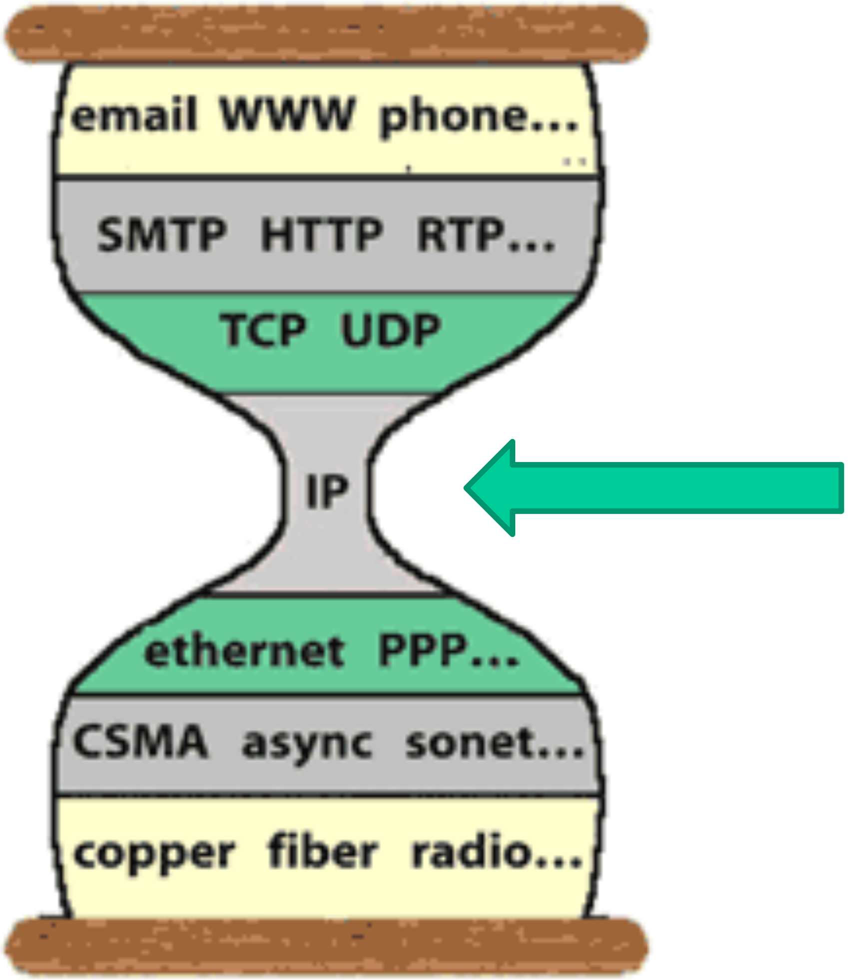
Lecture 9: Network Layer



Chapter 4:

4.1 Overview of Network layer

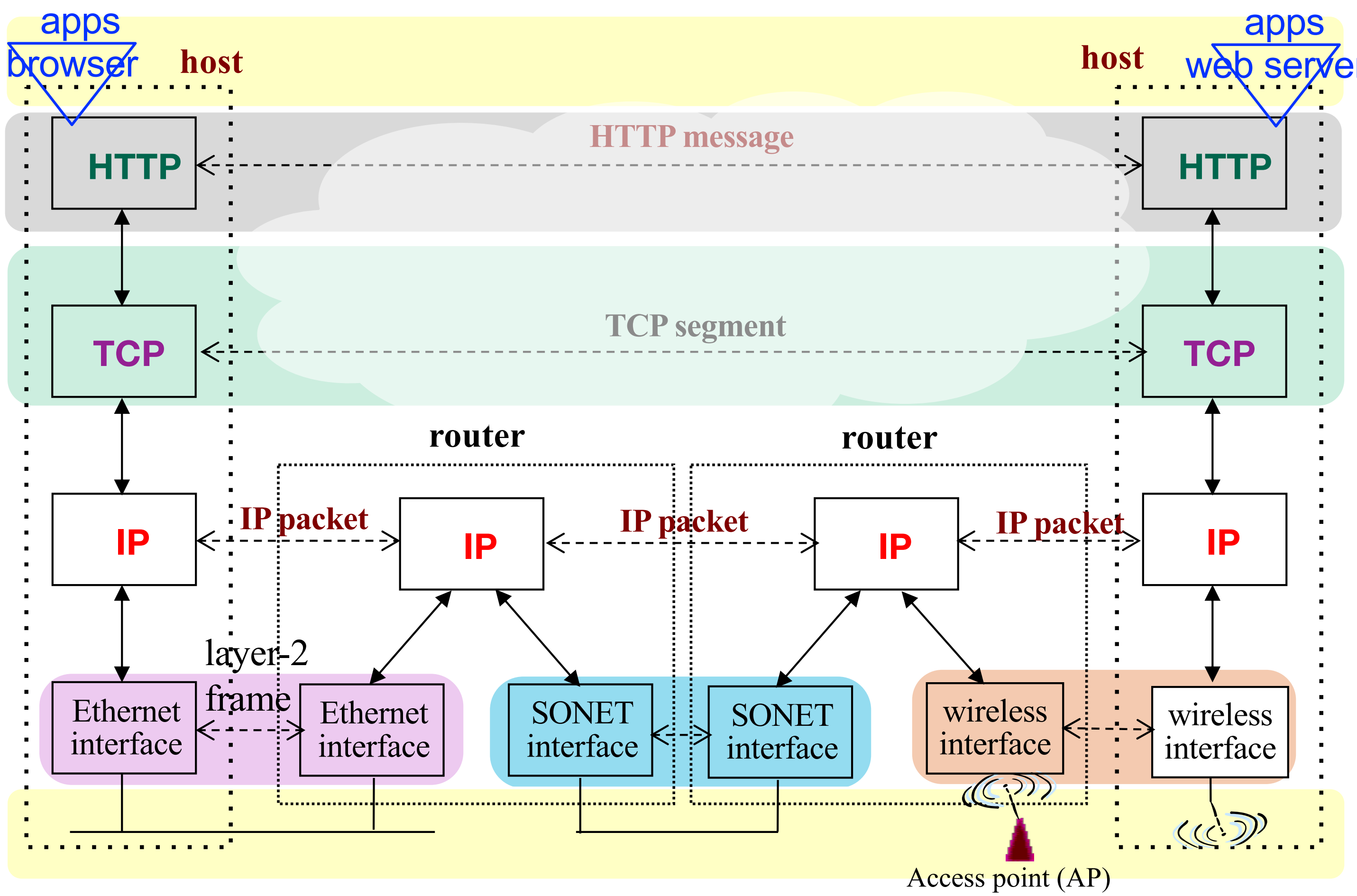
4.2 What's inside a router

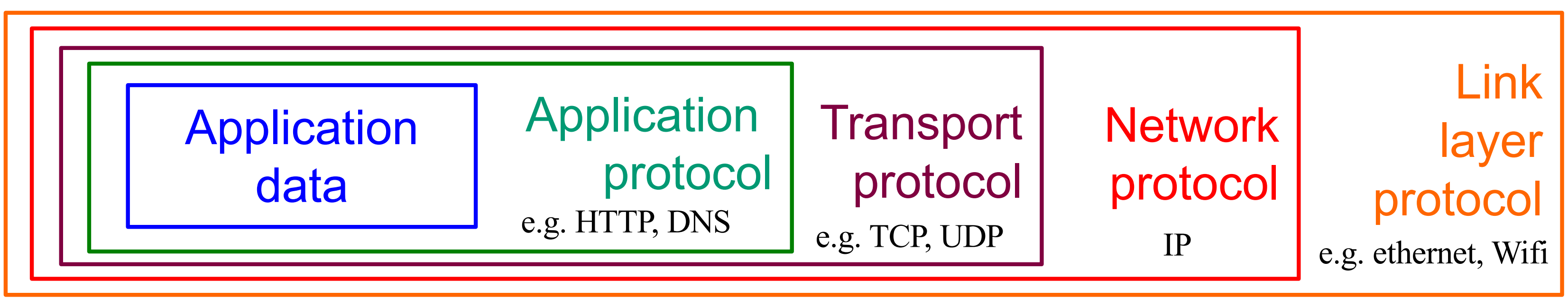
4.3 IP: Internet Protocol

- IP packet format
- fragmentation
- IPv4 addressing
- network address translation
- IPv6

4.4 Generalized Forward and SDN

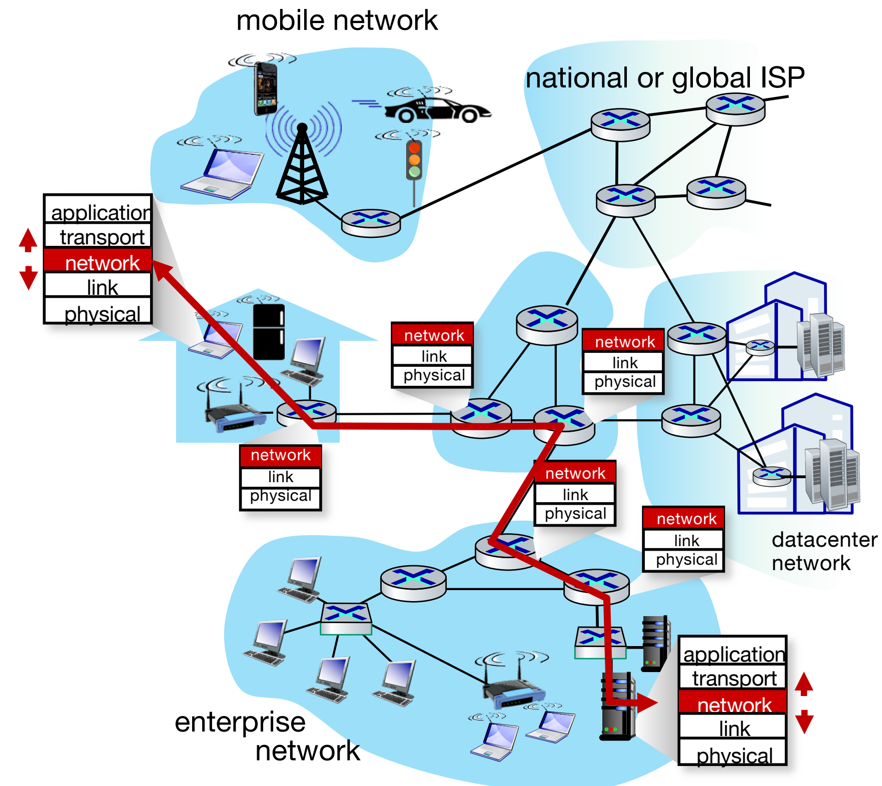
Always keep in mind the big picture





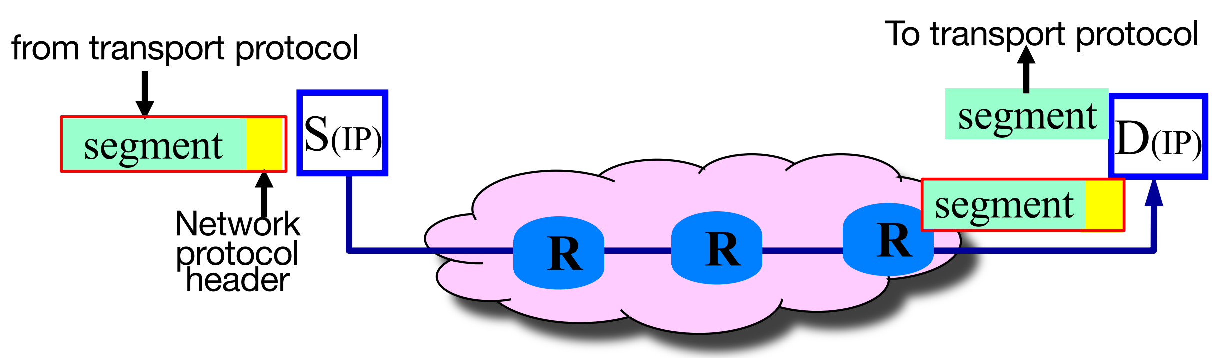
Moving Datagrams Hop-by-Hop

- 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



Network layer

- Network layer protocols: run in *every* host and router
- Source host: encapsulates data segments from transport layer into *IP packets*
- Destination host: decapsulates received IP paclet (remove IP header), delivers segments to transport layer
- Each router along the path: move packets from sending towards receiving host
  - Routing: fill in router's forwarding table (FIB) with the best path to each destination
  - Forwarding: use the destination address in each packet to look up FIB to find next hop along the best path



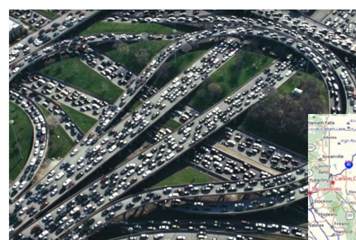
Two functions

- ***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*
  - *Routing protocols*

analogy: taking a trip

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



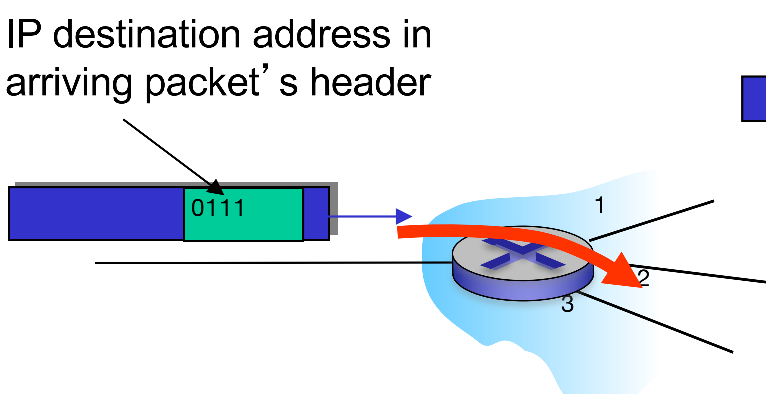
forwarding

routing

Data plane and control plane

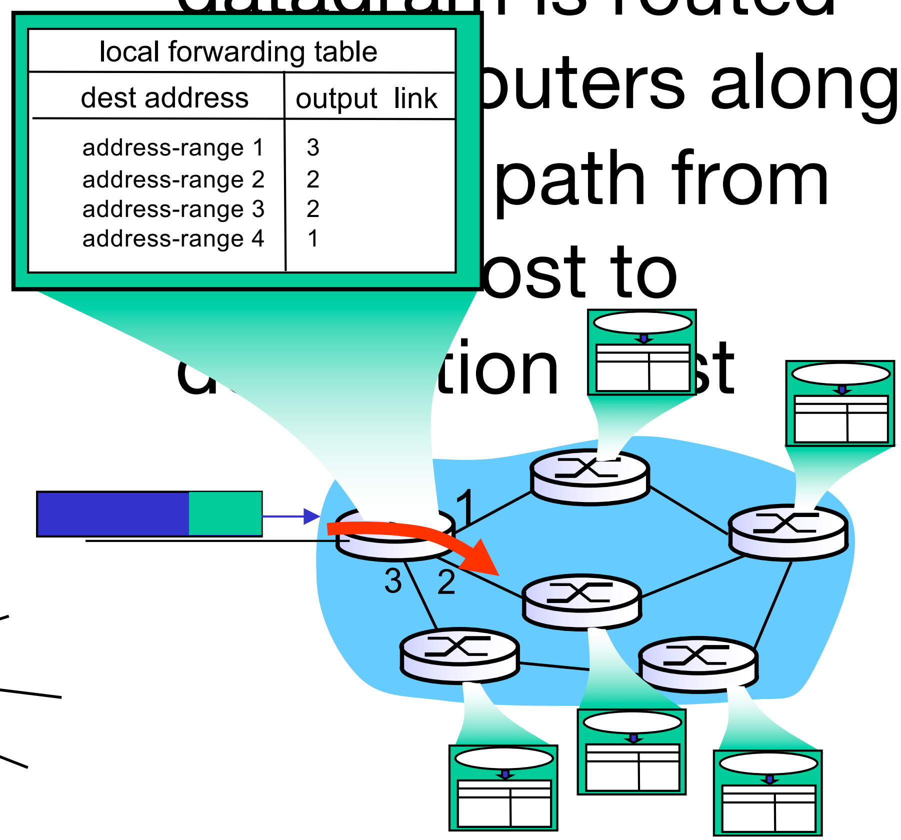
Data plane: forwarding

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



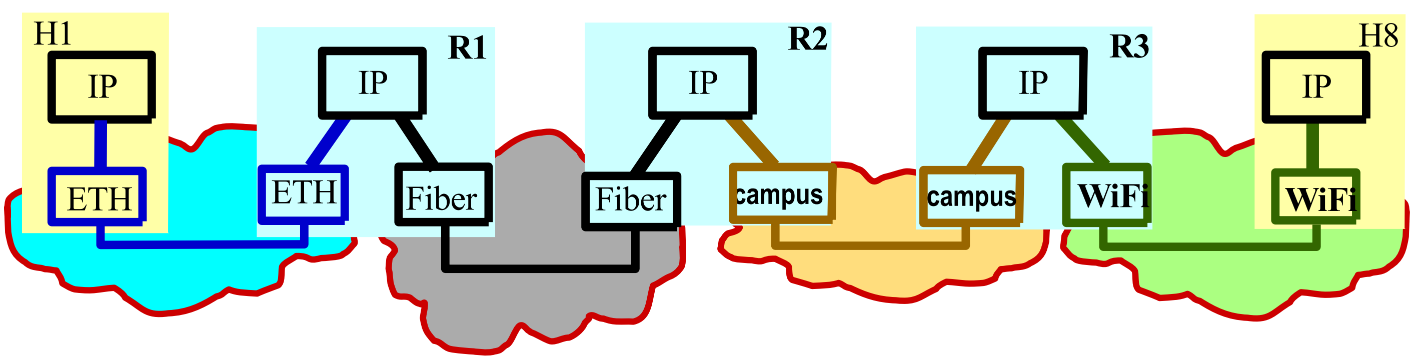
Control plane: routing

- *network-wide* logic
- determines how datagram is routed
  outers along
  path from
  ost to
  d tion st

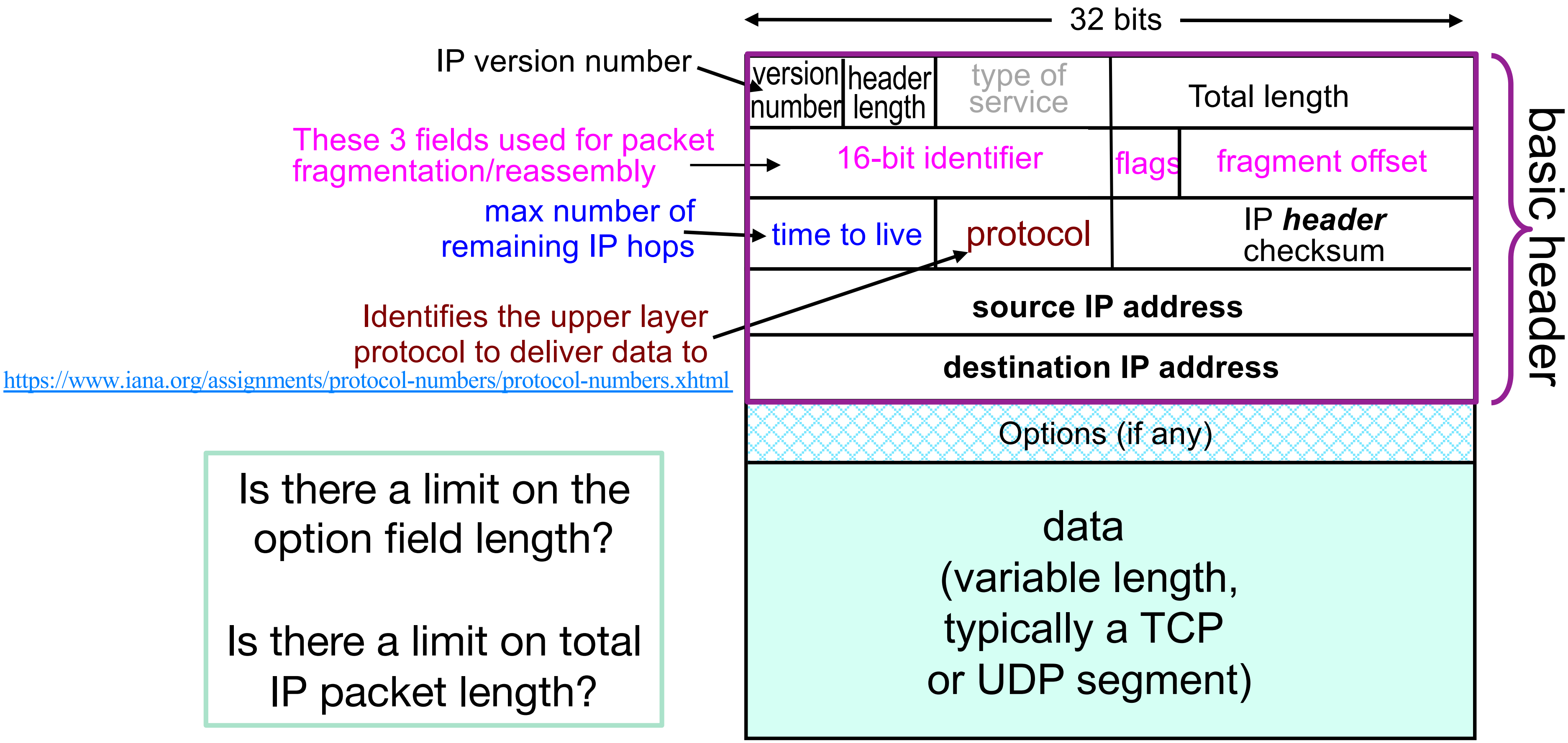


Internet: a datagram network

- hosts are connected to subnets
- subnets are interconnected by routers
- All hosts and routers speak IP
- IP provides two basic functions
  - *Datagram* delivery from source to destination hosts identified by IP addresses
  - Fragment packets along the way *if* needed, reassemble at destination host before passing to transport



IP datagram format



Is there a limit on the option field length?

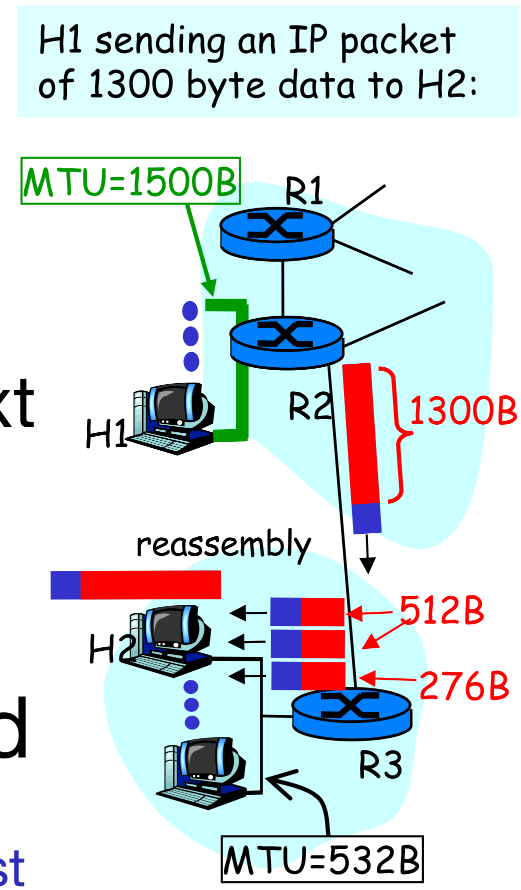
Is there a limit on total IP packet length?

Notes on IP header fields

- Why is TTL field needed?
  - Not measured in wall-clock time, but in hop count
  - Preventing IP packets from staying inside the Internet forever
- Why are both header length field and length field needed?
  - Both headers and payloads can have variable lengths
- Why do we need header checksum field, and upper layer field?
  - Verify the header checksum, know what transport layer to use
- What is the purpose of having three fields of 16-bit identifier, fragment offset, and 3-bit flags?
  - Fragment and re-assembly large IP packets

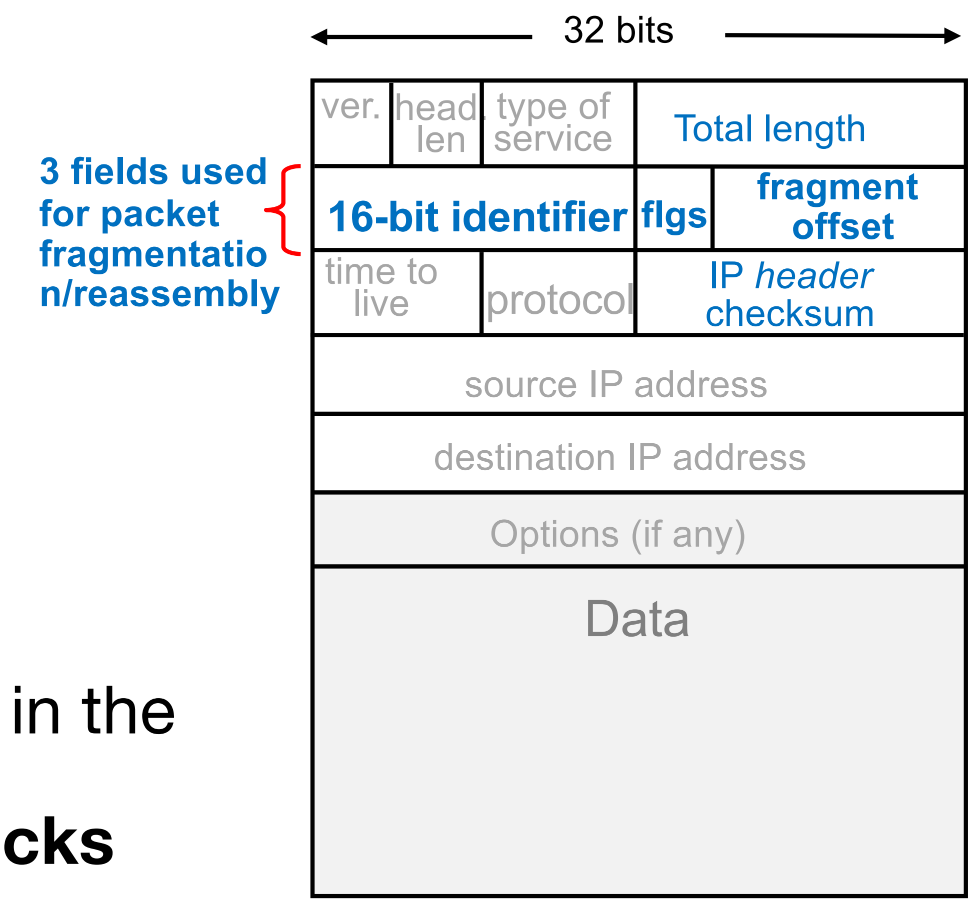
IP Fragmentation & Reassembly

- Different **links** have different Maximum Transmission Unit (MTU)
- Sending host uses its local MTU size
- if the next link has a smaller MTU, routers fragment IP packets
  - chop packets to the MTU size of next link
  - further fragmentation down the path possible
- packet fragments are reassembled at destination host
  - Receiving host sets a timer when receiving the first segment of an IP packet
  - When assembles a full packet: pass to transport
  - When timeout: delete received segment(s)

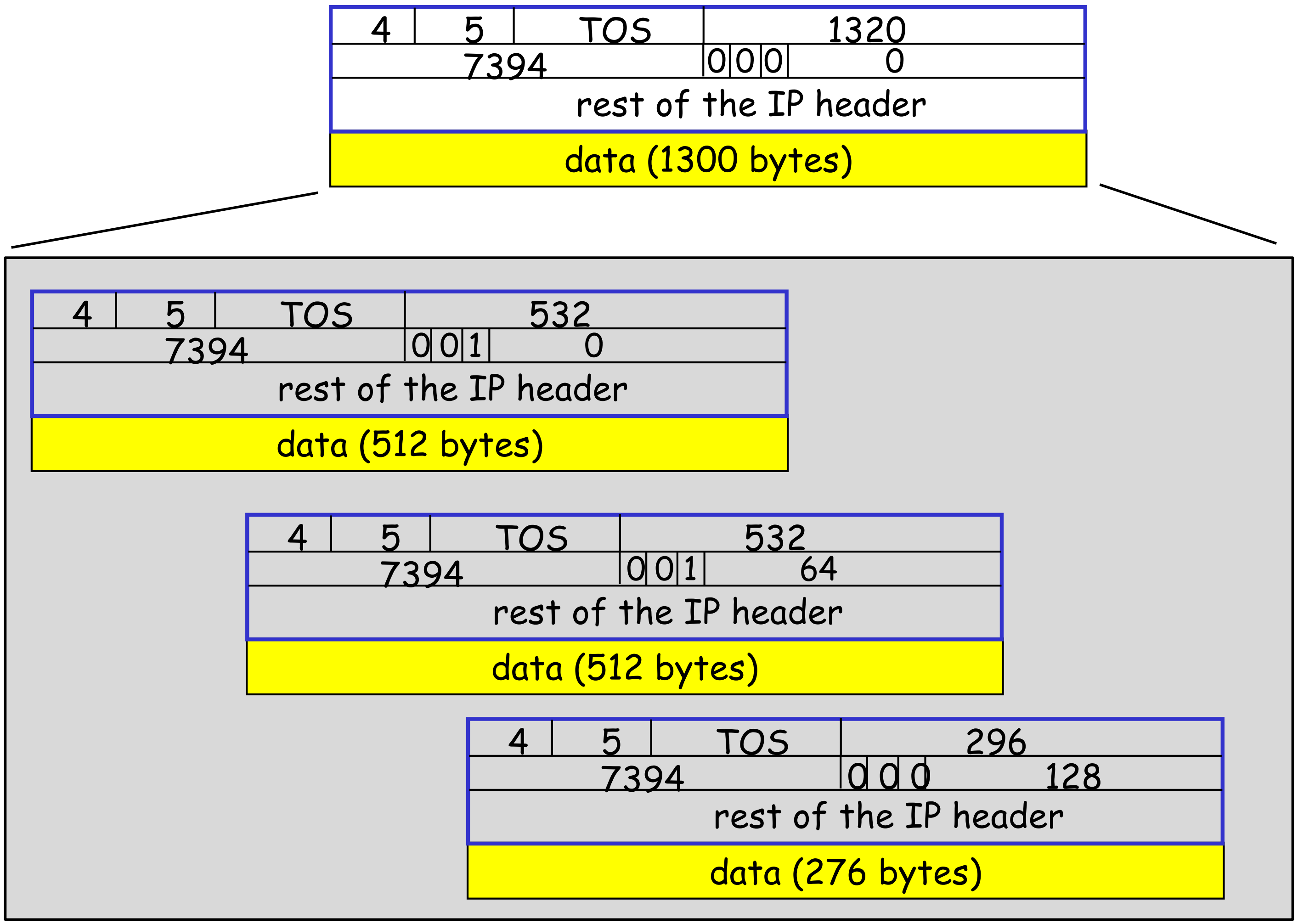


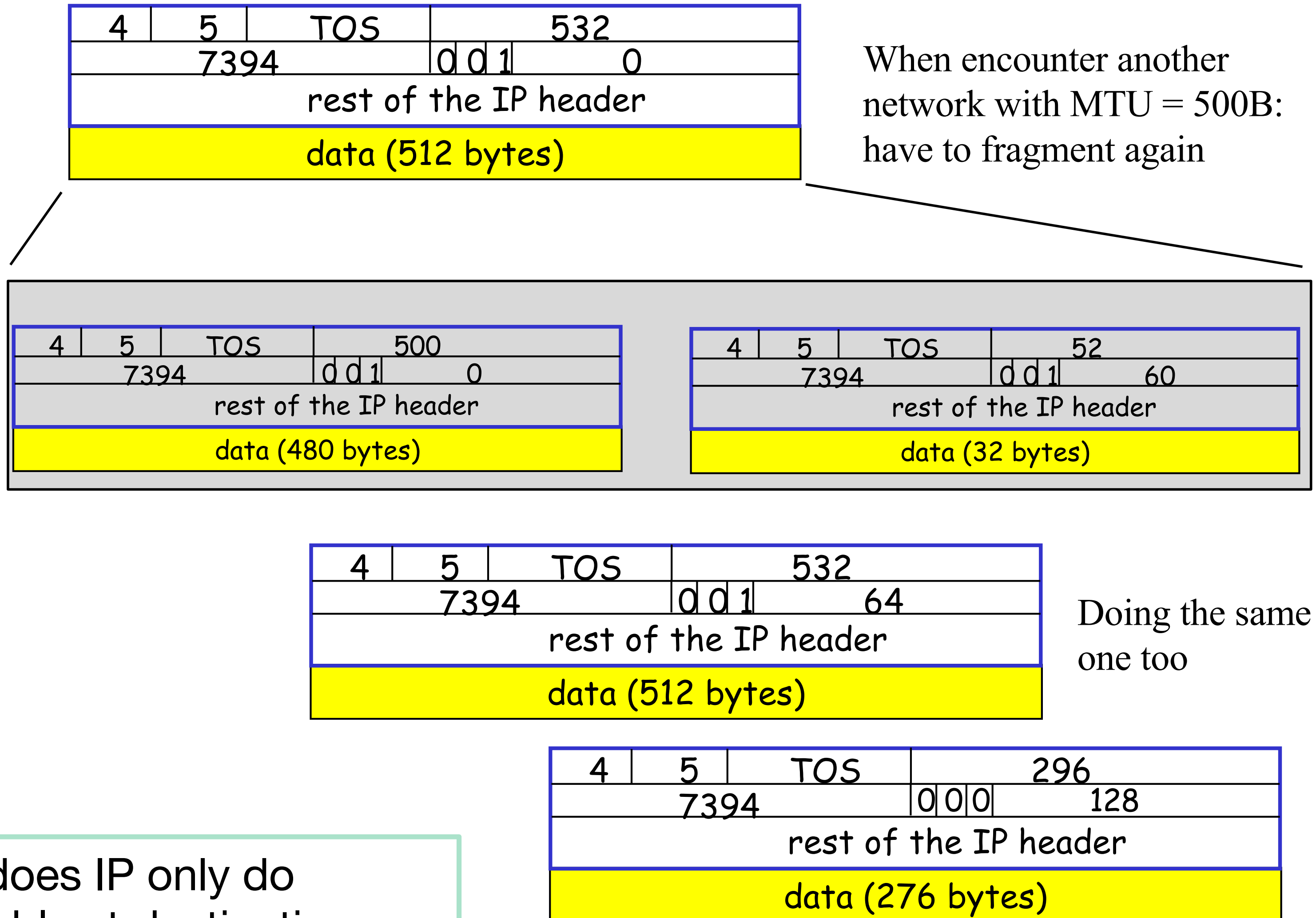
IPv4 Fragmentation Details

- Identifier: generated by the sending host
  - identify all the segments in the same IP packet
  - Stay unchanged when re-fragmented
- Flags
  - bit 0: reserved left most bit
  - bit 1: don't fragments
  - bit 2: more fragments (MF)
    - MF=0 in the last fragment
- Fragment offset
  - counting from the first byte in the original payload
  - count in units of **8-byte blocks**
- Total length & header checksum adjusted when packet fragmented



IPv4 Fragmentation: an example





When encounter another network with MTU = 500B: have to fragment again

Doing the same for this one too

Where does IP only do reassembly at destination host?

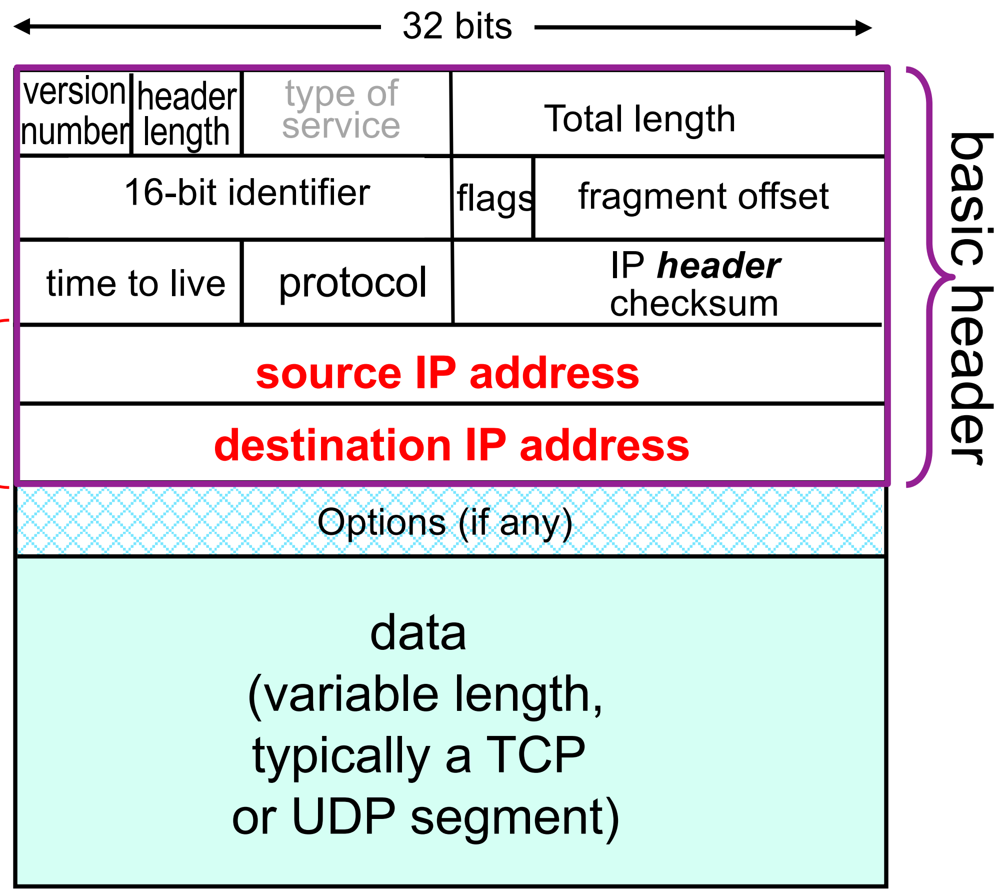
Summary of IP Fragmentation

- At the time of IPv4 design, fragmentation was considered important
  - Different network links and subnets have different MTUs
- What's good about in-network fragmentation
  - Allows packets getting through networks with MTUs smaller than the size of original packets
- What's bad about in-network fragmentation
  - Can be fragmented and re-fragmented to many pieces
  - Losing a single fragment → losnig the whole IP packet
    - resources used in forwarding non-lost fragments wasted

FYI today's practice: avoid in-network IP packet fragmentation

IP datagram format: IP address

most changed fields in the IP header (in meaning and in format)



IP: addressing network interfaces

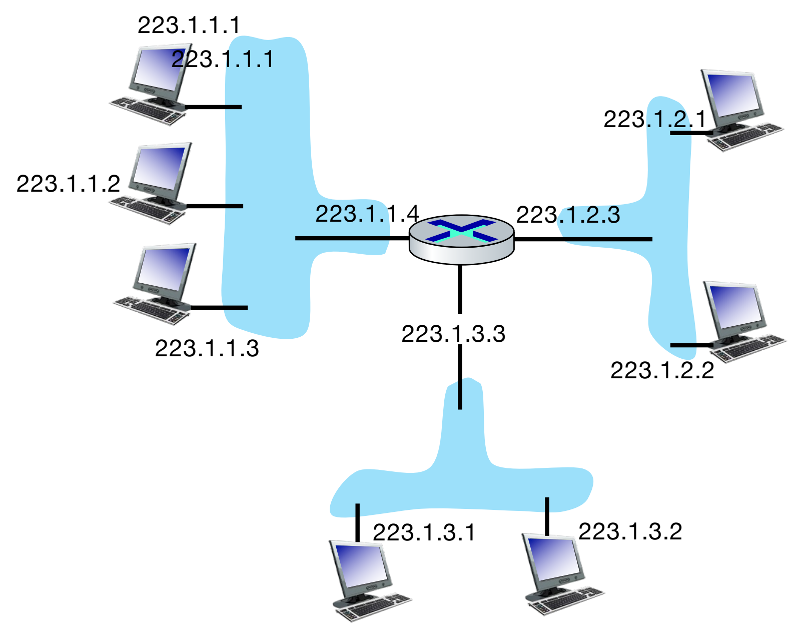
IP address:

32-bit identifier associated with each network (host or router) *interface*

Network 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)



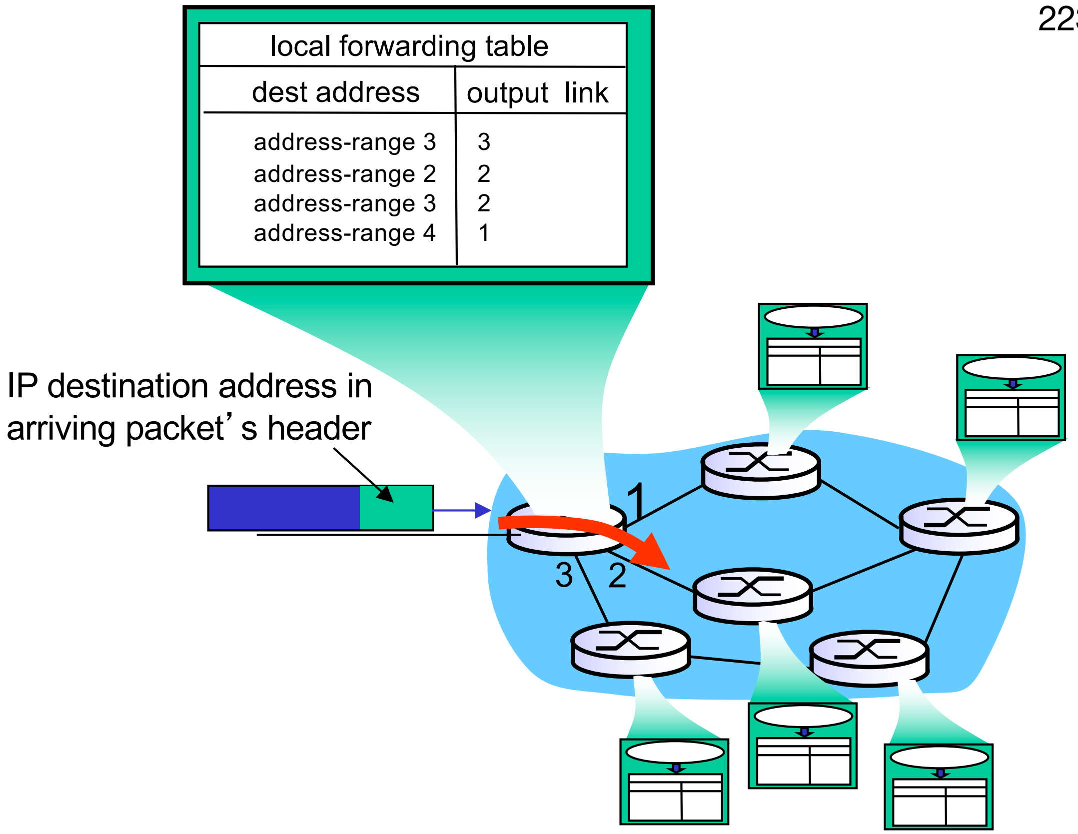
dotted-decimal IP address notation:

$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP ranges in forwarding tables

32-bits, uniquely identifies a host or network *interface*

– *interface:* connecting point between host/router and physical link



223.1.1.1 = $\underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$

- 4 billion IP addresses
- Routers list ***range*** of addresses (address block), with the output link/interface towards the destinations

IP addresses: how to get one?

That's actually two questions:

Q1: How does a *host* get IP address within its network

Q2: How does a *network* get IP address **ranges** for itself

How does *host* get IP address?

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

How the network gets IP address ranges?

- Internet Service Providers (ISPs), and some large user sites, get blocks of IP addresses from the Regional Internet Registries (RIRs)
- Internet customers get a sub-block from their ISP's address block
- Network portion can take any arbitrary number of bits
  - Note that this split is **logical** and always from the perspective of the specific router

An example:

ISP's block	11011111 00000001 00000000 00000000	12 bits "free space"
Organization 0	11011111 00000001 00000000 00000000	10 bits to address interfaces
Organization 1	11011111 00000001 00000100 00000000	
Organization 2	11011111 00000001 00001000 00000000	
Organization 3	11011111 00000001 00001100 00000000	

IP addressing: CIDR and subnetting

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

network part — host part

11011111 00000001 000000**00** **00000000**

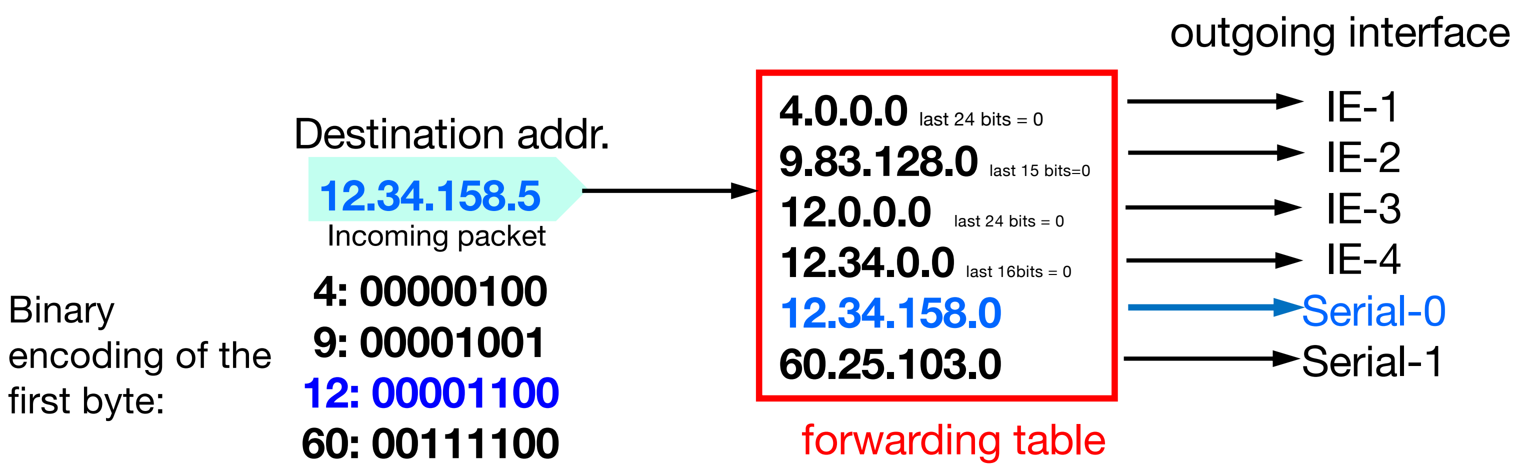
223.1.0.0/22

Subnetting (discuss shortly)

223.1.0.0/24	11011111	00000001	000000**00**	00000000
223.1.1.0/24	11011111	00000001	000000**01**	00000000
223.1.2.0/24	11011111	00000001	000000**10**	00000000
223.1.3.0/24	11011111	00000001	000000**11**	00000000

IP Packet Forwarding: longest match lookup

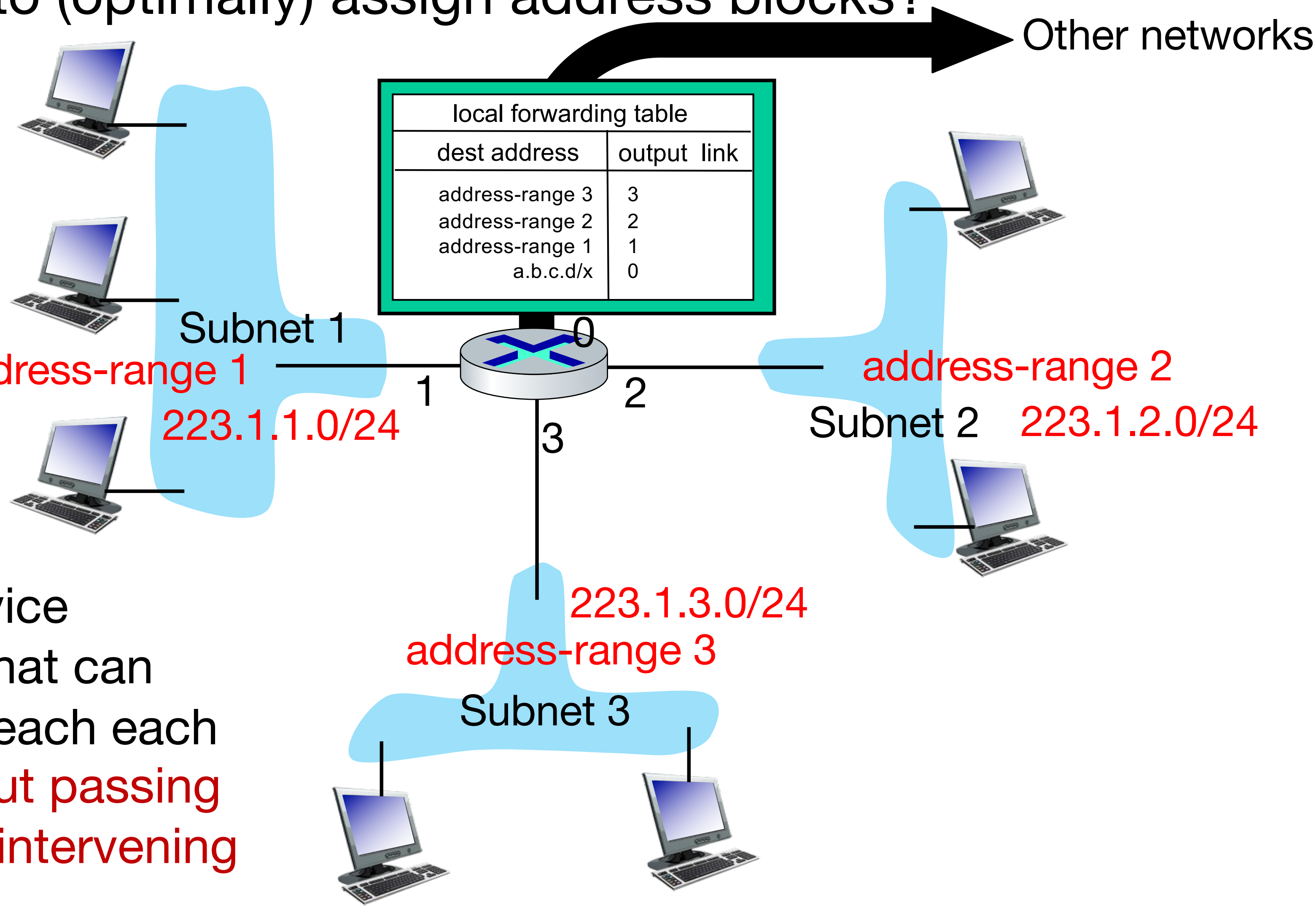
- Destination-based forwarding
  - *Only* look at destination address
- Routing protocol builds forwarding table (FIB) in all routers
  - FIB: mapping each IP prefix to an outgoing interface
- To forward a packet: Router finds longest-matching prefix entry for the destination address



If you were the local network operator

223.1.0.0/22 (11011111 00000001 000000**00 00000000**)

Q1: How to (optimally) assign address blocks?

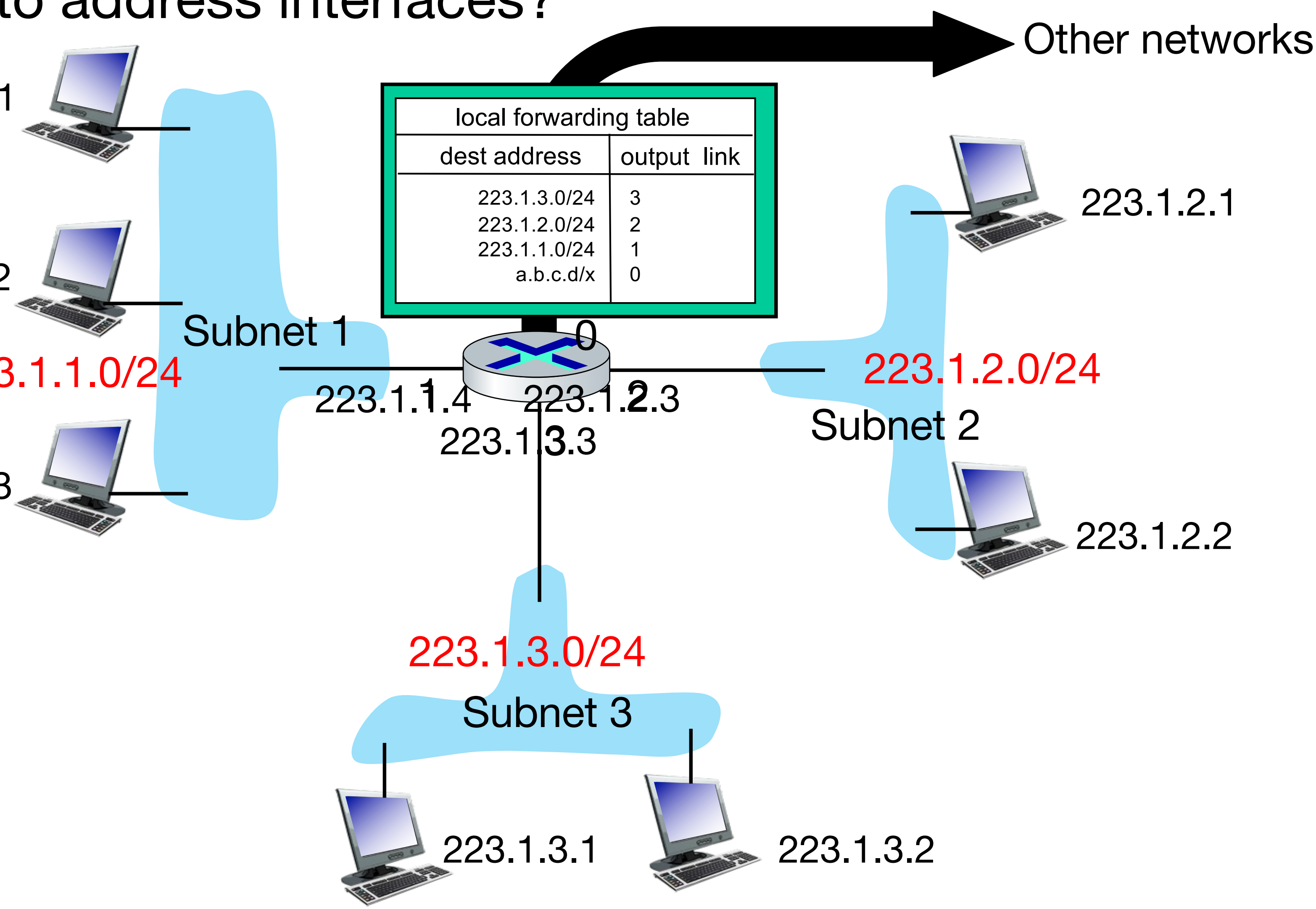


Subnet: device interfaces that can physically reach each other without passing through an intervening router

If you were the local network operator

223.1.0.0/22 (11011111 00000001 000000**00 00000000**)

Q2: How to address interfaces?

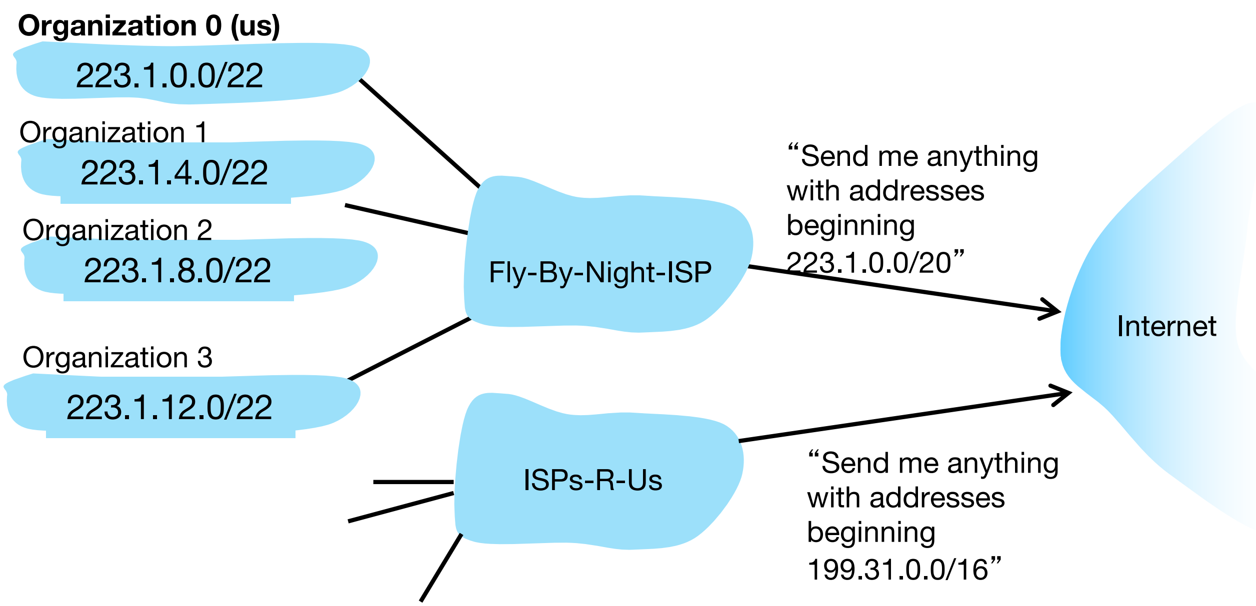


Special Addresses

- 255.255.255.255/32
  - broadcast address of "this network" determined by subnet mask
- last address of the network (e.g., 223.1.1.255 for 223.1.1.0/24)
  - broadcast address for the network
- first address of the network (e.g., 223.1.1.0 for 223.1.1.0/24)
  - network address (as a convention, not assigned to end-hosts)
- 0.0.0.0: *indicating "default route"*
  - Used in packet forwarding when no specific route can be determined for a given IP destination

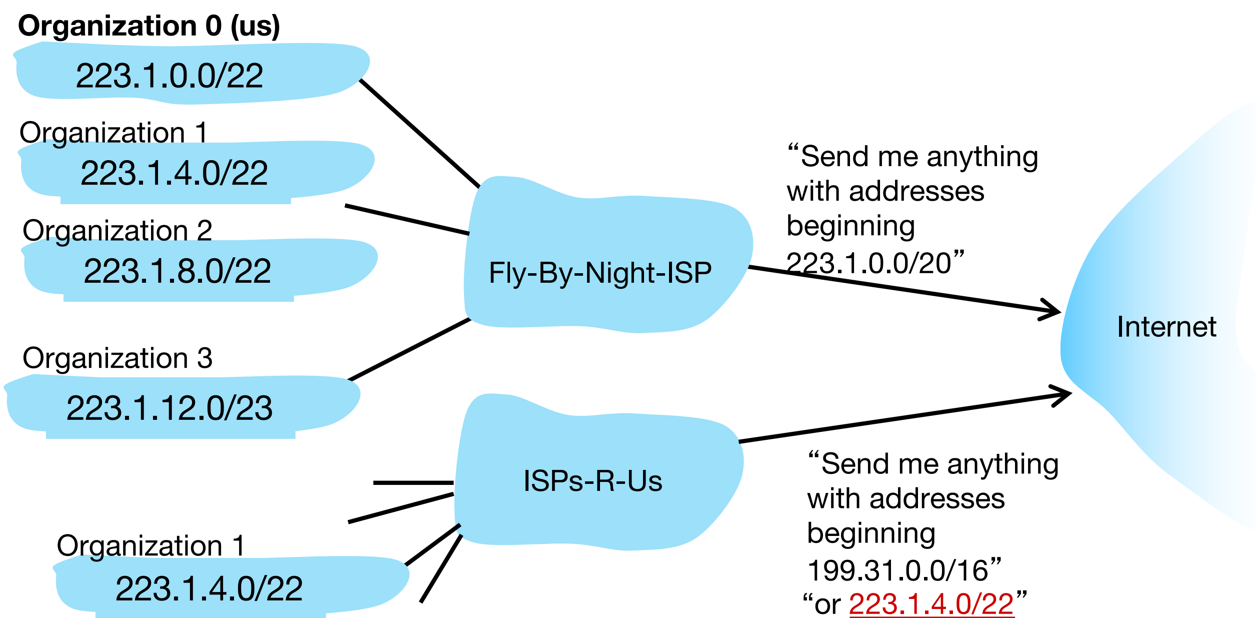
Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



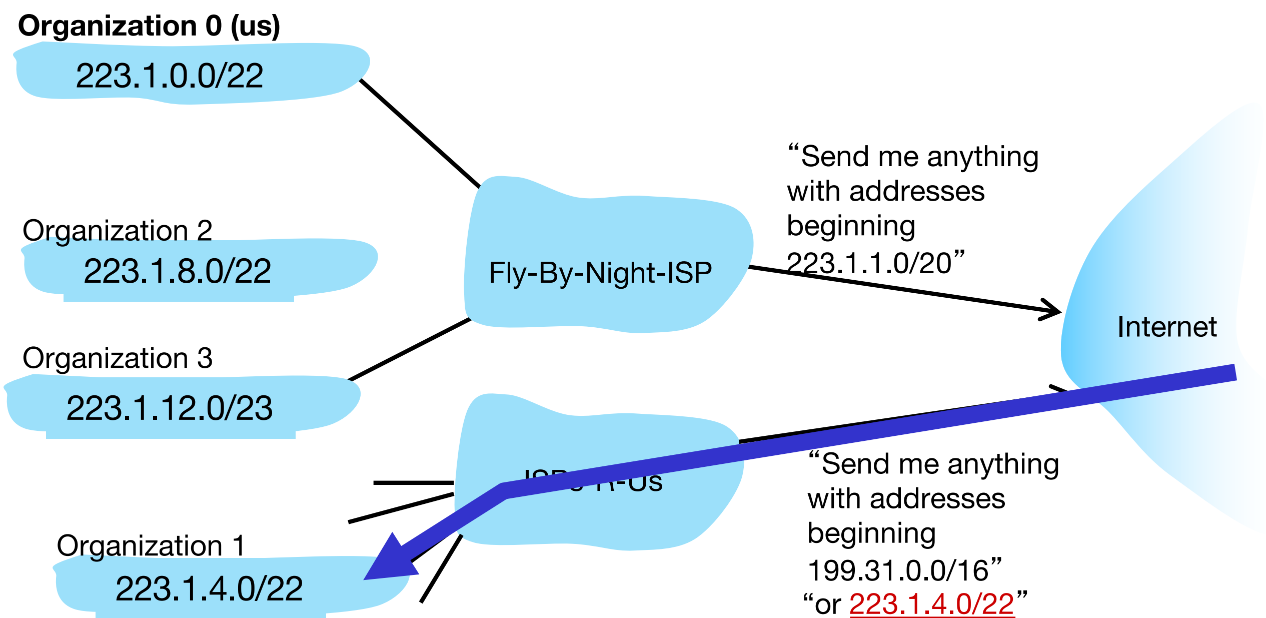
Hierarchical addressing: route aggregation

- 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: route aggregation

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



Summary: why we need CIDR, subnetting

important

and how to distinguish the two

- Two different solutions to the same goal: more efficient use out of limited IP address space
  - By deciding the #bits in an IP address for network ID
- Subnetting: network operators configure a subnet mask to the routers within the destination network the length of each address block (network ID) in the router's forwarding table
- Classless InterDomain Routing:
  - Routing protocols tell routers the length of each address block (network ID) in the router's forwarding table (e.g. 31.179.0.0/16)

Some notes and clarifications

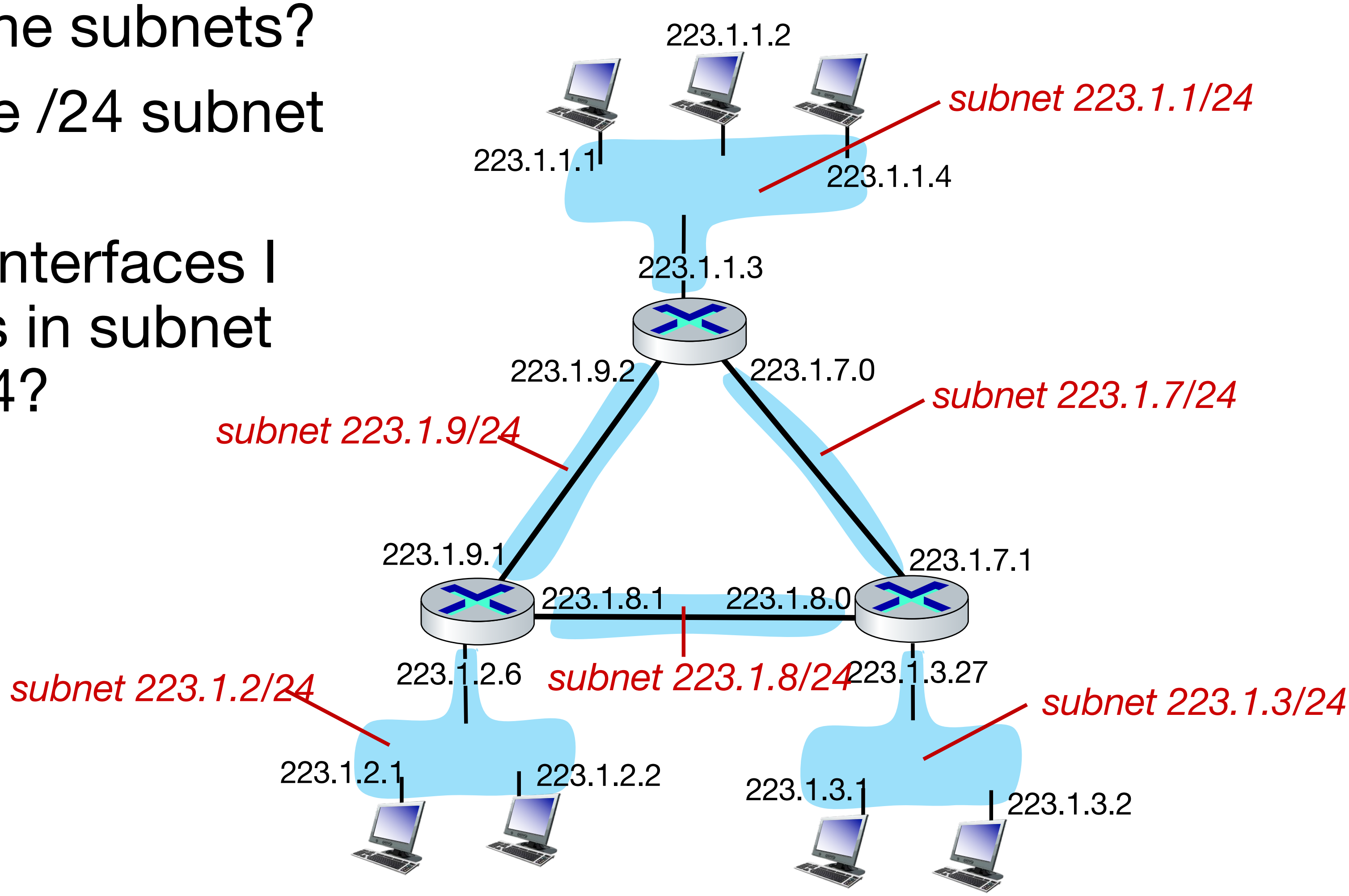
- You configure router interfaces with address and subnets first
  - Configure link1 223.1.1.4 255.255.255.0
    - Another presentation of /24, used **only in subnetting**
  - Configure link2 …
- Then configure individual host
  - Printer
    - address 223.1.1.1/24
    - router 223.1.1.4 (if dest not in local subnet, go this way)
  - PC
    - address 223.1.1.2/24
    - router 223.1.1.4
- At the end, router runs routing algo to build FIB

Few more words on subnets

- Subnets are determined by physical connectivity
  - Subnets pre-exist before you address them
- Subnets are identified by address block
- A random address block != a subnet

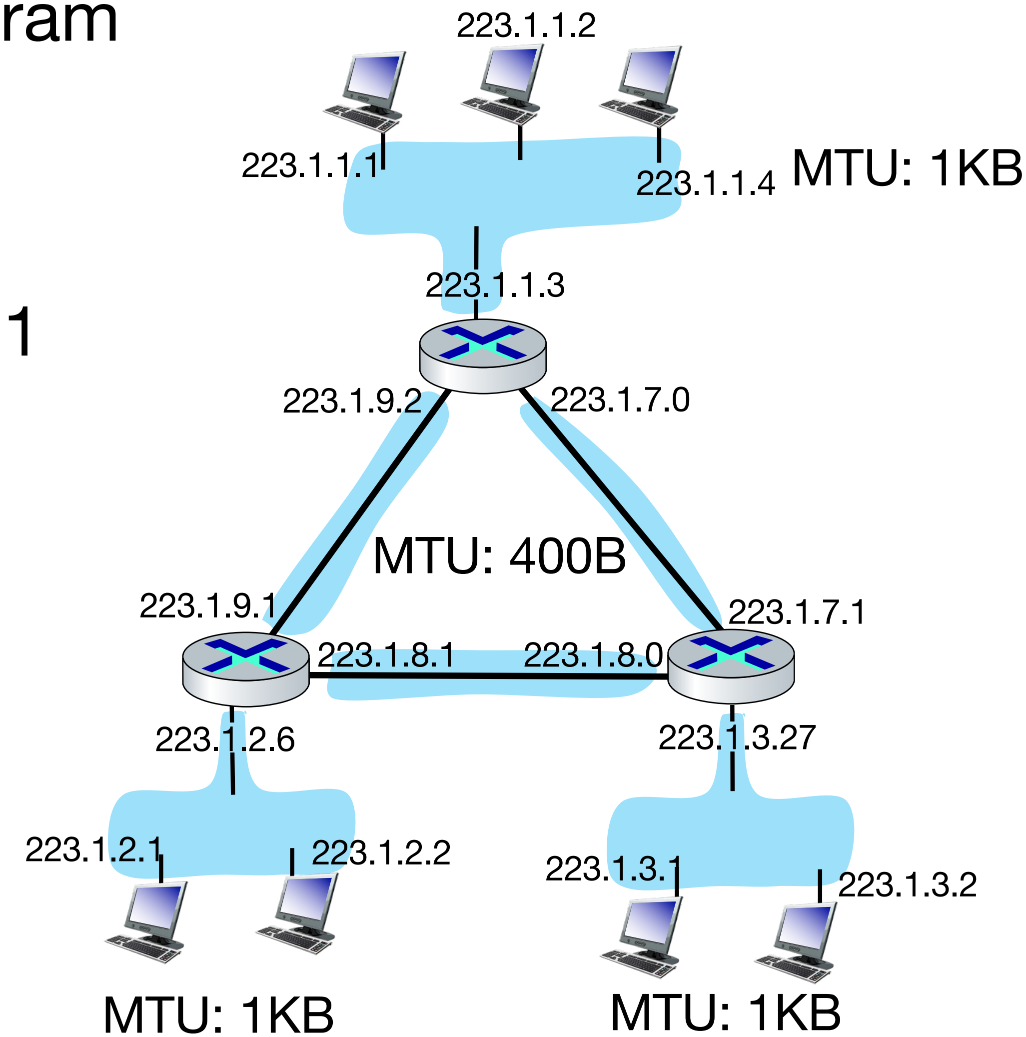
Practice Question 1: subnets

- Where are the subnets?
- What are the /24 subnet addresses?
- How many interfaces I can address in subnet 223.1.2.1/24?



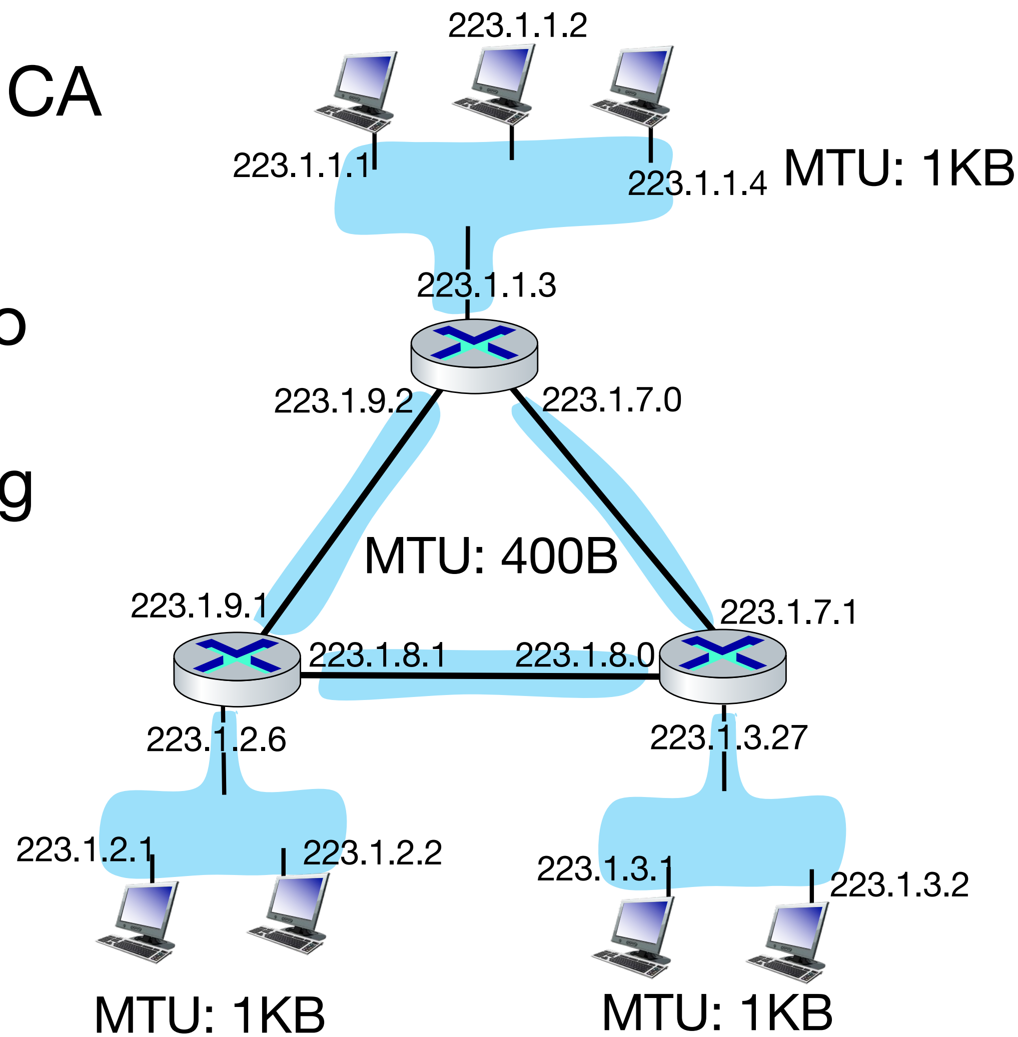
Practice Question 2: fragmentation

- Assuming an IP datagram total size of 1KB
  - 223.1.2.1 to 223.1.1.1
- How many datagram fragments will 223.1.1.1 see?
- Try write the fragment details?



Practice Question 3: coupled

- (Assuming TCP connection is in stable CA stage, cwnd = 4, each segment carries 960B payload, always data to tx)
- Consider 4 outstanding TCP segments
  - Src: 223.1.2.1: 80
  - Dst: 223.1.1.1: 1234
- The first segment lost, how many fragments 223.1.1.1 will receive before sender gets the first seg acked?



Practice Question 4

- (Assuming TCP connection is in stable CA stage, cwnd = 4, each segment carries 960B payload, always data to tx)
- Consider 4 outstanding TCP segments
  - Src: 223.1.2.1: 80
  - Dst: 223.1.1.1: 1234
- The first segment lost, how many fragments 223.1.1.1 will receive before sender gets the first seg acked?