Internet Protocols EBU5403
The Network Layer Part 1

Michael Chai (michael.chai@qmul.ac.uk)
Richard Clegg (r.clegg@qmul.ac.uk)
Cunhua Pan (c.pan@qmul.ac.uk)

	Week 1	Week 2	Week 3	Week 4
Group 1	Michael	Cunhua	Michael	Cunhua
Group 2	Richard			
Group 3	Michael	Cunhua	Michael	Cunhua

Structure of course

- Week 1
  - Introduction to IP Networks
  - The Transport layer (part 1)
- Week 2
  - The Transport layer (part II)
  - The Network layer (part I)
  - Class test
- Week 3
  - The Network layer (part II)
  - The Data link layer (part I)
  - Router lab tutorial (assessed lab work after this week)
- Week 4
  - The Data link layer (part II)
  - Network management and security
  - Class test

Network Layer: outline

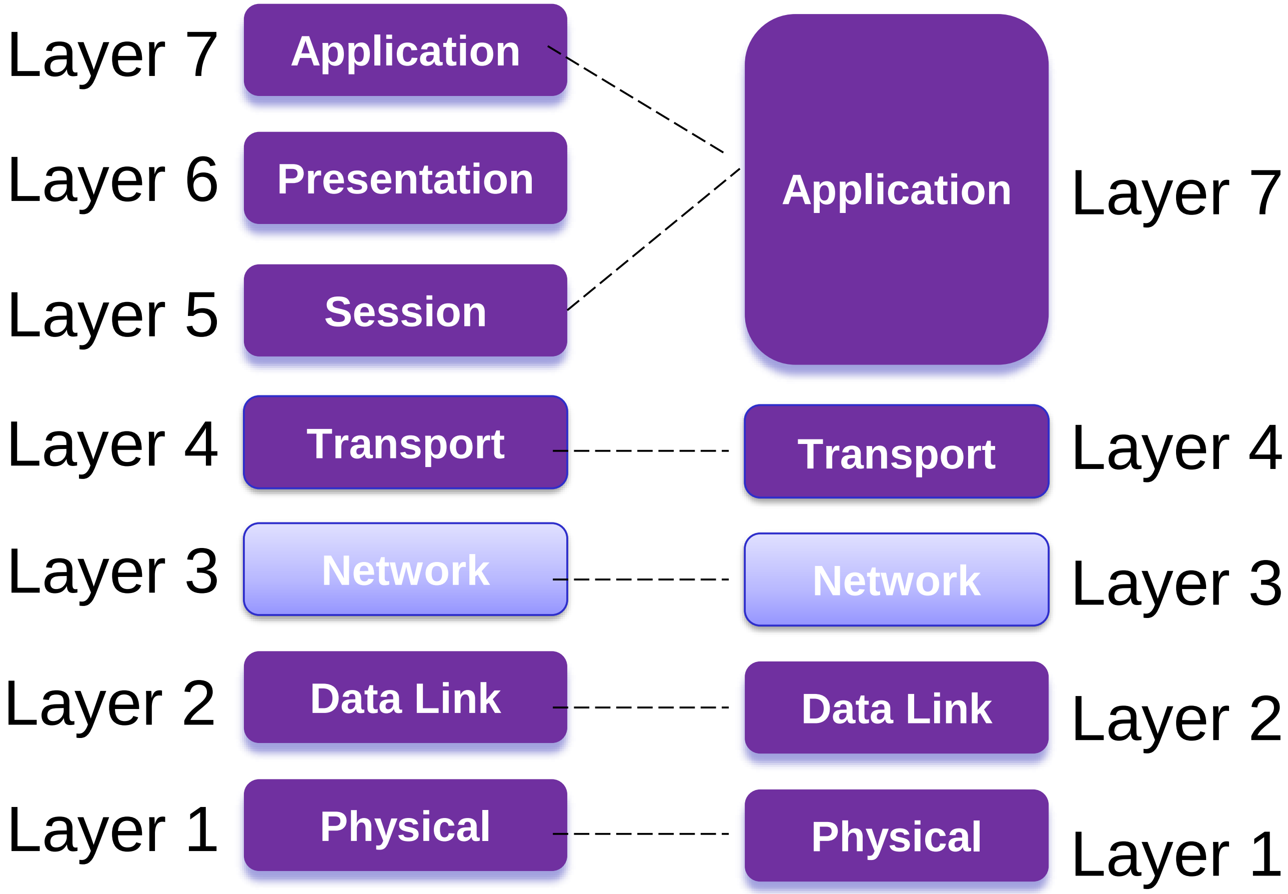
4.1 Overview of Network layer
- data plane
- control plane

4.2 What's inside a router

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

4.4 Generalized Forward and SDN
- match
- action
- OpenFlow examples of match-plus-action in action

Network Layer



Network layer

Goals:

- understand principles behind network layer services, focusing on data plane:
  - network layer service models
  - forwarding versus routing
  - how a router works
  - generalized forwarding
- instantiation, implementation in the Internet

Two key network-layer functions

network-layer functions:

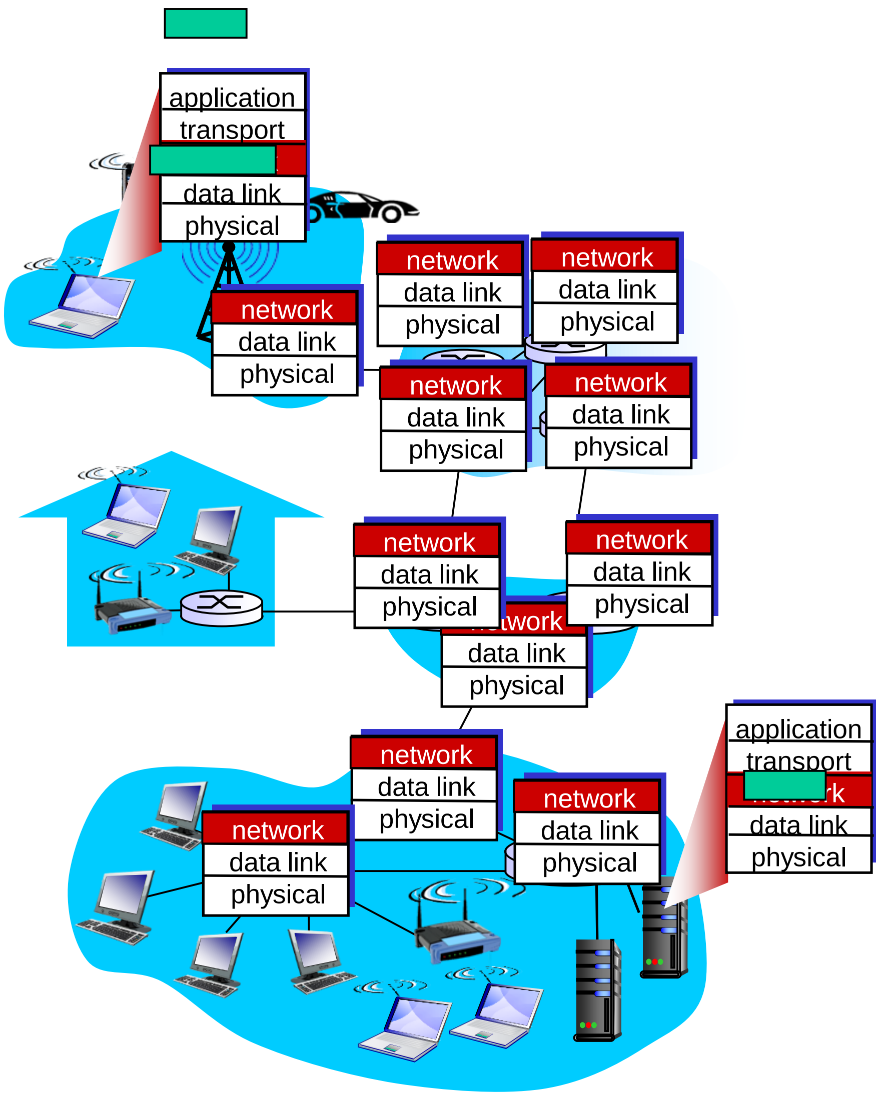
- *forwarding:* move packets from router's input to appropriate router output
- *routing:* determine route taken by packets from source to destination
  - *routing algorithms*

analogy: taking a trip

- *forwarding:* process of getting through one road junction
- *routing:* process of planning trip from source to destination

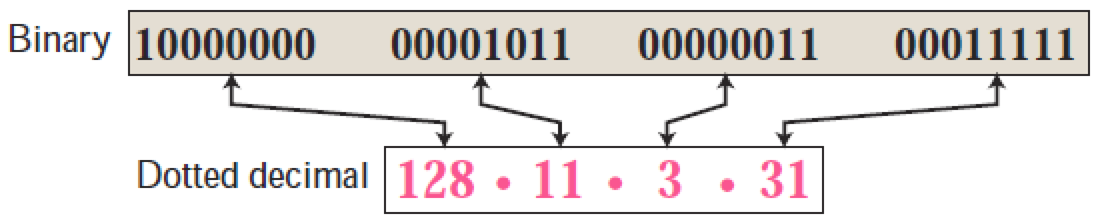
Network layer

- 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



IPv4 address notation

- There are three common notations to show an IPv4 address:
  - binary notation
  - dotted-decimal notation (most commonly used)
  - hexadecimal notation



Exercise: convert the following IPv4 addresses from binary to dotted-decimal notation.

a. 10000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 11100111 11011011 10001011 01101111
d. 11111001 10011011 11111011 00001111

Converting binary to/from decimal (8 bits)

Convert 10100010 to decimal

Factor	128	64	32	16	8	4	2	1
Binary	1	0	1	0	0	0	1	0
Decimal	128	0	32	0	0	0	2	0

Add together 128 + 32 + 2 = 162

Convert decimal to binary – if number is bigger than or equal to factor subtract factor and put 1 in binary column. Example 155

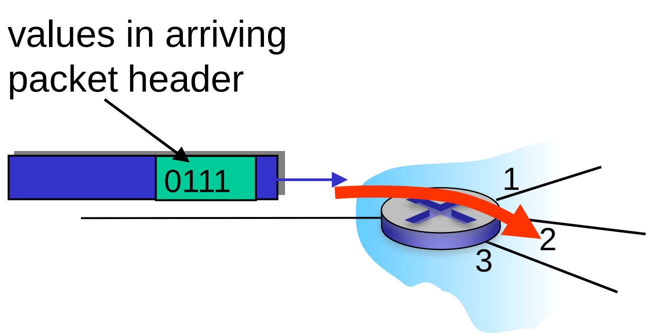
Factor	128	64	32	16	8	4	2	1
Decimal	155-128	27	27	27-16	11-8	3	3-2	1
Binary	1	0	0	1	1	0	1	1

155 in decimal is 10011011

Network layer: data plane, control plane

Data plane

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

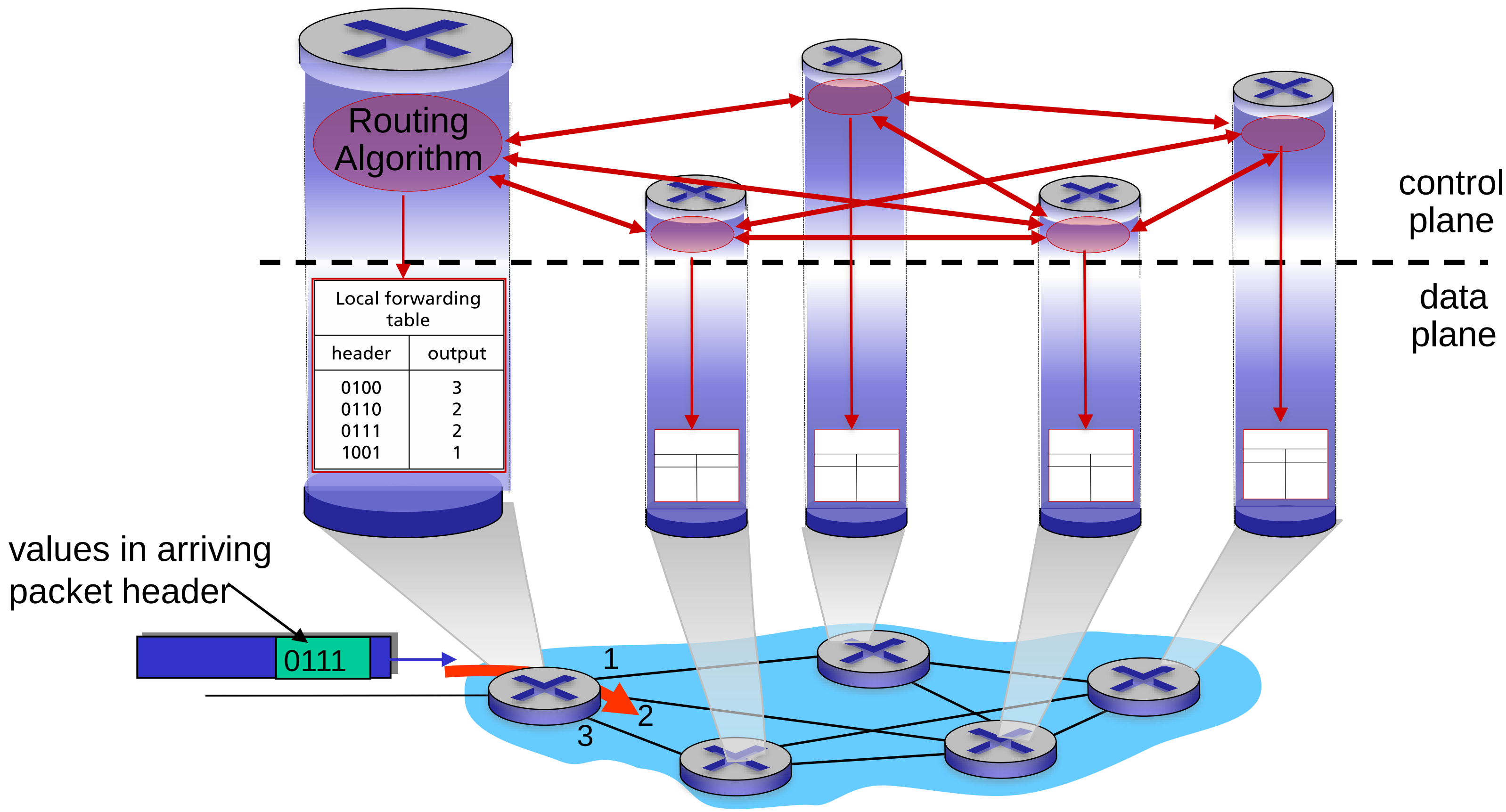


Control plane

- 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)*: implemented in (remote) servers

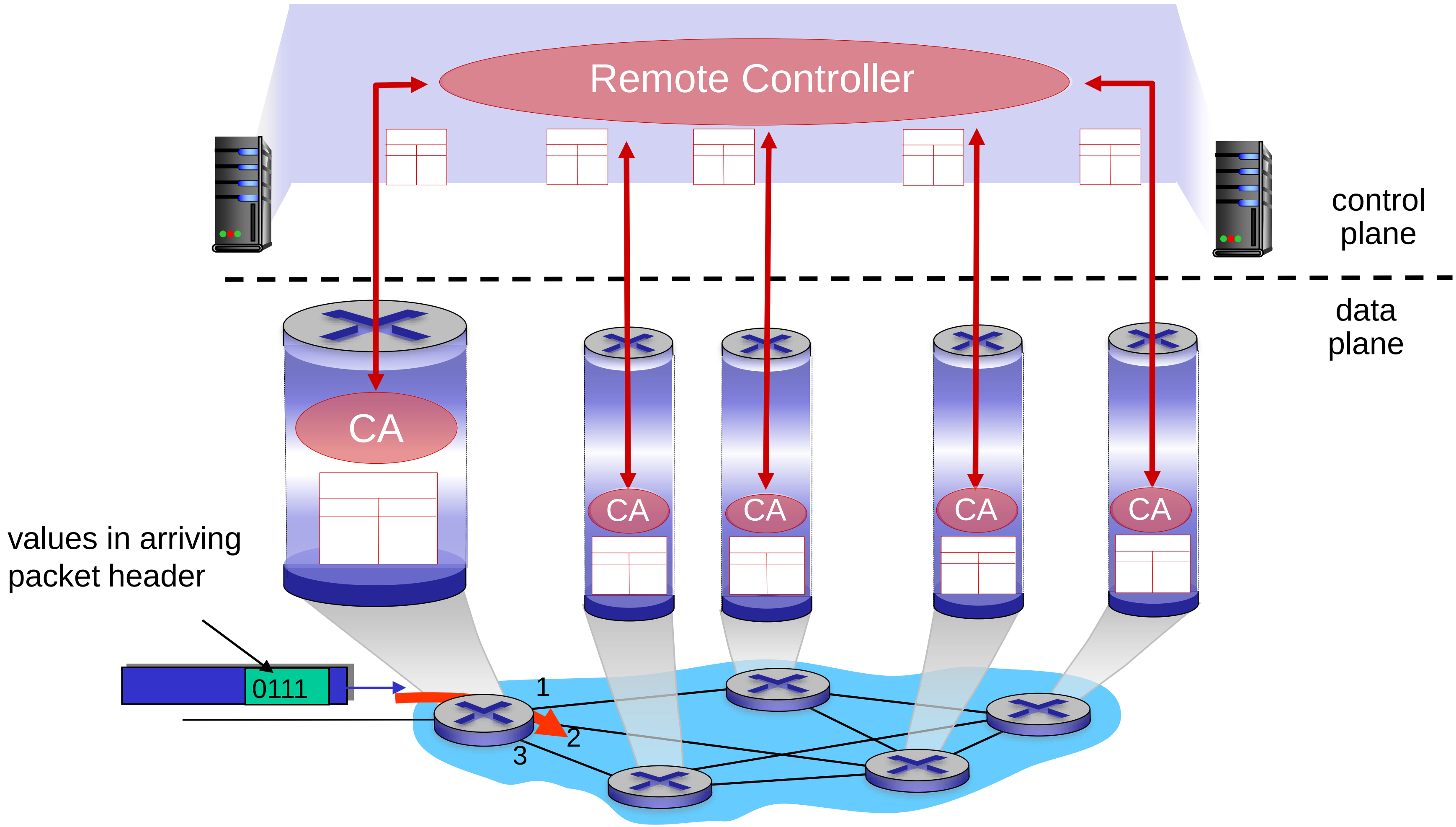
Per-router control plane

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



Logically centralized control plane

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



Network service model

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

example services for individual datagrams:

- guaranteed delivery
- guaranteed delivery with less than 40 msec delay

example services for a flow of datagrams:

- in-order datagram delivery
- guaranteed minimum bandwidth to flow
- restrictions on changes in inter-packet spacing

Network layer service models:

Network Architecture	Service Model	Guarantees ? Bandwidth	Loss	Order	Timing	Congestion feedback
Internet	best effort	none	no	no	no	no (inferred via loss)
ATM	CBR	constant rate	yes	yes	yes	no congestion
ATM	VBR	guaranteed rate	yes	yes	yes	no congestion
ATM	ABR	guaranteed minimum	no	yes	no	yes
ATM	UBR	none	no	yes	no	no

Network Layer (Data): outline

4.1 Overview of Network layer
- data plane
- control plane

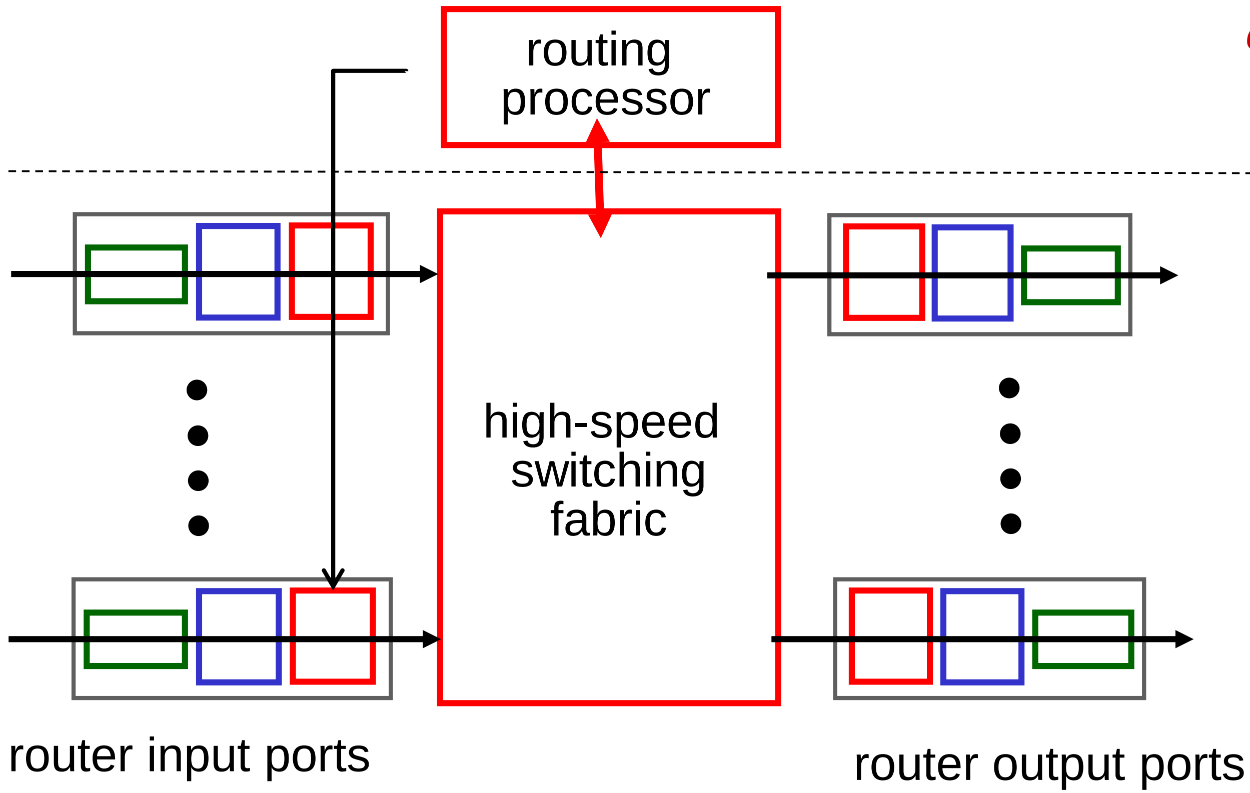
4.2 What's inside a router

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

4.4 Generalized Forward and SDN
- match
- action
- OpenFlow examples of match-plus-action in action

Router architecture overview

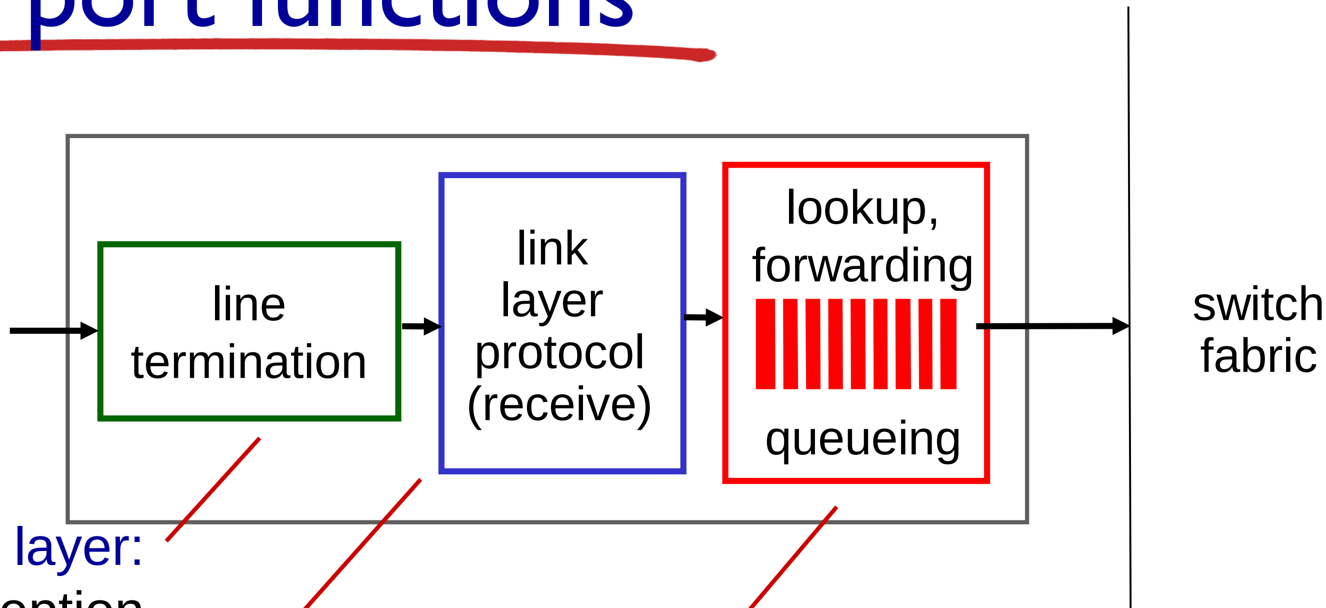
- high-level view of generic router architecture:



routing, management control plane (software) operates in millisecond time frame

forwarding data plane (hardware) operttes in nanosecond timeframe

Input port functions



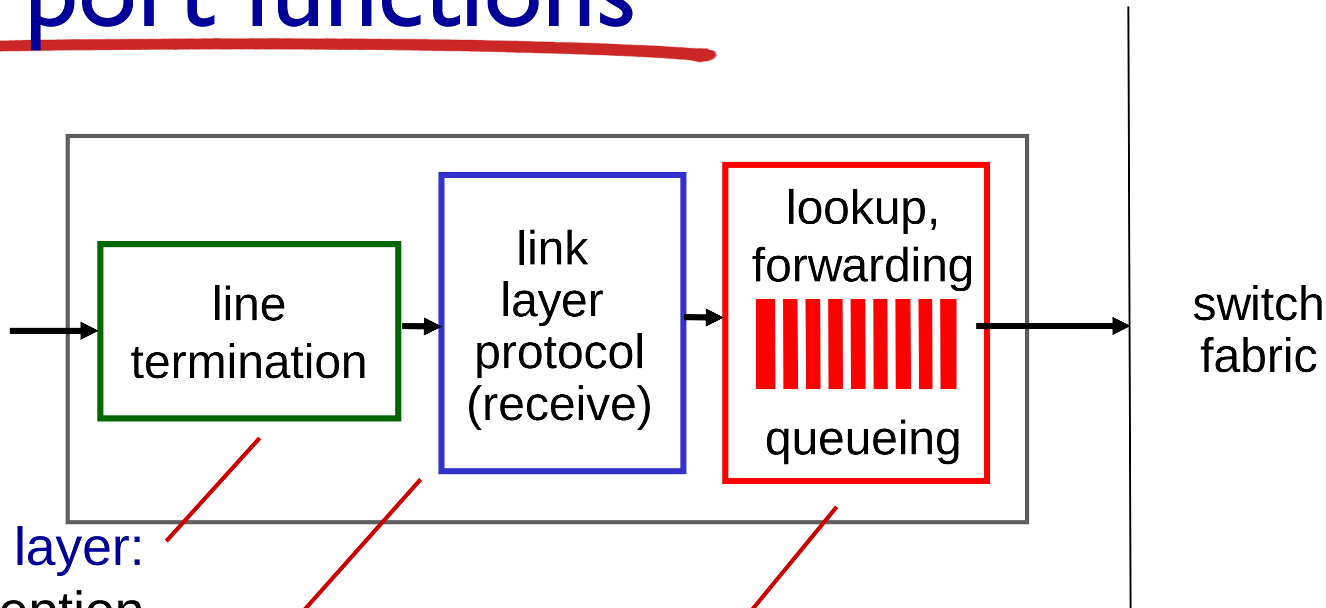
physical layer:
bit-level reception

data link layer:
e.g., Ethernet
see chapter 5
of course text

decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Input port functions



physical layer:
bit-level reception

data link layer:
e.g., Ethernet
see chapter 5
of course text

decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- *destination-based forwarding:* forward based only on destination IP address (traditional)
- *generalized forwarding:* forward based on any set of header field values

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?

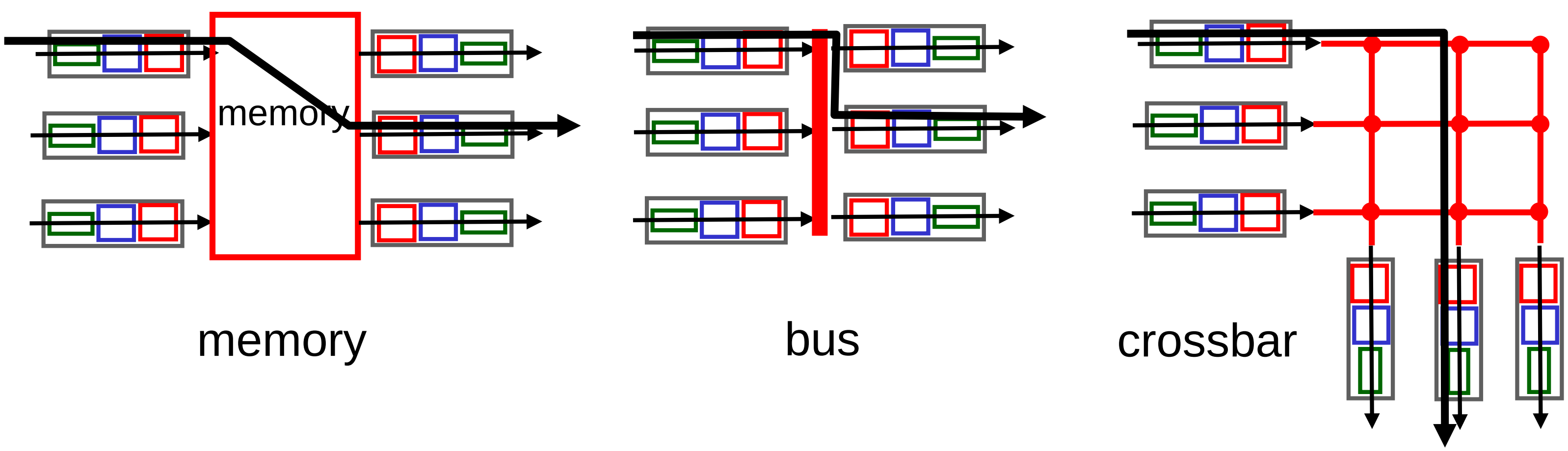
DA: 11001000 00010111 00011000 10101010 which interface?

Longest prefix matching

- we'll see *why* longest prefix matching is used shortly, when we study addressing
- longest prefix matching: often performed using ternary content addressable memories (TCAMs) specialised very high speed memory
  - *content addressable:* present address to TCAM: retrieve address in one clock cycle, regardless of table size
  - Cisco Catalyst: can up ~1M routing table entries in TCAM

Switching fabrics

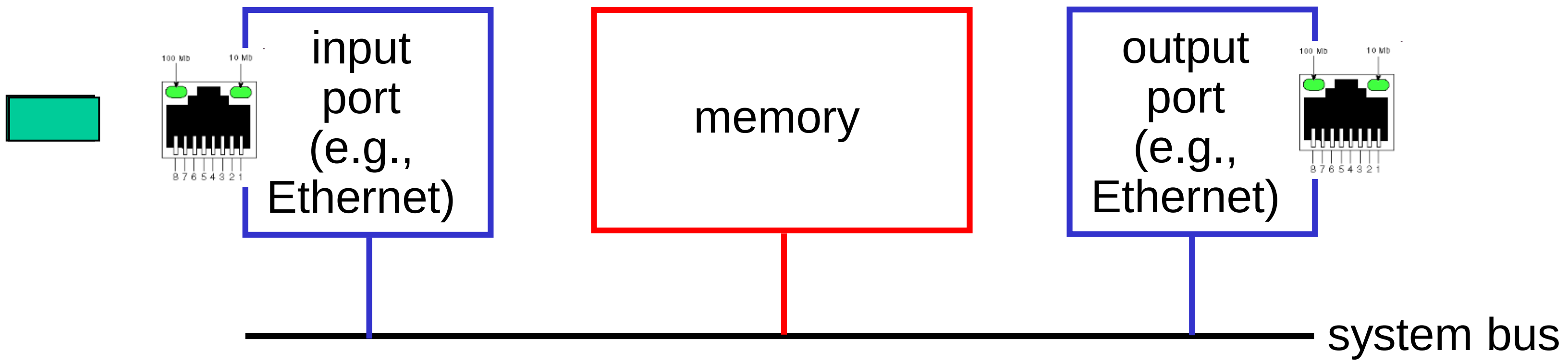
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Switching via memory

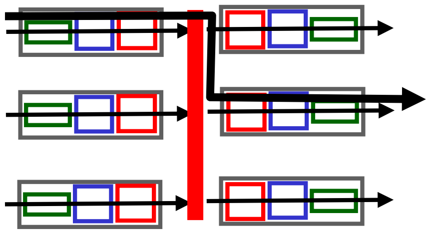
first generation routers:

- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)



Switching via a bus

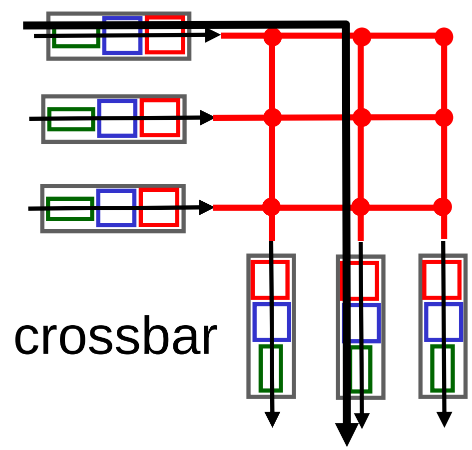
- datagram from input port memory to output port memory via a shared bus
- *bus contention:* switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers



bus

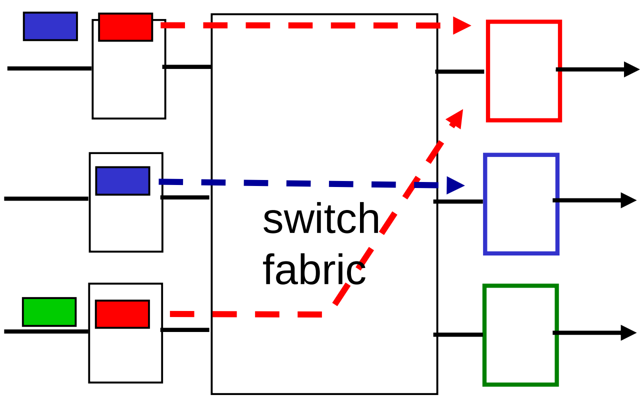
Switching via interconnection network

- overcome bus bandwidth limitations
- banyan networks, crossbar, other interconnection nets initially developed to connect processors in multiprocessor
- advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric.
- Cisco 12000: switches 60 Gbps through the interconnection network

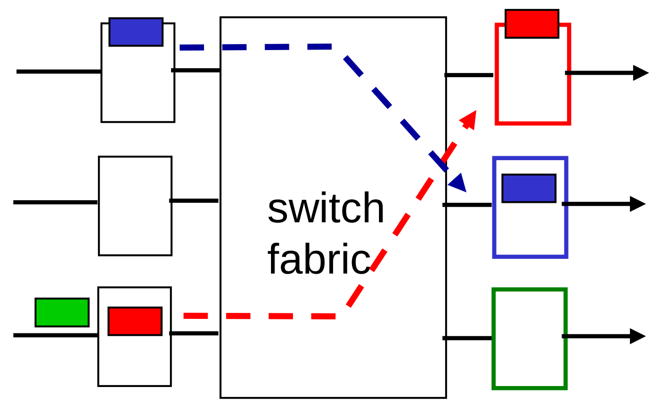


Input port queuing

- fabric slower than input ports combined -> queueing may occur at input queues
  - *queueing delay and loss due to input buffer overflow!*
- Head-of-the-Line (HOL) blocking: queued datagram at front of queue prevents others in queue from moving forward



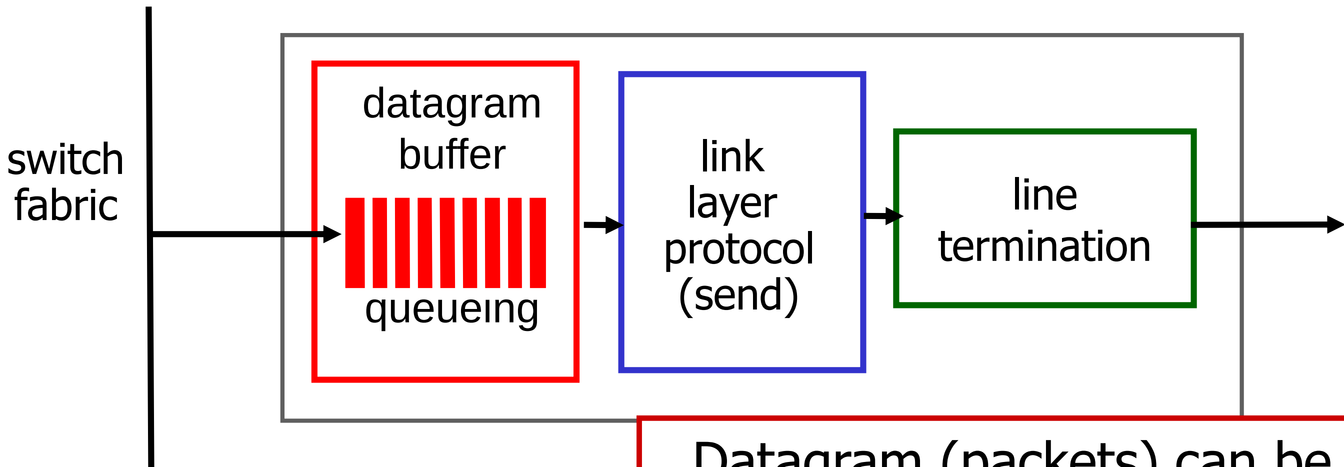
output port contention:
only one red datagram can be transferred.
lower red packet is blocked



one packet time later:
green packet experiences HOL blocking

Output ports

This slide is HUGELY important!

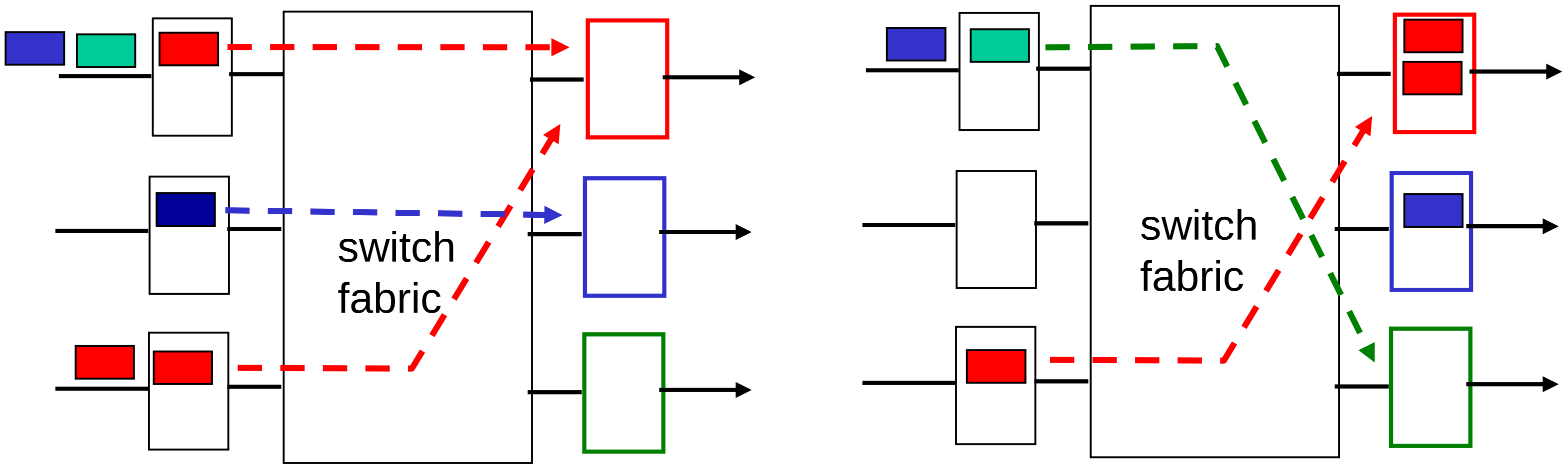


Datagram (packets) can be lost due to congestion, lack of buffers

- *buffering* required when datagrams arrive from fabric faster than the transmission rate
- *scheduling discipline* chooses among queued datagrams for transmission

Priority scheduling – who gets best performance, network neutrality

Output port queueing



at *t*, packets more from input to output

one packet time later

- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

How much buffering?

- RFC 3439 rule of thumb: average buffering equal to "typical" RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gpbs link: 2.5 Gbit buffer
- recent recommendation: with *N* flows, buffering equal to

$$\frac{\mathrm{RTT} \cdot \mathrm{C}}{\sqrt{\mathrm{N}}}$$

- Why? Why not simply have very large buffers so no loss?

Scheduling mechanisms

- *scheduling:* choose next packet to send on link.
- FIFO scheduling – first in first out
  - Like an orderly queue of people, no pushing in.
  - If queue is full last packets are dropped.
- Priority scheduling
  - Some packets are more important
  - Example: You need live video packets now, email could wait.
- Round robin scheduling
  - If your queue is from several inputs ports treat them fairly
  - Pick a packet from input port 1, then 2, then 3, then 4
  - Port which is sending lots of traffic doesn't block others.
  - Weighted Fair Queue (like this but give some queues a little more priority -- give a little more traffic to port 1)

Network Layer: outline

4.1 Overview of Network layer
- data plane
- control plane

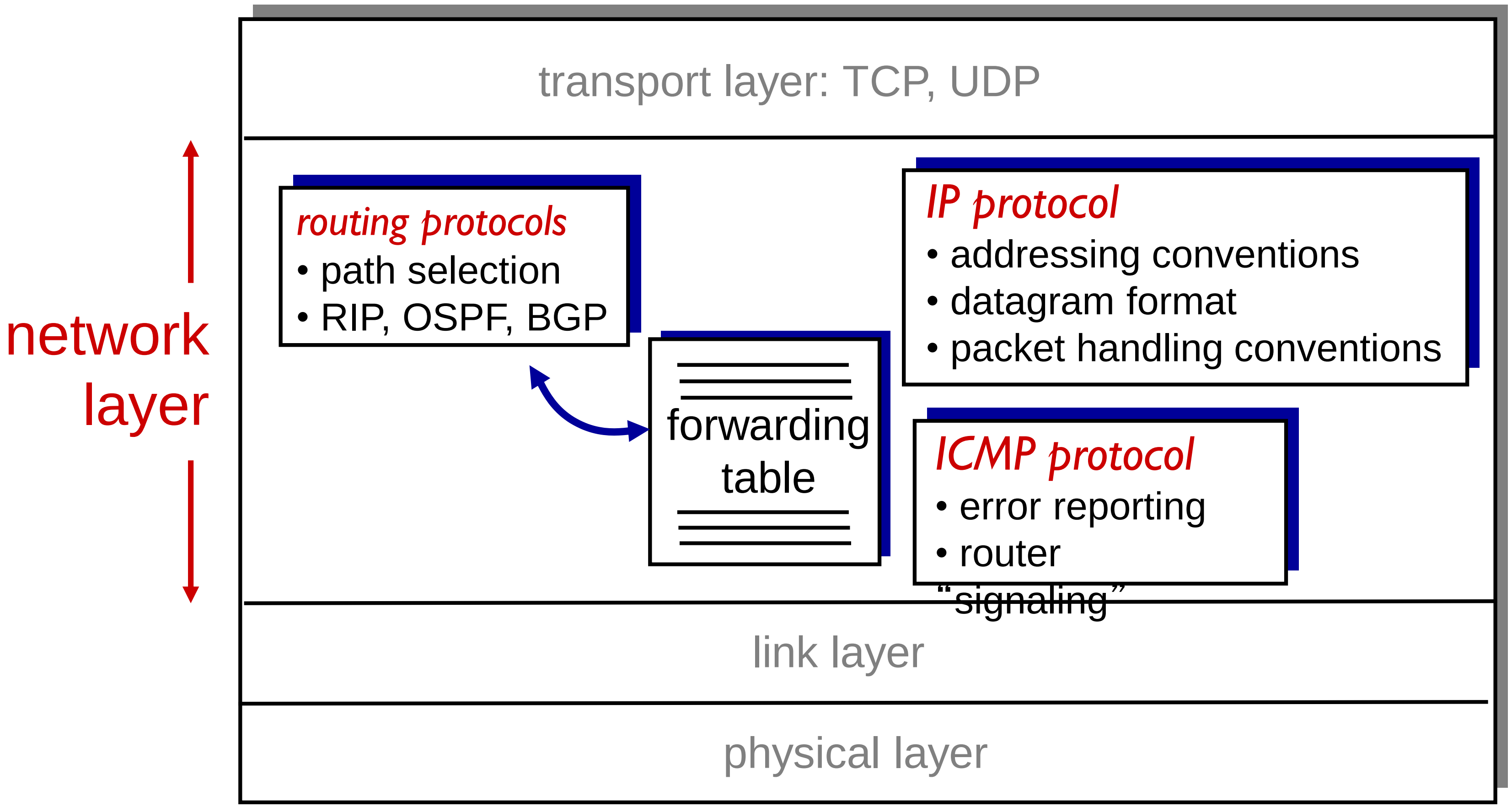
4.2 What' s inside a router

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

4.4 Generalized Forward and SDN
- match
- action
- OpenFlow examples of match-plus-action in action

The Internet network layer

host, router network layer functions:



IP datagram format

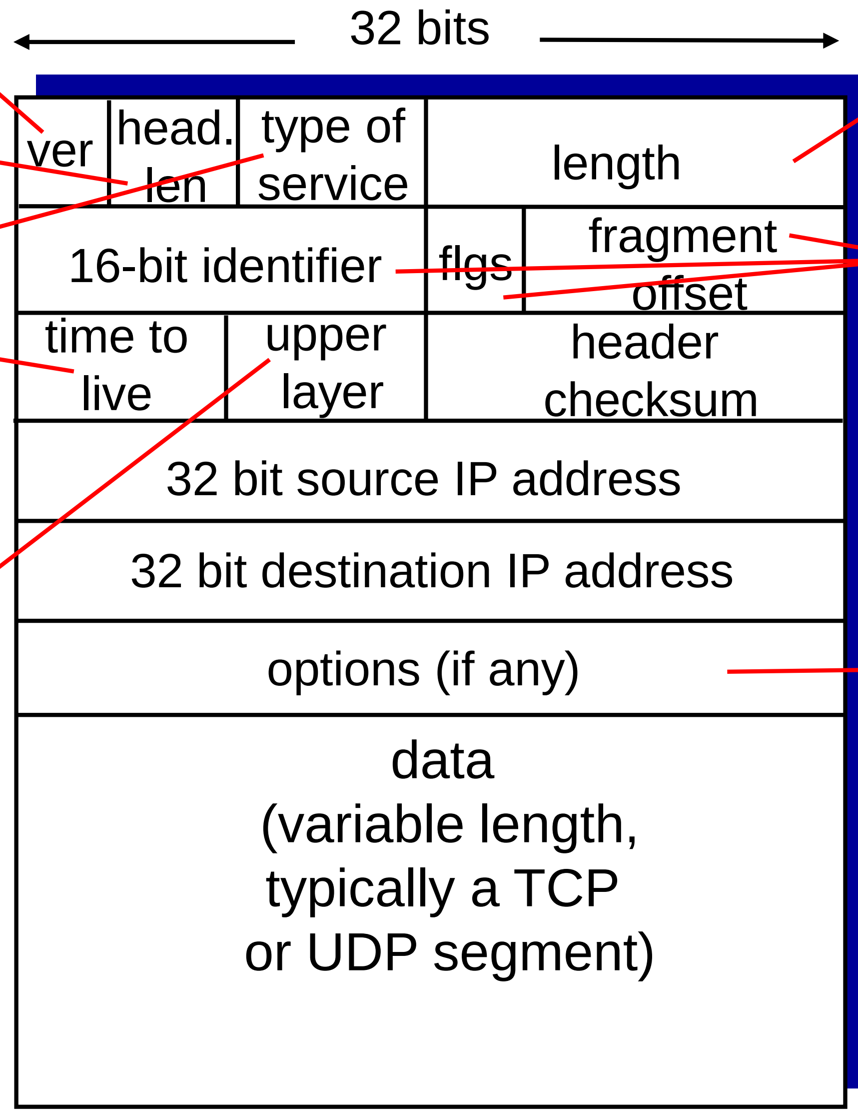
IP protocol version number

header length (bytes)

"type" of data

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to



total datagram length (bytes)

for fragmentation/ reassembly

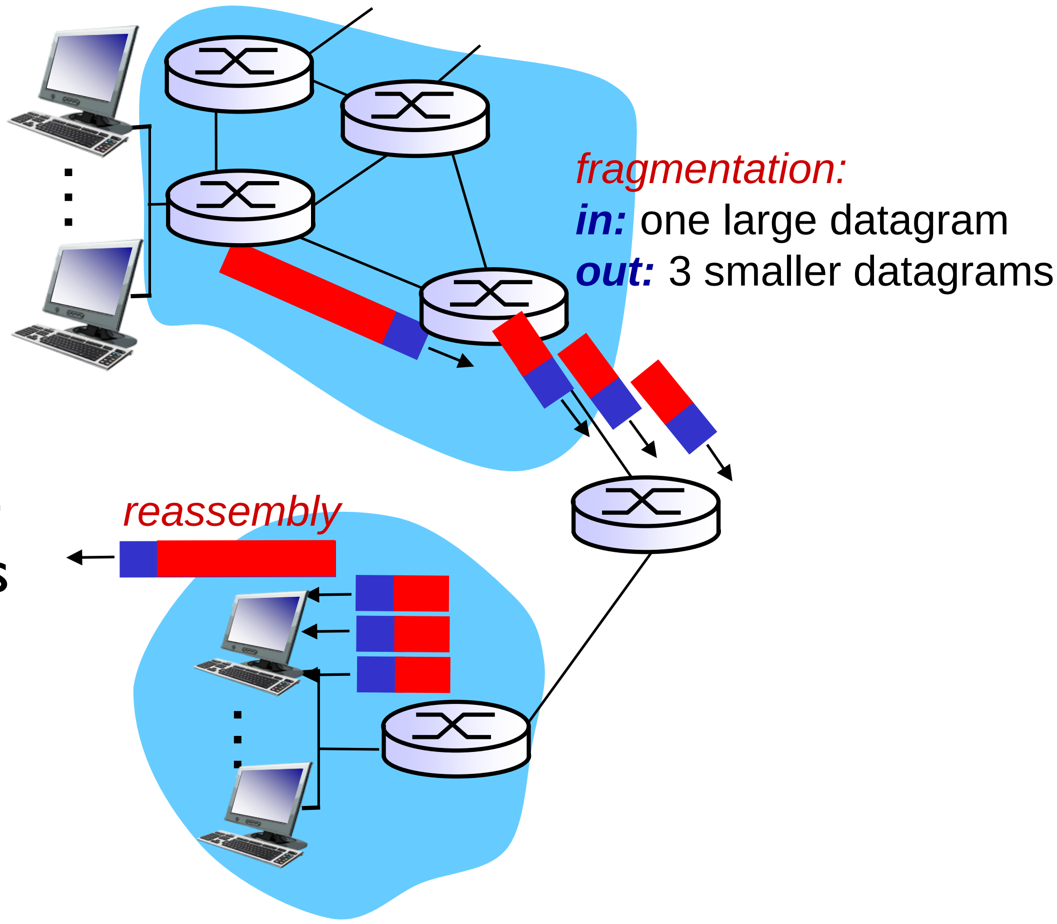
e.g. timestamp, record route taken, specify list of routers to visit.

how much overhead?

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

IP fragmentation, reassembly

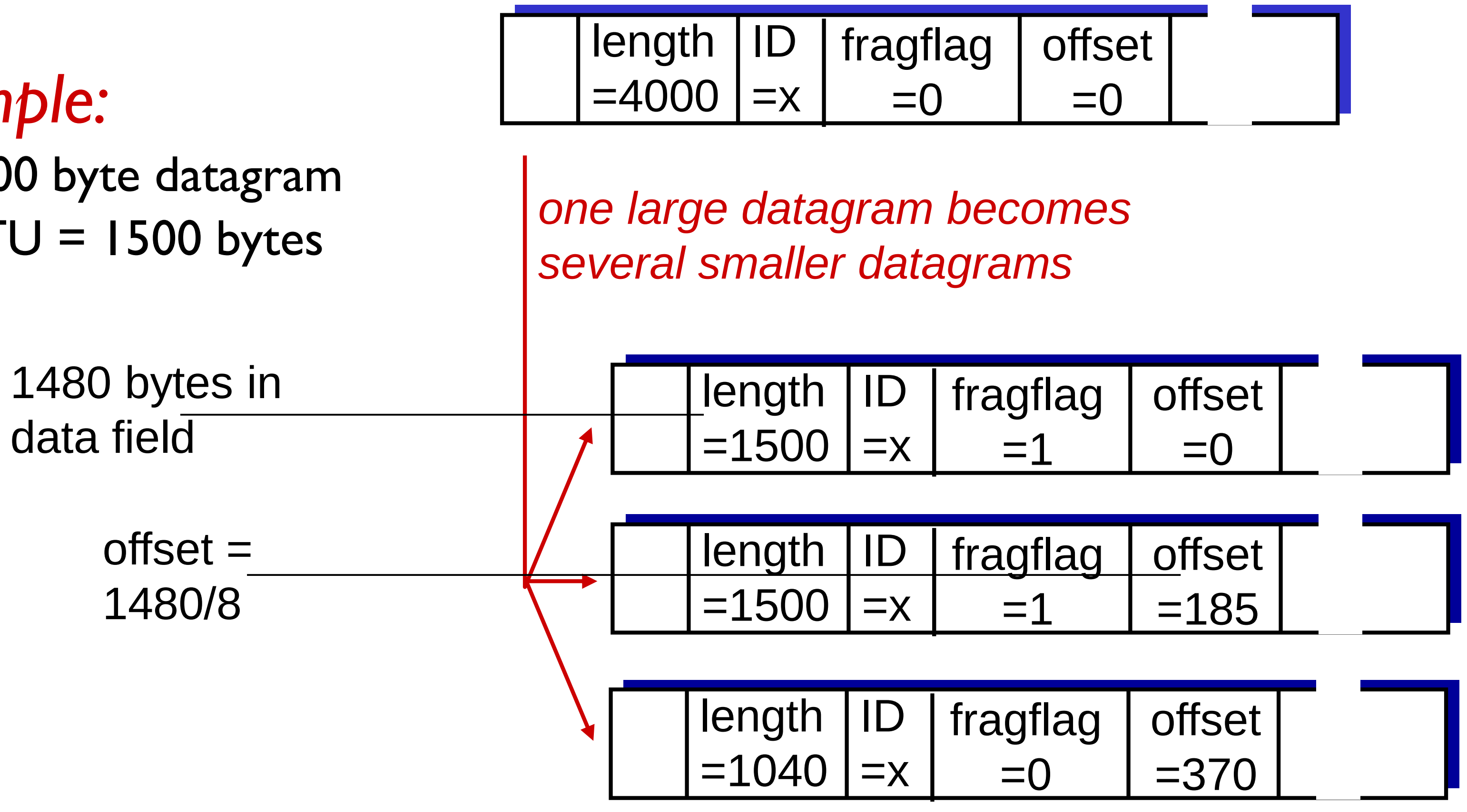
- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order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



IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



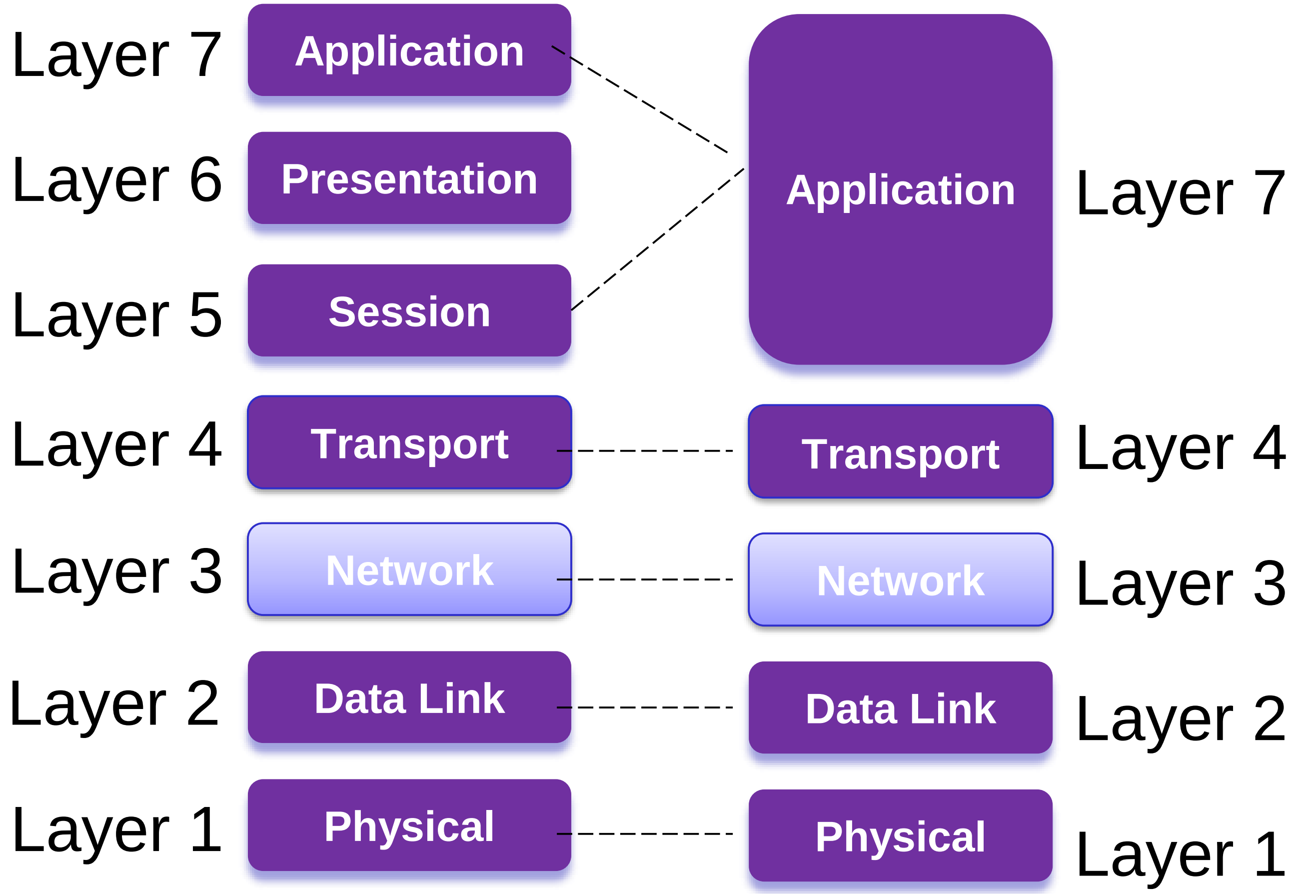
What have we learned?

- Network layer moves data from "source" all the way to the "destination".
- Control plane makes decisions about routes taken by packets it spans a network.
- Data plane implements the decisions and gets the data from place to place. It is local to a router.
- Longest prefix matching on a router is used to pick where to send data.
- Queuing at routers can have huge effects.
- Scheduling polices can benefit some traffic over other traffic.
- Fragmentation and reassembly can be used to split and rejoin IP packets.

Structure of course

- Week 1
  - Introduction to IP Networks
  - The Transport layer (part 1)
- Week 2
  - The Transport layer (part II)
  - The Network layer (part I)
  - Class test
- Week 3
  - The Network layer (part II)
  - The Data link layer (part I)
  - Router lab tutorial (assessed lab work after this week)
- Week 4
  - The Data link layer (part II)
  - Network management and security
  - Class test

Network Layer



Network Layer (Data): outline

4.1 Overview of Network layer
- data plane
- control plane

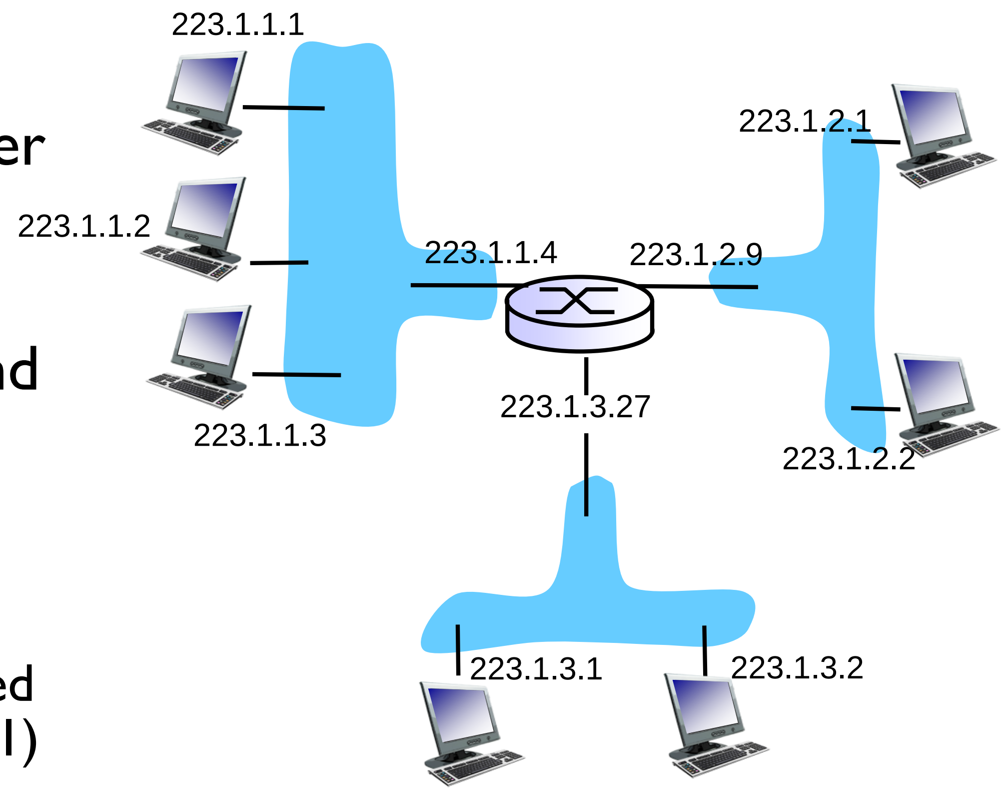
4.2 What's inside a router

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

4.4 Generalized Forward and SDN
- match
- action
- OpenFlow examples of match-plus-action in action

IP addressing: introduction

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

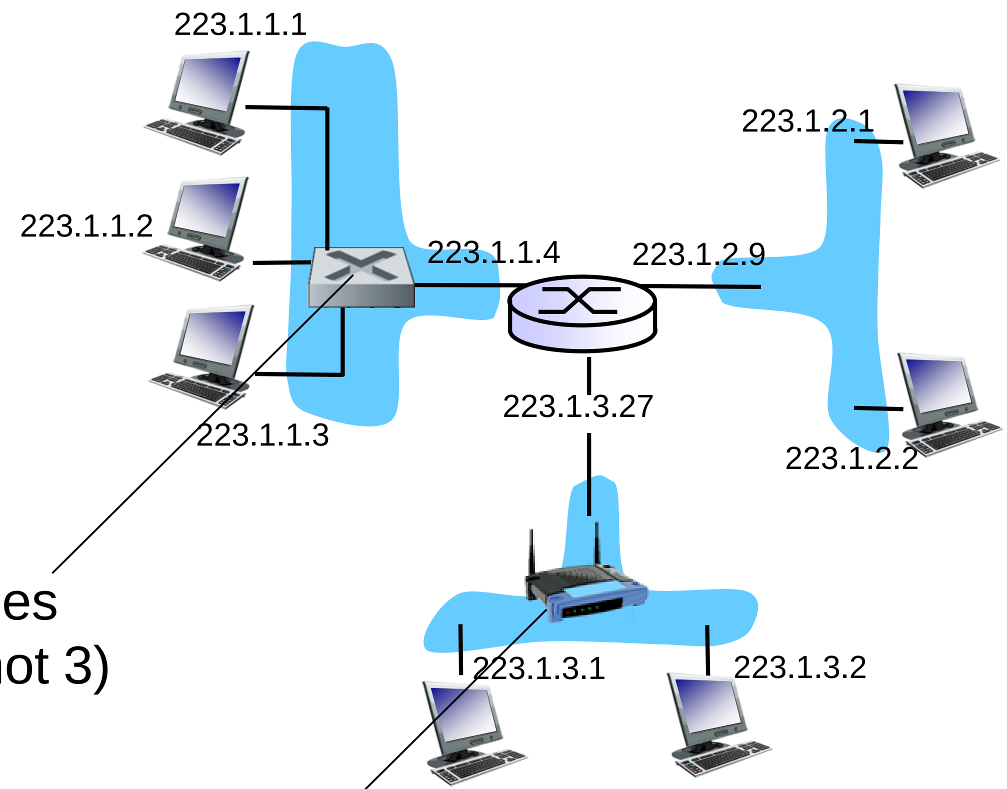


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in later lectures.

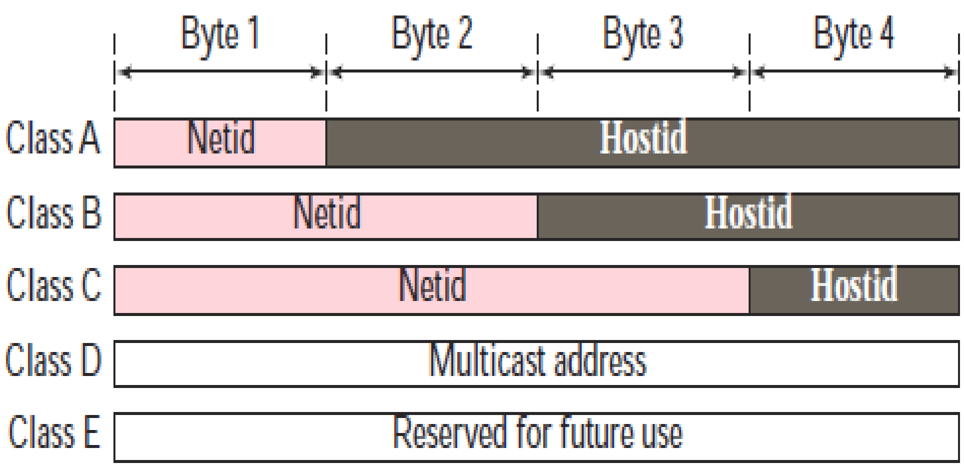


A: wired Ethernet interfaces connected by Ethernet switches (remember switches layer 2 not 3)

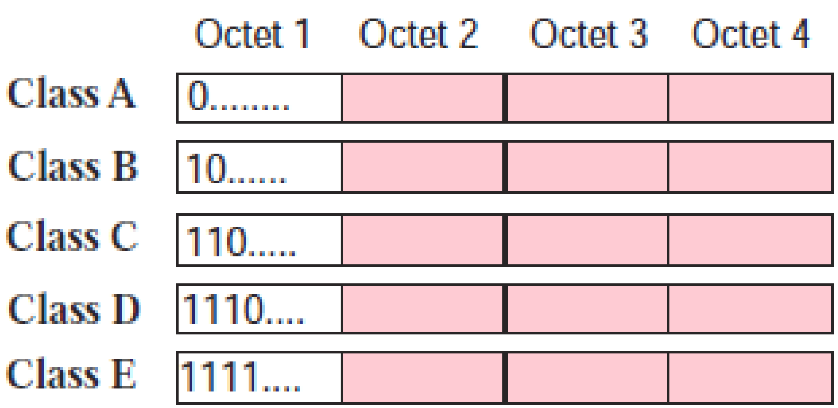
For now: don't need to worry about how one interface is connected to another (with no intervening router)

A: wireless WiFi interfaces connected by WiFi base station

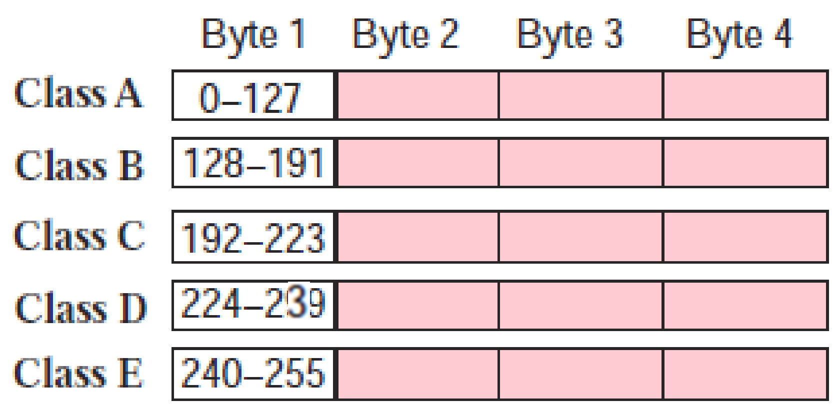
Classful IP addressing



- In classful addressing, the IP address space is divided into five classes: A, B, C, D, E.
- Starting number, n (first byte), shows whether Class A, B or C
  - **Class A**: n < 128 (up to 16m hosts)
  - **Class B**: 128 ≤ n < 192 (up to 65K hosts)
  - **Class C**: 192 ≤ n < 224 (up to 254 hosts)



Binary notation



Dotted-decimal notation

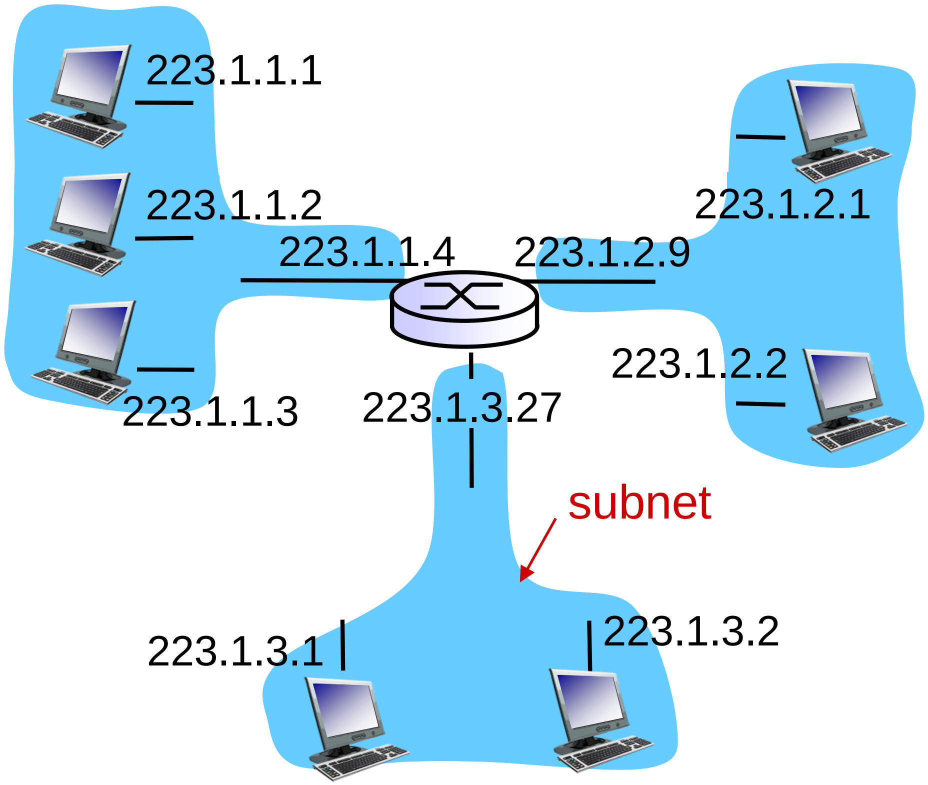
Addresses for private networks

Class	*Netids*	*Blocks*
A	10.0.0	1
B	172.16 to 172.31	16
C	192.168.0 to 192.168.255	256

These addresses are "special" and not used on the general Internet. You use them to set up test networks or networks of machines not accessible from outside.

Subnets

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

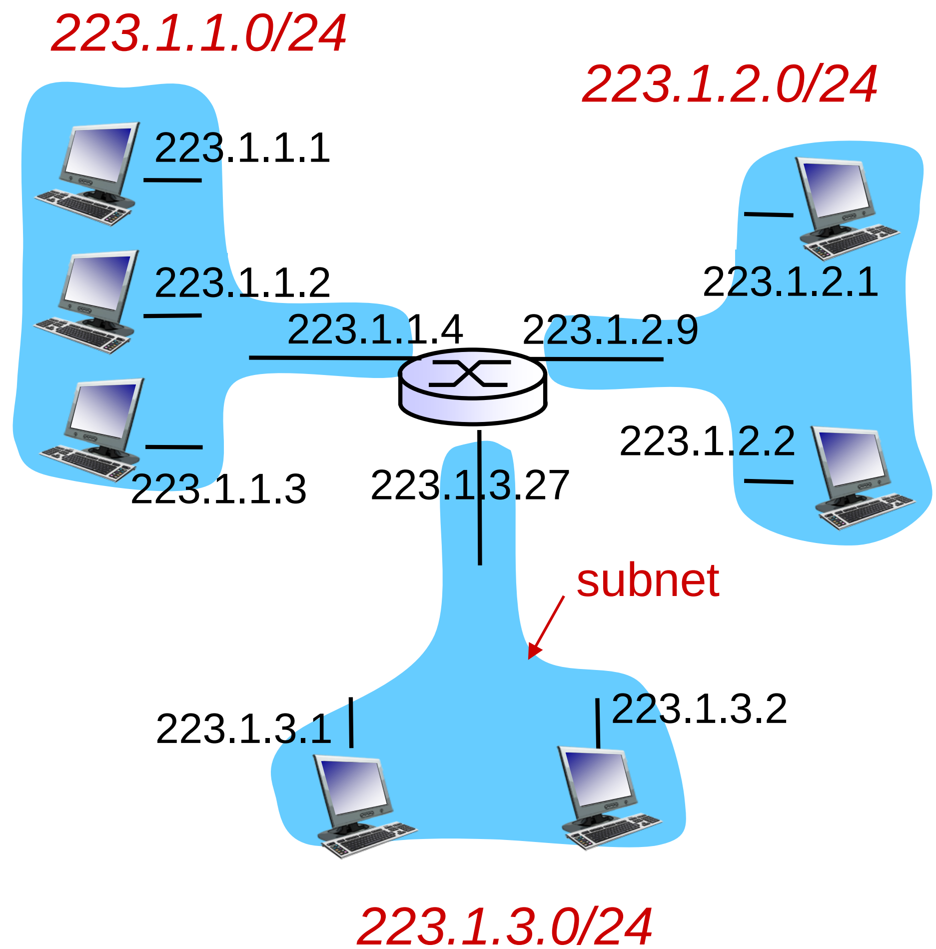


network consisting of 3 subnets

Subnets

recipe

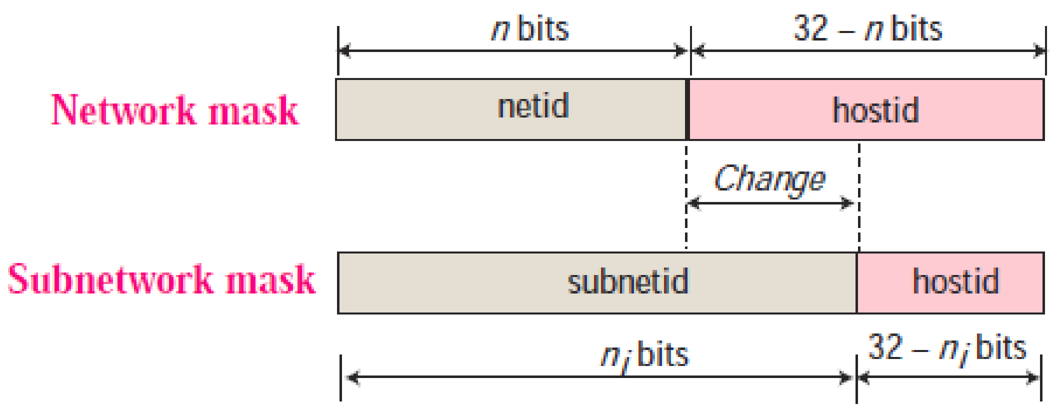
- to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- each isolated network is called a *subnet*
- *subnet mask (or slash notation) number of bits taken to identify network*
  - */8 size of old class A*
  - */16 size of class B*
  - */24 size of class C*



subnet mask: /24

Network mask and subnetwork mask

- Subnetting increases length of **netid** and decreases length of **hostid**



- To divide a network to ***s*** number of subnetworks, each of equal numbers of hosts, the **subnetid** for each subnetwork can be calculated as

$$n_{sub} = n + \log_2 s$$

n **-** length of netid, $\boldsymbol{n_{sub}}$ **-** length of each subnetid, ***s*** – number of subnets

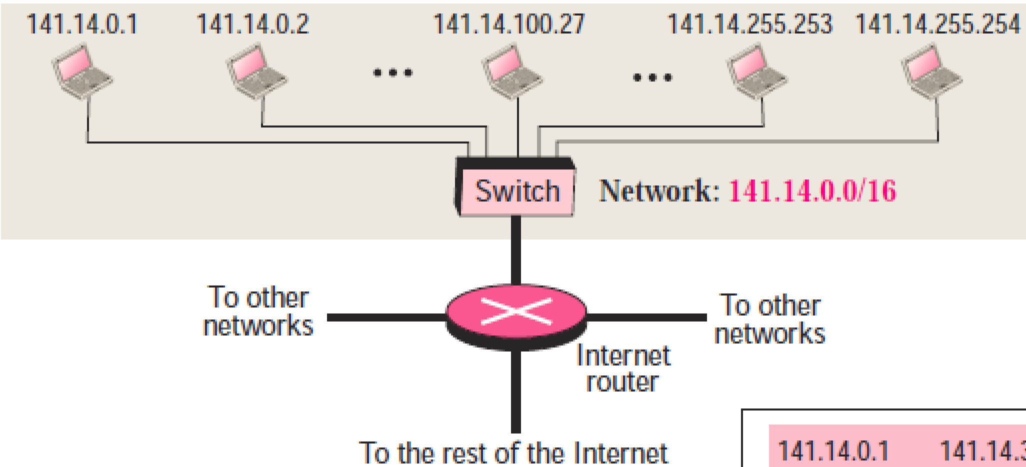
Three-Level Addressing: Subnetting

- The idea of splitting a block to smaller blocks is referred to as subnetting.
- In subnetting, a network is divided into several smaller subnetworks (subnets) with each subnetwork having its own subnetwork address.

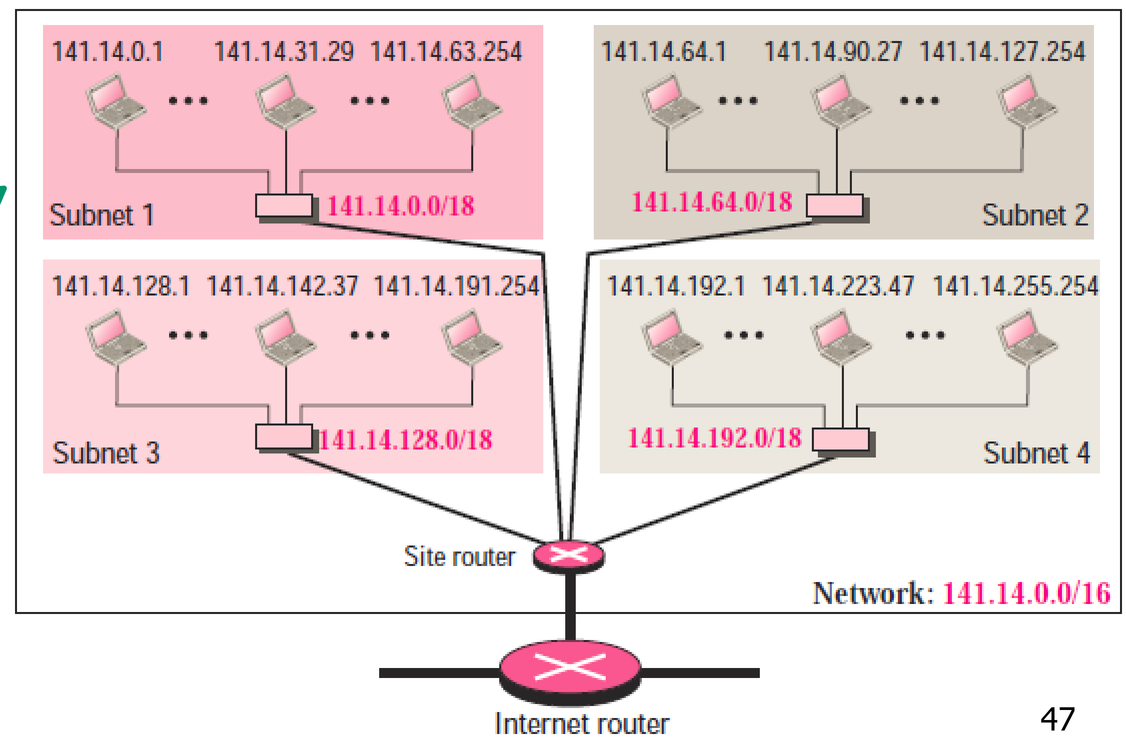
Why subnetting ?

- an organization that was granted a large block of IP addresses (a long time ago this would be class A, class B etc)
- wants to divide this into smaller blocks of addresses that are individual networks.
- Or perhaps organization wants to sell some of its IP addresses off.

A subnetting example



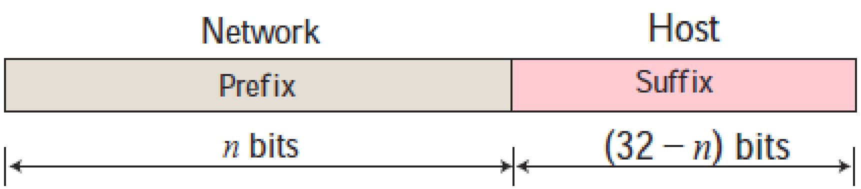
- A network using class B addresses before subnetting
- /16 to show the length of the netid (class B)

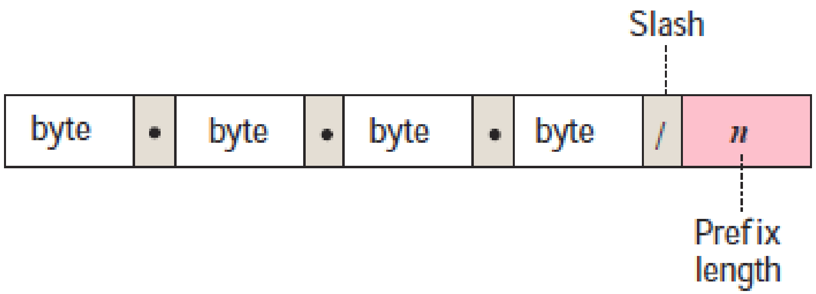


- a private site router is used to divide the network into four subnetworks.
- after subnetting, each subnetwork can now have almost 2^{14} hosts.
- /16 and /18 show the length of the netid and subnetids.

Classless addressing

- In classless addressing, variable-length blocks are assigned that belong to no class.
- In this architecture, the entire address space (2^{32} addresses) is divided into blocks of different sizes.





- The slash notation is formally referred to as classless interdomain routing or CIDR (Classless InterDomain Routing) notation.

Subnets and / notation

- Example 129.66.25.5/22

Network part (22 bits – because it is /22)

Host part 10 bits (32 bits – 22 bits)

`10000001.01000010.00011001.00000101`

Network address is the address with all the host bits set to zero – address like any other but represents the network.

`10000001.01000010.00011000.00000000`

Network address: 129.66.24.0/22

The network address is also the first usable IP address in the block

Broadcast address (sends to everyone) is the address with all the host bits set to one.

`10000001.01000010.00011011.11111111`

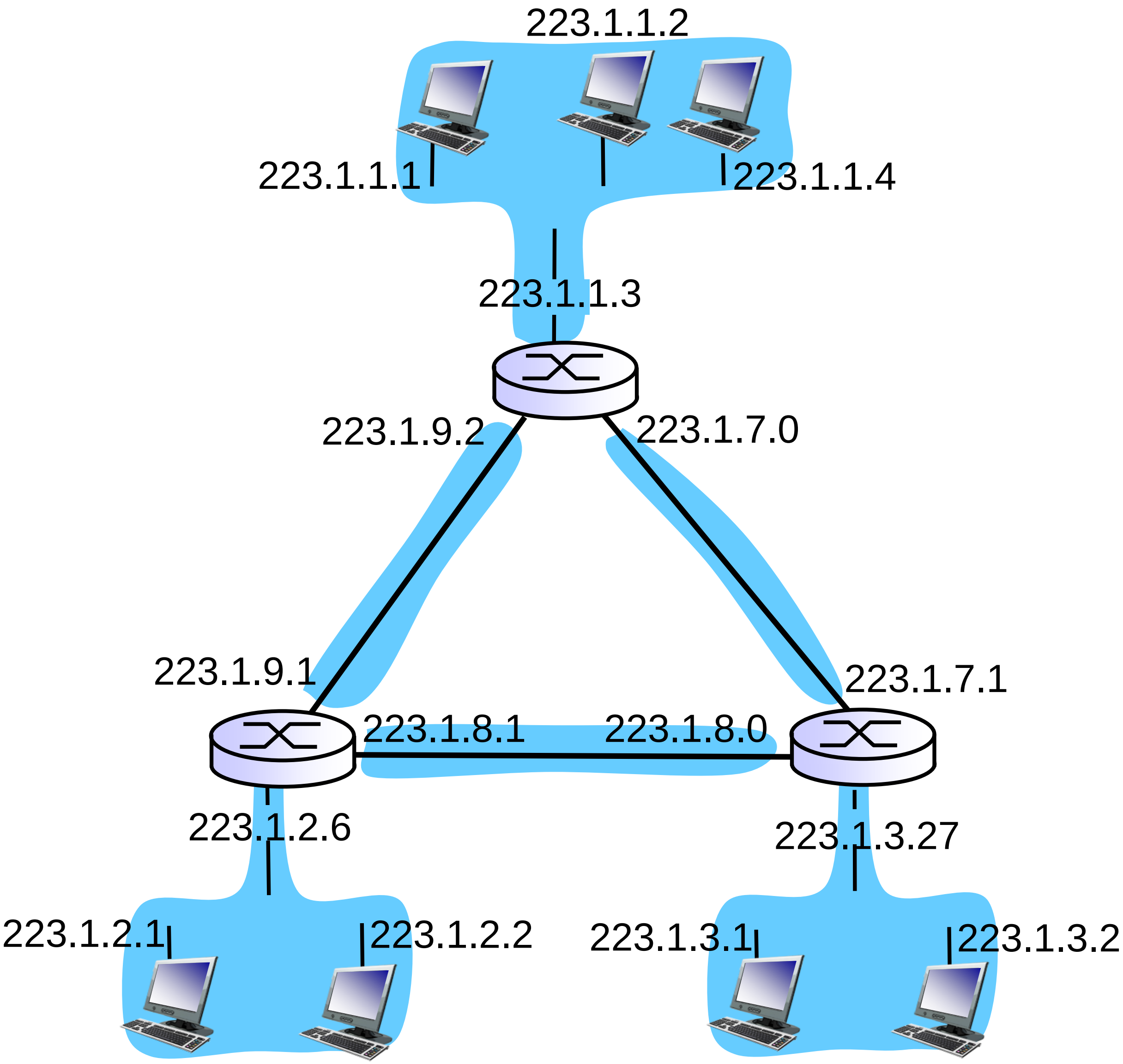
Broadcast address: 129.66.27.255/22

Splitting IP address by slash notation

- For a /n address, the first n bits are the network address and the last 32-n bits are for the host.
- If the host parts are all 1s then this is the broadcast address – sent to all hosts on the network.
- Host has m = 32-n bits (e.g. /20 has m=12)
- Room for 2^m – 1 hosts (e.g. /20 has 4095)
- Example:
  - 140.120.84.24/20
  - Network address is 140.120.80.0/20
  - Broadcast address is 140.120.95.255/20
  - (95 is 01011111 255 is 11111111)
- /31 only has 1 host address
- /30 is smallest subnet we can have – 3 hosts
- /30 commonly used to connect just two routers (which must be on same subnet).

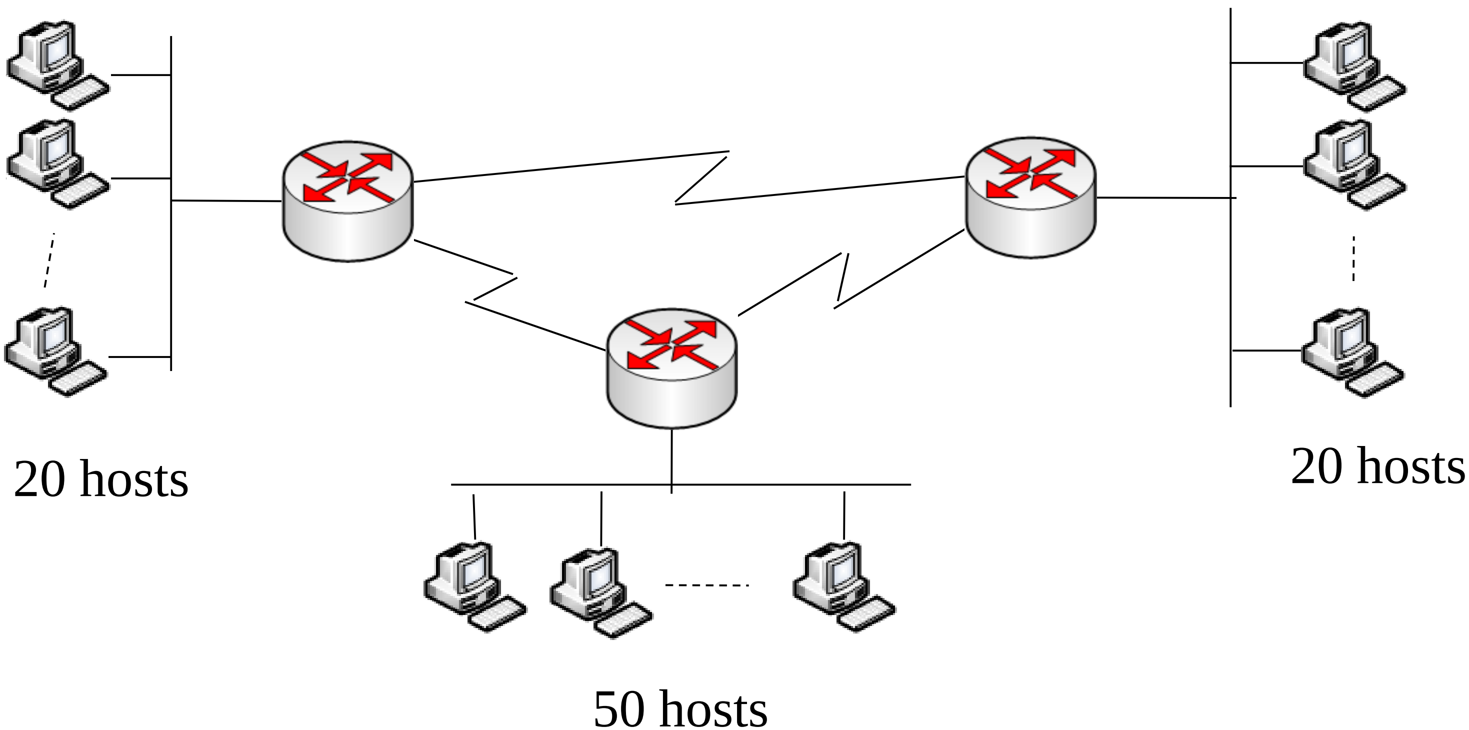
Subnets

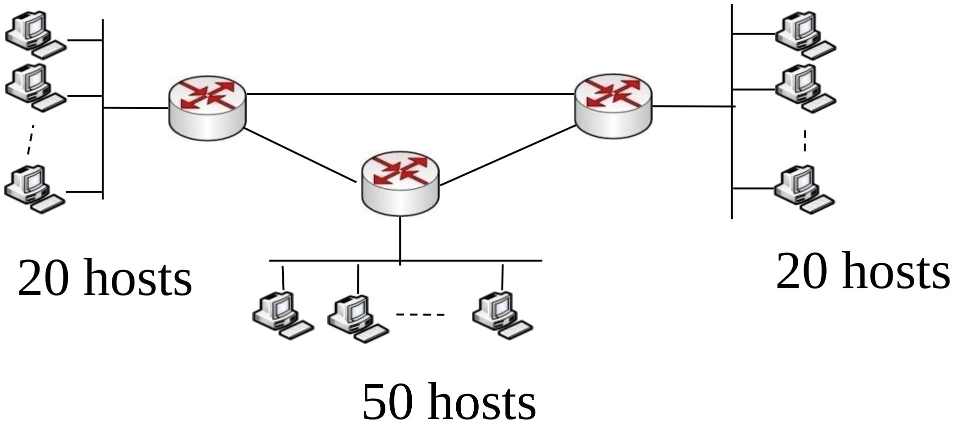
how many?



VLSM (variable length subnet mask)

Subnet 192.168.1.0/24 to address this network by using the most efficient addressing possible.





192.168.1.0/24

Network 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

4.4 Generalized Forward and SDN
- match
- action
- OpenFlow examples of match-plus-action in action

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: Dynamic Host Configuration Protocol

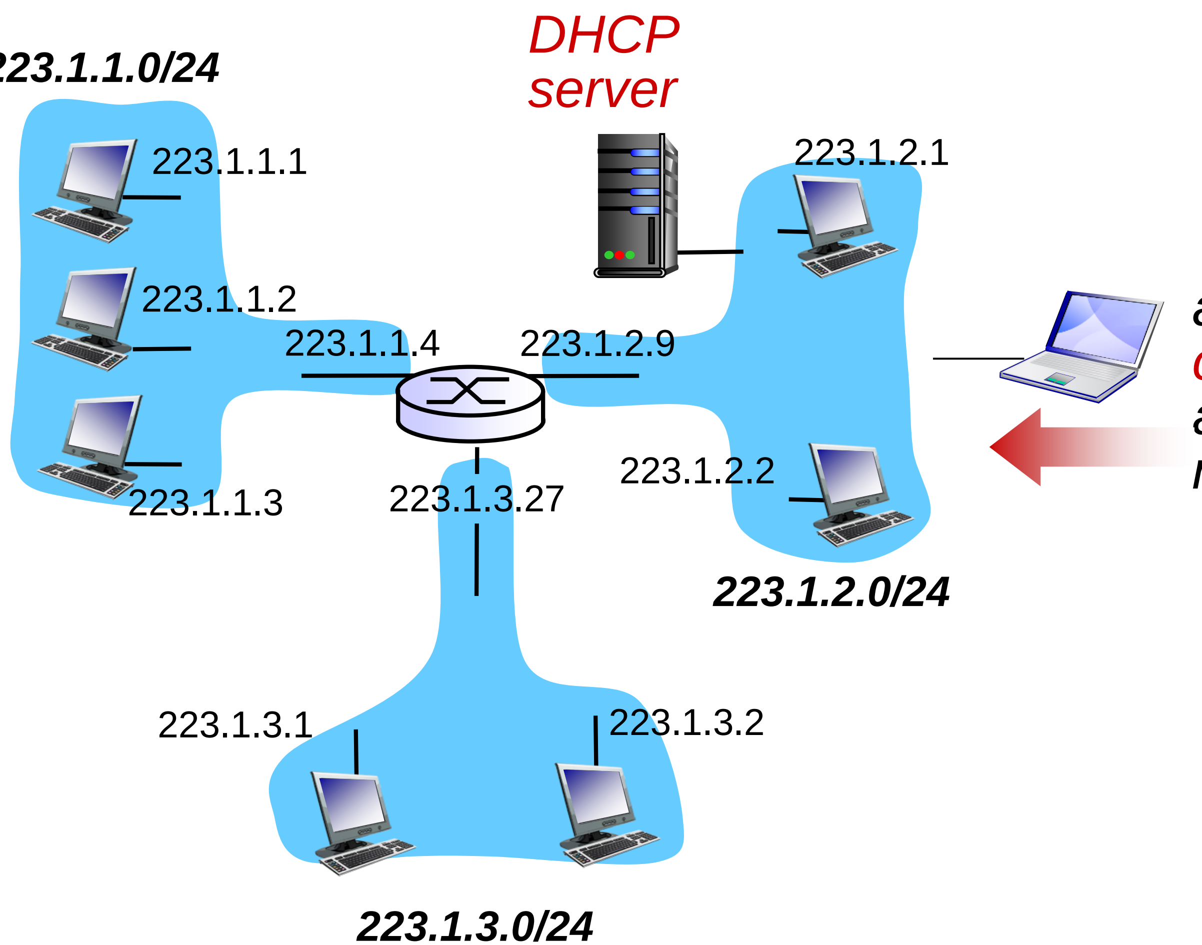
goal: allow host to *dynamically* obtain its IP address from network server when it joins network

- 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 (more shortly)

DHCP overview:

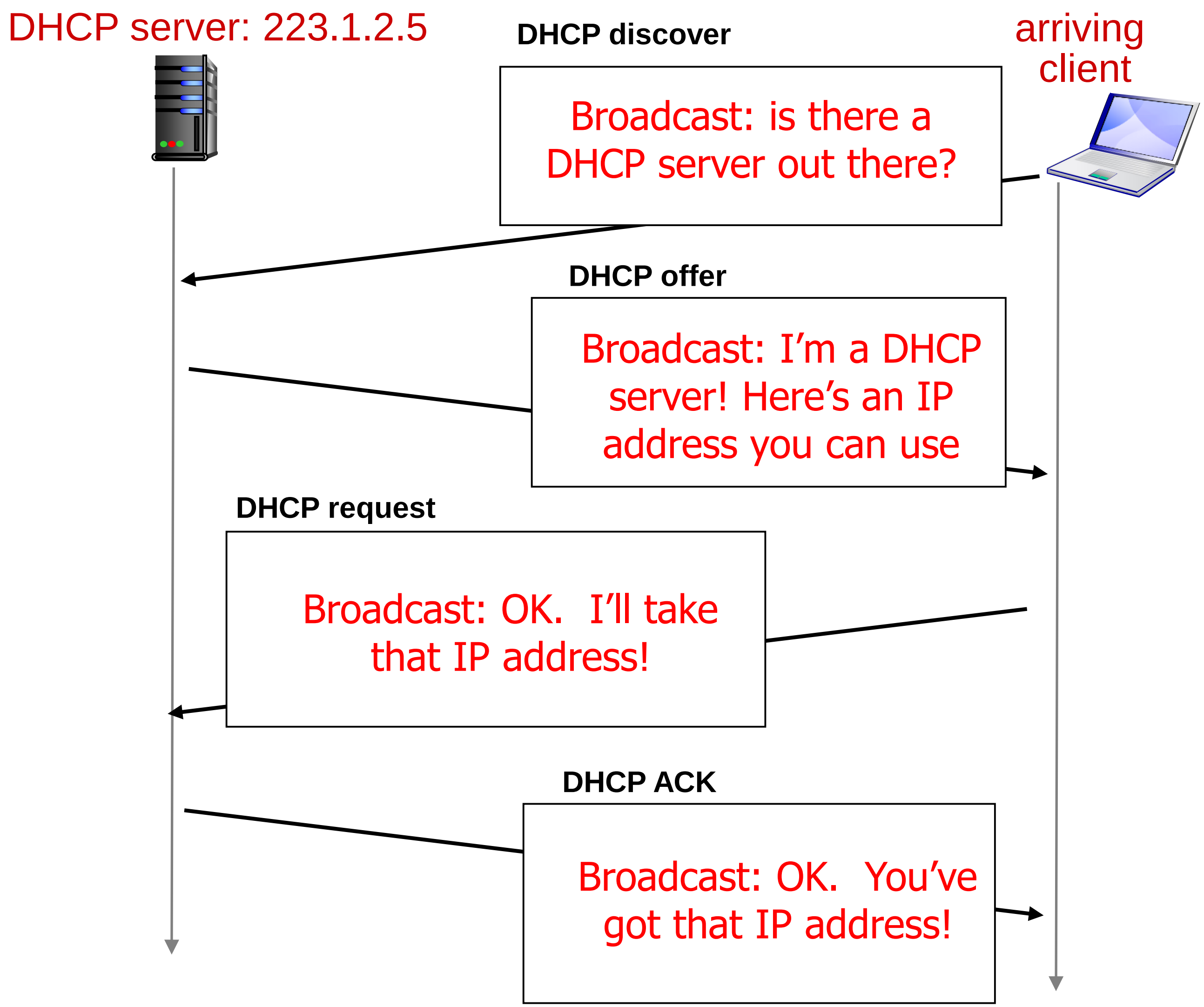
- host broadcasts "DHCP discover" msg [optional]
- DHCP server responds with "DHCP offer" msg [optional]
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

DHCP client-server scenario

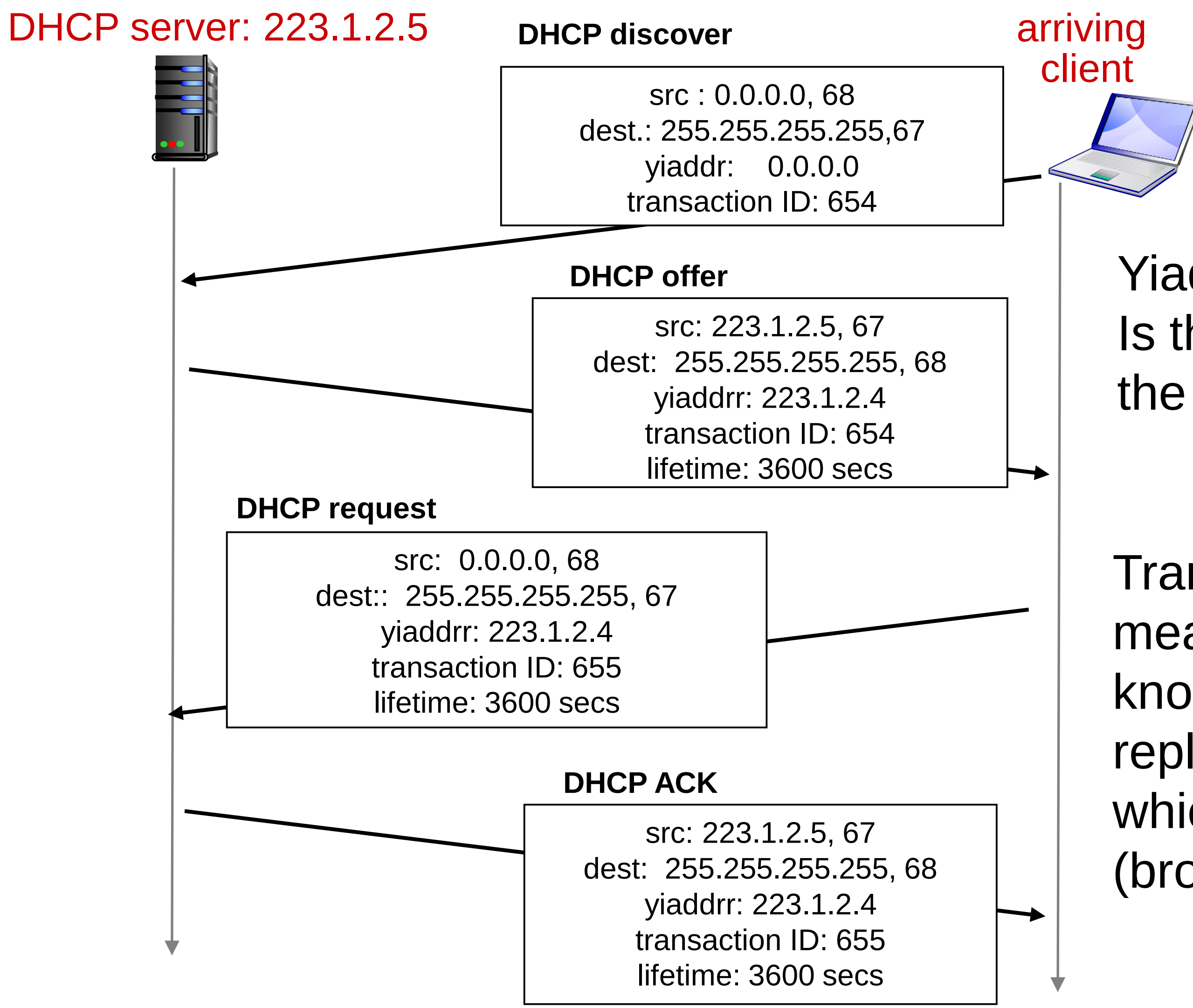


arriving DHCP client needs address in this network

DHCP client-server scenario



DHCP client-server scenario



Yiaddrr
Is the IP address the server offers

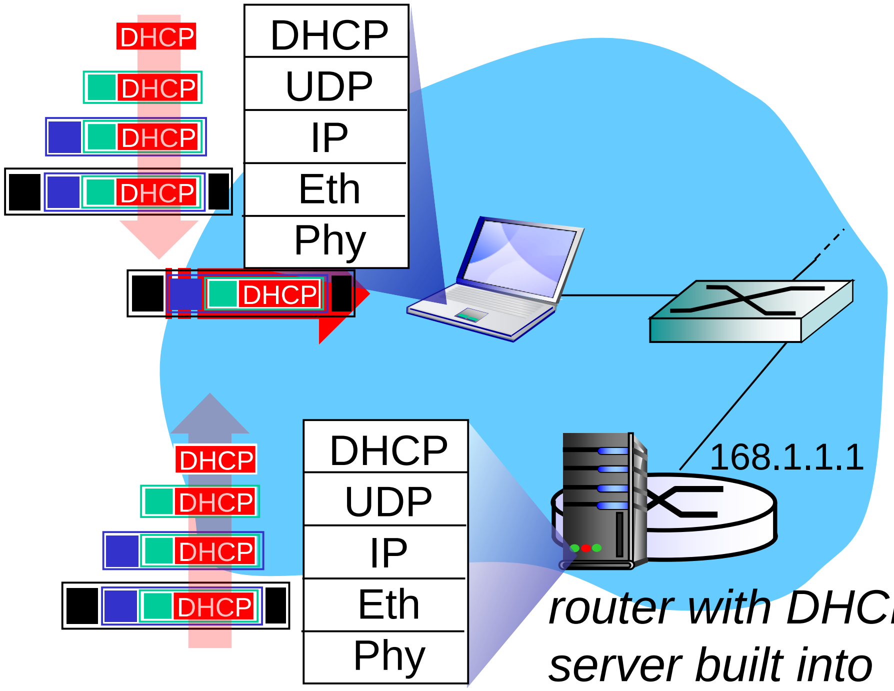
Transaction ID means client knows which reply goes with which request (broadcast)

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)

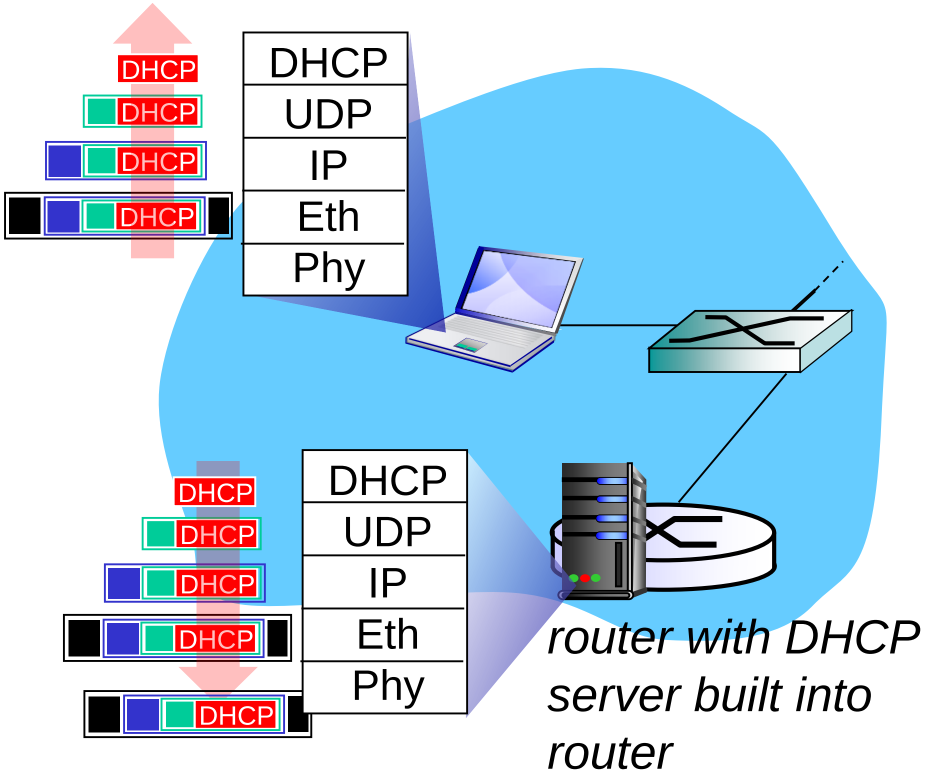
DHCP: example



router with DHCP server built into router

- 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



- DCP 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 DNS 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."

IP addresses: how to get one?

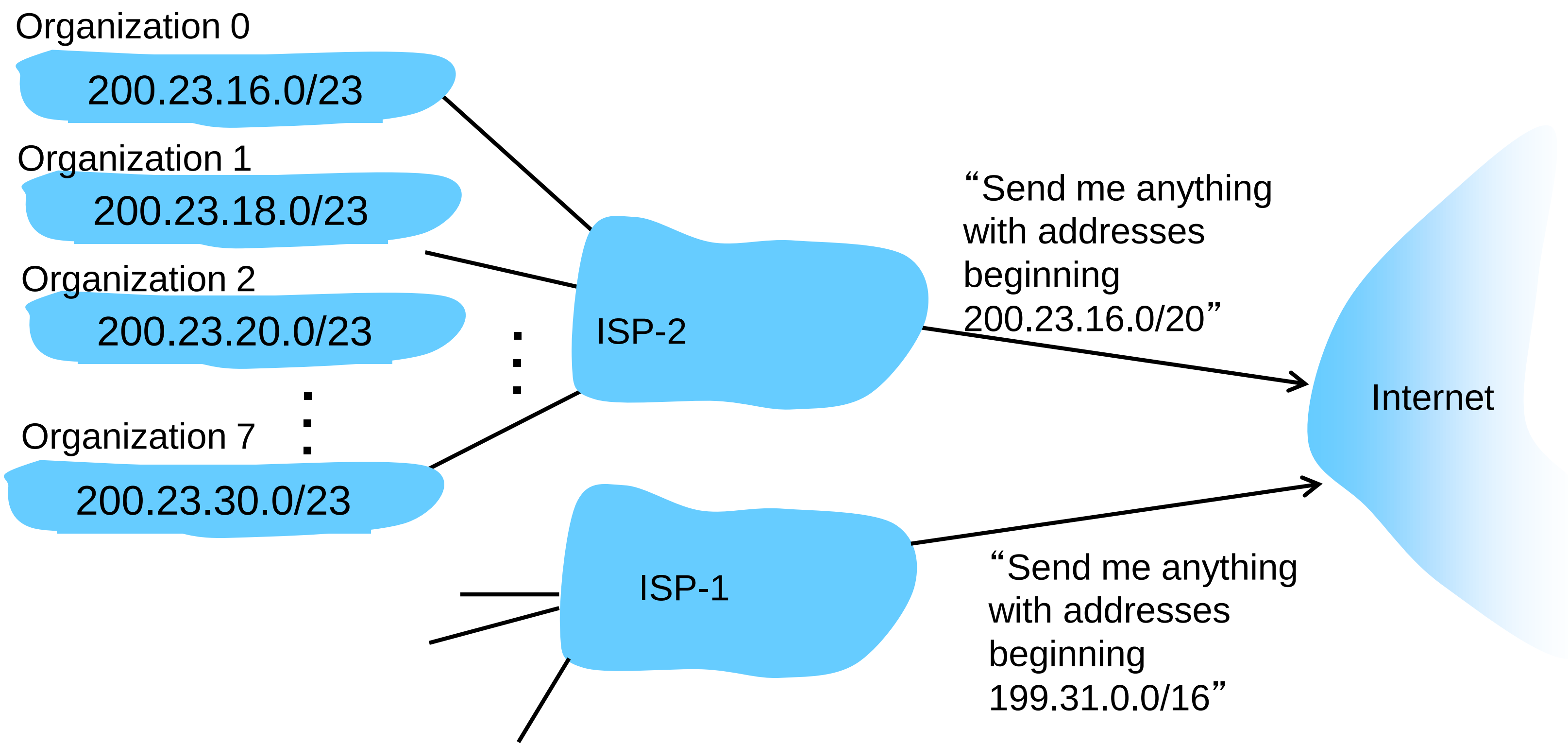
Q: how does *network* get subnet part of IP addr?

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 0001000 0	00000000	200.23.16.0/23
Organization 1	11001000 00010111 0001001 0	00000000	200.23.18.0/23
Organization 2	11001000 00010111 0001010 0	00000000	200.23.20.0/23
...	.....	....	....
Organization 7	11001000 00010111 0001111 0	00000000	200.23.30.0/23

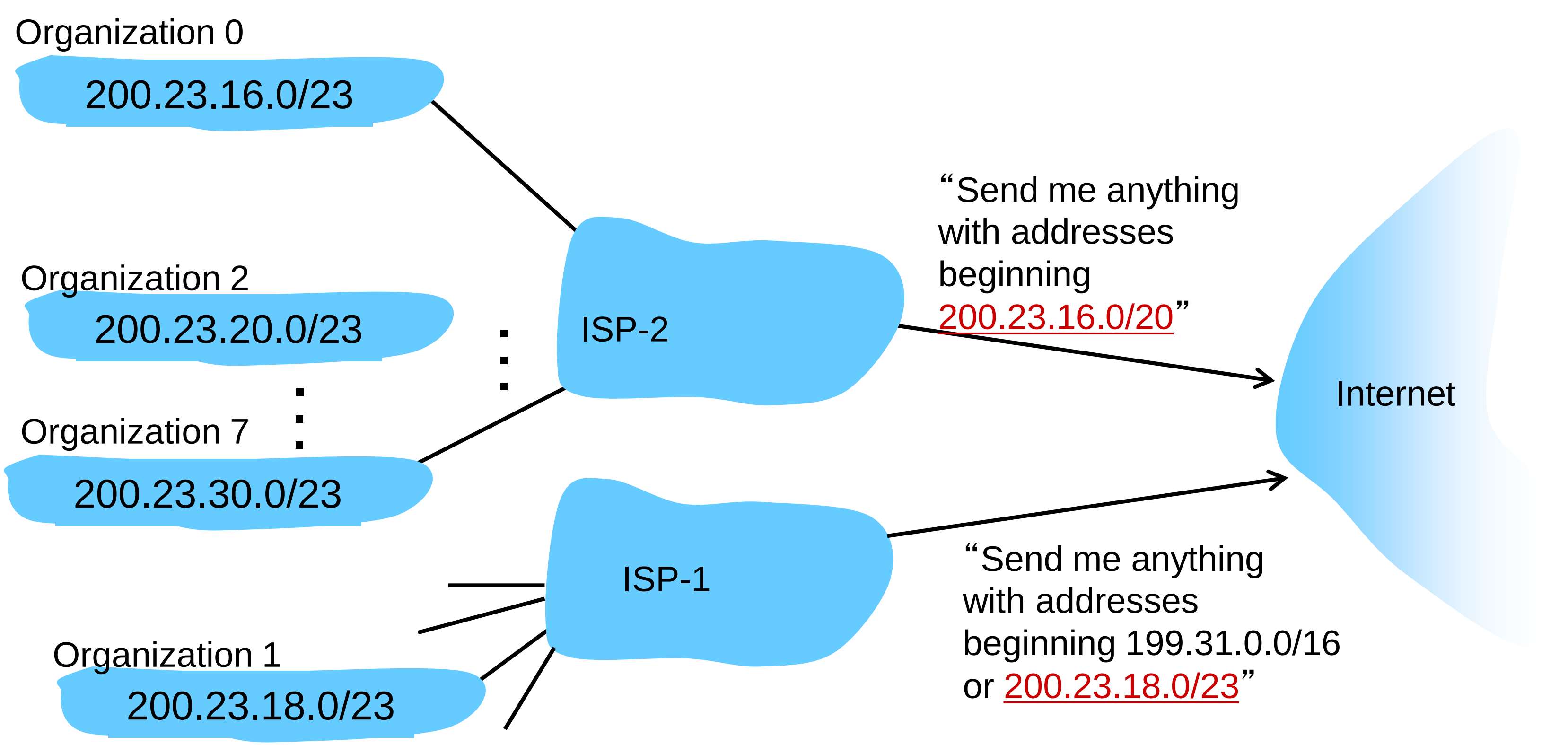
Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



Hierarchical addressing: more specific routes

ISP-1 now has a more specific route to Organization 1



What have we learned?

- IPv4 addresses 4 dotted decimal numbers – 127.0.0.1
- Classless InterDomain Routing. The "slash" notation says which part of the IPv4 address is "network" and what is "host".
  - Example: 135.10.10.0/24 – first 24 bits are "network" that is 135.10.10.? all on same network
- Subnetting splits a network into several subnets.
- We subnet most efficiently by picking the largest subnet first.
- DHCP (Dynamic Host Configuration Protocol) allows a host to get an IP address from the network it connects to.