Lecture 16: Network Layer Overview, Internet Protocol

COMP 332, Spring 2018

Victoria Manfredi

WESLEYAN UNIVERSITY

Acknowledgements: materials adapted from Computer Networking: A Top Down Approach 7th edition: ©1996-2016, J.F Kurose and K.W. Ross, All Rights Reserved as well as from slides by Abraham Matta at Boston University, and some material from Computer Networks by Tannenbaum and Wetherall.

Today

1. Announcements
   – homework 6 posted
     - discuss: UDP ping server, chat server + reliability
   – midterm graded
2. Network layer
   – overview
   – what's inside a router
   – Internet protocol (IP)
3. Addressing
   – IPv4 addressing
   – usage in routing
   – how to get an IP address
   – IPv6 addressing
   – Dynamic Host Configuration Protocol (DHCP)
   – Network Address Translation (NAT)

Network Layer

OVERVIEW

Network layer

Goal

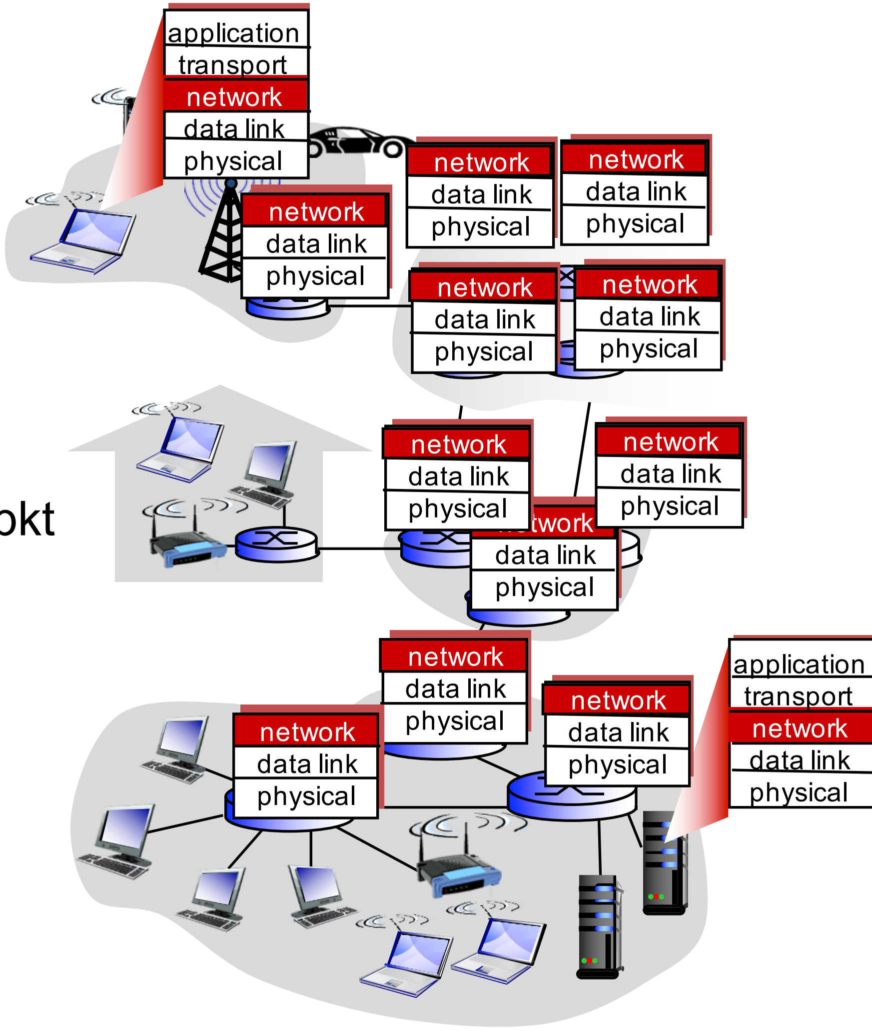
- move pkt from one host to another

How done on Internet?

- routers
  - examine header fields in every IP pkt
  - determines outgoing link

Internet e2e argument

- some functionality only properly implemented in end systems
- smart hosts vs. dumb routers



Network layer is in every host and router on Internet

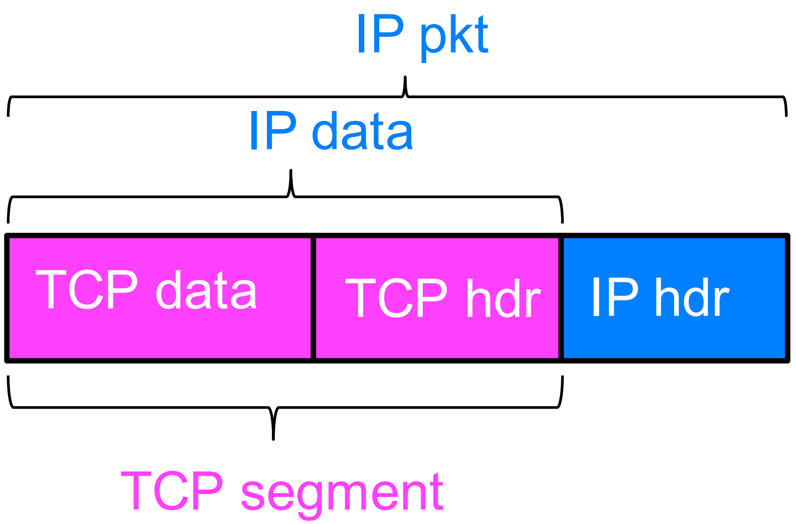
Encapsulation and decapsulation

Sender
- encapsulates segments into packets, puts src, dest IP in IP pkt hdr

Receiver
- decapsulates packets into segments, delivers to transport layer

Max len of TCP data in bytes
- MSS: Max Segment Size
- MSS = MTU – IP hdr – TCP hdr
  - TCP header >= 20bytes

Max len of IP packet in bytes
- MTU: Max Transmission Unit
- 1500 bytes if Ethernet used as link layer protocol



Division of network layer functionality

Trip analogy

1. Control plane
   - comprises traffic only between routers, to compute routes between src and dst
   - network-wide: routers run routing algorithms

→ Plan trip from src to dst

2. Data plane
   - comprises traffic between end hosts, forwarded by routers
   - forwarding table set based on routes computed in control plane
   - local: each router stores, forwards packets

→ Get through one interchange

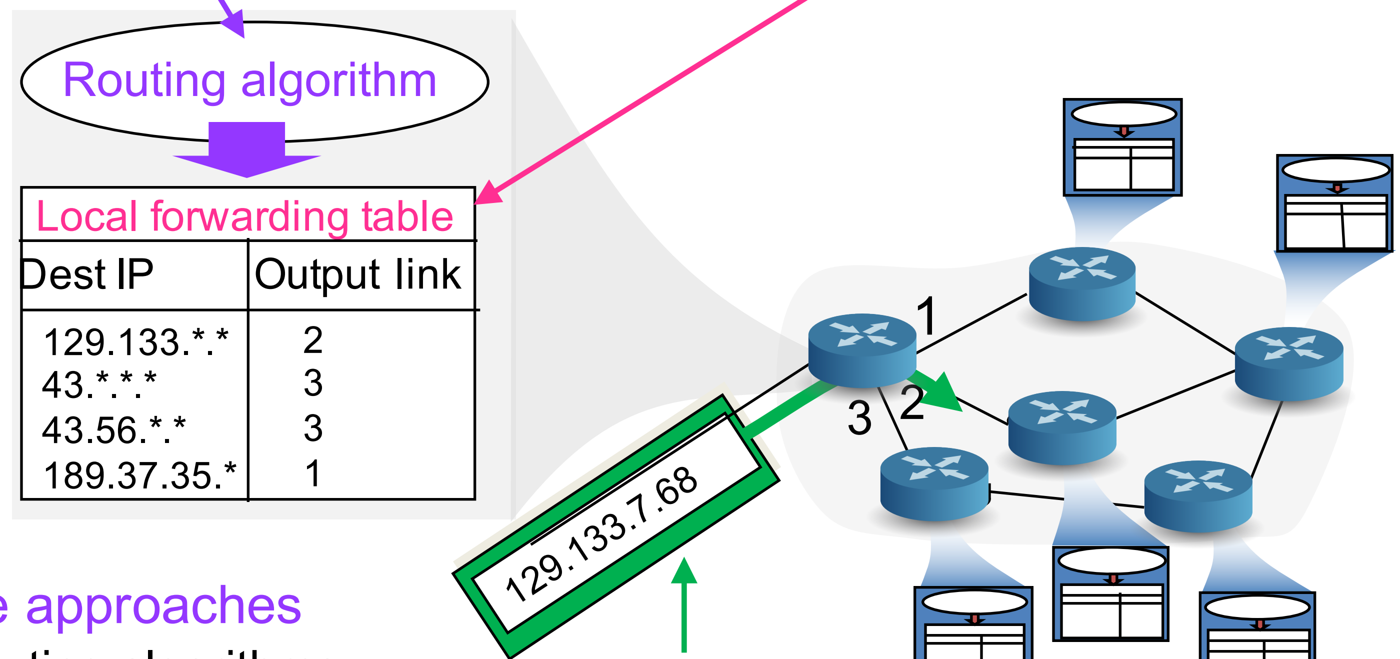
Interplay between routing and forwarding

Routing (slower time scale)

- routers view Internet as graph
- run shortest path algorithms

Forwarding (faster time scale)

- routers use paths to choose best output link for packet destination IP address
- if one link fails, chooses another



Dest IP	Output link
129.133.*.*	2
43.*.*.*	3
43.56.*.*	3
189.37.35.*	1

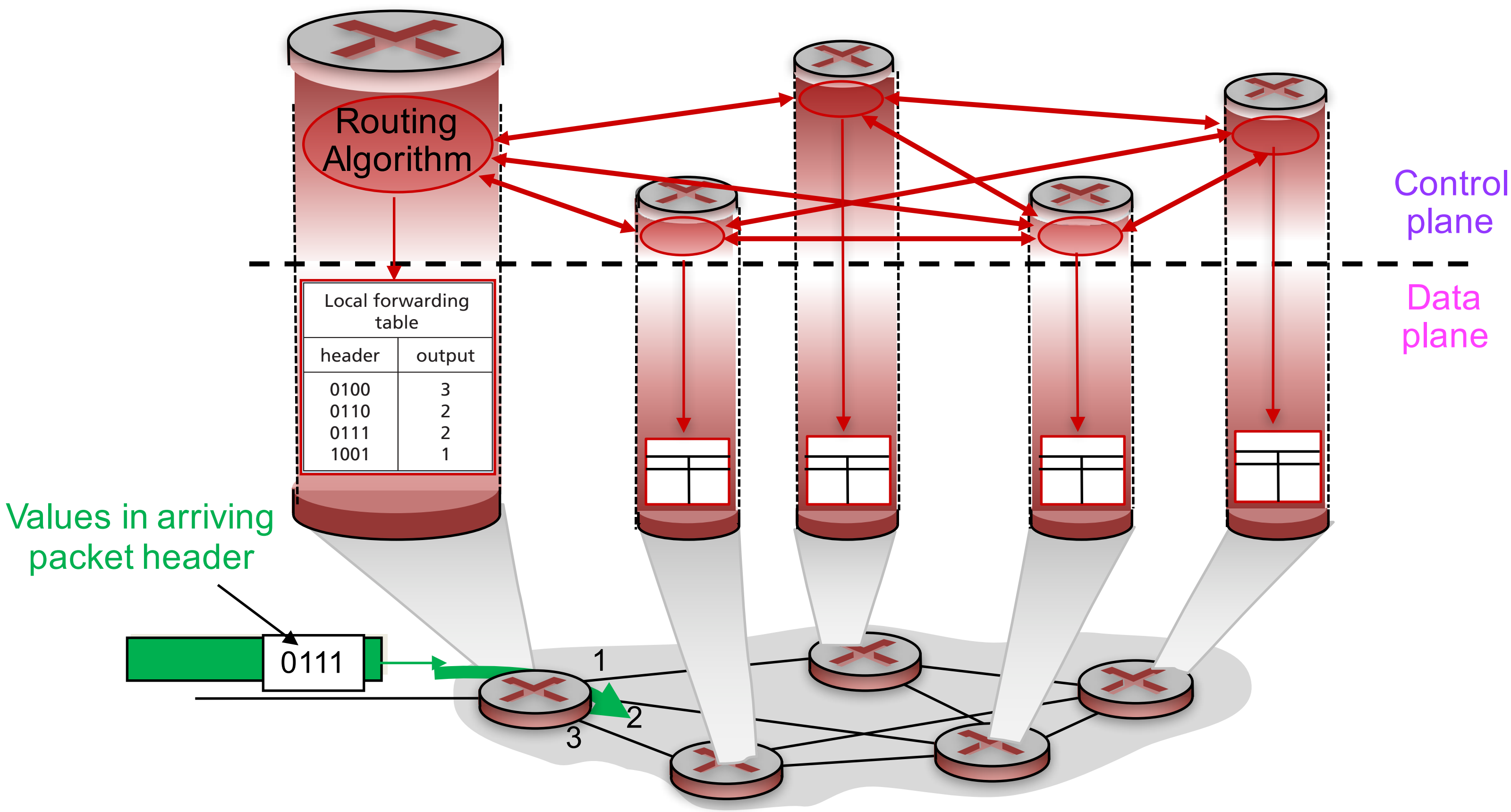
Dest IP addr in header of arriving packet

2 control-plane approaches

1. traditional routing algorithms implemented in routers
2. software-defined networking (SDN) implemented in (remote) servers

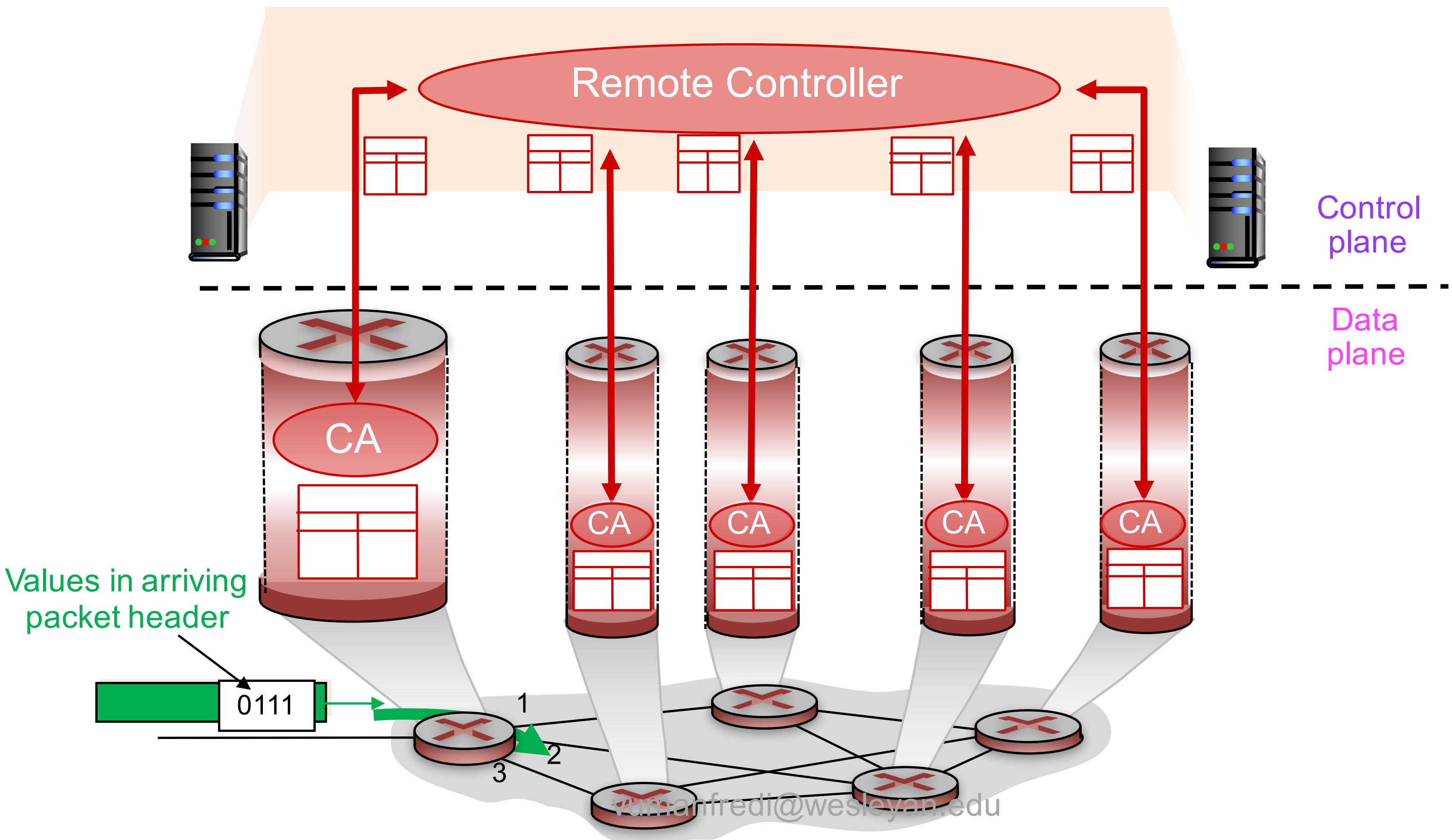
Approach 1: per-router control plane

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



Approach 2: logically centralized control plane

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



Network layer service model

Q: What service model does network layer provide to transport layer for moving packets from sender to receiver?

Example services

- individual packets
  - guaranteed delivery
  - guaranteed delivery with less than 40 ms delay
- flow of packets
  - in-order packet 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

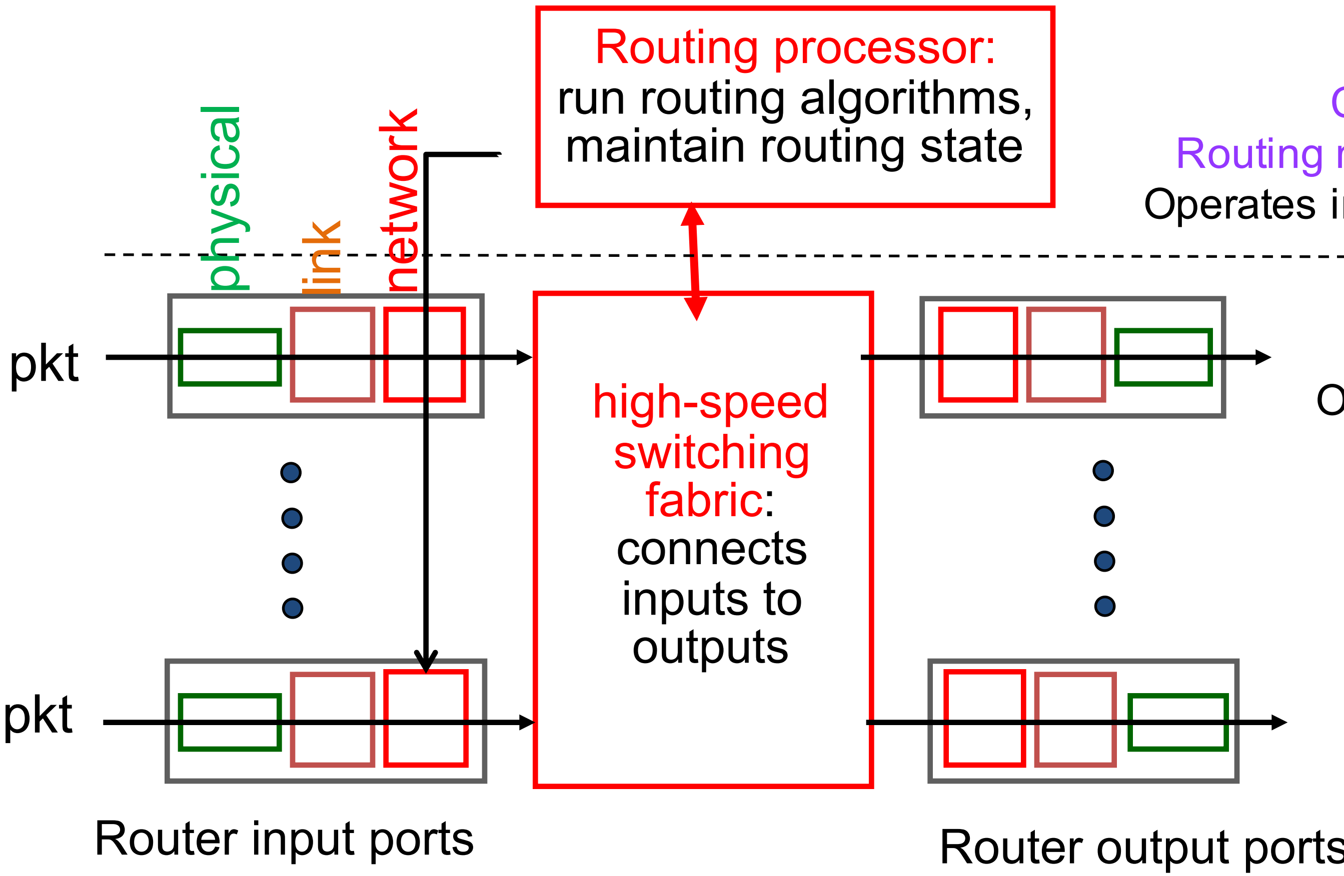
ATM: Asynchronous Transfer Mode
e.g., used in public switched telephone network

Network Layer

WHAT'S INSIDE A ROUTER?

What does a router need to do?

Run routing protocols (control) and store and forward pkts (data)



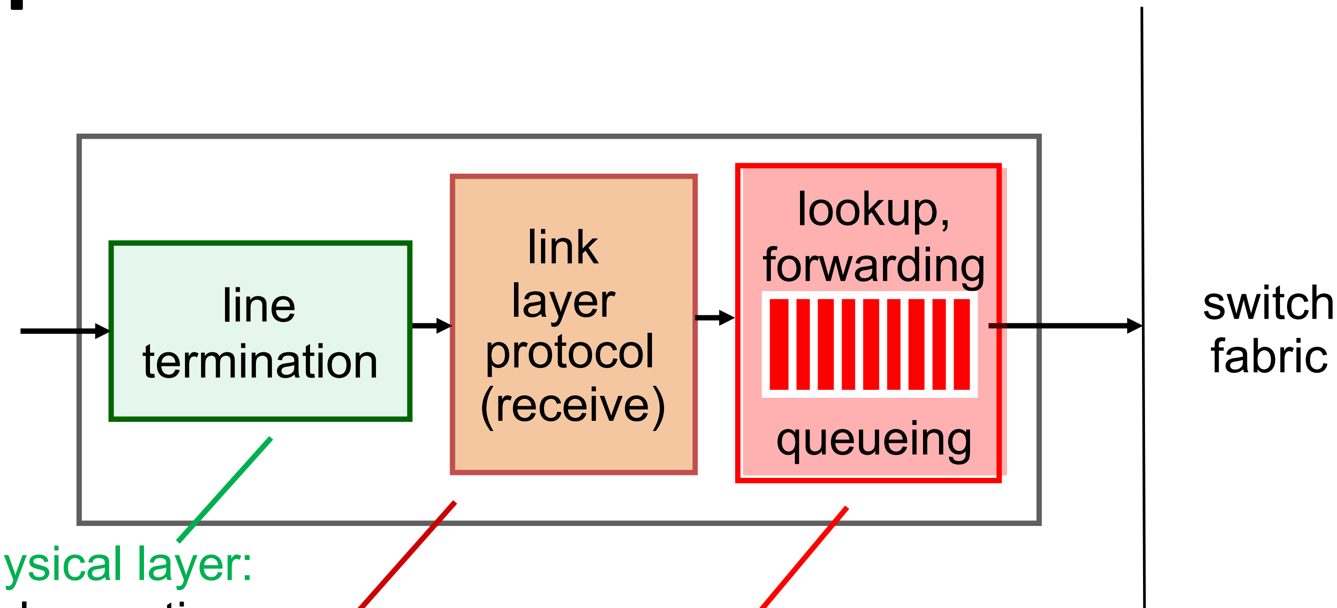
Control plane - complex
Routing management (software)
Operates in millisecond timescale

Data plane - simple
Forwarding (hardware)
Operates in nanosecond timescale

High-level view of generic router architecture

Port is an interface not a socket

Input port functions



Physical layer:
bit-level reception,
terminate phys. conn.

Data link layer:
e.g., Ethernet processing,
error-checking, de-capsulation,

Network layer

- validate/update checksum, decrement TTL
- switching: use header field values, lookup output port
- queue: if packets arrive faster than forwarding rate into switch fabric

Switching fabrics

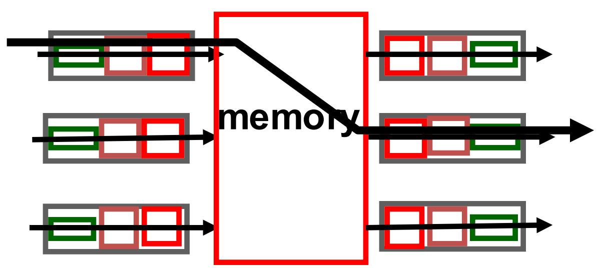
Transfer packet

– from input buffer to appropriate output buffer

Switching rate

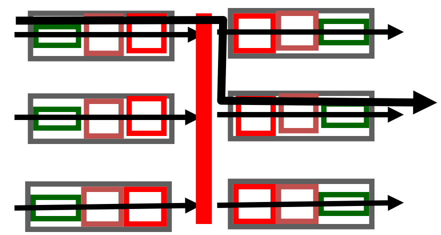
– rate at which packets can be transferred from inputs to outputs
– N inputs: switching rate = N x line rate desirable

3 types of switching fabrics



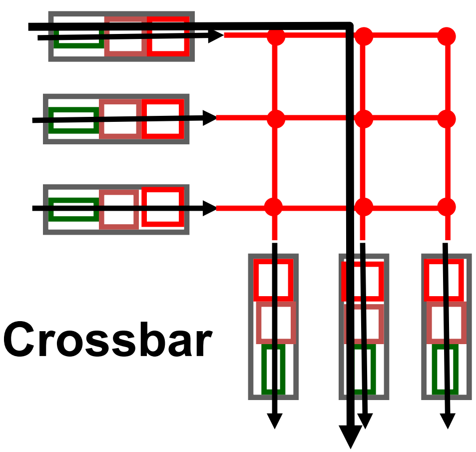
Memory

Speed limited by memory bandwidth



Bus

Speed limited by bus contention



Crossbar

Forward multiple pkts in parallel

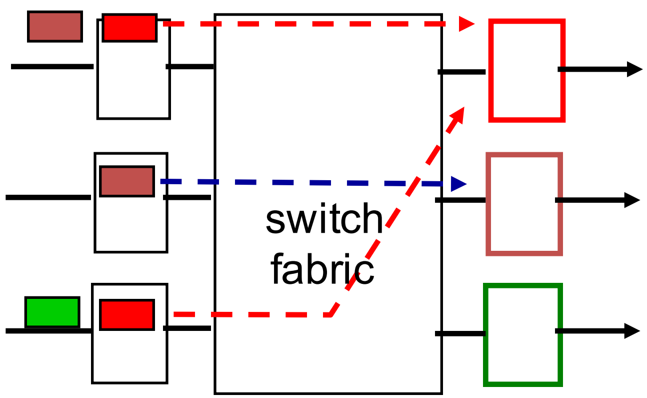
Contention at input ports

Switching fabric slower than input ports combined

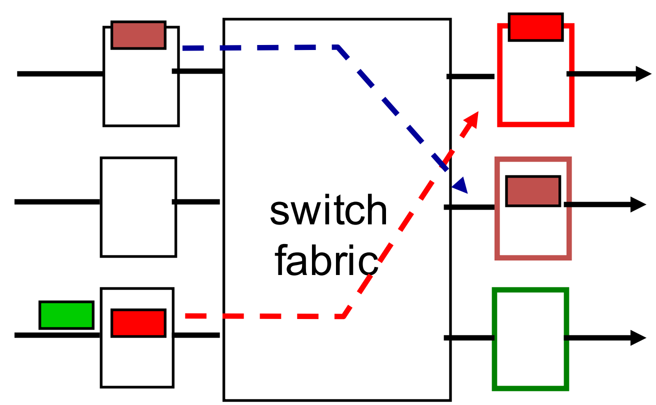
- queueing may occur at input queues
- queueing delay and loss due to input buffer overflow!

Head-of-the-Line (HOL) blocking

- queued pkt at front of queue prevents others from moving forward



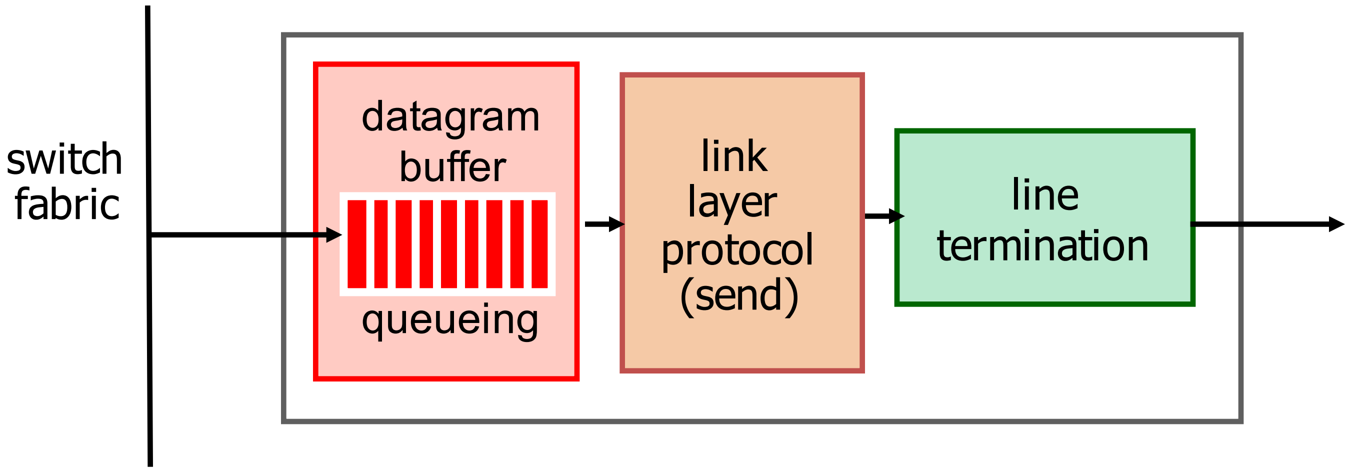
Output port contention: only one red datagram can be transferred.
Lower red packet is blocked



One packet time later: green packet experiences HOL blocking

Contention at output ports

This slide in HUGELY important!



Buffering

- when packets arrive from fabric faster than transmission rate
- packet loss: due to congestion, lack of buffers

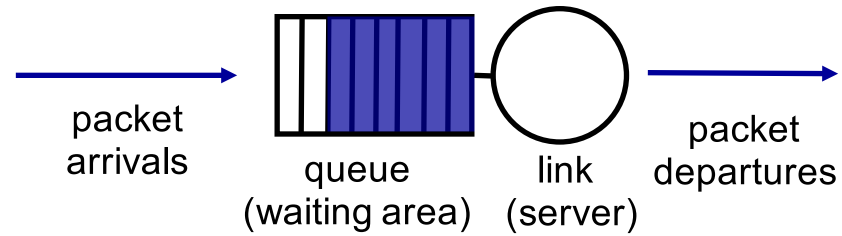
Scheduling

- chooses next among queued packets to transmit on link
- net neutrality: who gets best performance

Scheduling mechanisms

FIFO (first in first out)

– send in order of arrival to queue



Priority

– multiple classes, with different priorities (e.g., based on hdr info)
  - send highest priority queued packet

Round robin scheduling

– multiple classes, cyclically scan class queues
  - send one packet from each class (if available)

Weighted fair queueing

– generalized round robin
  - each class gets weighted amount of service in each cycle

In practice: hardware queues use FIFO,
need software to do priority

Network Layer

INTERNET PROTOCOL

Internet Protocol (IP)

THE network layer protocol of the Internet

- protocol your device **must** implement to run on Internet
- RFC published ~1980

Provides

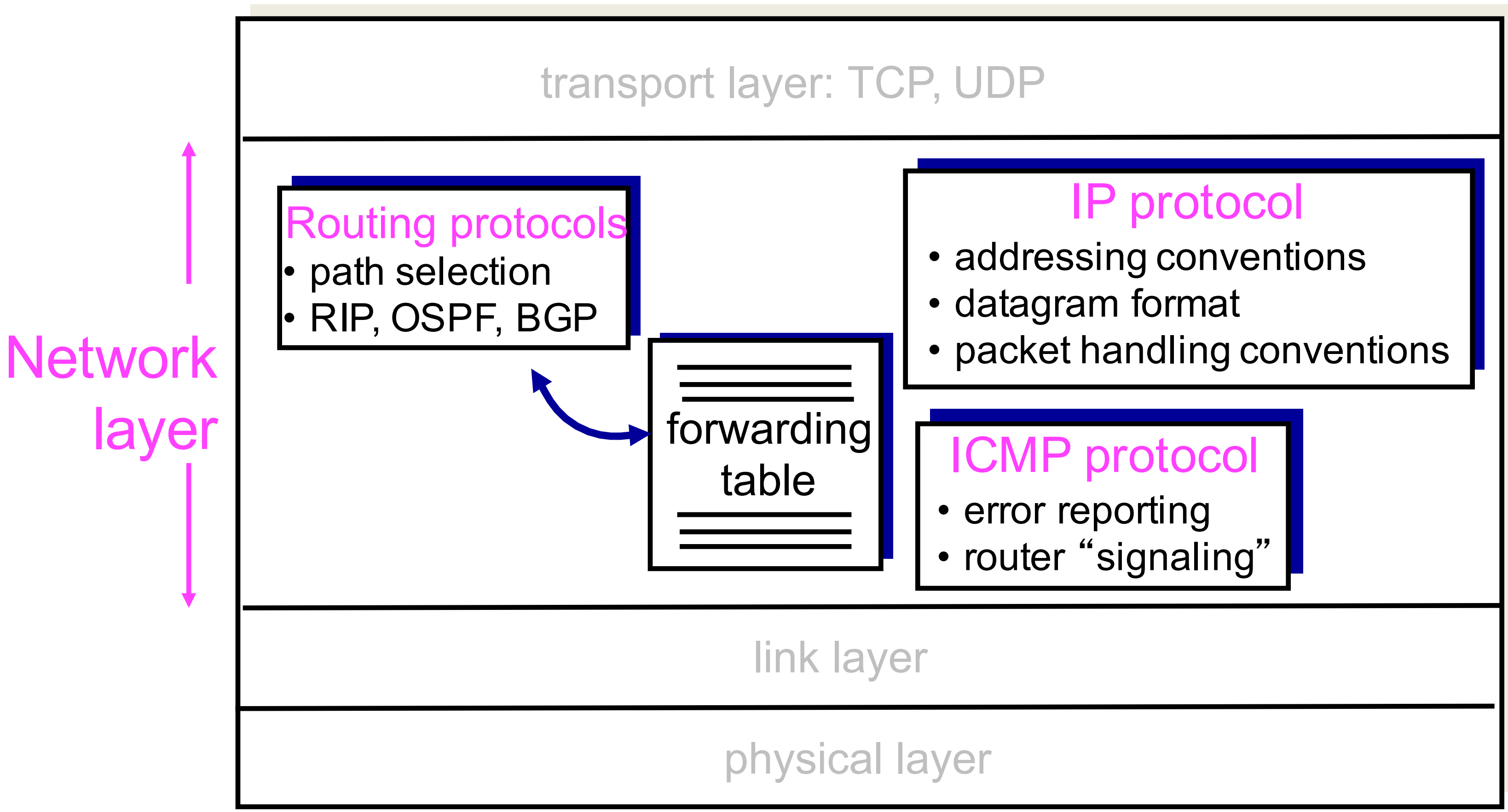
- best effort service
  - to get pkts from one end host to another across many interconnected networks using dst IP address in IP hdr
- addressing
  - format and usage of addresses
- fragmentation
  - e.g., if pkt size exceeds Ethernet MTU of 1500 bytes
- some error detection

Q: what does IP not provide?

- QoS, reliability, ordering, persistent state for e2e flows, connections

Internet's network layer

Network layer functions on hosts and routers



IP packet format

Q: Why is version 1st?
So you know how to parse pkt

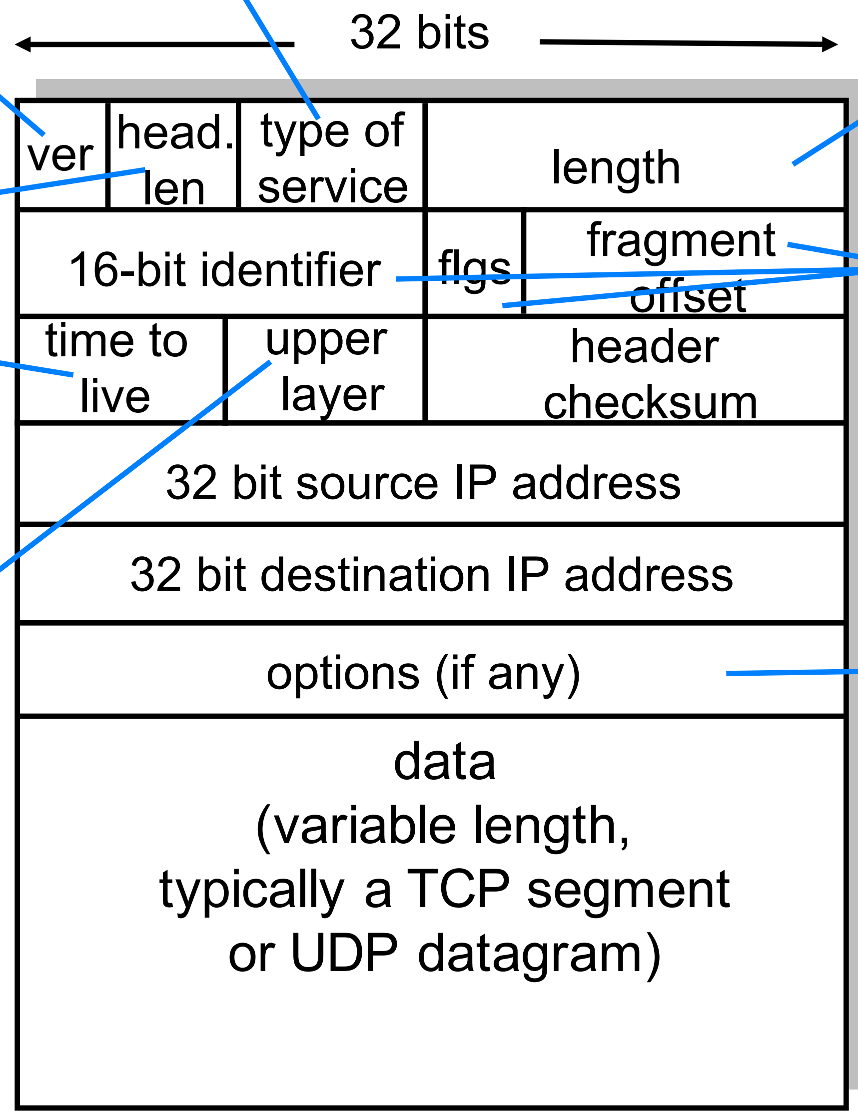
Not widely used

IP protocol version number

header length (# of 32bit words)

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to ICMP=1, UDP=17, TCP=6



total packet length (bytes)

for fragmentation/ reassembly

e.g. timestamp, record route taken, specify routers to visit. When used usually indicate suspicious activity

How much overhead?

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

Bits transmitted left to right, top to bottom

Wireshark

Look at IP headers and ping/traceroute

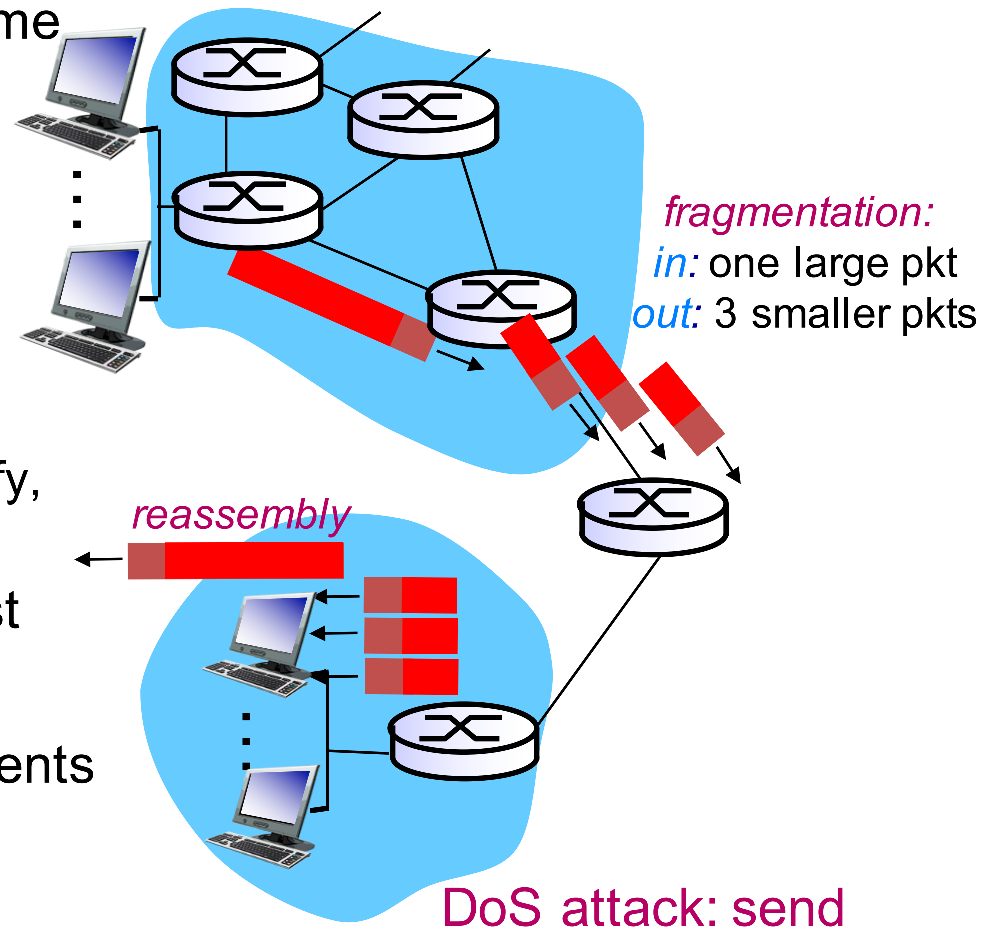
IP fragmentation and reassembly

Network links have MTU

- largest possible link-level frame
- different link types have different MTUs

Fragment when pkt > MTU

- 1 pkt becomes several pkts
  - IP header bits used to identify, order related fragments
- reassembled only at final dest
- re-fragmentation possible
- don't recover from lost fragments
- (IPv6 does not support)

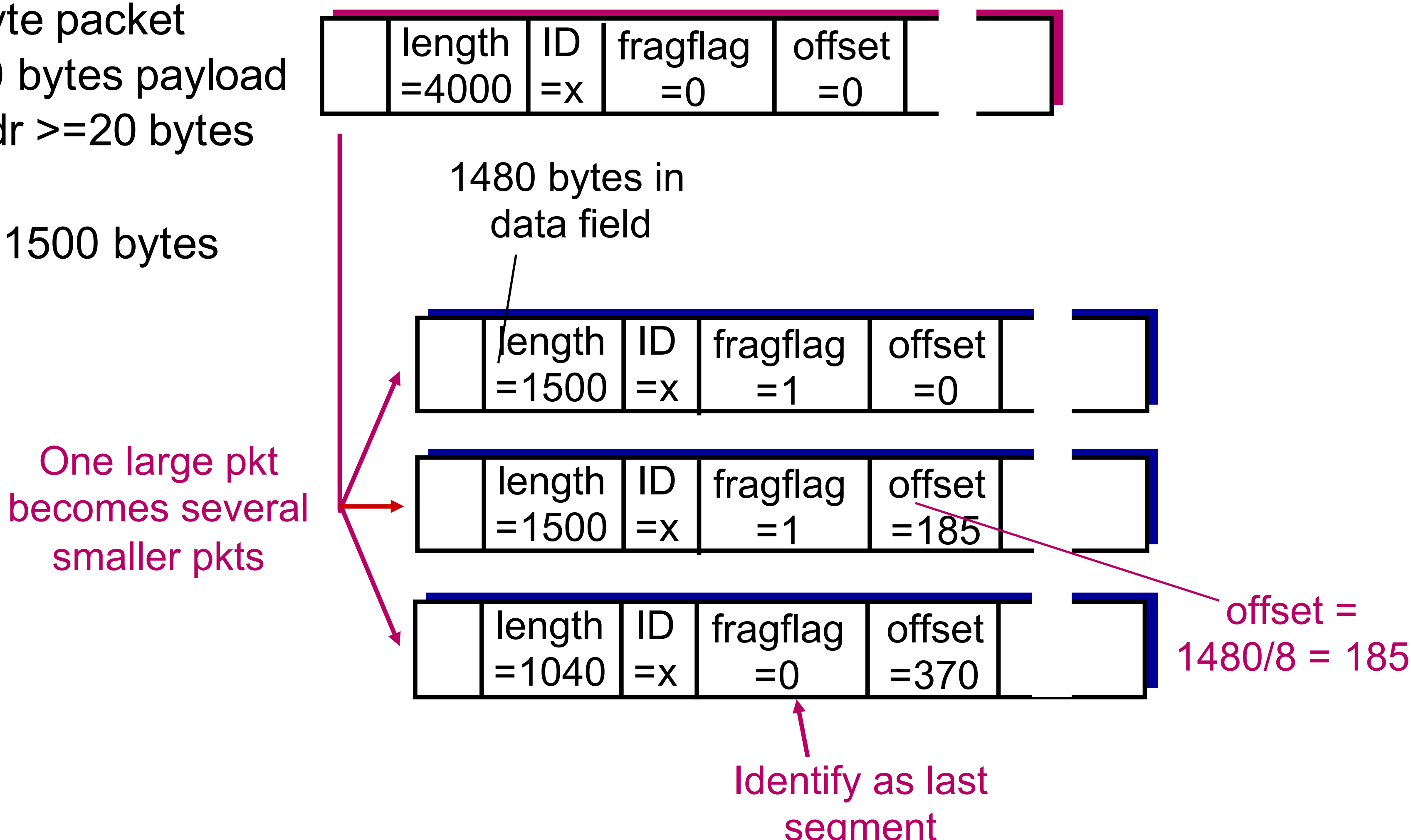


DoS attack: send fragmented pkts but leave one out

IP fragmentation and reassembly

4000 byte packet
- 3980 bytes payload
- IP hdr >=20 bytes

MTU = 1500 bytes



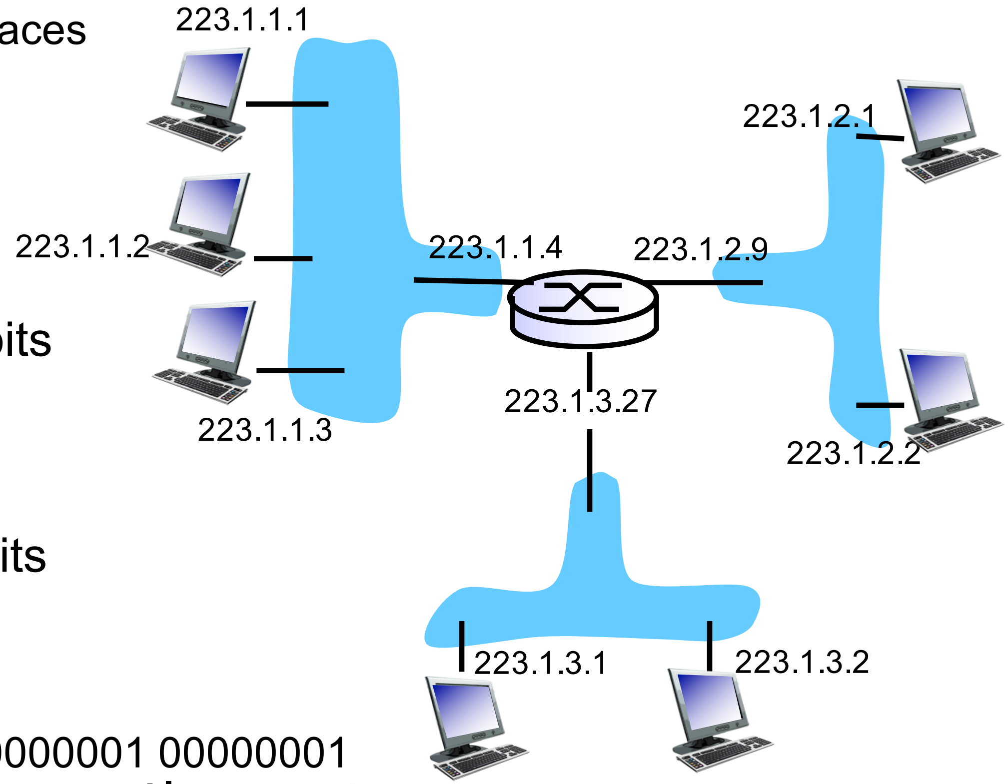
Addressing

IPV4 ADDRESSES

IPv4 addresses

Globally unique 32-bit identifier

- associated with host or router interface
- Interface: connection between host/router and physical link
  - host: usually 1 or 2 interfaces
  - router: usually many interfaces



Hierarchical

- subnet part
  - variable len# of high order bits
  - assigned by ICANN
- host part
  - variable len # of low order bits
  - network + host

223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Hierarchical addresses

Pros

- scalable: routers don't need to look at host part
- all pkts on same network forwarded in same direction
  - only when pkt reaches network does host matter

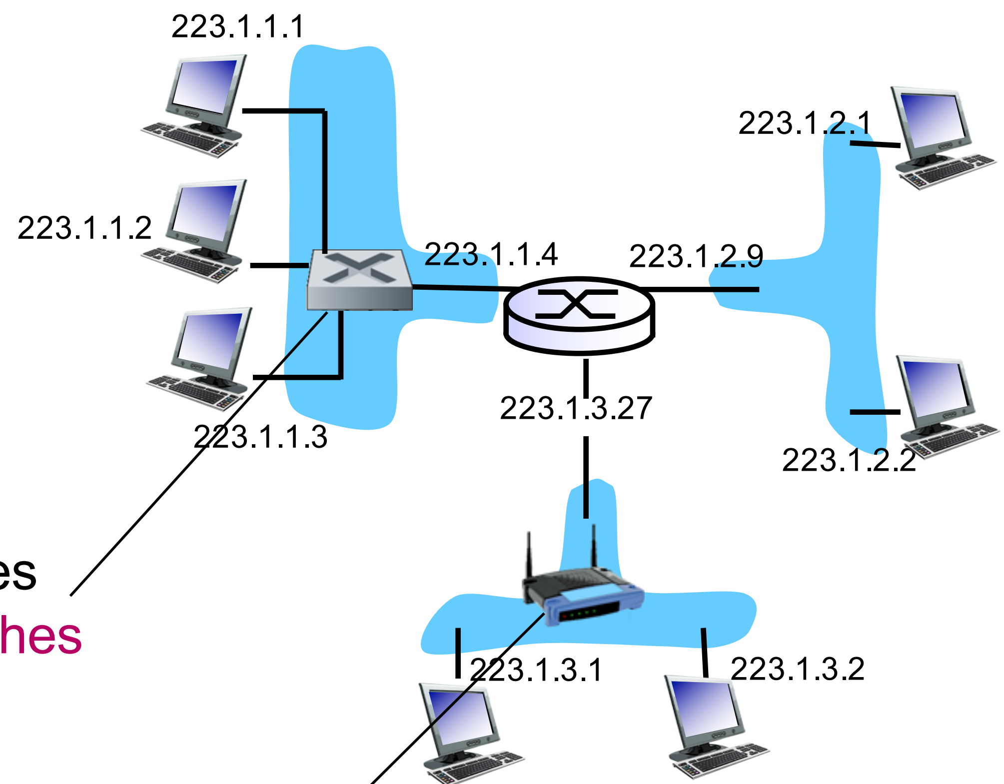
Cons

- every IP addr belongs to specific network … but what if host moves networks and wants to keep same addr?
  - mobile IP
  - contrast with fixed Ethernet link layer addr

IPv4 addresses

Q: how are interfaces actually connected?

A: see Ch 5, 6.



A: wired Ethernet interfaces connected by Ethernet switches

A: wireless WiFi interfaces connected by WiFi base station

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

Subnets

IP address

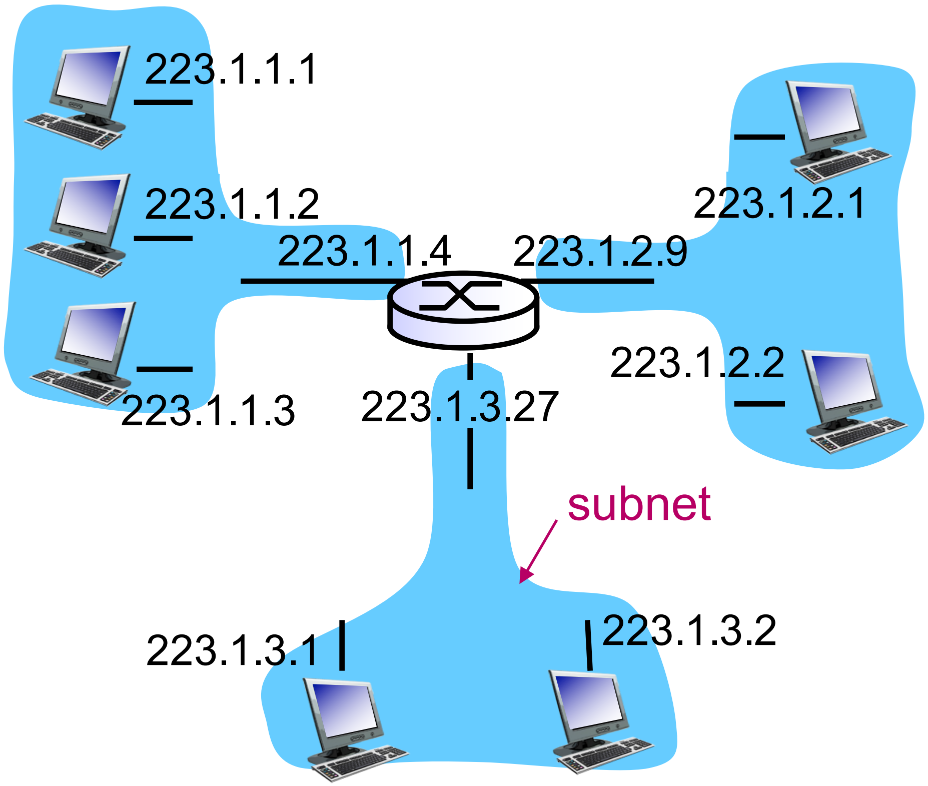
- subnet part: high order bits
- host part: low order bits

What's a subnet?

- set of interfaces that all have same subnet part of IP addr
- devices can reach other without intervening routers

Subnet mask

- divides IP addr into subnet addr + host addr
- included in routing protocol info given to routers



Network comprising 3 subnets

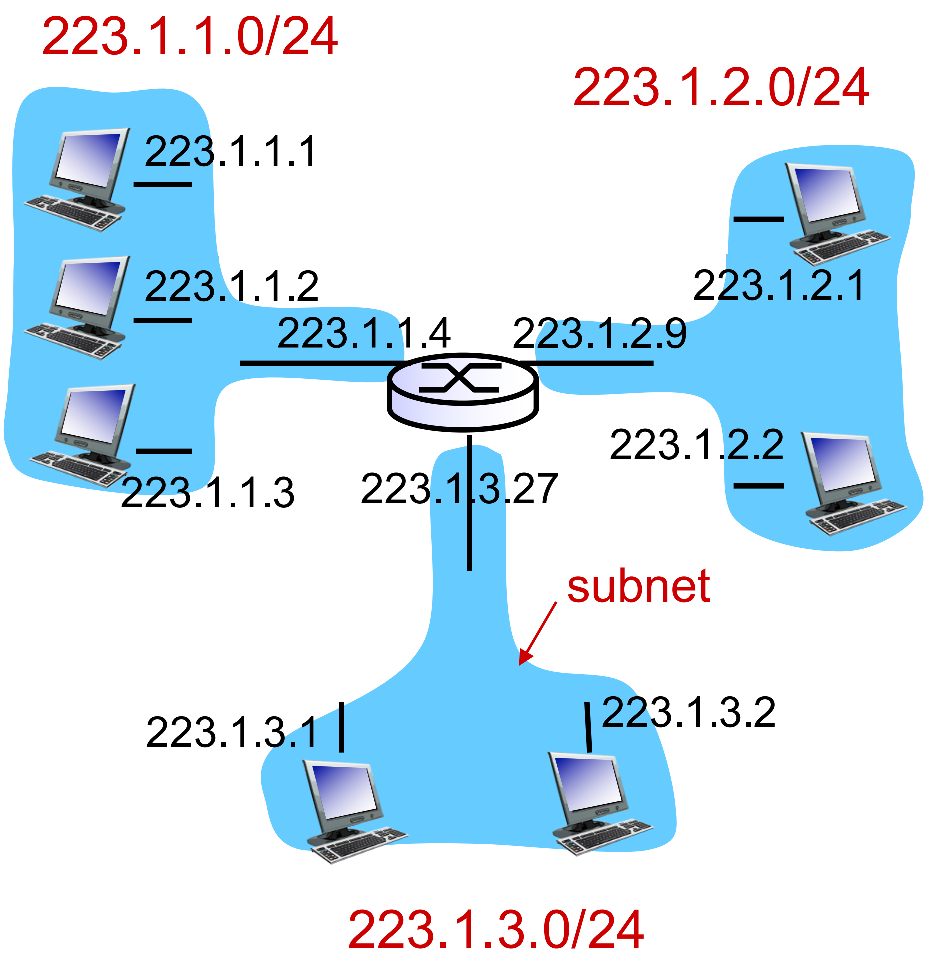
Subnets

Recipe to find subnets

- detach each interface from its host or router
- create islands of isolated networks

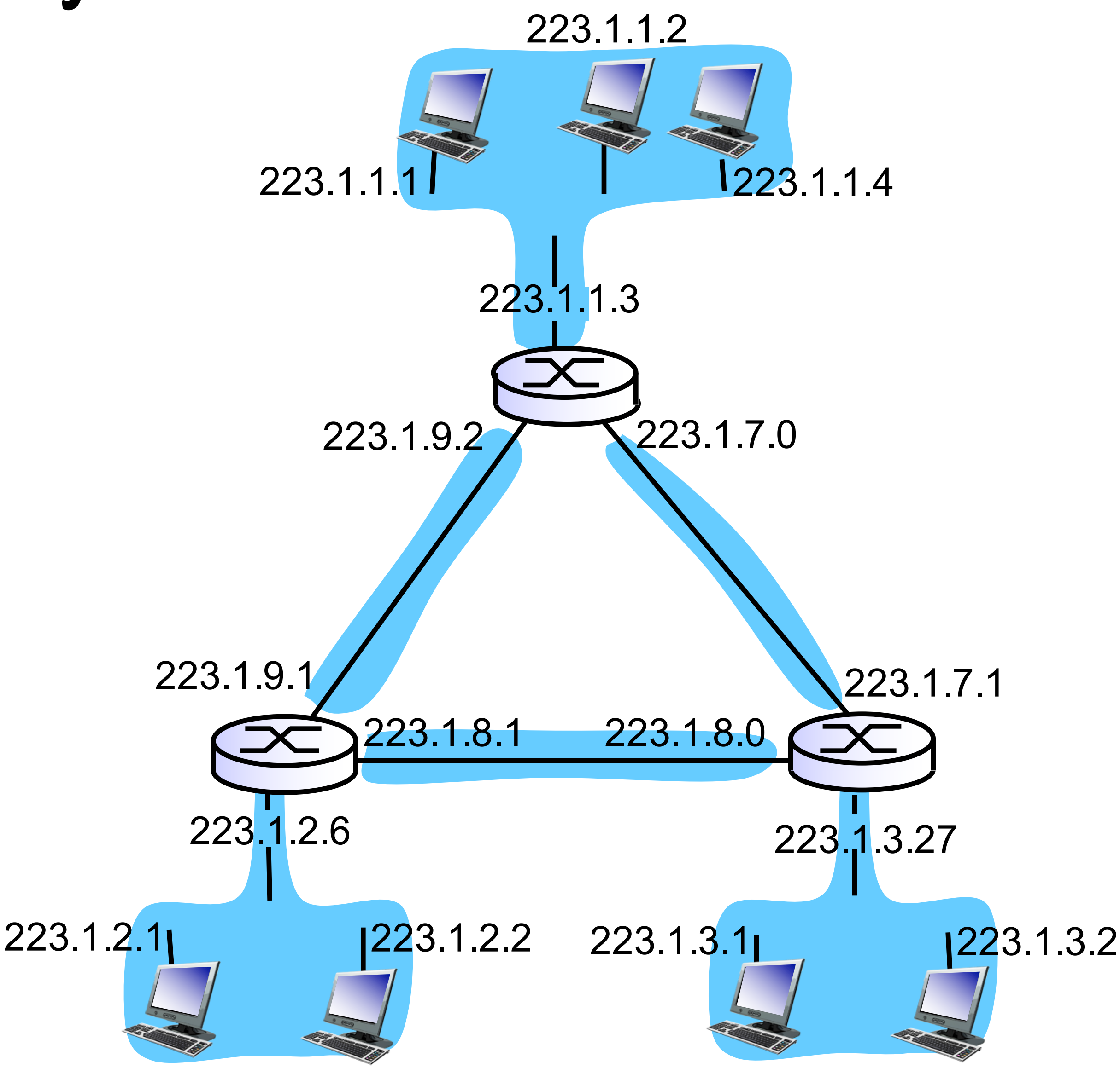
Each isolated network

- is called a subnet



subnet mask: /24

How many subnets?



CIDR: Classless InterDomain Routing

Allows more fine-grained division of blocks of addresses

Subnet masks

- define variable partition of host part of addresses
- e.g.,
  - /23 subnet mask: 11111111 11111111 11111110 00000000
  - logical “and” of subnet mask with addr
    - if a and b are both 1 then 1 otherwise 0

Address format

- a.b.c.d/x, where x is # bits in subnet portion of address

subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Subnet masks and address blocks

Suppose

- we must have 223.1.1 as network prefix
- we need block of 90 addresses

What should subnet mask be?

- how many bits for 90 addresses?

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

223.1.1.0/24 gives 256 addresses [0-255]

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	64	32	16	8	4	2	1

223.1.1.0/25 gives 128 addresses [0-127]

	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	64	32	16	8	4	2	1

223.1.1.128/25 gives a different set of 128 addresses [128-255]

Special addresses

Private subnet (used in NAT), do not appear on Internet

- 172.16-31.*.*
- 10.*.*.*
- 192.168.*.*

Loopback address:

- 127.*.*.*

Addresses you can’t assign to devices

- *.*.*.255: broadcast addr
- *.*.*.0: used for subnet name

Broadcast address

- 255.255.255.255: broadcast to all hosts on network indicated
  - if no mask: local network
  - if mask: broadcast on that network

Address when device booting up

- 0.0.0.0