

# COMP 3331/9331: Computer Networks and Applications

1. IP datagram header format
2. IP fragmentation
3. IP addressing, subnets, network mask, longest prefix match, DHCP
4. Network address translation (NAT)

Week 7

Network Layer: Data Plane

Reading Guide: Chapter 4: Sections 4.1, 4.3

# Network Layer: outline

*Our goals:*

- understand principles behind network layer services, focusing on data plane:
  - network layer service models
  - forwarding versus routing
  - addressing
- instantiation, implementation in the Internet
  - IP, NAT, ICMP

# 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)

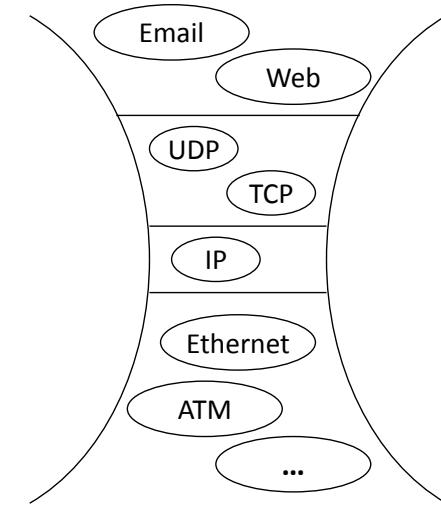
– **Not Covered**

# Some Background

- 1968: DARPAnet/ARPAnet (precursor to Internet)
  - (Defense) Advanced Research Projects Agency Network
- Mid 1970's: new networks emerge
  - SATNet, Packet Radio, Ethernet
  - All “islands” to themselves – didn't work together
- Big question: How to connect these networks?

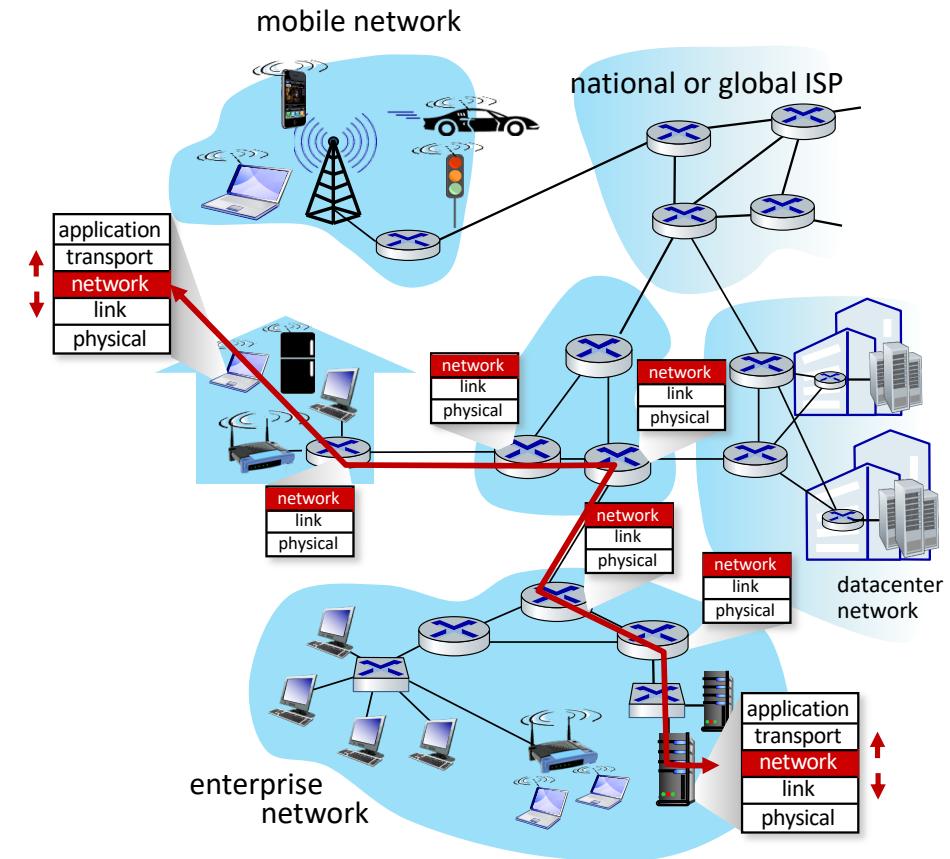
# Internetworking

- 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 services and protocols

- transport segment from sending to receiving host
  - **sender:** encapsulates segments into datagrams, passes to link layer
  - **receiver:** delivers segments to transport layer protocol
- network layer protocols in *every Internet device*: hosts, routers
- **routers:**
  - examines header fields in all IP datagrams passing through it
  - moves datagrams from input ports to output ports to transfer datagrams along end-end path



# Two key network-layer functions

## network-layer functions:

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

## analogy: taking a trip

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

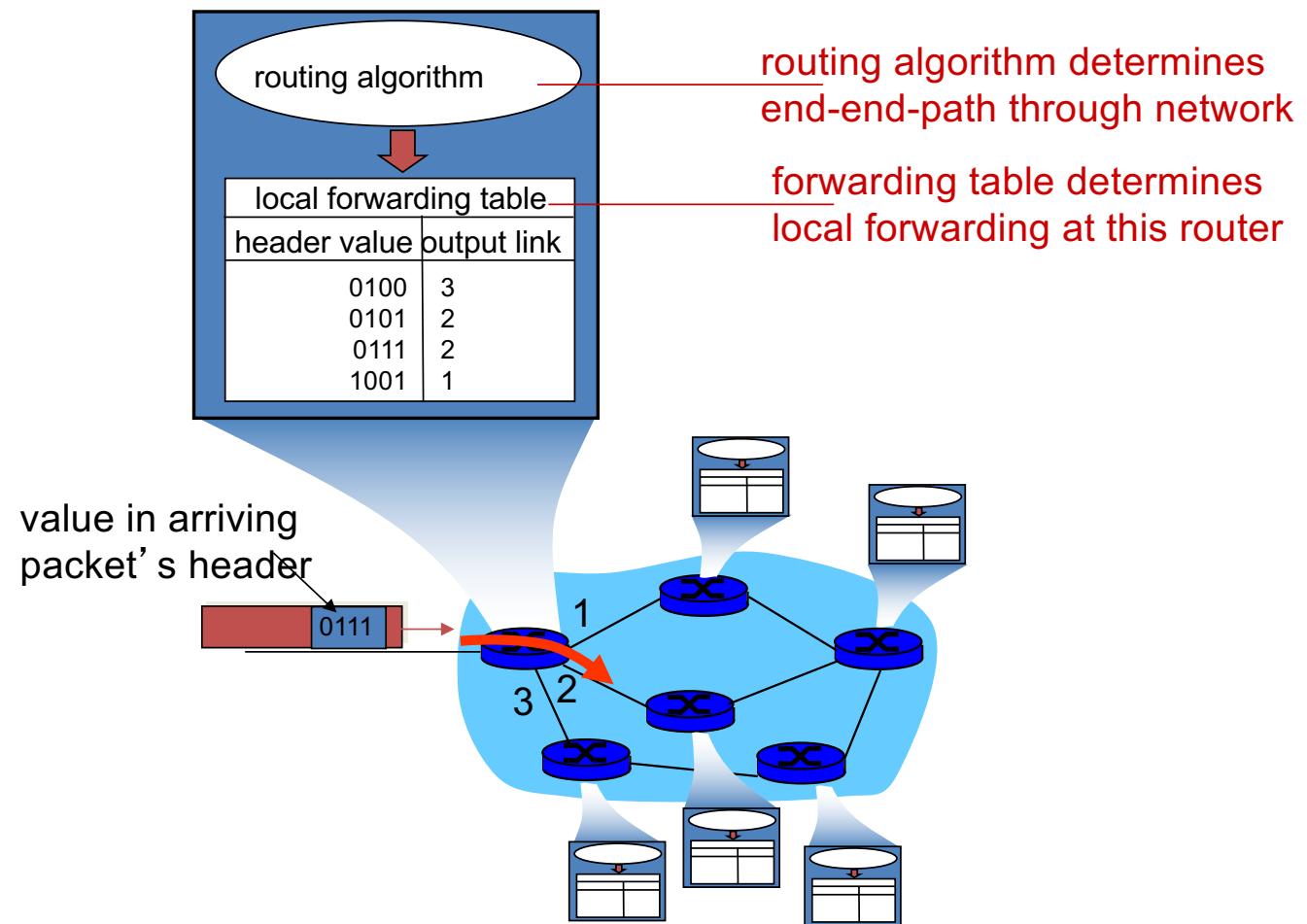


forwarding



routing

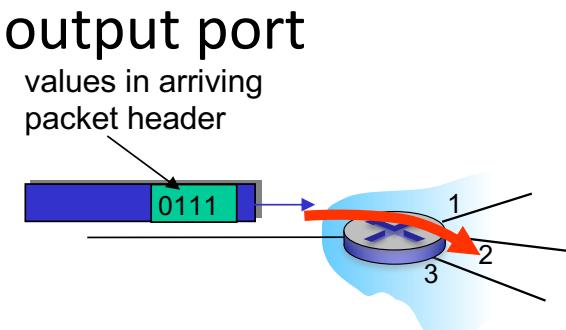
# Interplay between routing and forwarding



# Network layer: data plane, control plane

## Data plane:

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

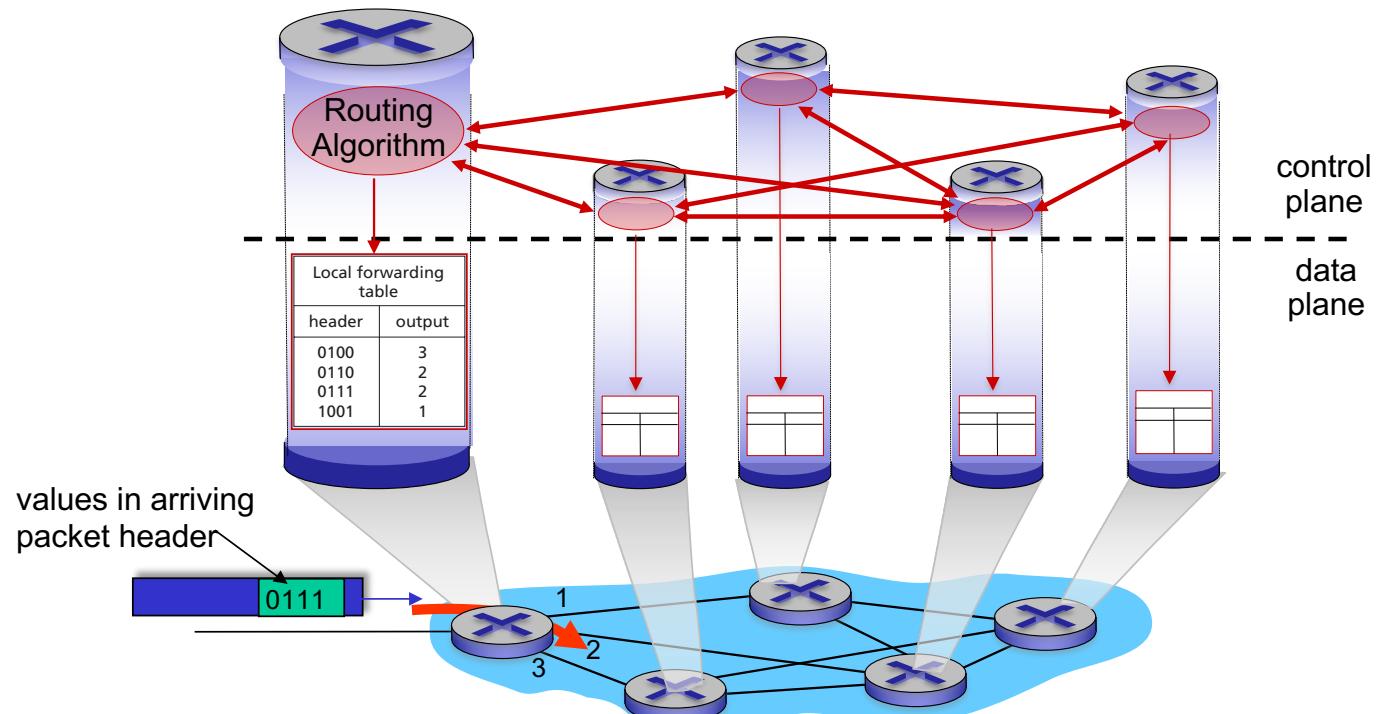


## Control plane

- *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)*: implemented in (remote) servers

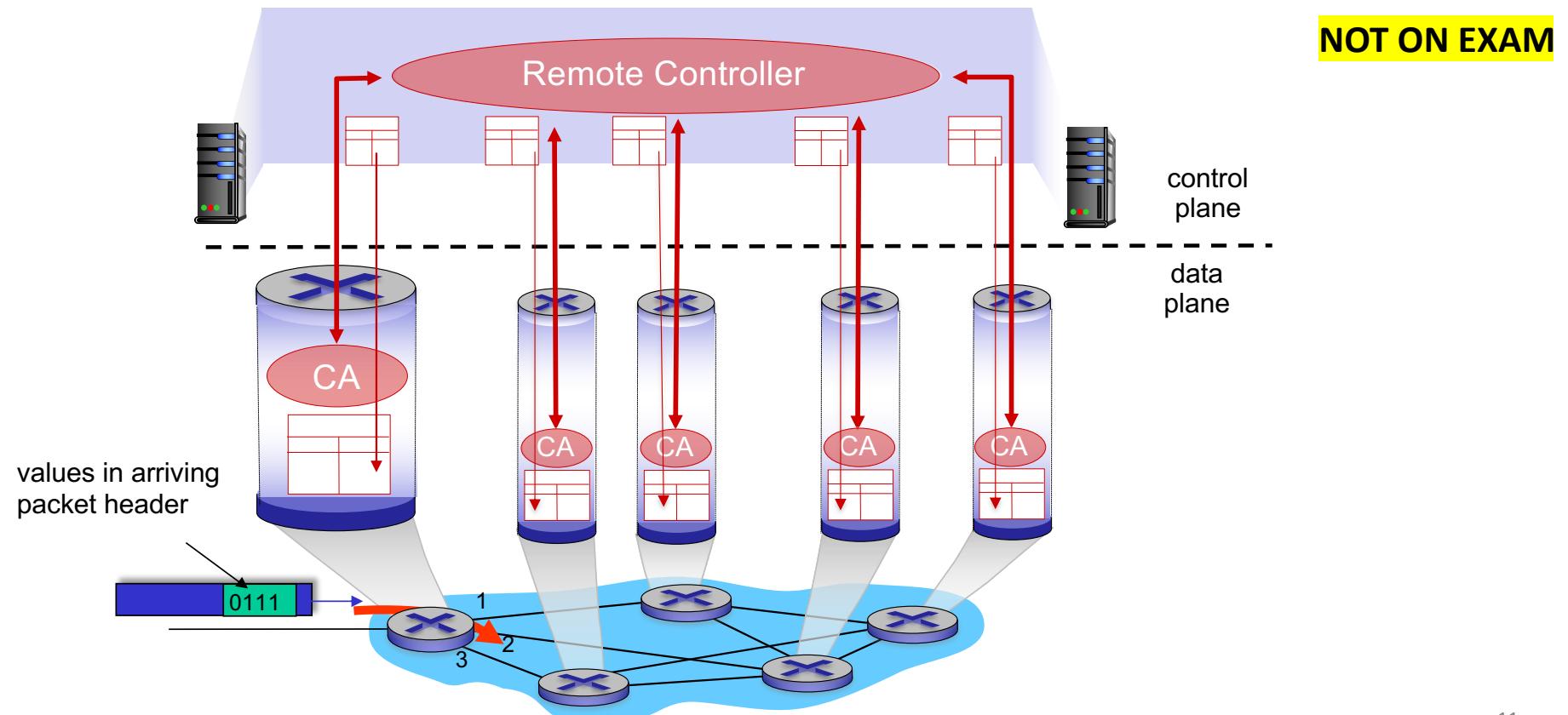
# Per-router control plane

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



# Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers



# Network Layer: service model

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

Internet “best effort” service model

**No** guarantees on:

- i. successful datagram delivery to destination
- ii. timing or order of delivery
- iii. bandwidth available to end-end flow

# Reflections on best-effort service:

- simplicity of mechanism has allowed Internet to be widely deployed adopted
- sufficient provisioning of bandwidth allows performance of real-time applications (e.g., interactive voice, video) to be “good enough” for “most of the time”
- replicated, application-layer distributed services (datacenters, content distribution networks) connecting close to clients’ networks, allow services to be provided from multiple locations
- congestion control of “elastic” services helps

*It's hard to argue with success of best-effort service model*

# 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

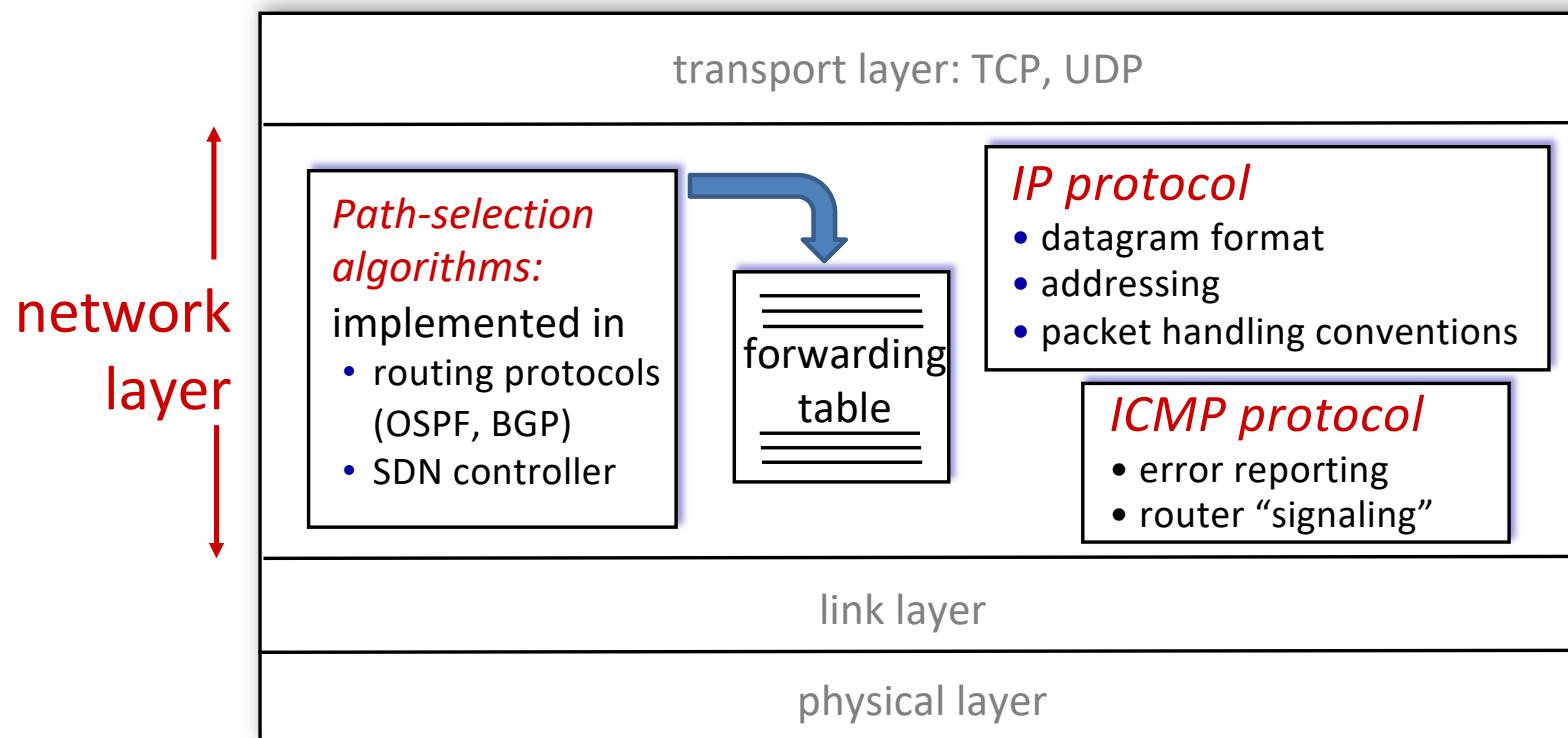
- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

## 4.4 Generalized forwarding and Software Defined Networking (SDN)

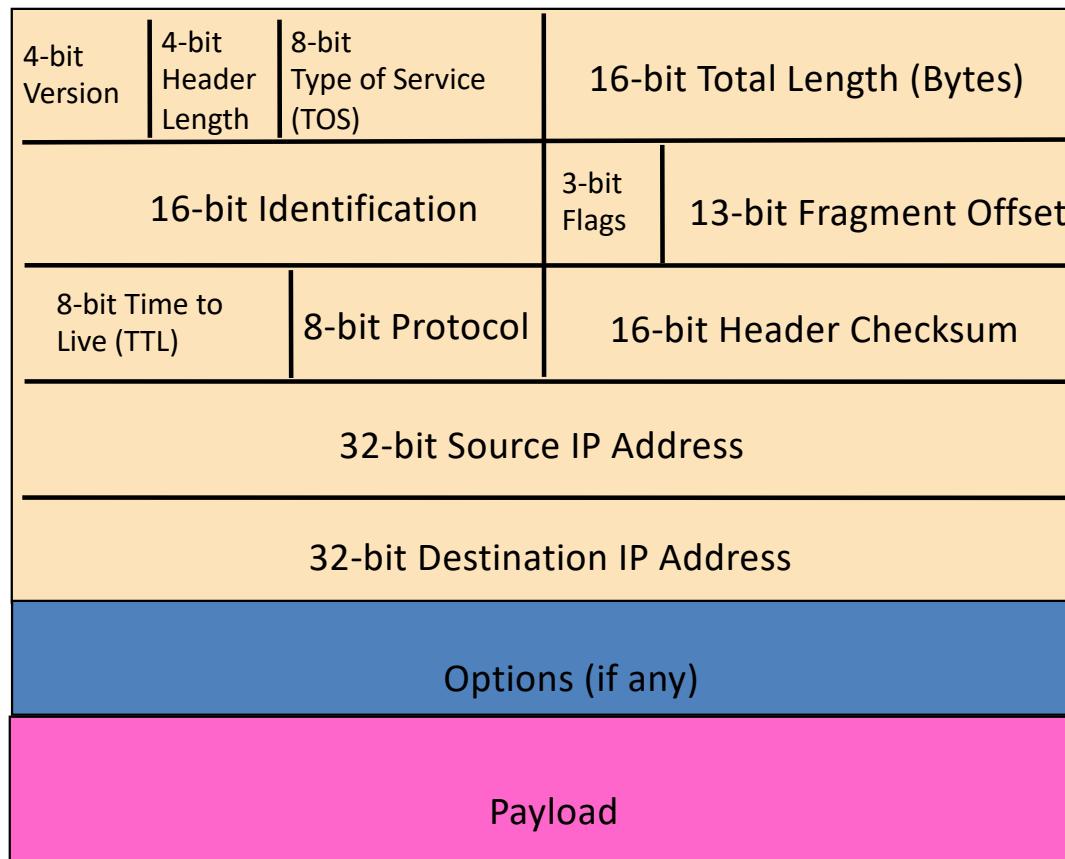
– **Not Covered**

# Network Layer: Internet

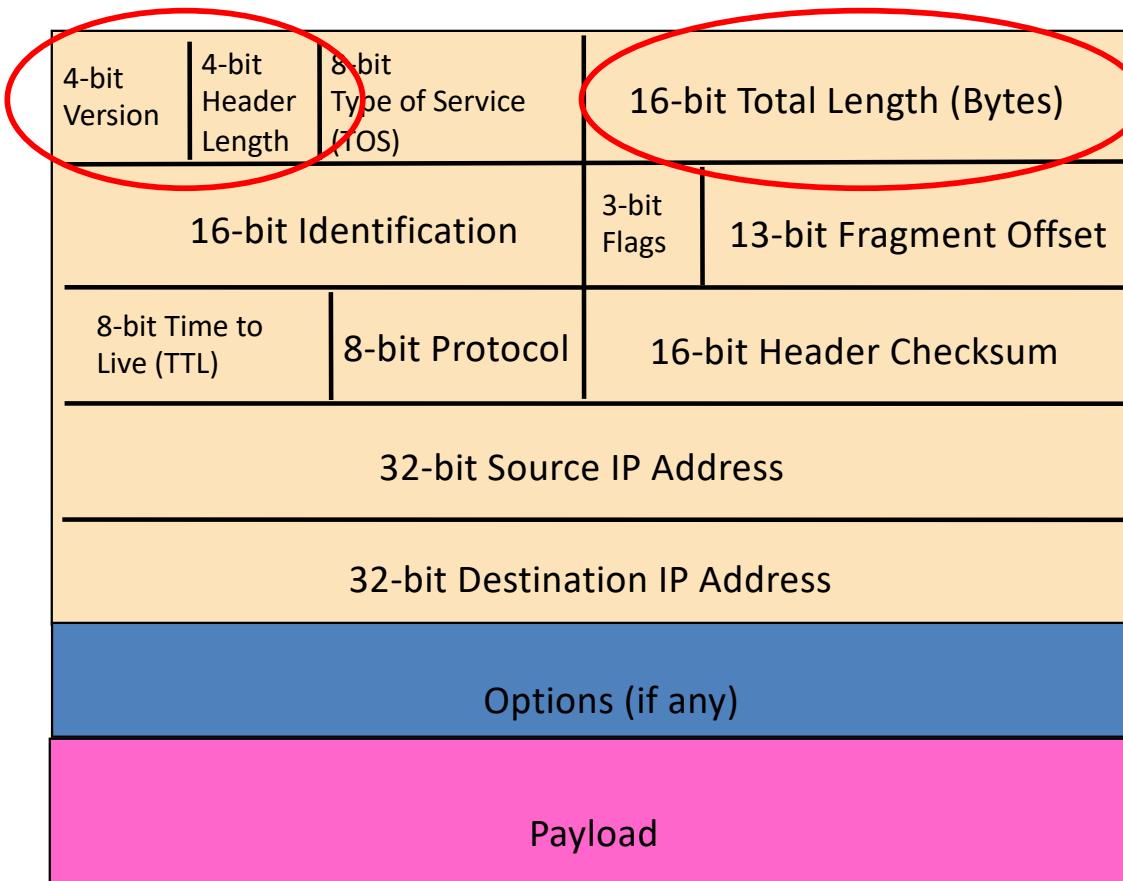
host, router network layer functions:



# IP Datagram Format



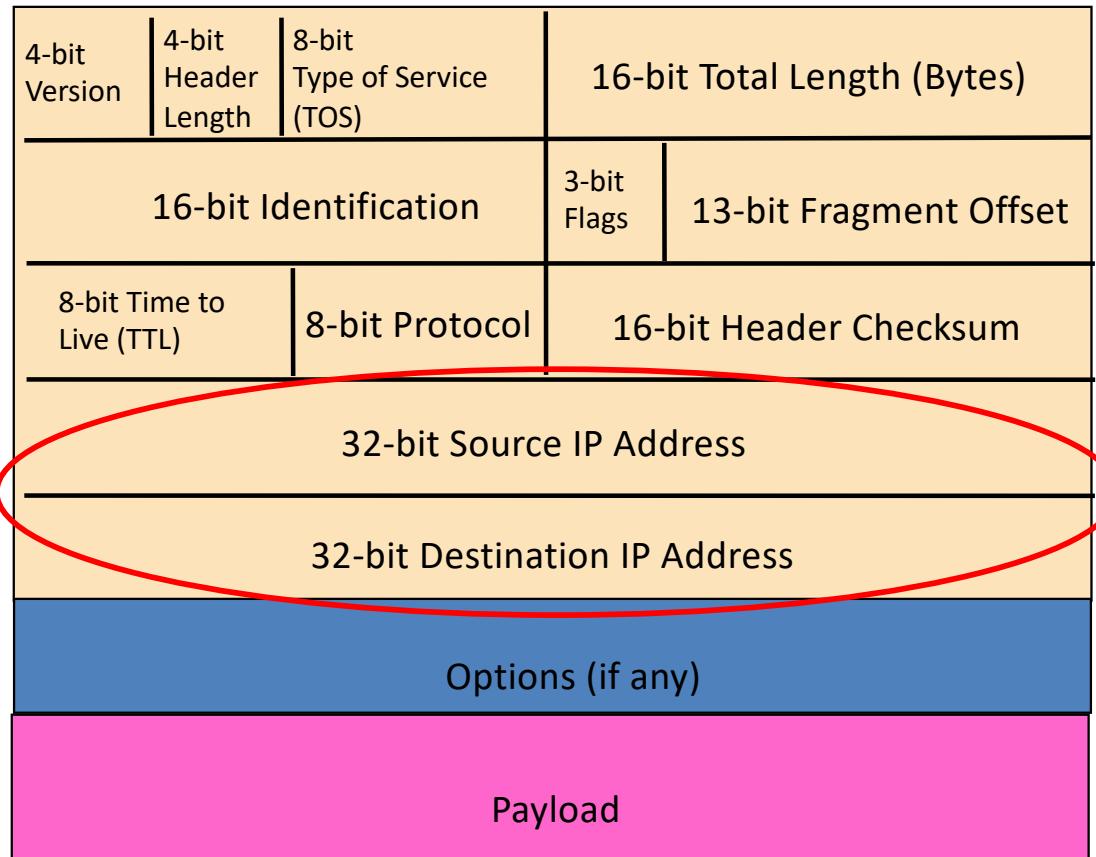
# Fields for Reading Packet Correctly



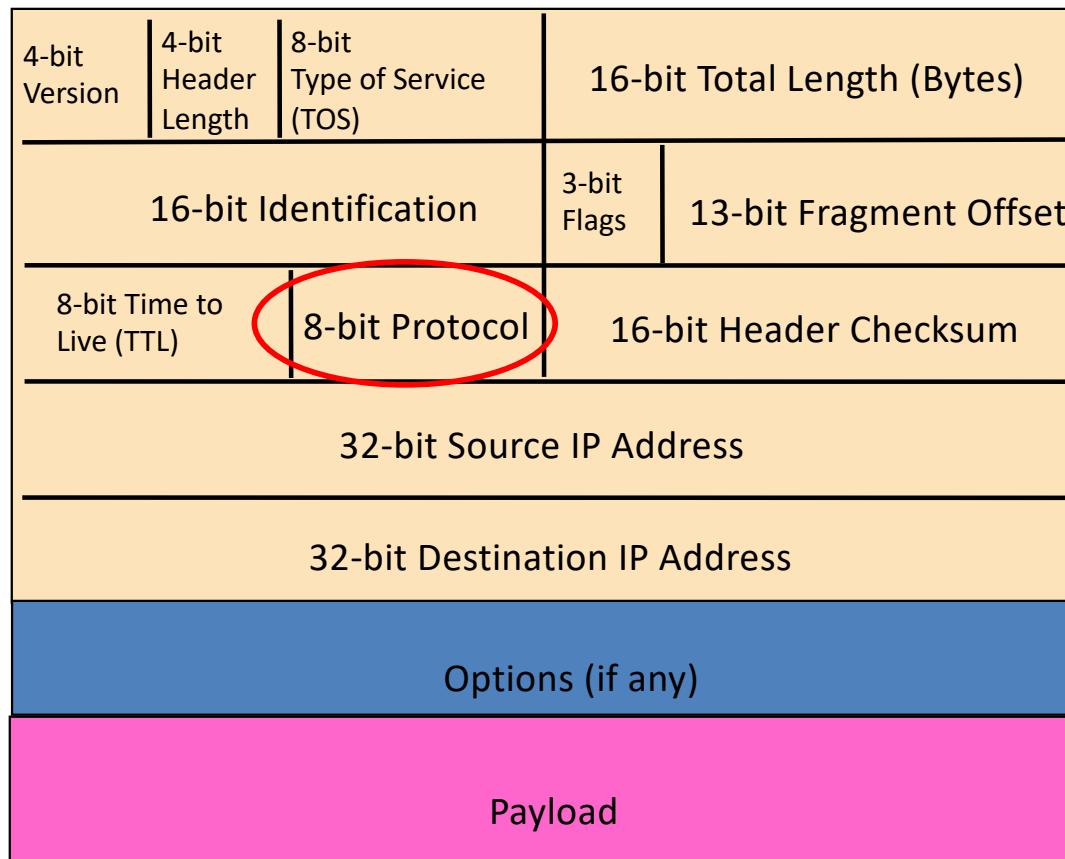
# Reading Packet Correctly

- 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)
  - 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 link layer protocols may impose smaller limits

# Fields for Reaching Destination and Back

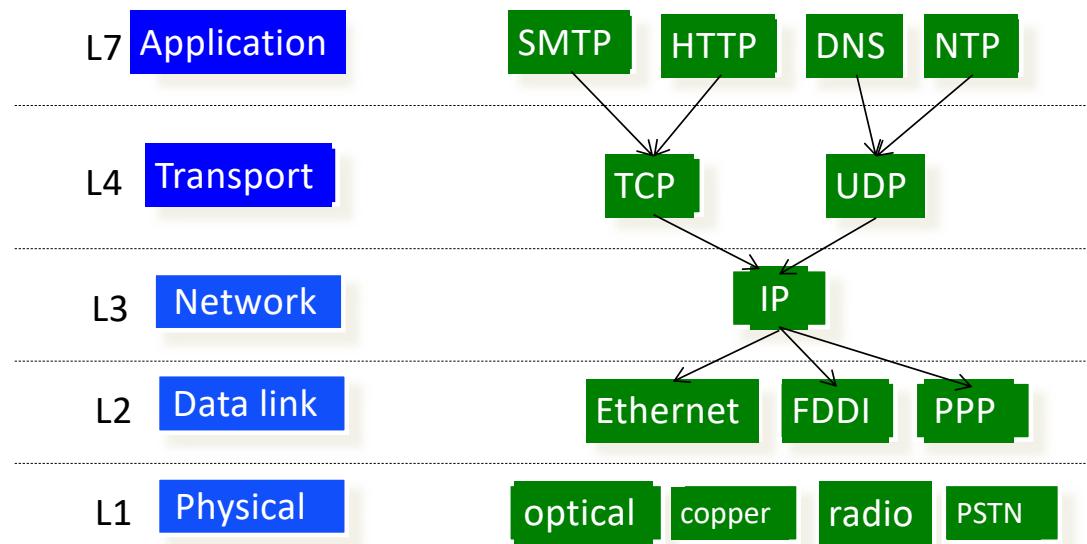


# Telling End-Host How to Handle Packet

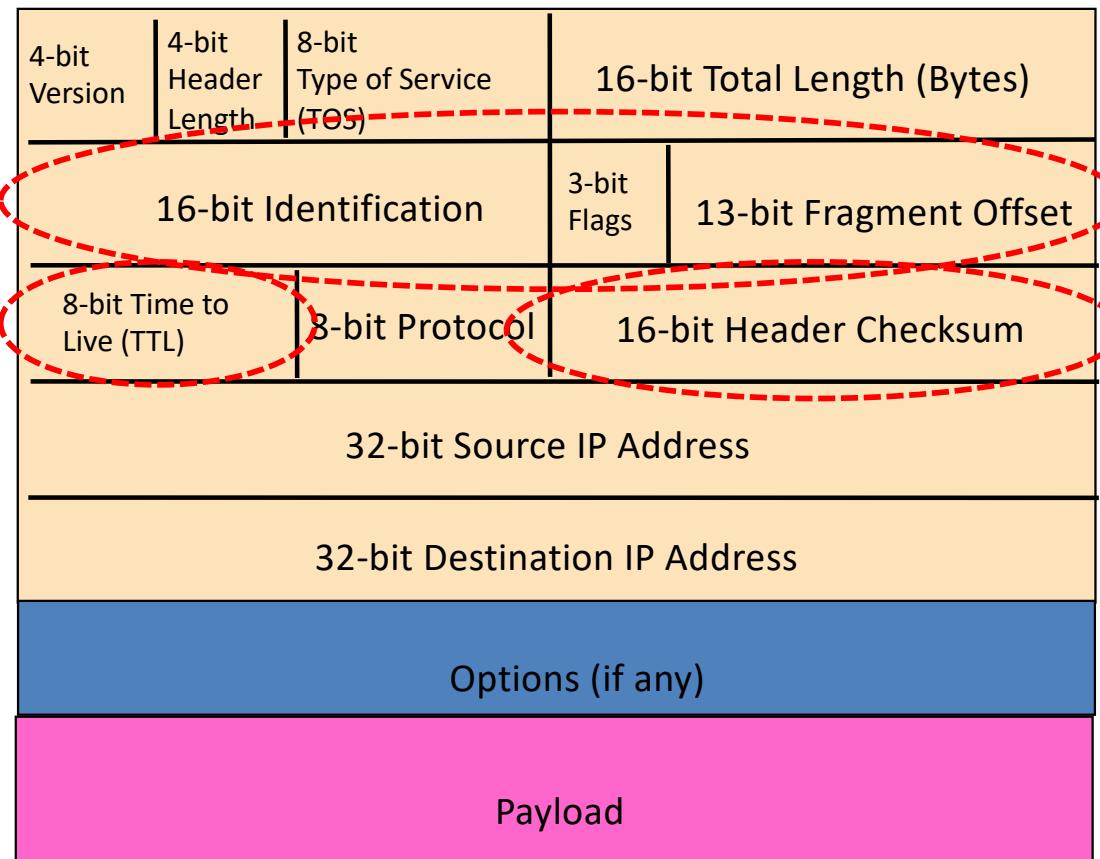


# Telling End-Host How to Handle Packet

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



# Checksum, TTL and Fragmentation Fields

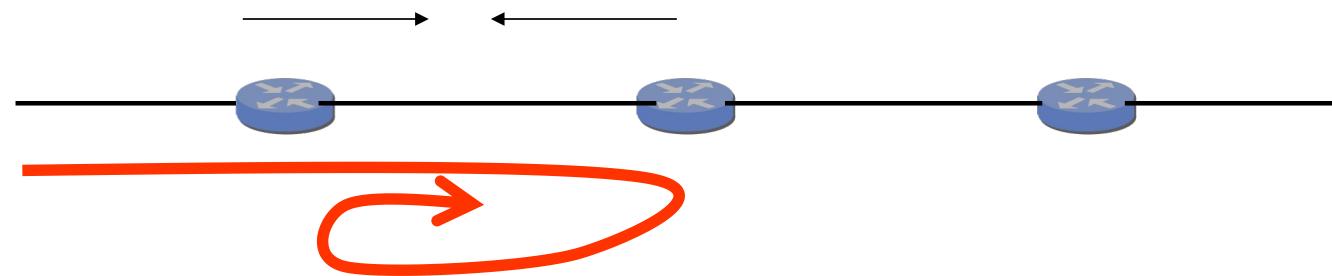


# Potential Problems

- Loop: **TTL**
- Header Corrupted: **Checksum**
- Packet too large: **Fragmentation**

# Preventing Loops (TTL)

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



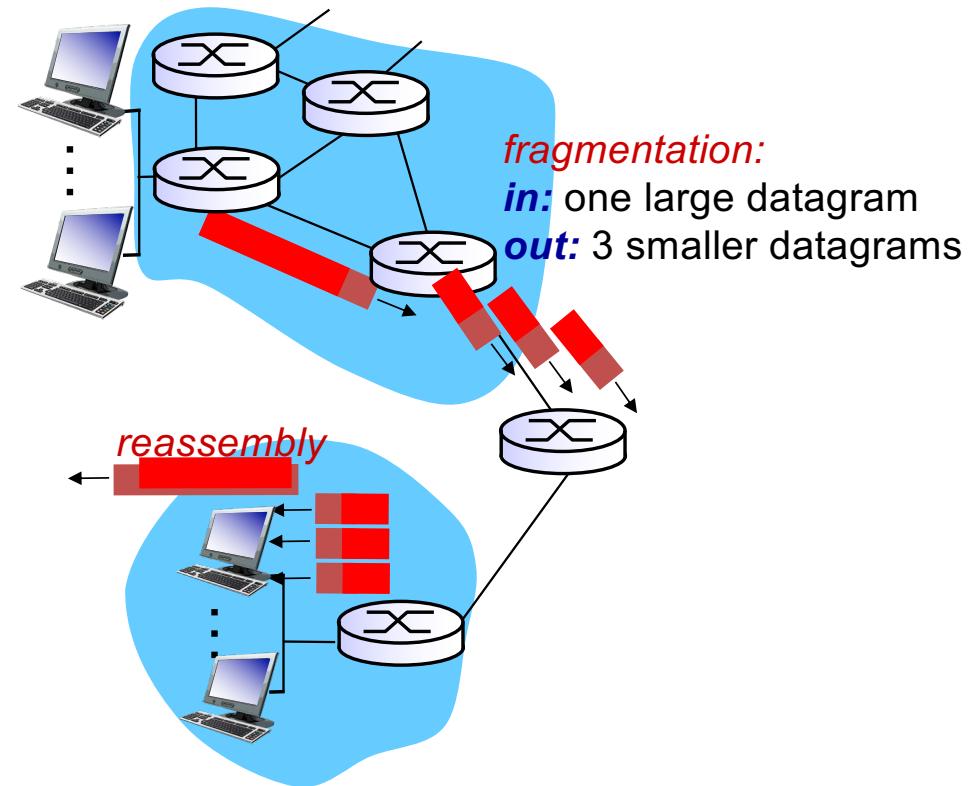
- Time-to-Live (TTL) Field (8 bits)
  - Decremented at each hop, packet discarded if reaches 0
  - ...and “time exceeded” message is sent to the source
  - Recommended default value is 64

# Header Corruption (Checksum)

- Checksum (16 bits)
  - Only computed over packet header, method is same as UDP/TCP checksum
- If not correct, router discards packets
  - So, it doesn't act on bogus information
- Checksum recalculated at every router
  - Why?
  - Why include TTL?
  - Why only header?

# 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

Note: Offset is expressed as multiple of 8 bytes

## example:

- ❖ 4000-byte datagram
- ❖ MTU = 1500 bytes

Data =  $4000 - 20$  (IP header)  
= 3980  
 $F_1 = 1480$   
 $F_2 = 1480$   
 $F_3 = 1020$

|  |                 |          |               |              |  |
|--|-----------------|----------|---------------|--------------|--|
|  | length<br>=4000 | ID<br>=x | MF flag<br>=0 | offset<br>=0 |  |
|--|-----------------|----------|---------------|--------------|--|

*one large datagram becomes several smaller datograms*

|  |                 |          |               |              |  |
|--|-----------------|----------|---------------|--------------|--|
|  | length<br>=1500 | ID<br>=x | MF flag<br>=1 | offset<br>=0 |  |
|--|-----------------|----------|---------------|--------------|--|

offset =  
 $1480/8$

|  |                 |          |               |                |  |
|--|-----------------|----------|---------------|----------------|--|
|  | length<br>=1500 | ID<br>=x | MF flag<br>=1 | offset<br>=185 |  |
|--|-----------------|----------|---------------|----------------|--|

|  |                 |          |               |                |  |
|--|-----------------|----------|---------------|----------------|--|
|  | length<br>=1040 | ID<br>=x | MF flag<br>=0 | offset<br>=370 |  |
|--|-----------------|----------|---------------|----------------|--|

# IPv4 fragmentation procedure

## ➤ Fragmentation

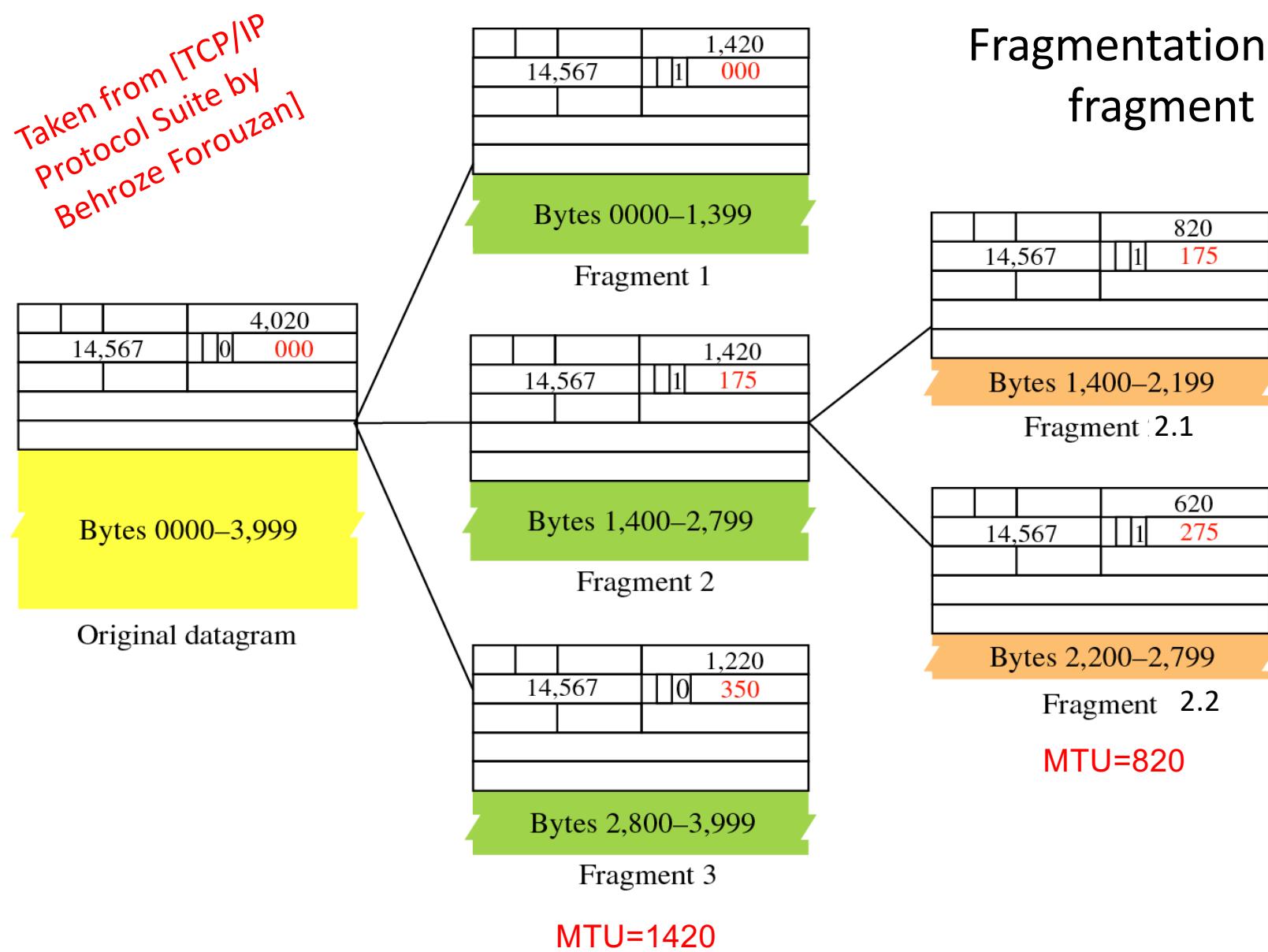
- Router breaks up datagram in size that output link can support
- Copies IP header to pieces
- Adjust length on pieces
- Set offset to indicate position
- Set MF (More fragments) flag on pieces except the last
- Re-compute checksum

## ➤ Re-assemble

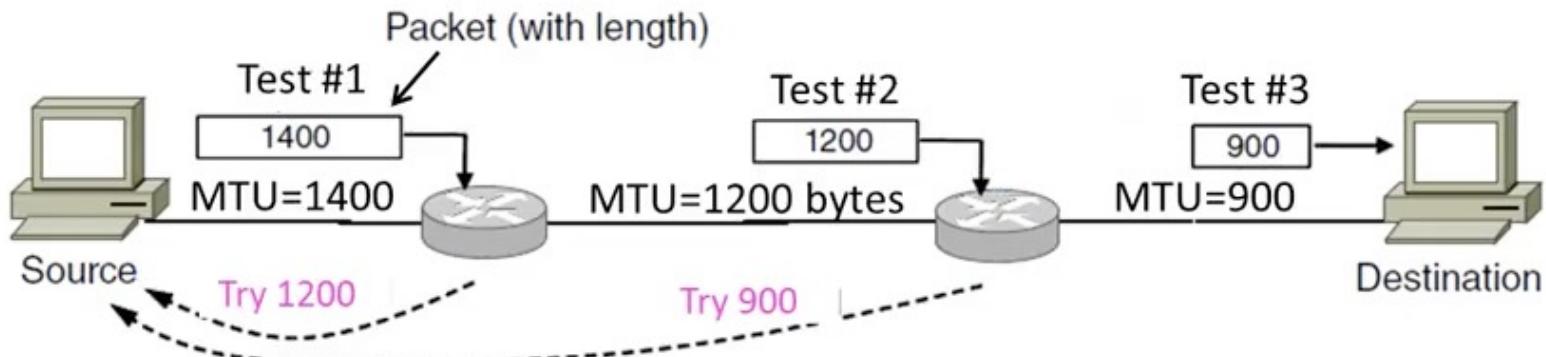
- Receiving host uses identification field with MF and offsets to complete the datagram.

## ➤ Fragmentation of fragments also supported

## Fragmentation of a fragment



# Path MTU Discovery procedure



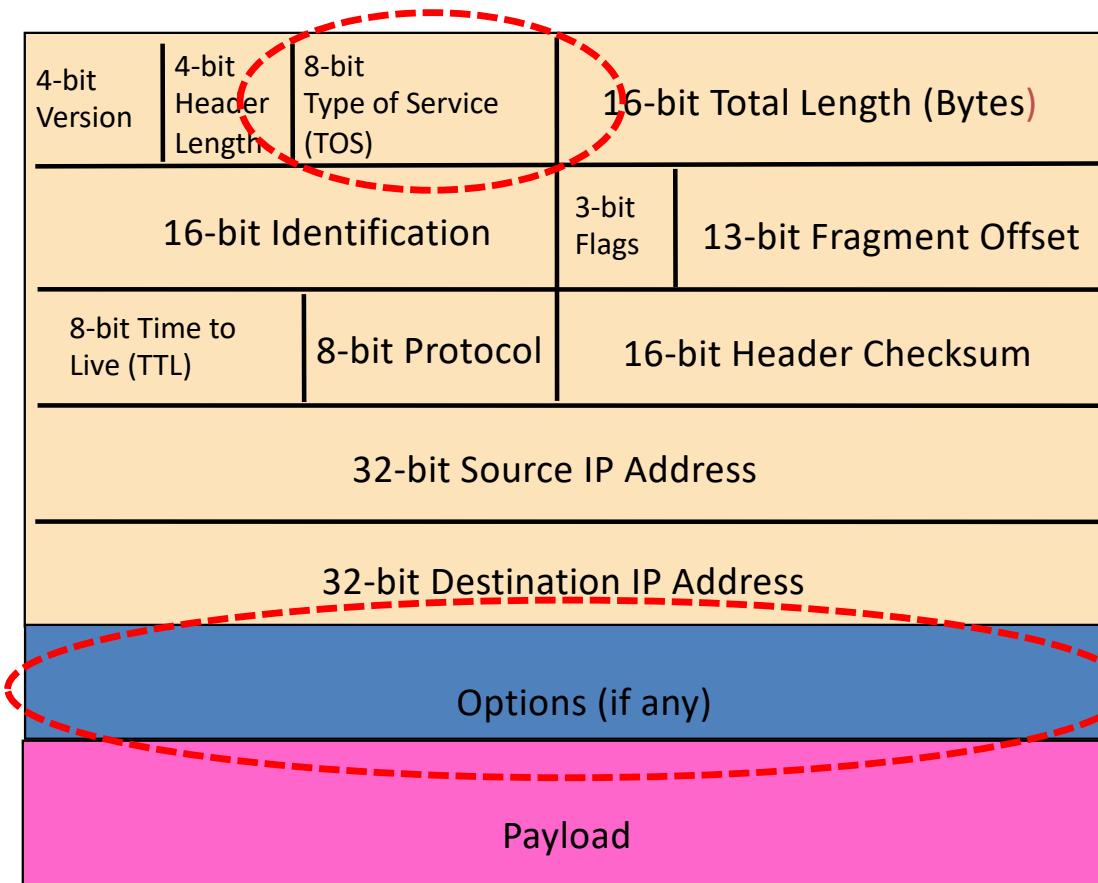
## ➤ Host

- 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

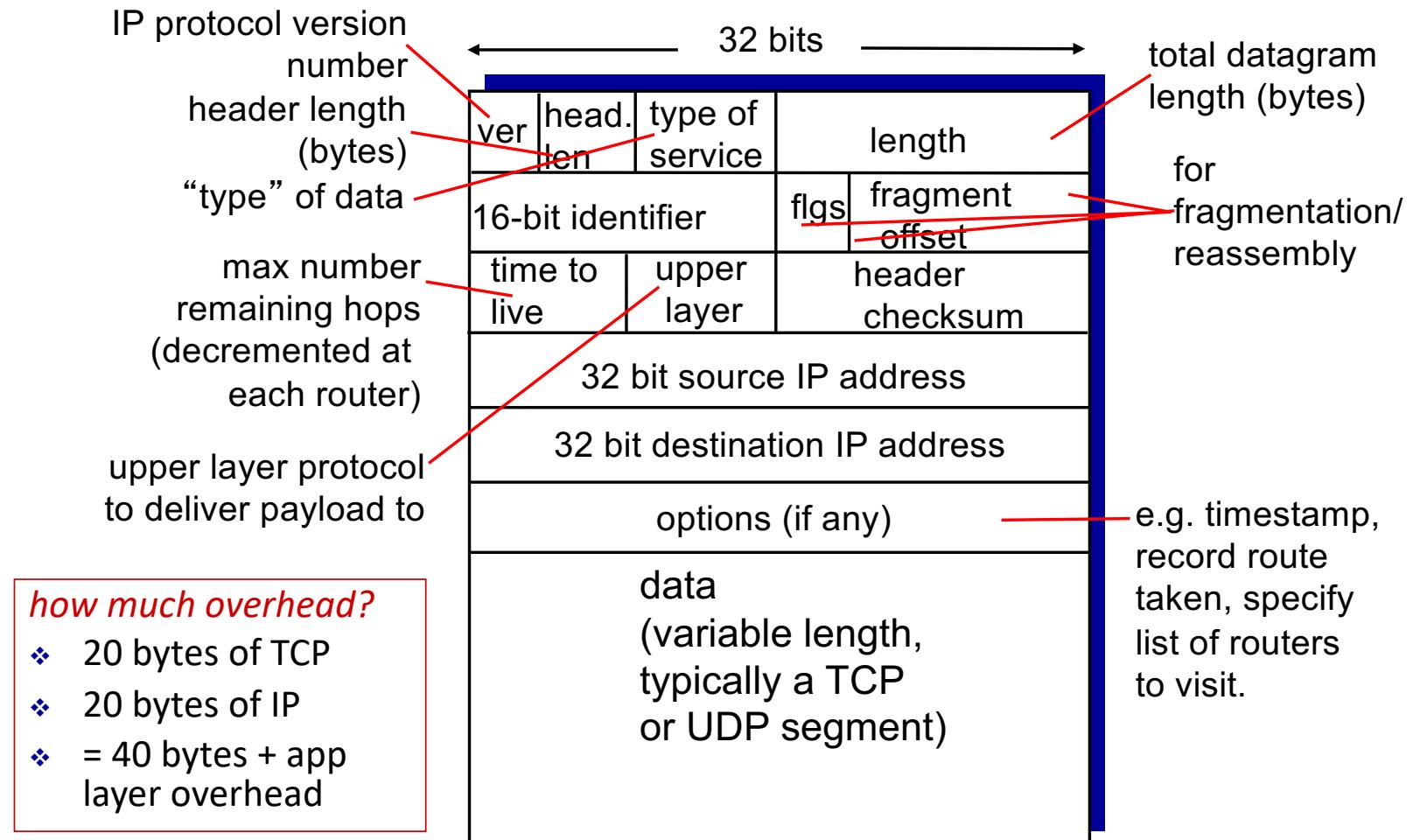
# Fields for Special Handling



# 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
  - Not widely used
- Options (not often used)

# RECAP: IP datagram format



# Network Layer, data plane: outline

## 4.1 Overview of Network layer

- data plane
- control plane

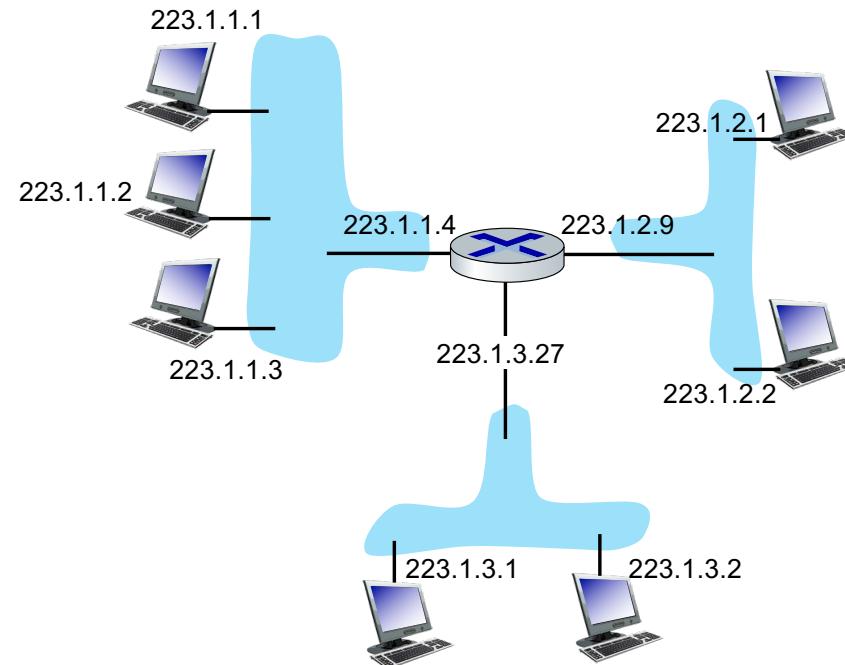
## 4.2 What's inside a router

## 4.3 IP: Internet Protocol

- datagram format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

# IP addressing: introduction

- **IP address:** 32-bit identifier associated with each host or 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)

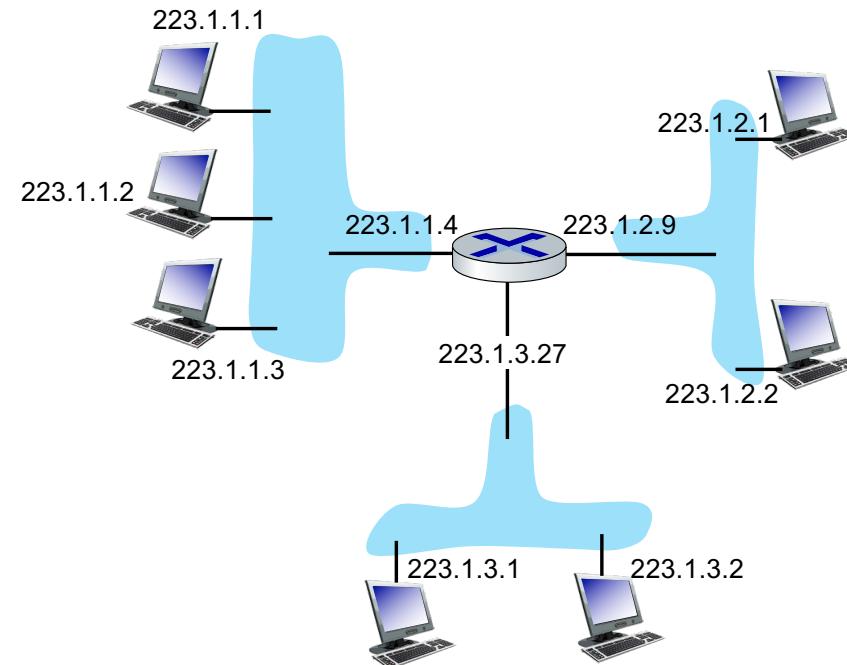


dotted-decimal IP address notation:

223.1.1.1 = 11011111 00000001 00000001 00000001  
                  |                |                |  
          223    1      1      1

# IP addressing: introduction

- IP address: 32-bit identifier associated with each host or 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)



dotted-decimal IP address notation:

$223.1.1.1 = 11011111\ 00000001\ 00000001\ 00000001$

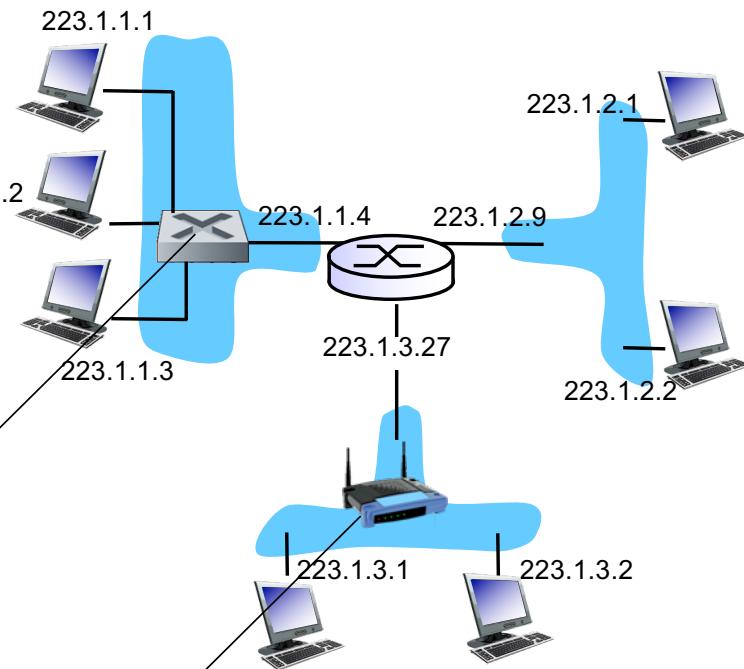
|     |   |   |   |   |
|-----|---|---|---|---|
| 223 | 1 | 1 | 1 | 1 |
|-----|---|---|---|---|

# 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*

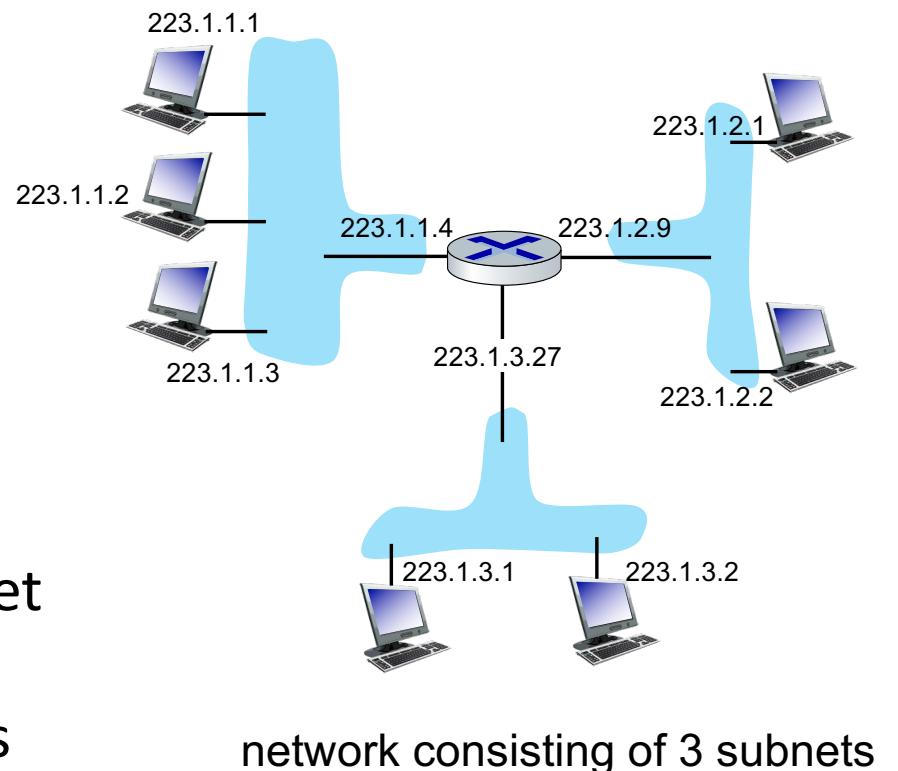
# Subnets

## ■ What's a subnet ?

- device interfaces that can physically reach each other **without passing through an intervening router**

## ■ IP addresses have structure:

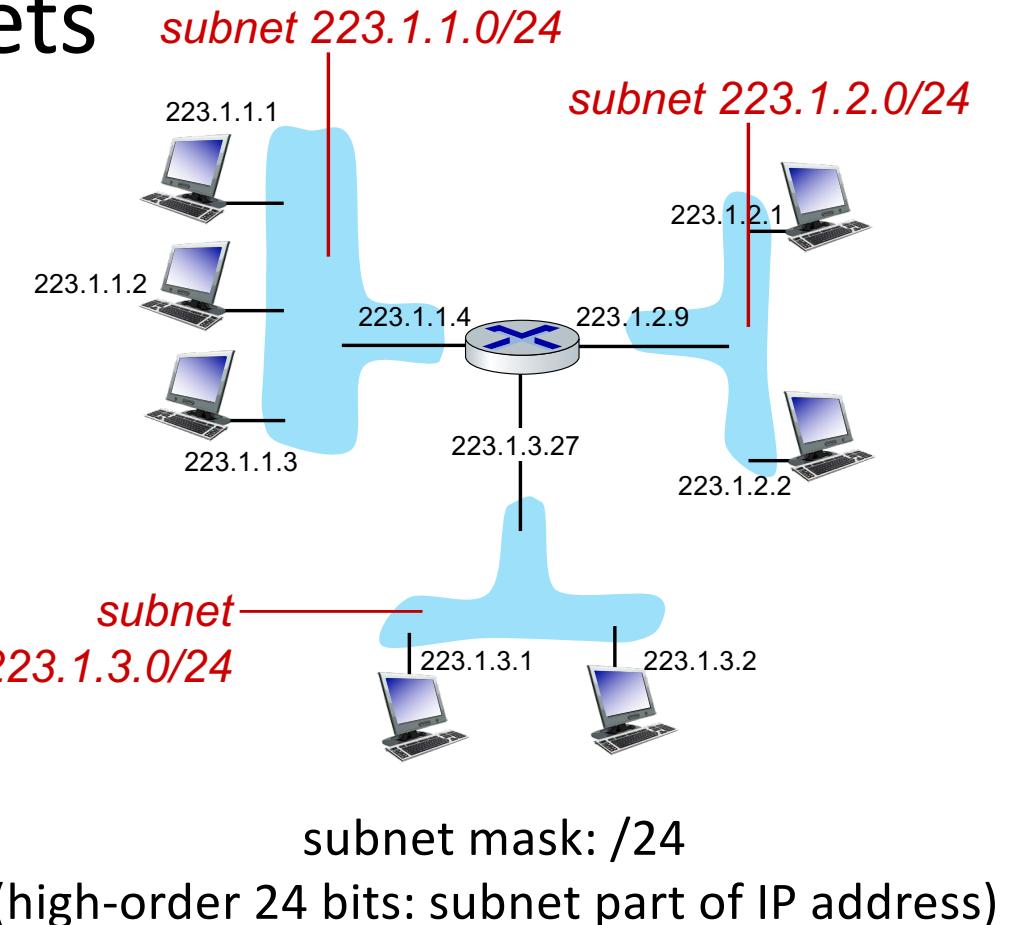
- **subnet part:** devices in same subnet have common high order bits
- **host part:** remaining low order bits



# Subnets

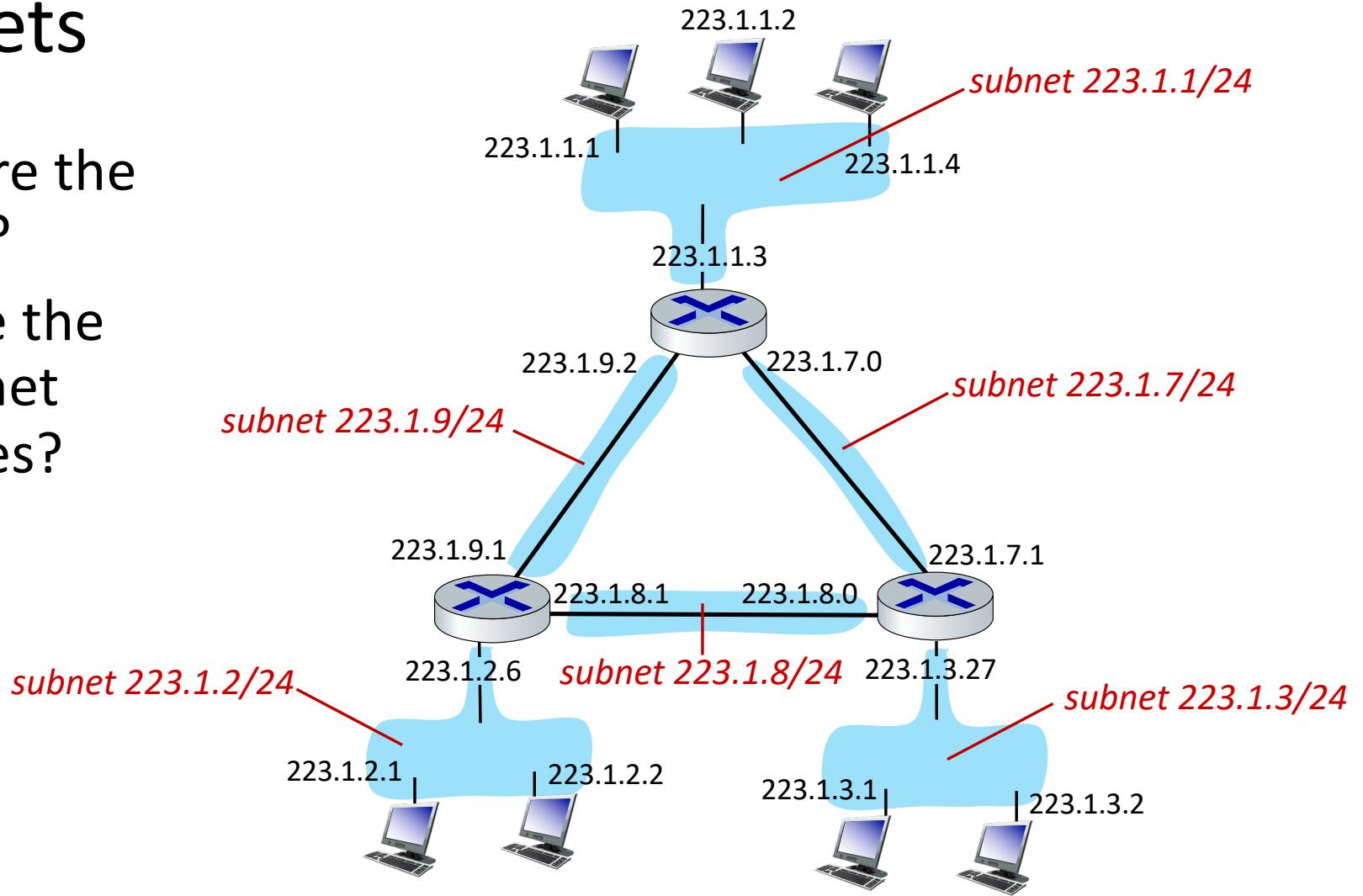
*Recipe for defining subnets:*

- detach each interface from its host or router, creating “islands” of isolated networks
- each isolated network is called a *subnet*



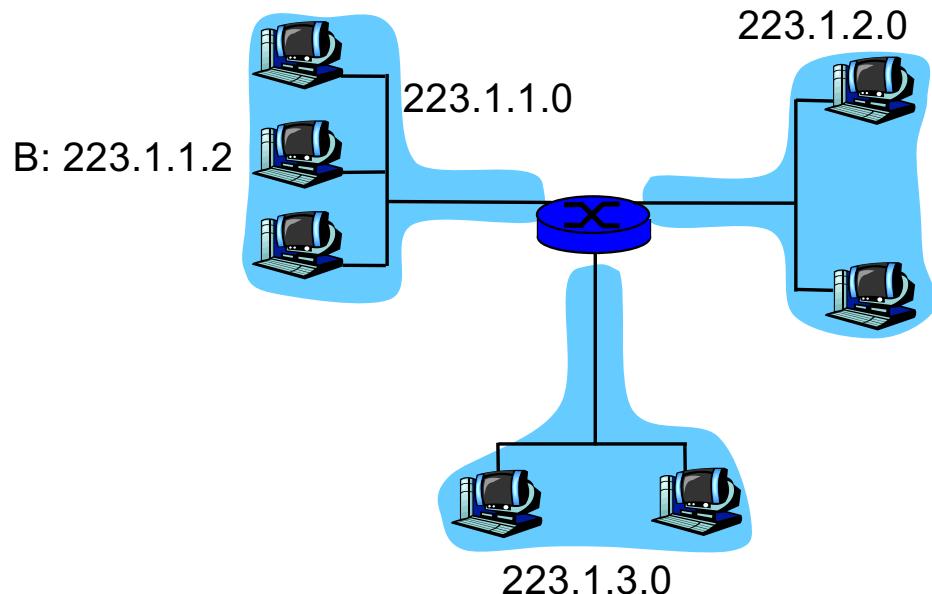
# Subnets

- where are the subnets?
- what are the /24 subnet addresses?



# Network Mask

- 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
- 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 are typically not assigned to any host



| Host B       | Dot-decimal address | Binary                              |
|--------------|---------------------|-------------------------------------|
| IP address   | 223.1.1.2           | 11011111.00000001.00000001.00000010 |
| Mask         | 255.255.255.0       | 11111111.11111111.11111111.00000000 |
| Network Part | 223.1.1.0           | 11011111.00000001.00000001.00000000 |
| 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

|         | 0    | 8            | 16                             | 24            | 31 |                                  |                                 |
|---------|------|--------------|--------------------------------|---------------|----|----------------------------------|---------------------------------|
| Class A | 0    | <i>netid</i> |                                | <i>hostid</i> |    | $2^7$ nets,<br>$2^{24}$ hosts    | 1.0.0.0 to<br>127.255.255.255   |
| Class B | 10   | <i>netid</i> |                                | <i>hostid</i> |    | $2^{14}$ nets,<br>$2^{16}$ hosts | 128.0.0.0 to<br>191.255.255.255 |
| Class C | 110  | <i>netid</i> |                                | <i>hostid</i> |    | $2^{21}$ nets,<br>$2^8$ hosts    | 192.0.0.0 to<br>223.255.255.255 |
| Class D | 1110 |              | <i>multicast address</i>       |               |    |                                  | 224.0.0.0 to<br>239.255.255.255 |
| Class E | 1111 |              | <i>reserved for future use</i> |               |    |                                  | 240.0.0.0 to<br>255.255.255.255 |

Problem: Networks only come in three sizes!

# What are the issues?

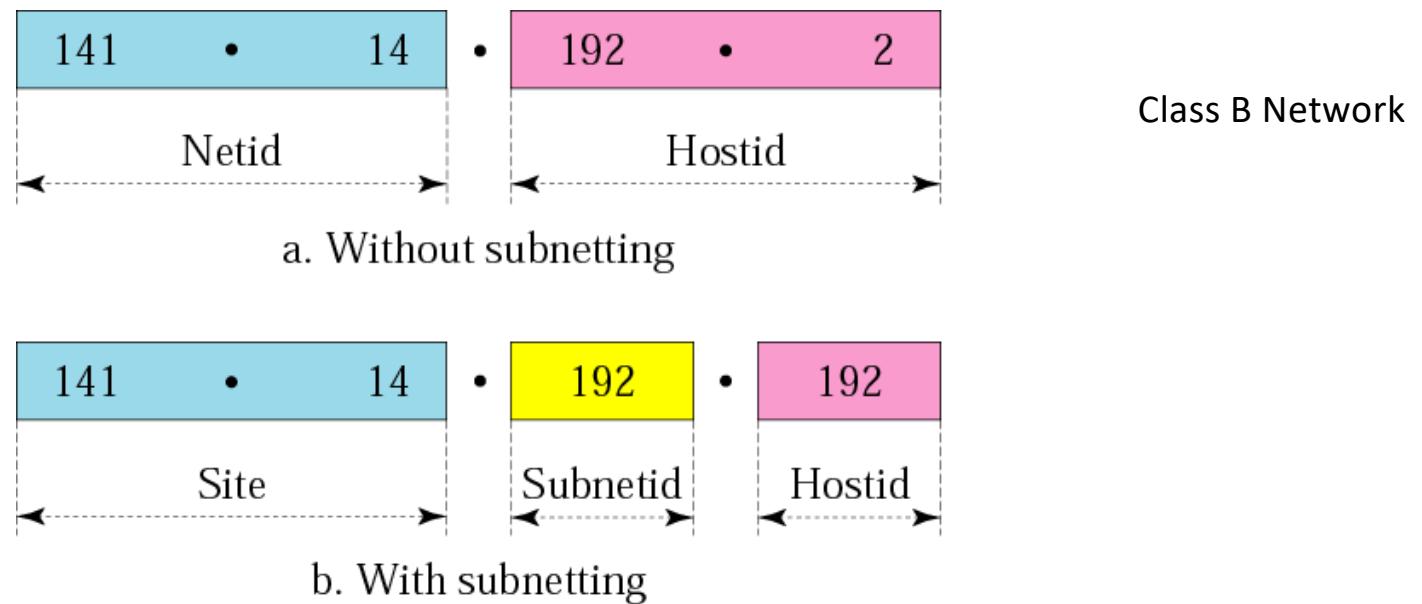
- An organization requires 6 nets each of size 30.  
Does it have to buy 6 class C address blocks?
  
- An organization requires 512 addresses? How  
many IP addresses should it buy?

# 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

# Subnetting

- 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



# Quiz?

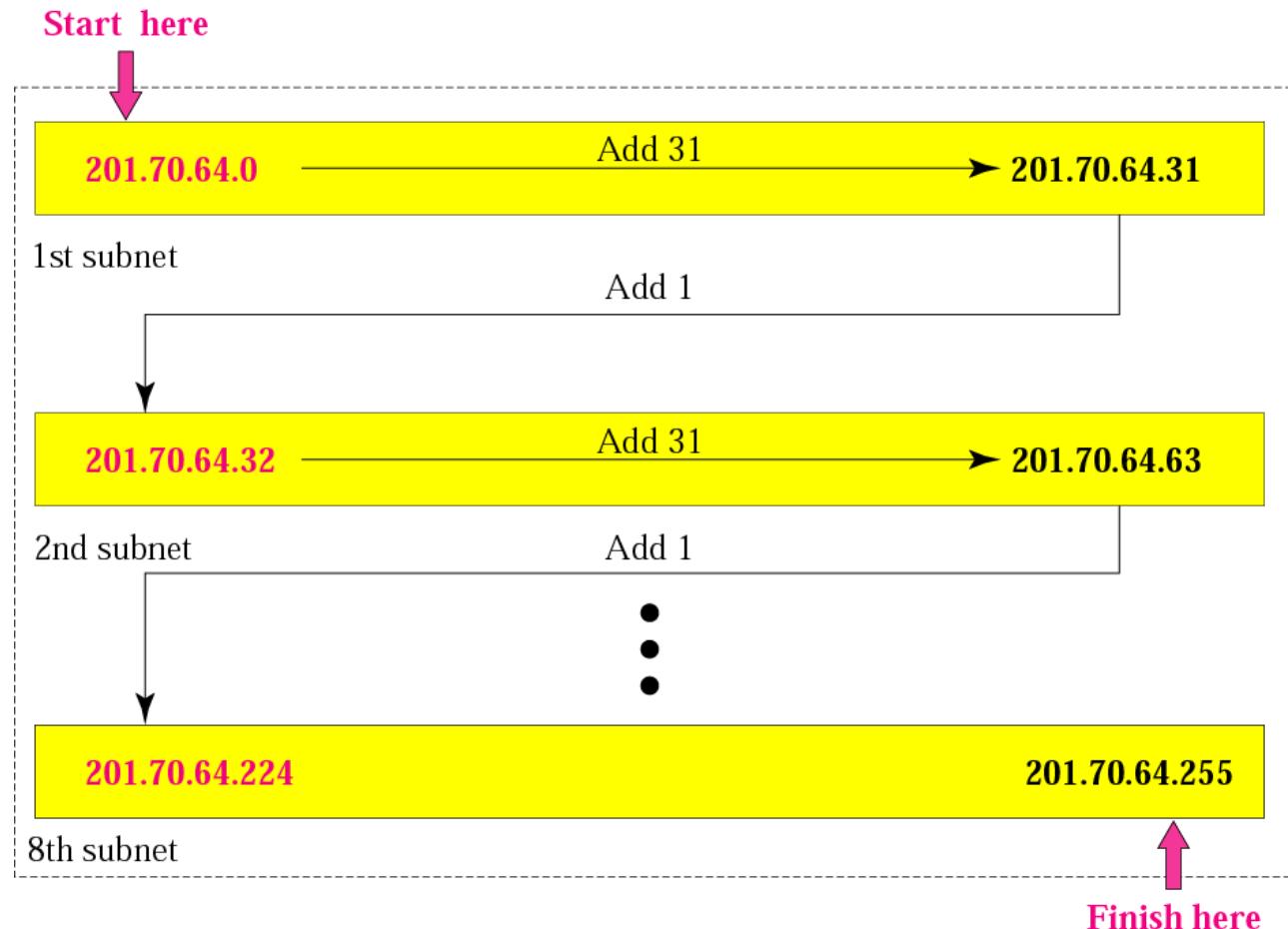
A company is granted the site address 201.70.64.0 (class C). The company needs six subnets. Design the subnets.

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

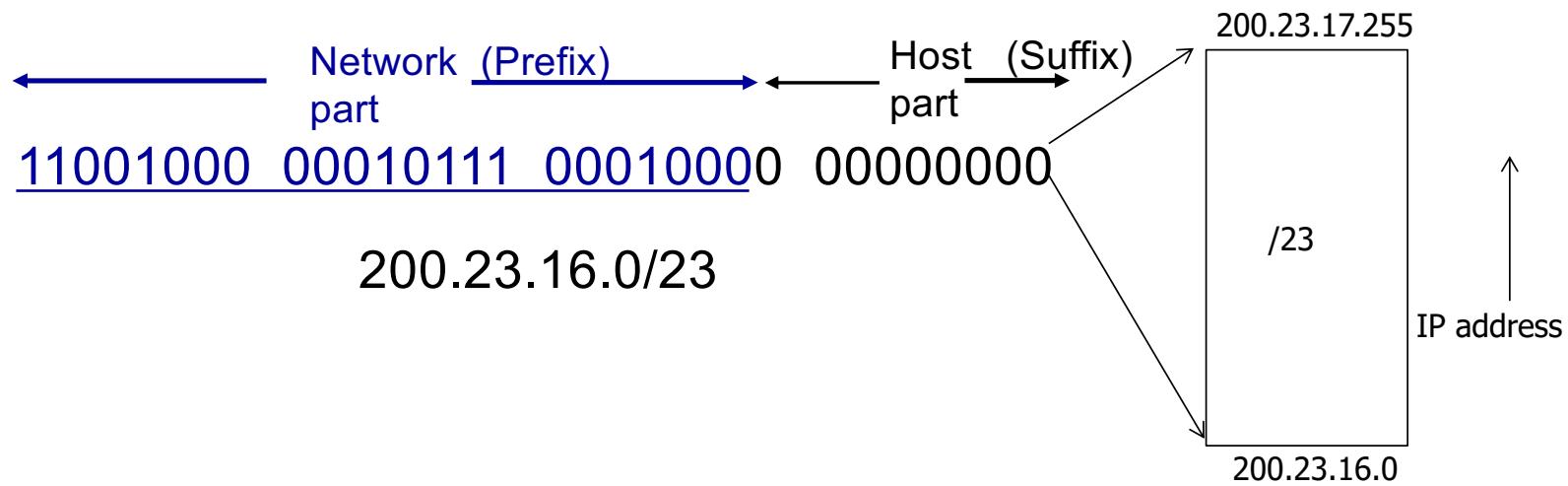
The number of addresses in each subnet is  $2^5$  or 32.



# 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

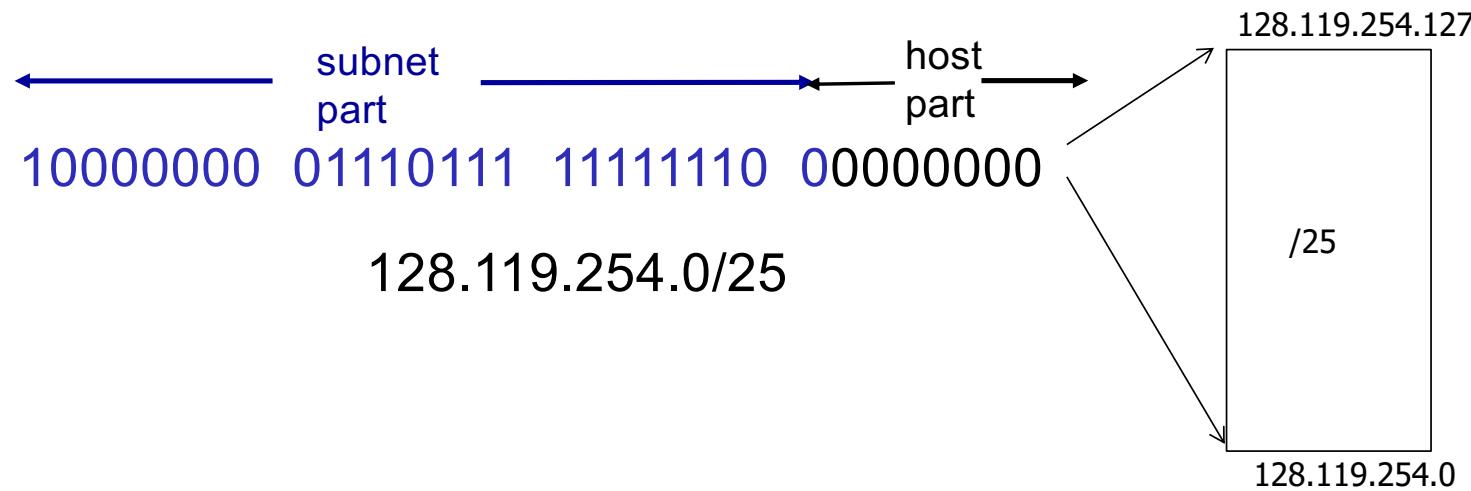


# Quiz: IP Addressing



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

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



# Quiz: IP Addressing



How many IP addresses belong to the subnet  
 $134.45.22.0/23$ ?

- A) 32
- B) 64
- C) 128
- D) 256
- E) 512



# Quiz: IP Addressing

An ISP is granted a block of addresses starting with 190.100.0.0/16 (Class B). 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

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

01: 190.100.0.0/24 → 190.100.0.255/24

02: 190.100.1.0/24 → 190.100.1.255/24

.....

64: 190.100.63.0/24 → 190.100.63.255/24

Total =  $64 \times 256 = 16,384$

## Group 2

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:

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?

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)

How does *host* get IP address?

- hard-coded by sysadmin in config file (e.g., `/etc/rc.config` in UNIX)
- **DHCP: Dynamic Host Configuration Protocol:** dynamically get address from server
  - “plug-and-play”

# DHCP: Dynamic Host Configuration Protocol

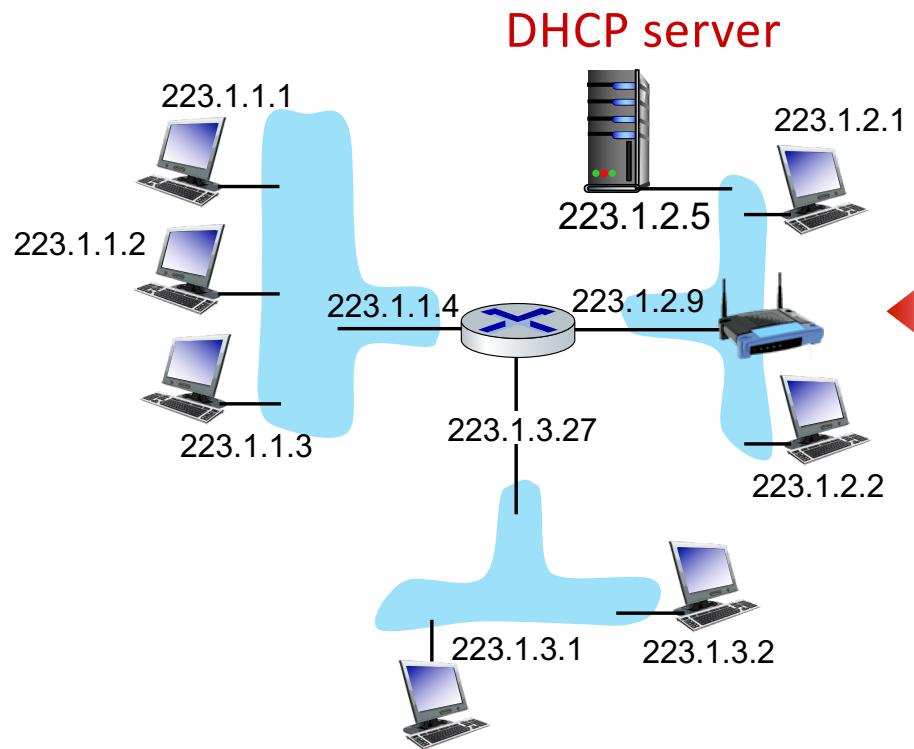
**goal:** host *dynamically* obtains IP address from network server when it “joins” network

- 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

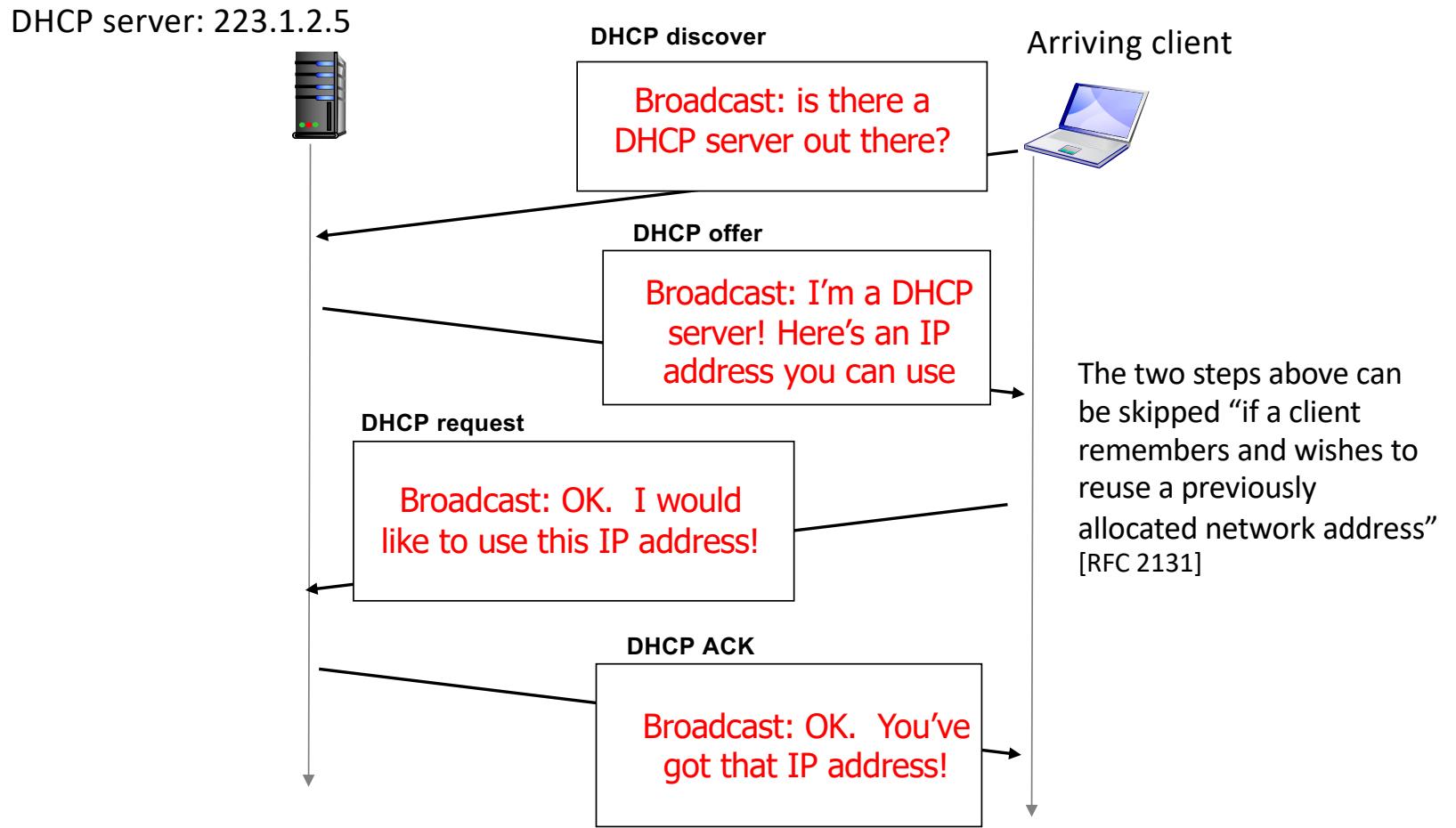


Typically, DHCP server will be co-located in router, serving all subnets to which router is attached



arriving **DHCP client** needs address in this network

# DHCP client-server scenario

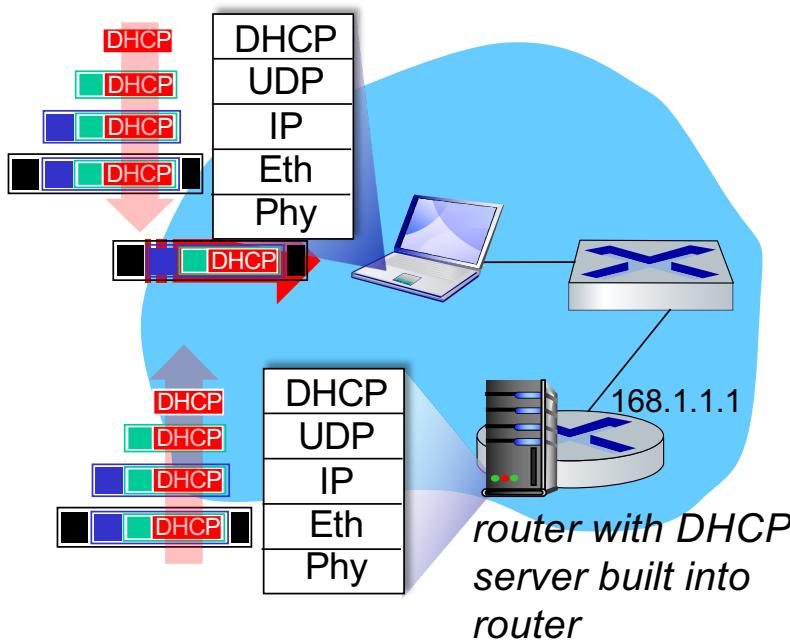


# DHCP: more than IP addresses

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

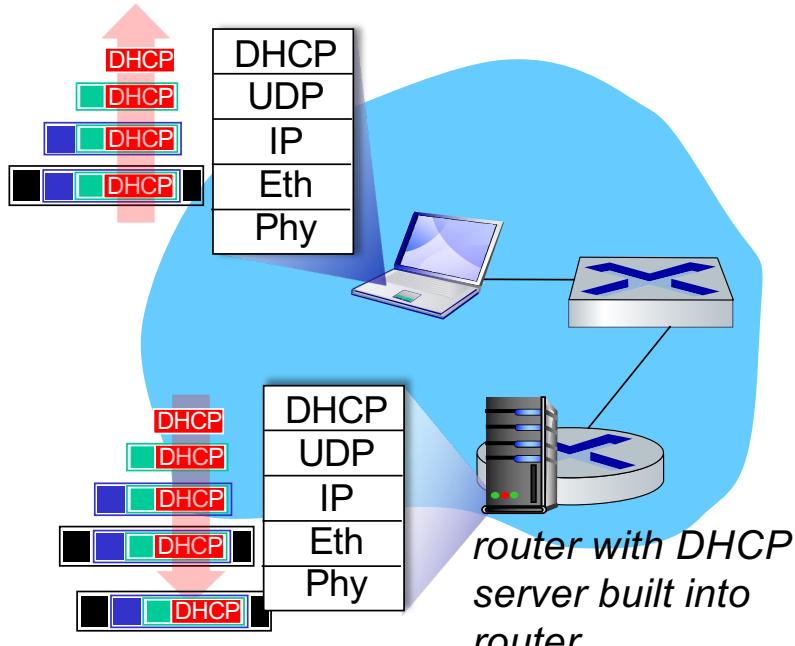
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

# DHCP: example



- Connecting laptop 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



- DCP 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

Note: Only last 2 steps of the 4-way message exchange are shown

# IP addresses: how to get one?

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

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

ISP's block      11001000 00010111 00010000 00000000    200.23.16.0/20

ISP can then allocate out its address space in 8 blocks:

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

...

.....

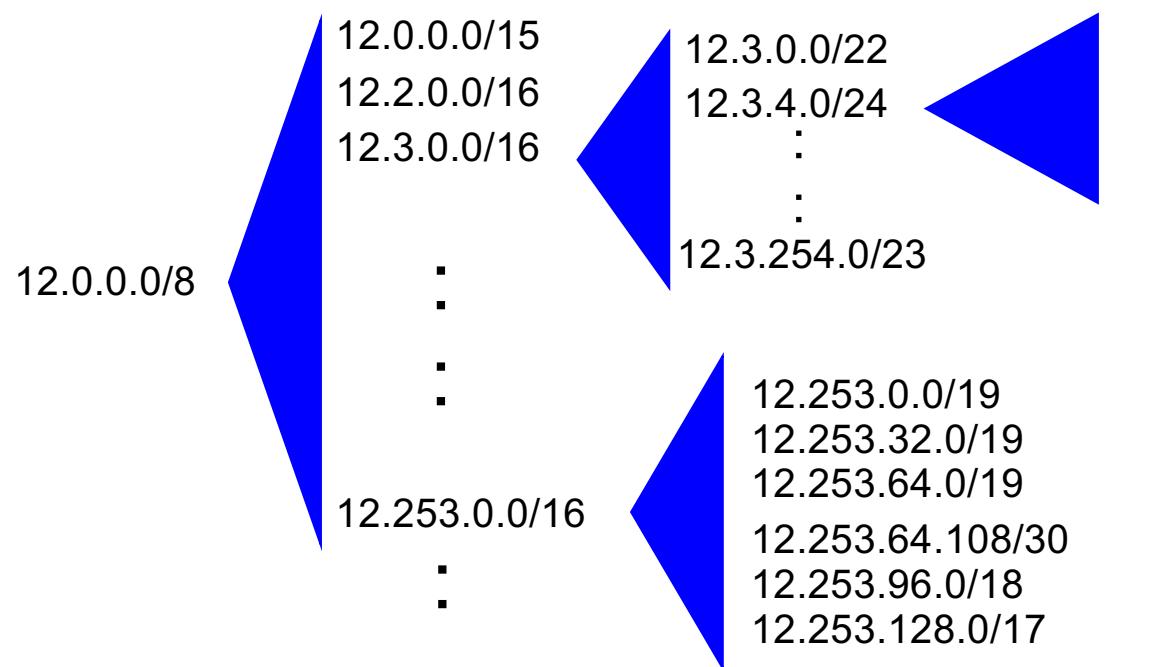
....

....

Organization 7    11001000 00010111 00011110 00000000    200.23.30.0/23

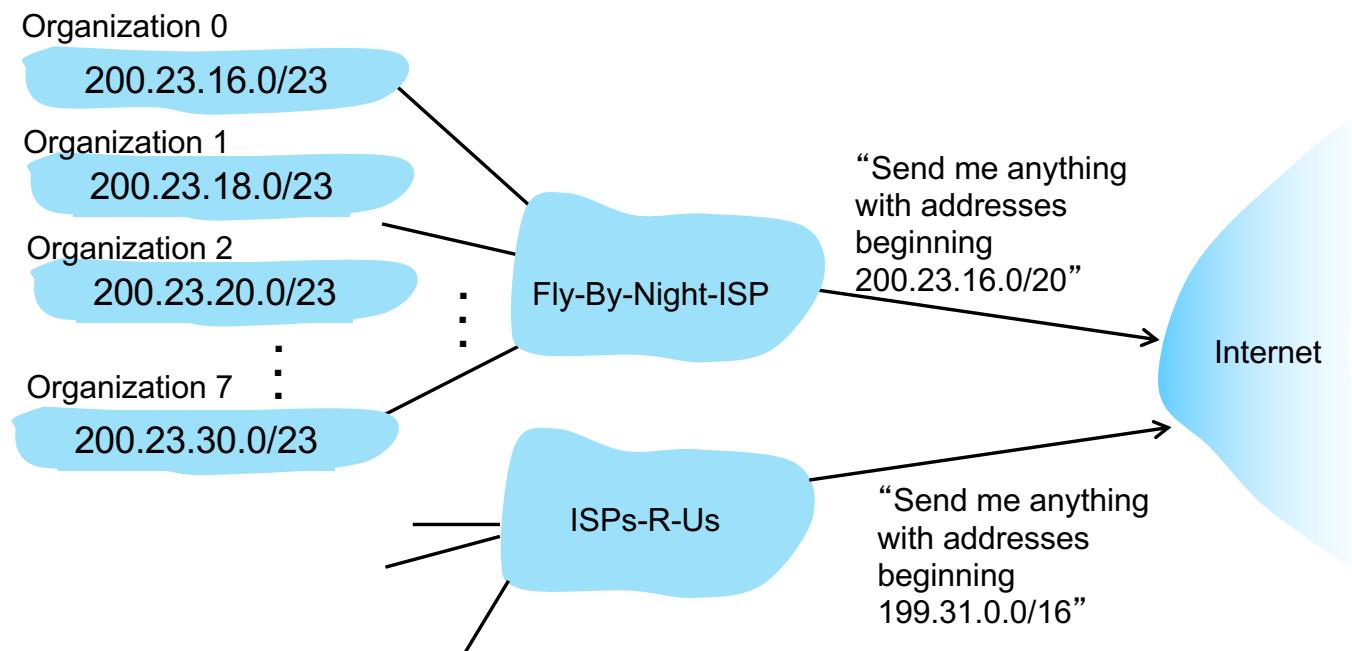
# CIDR: Addresses allocated in contiguous prefix chunks

Recursively break down chunks as get closer to host



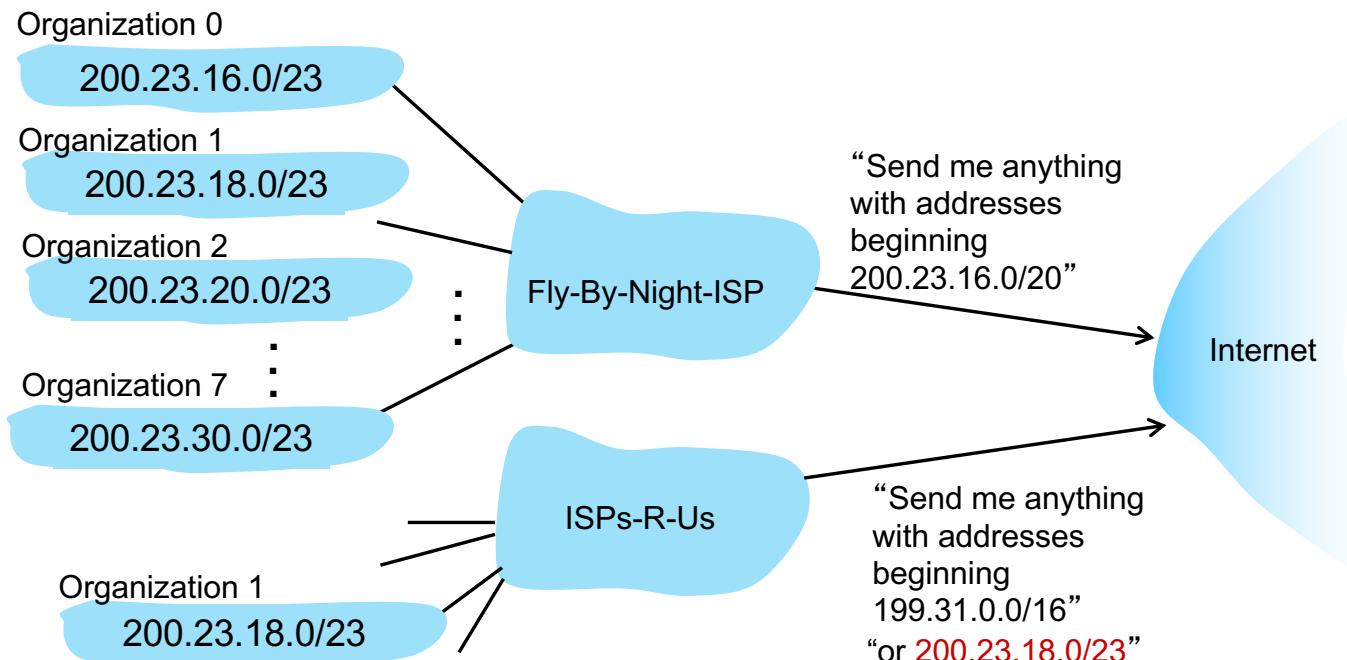
# Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



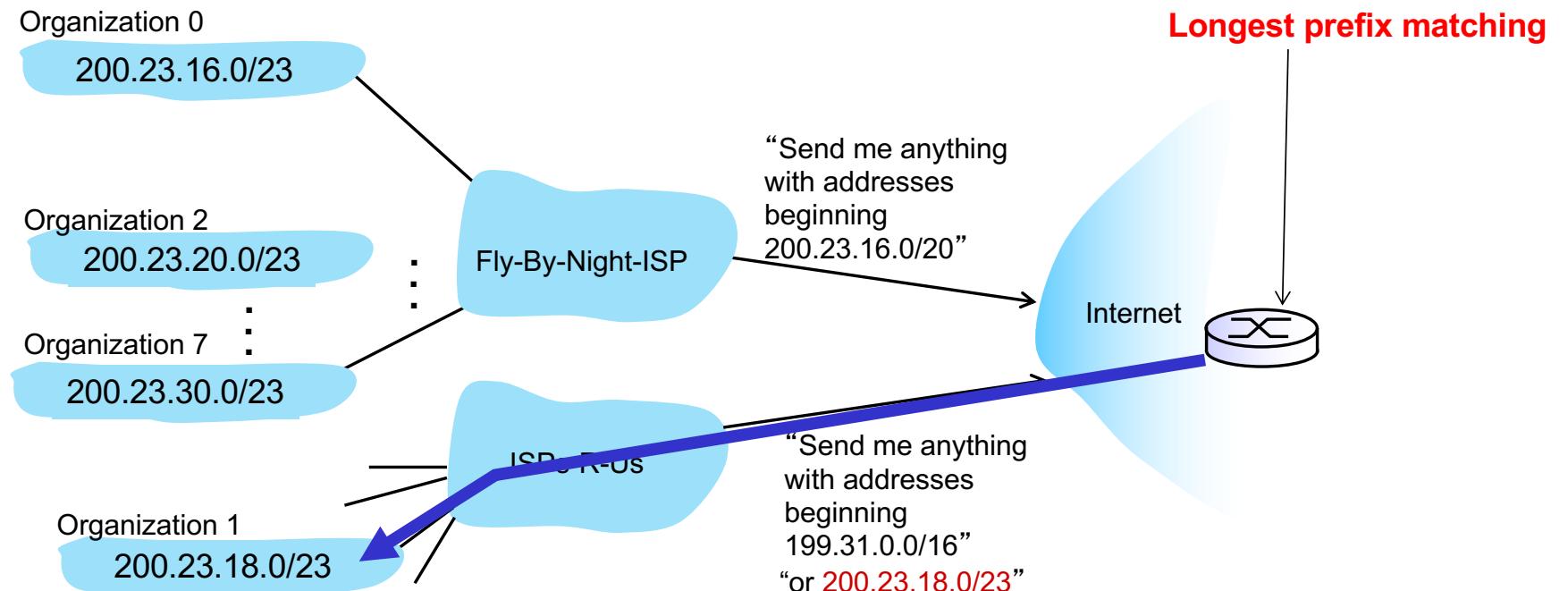
# Hierarchical addressing: more specific routes

- Organization 1 moves from Fly-By-Night-ISP to ISPs-R-Us
- ISPs-R-Us now advertises a more specific route to Organization 1



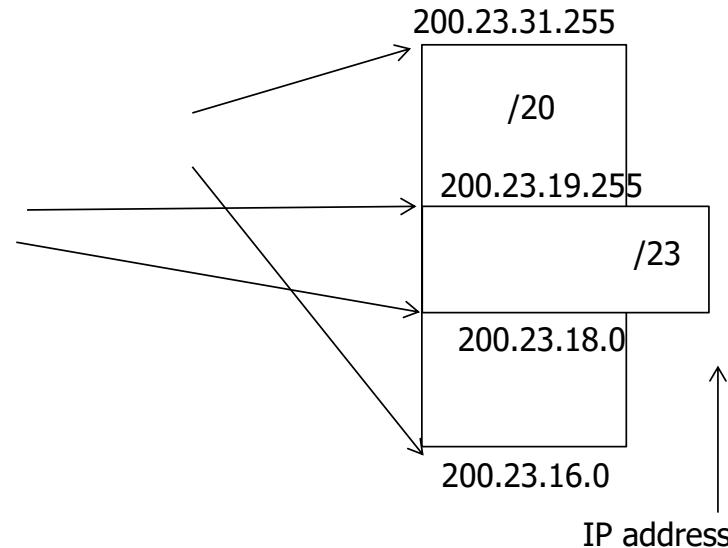
# Hierarchical addressing: more specific routes

- Organization 1 moves from Fly-By-Night-ISP to ISPs-R-Us
- ISPs-R-Us now advertises 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?

# More on IP addresses

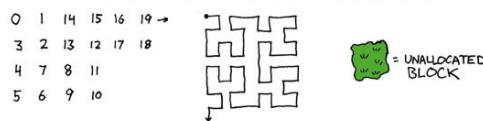
Source: www.xkcd.com

- 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>



THIS CHART SHOWS THE IP ADDRESS SPACE ON A PLANE USING A FRACTAL MAPPING WHICH PRESERVES GROUPING--ANY CONSECUTIVE STRING OF IPs WILL TRANSLATE TO A SINGLE COMPACT, CONTIGUOUS REGION ON THE MAP. EACH OF THE 256 NUMBERED BLOCKS REPRESENTS ONE /8 SUBNET (CONTAINING ALL IPs THAT START WITH THAT NUMBER). THE UPPER LEFT SECTION SHOWS THE BLOCKS SOLD DIRECTLY TO CORPORATIONS AND GOVERNMENTS IN THE 1990'S BEFORE THE RIPE TOOK OVER ALLOCATION.



# 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):



IRéseaux IP Européens Network Coordination Centre



American Registry for Internet Numbers



Asia-Pacific Network Information Center

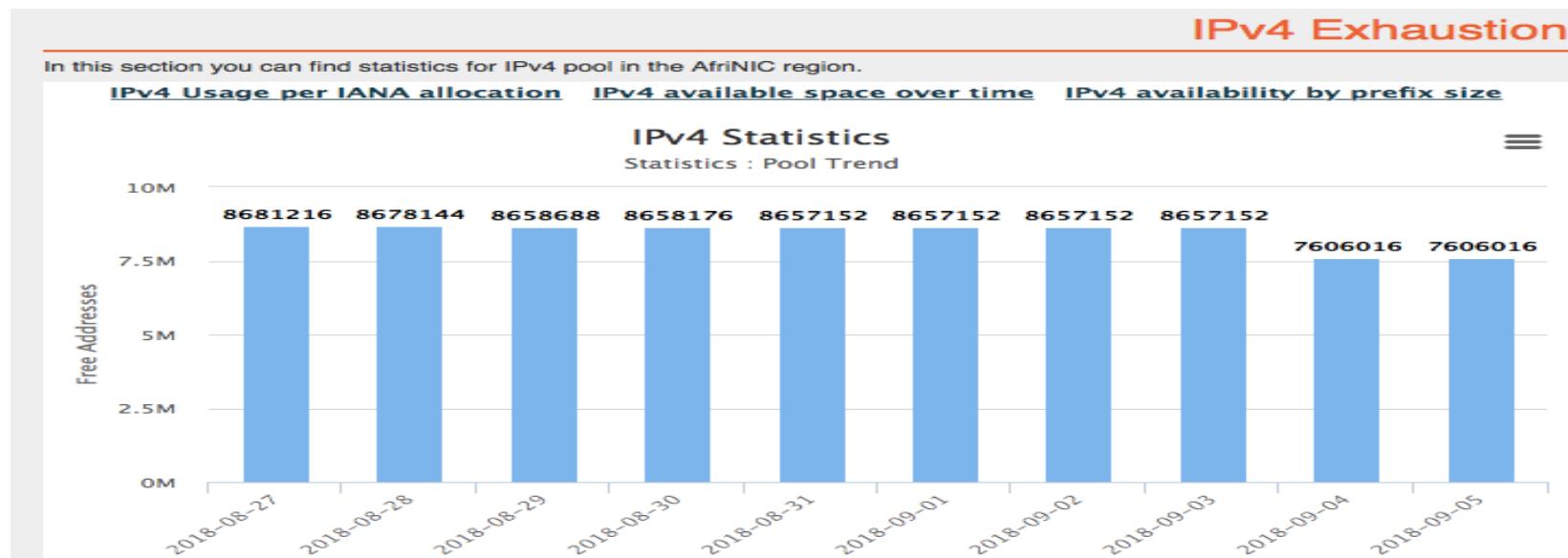
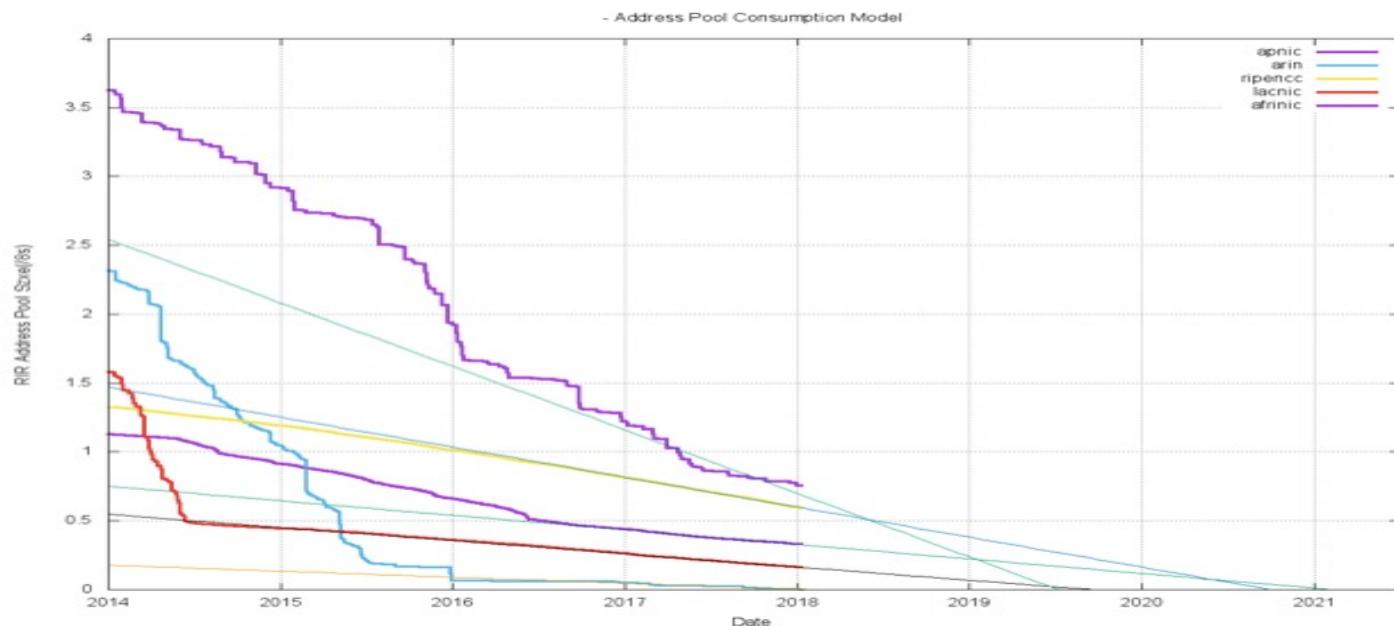


Latin America and Caribbean Network Information Centre



African Network Information Centre





# Made-up Example

- 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

# Second-hand IPv4 addresses reaping big bucks

By Juha Saarinen  
Sep 11 2015  
6:33AM

7 Comments 



## RELATED ARTICLES

[TPG founder David Teoh resigns from board](#)

[NBN Co uses stockpiled HFC NTDs to work down order backlog](#)

[OneWeb launches 36 satellites from Russia to extend internet orbit](#)

[Telstra warns NBN Co will hit fibre delivery](#)

## Up to US\$100 per IPv4 address could become the norm.

Network operators continue to demand IPv4 address space and are willing to pay rising prices and enter into potentially risky trading to obtain allocations of the older protocol.

Explosive growth of internet-connected devices has led to the worldwide pool of approximately 4.2 billion IPv4 addresses emptying out.

The situation was predicted many years ago, but it was never adequately dealt with by concerted migration to the newer IPv6 protocol, which holds a far larger numbering space.

The looming shortage has created a market for second-hand IPv4 addresses, which started in earnest in 2011 with Microsoft [famously paying](#) US\$11.25 per IP to bankrupt Nortel's receivers.



 ONEQODE™



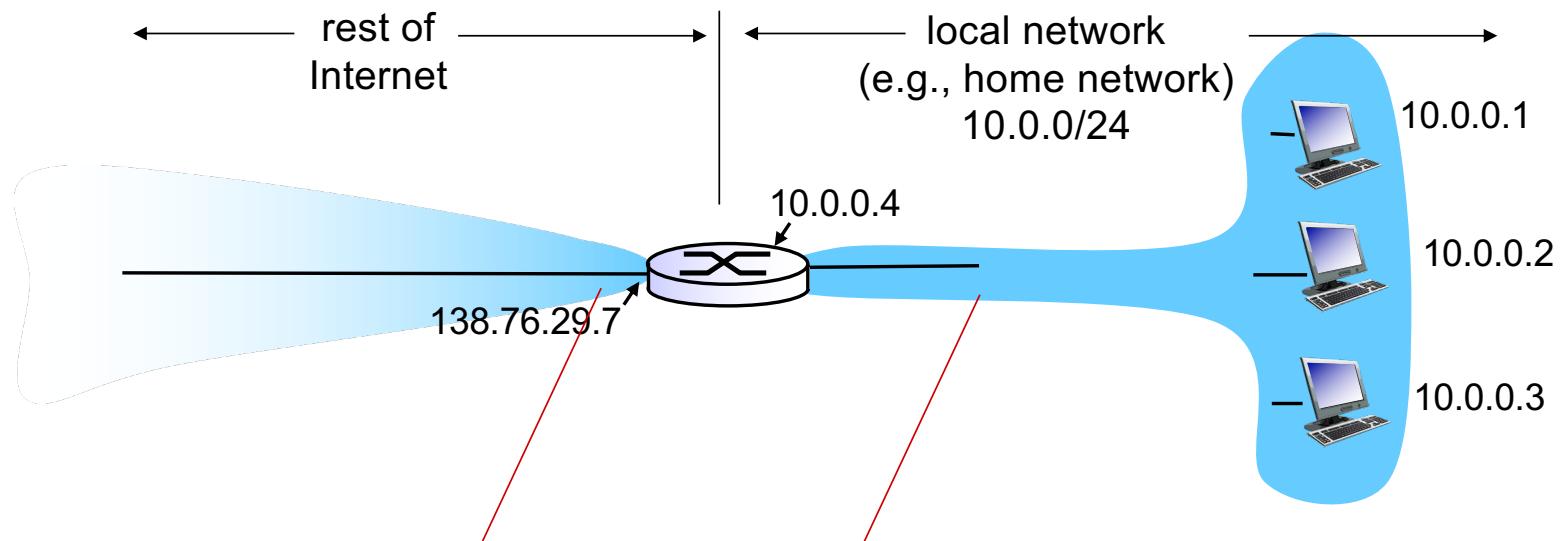
**Fastest transit path to Asia? Right here.**

**Global IP Transit with Australia's premium provider →**

# 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

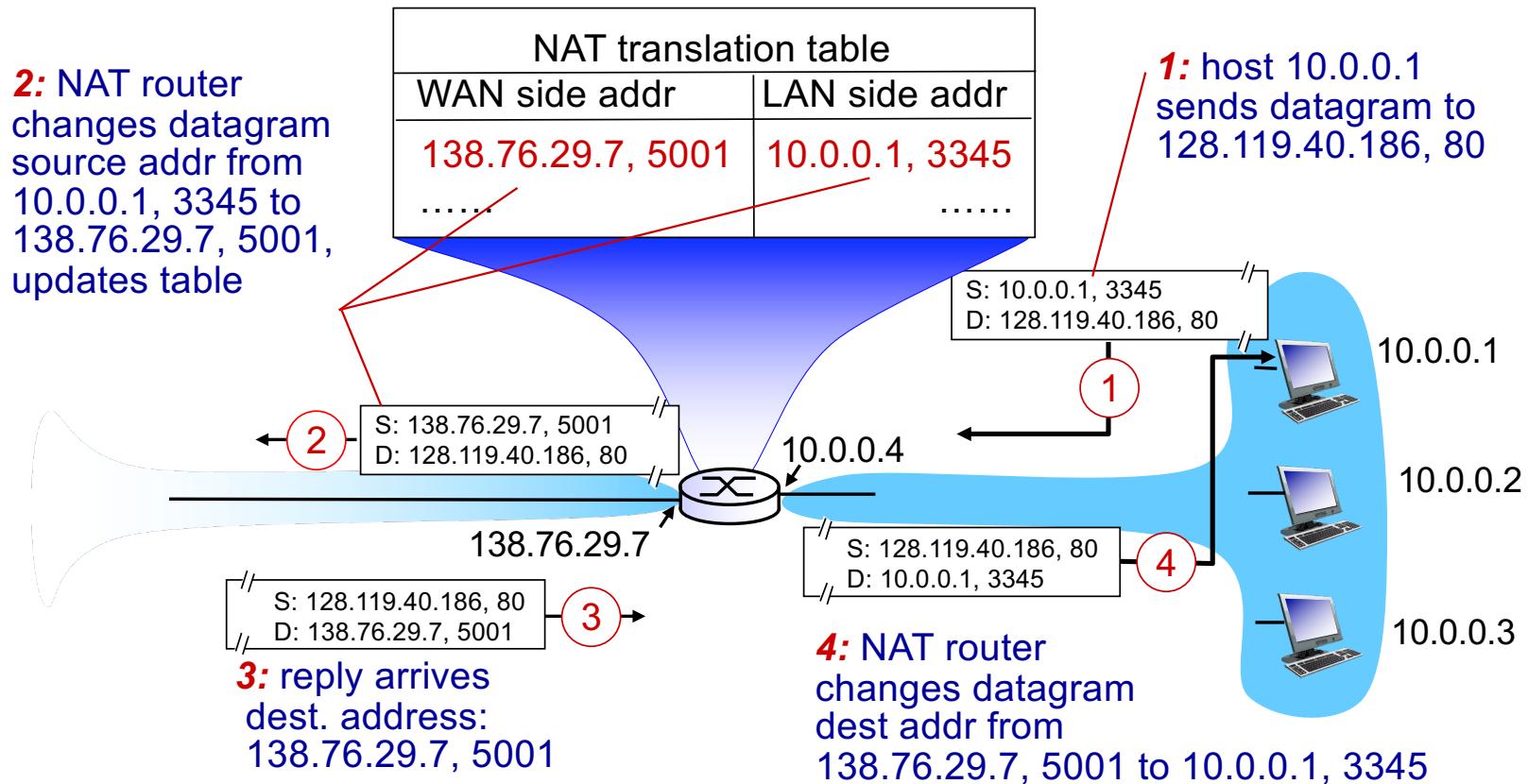
datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

# 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
  - 16-bit port-number field: ~65,000 simultaneous connections with one WAN-side address!
- 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

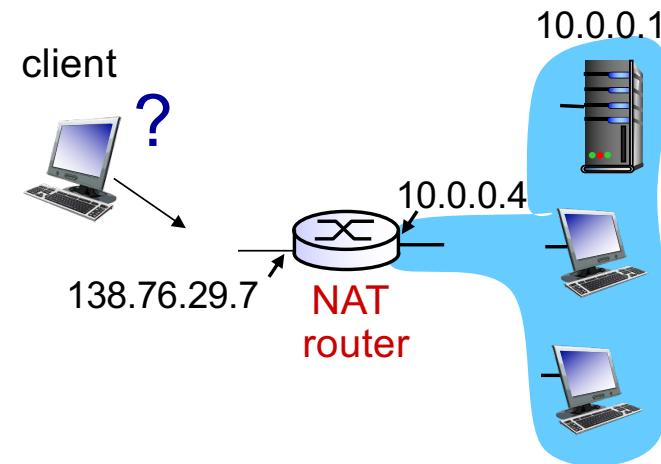
- NAT violates the architectural model of IP
  - Every IP address uniquely identifies a single node on Internet
  - routers should only process up to layer 3
- NAT changes the Internet from connection less to a kind of connection-oriented network
- NAT possibility must be taken into account by app designers, e.g., P2P applications

# 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

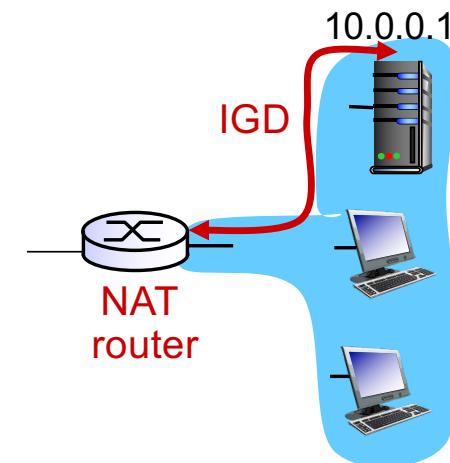
- client wants to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATed address: 138.76.29.7
- *solution1:* Inbound-NAT.  
Statically configure NAT to forward incoming connection requests at given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



# NAT traversal problem

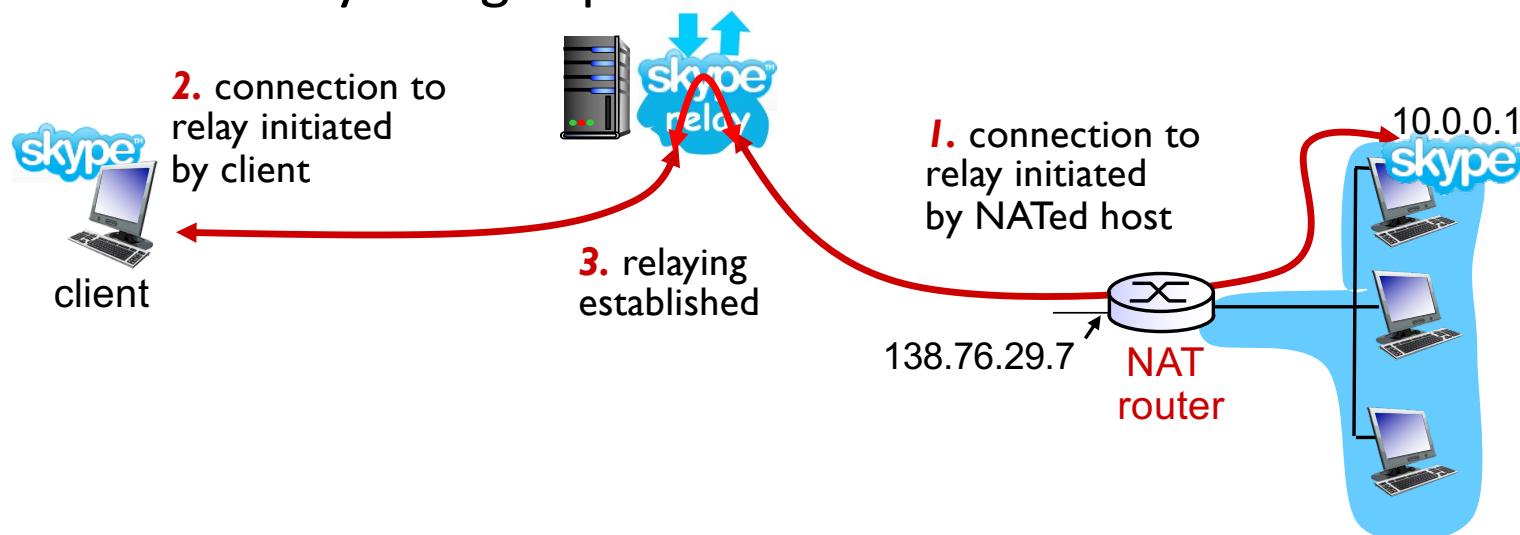
- *solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - ❖ learn public IP address (138.76.29.7)
  - ❖ add/remove port mappings (with lease times)

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 two connections



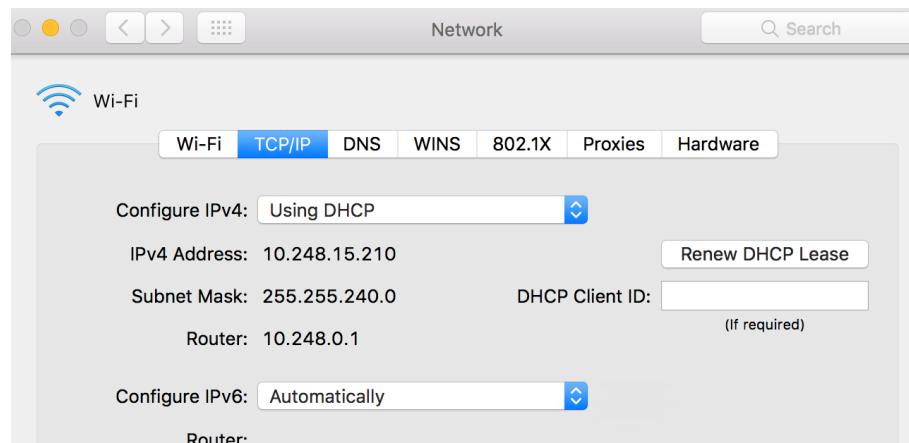
# NAT: Devil in the details

- 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 (NOT COVERED IN THE COURSE)



# Discussion

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



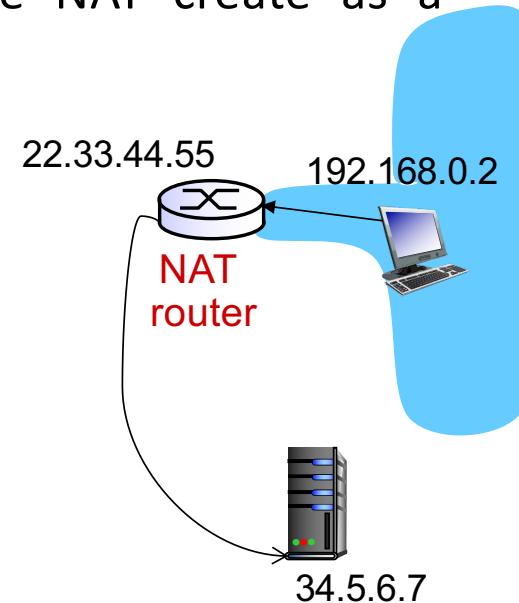
- However when I ask Google it says my IP address is as noted below. Can you explain the discrepancy?

129.94.8.210  
Your public IP address

# 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?
  - [22.33.44.55, 4567] → [192.168.0.2, 80]
  - [34.5.6.7, 80] → [22.33.44.55, 4567]
  - [192.168.0.2, 80] → [34.5.6.7, 4567]
  - [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  $[22.33.44.55, 3967] \rightarrow [192.168.0.2, 4567]$  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

