COMP 3331/9331: Computer Networks and Applications

Week 7

Network Layer: Data Plane

Reading Guide: Chapter 4: Sections 4.1, 4.3

Network Layer, data plane: outline

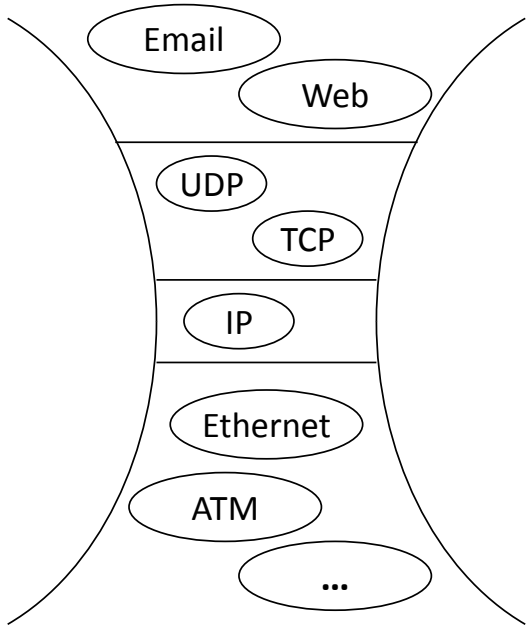
4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router
-- **Not Covered**

4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

4.4 Generalized forwarding and Software Defined Networking (SDN)
- **Not Covered**

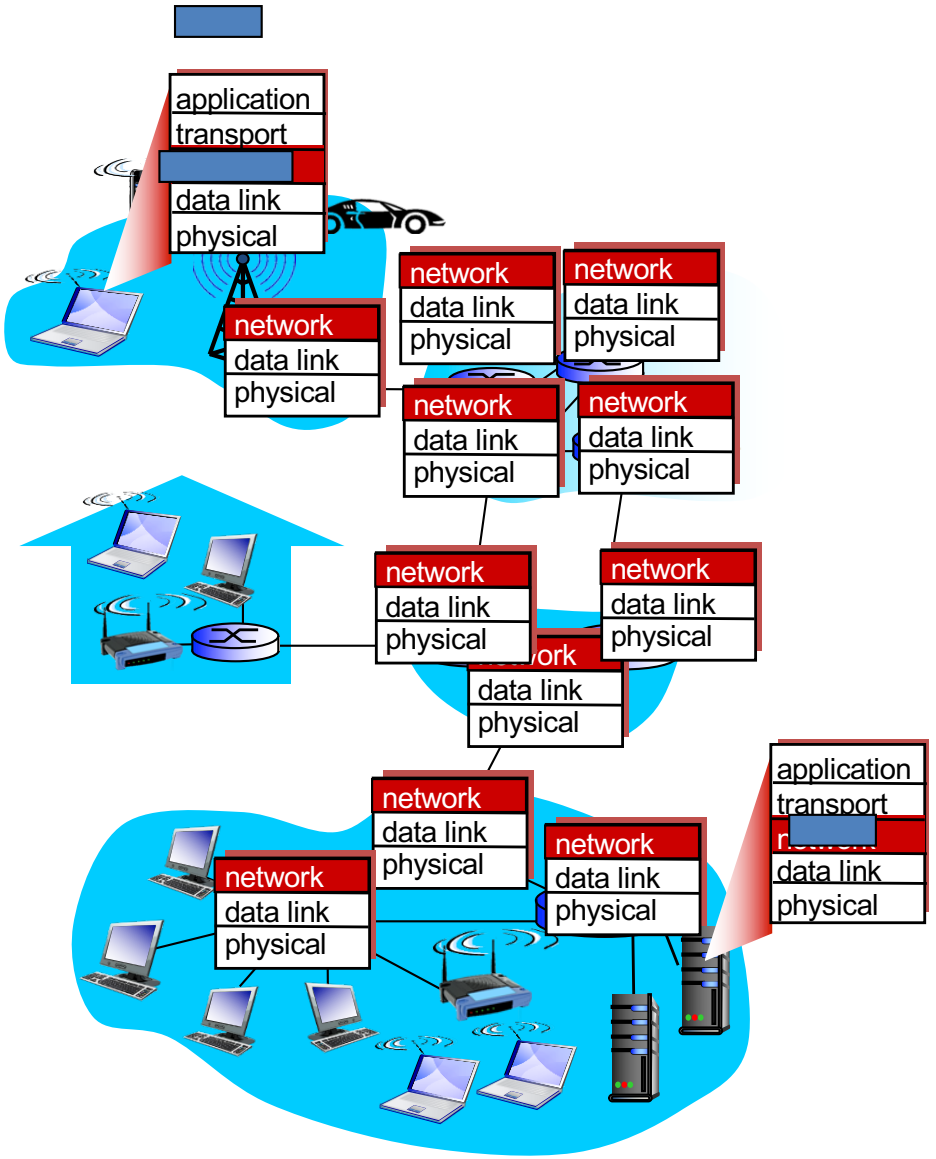
Internetworking



- Cerf & Kahn in 1974,
  - "A Protocol for Packet Network Intercommunication"
  - Foundation for the modern Internet
- **Routers** forward **packets** from source to destination
  - May cross many separate networks along the way
- All packets use a common **Internet Protocol**
  - Any underlying data link protocol
  - Any higher layer transport protocol

Network Layer

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



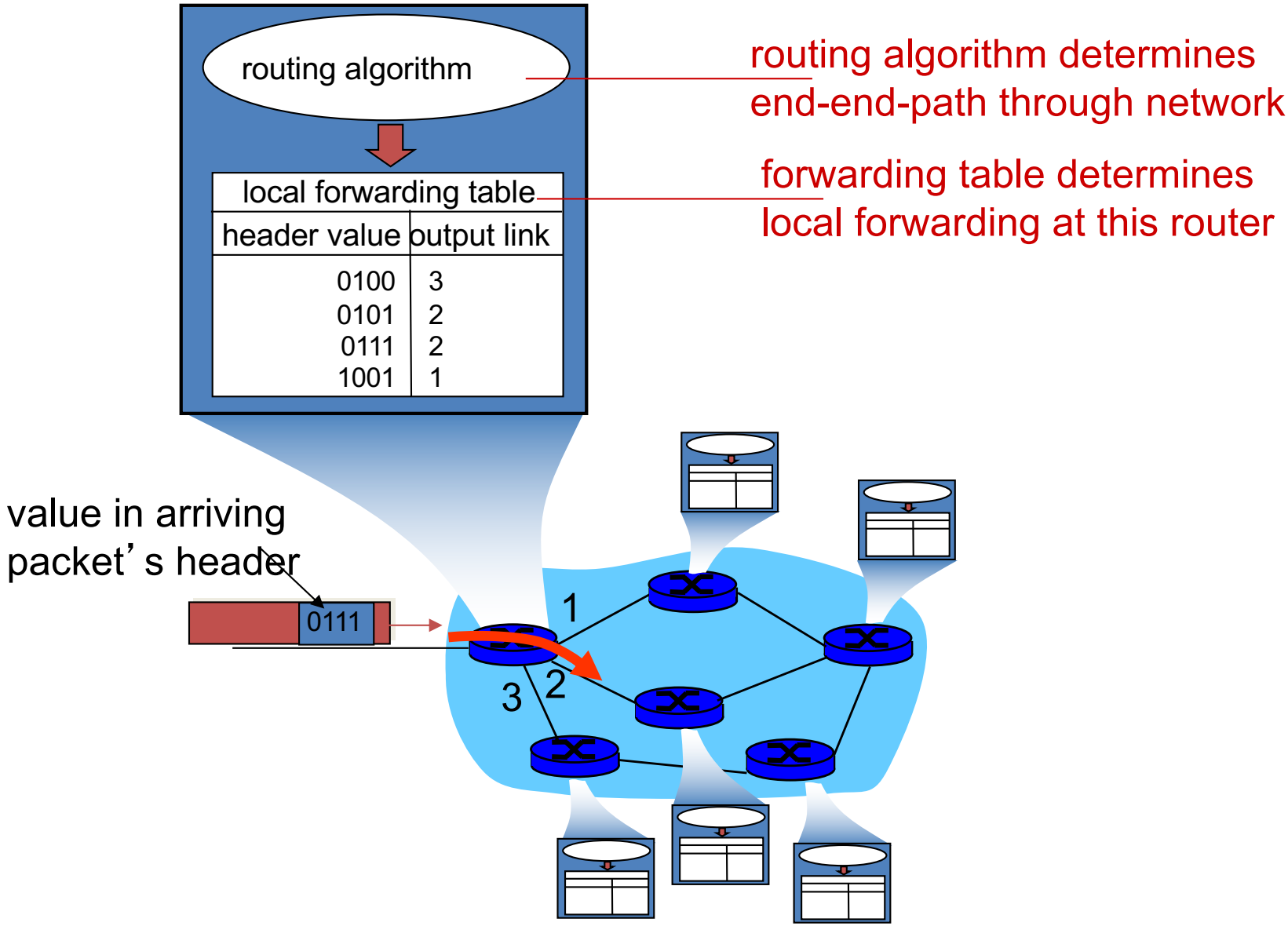
Two key nework-layer functions

- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to dest.
  - *routing algorithms*

analogy:

- *routing:* process of planning trip from source to dest
- *forwarding:* process of getting through single interchange

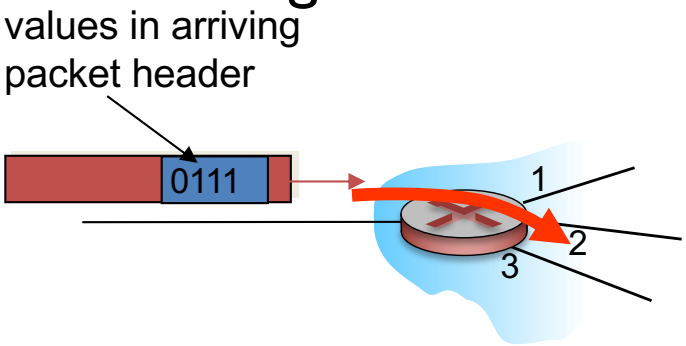
Interplay between routing and forwarding



Network Layer: data vs control plane

Data plane

- local, per-router function
- determines how datagram arriving on router input port is forwarded to router output port
- forwarding function

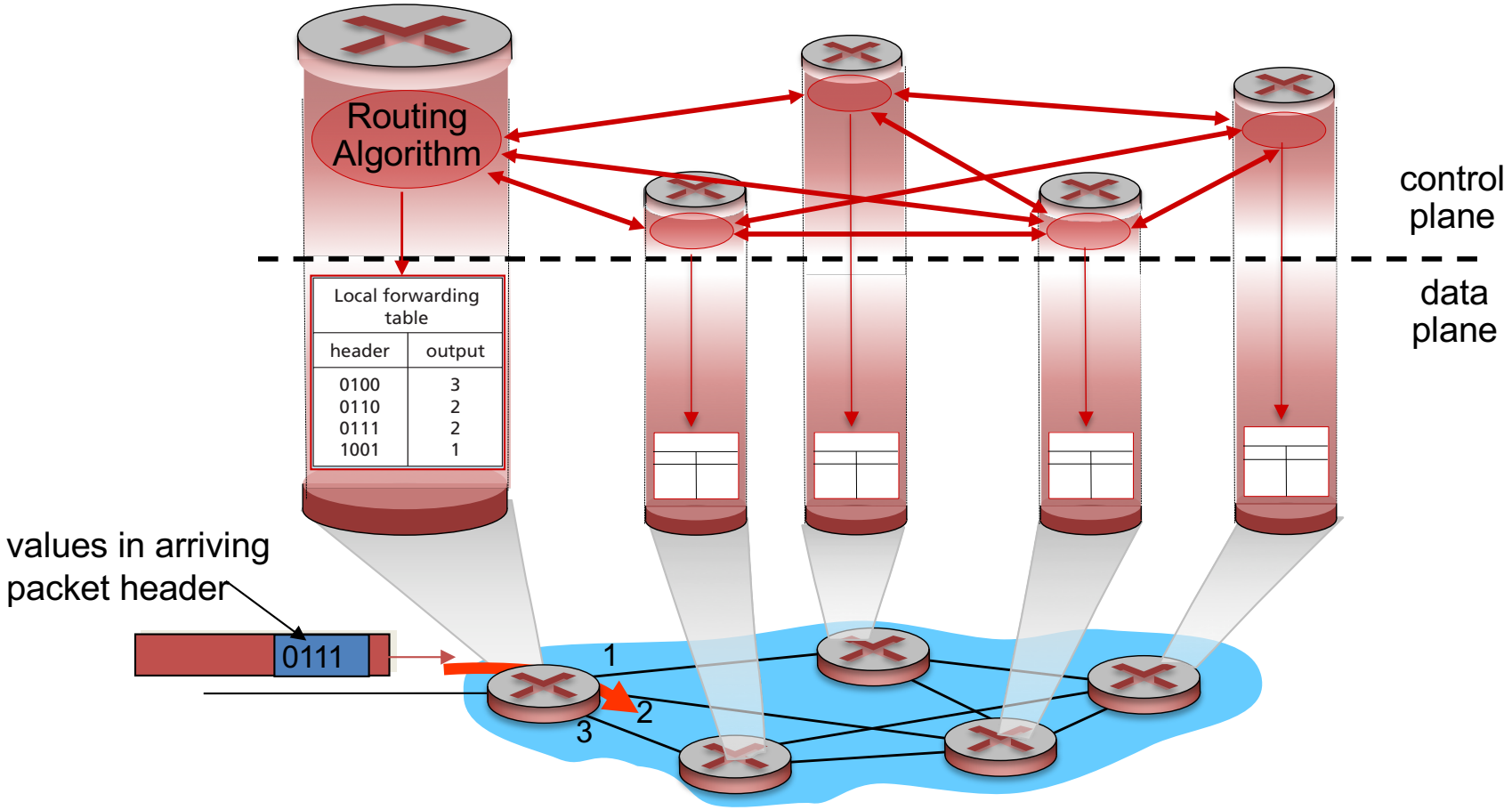


Control plane

- network-wide logic
- determines how datagram is routed among routers along end-end path from source host to destination host
- two control-plane approaches:
  - *traditional routing algorithms:* implemented in routers
  - *software-defined networking (SDN)*: centralised (remote) servers

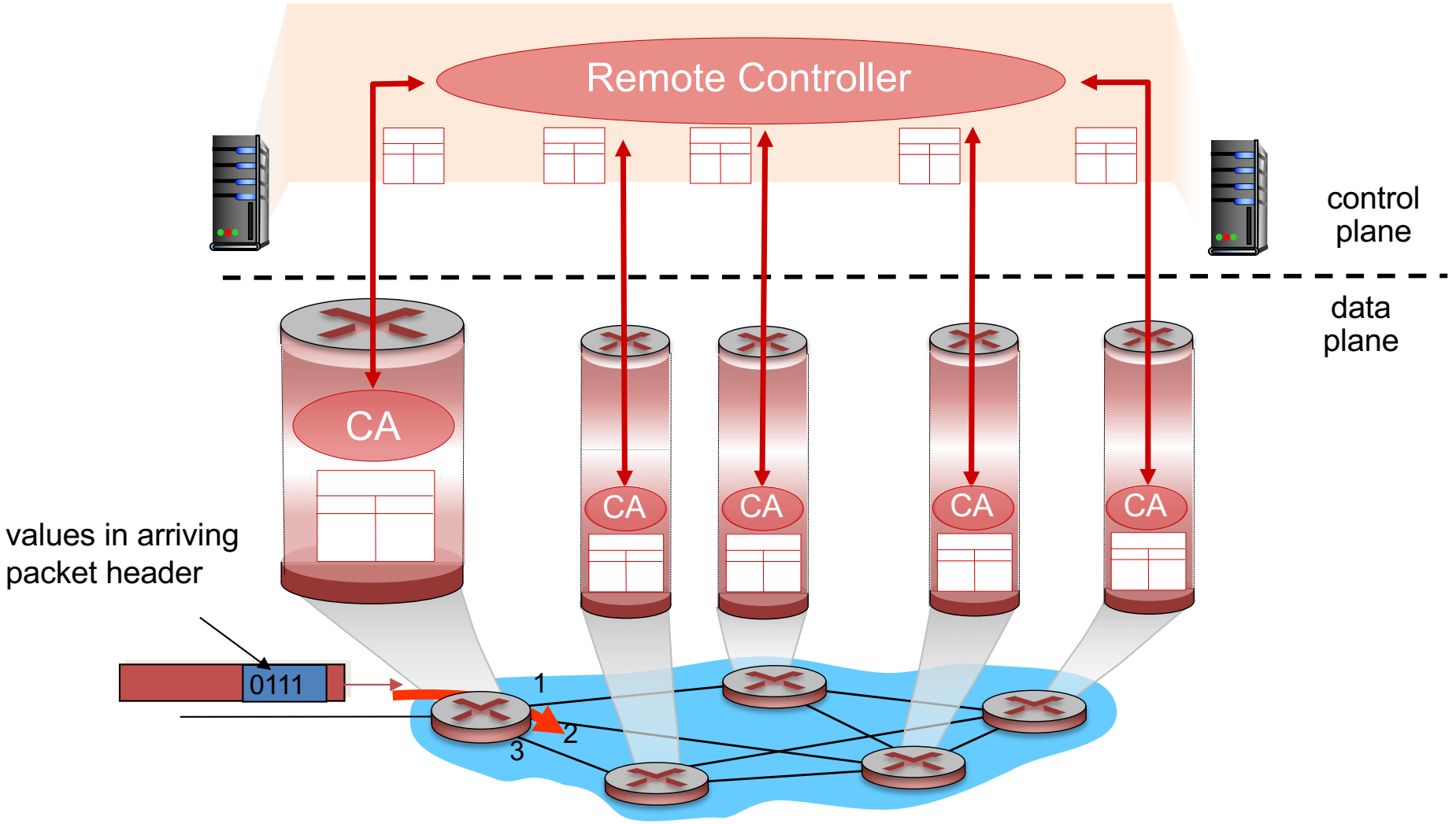
Per-router control plane

Individual routing algorithm components *in each and every router* interact in the control plane



Logically centralized control plane (SDN)

A distinct (typically remote) controller interacts with local control agents (CAs)



Network Layer: service model

Q: What *service model* for "channel" transporting datagrams from sender to receiver?

A. No guarantee whatsoever is provided by IP layer in TCP/IP protocol stack. It's "best effort service".

Network Layer, data plane: outline

4.1 Overview of Network layer
- data plane
- control plane

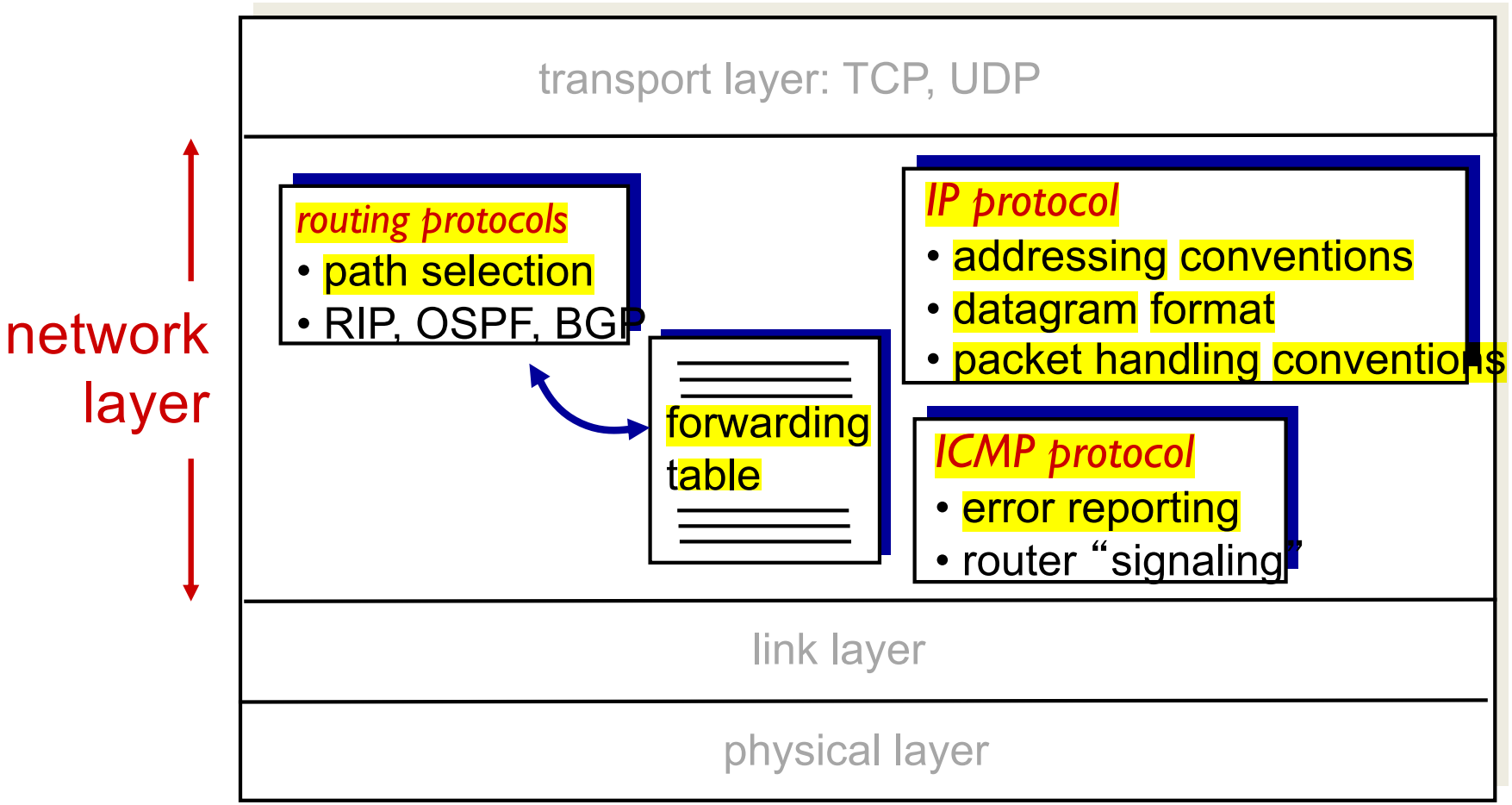
4.2 What's inside a router
-- **Not Covered**

4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

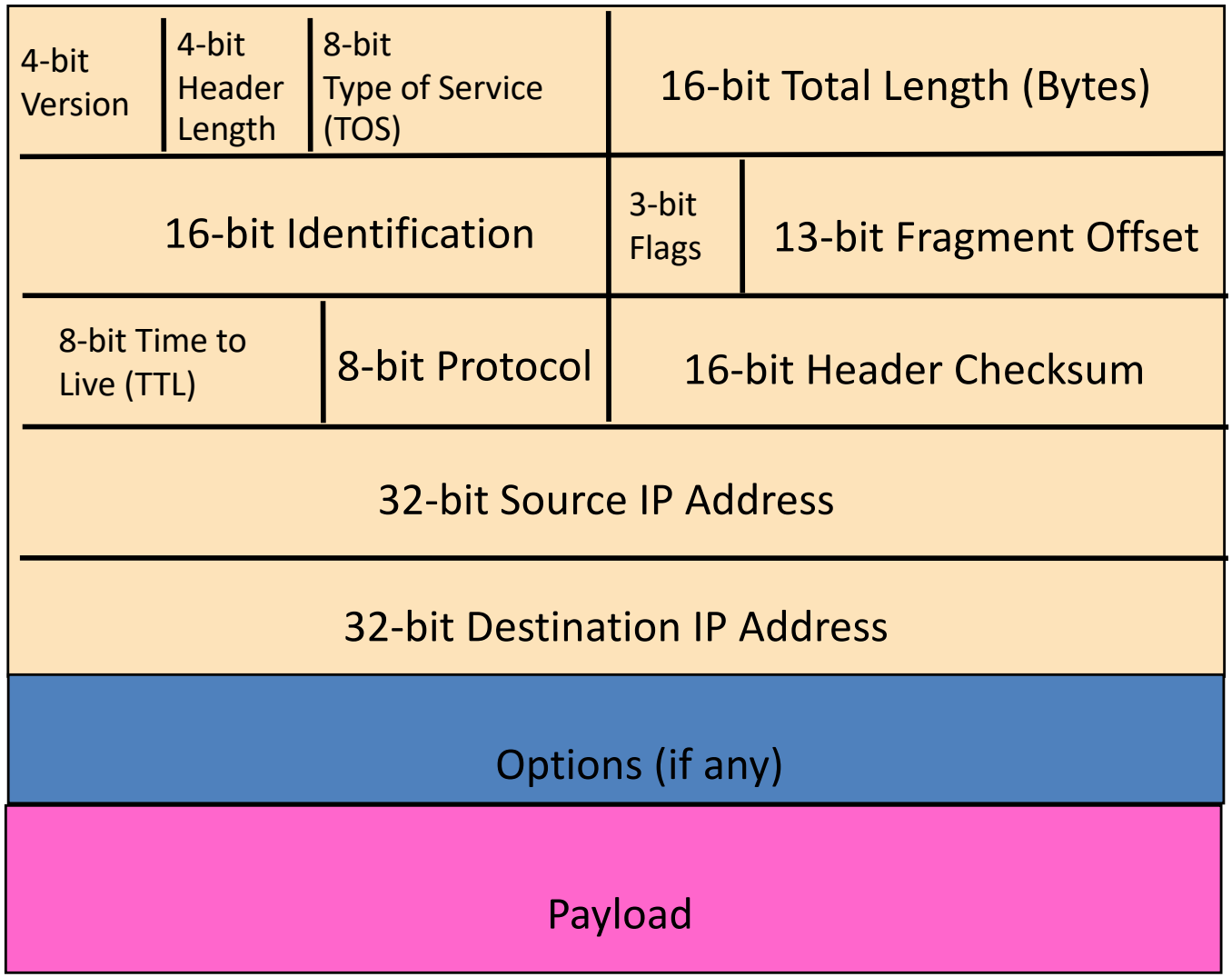
4.4 Generalized forwarding and Software Defined Networking (SDN)
- **Not Covered**

The Internet network layer

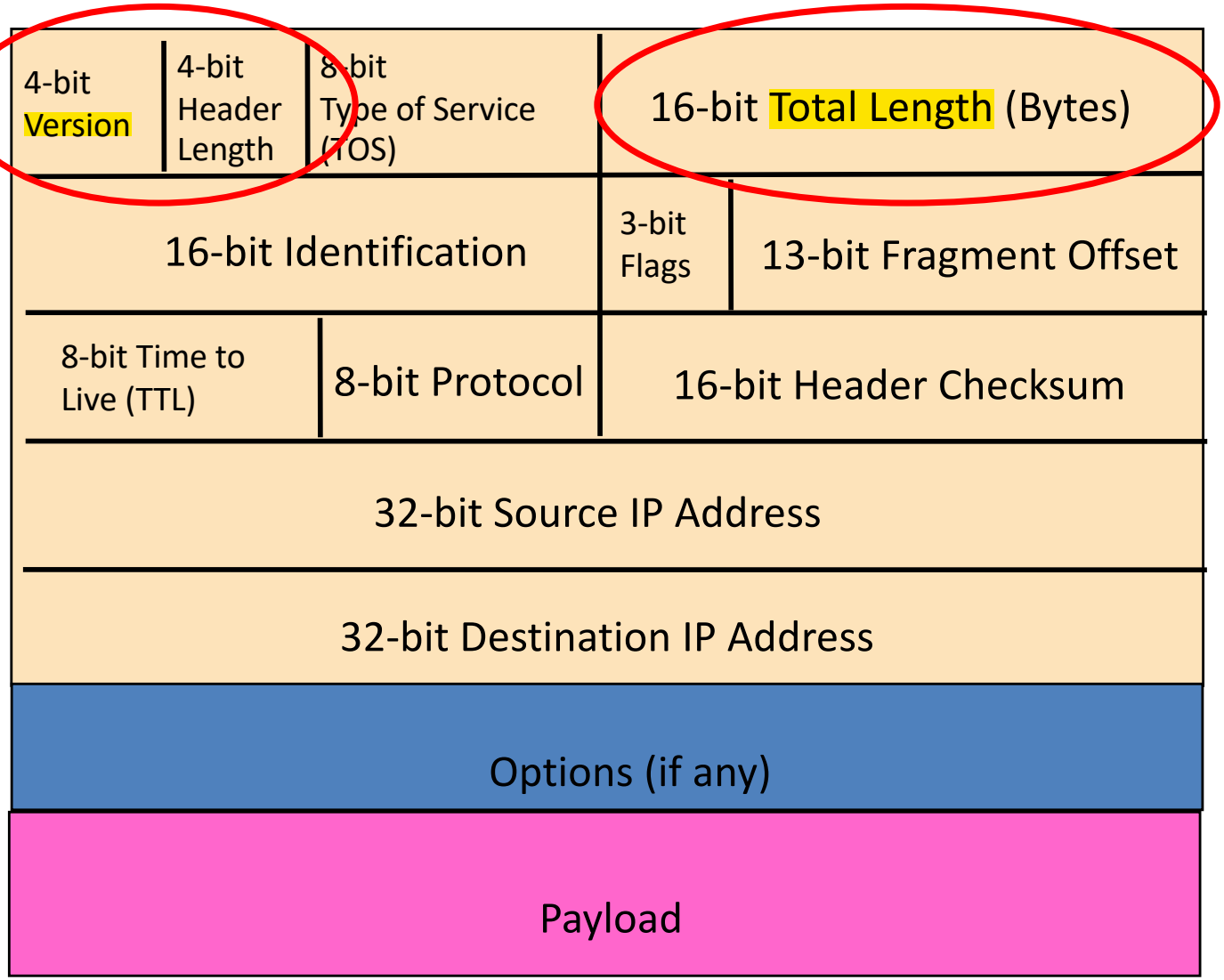
host, router network layer functions:



IP Packet Structure



Fields for Reading Packet Correctly



Reading Packet Correctly

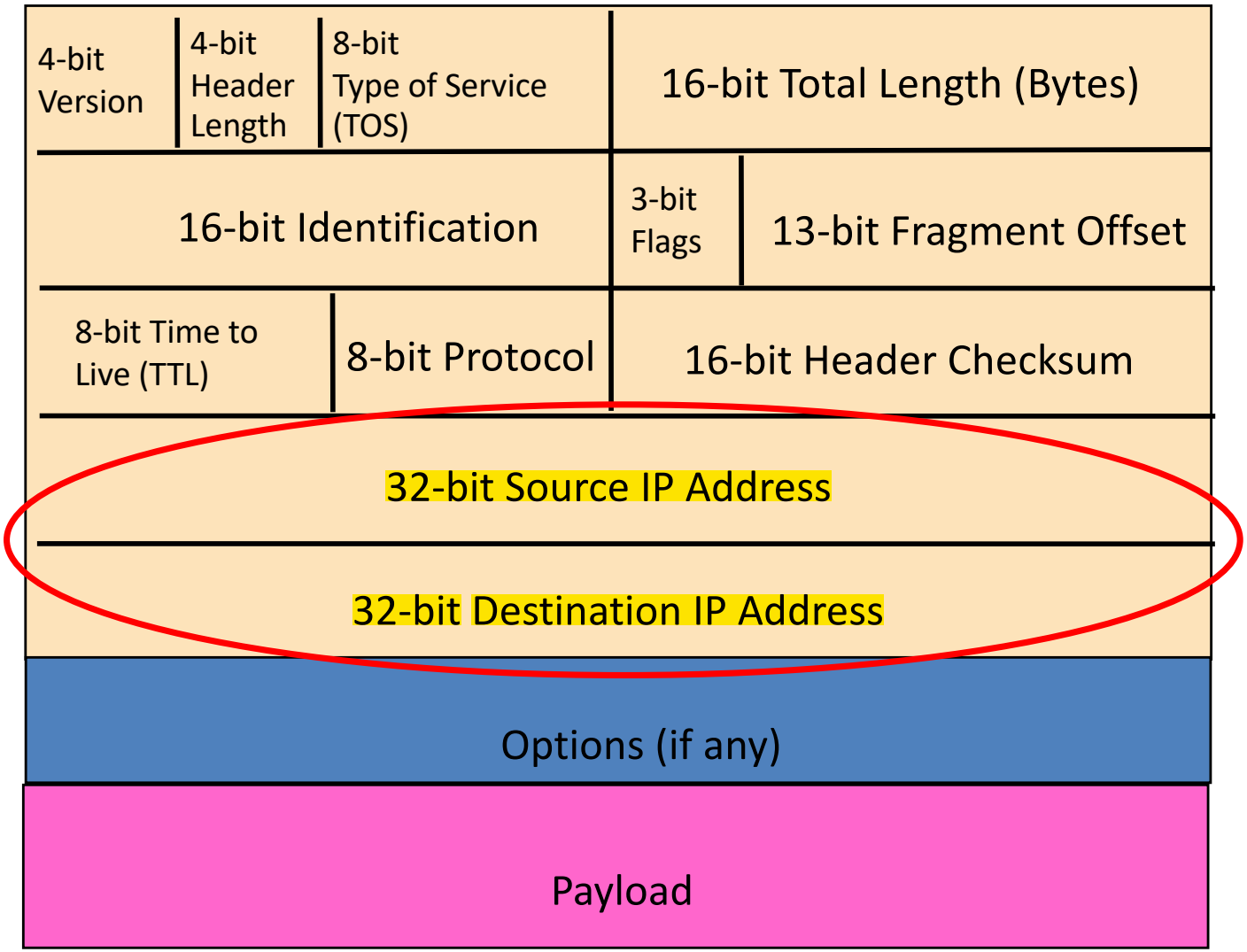
- Version number (4 bits)
  - Indicates the version of the IP protocol
  - Necessary to know what other fields to expect
  - Typically "4" (for IPv4)

- Header length (4 bits)
  - Number of 32-bit words in the header
  - Typically "5" (for a 20-byte IPv4 header) 4 bytes per 32 bit word, 5 * 4 = 20 bytes
  - Can be more when IP options are used

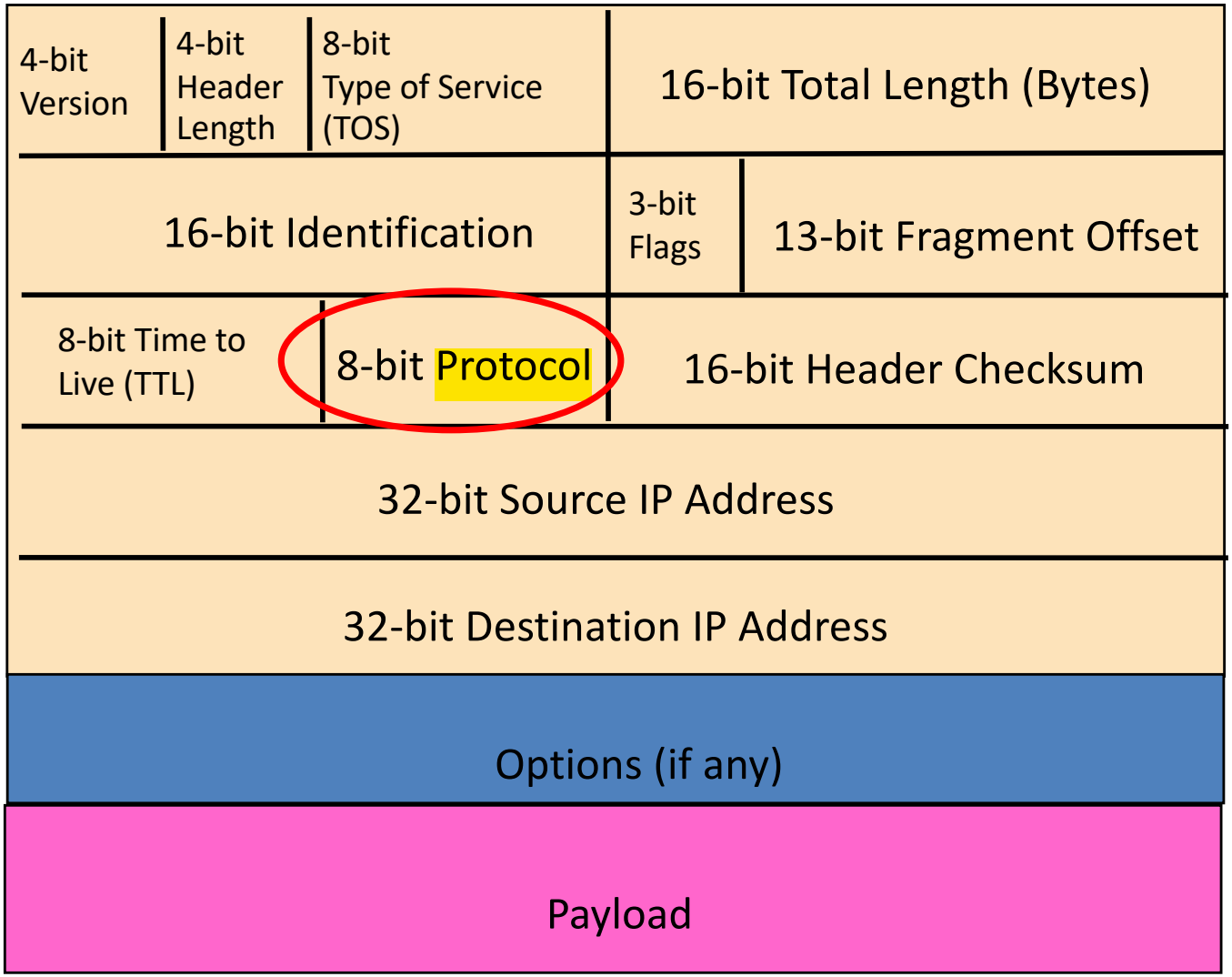
- Total length (16 bits)
  - Number of bytes in the packet
  - Maximum size is 65,535 bytes (2^{16} -1)
  - … though underlying links may impose smaller limits

in which case i'm pretty sure they split it up themselves

Fields for Reaching Destination and Back

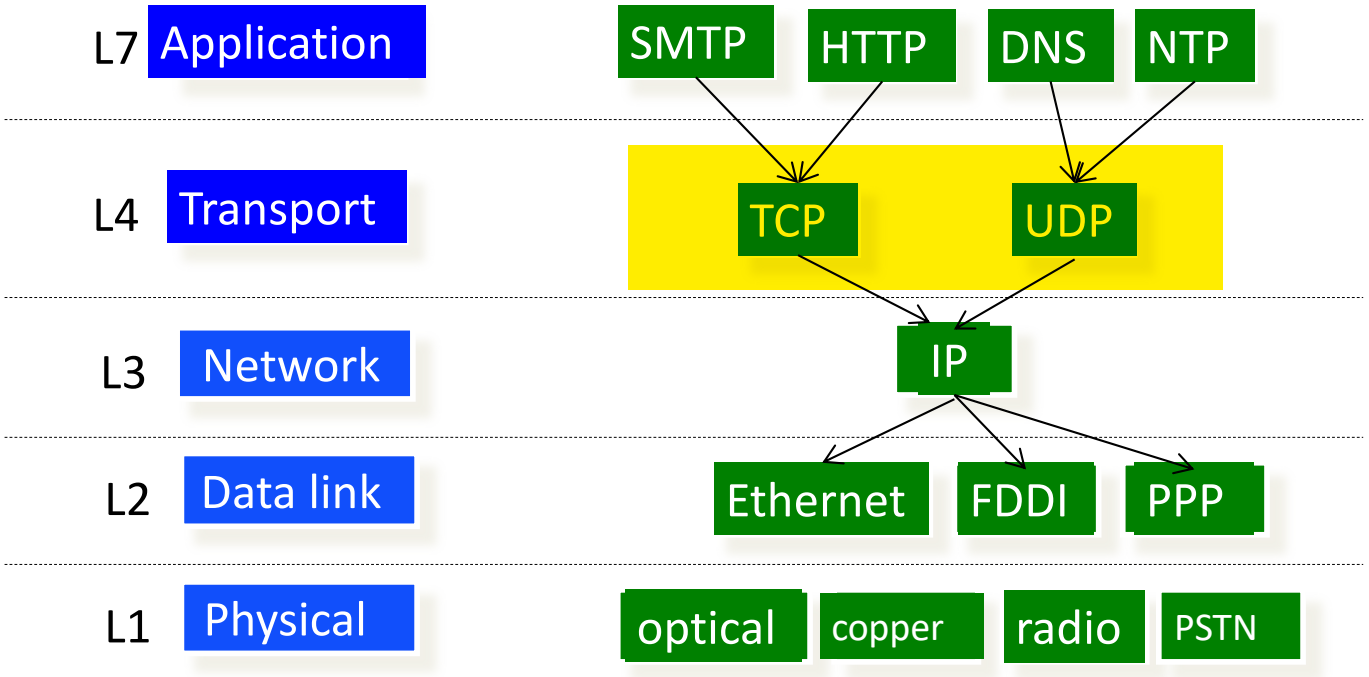


Telling End-Host How to Handle Packet

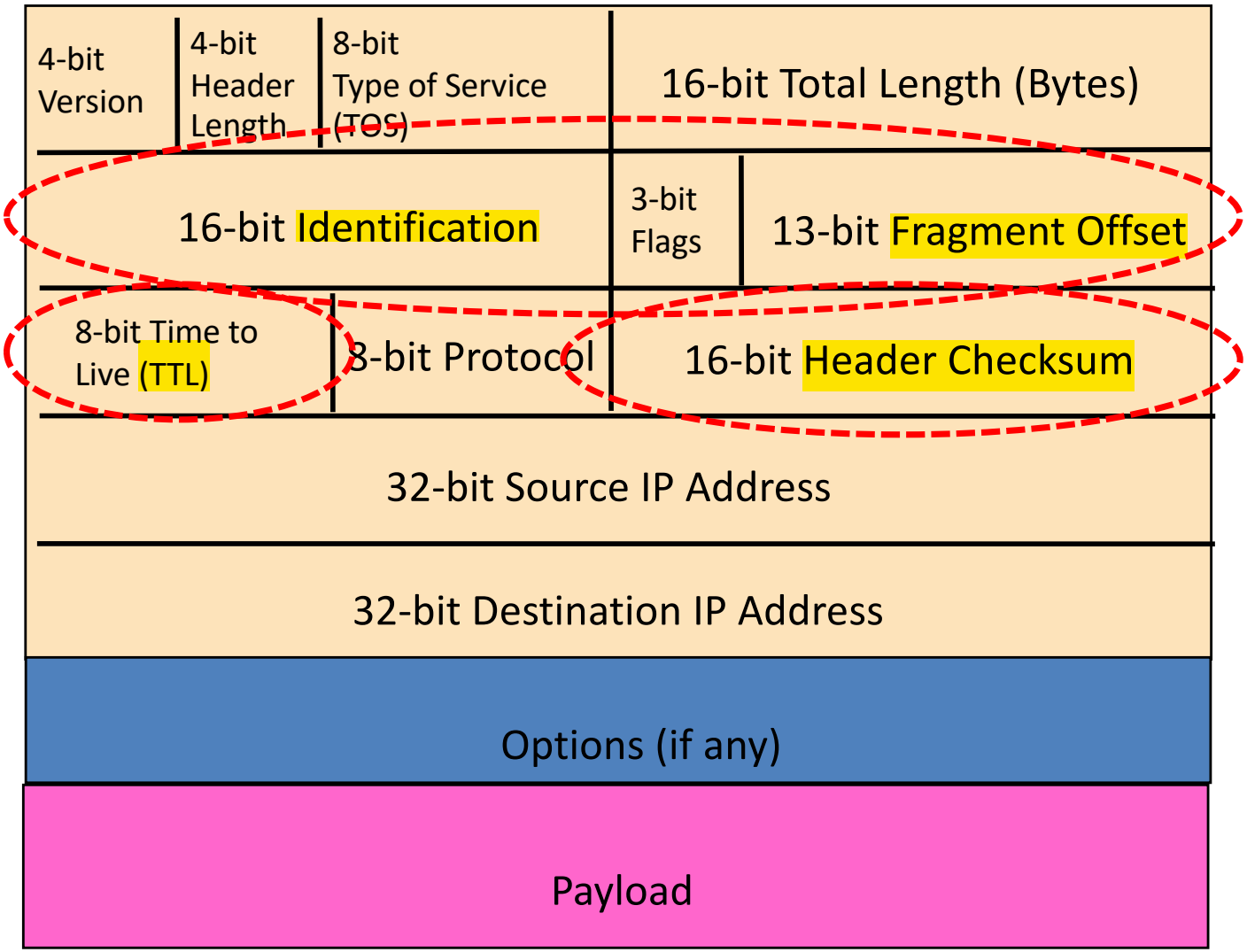


Telling End-Host How to Handle Packet

- Protocol (8 bits)
  - Identifies the higher-level Transport protocol
  - Important for demultiplexing at receiving host



Checksum, TTL and Fragmentation Fields



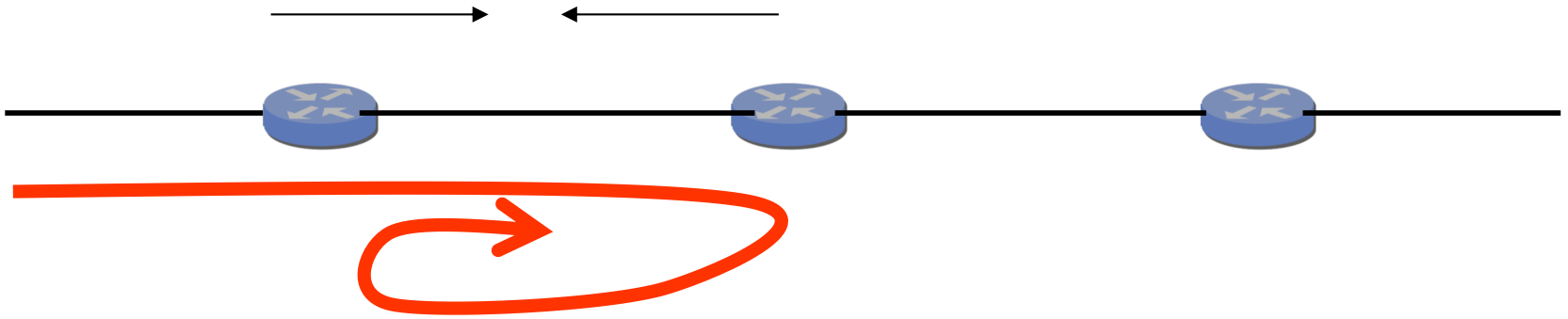
fragment offset for when the packet needs to be split up even more

Potential Problems

- Loop: **TTL** cannot loop indefinitely because the TTL value will eventually hit 0 and it will be dropped
- Header Corrupted: **Checksum**
- Packet too large: **Fragmentation**

Preventing Loops (TTL)

- Forwarding loops cause packets to cycle for a looong time
  - As these accumulate, eventually consume **all** capacity



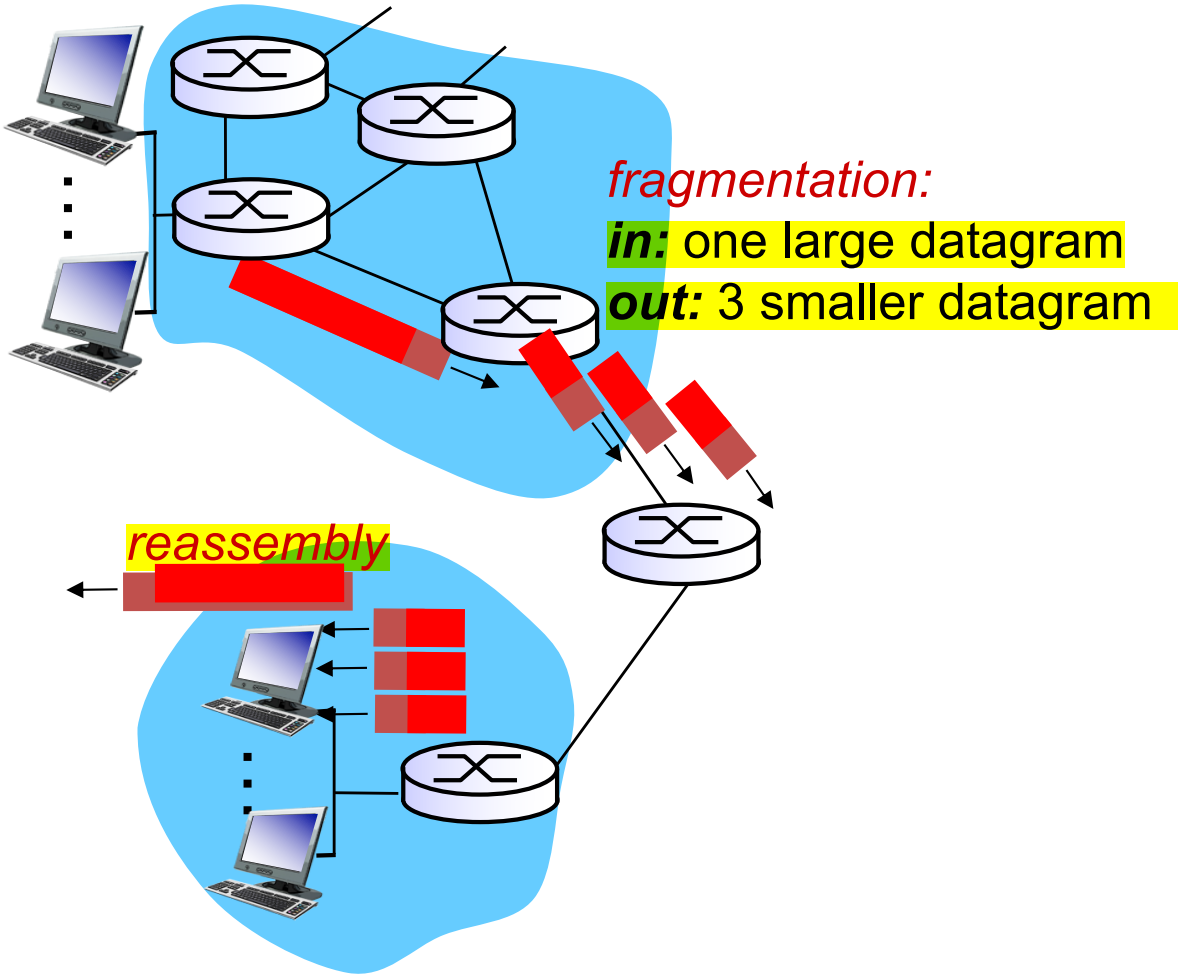
- Time-to-Live (TTL) Field (8 bits)
  - Decremented at each hop, packet discarded if reaches 0
  - …and "time exceeded" message is sent to the source

Header Corruption (Checksum)

- Checksum (16 bits)
  - Particular form of checksum over packet header
- If not correct, router discards packets
  - So it doesn't act on bogus information
- Checksum recalculated at every router
  - **Why?** because the TTL value (and possibly other header fields) are being changed
  - **Why include TTL?** because it could be erroneous
  - **Why only header?** data part is already covered by the transport level checksum i.e. tcp or udp checksum

IP fragmentation, reassembly

- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes

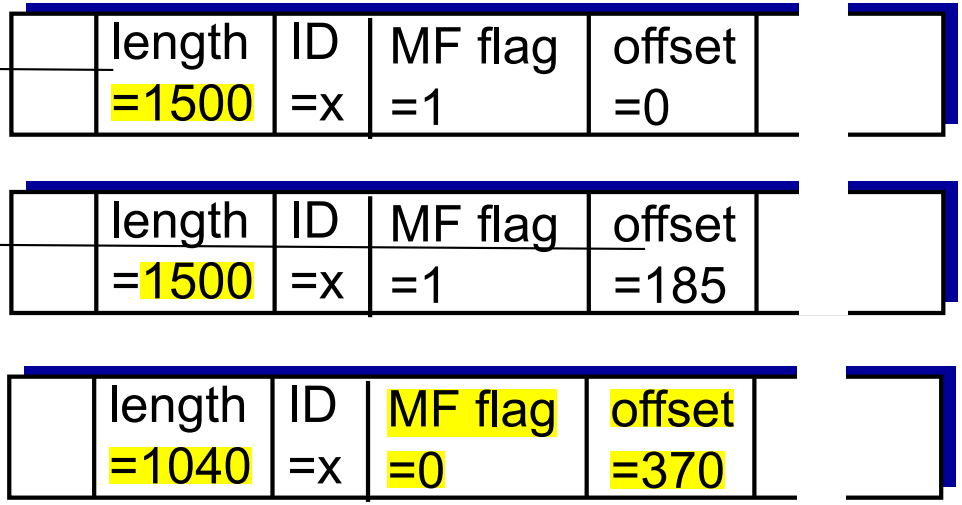
one large datagram becomes several smaller datagrams

Data = 4000 – 20 (IP header) =3980
F1=1480
F2=1480
F3=1020

1480 bytes in data field

offset = 1480/8

byte offsets, not bit offsets

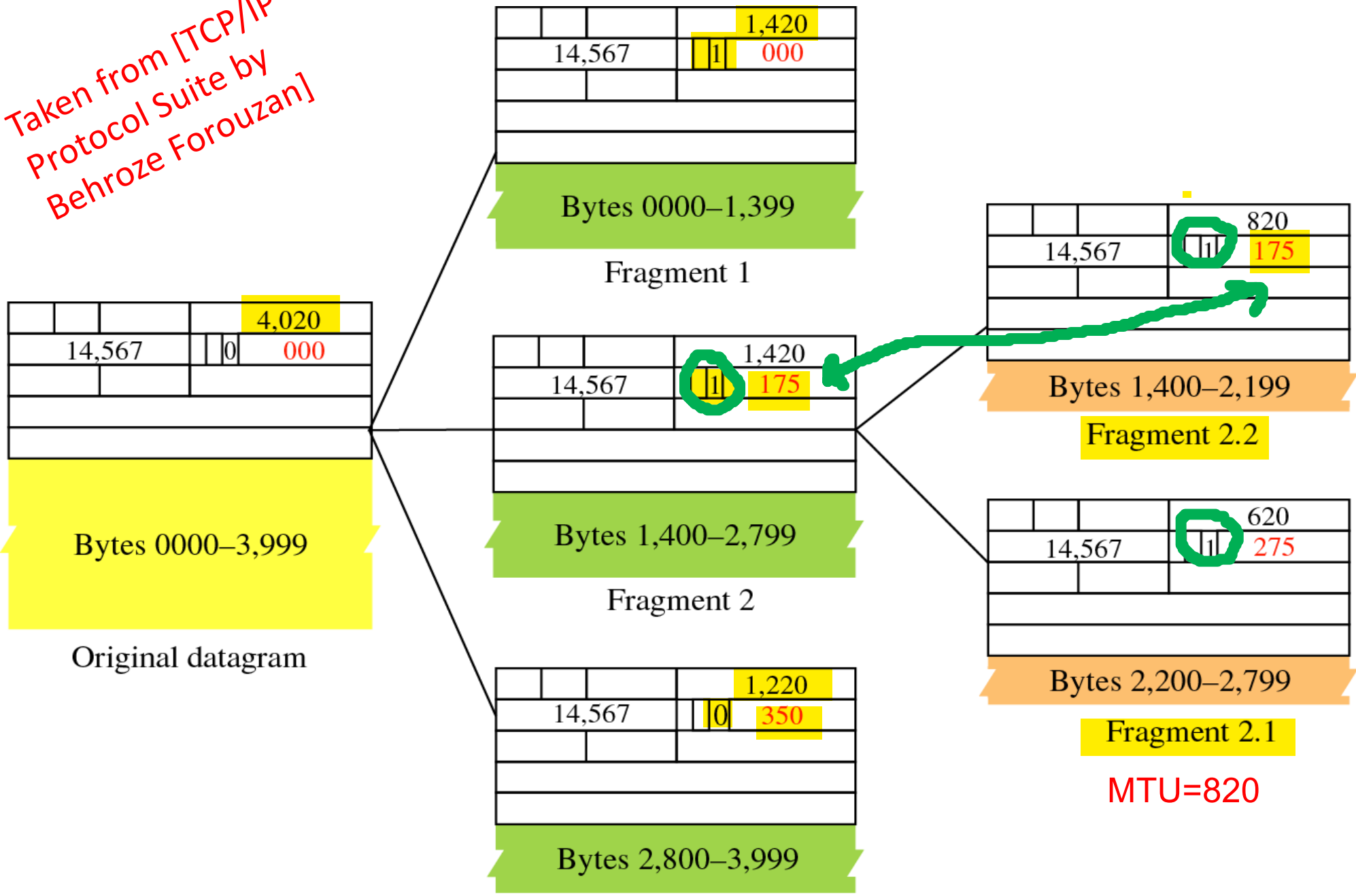


Applet:
http://media.pearsoncmg.com/aw/aw_kurose_network_2/applets/ip/ipfragmentation.html

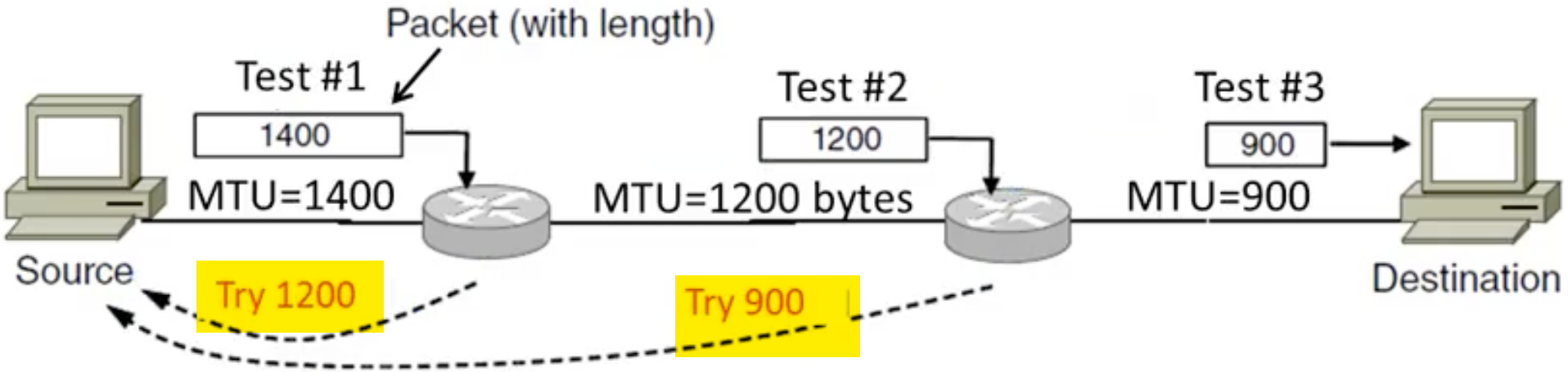
IPv4 fragmentation procedure

- **Fragmentation**
  - Router breaks up datagram in size that output link can support
  - Copies IP header to pieces
  - Adjust length on pieces
  - Set offset to indicate position
  - Set MF (More fragments) flag on pieces except the last
  - Re-compute checksum
- **Re-assembly**
  - Receiving host uses identification field with MF and offsets to complete the datagram.
- Fragmentation of fragments also supported

Taken from [TCP/IP Protocol Suite by Behroze Forouzan]



Path MTU Discovery procedure



- Host
  - Sends a big packet to test whether all routers in path to the destination can support or not
  - Set DF (Do not fragment) flag
- Routers
  - Drops the packet if it is too large (as DF is set)
  - Provides feedback to Host with ICMP message telling the maximum supported size

(Internet Control Message Protocol) error reporting protocol

How can we use fragmentation for evil?

A: Send fragments that overlap.

B: Send many tiny fragments, none of which have offset 0.

C: Send segments that when assembled, are bigger than the maximum IP datagram.

D: More than one of the above.

E: Nah, networks (and operating systems) are too robust for this to cause problems.

www.zeetings.com/salil

IP Fragmentation Attacks …

https://en.wikipedia.org/wiki/IP_fragmentation_attack

IP fragmentation exploits [edit]

IP fragment overlapped [edit]

The IP fragment overlapped exploit occurs when two fragments contained within the same IP datagram have offsets that indicate that they overlap each other in positioning within the datagram. This could mean that either fragment A is being completely overwritten by fragment B, or that fragment A is partially being overwritten by fragment B. Some operating systems do not properly handle fragments that overlap in this manner and may throw exceptions or behave in other undesirable ways upon receipt of overlapping fragments. This is the basis for the teardrop Denial of service attacks.

IP fragmentation buffer full [edit]

The IP fragmentation buffer full exploit occurs when there is an excessive amount of incomplete fragmented traffic detected on the protected network. This could be due to an excessive number of incomplete fragmented datagrams, a large number of fragments for individual datagrams or a combination of quantity of incomplete datagrams and size/number of fragments in each datagram. This type of traffic is most likely an attempt to bypass security measures or Intrusion Detection Systems by intentional fragmentation of attack activity.

IP fragment overrun [edit]

The IP Fragment Overrun exploit is when a reassembled fragmented datagram exceeds the declared IP data length or the maximum datagram length. By definition, no IP datagram should be larger than 65,535 bytes. Systems that try to process these large datagrams can crash, and can be indicative of a denial of service attempt.

IP fragment overwrite [edit]

Overlapping fragments may be used in an attempt to bypass Intrusion Detection Systems. In this exploit, part of an attack is sent in fragments along with additional random data; future fragments may overwrite the random data with the remainder of the attack. If the completed datagram is not properly reassembled at the IDS, the attack will go undetected.

IP fragment too many datagrams [edit]

The Too Many Datagrams exploit is identified by an excessive number of incomplete fragmented datagrams detected on the network. This is usually either a denial of service attack or an attempt to bypass security measures. An example of "Too Many Datagrams", "Incomplete Datagram" and "Fragment Too Small" is the Rose Attack.[1]

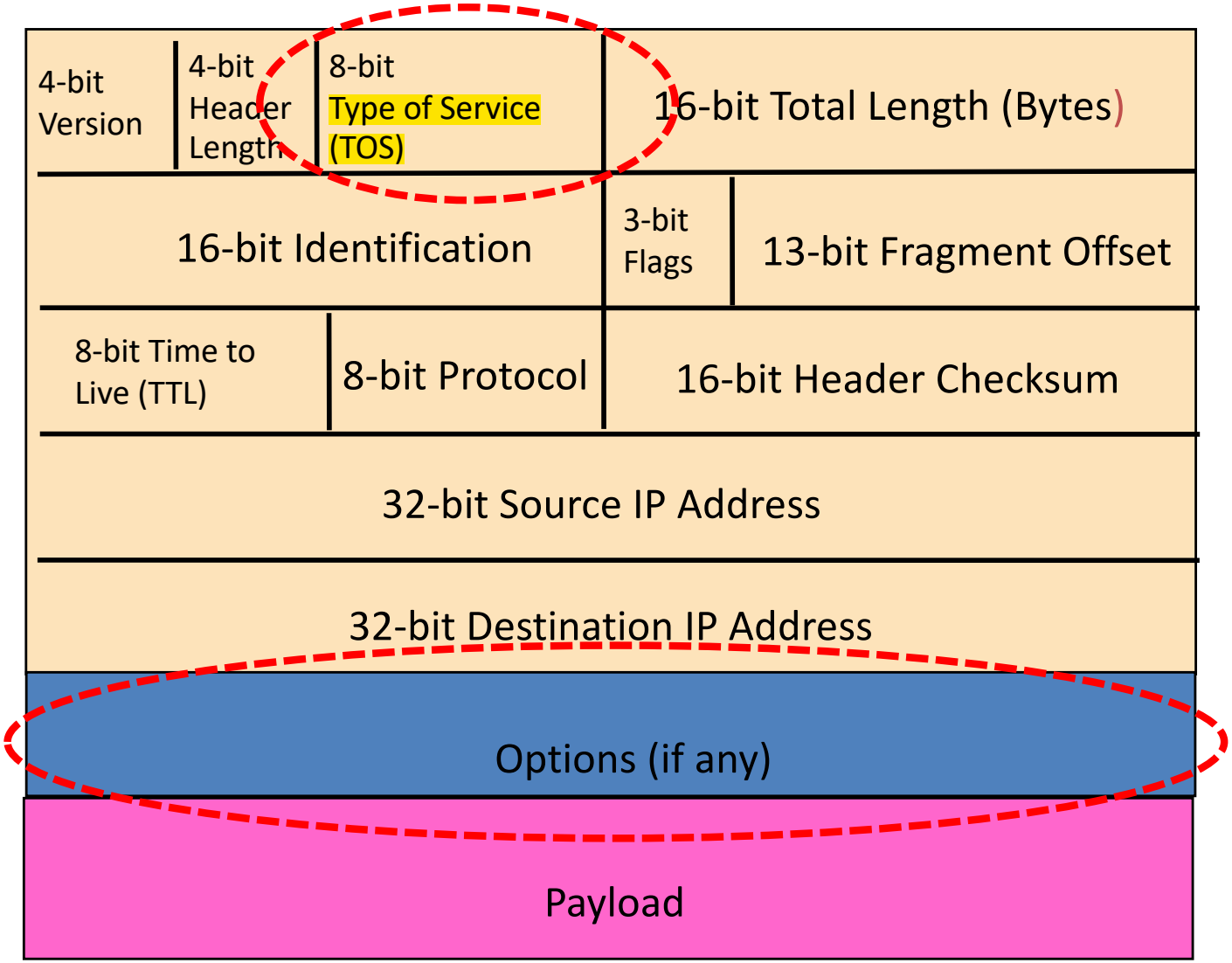
IP fragment incomplete datagram [edit]

This exploit occurs when a datagram can not be fully reassembled due to missing data. This can indicate a denial of service attack or an attempt to defeat packet filter security policies.

IP fragment too small [edit]

An IP Fragment Too Small exploit is when any fragment other than the final fragment is less than 400 bytes, indicating that the fragment is likely intentionally crafted. Small fragments may be used in denial of service attacks or in an attempt to bypass security measures or detection.

Fields for Special Handling

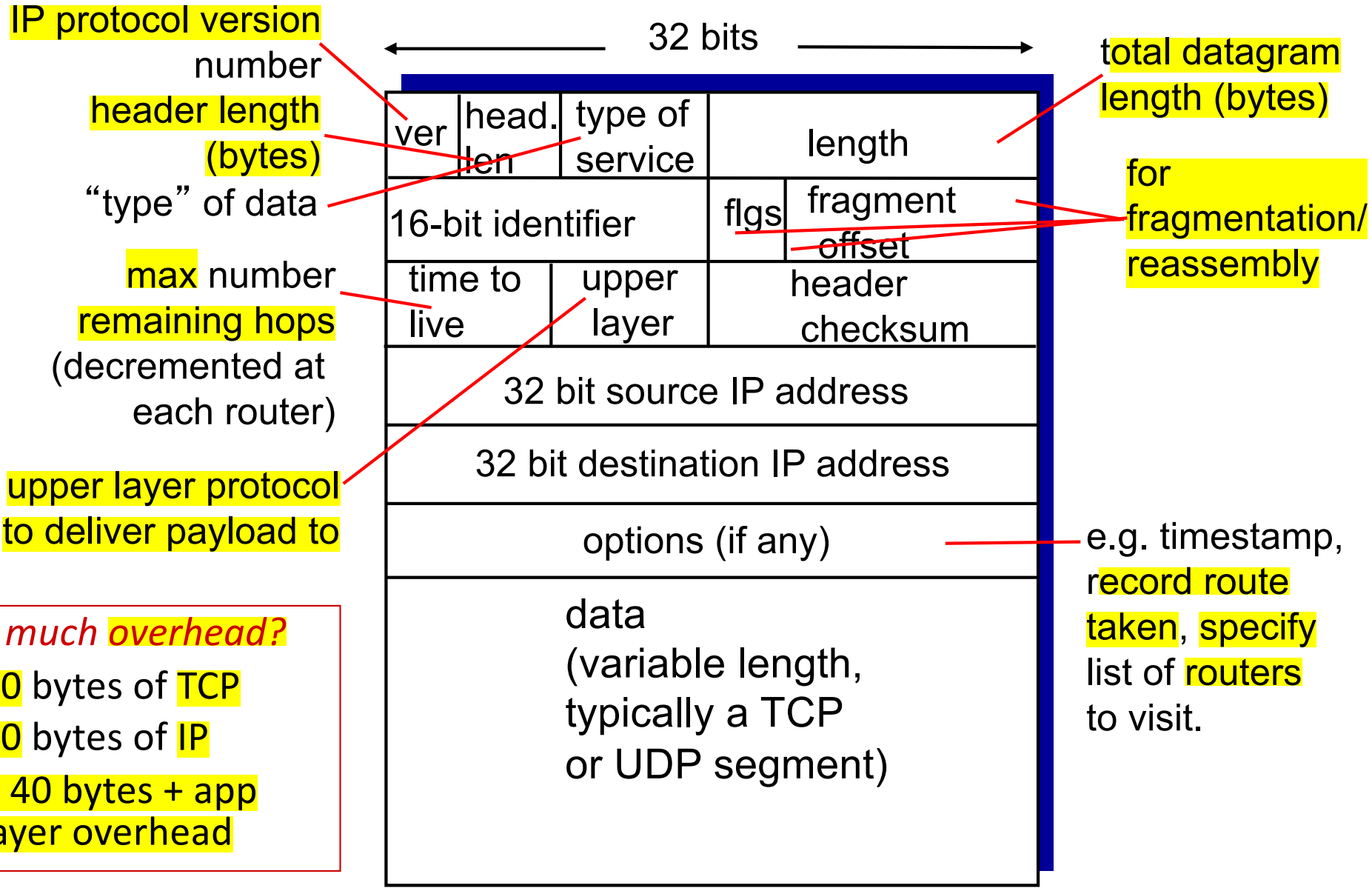


Special Handling

- "Type of Service", or "Differentiated Services Code Point (DSCP)" (8 bits)
  - Allow packets to be treated differently based on needs
  - E.g., low delay for audio, high bandwidth for bulk transfer
  - Has been redefined several times

- Options (not often used)

RECAP: IP datagram format



how much overhead?

- 20 bytes of TCP
- 20 bytes of IP
- = 40 bytes + app layer overhead

Network Layer, data plane: outline

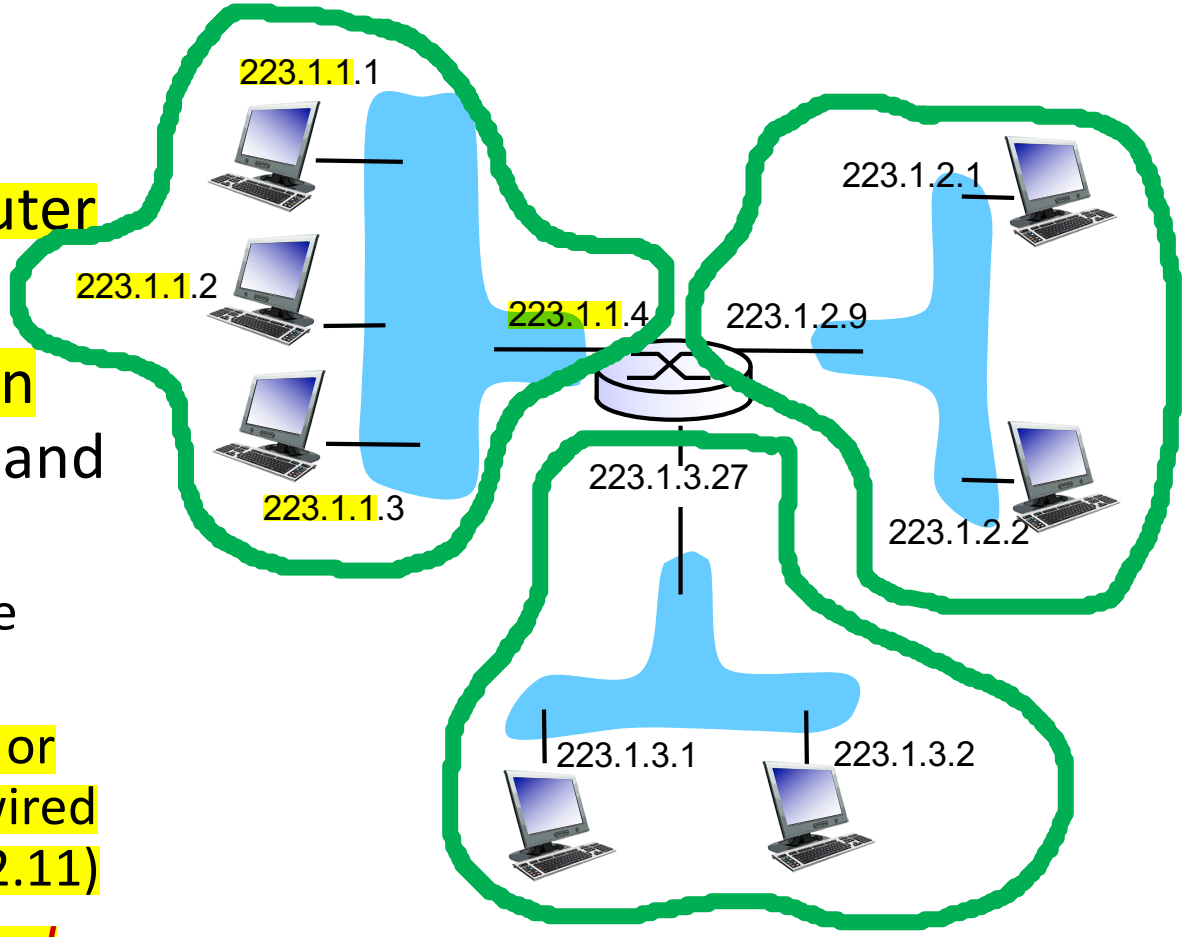
4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router

4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

IP addressing: introduction

- *IP address:* 32-bit identifier for host, router *interface*
- *interface:* connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)
- *IP addresses associated with each interface*



223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

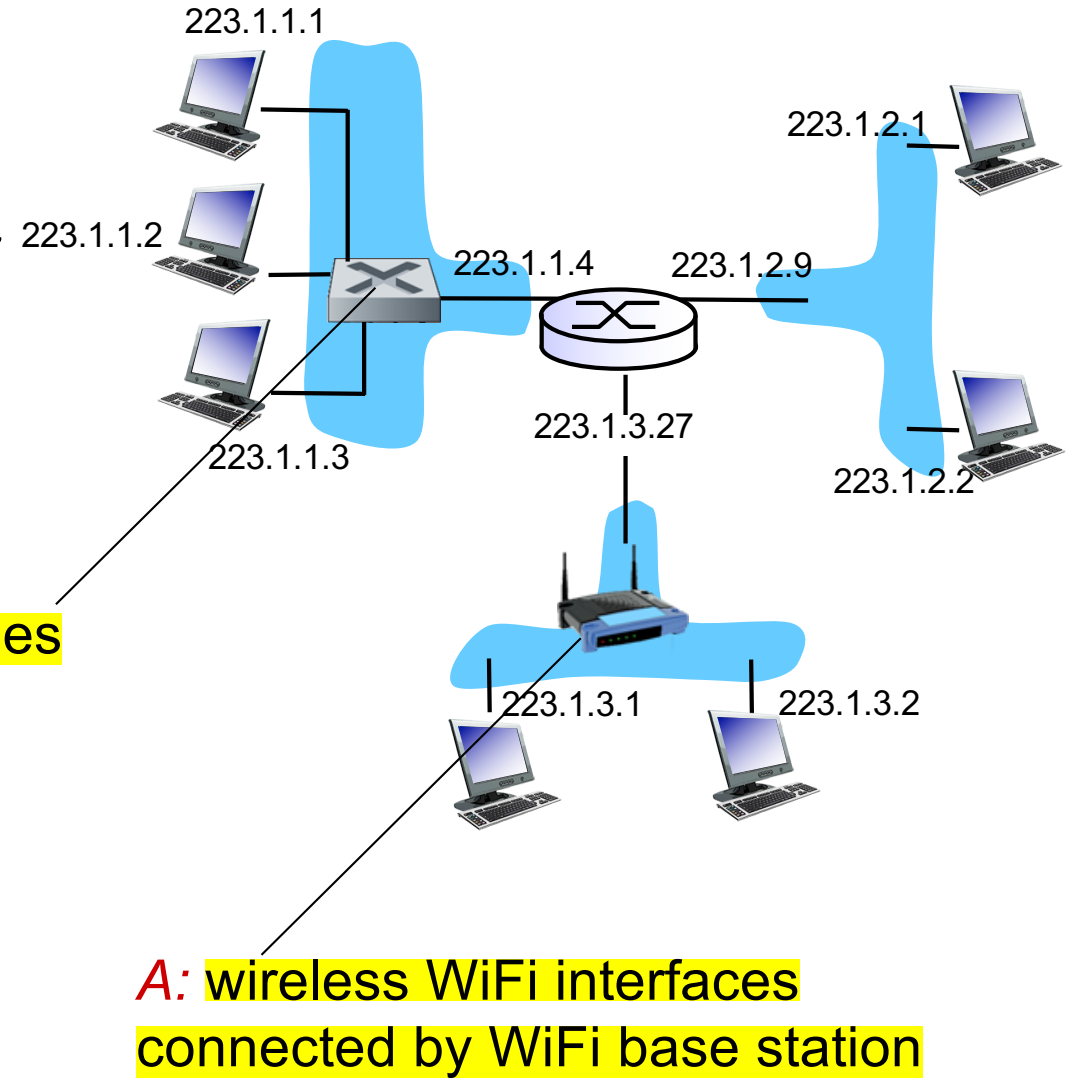
so it's not the IP address of your machine, it's the IP address of your machines interface
my ethernet IP will be different than my wifi IP

uses dot notation between each of the 4 bytes values

IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in the link layer

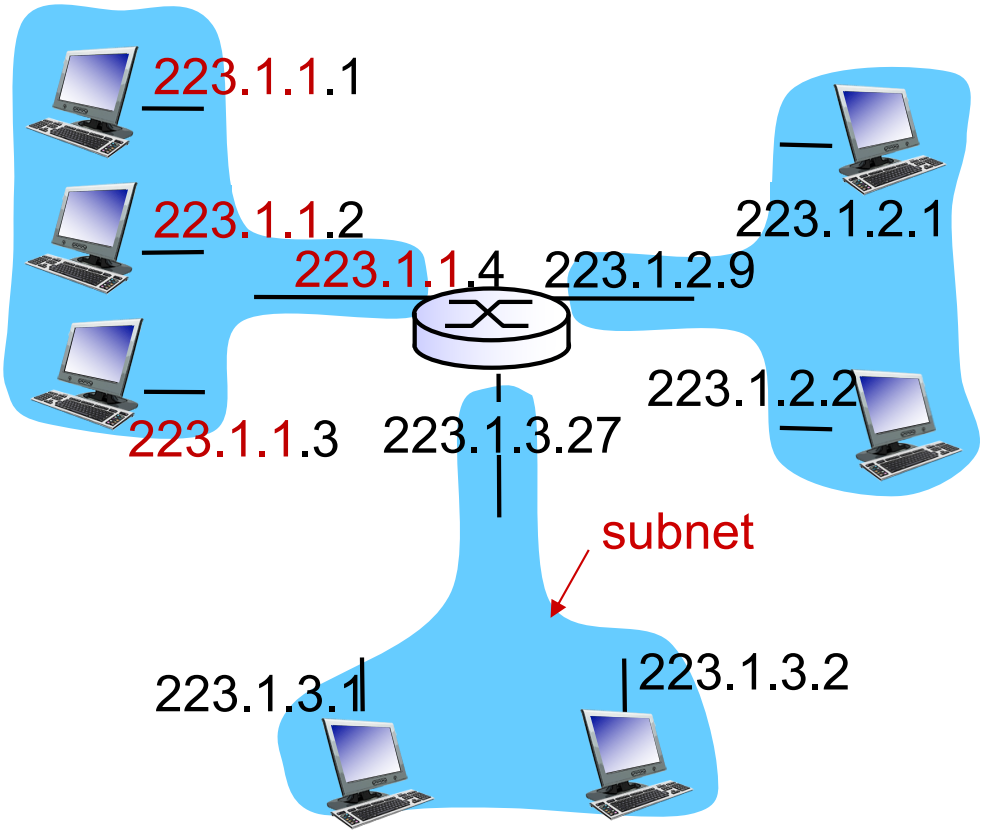


A: wired Ethernet interfaces connected by Ethernet switches

A: wireless WiFi interfaces connected by WiFi base station

Networks

- IP address:
  - network part - high order bits
  - host part - low order bits
- *what's a network ?*
  - device interfaces with same network part of IP address
  - can physically reach each other *without intervening router*



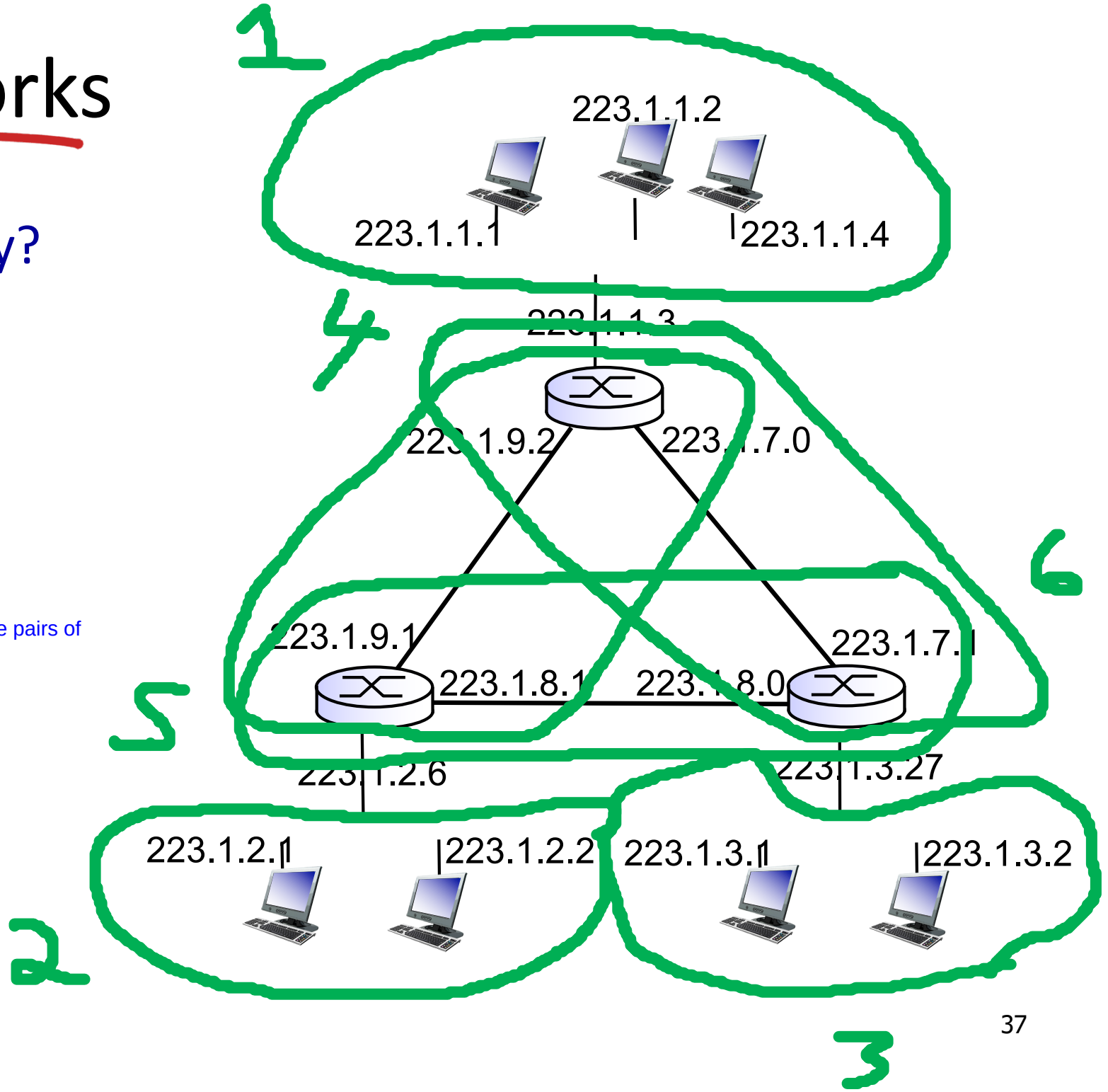
inter-network consisting of 3 nets

Networks

how many?

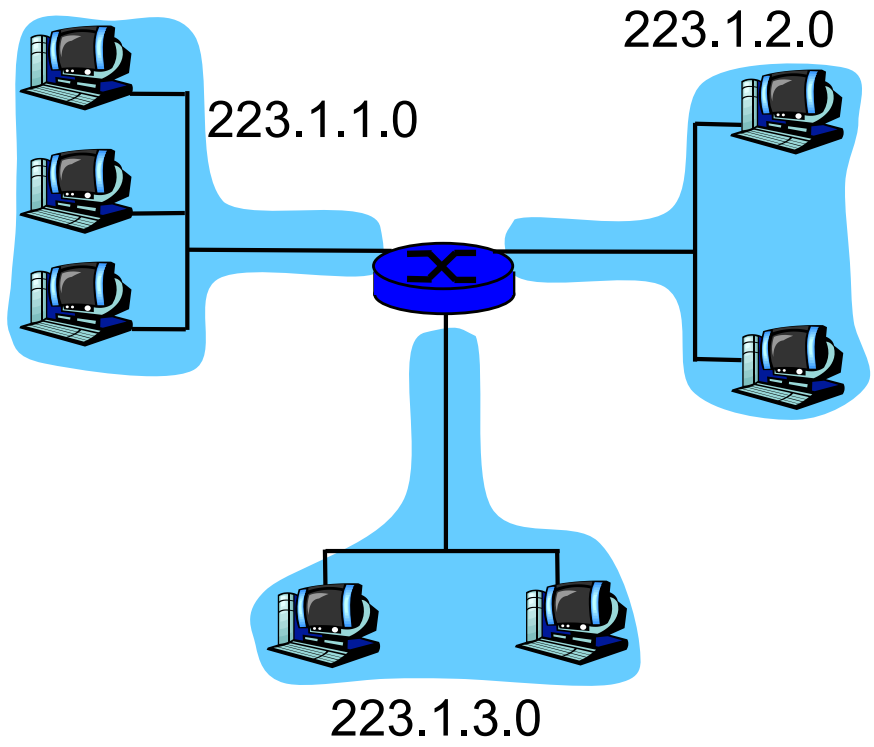
6

the three obvious networks, then the pairs of routers also form networks



Masking

- Mask
  - Used in conjunction with the network address to indicate how many higher order bits are used for the network part of the address
    - Bit-wise AND
  - 223.1.1.0 with mask 255.255.255.0 hence first 24 bits used for network part
- Broadcast Address
  - host part is all 111's
  - E.g. 223.1.1.255
- Network Address
  - Host part is all 0000's
  - E.g. 223.1.1.0
- Both of these are not assigned to any host reserved as special addresses



Host B	Dot-decimal address	Binary this binary is wrong, should be 11011111.
IP address	223.1.1.2	11111101.00000001.00000001.00000010
Mask	255.255.255.0	11111111.11111111.11111111.00000000
Network Part	223.1.1.0	11111101.00000001.00000001.00000000
Host Part	0.0.0.2	00000000.00000000.00000000.00000010

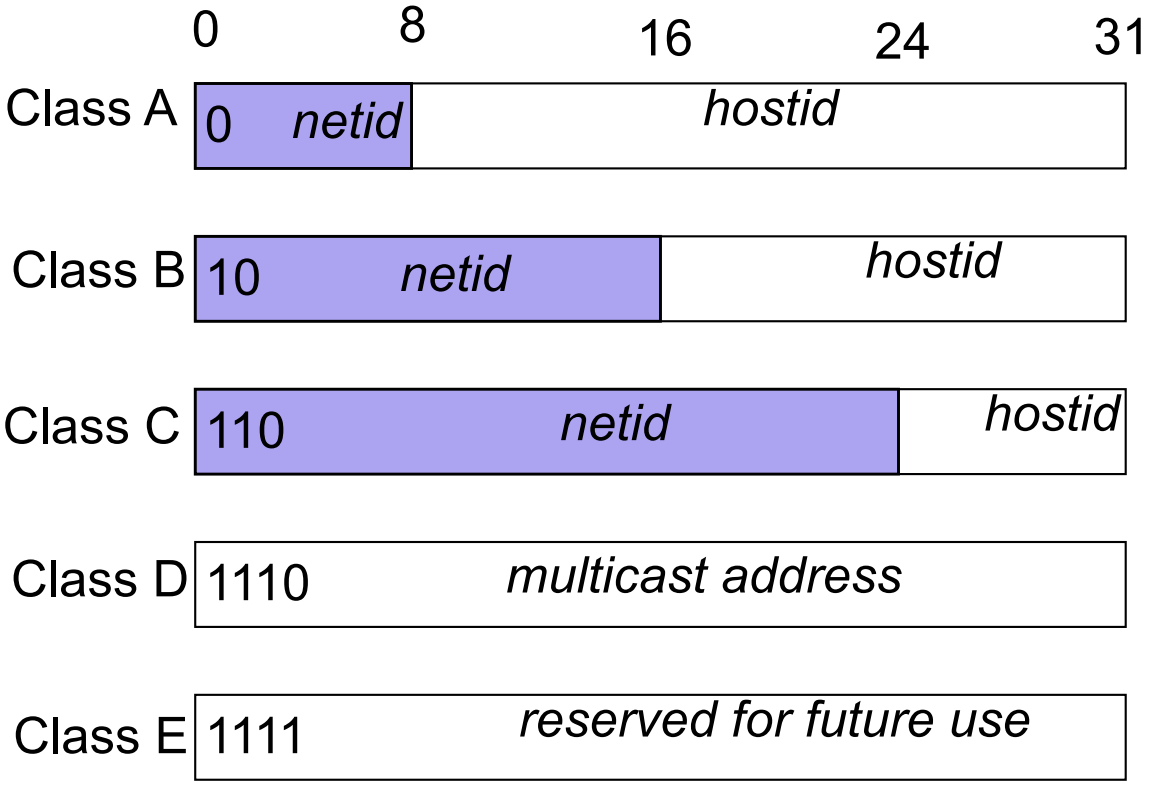
Original Internet Addresses

- First eight bits: network address (/8)
- Last 24 bits: host address, ~16.7 million

Assumed 256 networks were more than enough!

Next design: Class-ful Addresses

Used till the introduction of CIDR 1993



2^7 nets, 2^{24} hosts

1.0.0.0 to 127.255.255.255

2^{14} nets, 2^{16} hosts

128.0.0.0 to 191.255.255.255

2^{21} nets, 2^8 hosts

192.0.0.0 to 223.255.255.255

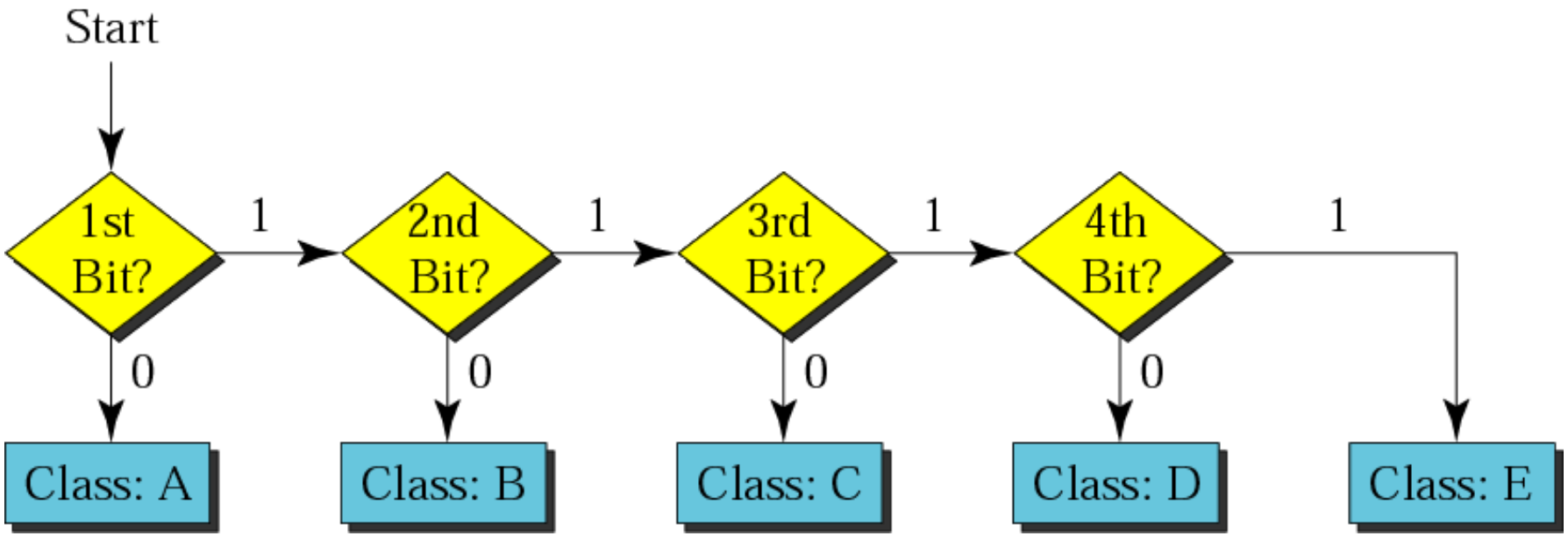
224.0.0.0 to 239.255.255.255

240.0.0.0 to 255.255.255.255

Problem: Networks only come in three sizes!

what if you wanted only 1 more than 2^8 hosts? well then you have no choice but to get 2 x C networks, or 1 B network, in either case you waste sooo many host addresses! WASTE! (even in the actual more efficient way we do it we still ran out!)

Finding the address class



i.e. the position of the first 0 tells you the class, starting from A

i.e. 00010101010 class A, 11001010 class C

What are the issues?

- An organization requires 6 nets each of size 30. Does it have to buy 6 class C address blocks?

  yes, and waste 256-30 per network :(

- An organization requires 512 addresses? How many IP addresses should it buy?

  class B, again waste :(

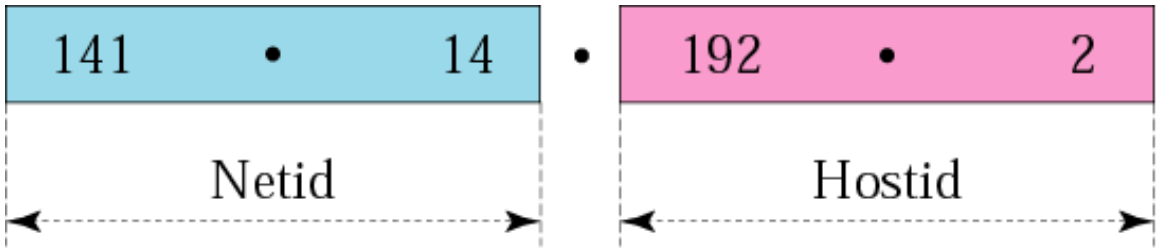
Subnetting

- Subnetting is the process of dividing the class A, B or C network into more manageable chunks that are suited to your network's size and structure.
- Subnetting allows 3 levels of hierarchy
  - netid, subnetid, hostid
- Original netid remains the same and designates the site
- Subnetting remains transparent outside the site only the site knows if it is subnetting???

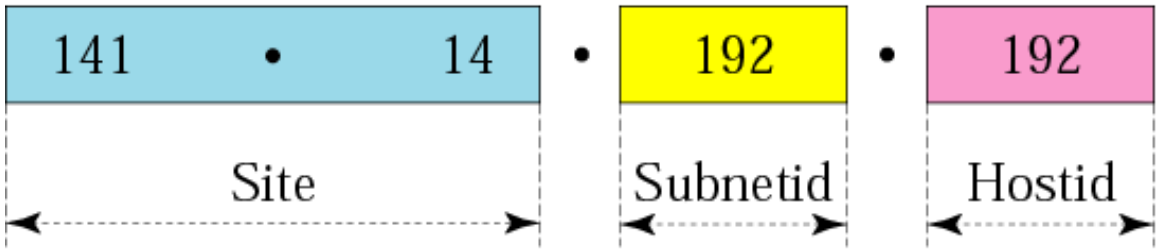
Subnetting

it is not the other way around because the netid is already fixed, set in stone!
i.e. you have been given some network address from a provider, and so you
are free to designate (own) any IP addresses that start with that netid
e.g. you buy netid 11011111.11011111.01xxxxxx.xxxxxxxx from them. Subnetting means
you aren't going to use all those x bits on hosts, but rather you will sacifice a chunk of them
so that you yourself become a distributer of netids e.g.
11011111.11011111.01yyyyyy.xxxxxxxx
where the y's are changed to represent networks and the x's are host bits. So you can now
hand out 64 networks, each with 2^8 host addresses each, rather than just one network with 2^14 host id's

- The process of subnetting simply extends the point where the 1's of Mask stop and 0's start
- You are sacrificing some host ID bits to gain Network ID bits



a. Without subnetting



b. With subnetting

Quiz?

A company is granted the site address 201.70.64.0 (class C). The company needs six subnets. Design the subnets.

The company needs six subnets. 6 is not a power of 2. The next number that is a power of 2 is 8 (2^3). We need 3 more 1s in the subnet mask. The total number of 1s in the subnet mask is 27 (24 + 3). The mask is

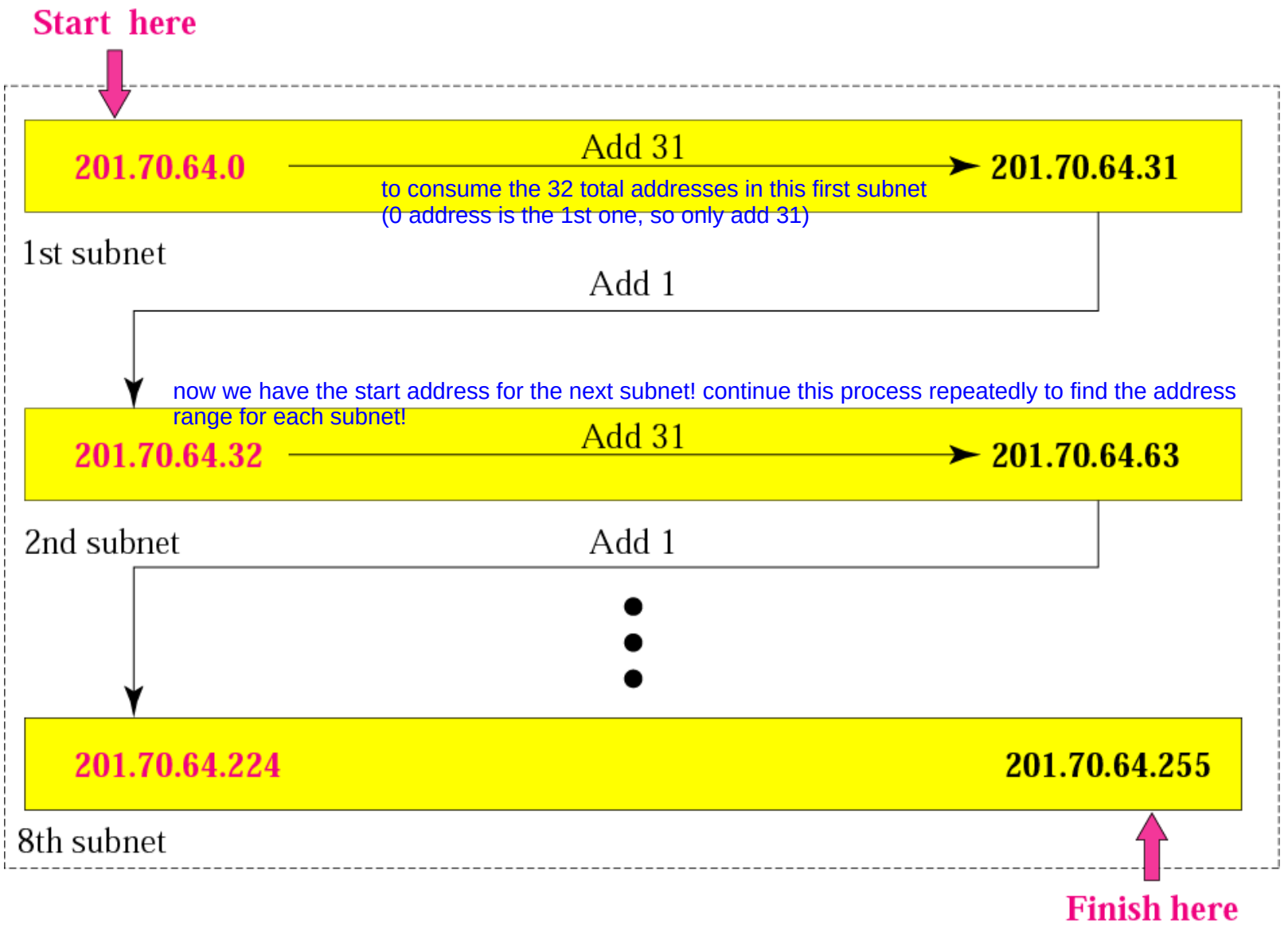
11111111 11111111 11111111 11100000

or 255.255.255.224

Number of addresses in each subnet = 2^5

= 32

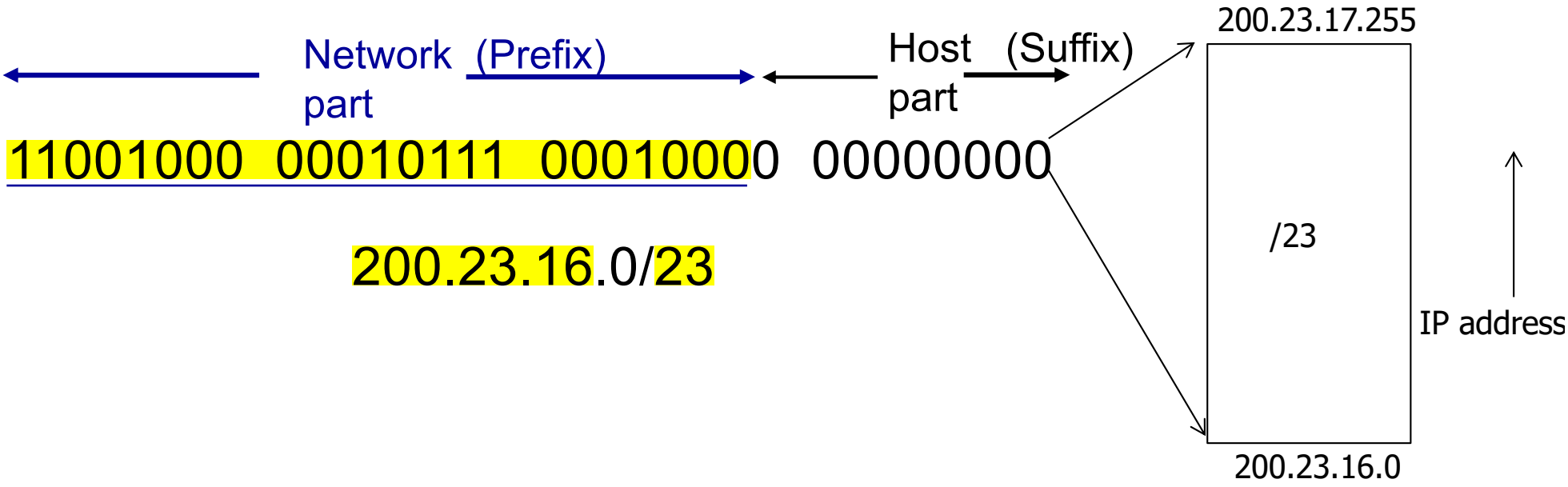
The number of addresses in each subnet is 2^5 or 32.



Today's addressing: CIDR

CIDR: Classless InterDomain Routing

- network portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in network portion of address



Quiz: IP Addressing

A small organization is given a block with the beginning address and the prefix length 205.16.37.24/29 (in slash notation). What is the range of the block?

The beginning address is 205.16.37.24. To find the last address we keep the first 29 bits and change the last 3 bits to 1s.

as they are the fixed network bits

largest host address

Beginning: 11001111 00010000 00100101 00011000
Ending : 11001111 00010000 00100101 00011111

There are only 8 addresses in this block.

205.16.37.24 to 205.16.37.31

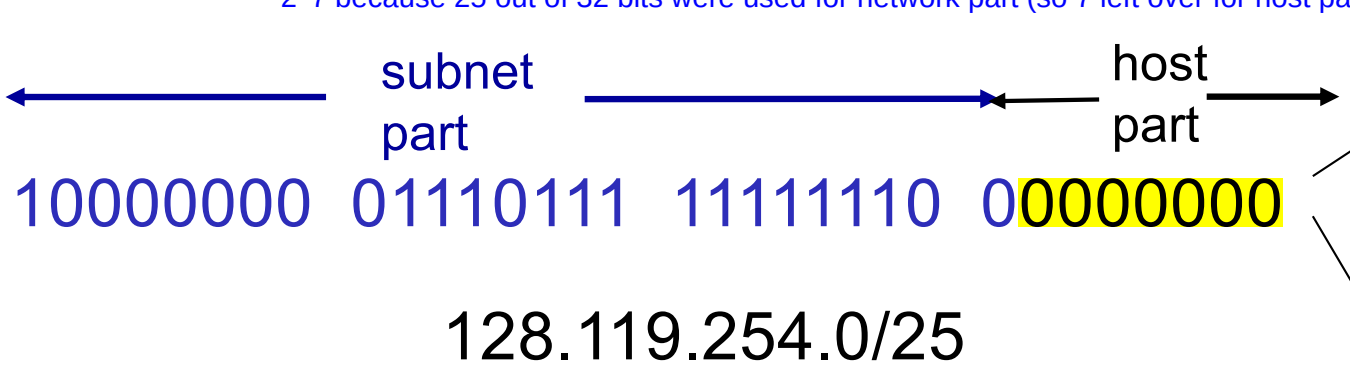
Quiz: IP Addressing

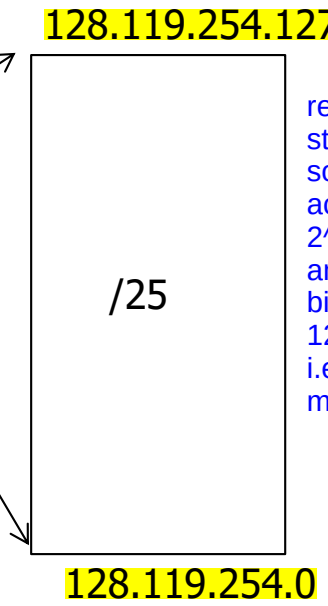
- How many IP addresses belong to the subnet 128.119.254.0/25 ? What are the IP addresses at the two end-points of this range ?

Answer: 2^7 = 128 addresses (126 are usable)

broadcast and network id addresses are reserved

2^7 because 25 out of 32 bits were used for network part (so 7 left over for host part)





remember binary starts from 2^0 and so the 8th LSB is actually representing 2^7. Now 2^7 is 128, and so all previous bits turned on is 128-1 = 127
i.e. first 7 bits as 1 means 127!

Quiz: IP Addressing

An ISP is granted a block of addresses starting with 190.100.0.0/16. The ISP needs to distribute these addresses to three groups of customers as follows:

1. The first group has 64 customers; each needs 256 addresses.
2. The second group has 128 customers; each needs 128 addresses.
3. The third group has 128 customers; each needs 64 addresses.

Design the sub-blocks and give the slash notation for each sub-block. Find out how many addresses are still available after these allocations.

Group 1

of the 64

For this group, each customer needs 256 addresses. This means the suffix length is 8 ($2^8 = 256$). The prefix length is then $32 - 8 = 24$.

i.e. first host address for first customer network — last host address for first customer network

01: 190.100.0.0/24 → 190.100.0.255/24

02: 190.100.1.0/24 → 190.100.1.255/24

first host address for second customer network — last host address for second customer network

.....

64: 190.100.63.0/24 → 190.100.63.255/24

all the way up to the 64th customer network

Total = 64 × 256 = 16,384

networks — addresses per network — total addresses

Group 2

of the 128

For this group, each customer needs 128 addresses. This means the suffix length is 7 ($2^7 = 128$). The prefix length is then $32 - 7 = 25$. The addresses are:

the network directly after the previous one ^

001: 190.100.64.0/25 → 190.100.64.127/25

002: 190.100.64.128/25 → 190.100.64.255/25

.....

128: 190.100.127.128/25 → 190.100.127.255/25

Total = $128 \times 128 = 16{,}384$

Group 3

For this group, each customer needs 64 addresses. This means the suffix length is 6 ($2^6 = 64$). The prefix length is then $32 - 6 = 26$.

001:190.100.128.0/26 ➔ 190.100.128.63/26

002:190.100.128.64/26 ➔ 190.100.128.127/26

.....

128:190.100.159.192/26 ➔ 190.100.159.255/26

Total = $128 \times 64 = 8{,}192$

Number of granted addresses: 65,536

Number of allocated addresses: 40,960

Number of available addresses: 24,576

IP addresses: how to get one?

Q: How does a *host* get IP address?

- hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol: dynamically get address from as server
  - "plug-and-play"

DHCP

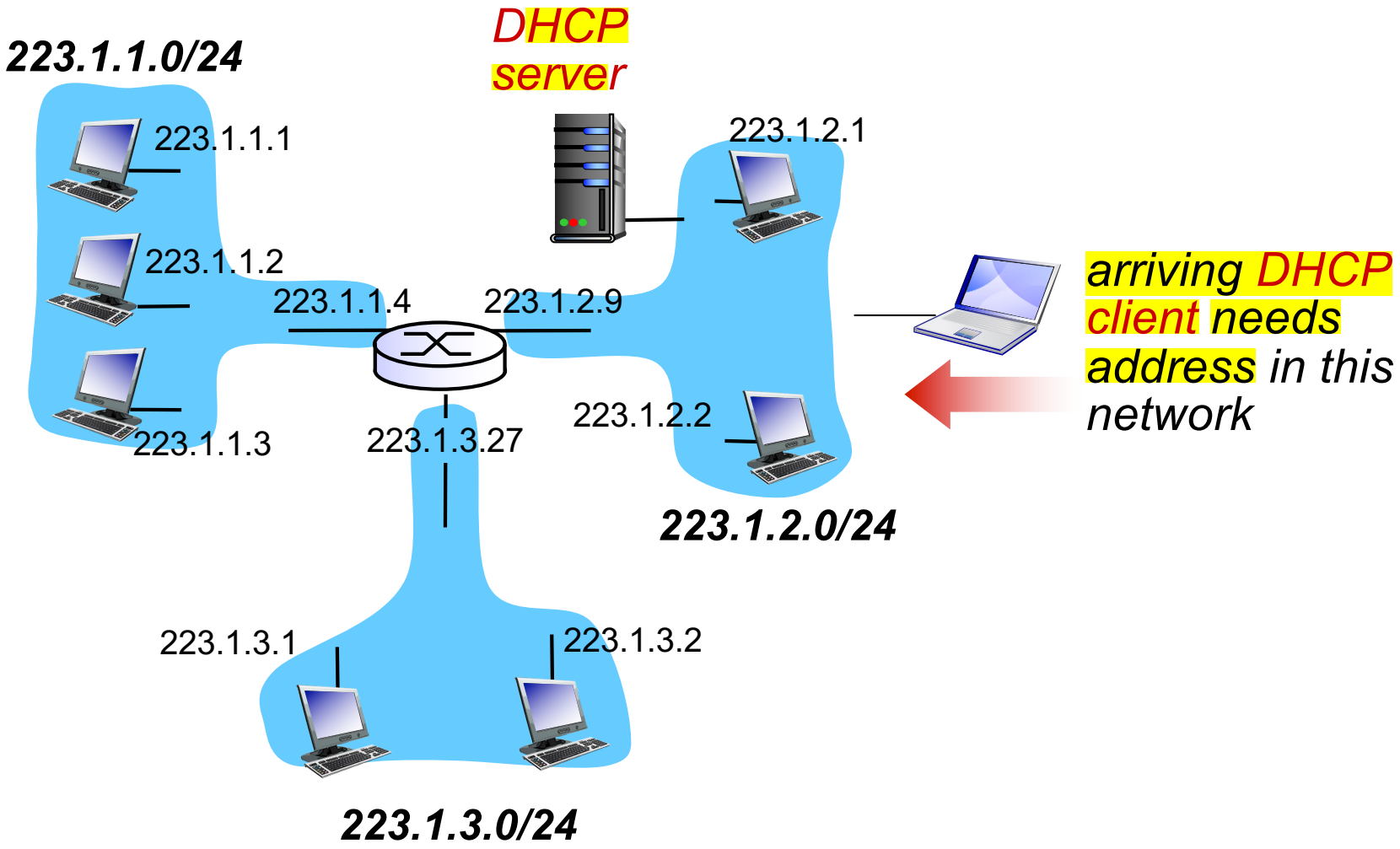
goal: allow host to *dynamically* obtain its IP address from network server when it joins network

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

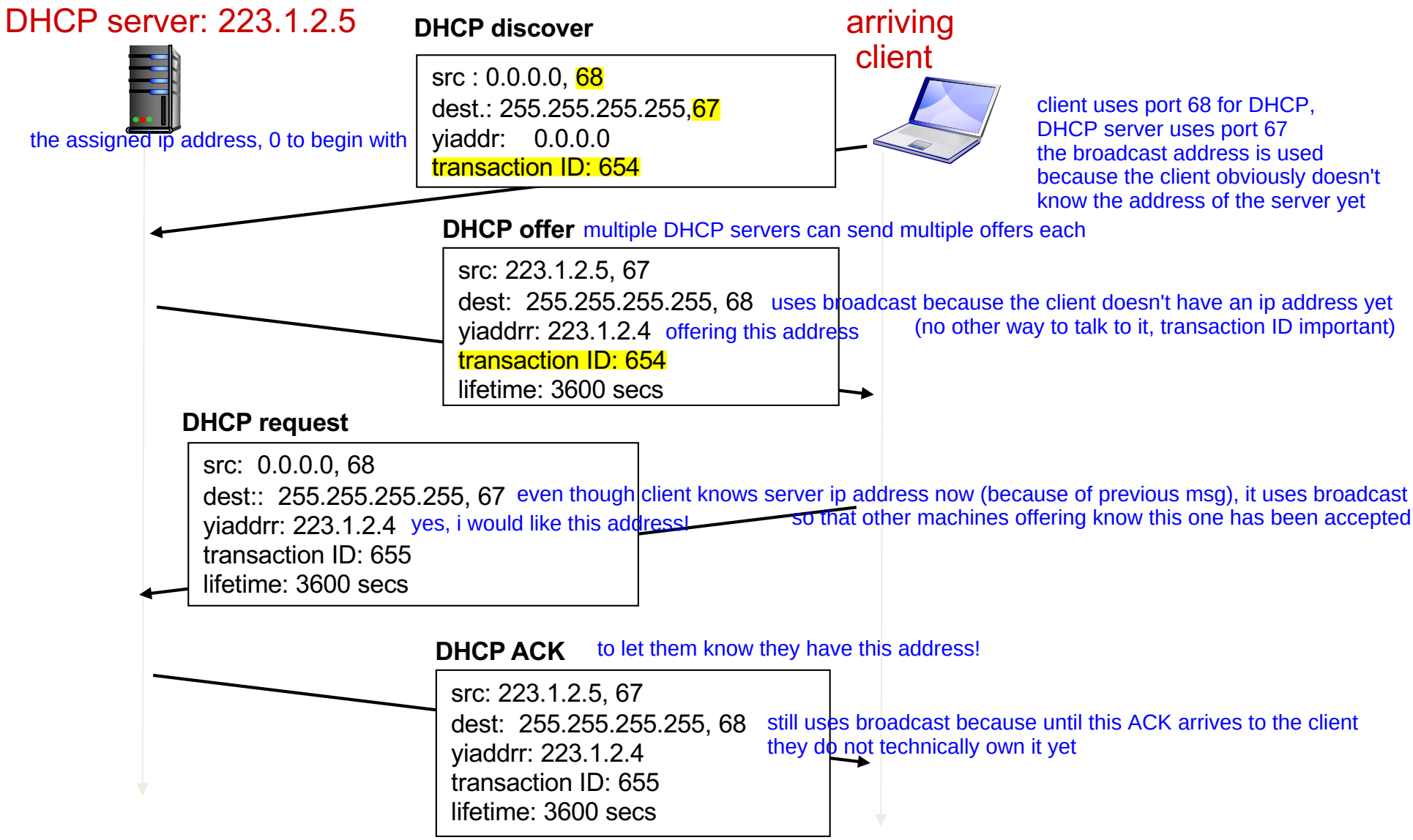
DHCP overview:

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

DHCP client-server scenario



DHCP client-server scenario

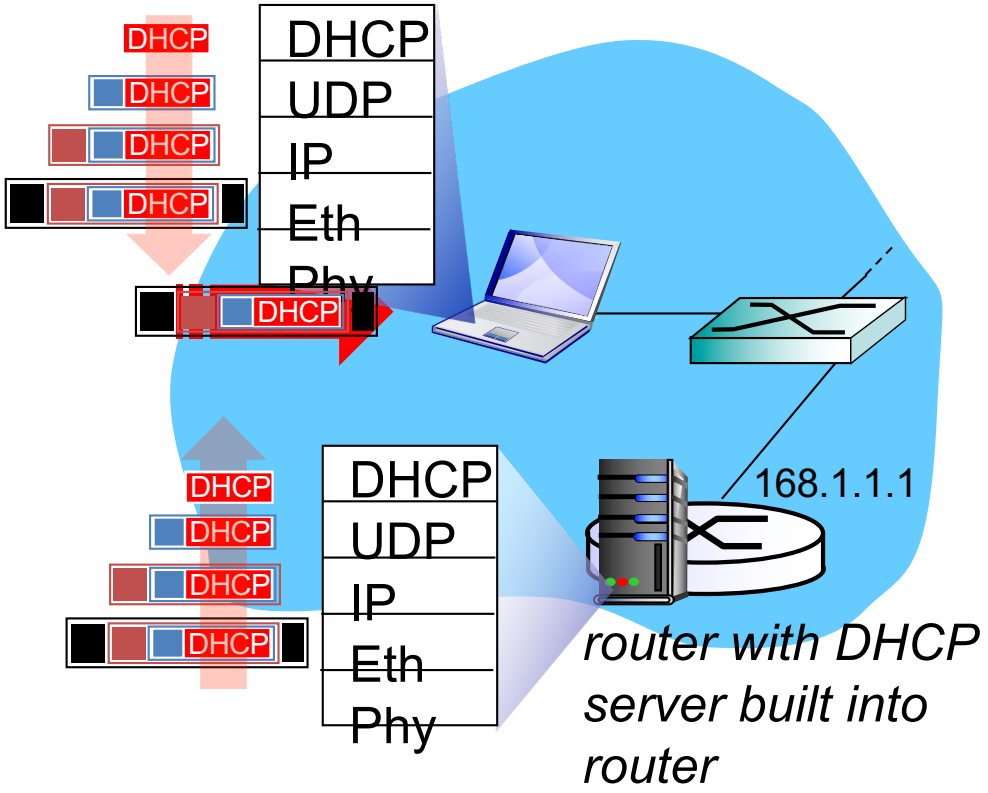


DHCP: more than IP addresses

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

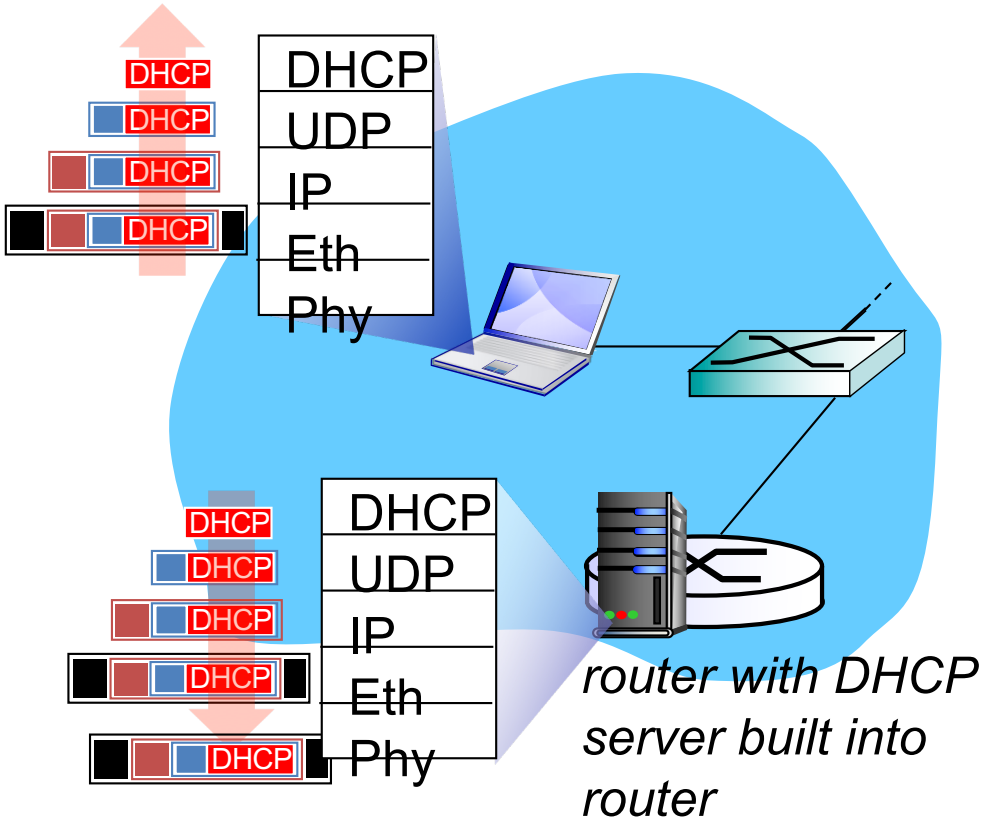
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

DHCP: example



- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.1 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

DHCP: example



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

DHCP: Wireshark output (home LAN)

request

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
    Length: 7; Value: 010016D323688A;
    Hardware type: Ethernet
    Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
    Length: 11; Value: 010F03062C2E2F1F21F92B
    1 = Subnet Mask; 15 = Domain Name
    3 = Router; 6 = Domain Name Server
    44 = NetBIOS over TCP/IP Name Server
    ……

reply

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
    Length: 12; Value: 445747E2445749F244574092;
    IP Address: 68.87.71.226;
    IP Address: 68.87.73.242;
    IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

DHCP: further details

- DHCP uses UDP and port numbers 67 (server side) and 68 (client side)
- Usually the MAC address is used to identify clients
  - DHCP server can be configured with a "registered list" of acceptable MAC addresses
- DHCP offer message includes ip address, length of lease, subnet mask, DNS servers, default gateway
- DHCP security holes
  - DoS attack by exhausting pool of IP addresses
  - Masquerading as a DHCP server
  - Authentication for DHCP - RFC 3118

IP addresses: how to get one?

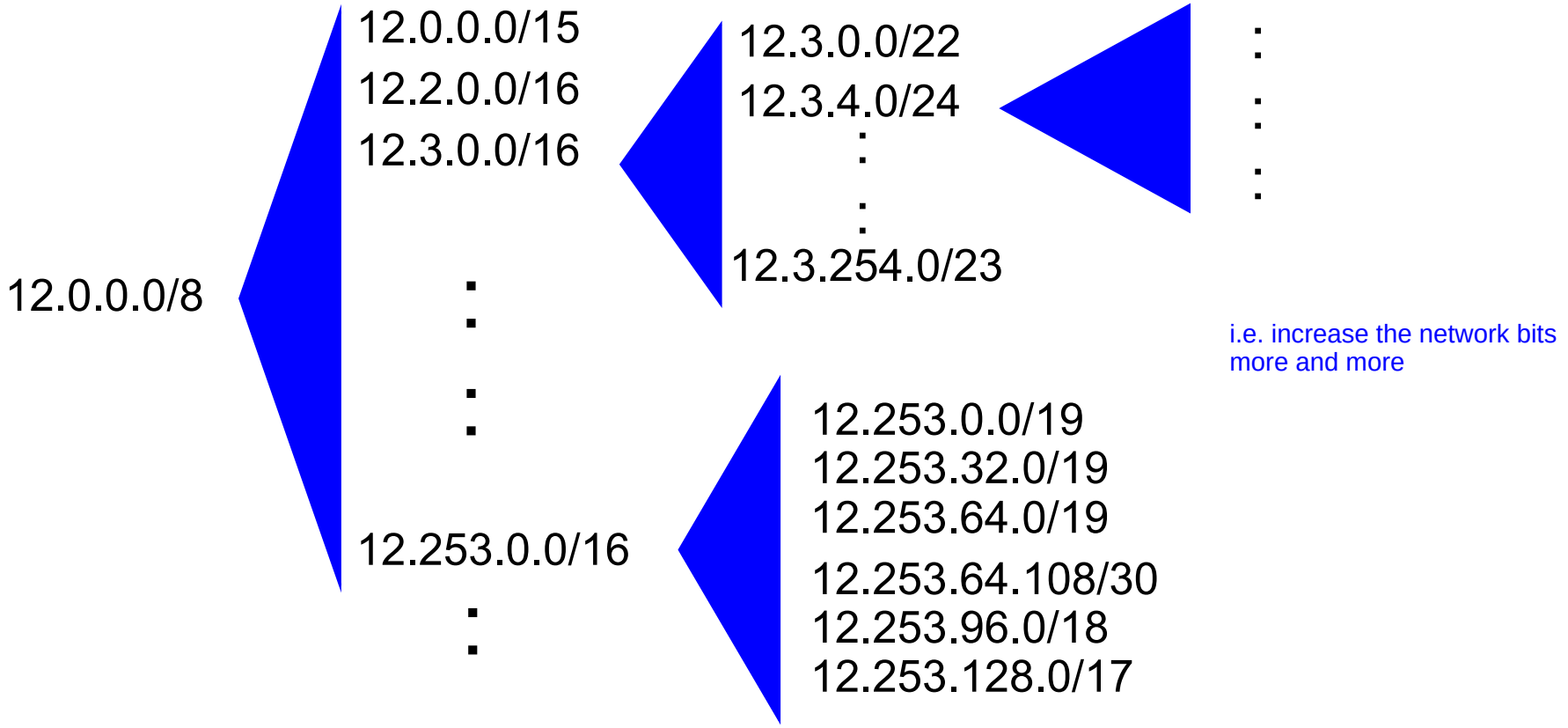
Q: how does *network* get subnet part of IP addr?

A: gets allocated portion of its provider ISP's address space

ISP's block	11001000 00010111 00010000	00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000	00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010	00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 00011110	00000000	200.23.30.0/23

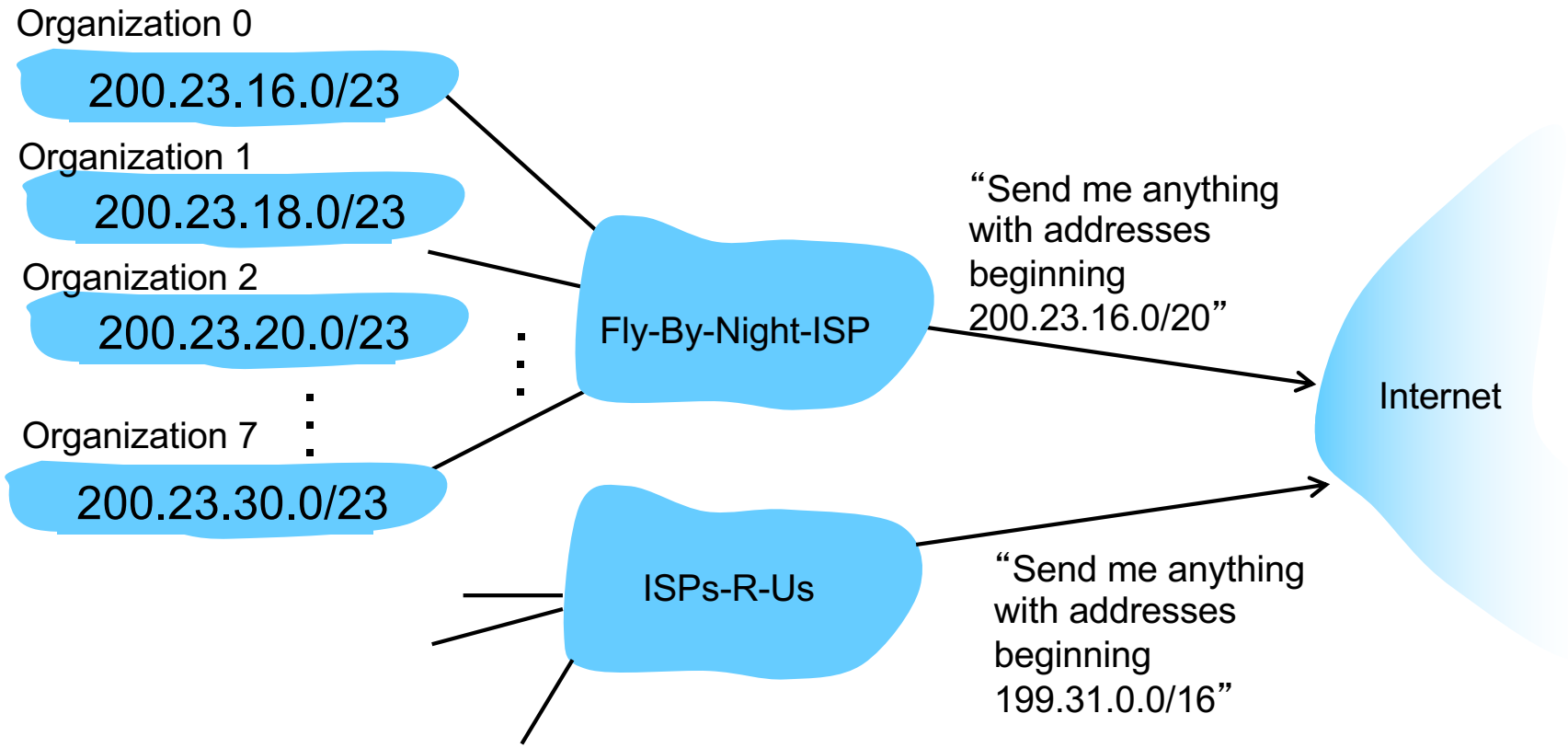
CIDR: Addresses allocated in contiguous prefix chunks

Recursively break down chunks as get closer to host

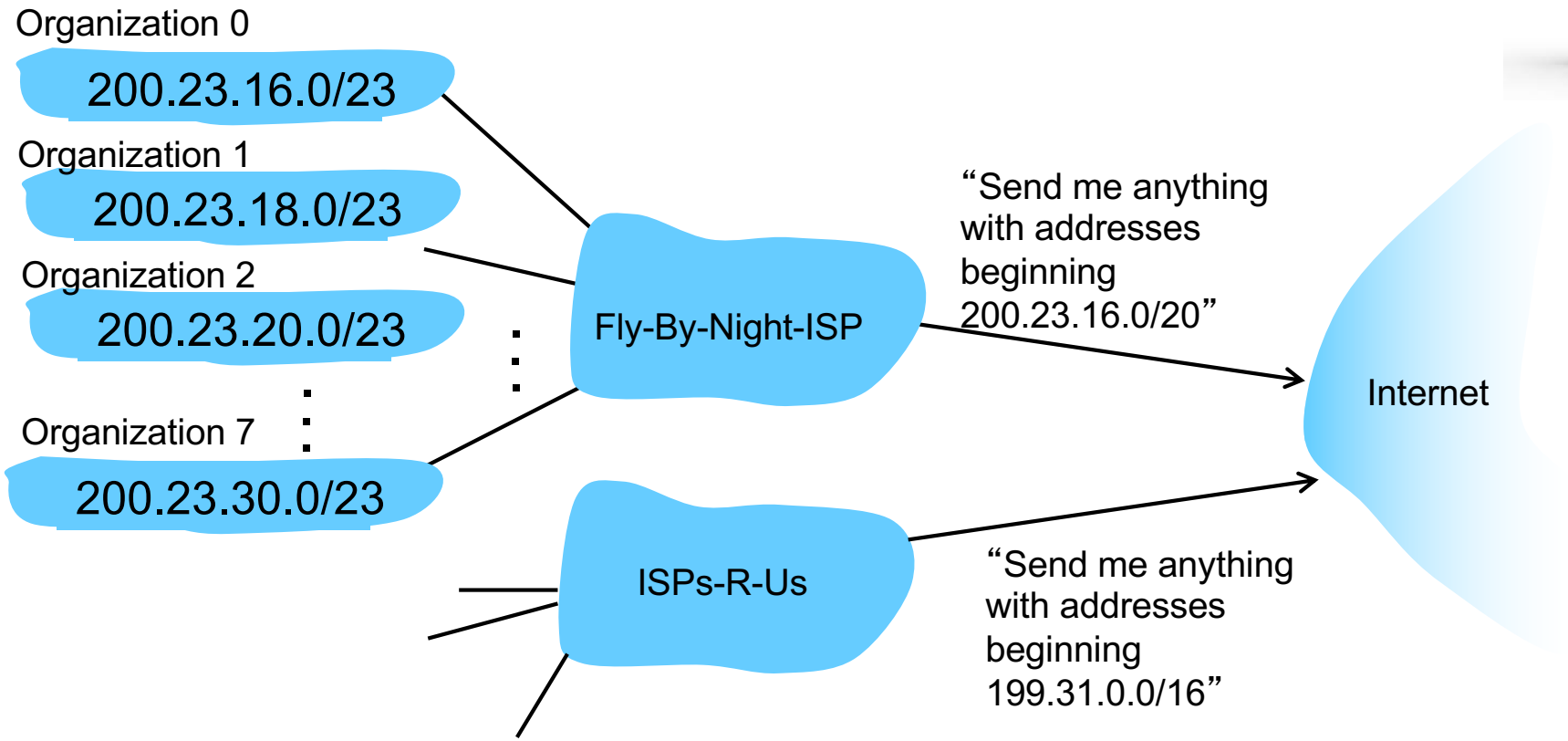


Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



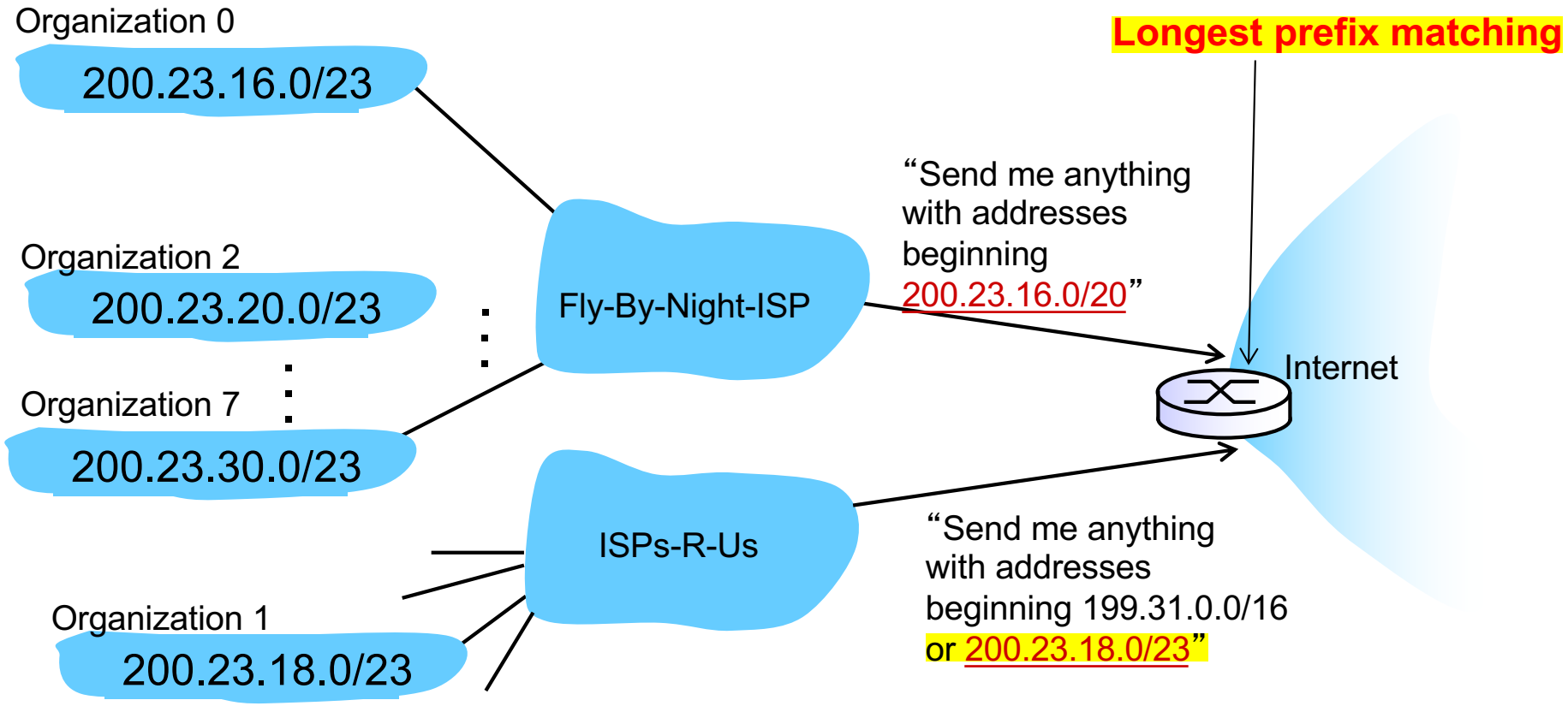
Quiz: What should we do if organization 1 decides to switch to ISPs-R-Us



A: Move 200.23.18.0/23 to ISPs-R-Us (and break up Fly-By-Night's/20 block).
B: Give new addresses to Organization 1 (and force them to change all their addresses)
C: Some other solution

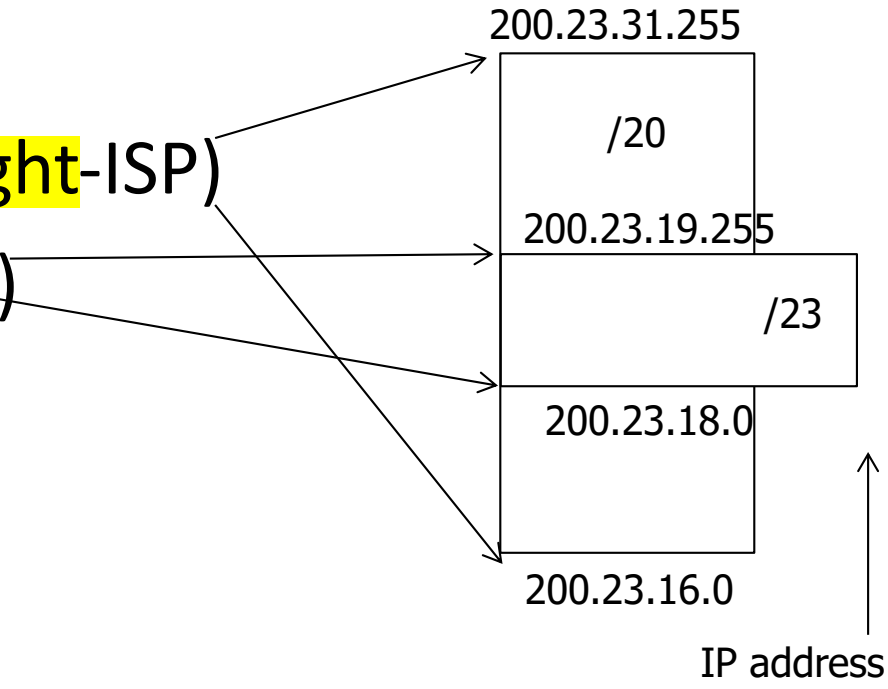
Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1



Example: continued

- But how will this work?
- Routers in the Internet will have two entries in their tables
  - 200.23.16.0/20 (Fly-by-Night-ISP)
  - 200.23.18.0/23 (ISPs-R-Us)
- Longest prefix match



Longest prefix matching

longest prefix matching

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

Destination Address Range	Link interface
11001000 00010111 00010*** *********	0
11001000 00010111 00011000 *********	1
11001000 00010111 00011*** *********	2
otherwise	3

examples:

DA: 11001000 00010111 00010110 10100001 — which interface?

DA: 11001000 00010111 00011000 10101010 — which interface?

Quiz: Longest prefix matching

- On which outgoing interface will a packet destined to 11011001 be forwarded?

Prefix	Interface
1*	A
11*	B
111*	C
Default	D

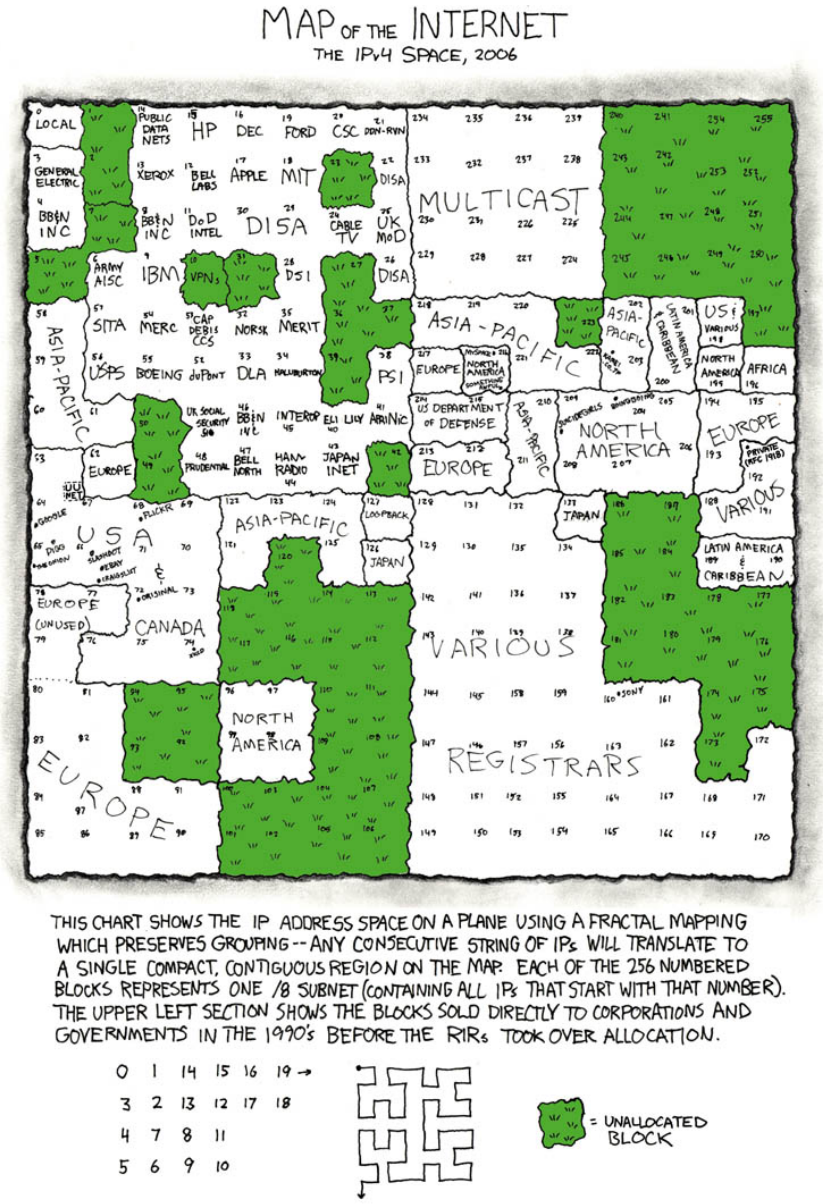
www.zeetings.com/salil

More on IP addresses

- IP addresses are allocated as blocks and have geographical significance
- It is possible to determine the geographical location of an IP address

http://www.geobytes.com/IpLocator.htm

Source: www.xkcd.com



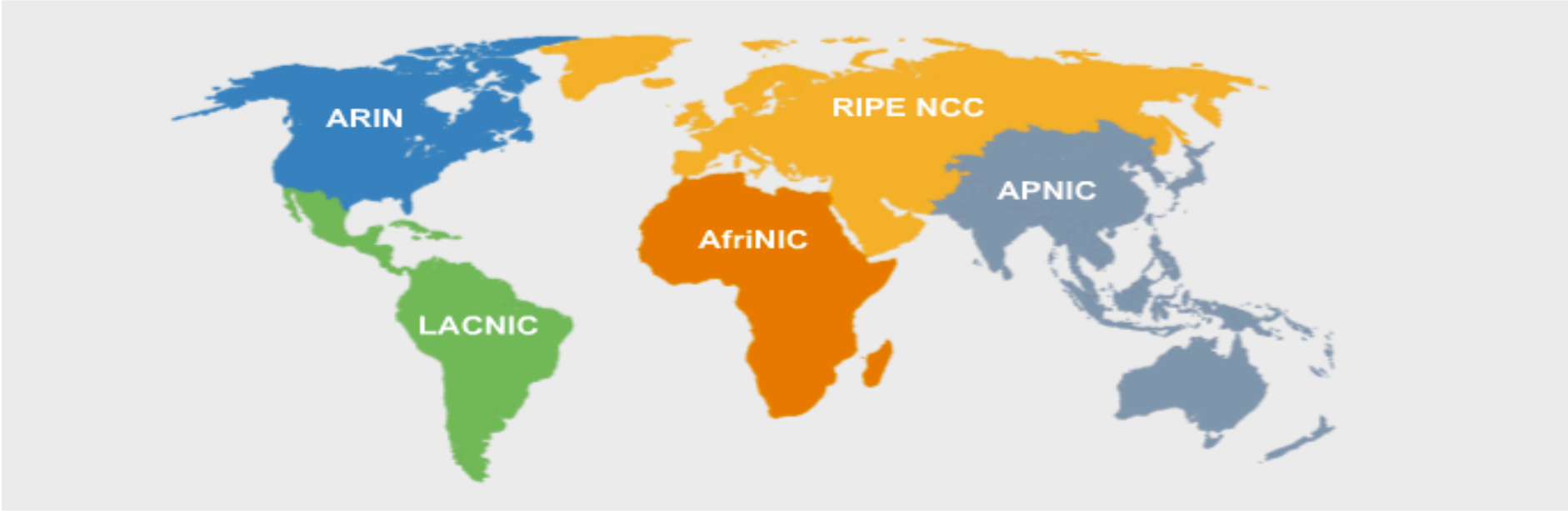
IP Addressing: the last word...

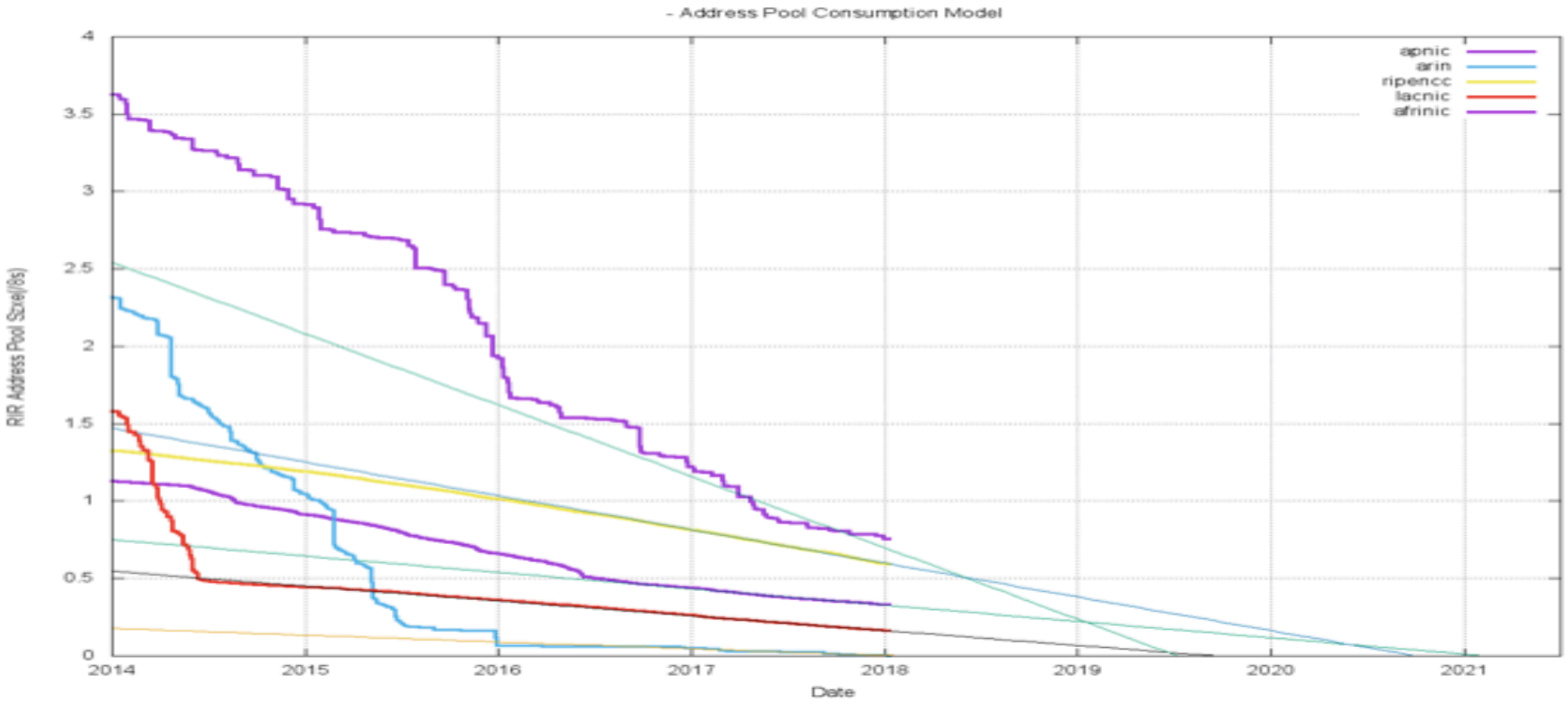
Q: How does an ISP get block of addresses?

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

IANA works through Regional Internet Registries (RIRs):

- ARIN — American Registry for Internet Numbers
- LACNIC — Latin America and Caribbean Network Information Centre
- RIPE NCC — IRéseaux IP Européens Network Coordination Centre
- APNIC — Asia-Pacific Network Information Center
- AfriNIC — African Network Information Centre

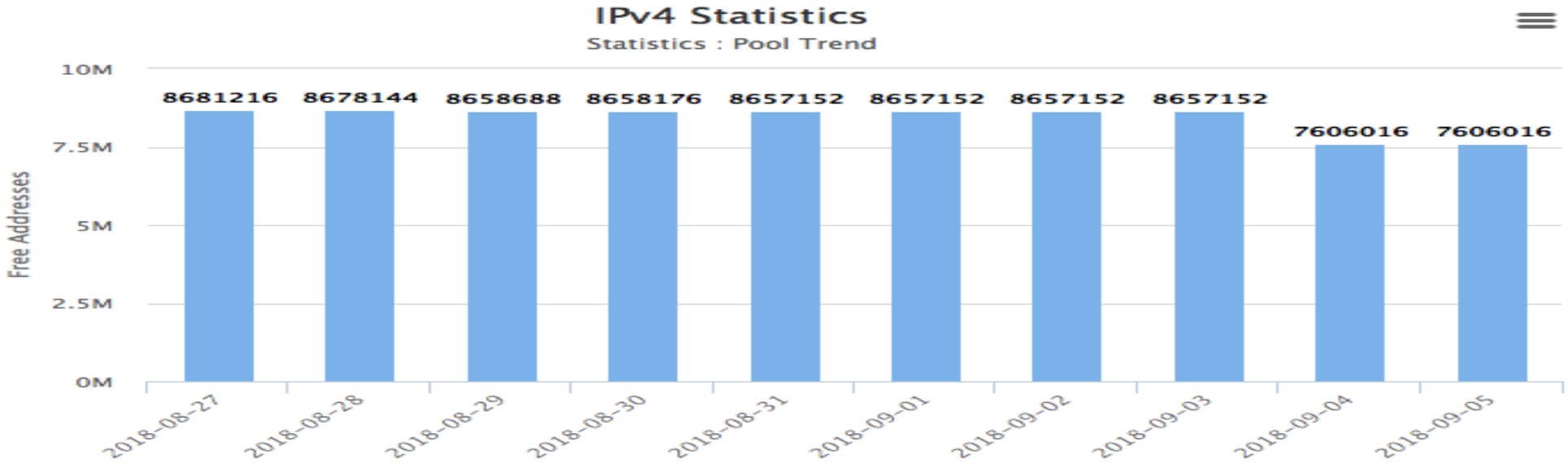




IPv4 Exhaustion

In this section you can find statistics for IPv4 pool in the AfriNIC region.

IPv4 Usage per IANA allocation **IPv4 available space over time** **IPv4 availability by prefix size**



Made-up Example in More Detail

- ICANN gives APNIC several /8s
- APNIC gives Telstra one /8, **129/8**
  - Network Prefix**: 10000001**
- Telstra gives UNSW a /16, **129.94/16**
  - Network Prefix**: 1000000101011110**
- UNSW gives CSE a /24, **129.94.242/24**
  - Network Prefix**: 100000010101111011110010**
- CSE gives me a specific address **129.94.242.51**
  - Address: **10000001010111101111001000110011**

Network Layer, data plane: outline

4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router

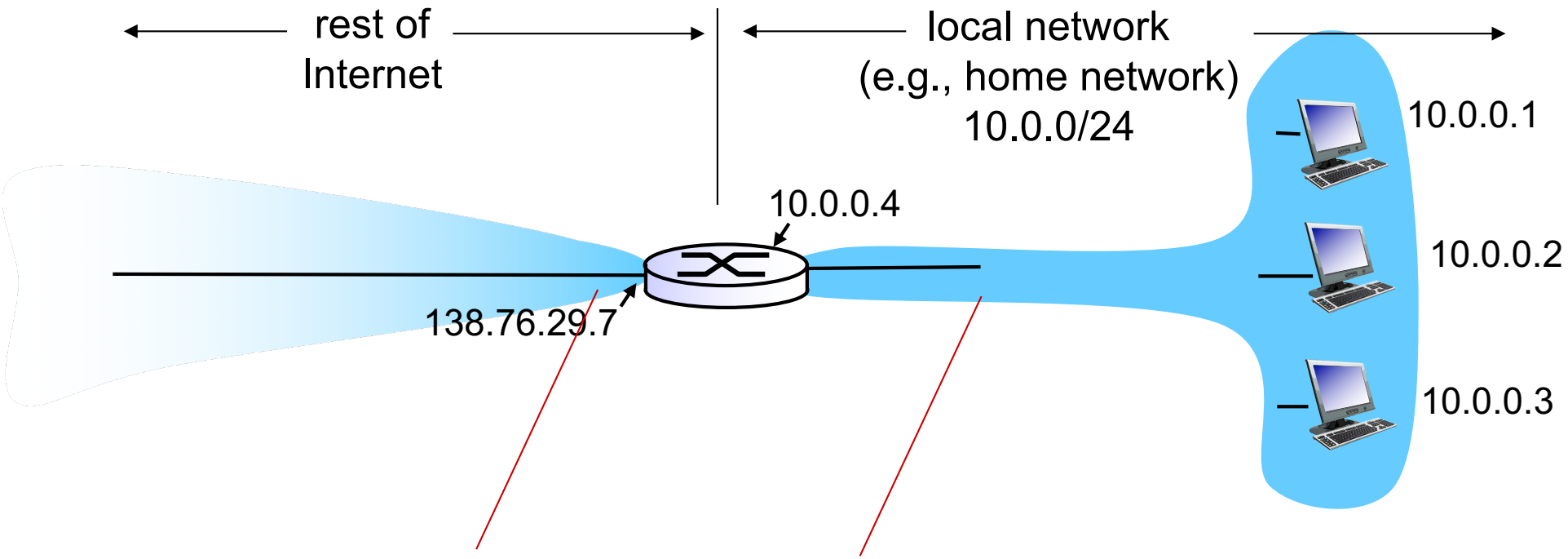
4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

Private Addresses

- Defined in RFC 1918:
  - 10.0.0.0/8 (16,777,216 hosts)
  -172.16.0.0/12 (1,048,576 hosts)
  -192.168.0.0/16 (65536 hosts)

- These addresses cannot be routed
  - Anyone can use them
  -Typically used for NAT

NAT: network address translation



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

datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: network address translation

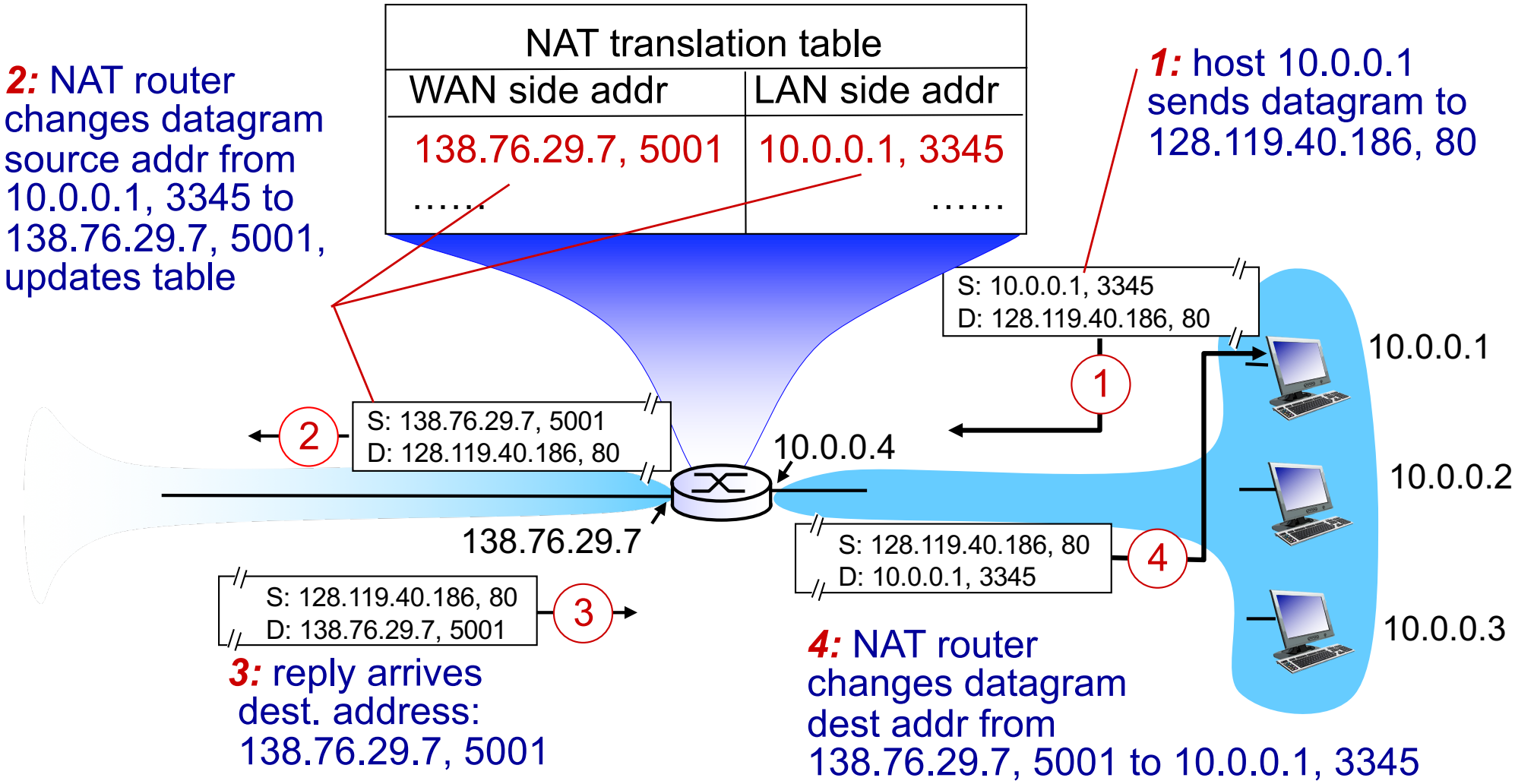
implementation: NAT router must:

- *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 addr

- *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 dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: network address translation



NAT Advantages

Local network uses just one IP address as far as outside world is concerned:

- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network

NAT Disadvantages

- NAT violates the architectural model of IP
  - Every IP address uniquely identifies a single node on Internet
  - routers should only process up to layer 3
- NAT changes the Internet from connection less to a kind of connection oriented network

Discussion: NAT

- Devices inside the local network are not explicitly addressable or visible by outside world.

  A: This is an advantage

  B: This is a disadvantage

NAT: network address translation

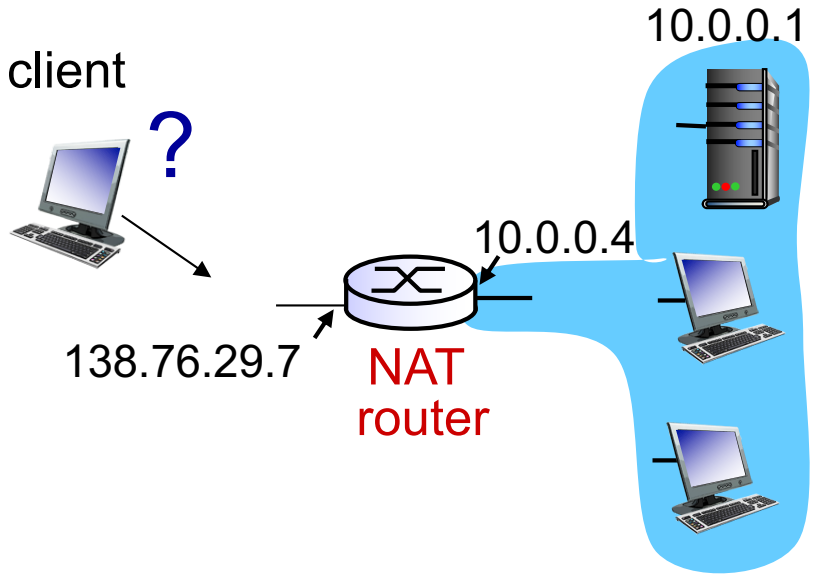
- 16-bit port-number field:
  - ~65,000 simultaneous connections with a single WAN-side address!
- NAT is controversial:
  - routers should only process up to layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, e.g., P2P applications
  - address shortage should instead be solved by IPv6

NAT: Practical Issues

- NAT modifies port # and IP address
  - *Requires recalculation of TCP and IP checksum*
- Some applications embed IP address or port numbers in their message payloads
  - DNS, FTP (PORT command), SIP, H.323
  - For legacy protocols, NAT must look into these packets and translate the embedded IP addresses/port numbers
  - Duh, What if these fields are encrypted ?? (SSL/TLS, IPSEC, etc.)
  - **Q: In some cases why may NAT need to change TCP sequence number??**
- E.g: If applications change port numbers periodically, the NAT must be aware of this
- NAT Traversal Problems
  - How to setup a server behind a NAT router?
  - How to talk to a Skype user behind a NAT router?

NAT traversal problem

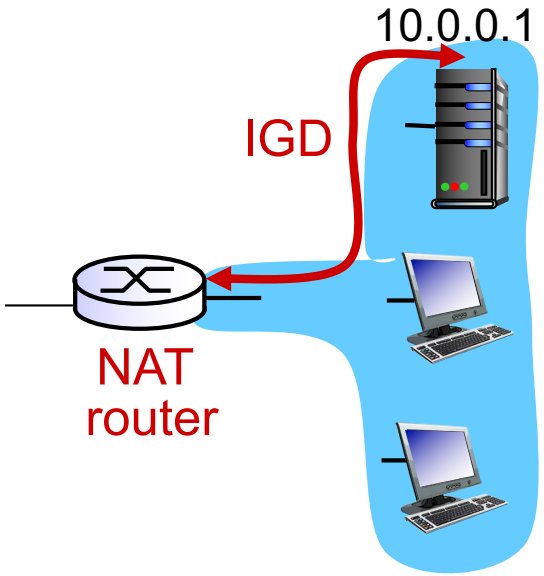
- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATed address: 138.76.29.7
- *solution1:* Inbound-NAT. Statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000



NAT traversal problem

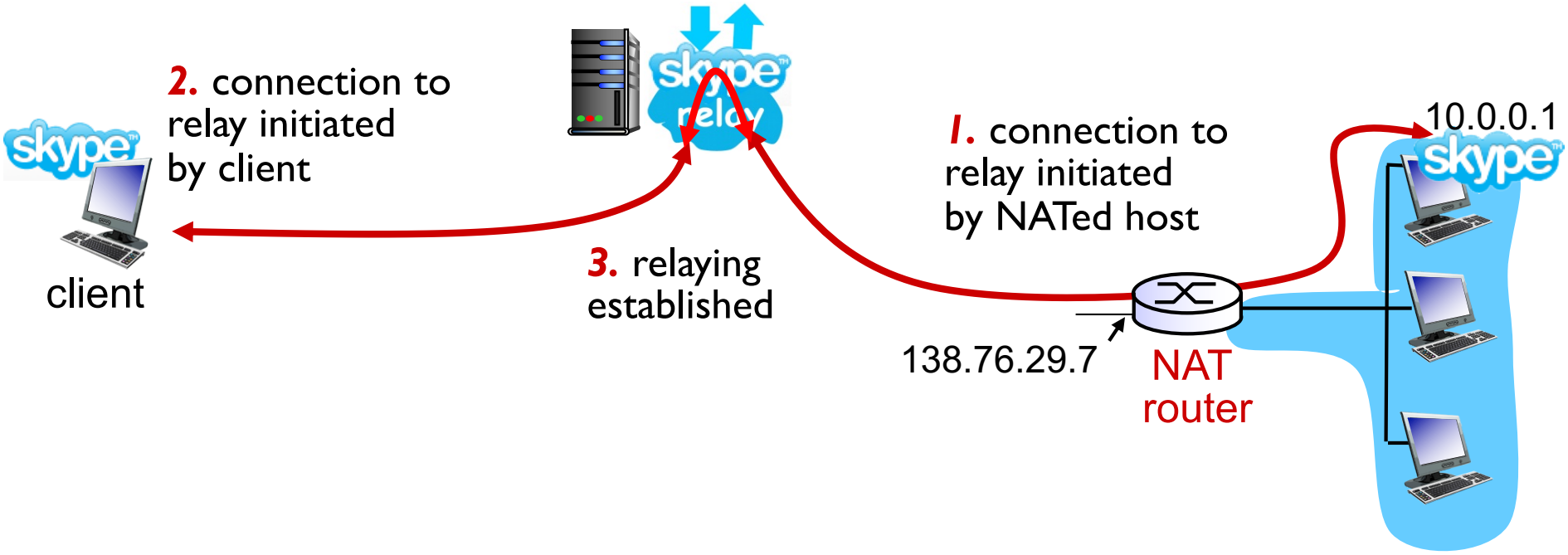
- *solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - learn public IP address (138.76.29.7)
  - add/remove port mappings (with lease times)

  i.e., automate static NAT port map configuration



NAT traversal problem

- *solution 3:* relaying (used in Skype)
  - NATed client establishes connection to relay
  - external client connects to relay
  - relay bridges packets between to connections

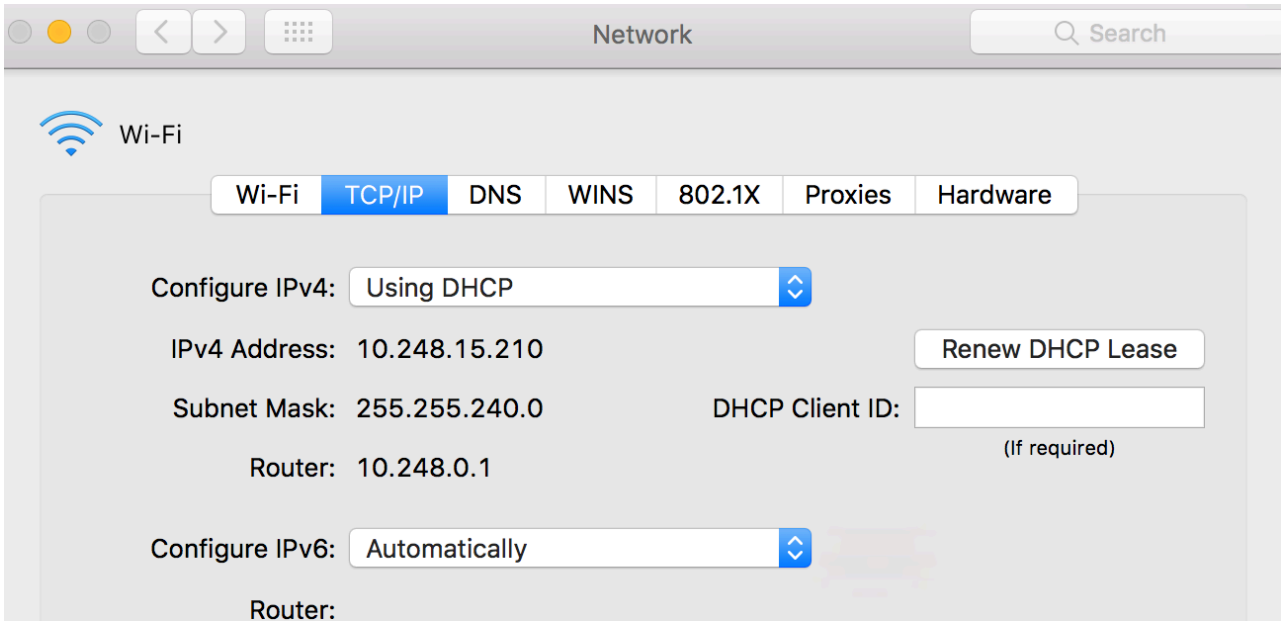


NAT: Devil in the details

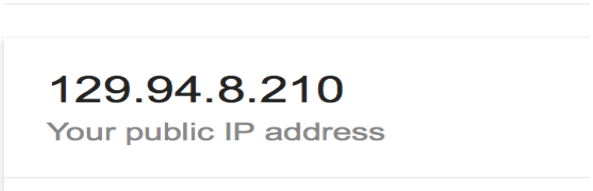
- Despite the problems, NAT has been widely deployed
- Most protocols can be successfully passed through a NAT, including VPN
- Modern hardware can easily perform NAT functions at > 100 Mbps
- IPv6 is still not widely deployed commercially, so the need for NAT is real
- After years of refusing to work on NAT, the IETF has been developing “NAT control protocols” for hosts
- Lot of practical variations
  - Full-cone NAT, Restricted Cone NAT, Port Restricted Cone NAT, Symmetric NAT, …..
    - The devil is in the detail

Discussion

- The picture below shows you the IP address of my machine connected to the uniwide wireless network.



- However when I ask Google it says my IP address is as noted below. Can you explain the discrepancy?



Quiz: NAT

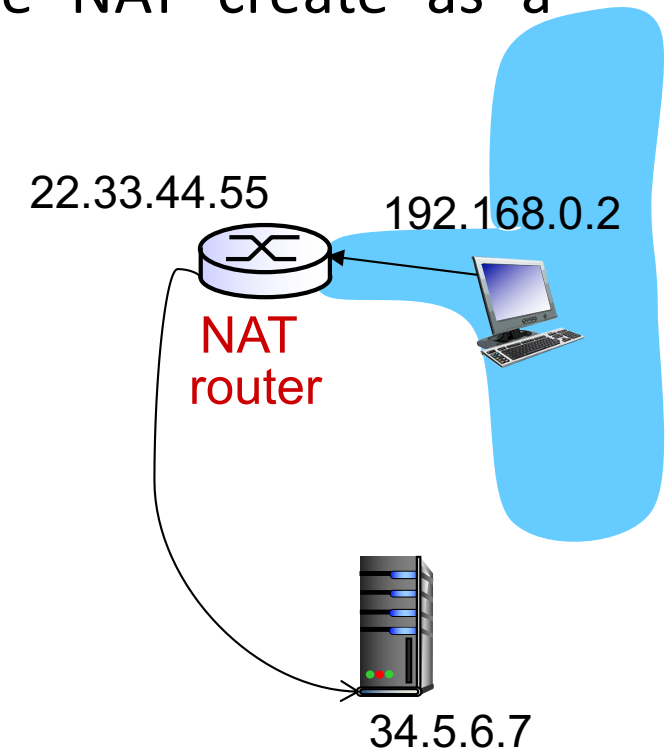
- A host with a private IP address 192.168.0.2 opens a TCP socket on its local port 4567 and connects to a web server at 34.5.6.7. The NAT's public IP address is 22.33.44.55. Which of the following mapping entries *could* the NAT create as a result?

A. [22.33.44.55, 4567]→[192.168.0.2, 80]

B. [34.5.6.7, 80] → [22.33.44.55, 4567]

C. [192.168.0.2, 80]→[34.5.6.7, 4567]

D. [22.33.44.55, 3967]→[192.168.0.2, 4567]



Quiz: NAT

- A host with a private IP address 192.168.0.2 opens a TCP socket on its local port 4567 and connects to a web server at 34.5.6.7. The NAT's public IP address is 22.33.44.55. Suppose the NAT created the mapping [22.33.44.55, 3967]→[192.168.0.2, 4567] as a result. What are the source and destination port numbers in the SYN-ACK response from the server?

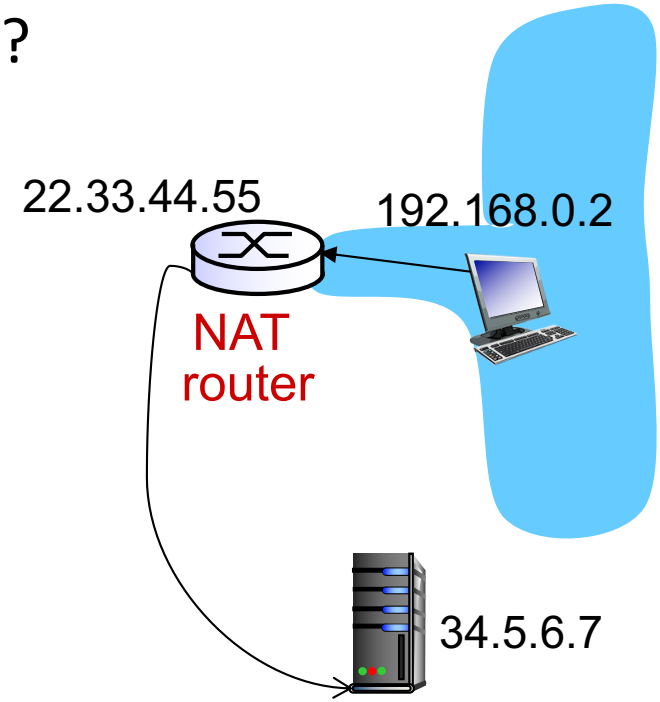
A. 80, 3967

B. 4567, 80

C. 3967, 80

D. 3967, 4567

E. 80, 4567



Network Layer, data plane: outline

4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router

4.3 IP: Internet Protocol
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

IPv6: motivation

- *initial motivation:* 32-bit address space soon to be completely allocated.
- additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

IPv6 datagram format:

- fixed-length 40 byte header
- no fragmentation allowed

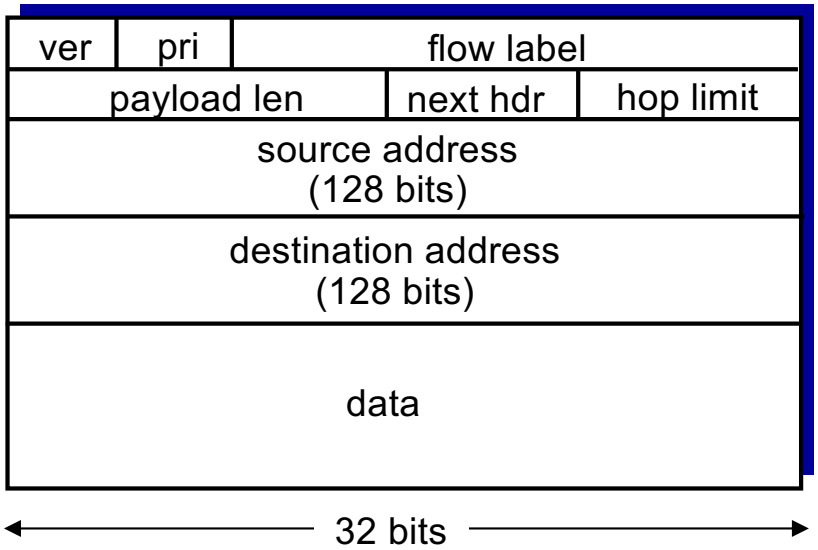
https://www.google.com/intl/en/ipv6/statistics.html

IPv6 datagram format

priority: identify priority among datagrams in flow (traffic class)

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

next header: identify upper layer protocol for data

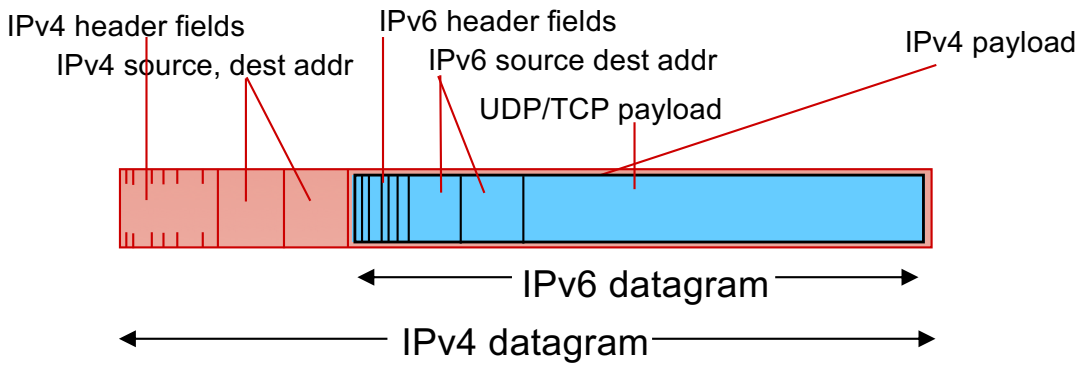


Other changes from IPv4

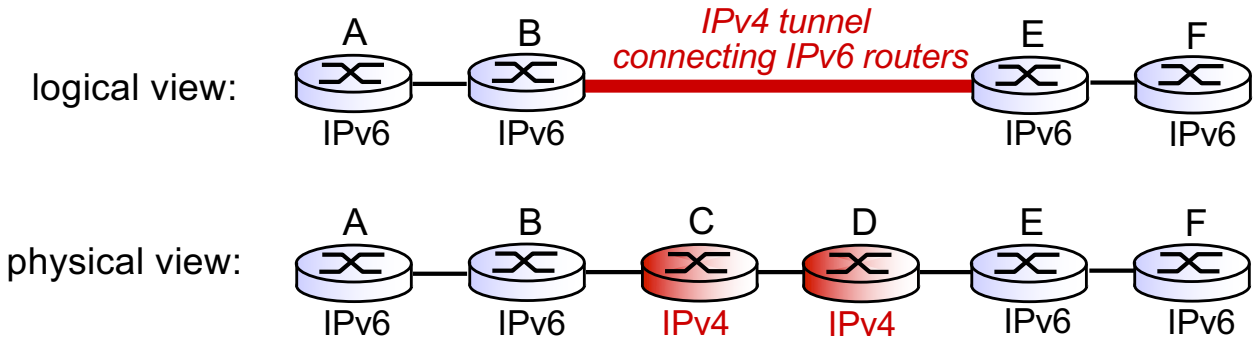
- *checksum*: removed entirely to reduce processing time at each hop
- *options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition from IPv4 to IPv6

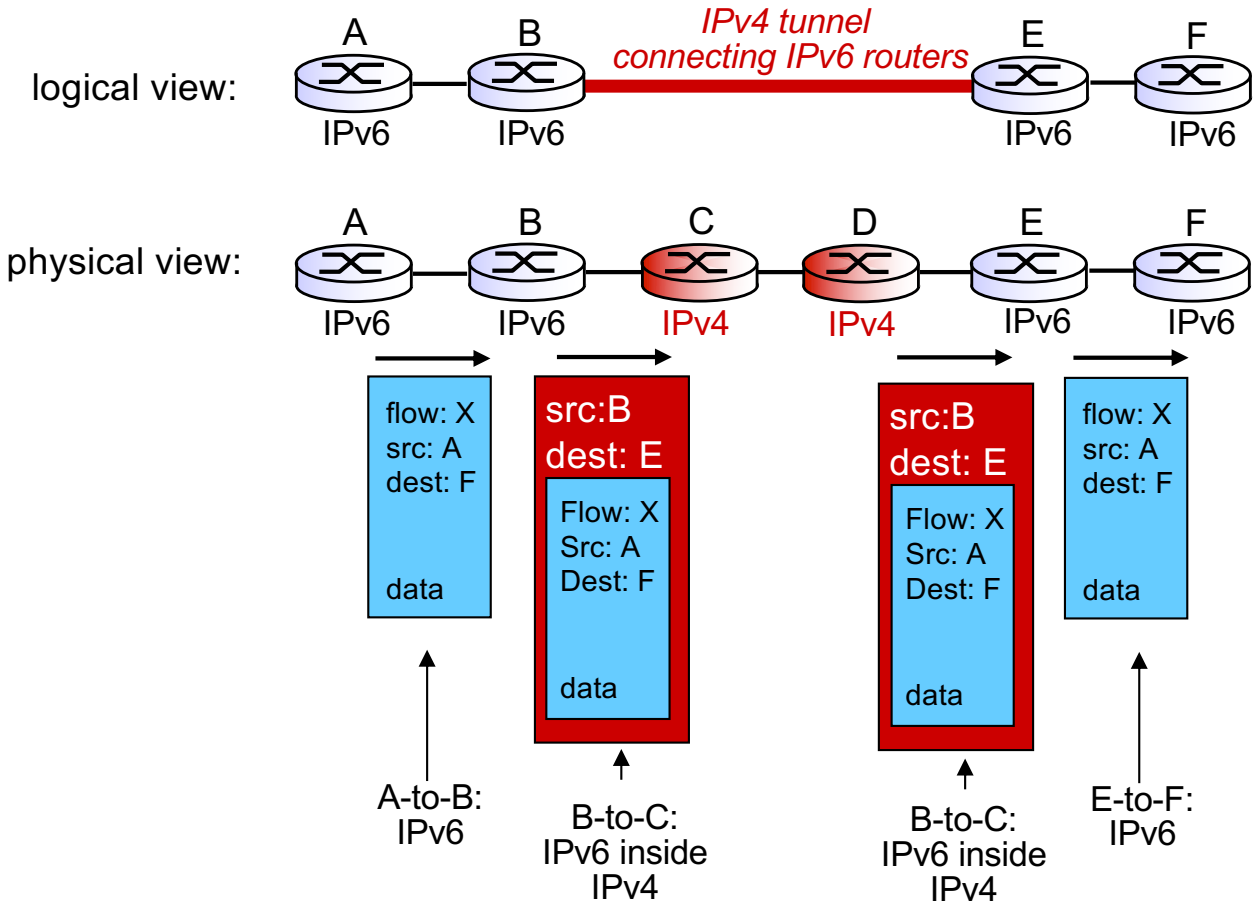
- not all routers can be upgraded simultaneously
  - no "flag days"
  - how will network operate with mixed IPv4 and IPv6 routers?
- *tunneling:* IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers



Tunneling (IPv6 over IPv4)



Tunneling (IPv6 over IPv4)



Tunneling (IPv4 over IPv4)

Used in VPNs