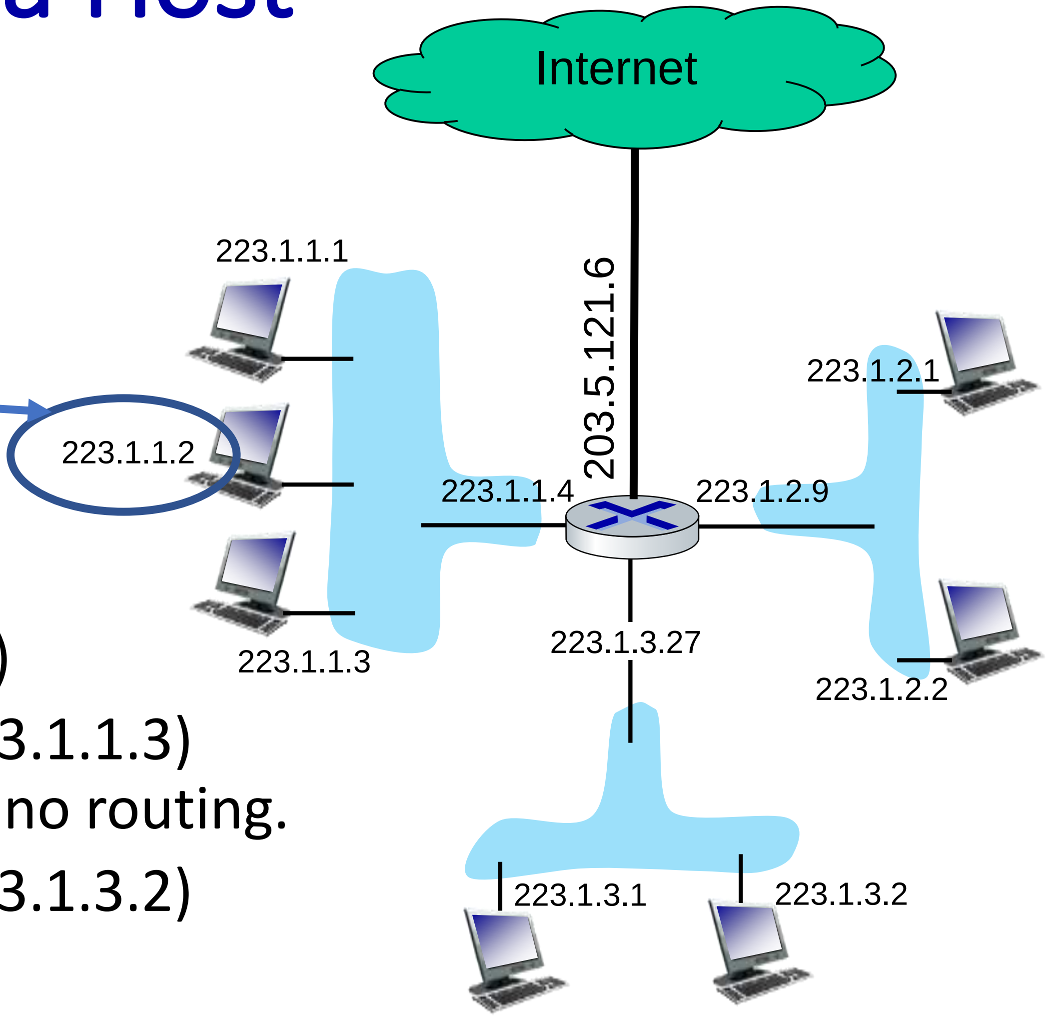
- Forwarding in a Host
- Forwarding in a Router
  - Forwarding table
  - Aggregation
  - Longest match



IP routing: forwarding in a Host

- Configure host IP address:

IP address:	223.1.1.2
Subnet mask:	255.255.255.0
Router:	223.1.1.4



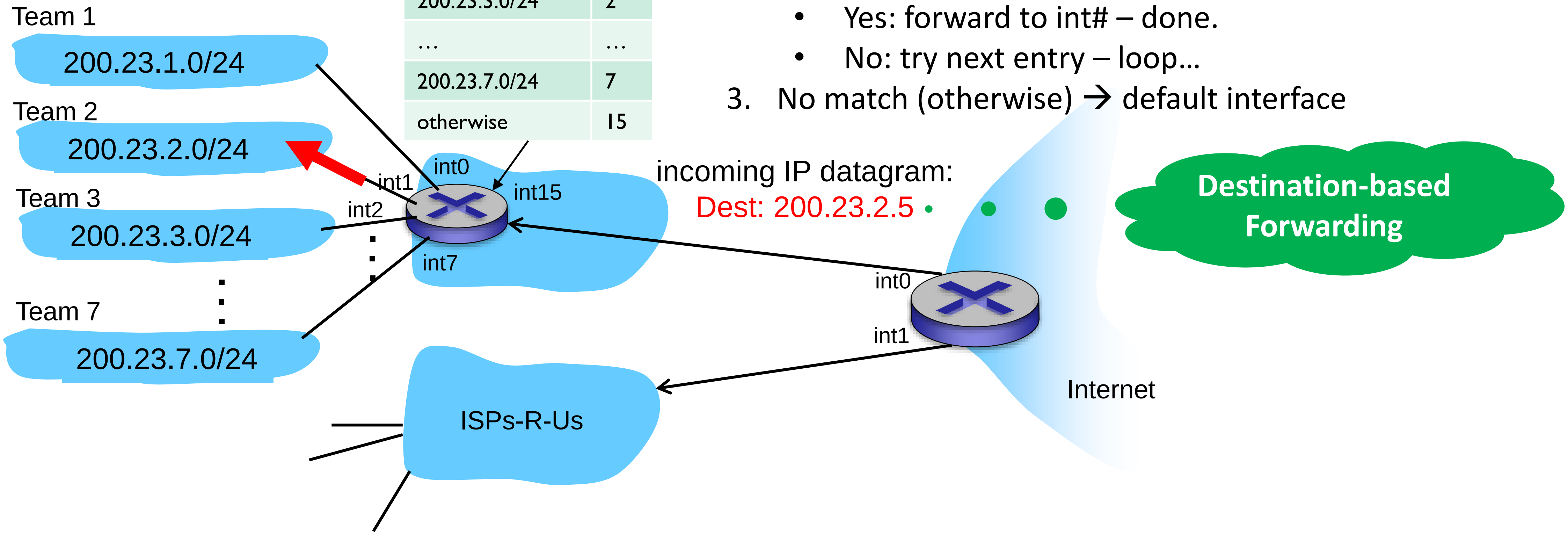
- Forwarding a datagram:
  1. subnet = (IP address) AND (subnet mask)
  2. if src.subnet == dest.subnet (dest: 223.1.1.3)
     send directly to current subnet – layer2, no routing.
  3. if src.subnet != dest.subnet (dest: 223.1.3.2)
     go via router (223.1.1.4)
  4. Router to do further routing …

IP Routing: forwarding table in a router

Forwarding Table: match IP network address to outgoing interface

Dest Addr Range	Int
200.23.1.0/24	0
200.23.2.0/24	1
200.23.3.0/24	2
…	…
200.23.7.0/24	7
otherwise	15

1. Get dest.subnet
2. Match forwarding table entry?
   - Yes: forward to int# – done.
   - No: try next entry – loop…
3. No match (otherwise) → default interface

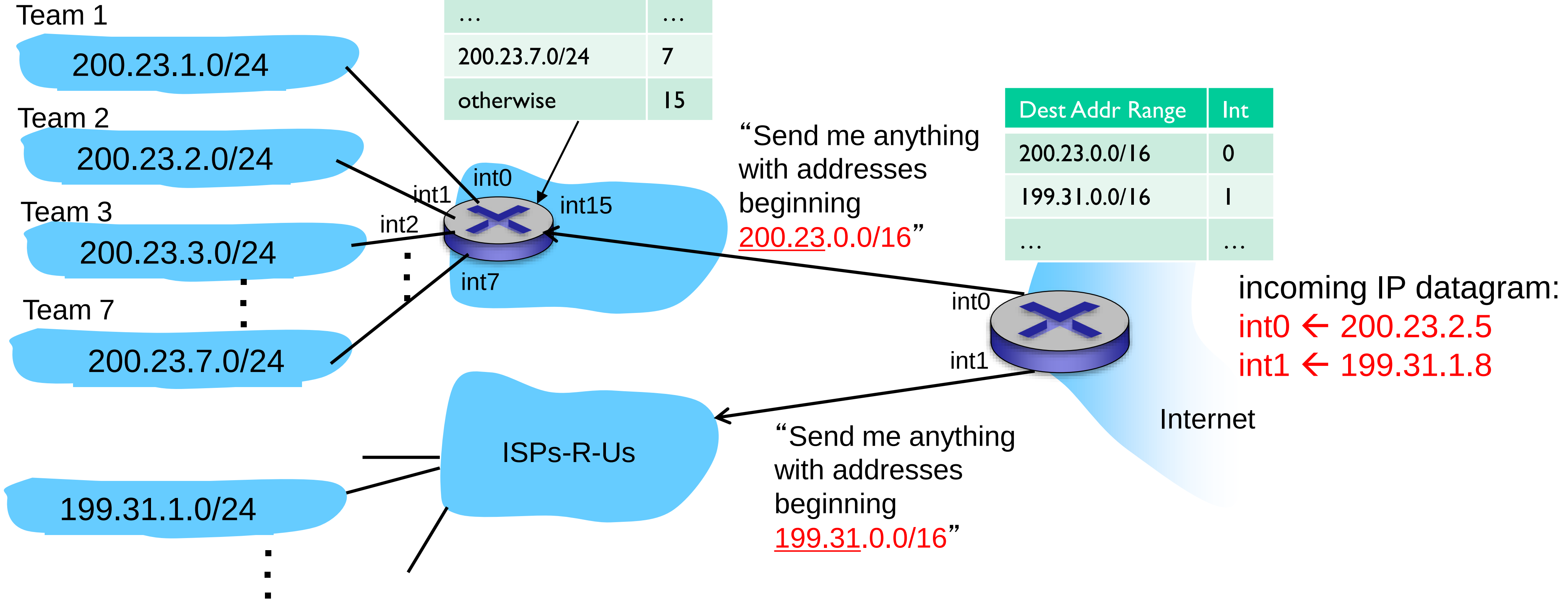


Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:

Dest Addr Range	Int
200.23.1.0/24	0
200.23.2.0/24	1
…	…
200.23.7.0/24	7
otherwise	15

Dest Addr Range	Int
200.23.0.0/16	0
199.31.0.0/16	1
…	…

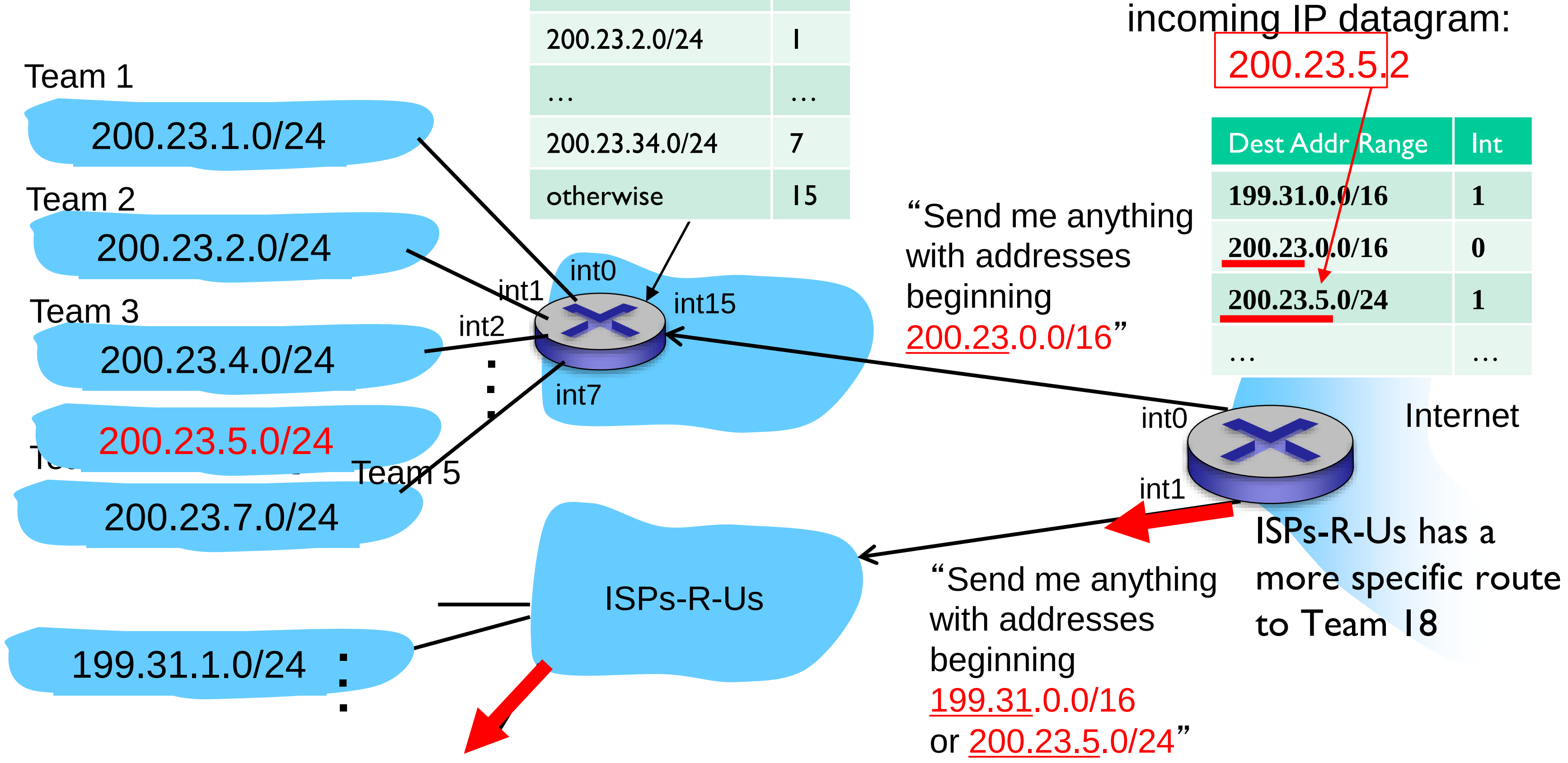


Longest prefix matching

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Dest Addr Range	Int
200.23.1.0/24	0
200.23.2.0/24	1
…	…
200.23.34.0/24	7
otherwise	15

Dest Addr Range	Int
199.31.0.0/16	1
200.23.0.0/16	0
200.23.5.0/24	1
…	…



IP addressing: last words ...

Q: are there enough 32-bit IP addresses?

- ICANN allocated last chunk of IPv4 addresses to RRs in 2011

"Who the hell knew how much address space we needed?" Vint Cerf (reflecting on decision to make IPv4 address 32 bits long)

- NAT (next) helps IPv4 address space exhaustion
- IPv6 has 128-bit address space

Mid-break

☕

- Q & A

Network layer: "data plane" roadmap

4.1 Network layer: overview
- data plane
- control plane

4.2 What's inside a router
- input ports, switching, output ports
- buffer management, scheduling

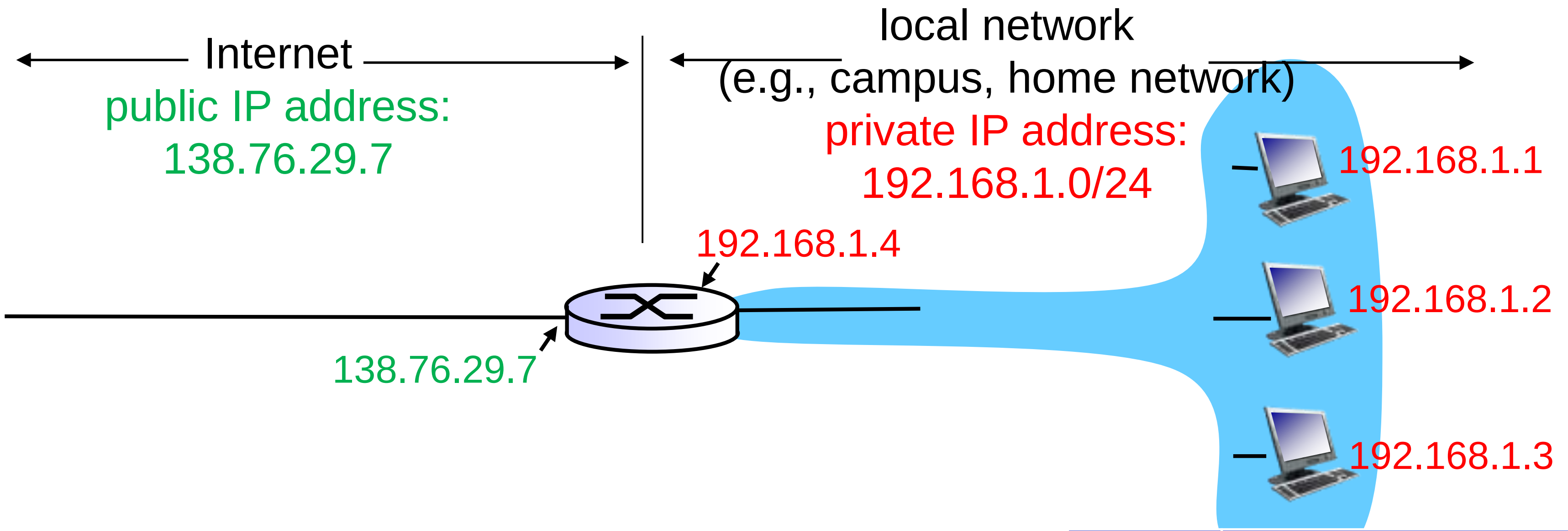
4.3 IP: the Internet Protocol
- datagram format
- addressing
- network address translation (NAT)
- IPv6

~~4.4 Generalized Forwarding, SDN~~
- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

NAT: network address translation

NAT: all devices in local network share just one public IPv4 address as far as outside world is concerned



- Private IP: free, reuse!
- Cannot go to public Internet

Private IP Address

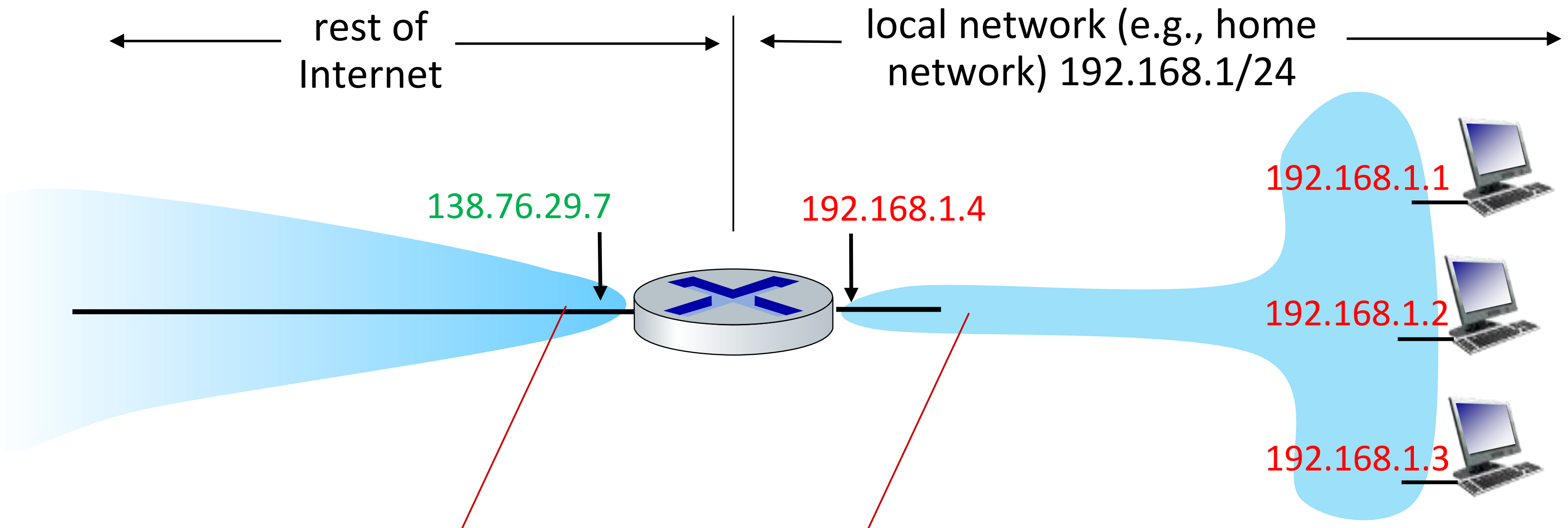
Class	Private IP address range	Subnet mask
A	10.0.0.0 - 10.255.255.255	255.0.0.0
B	172.16.0.0 - 172.16.31.255	255.255.0.0
C	192.168.0.0-192.168.255.255	255.255.255.0

NAT: network address translation

- all devices in local network have "private" IP addresses
  - 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16 prefixes
  - can only be used in local network, discarded in public routers
- advantages:
  - just one IP address needed from provider ISP for *all* devices
  - can change addresses of host in local network without notifying outside world
  - can change ISP without changing addresses of devices in local network
  - security: devices inside local net not directly addressable, invisible to outside world

NAT: network address translation

NAT: all devices in local network share just one IPv4 address as far as outside world is concerned



all datagrams *leaving* local network have *same* source NAT IP address: 138.76.29.7, but *different* source port numbers

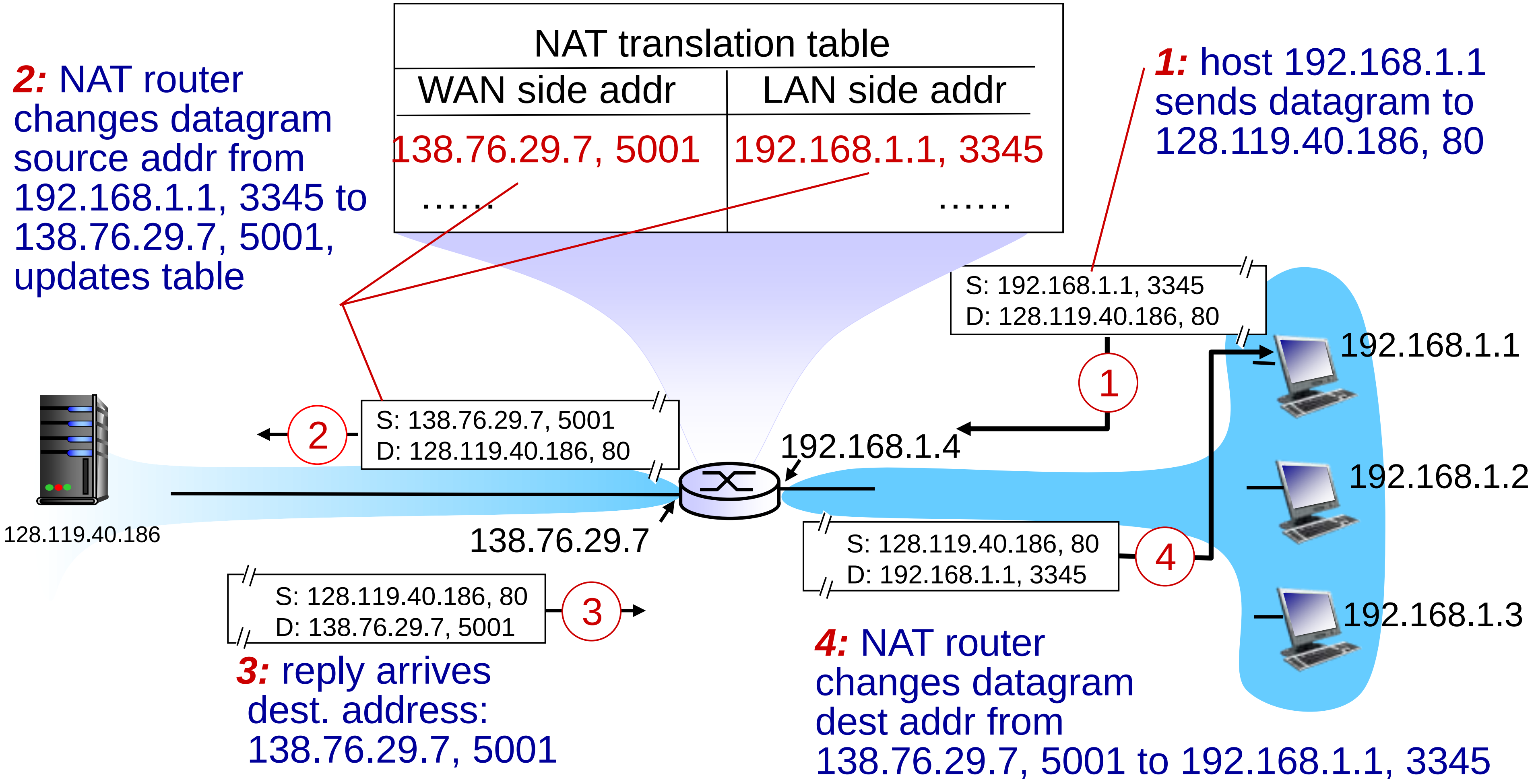
datagrams with source or destination in this network have 192.168.1/24 address for source, destination (as usual)

NAT: network address translation

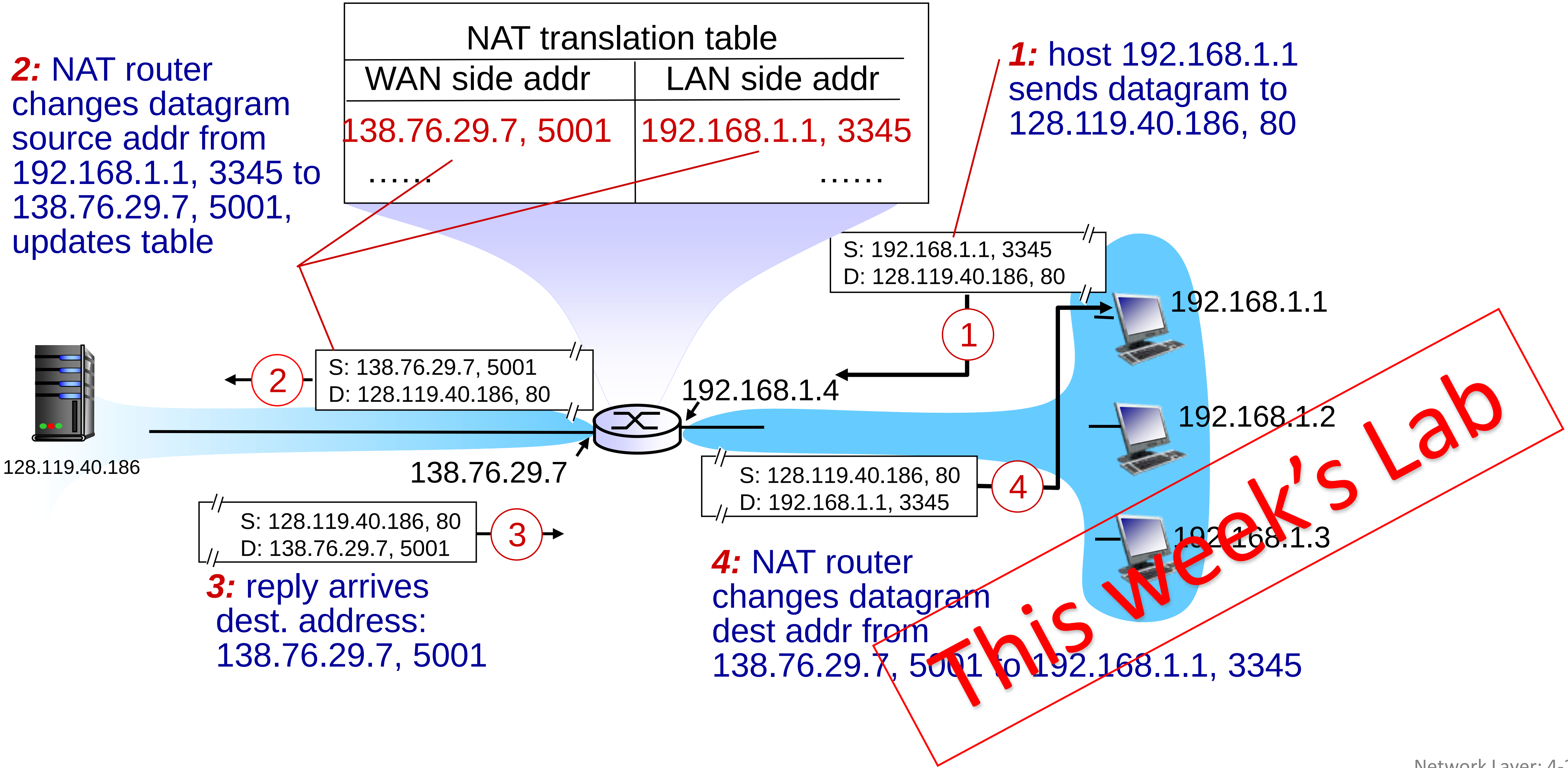
implementation: NAT router must (transparently):

- outgoing datagrams: replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
  - remote clients/servers will respond using (NAT IP address, new port #) as destination address
- remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair
- incoming datagrams: replace (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: network address translation



NAT: network address translation



NAT: network address translation

- NAT has been controversial:
  - routers "should" only process up to layer 3
  - address "shortage" should be solved by IPv6
  - violates end-to-end argument (port # manipulation by network-layer device)
  - NAT traversal: what if client wants to connect to server behind NAT?
- but NAT is here to stay:
  - extensively used in home and institutional nets, 4G/5G cellular nets

IPv6: motivation

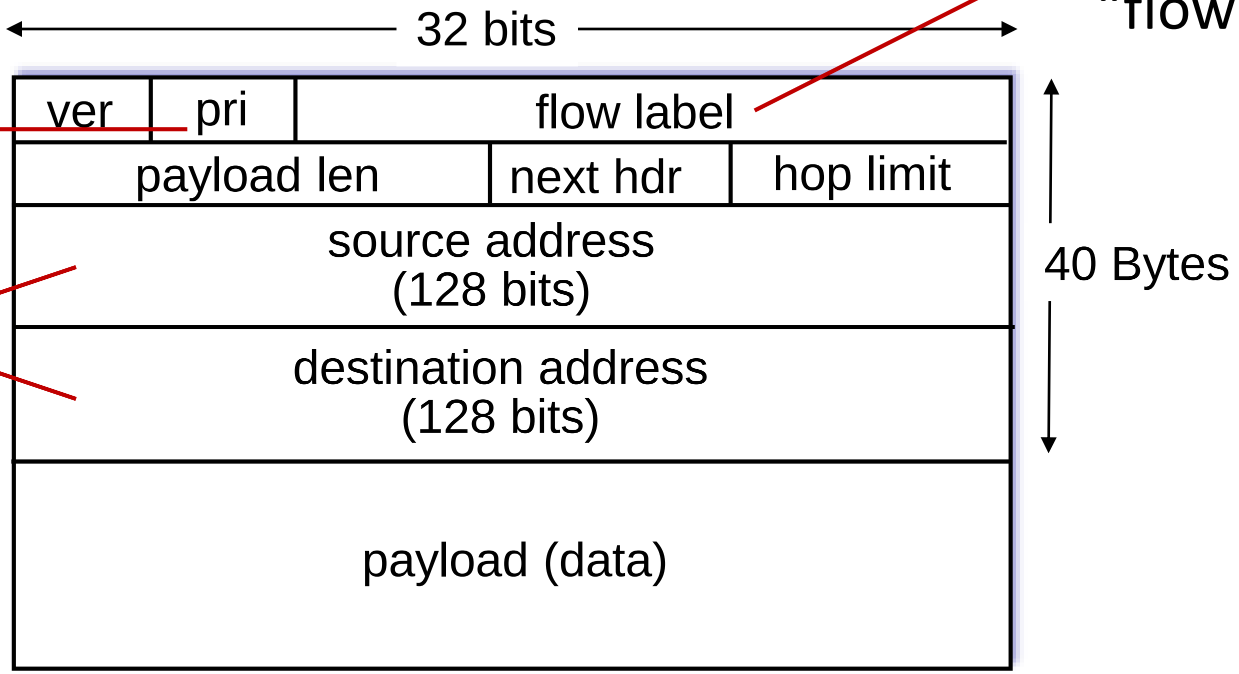
- initial motivation: 32-bit IPv4 address space would be completely allocated
- additional motivation:
  - speed processing/forwarding: 40-byte fixed length header
  - enable different network-layer treatment of "flows"

IPv6 datagram format

flow label: identify datagrams in same "flow." (concept of "flow" not well defined).

priority: identify priority among datagrams in flow

128-bit IPv6 addresses

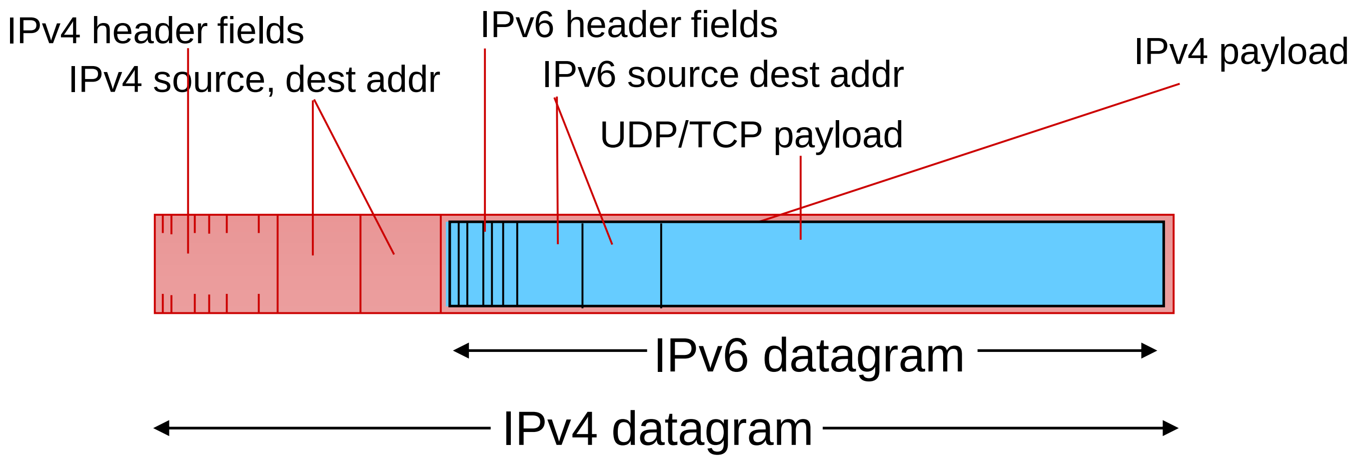


What's missing (compared with IPv4):

- no checksum (to speed processing at routers)
- no fragmentation/reassembly
- no options (available as upper-layer, next-header protocol at router)

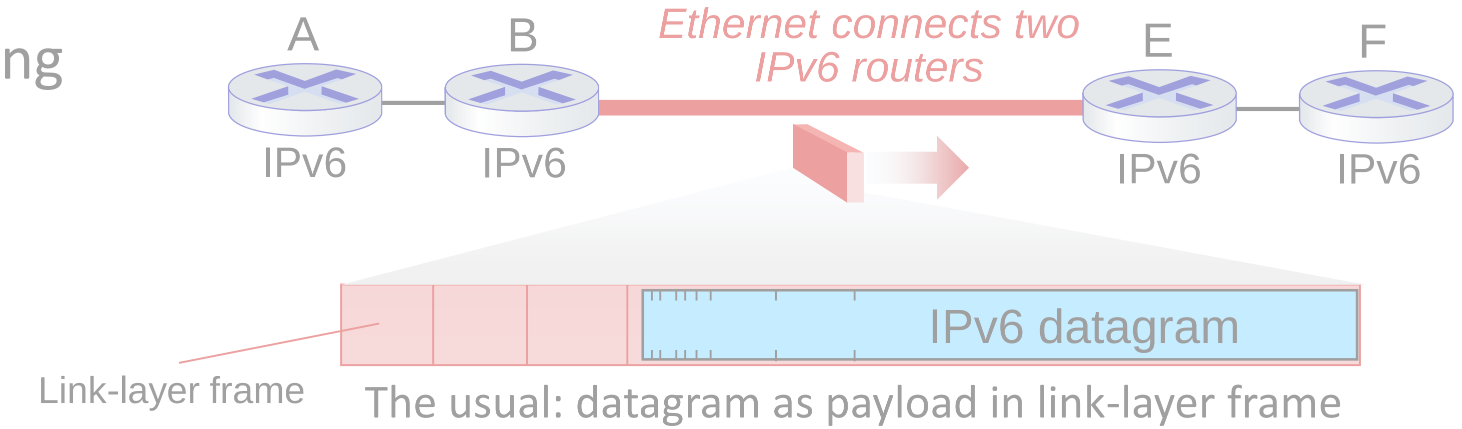
Transition from IPv4 to IPv6

- not all routers can be upgraded simultaneously
  - no "flag days"
  - how will network operate with mixed IPv4 and IPv6 routers?
- tunneling: IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers ("packet within a packet")
  - tunneling used extensively in other contexts (4G/5G)



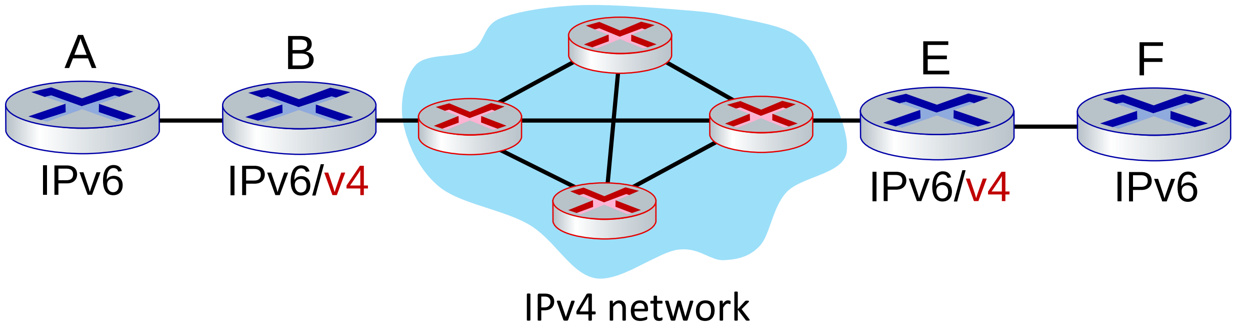
Tunneling and encapsulation

Ethernet connecting two IPv6 routers:



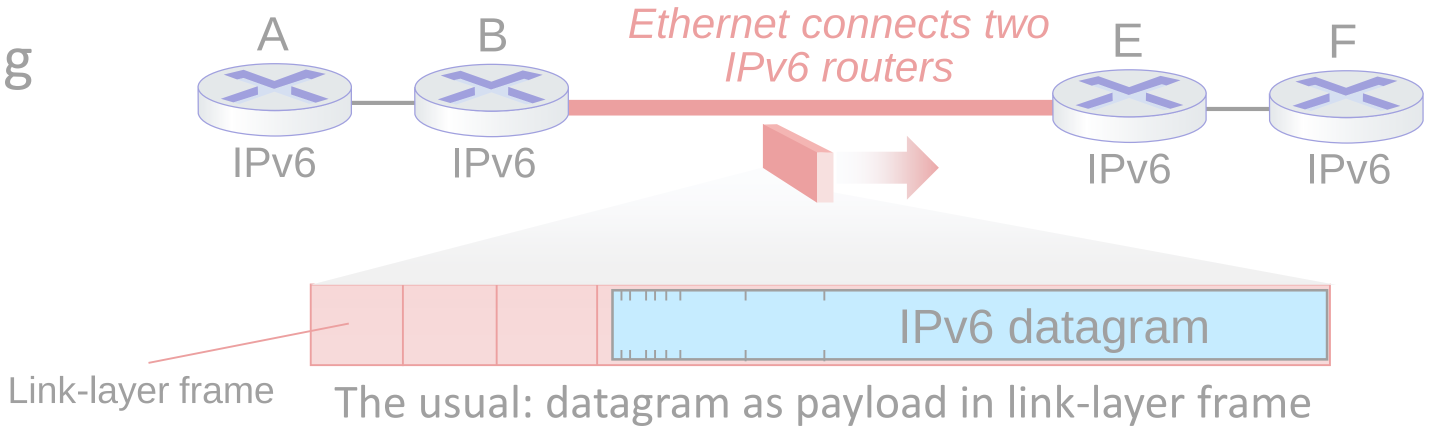
The usual: datagram as payload in link-layer frame

IPv4 network connecting two IPv6 routers



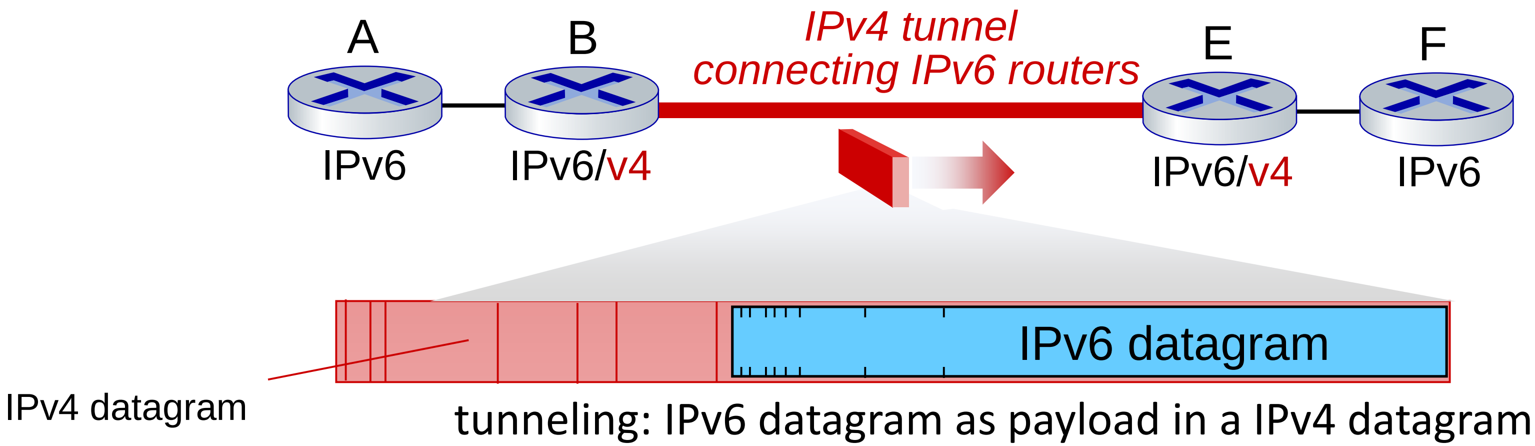
Tunneling and encapsulation

Ethernet connecting two IPv6 routers:



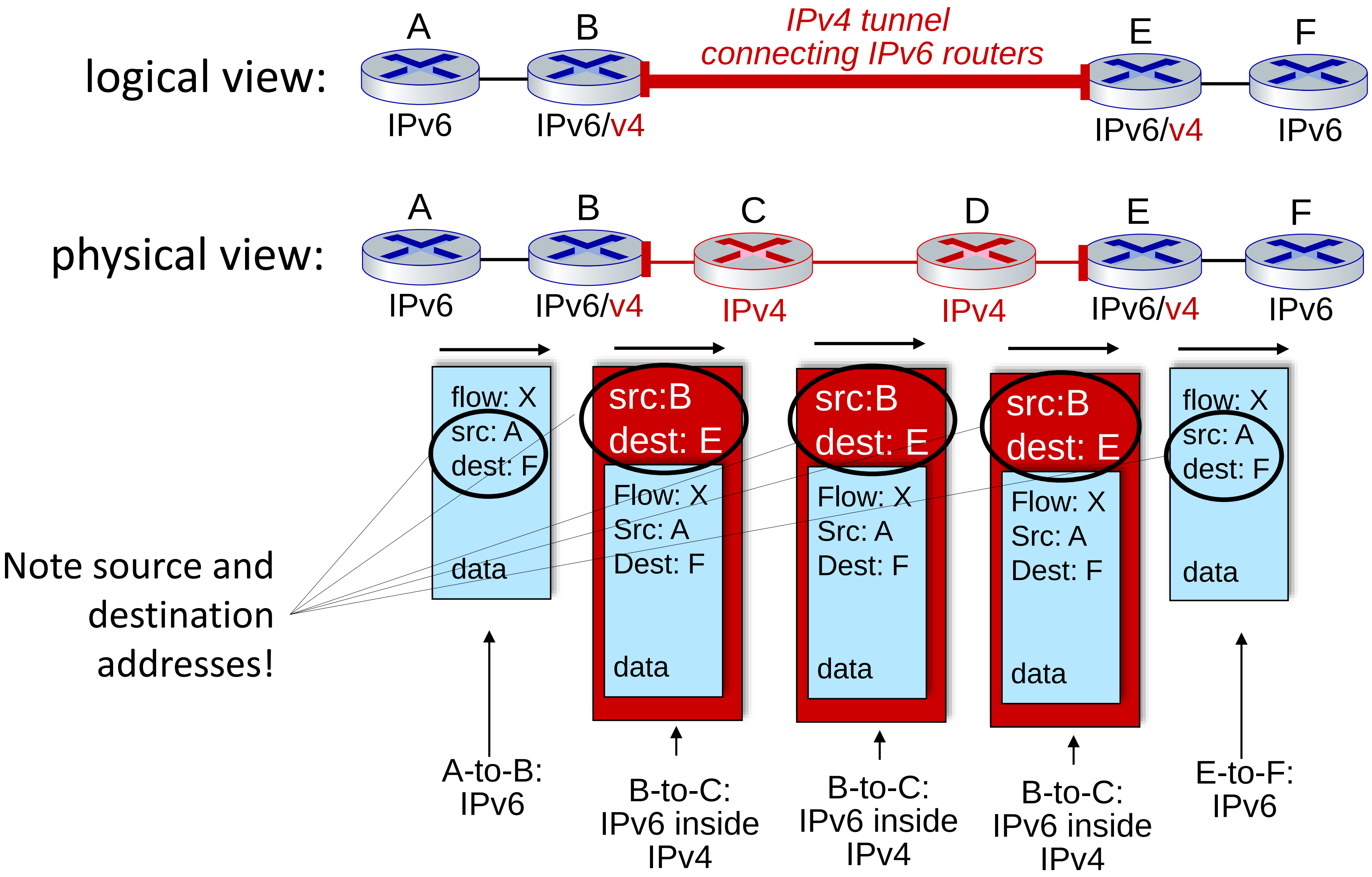
The usual: datagram as payload in link-layer frame

IPv4 tunnel connecting two IPv6 routers



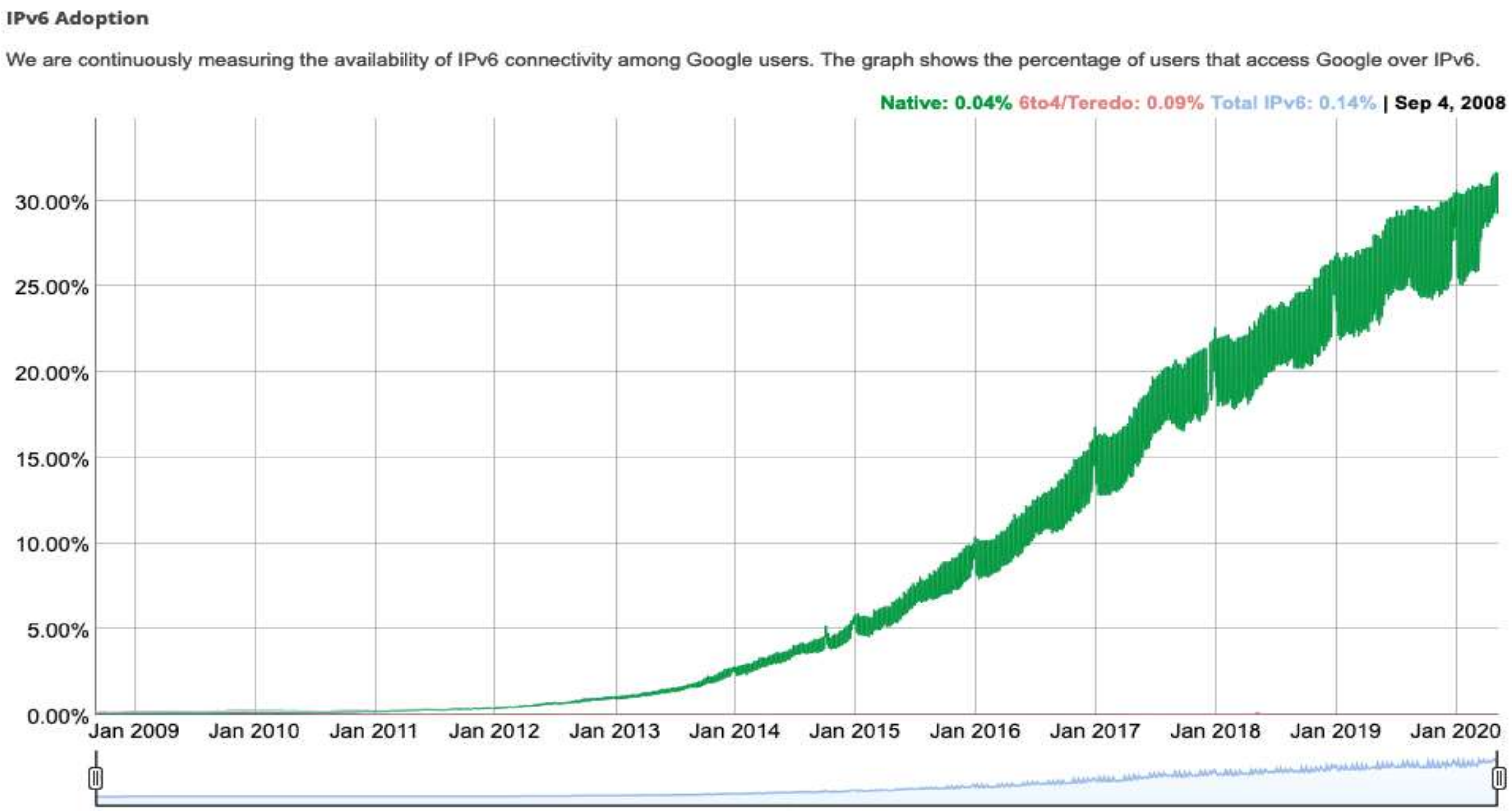
tunneling: IPv6 datagram as payload in a IPv4 datagram

Tunneling


logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6/v4
IPv6/v4
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6/v4
IPv4
IPv4
IPv6/v4
IPv6
flow: X
src: A
dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
flow: X
src: A
dest: F
data
Note source and
destination
addresses!
A-to-B:
IPv6
B-to-C:
IPv6 inside
IPv4
B-to-C:
IPv6 inside
IPv4
B-to-C:
IPv6 inside
IPv4
E-to-F:
IPv6

IPv6: adoption

- Google[1]: ~ 30% of clients access services via IPv6
- NIST: 1/3 of all US government domains are IPv6 capable



1 https://www.google.com/intl/en/ipv6/statistics.html

IPv6: adoption

- Google[1]: ~ 30% of clients access services via IPv6
- NIST: 1/3 of all US government domains are IPv6 capable
- Long (long!) time for deployment and use
  - 25 years and counting!
  - think of application-level changes in last 25 years: WWW, social media, streaming media, gaming, telepresence, …
  - *Why?*

[1] https://www.google.com/intl/en/ipv6/statistics.html

Chapter 4: done!

- Network layer: overview
- What's inside a router
- IP: the Internet Protocol
- ~~Generalized Forwarding, SDN~~
- ~~Middleboxes~~

Question: Where do we get the forwarding tables?

Answer: by the control plane (next chapter)

Lecture done ✓

- Q & A