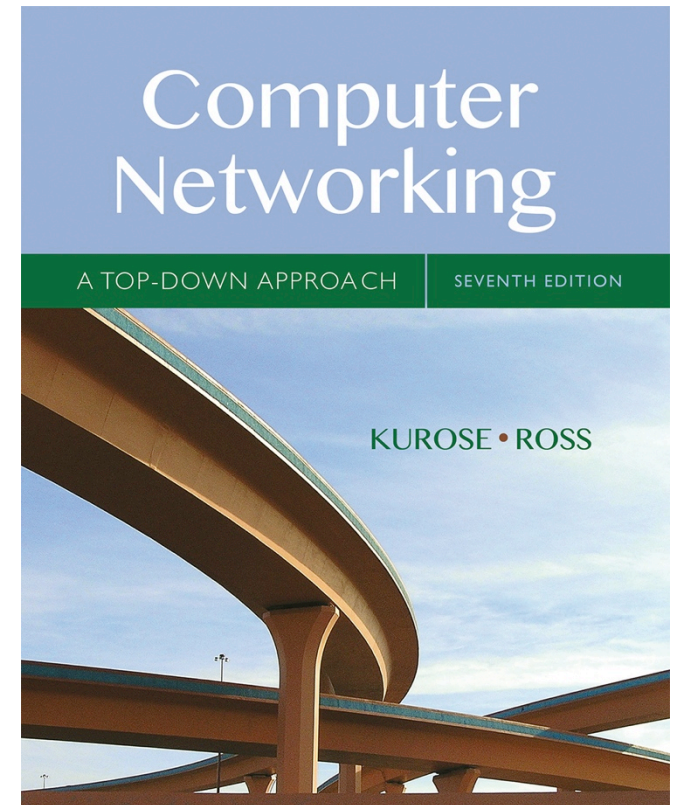# Chapter 4
Network Layer: The Data Plane

A note on the use of these Powerpoint slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Computer Networking: A Top Down Approach

7th edition
Jim Kurose, Keith Ross
Pearson/Addison Wesley
April 2016

Chapter 4: outline

Chapter 4: network layer

chapter 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

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
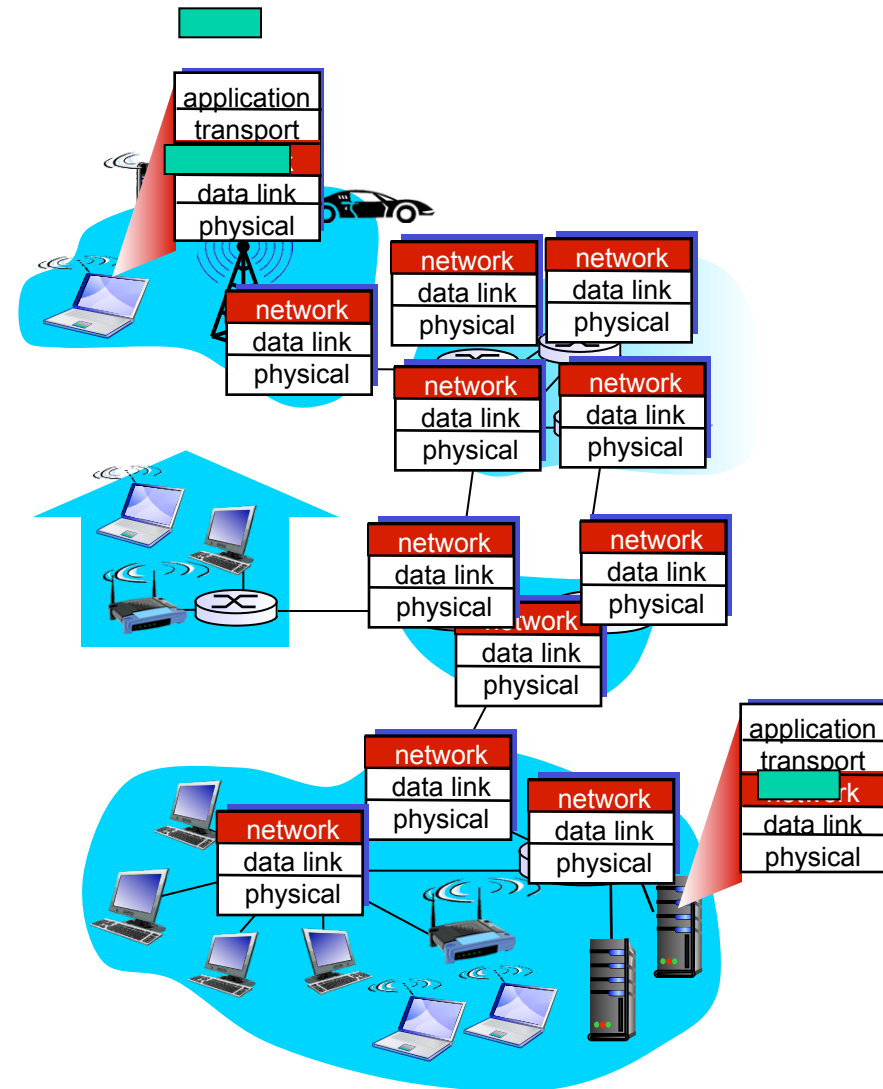
Two key network-layer functions

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 single interchange

- *routing:* process of planning trip from source to destination

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
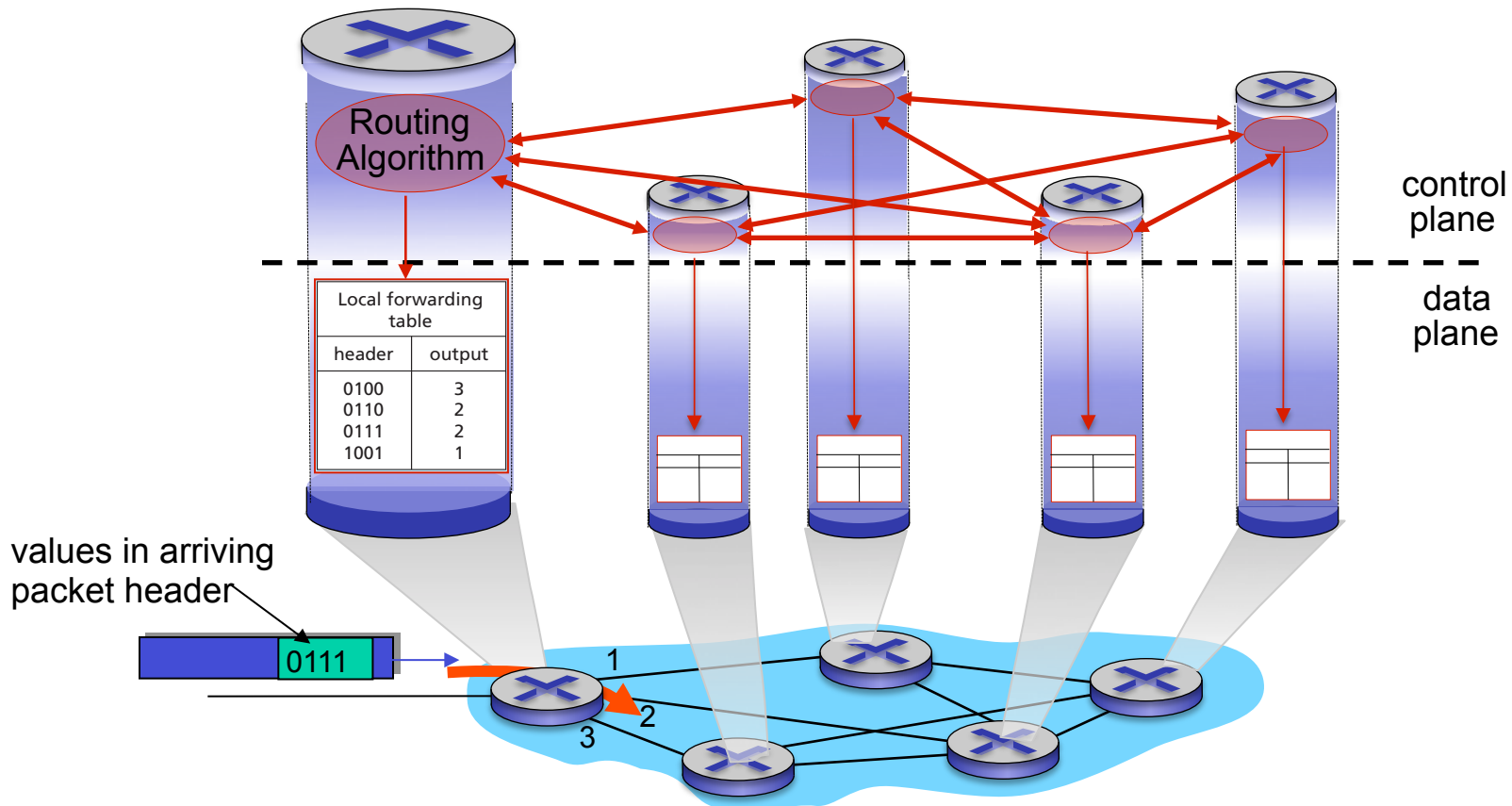- forwarding function

values in arriving packet header

0111

1
2
3
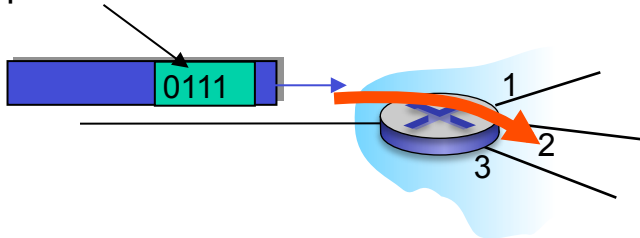
Control plane

- 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

Routing Algorithm

Local forwarding table

header	output
0100	3
0110	2
0111	2
1001	1

control plane

data plane

values in arriving packet header

0111

1

2

3

Logically centralized control plane

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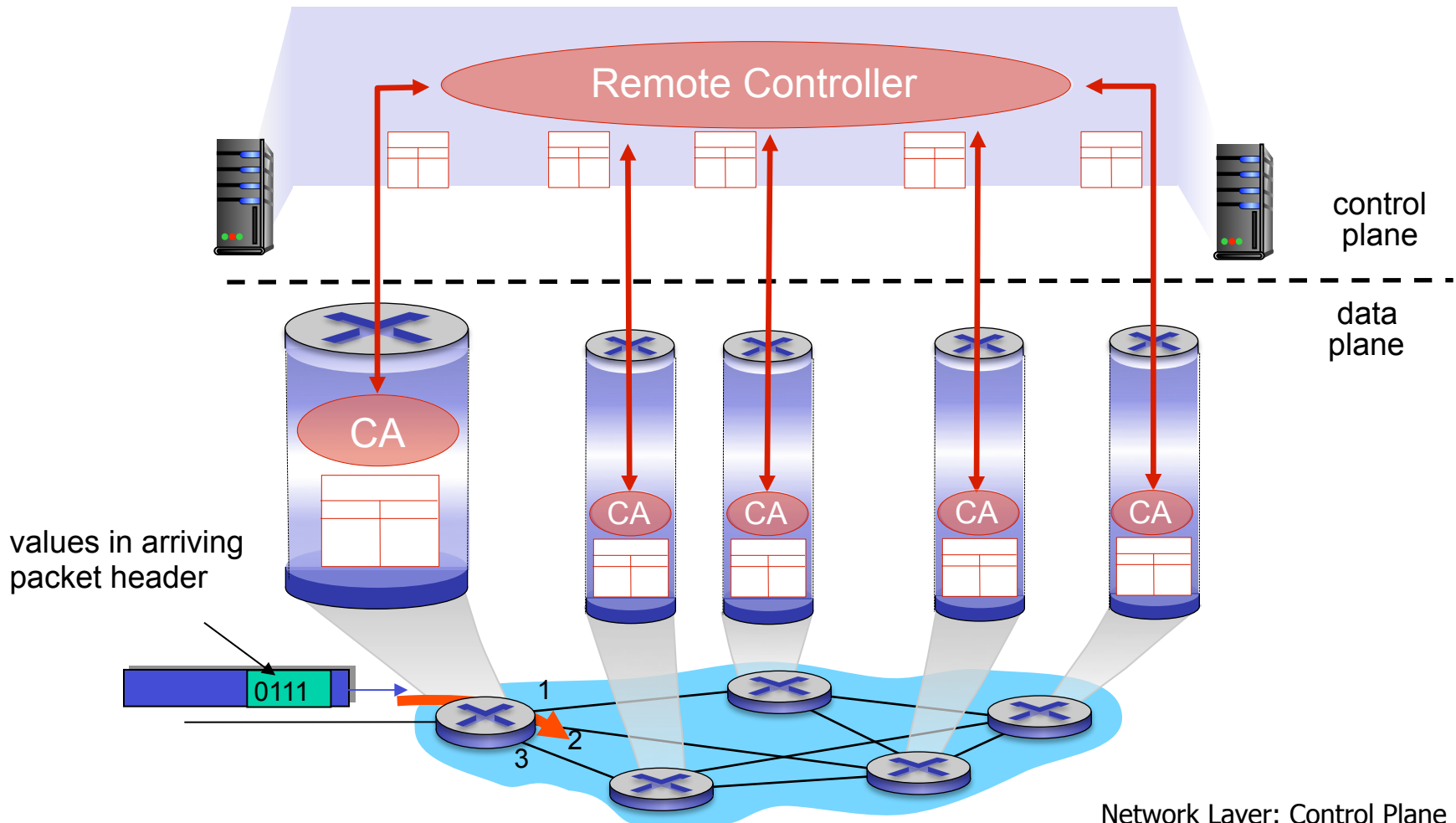
Network service model

Q: What *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 ?				Congestion feedback
		Bandwidth	Loss	Order	Timing	
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

Chapter 4: outline

Router architecture overview

- high-level view of generic router architecture:
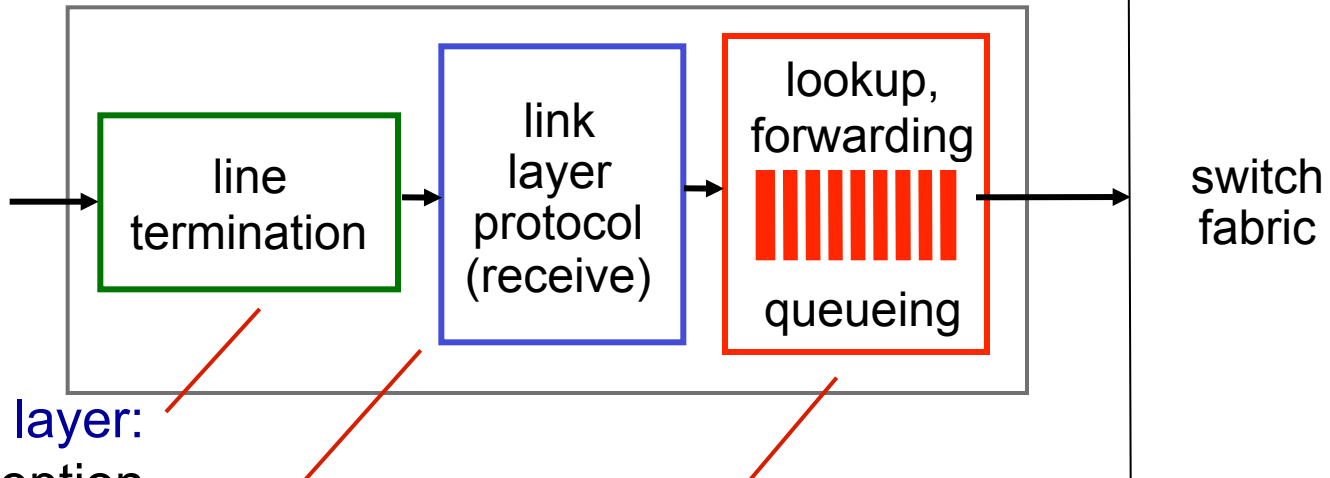
routing, management control plane (software) operates in millisecond time frame

forwarding data plane (hardware) operttes in nanosecond timeframe

routing processor

high-seed switching fabric

router input ports

router output ports

Input port functions

physical layer:
bit-level reception

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

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

line termination → link layer protocol (receive) → lookup, forwarding queueing → switch fabric
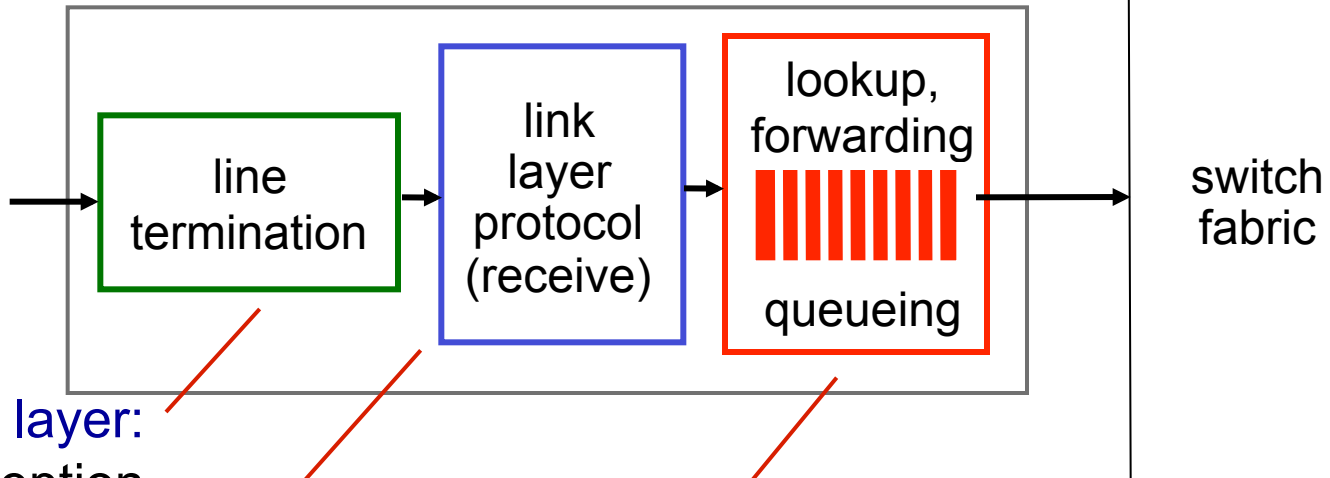
physical layer:
bit-level reception

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

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

Destination-based forwarding

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010111 11111111	0
11001000 00010111 00011000 00000000 through 11001000 00010111 00011000 11111111	1
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

forwarding table

Q: but what happens if ranges don't divide up so nicely?

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)
  - *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

memory   bus   crossbar
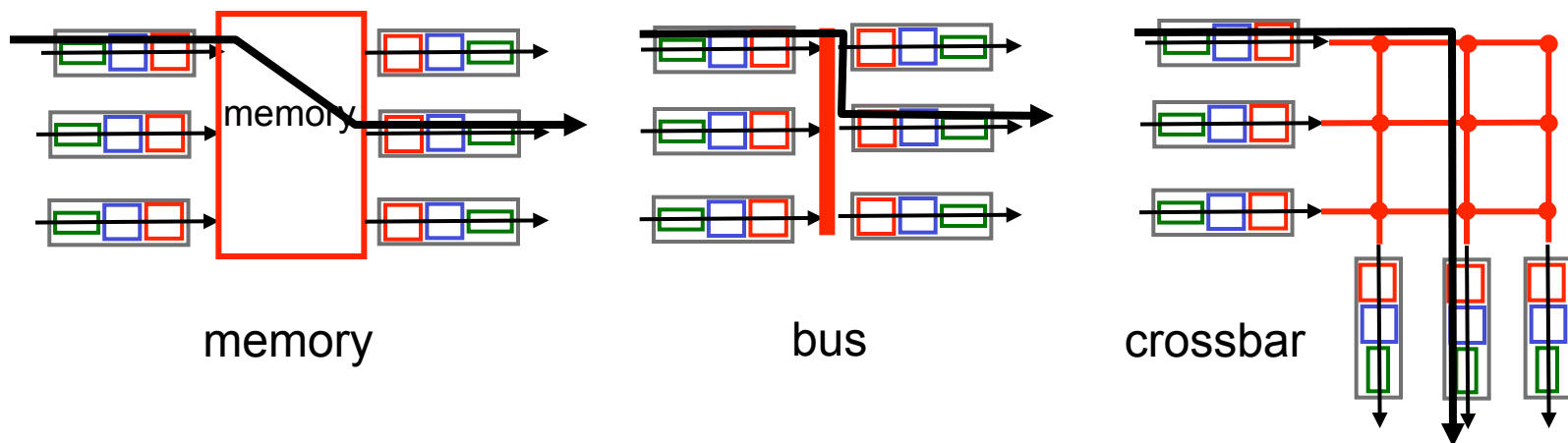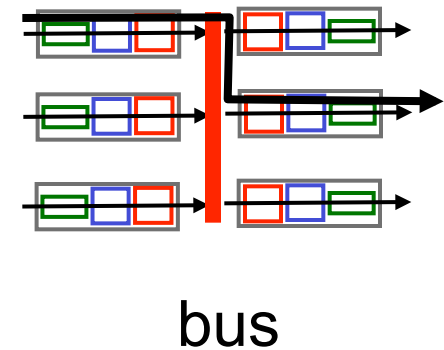
Switching via memory

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)

| input port (e.g., Ethernet) | memory | output port (e.g., Ethernet) |

system bus
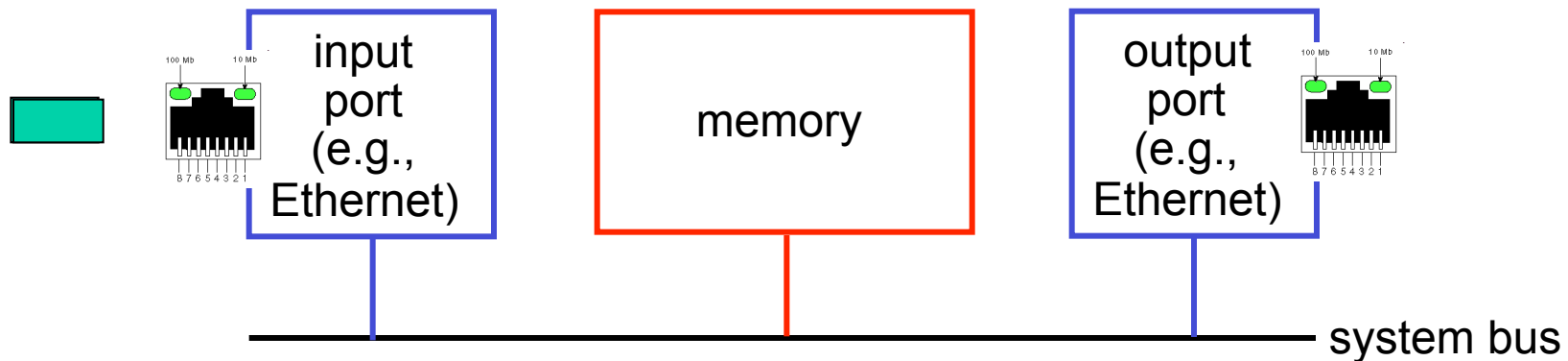
Switching via a bus

- 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
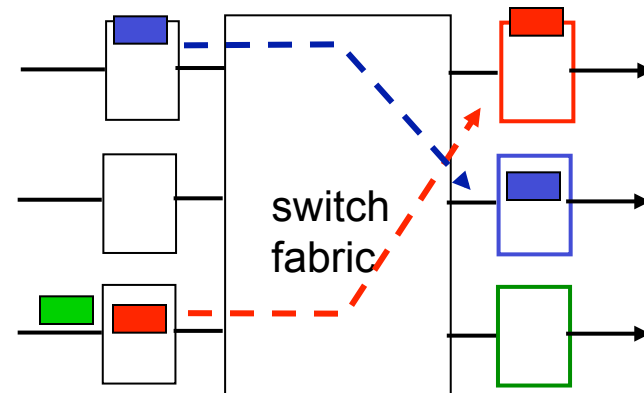
crossbar
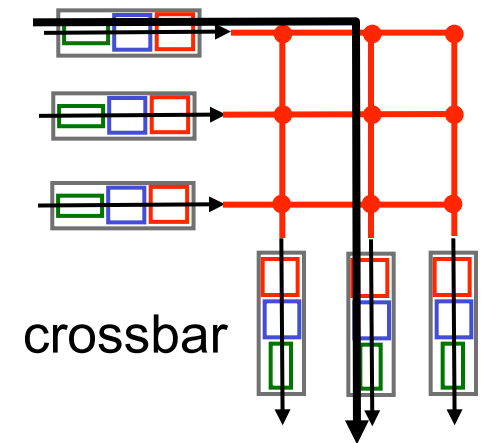
Input port queuing

- 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
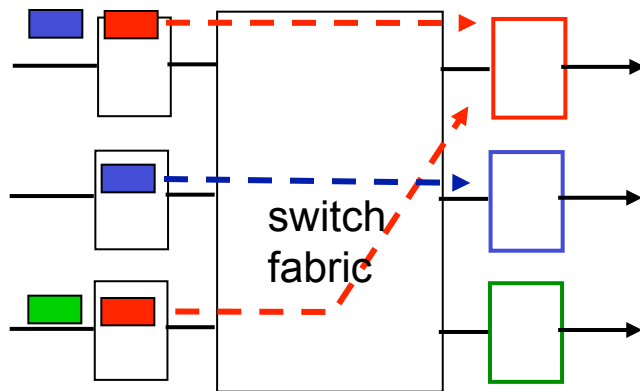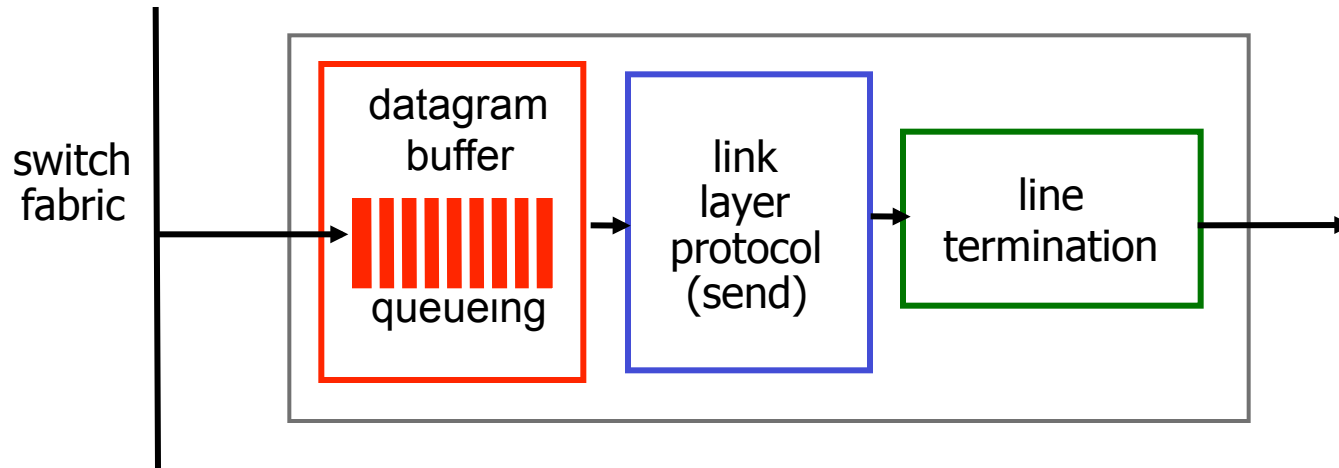

output port contention:
only one red datagram can be transferred.
lower red packet is blocked

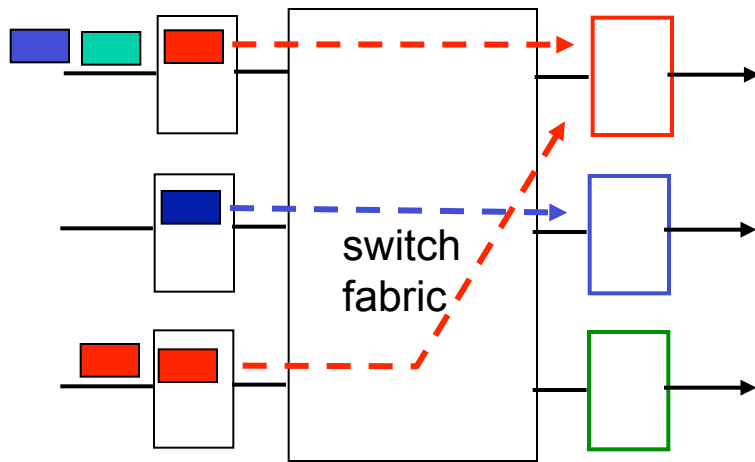one packet time later: green packet experiences HOL blocking

Output ports

- *buffering* required from fabric faster rate

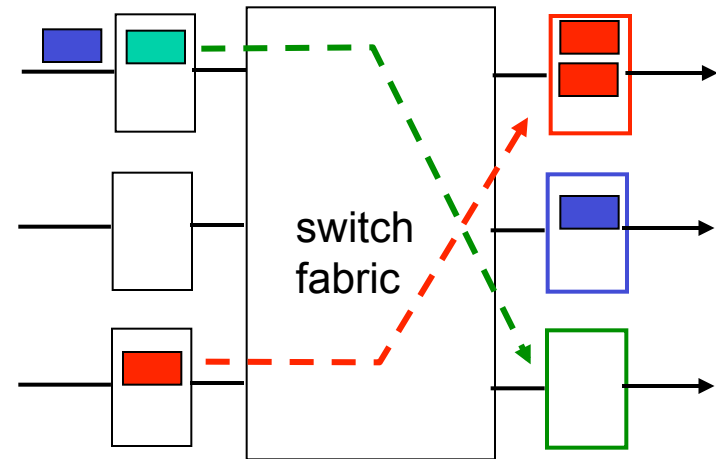 Datagram (packets) can be lost due to congestion, lack of buffers

- *scheduling* datagrams

  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

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

Scheduling mechanisms

■ *scheduling:* choose next packet to send on link

■ *FIFO (first in first out) scheduling:* send in order of arrival to queue

- real-world example?
- *discard policy:* if packet arrives to full queue: who to discard?
  - *tail drop:* drop arriving packet
  - *priority:* drop/remove on priority basis
  - *random:* drop/remove randomly

packet
arrivals

queue
(waiting area)

link
(server)
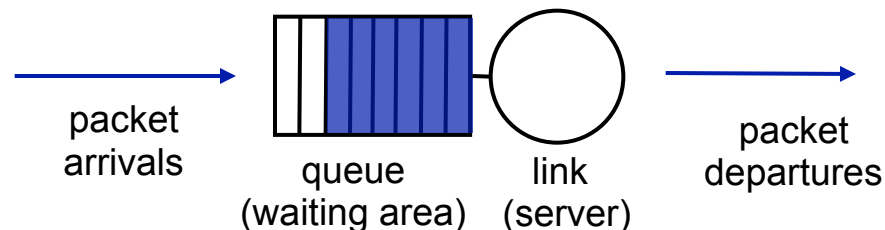
packet
departures

Scheduling policies: priority

priority scheduling: send highest priority queued packet

- multiple *classes*, with different priorities
  - class may depend on marking or other header info, e.g. IP source/dest, port numbers, etc.
  - real world example?
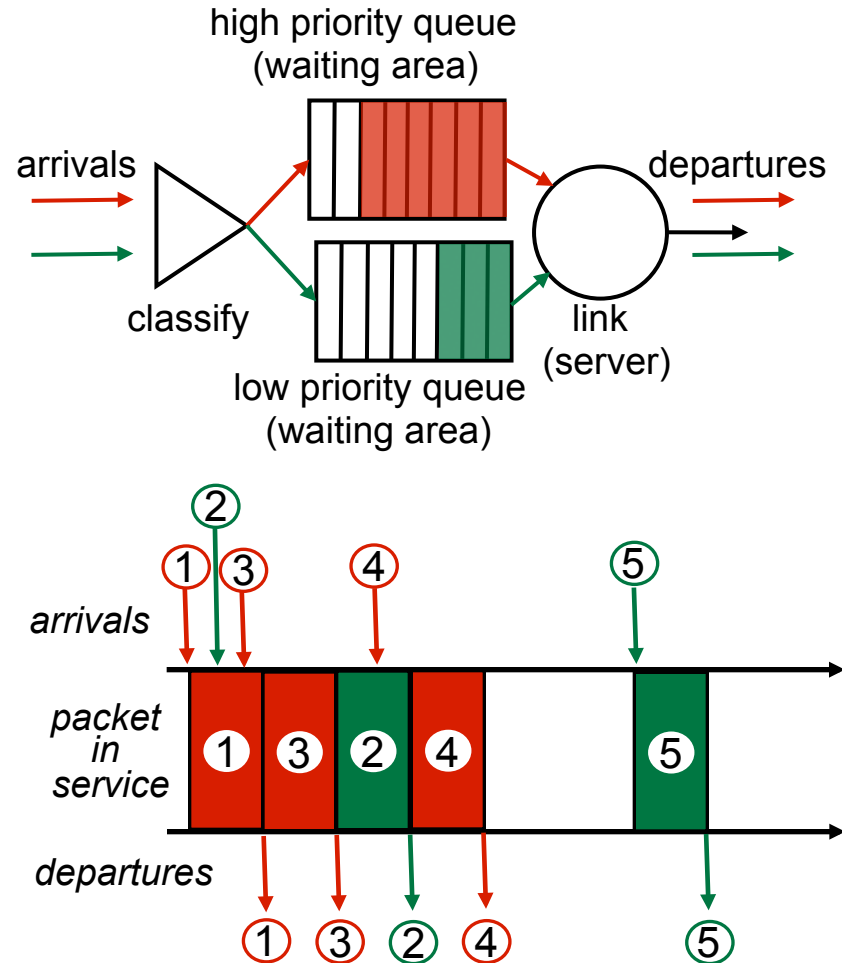
Scheduling policies: still more

Round Robin (RR) scheduling:

- multiple classes
- cyclically scan class queues, sending one complete packet from each class (if available)
- real world example?
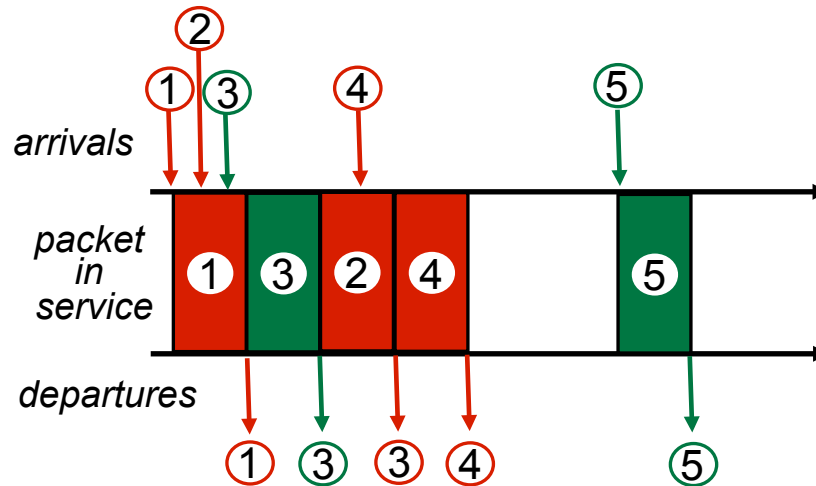
Scheduling policies: still more

Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class gets weighted amount of service in each cycle
- real-world example?
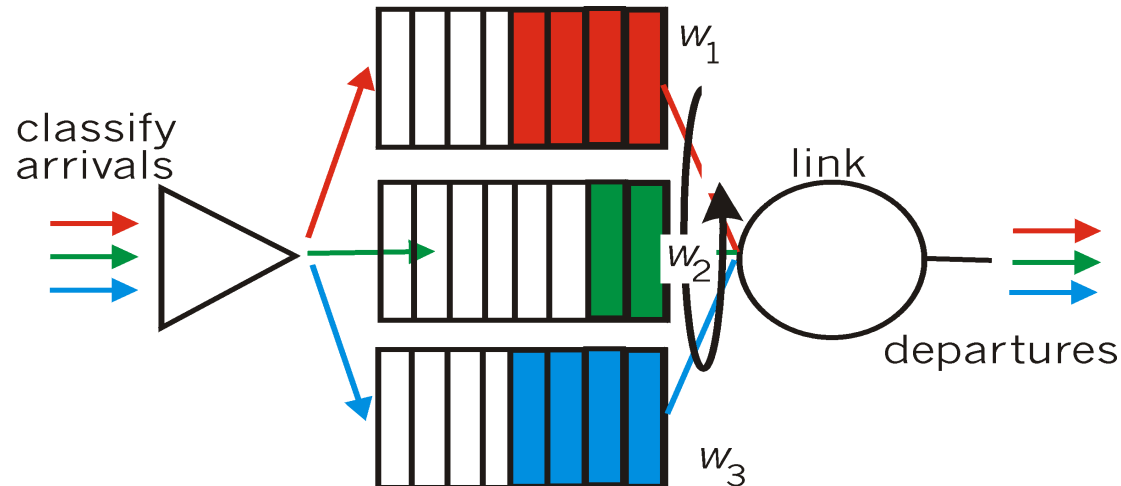
Chapter 4: outline

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:

network
layer

transport layer: TCP, UDP

routing protocols
• path selection
• RIP, OSPF, BGP

forwarding
table

IP protocol
• addressing conventions
• datagram format
• packet handling conventions

ICMP protocol
• error reporting
• router "signaling"

link layer

physical layer

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

32 bits

ver	head. len	type of service	length	
16-bit identifier			flgs	fragment offset
time to live	upper layer		header checksum	
32 bit source IP address				
32 bit destination IP address				
options (if any)				
data (variable length, typically a TCP or UDP segment)				

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

fragmentation:
in: one large datagram
out: 3 smaller datagrams
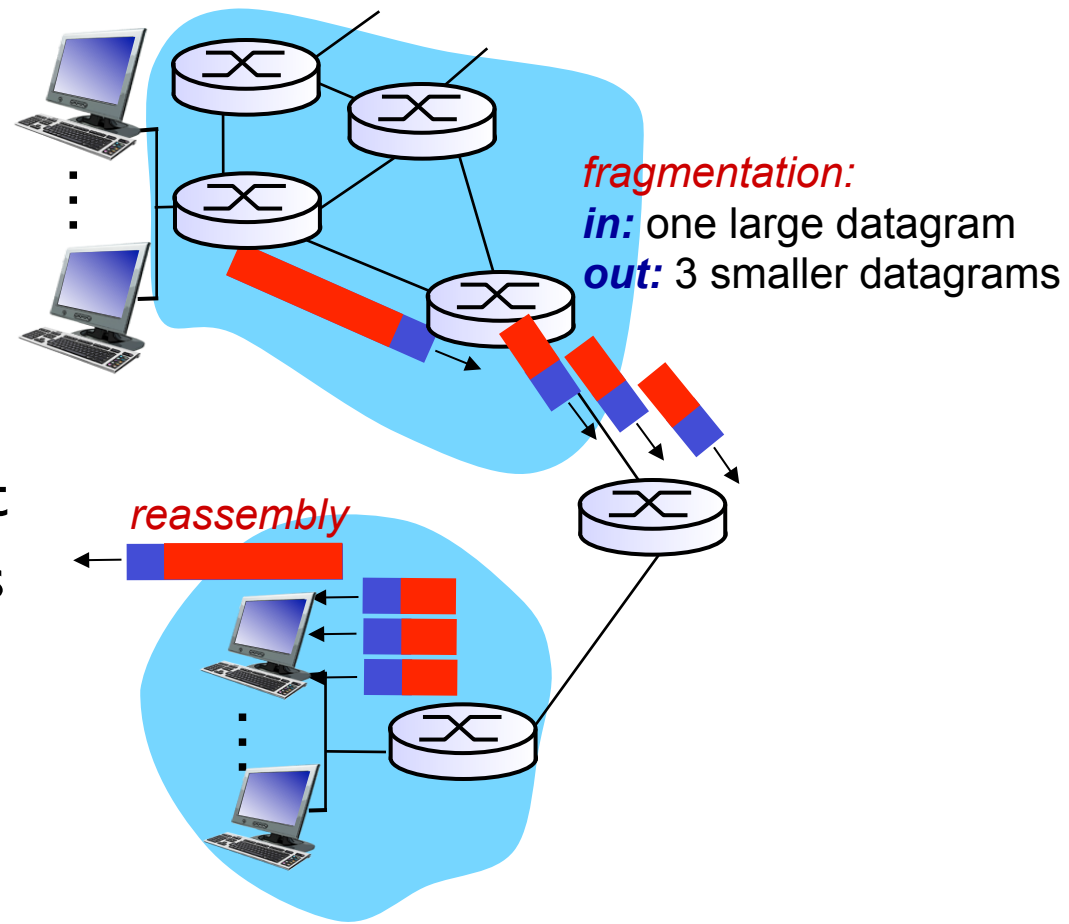
reassembly

IP fragmentation, reassembly

example:

❖ 4000 byte datagram
❖ MTU = 1500 bytes

1480 bytes in data field

offset = 1480/8

| | length =4000 | ID =x | fragflag =0 | offset =0 | |

one large datagram becomes several smaller datagrams

| | length =1500 | ID =x | fragflag =1 | offset =0 | |

| | length =1500 | ID =x | fragflag =1 | offset =185 | |

| | length =1040 | ID =x | fragflag =0 | offset =370 | |

Chapter 4: 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
  - 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*
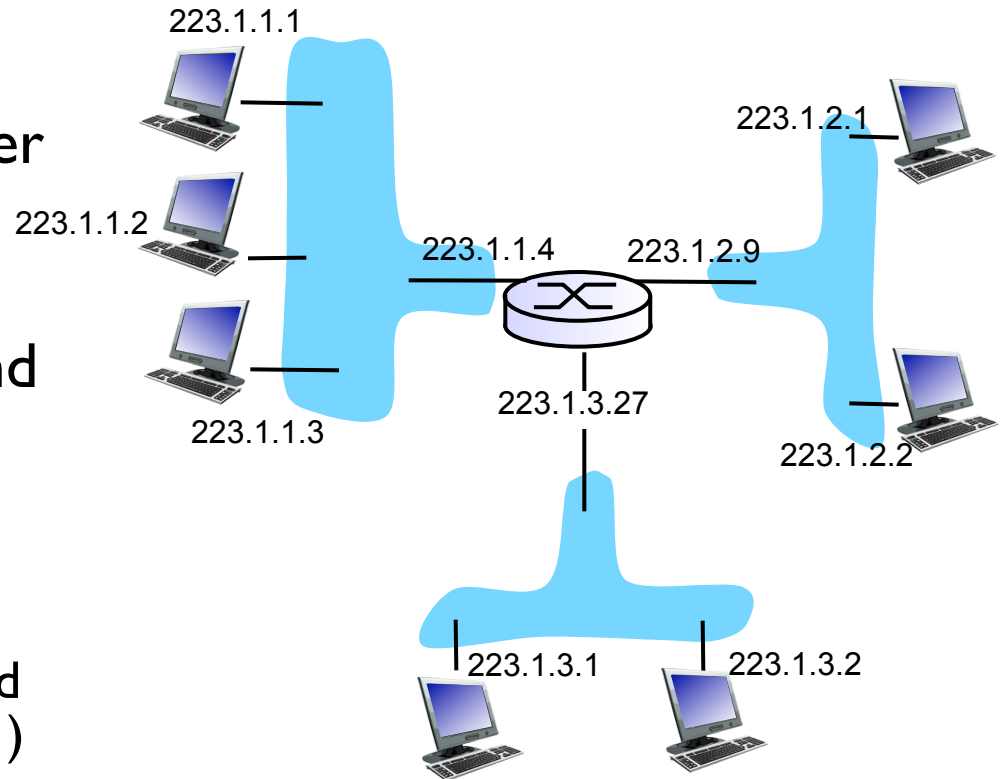
223.1.1.1 = 11011111 00000001 00000001 00000001

        223   1   1   1
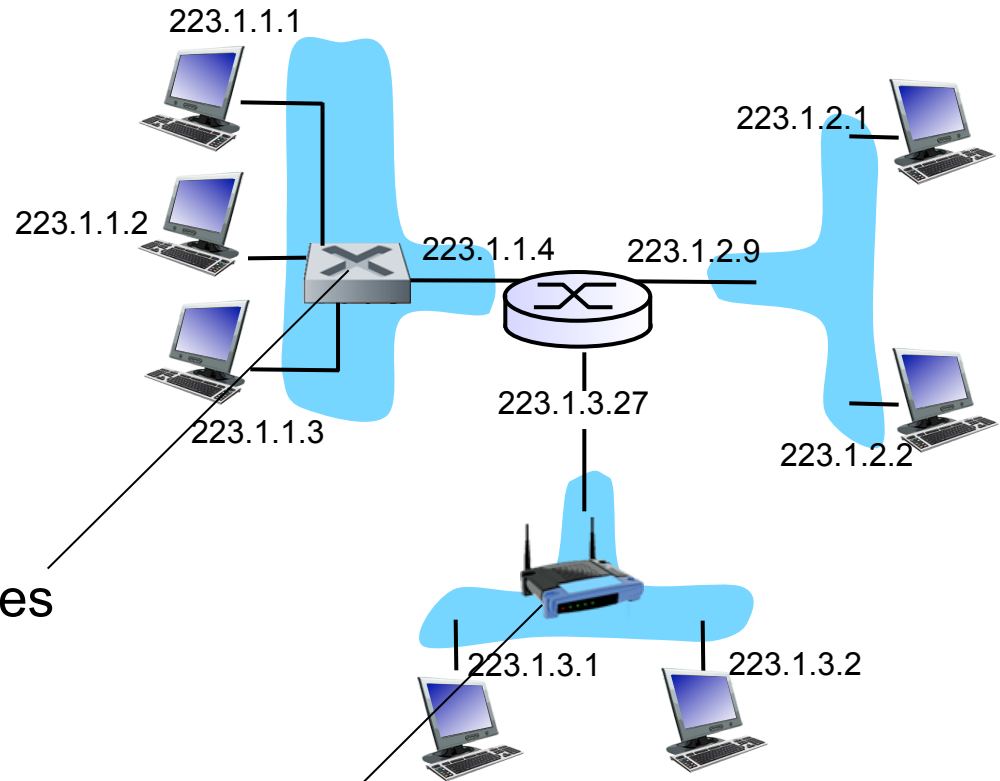
IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in chapter 5, 6.

A: wired Ethernet interfaces connected by Ethernet switches

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

223.1.1.1

223.1.1.2

223.1.1.4   223.1.2.9

223.1.1.3

223.1.2.1

223.1.2.2

223.1.3.27

223.1.3.1   223.1.3.2

A: wireless WiFi interfaces connected by WiFi base station

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
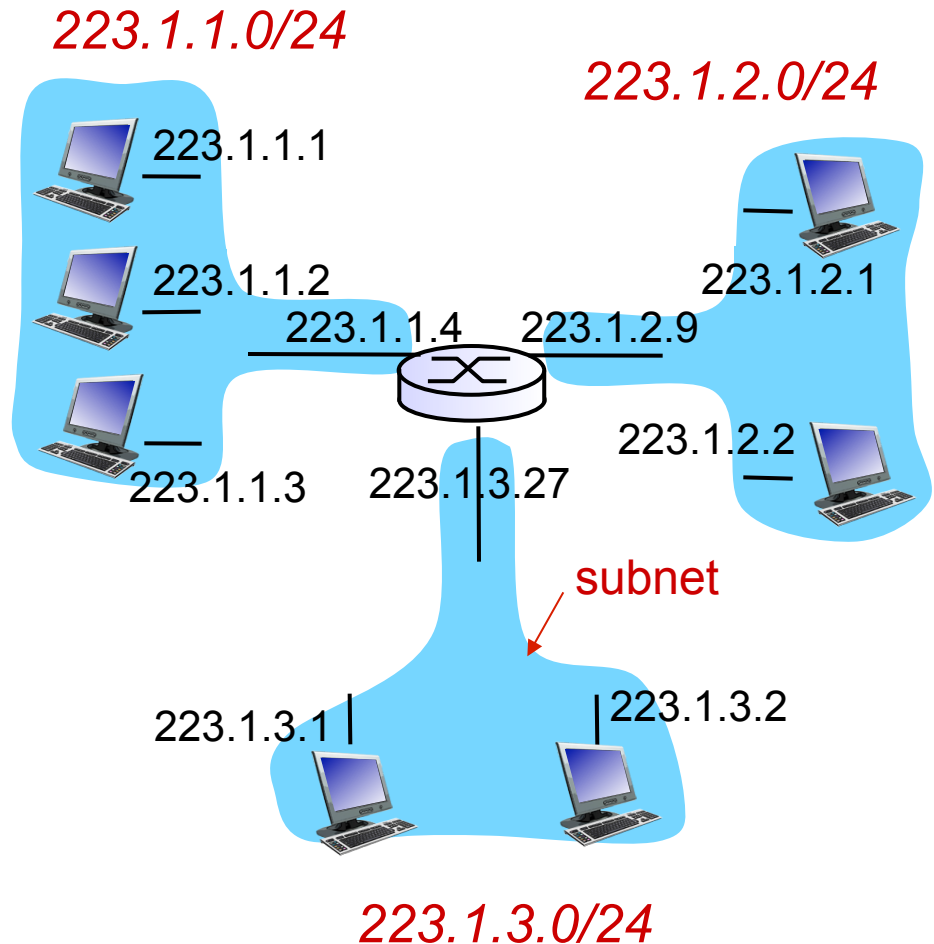 - can physically reach each other *without intervening router*
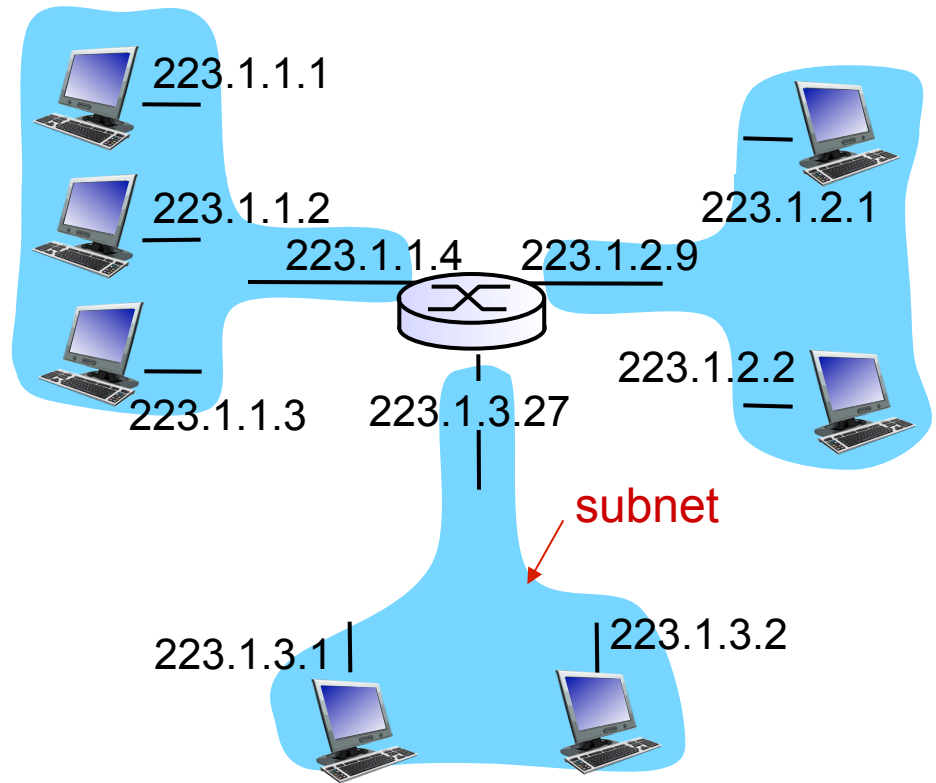
network consisting of 3 subnets

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*

223.1.1.0/24

223.1.2.0/24

223.1.1.1

223.1.1.2

223.1.1.4   223.1.2.9

223.1.2.1

223.1.2.2

223.1.1.3   223.1.3.27

subnet

223.1.3.1   223.1.3.2

223.1.3.0/24

subnet mask: /24

Subnets

how many?

223.1.1.2

223.1.1.1

223.1.1.4

223.1.1.3

223.1.9.2

223.1.7.0

223.1.9.1

223.1.7.1

223.1.8.1

223.1.8.0

223.1.2.6

223.1.3.27

223.1.2.1

223.1.2.2

223.1.3.1

223.1.3.2