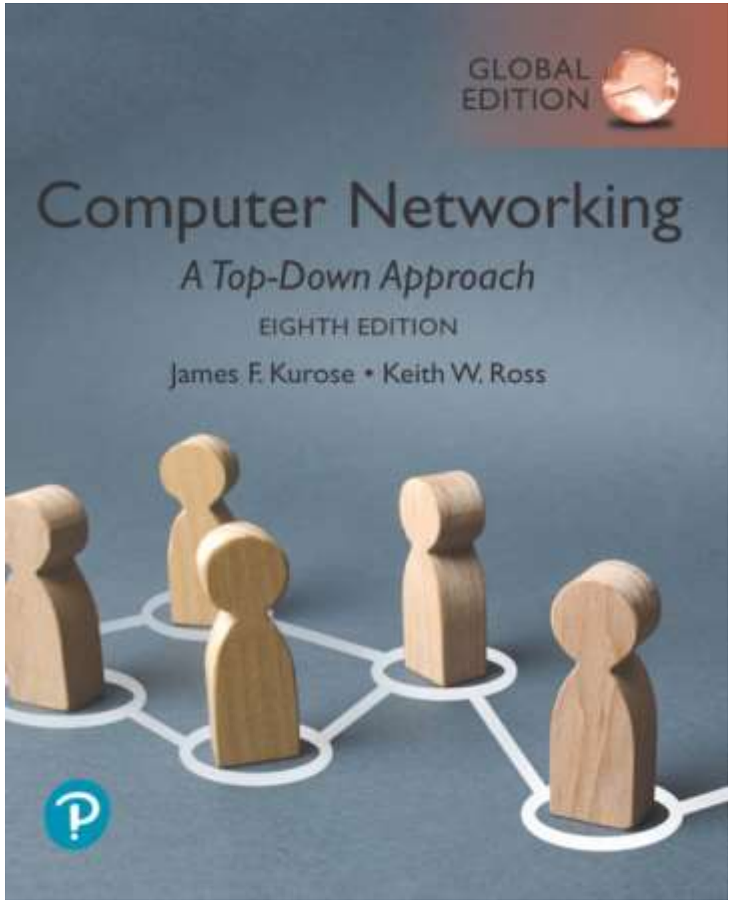
Chapter 4
Network Layer: Data Plane

Ren Ping Liu
renping.liu@uts.edu.au
adapted from textbook slides by JFK/KWR

1 April 2024



Computer Networking: A Top-Down Approach
8th Edition, Global Edition
Jim Kurose, Keith Ross

Network layer: “data plane” roadmap

4.1 Network layer: overview

- data plane
- control plane

4.2 What’s inside a router

- input ports, switching, output ports
- buffer management, scheduling

4.3 IP: the Internet Protocol

- datagram format
- addressing
- network address translation
- IPv6

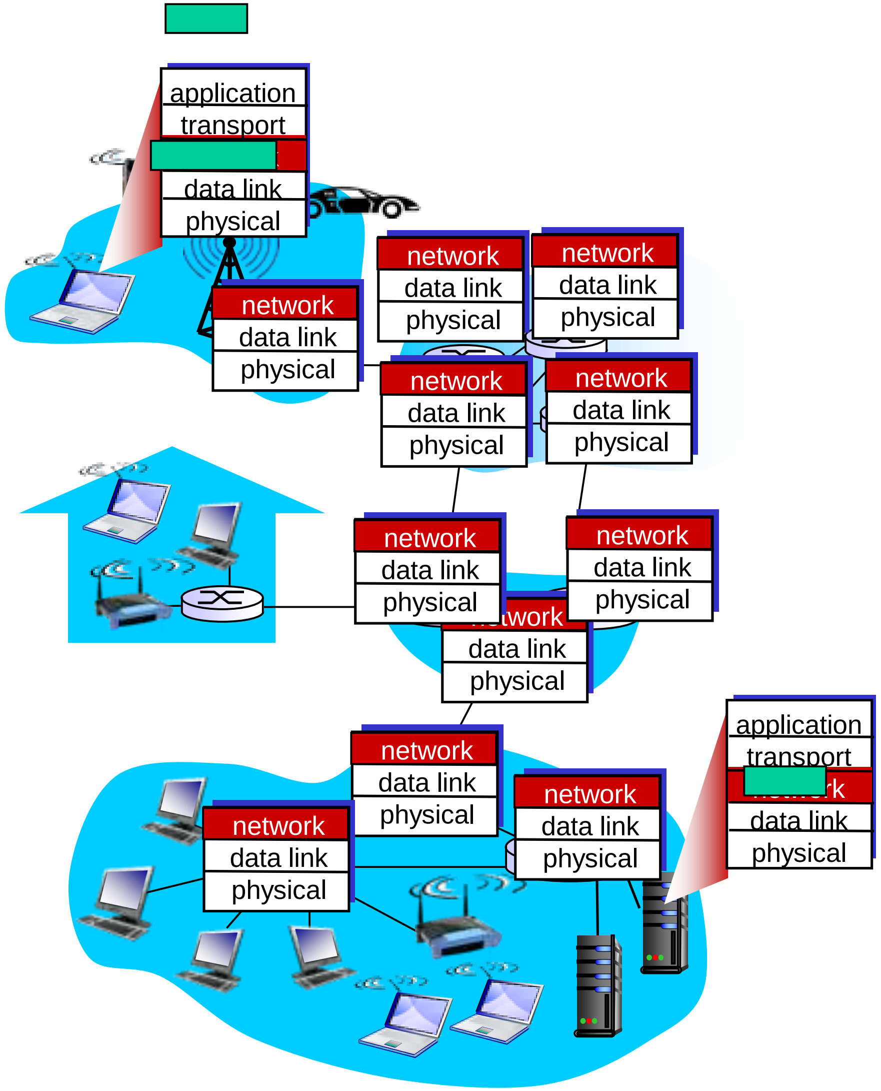
~~4.4 Generalized Forwarding, SDN~~

- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

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

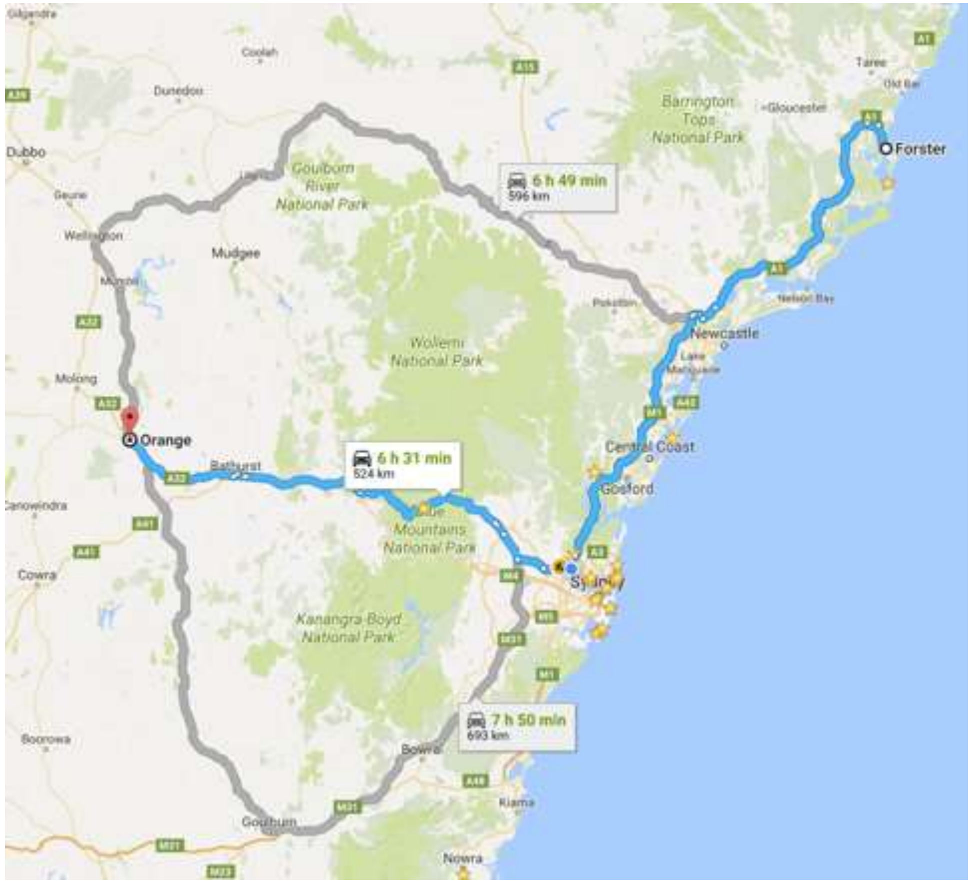


Two key network-layer functions

network-layer functions:

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

Trip: Forster → Orange



Network layer: data plane, control plane

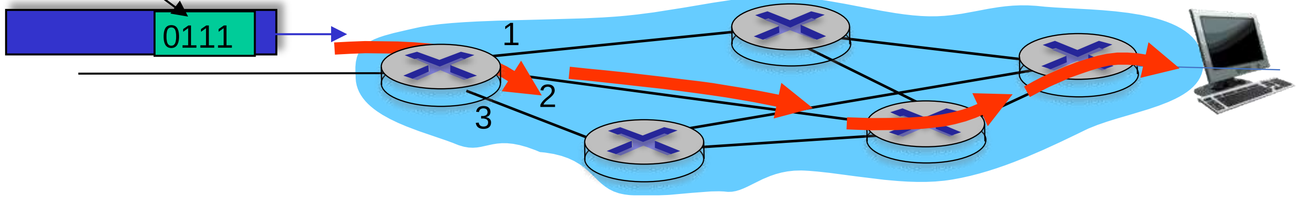
Control plane

- determines how datagram is routed among routers along end-end path from source host to destination host
- network-wide logic
- Based on network topology

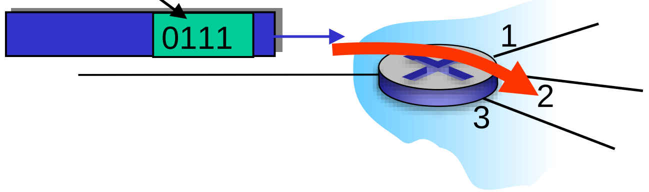
Data plane

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

values in arriving packet header



values in arriving packet header



Network layer: data plane, control plane

Control plane

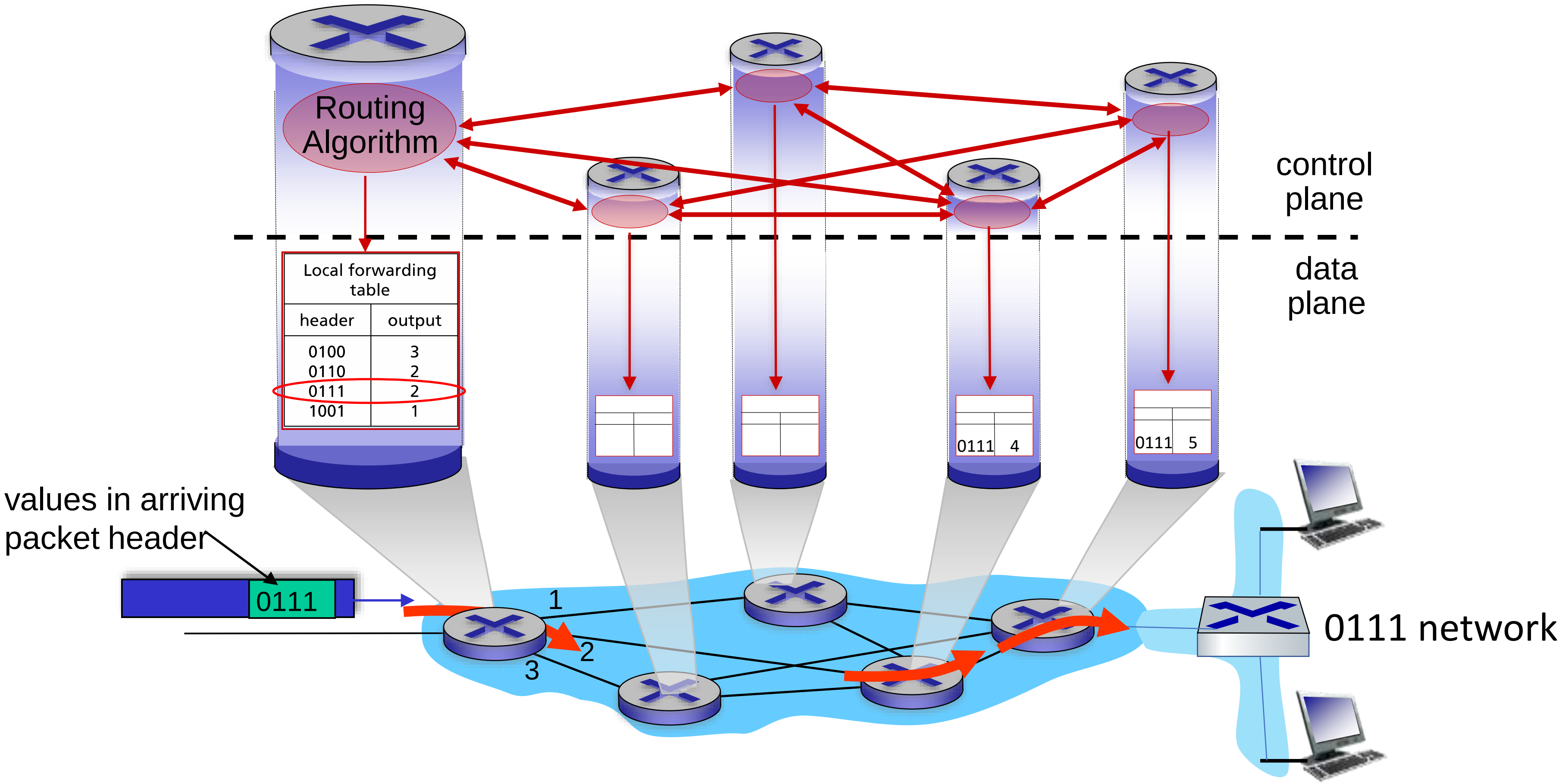
- determines how datagram is routed among routers along end-end path from source host to destination host
- network-wide logic
- Based on network topology

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

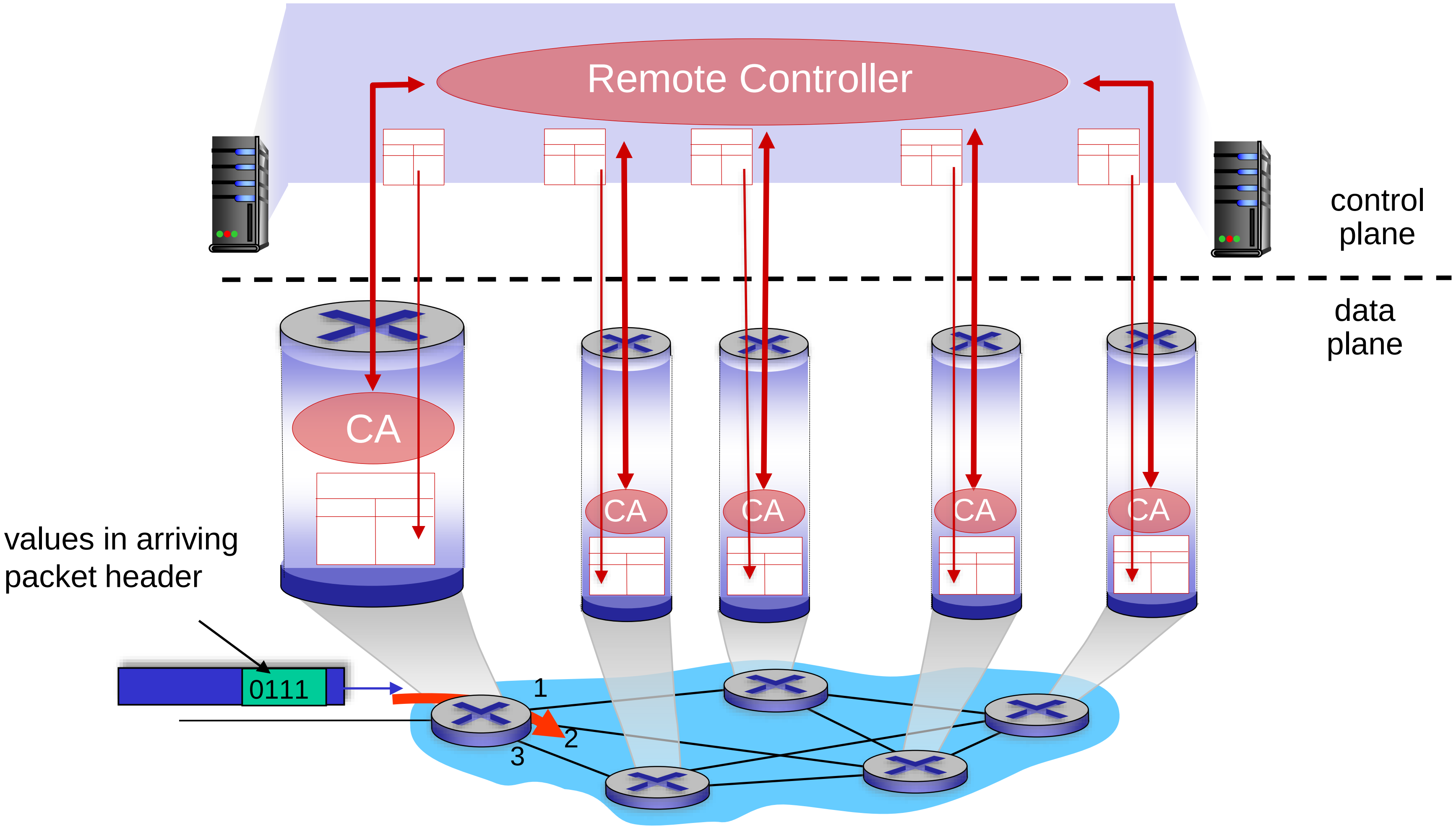


Local forwarding table

header	output
0100	3
0110	2
0111	2
1001	1

Software-Defined Networking (SDN) control plane

Remote controller computes, installs forwarding tables in routers



Network service model

think of services by Aust Post: letter, registered, express, parcel,

Q: What *service model* do we need for “network” to pass datagrams from sender to receiver?

example services for *individual* datagrams:

- guaranteed delivery
- guaranteed timing, e.g., delivery within 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 model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no

Internet “best effort” service model

No guarantees on*:*

- ✗ successful datagram delivery to destination
- ✗ timing or order of delivery
- ✗ bandwidth available to end-end flow

Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no

QoS: too hard, too expensive!

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” roadmap

4.1 Network layer: overview

- data plane
- control plane

4.2 What’s inside a router

- input ports, switching, output ports
- buffer management, scheduling

4.3 IP: the Internet Protocol

- datagram format
- addressing
- network address translation
- IPv6

~~4.4 Generalized Forwarding, SDN~~

- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

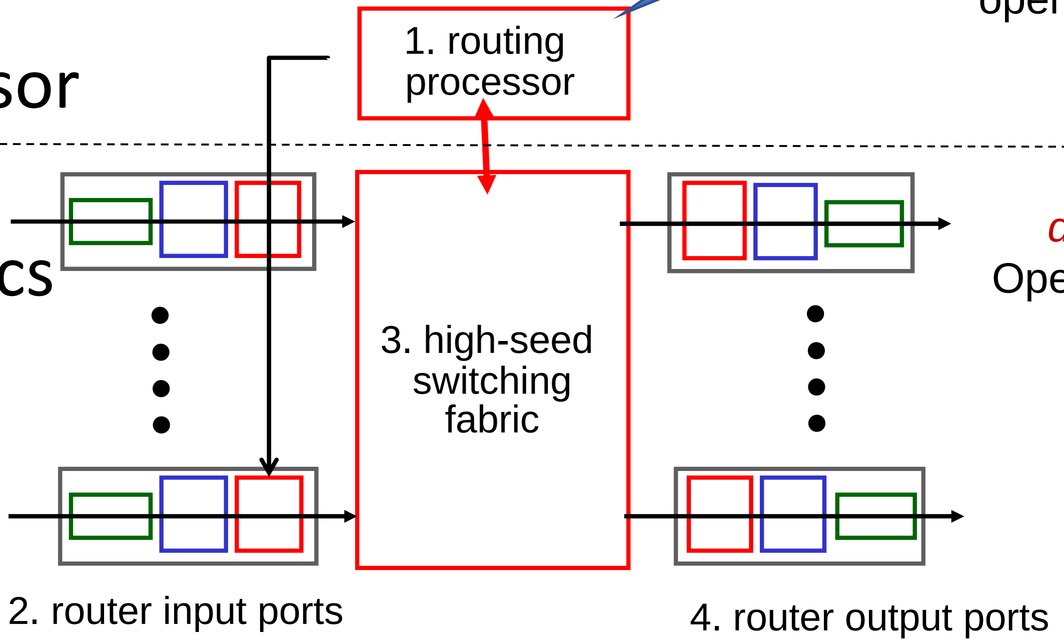
Router architecture overview

A specialised computer

- Four parts:

Software
1. routing processor

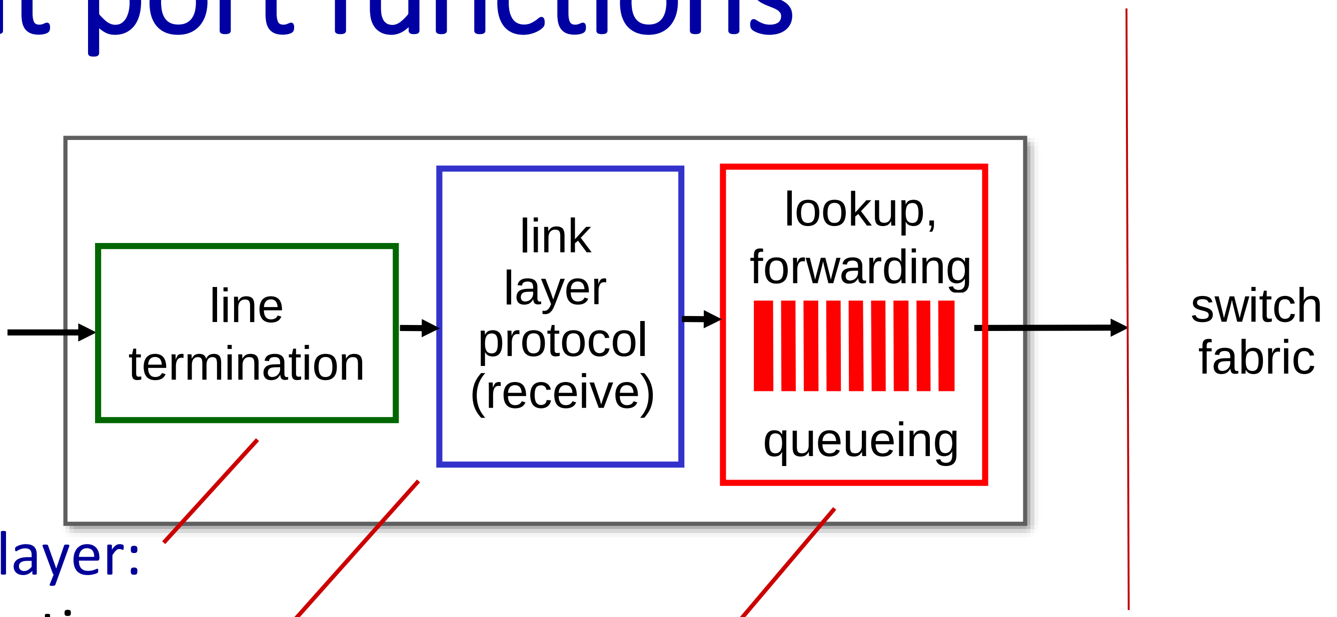
Hardware
2. Input ports
3. Switching fabrics
4. Output ports

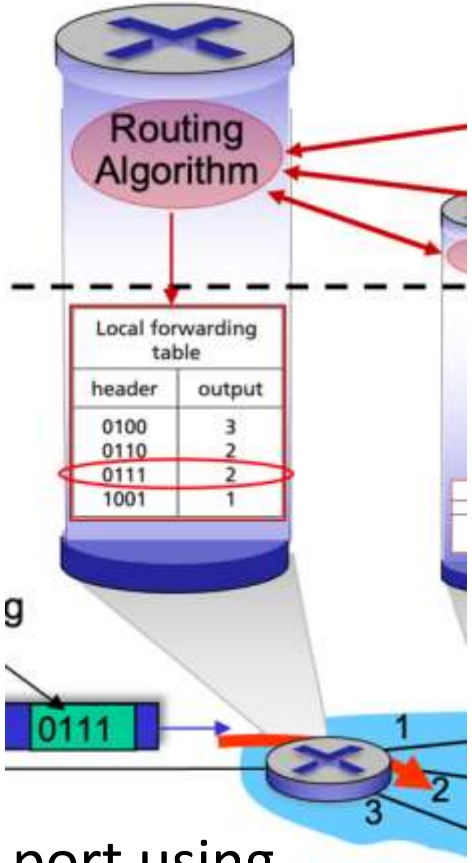


control plane (software)
operates in **ms** time frame
routing, management

data plane (hardware)
Operates in **ns** timeframe
forwarding datagram

Input port functions





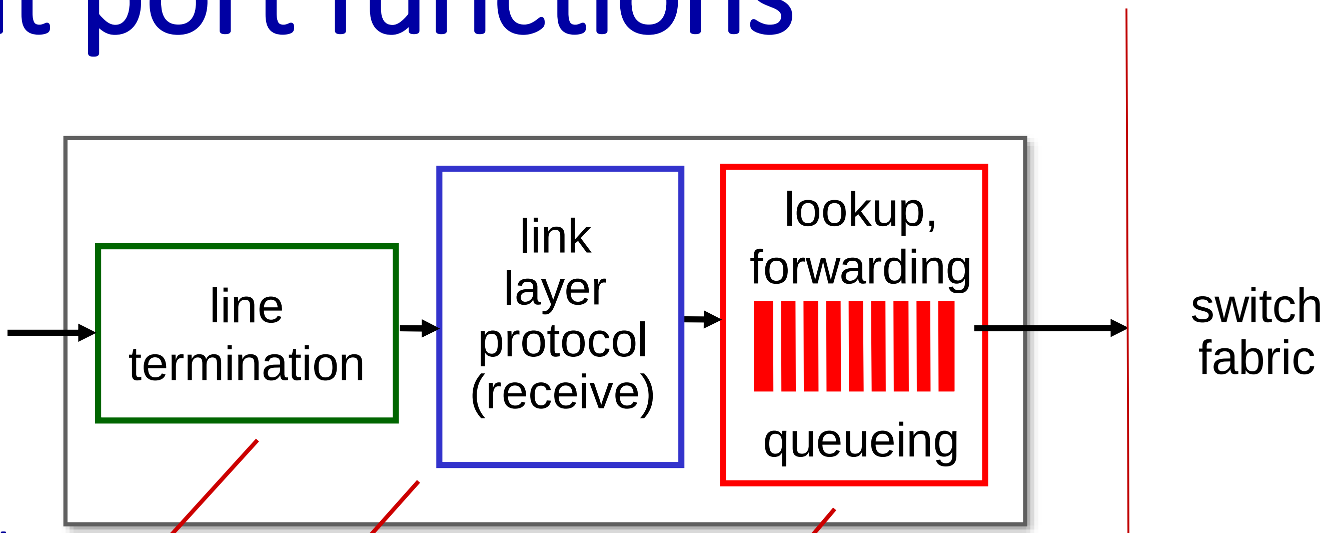
physical layer:
bit-level reception

link layer:
e.g., Ethernet
(chapter 6)

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’
- input port queuing: if datagrams arrive faster than forwarding rate into switch fabric

Input port functions



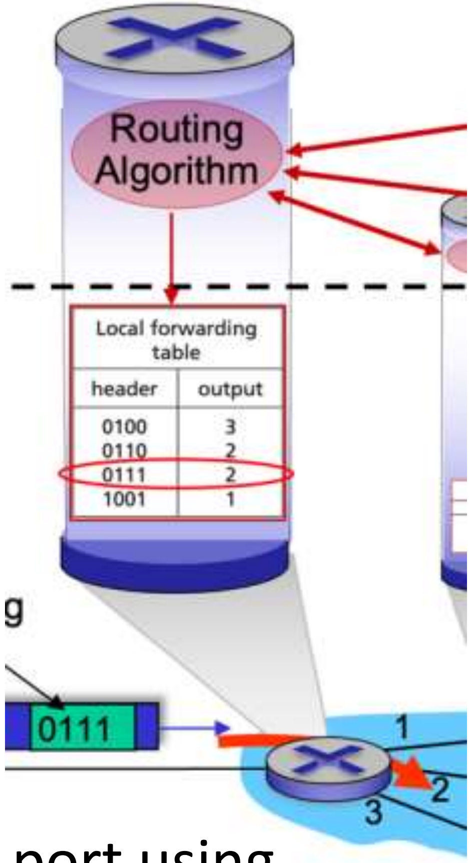
line termination

link layer protocol (receive)

lookup, forwarding

queueing

switch fabric



physical layer:
bit-level reception

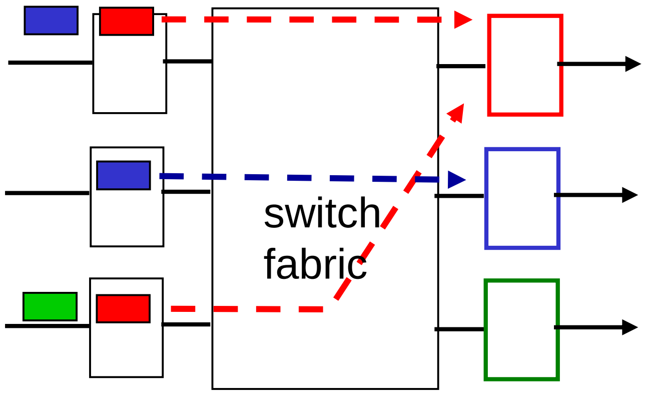
link layer:
e.g., Ethernet
(chapter 6)

decentralized switching:

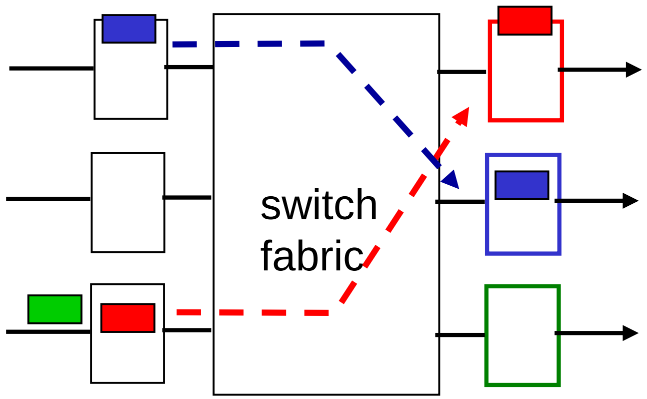
- using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- destination-based forwarding: forward based only on destination IP address (traditional)
- ~~generalized forwarding: forward based on any set of header field values~~

Input port queuing

- If switch 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



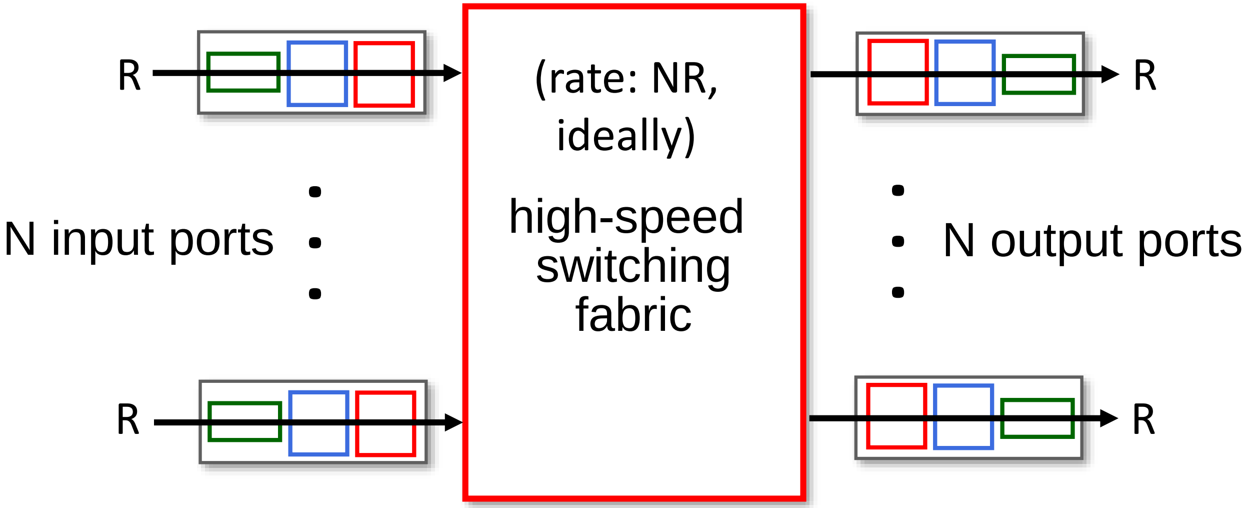
output port contention: only one red datagram can be transferred. lower red packet is *blocked*



one packet time later: green packet experiences HOL blocking

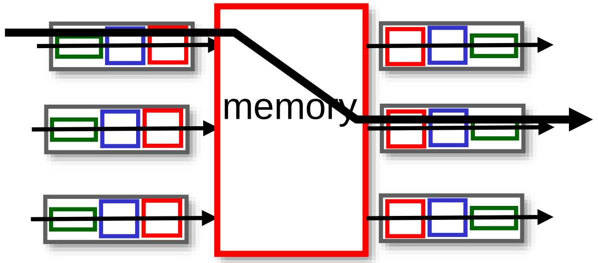
Switching fabrics

- transfer packet from input link to appropriate output link
- 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

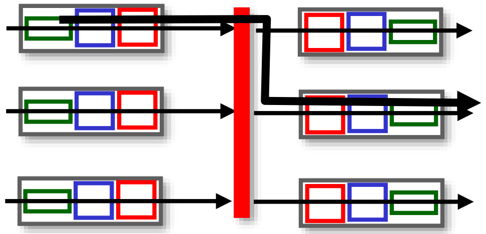


Switching fabrics

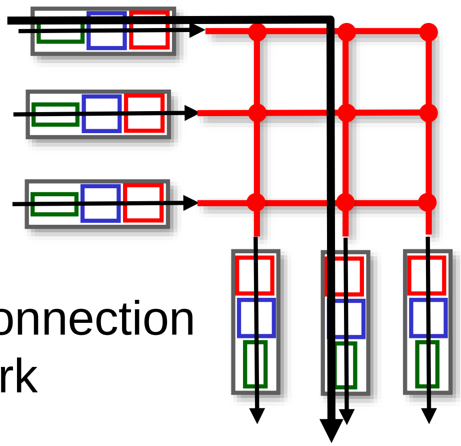
- transfer packet from input link to appropriate output link
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three major types of switching fabrics:



memory



bus

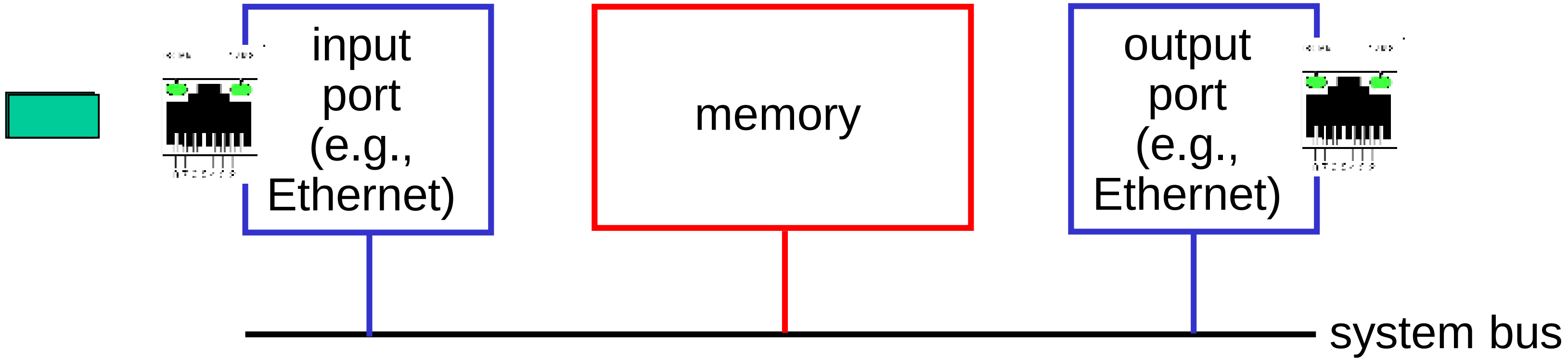


interconnection network

Switching via memory

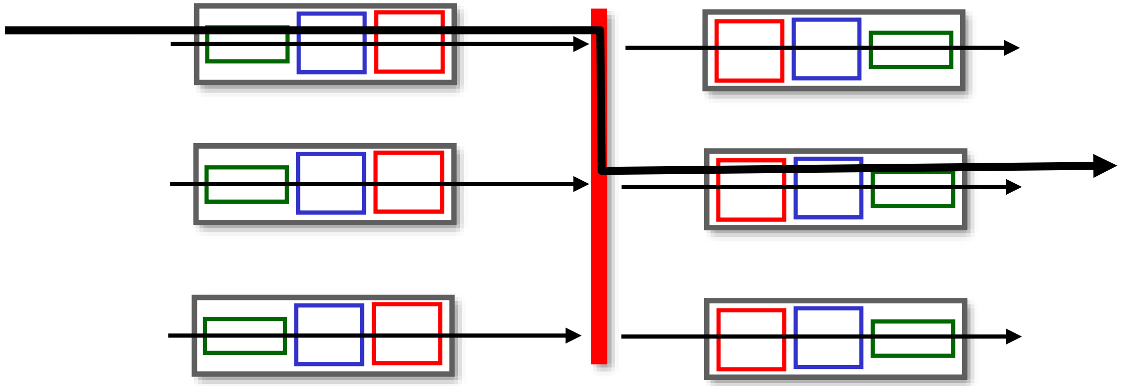
first generation routers:

- 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)



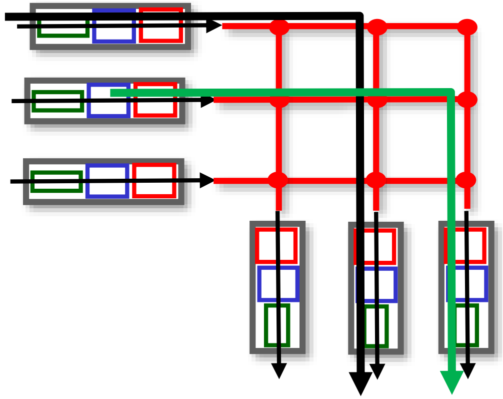
Switching via a bus

- 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 routers

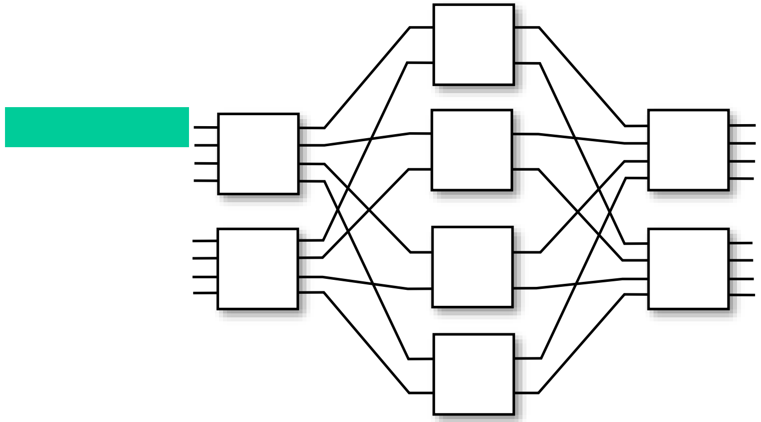


Switching via interconnection network

- Crossbar, Clos networks, other interconnection nets initially developed to connect processors in multiprocessor
- multistage switch: *nxn* switch from multiple stages of smaller switches
- exploiting parallelism:
  - fragment datagram into fixed length cells on entry
  - switch cells through the fabric, reassemble datagram at exit

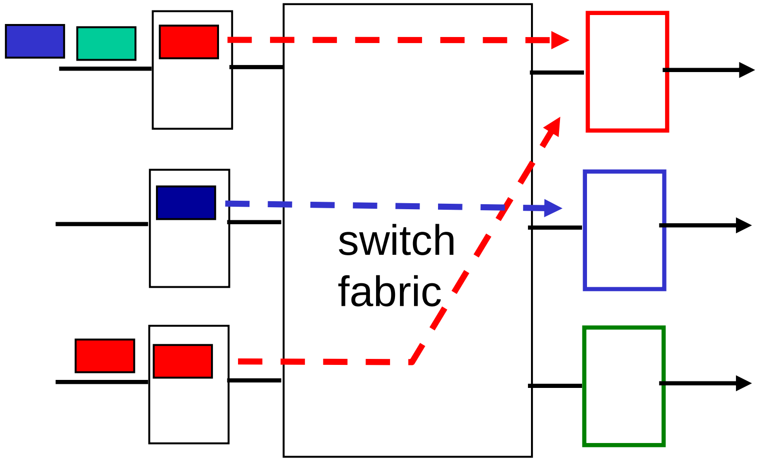


3x3 crossbar

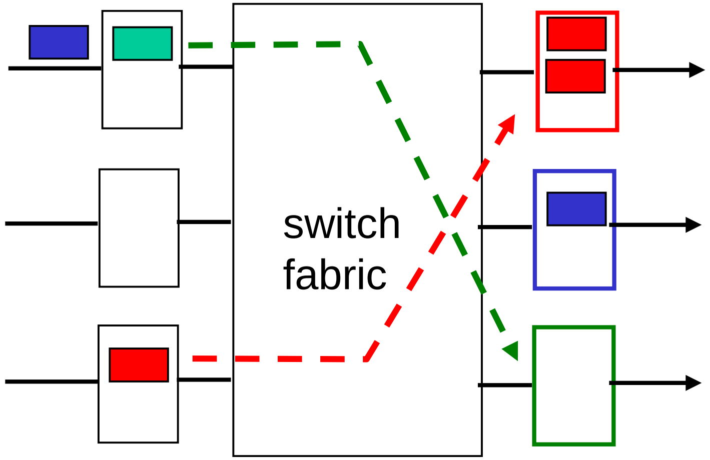


8x8 multistage switch
built from smaller-sized switches

Output port queuing



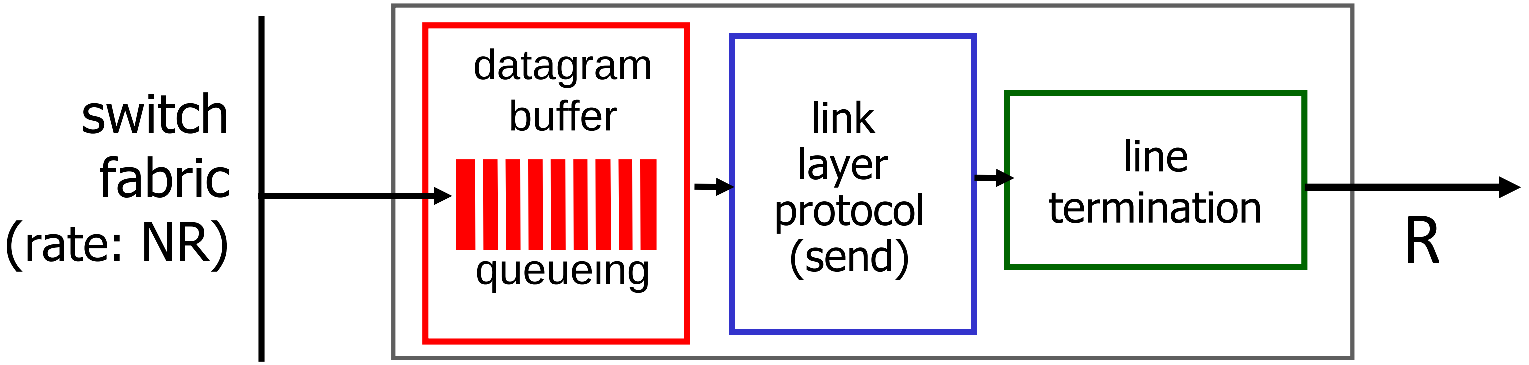
at *t*, packets more
from input to output



one packet time later

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

Output port queuing



- *Buffering* required when datagrams arrive from fabric faster than link transmission rate.
- *Scheduling discipline* chooses among queued datagrams for transmission
- *Drop policy:* which datagrams to drop if no free buffers?

Priority scheduling – who gets best performance, network neutrality

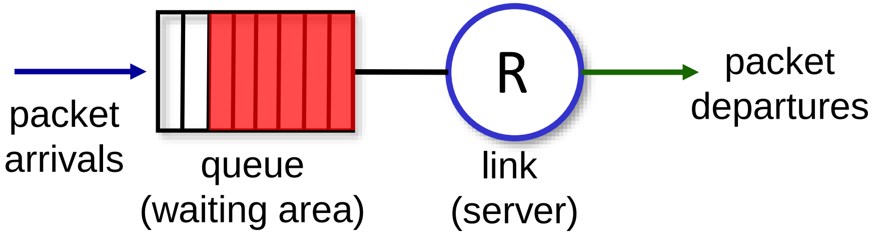
Datagrams can be lost due to congestion, lack of buffers

Packet Scheduling: FCFS

packet scheduling: deciding which packet to send next on link

- first come, first served
- priority
- round robin
- weighted fair queueing

Abstraction: queue



FCFS: packets transmitted in order of arrival to output port

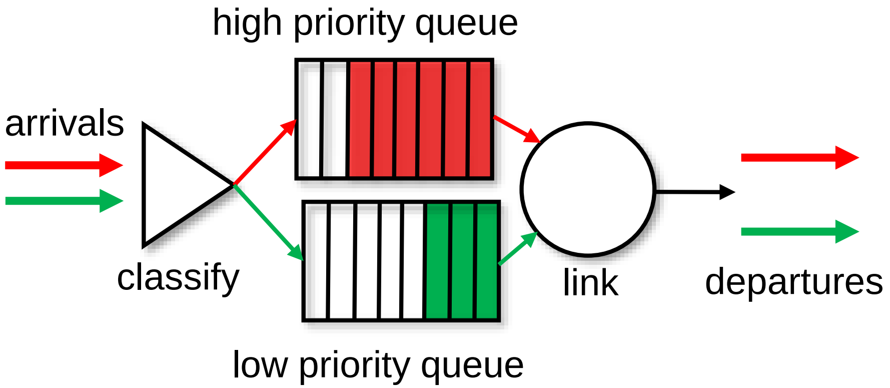
- also known as: First-in-first-out (FIFO)
- real world examples?

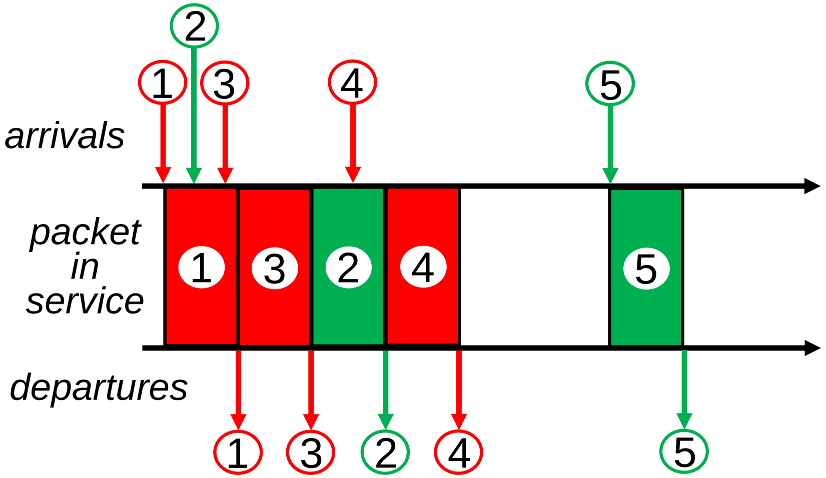
Scheduling policies: priority

Priority scheduling:

- arriving traffic classified, queued by class
  - any header fields can be used for classification
- send packet from highest priority queue that has buffered packets
  - FCFS within priority class

Real life example: Airline check-in

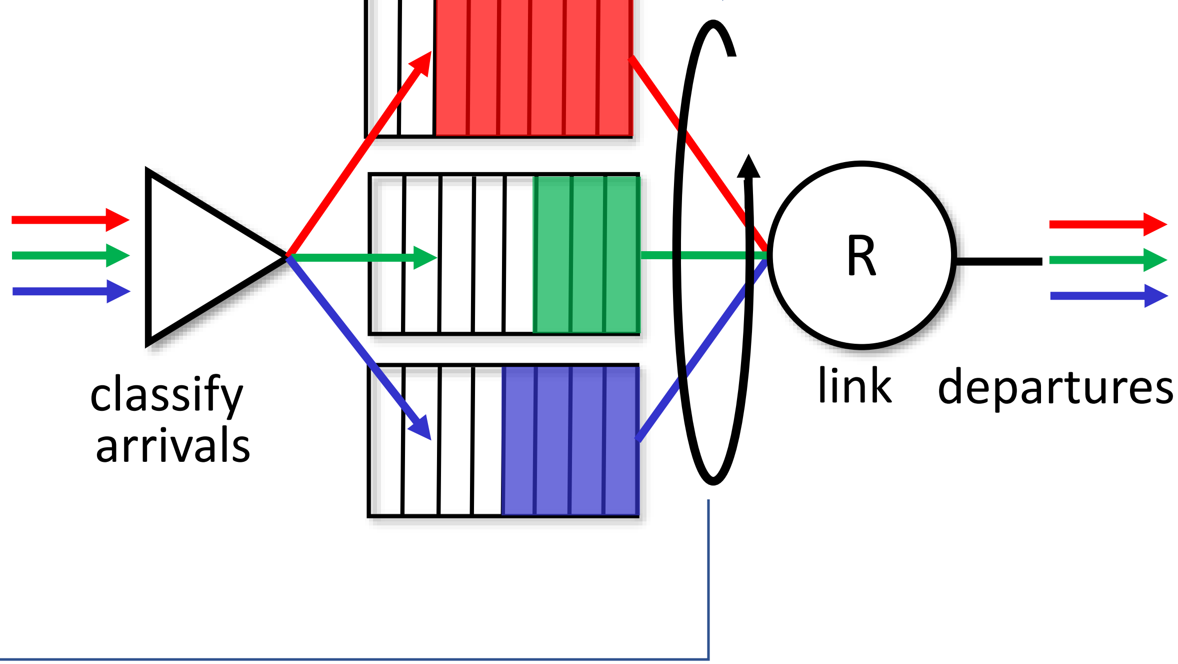




Scheduling policies: round robin

Round Robin (RR) scheduling:

- arriving traffic classified, queued by class
  - any header fields can be used for classification
- server cyclically, repeatedly scans class queues, sending one complete packet from each class (if available) in turn



Real life example: RMS, Medicare, … true round robin, or?

Scheduling policies: weighted fair queueing

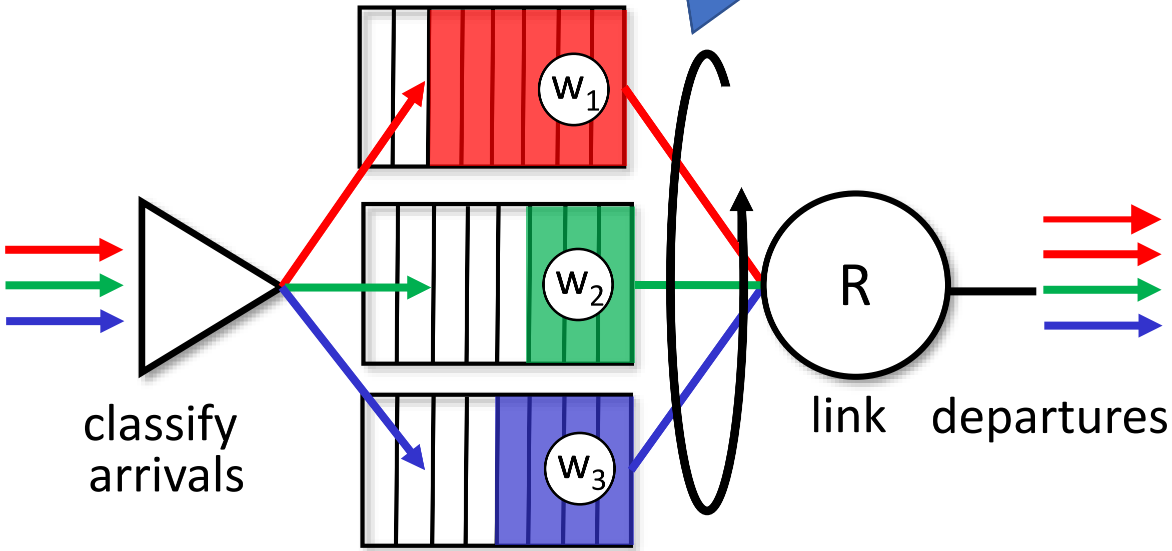
Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class, *i*, has weight, w_i, and gets weighted amount of service in each cycle:

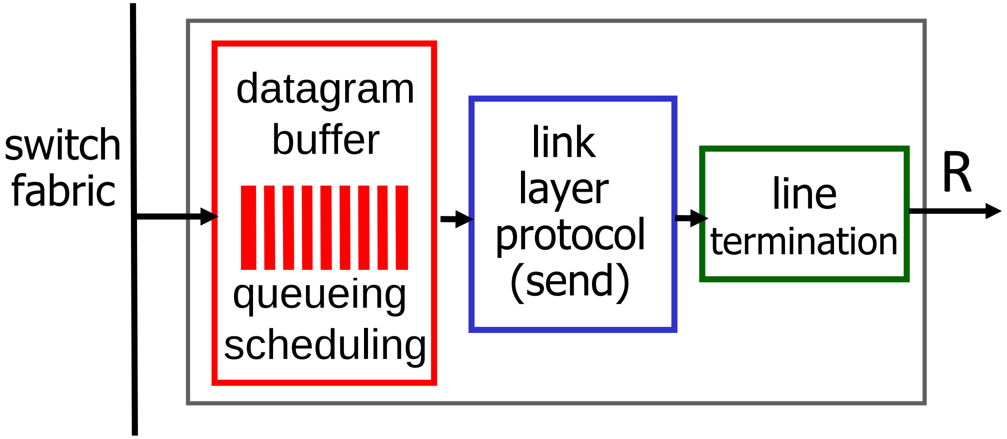
$$\frac{w_i}{\Sigma_j w_j}$$

- minimum bandwidth guarantee (per-traffic-class)

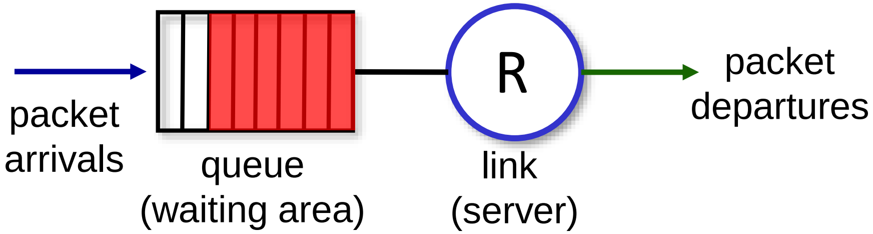
Real life example:
RMS, Medicare, …
true round robin, or WFQ



Buffer Management



Abstraction: queue



buffer management:

- drop: which packet to add, drop when buffers are full
  - tail drop: drop arriving packet
  - priority: drop/remove on priority basis
- marking: which packets to mark to signal congestion (ECN, RED)

Sidebar: Network Neutrality

What is network neutrality?

- *technical:* how an ISP should share/allocation its resources
  - packet scheduling, buffer management are the *mechanisms*
- *social, economic* principles
  - protecting free speech
  - encouraging innovation, competition
- enforced *legal* rules and policies

Optional – not tested

Different countries have different "takes" on network neutrality

Sidebar: Network Neutrality

2015 US FCC *Order on Protecting and Promoting an Open Internet:* three "clear, bright line" rules:

- no blocking … "shall not block lawful content, applications, services, or non-harmful devices, subject to reasonable network management."
- no throttling … "shall not impair or degrade lawful Internet traffic on the basis of Internet content, application, or service, or use of a non-harmful device, subject to reasonable network management."
- no paid prioritization. … "shall not engage in paid prioritization"

Optional – not tested

ISP: telecommunications or information service?

Is an ISP a “telecommunications service” or an “information service” provider?

- the answer *really* matters from a regulatory standpoint!

US Telecommunication Act of 1934 and 1996:

- *Title II:* imposes “common carrier duties” on *telecommunications services*: reasonable rates, non-discrimination and *requires regulation*
- *Title I:* applies to *information services:*
  - no common carrier duties (*not regulated*)
  - but grants FCC authority “… as may be necessary in the execution of its functions”.[4]

Optional – not tested

Mid-break

☕□

- Q & A

Network layer: "data plane" roadmap

4.1 Network layer: overview
- data plane
- control plane

4.2 What's inside a router
- input ports, switching, output ports
- buffer management, scheduling

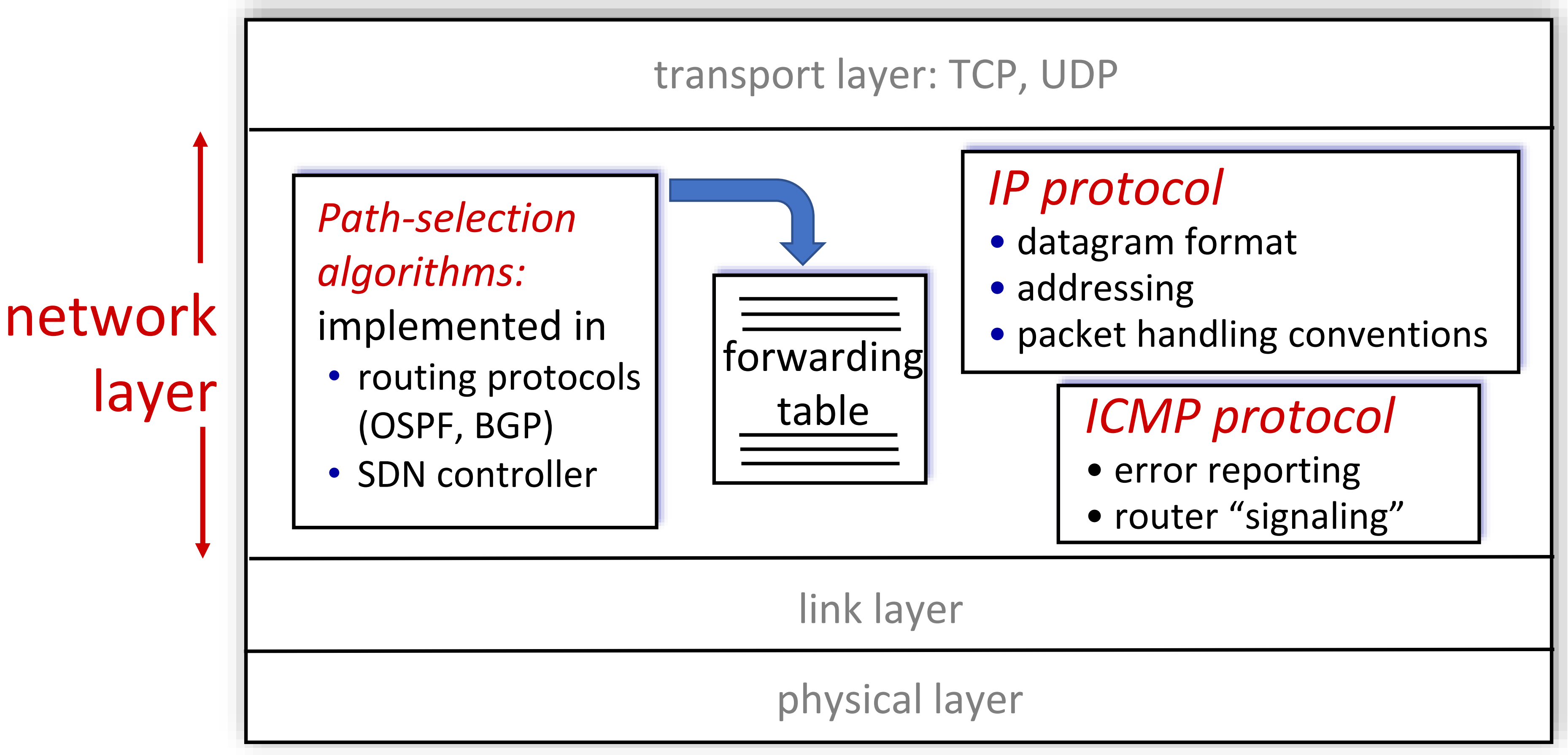
4.3 IP: the Internet Protocol
- datagram format
- addressing
- network address translation
- IPv6

~~4.4 Generalized Forwarding, SDN~~
- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

Network Layer: Internet

host, router network layer functions:



transport layer: TCP, UDP

network layer

Path-selection algorithms:
implemented in

- routing protocols (OSPF, BGP)
- SDN controller

forwarding table

IP protocol

- datagram format
- addressing
- packet handling conventions

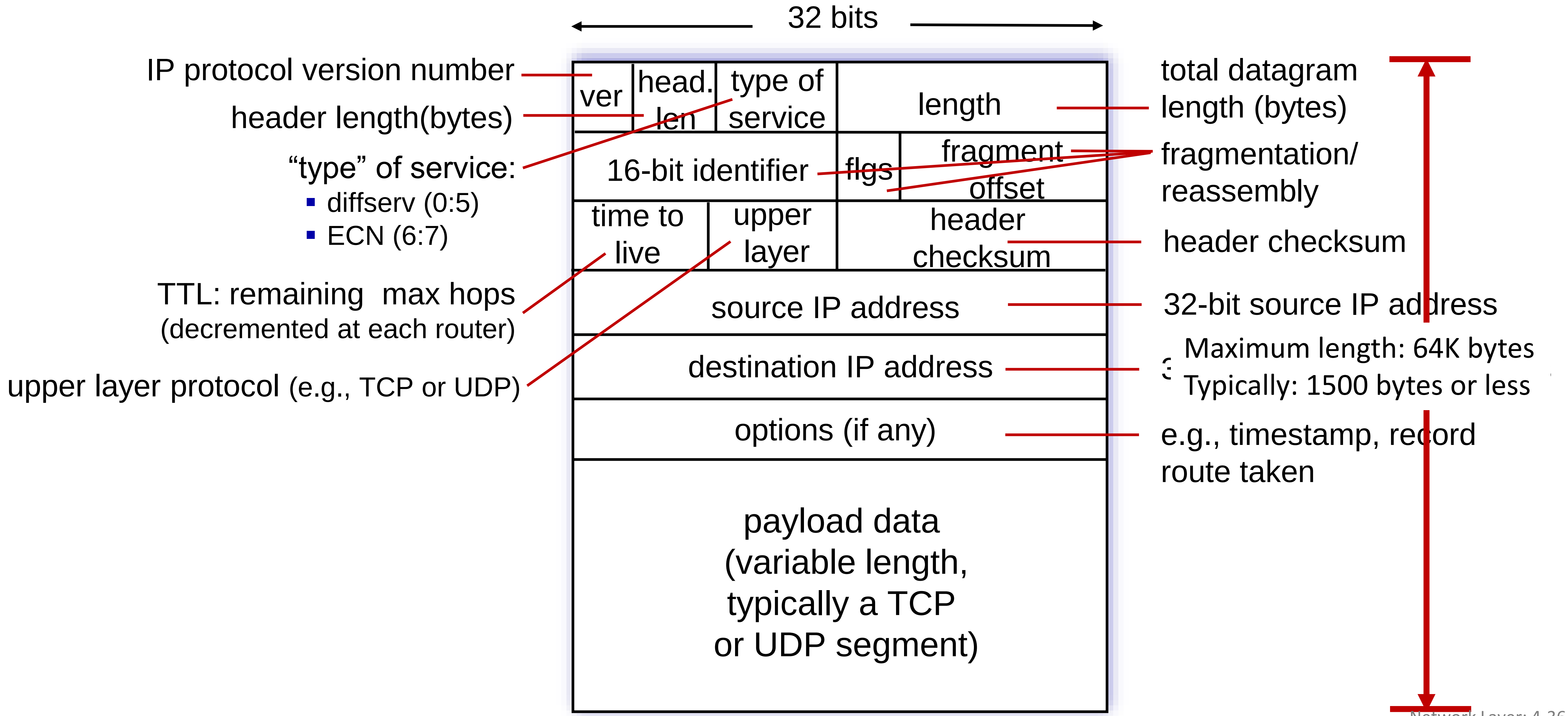
ICMP protocol

- error reporting
- router “signaling”

link layer

physical layer

IP Datagram format



32 bits

ver

head. len

type of service

length

16-bit identifier

flgs

fragment offset

time to live

upper layer

header checksum

source IP address

destination IP address

options (if any)

payload data (variable length, typically a TCP or UDP segment)

IP protocol version number

header length(bytes)

“type” of service:

- diffserv (0:5)
- ECN (6:7)

TTL: remaining max hops (decremented at each router)

upper layer protocol (e.g., TCP or UDP)

total datagram length (bytes)

fragmentation/ reassembly

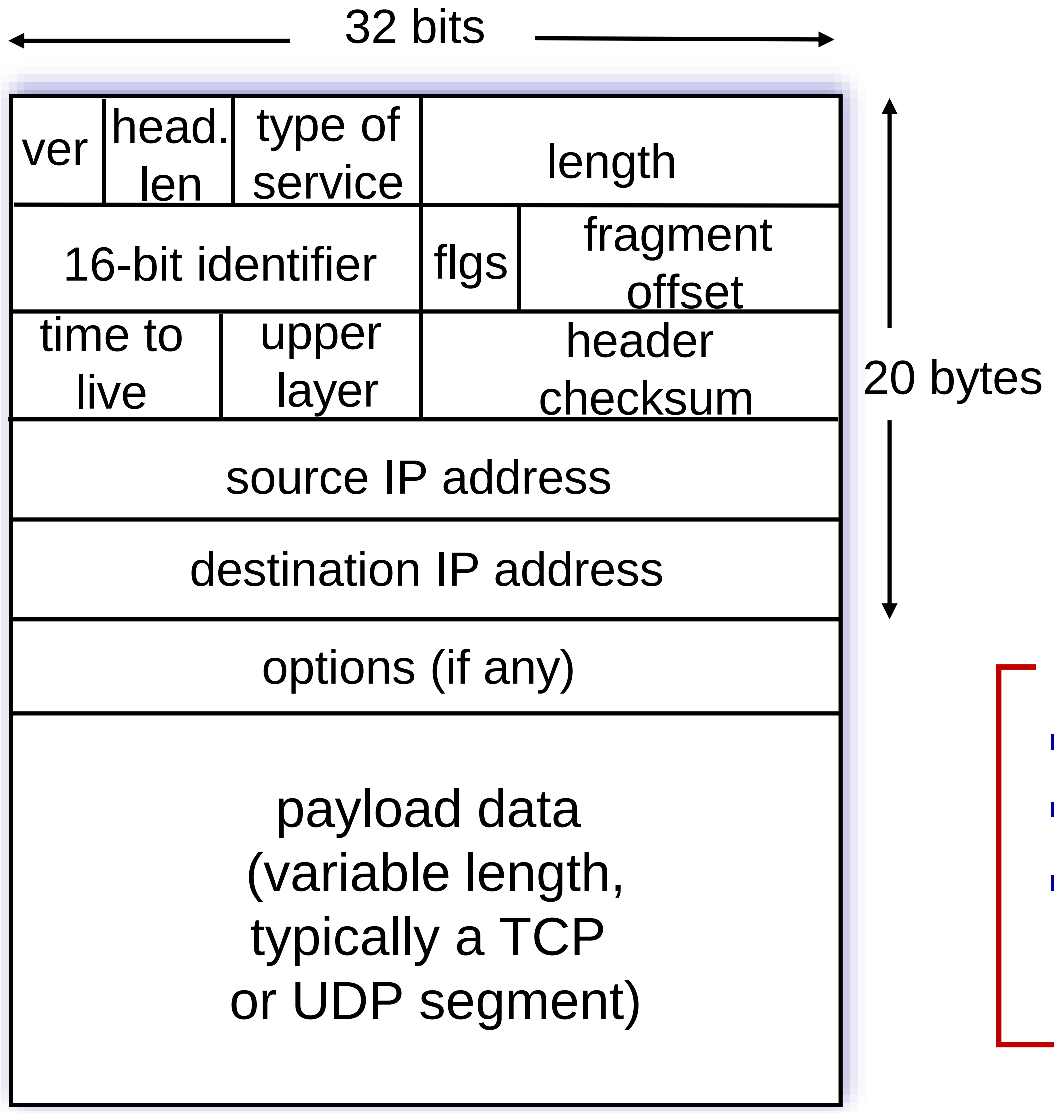
header checksum

32-bit source IP address

Maximum length: 64K bytes
Typically: 1500 bytes or less

e.g., timestamp, record route taken

IP Datagram format



32 bits

ver

head. len

type of service

length

16-bit identifier

flgs

fragment offset

time to live

upper layer

header checksum

source IP address

destination IP address

options (if any)

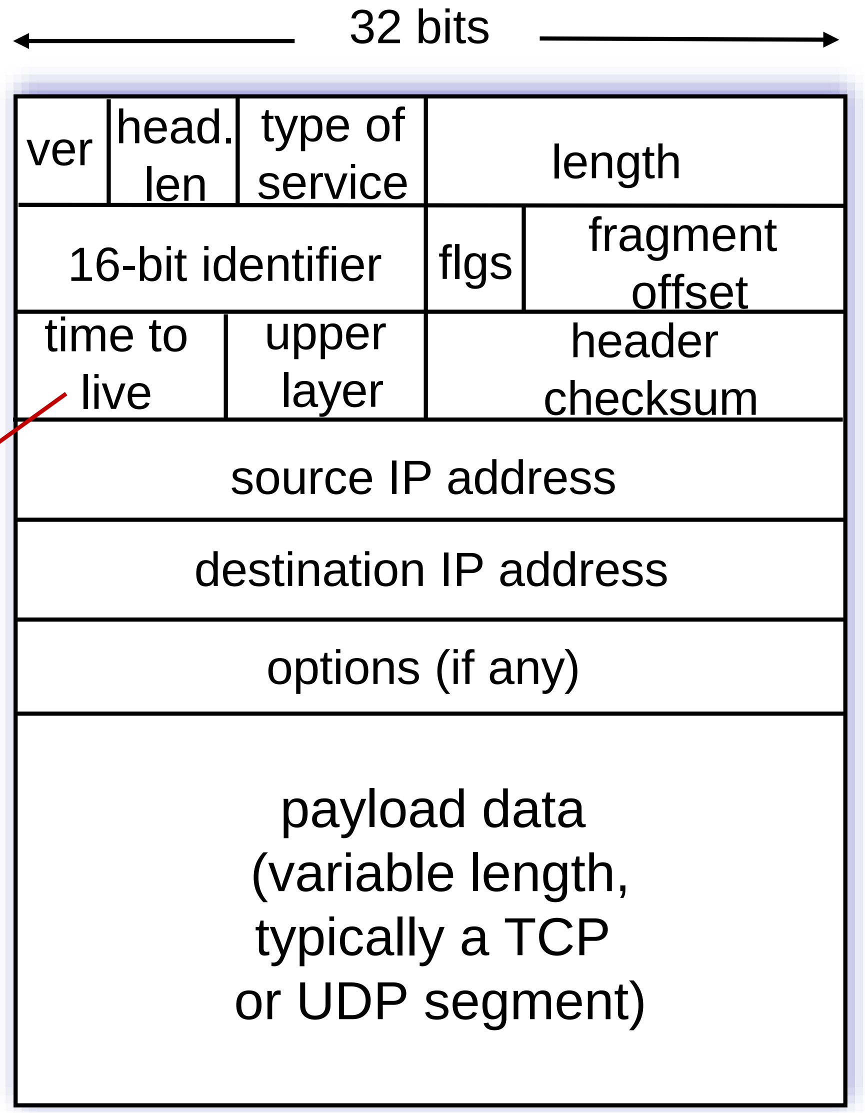
payload data (variable length, typically a TCP or UDP segment)

20 bytes

overhead

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

IP Datagram format - TTL



32 bits

ver | head. len | type of service | length

16-bit identifier | flgs | fragment offset

time to live | upper layer | header checksum

source IP address

destination IP address

options (if any)

payload data (variable length, typically a TCP or UDP segment)

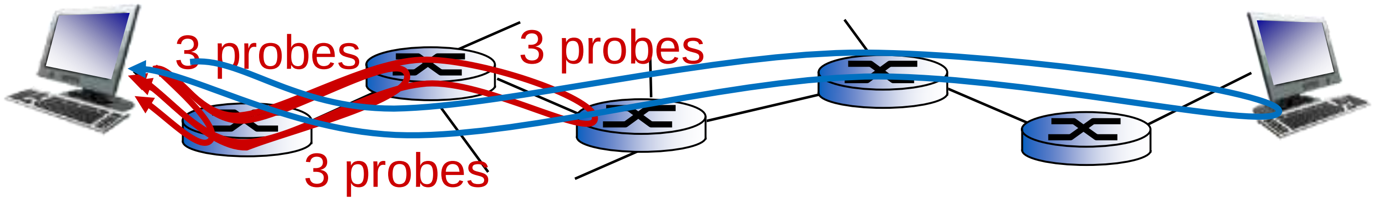
TTL: remaining max hops
(decremented at each router)

TTL

- Purpose:
  get rid of looping datagram
- Router process:

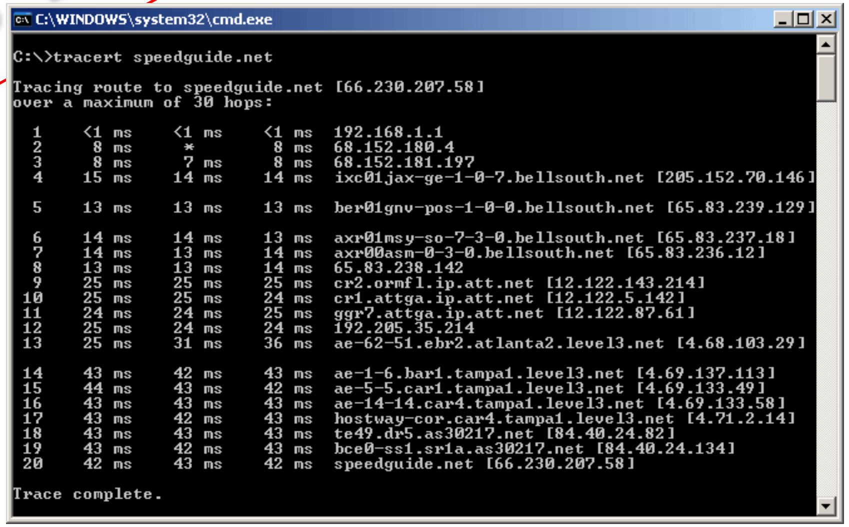
```
TTL -= 1
if TTL == 0:
    drop(datagram)
```

TTL and Traceroute



- source sends series of UDP segments to destination
  - 1st set has TTL =1, discarded at 1st router → notify source, source record router and round-trip time.
  - 2nd set has TTL=2, discarded at 2nd router → notify source, source record router and round-trip time.
  - …
  - reach destination with unlikely port number: destination host notify source with “*port unreachable*”, source record destination and round-trip time.
- Source now has
  - A list of all routers and their round-trip time along the path
  - the destination and destination round-trip time.

This week’s Lab



```
C:\>tracert speedguide.net

Tracing route to speedguide.net [66.230.207.58]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms  192.168.1.1
  2     8 ms     *        8 ms  68.152.180.4
  3     8 ms     7 ms     8 ms  68.152.181.197
  4    15 ms    14 ms    14 ms  ixc01jax-ge-1-0-7.bellsouth.net [205.152.70.146]

  5    13 ms    13 ms    13 ms  ber01gnv-pos-1-0-0.bellsouth.net [65.83.239.129]

  6    14 ms    14 ms    13 ms  axr01msy-so-7-3-0.bellsouth.net [65.83.237.18]
  7    14 ms    13 ms    14 ms  axr00asm-0-3-0.bellsouth.net [65.83.236.12]
  8    13 ms    13 ms    14 ms  65.83.238.142
  9    25 ms    25 ms    25 ms  cr2.ormfl.ip.att.net [12.122.143.214]
 10    25 ms    25 ms    24 ms  cr1.attga.ip.att.net [12.122.5.142]
 11    24 ms    24 ms    25 ms  ggr7.attga.ip.att.net [12.122.87.61]
 12    25 ms    24 ms    24 ms  192.205.35.214
 13    25 ms    31 ms    36 ms  ae-62-51.ebr2.atlanta2.level3.net [4.68.103.29]

 14    43 ms    42 ms    43 ms  ae-1-6.bar1.tampa1.level3.net [4.69.137.113]
 15    44 ms    43 ms    42 ms  ae-5-5.car1.tampa1.level3.net [4.69.133.49]
 16    43 ms    43 ms    43 ms  ae-14-14.car4.tampa1.level3.net [4.69.133.58]
 17    43 ms    42 ms    43 ms  hostway-cor.car4.tampa1.level3.net [4.71.2.14]
 18    43 ms    43 ms    43 ms  te49.dr5.as30217.net [84.40.24.82]
 19    43 ms    42 ms    43 ms  bce0-ss1.sr1a.as30217.net [84.40.24.134]
 20    42 ms    43 ms    42 ms  speedguide.net [66.230.207.58]

Trace complete.
```

Network layer: “data plane” roadmap

4.1 Network layer: overview
- data plane
- control plane

4.2 What’s inside a router
- input ports, switching, output ports
- buffer management, scheduling

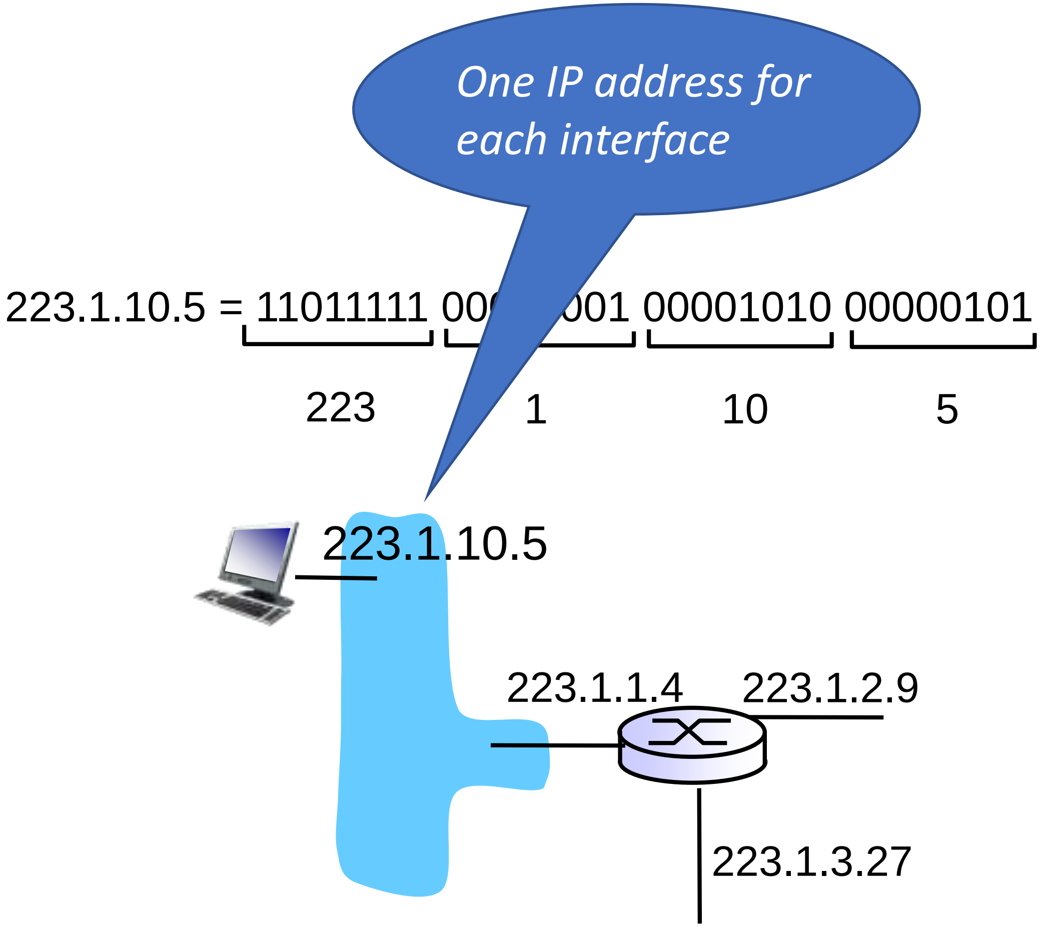
4.3 IP: the Internet Protocol
- datagram format
- addressing
- network address translation
- IPv6

~~4.4 Generalized Forwarding, SDN~~
- ~~Match+action~~
- ~~OpenFlow: match+action in action~~

~~4.5 Middleboxes~~

IP address: introduction

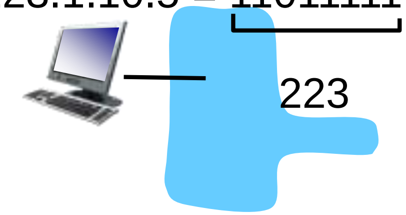
- IP address: 32-bit identifier associated with each host or router *interface*
- interface: connection between host/router and physical link
  - router's typically have multiple interfaces
  - host typically has one or two interfaces (e.g., wired Ethernet, wireless 802.11)
    - only one interface is active at a time



One IP address for each interface

IP address: decimal → binary

$$223.1.10.5 = \underbrace{11011111}_{223}.\underbrace{01100100}_{100}.\underbrace{00001010}_{10}.\underbrace{00000101}_{5}$$



1. Write down binary table:

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

2. convert 10 to binary: 10-8=2; 2-2=0. (deduct numbers: high to low)

3.

0	0	0	0	1	0	1	0

2. converting 100 to binary: deduct numbers: high to low

100 - 64 = 36; 36 - 32 = 4; 4 - 4 = 0

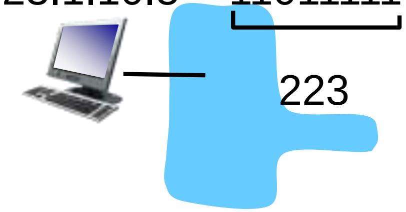
3.

0	1	1	0	0	1	0	0

Try 223?

IP address: binary → decimal

$$223.1.10.5 = \underbrace{11011111}_{223}.\underbrace{01100100}_{100}.\underbrace{00001010}_{10}.\underbrace{00000101}_{5}$$



1. Write down binary table:

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

2. write binary byte under the table:

0	1	1	0	0	1	0	0

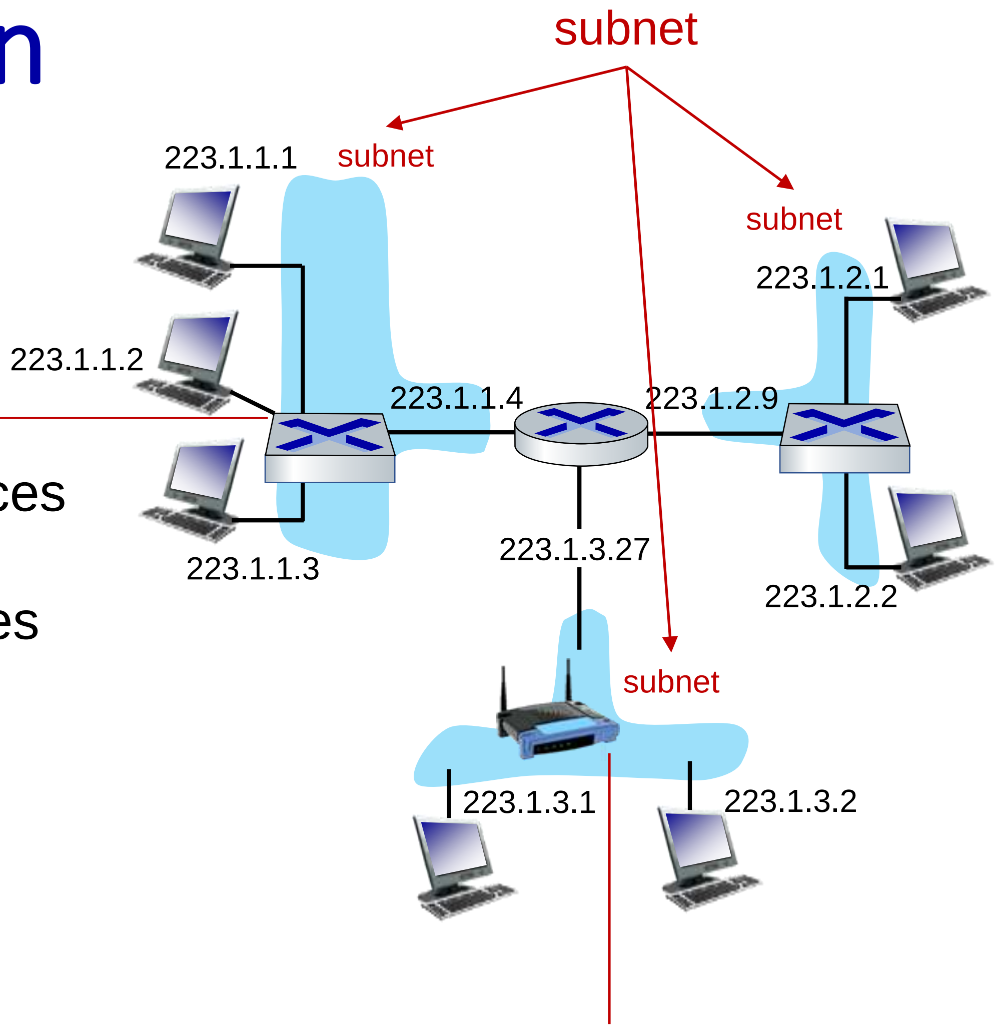
3. add 1’s values:

64 + 32 + 4 = 100

IP addressing: introduction

Q: how are interfaces actually connected?

A: we'll learn about that in chapters 6, 7



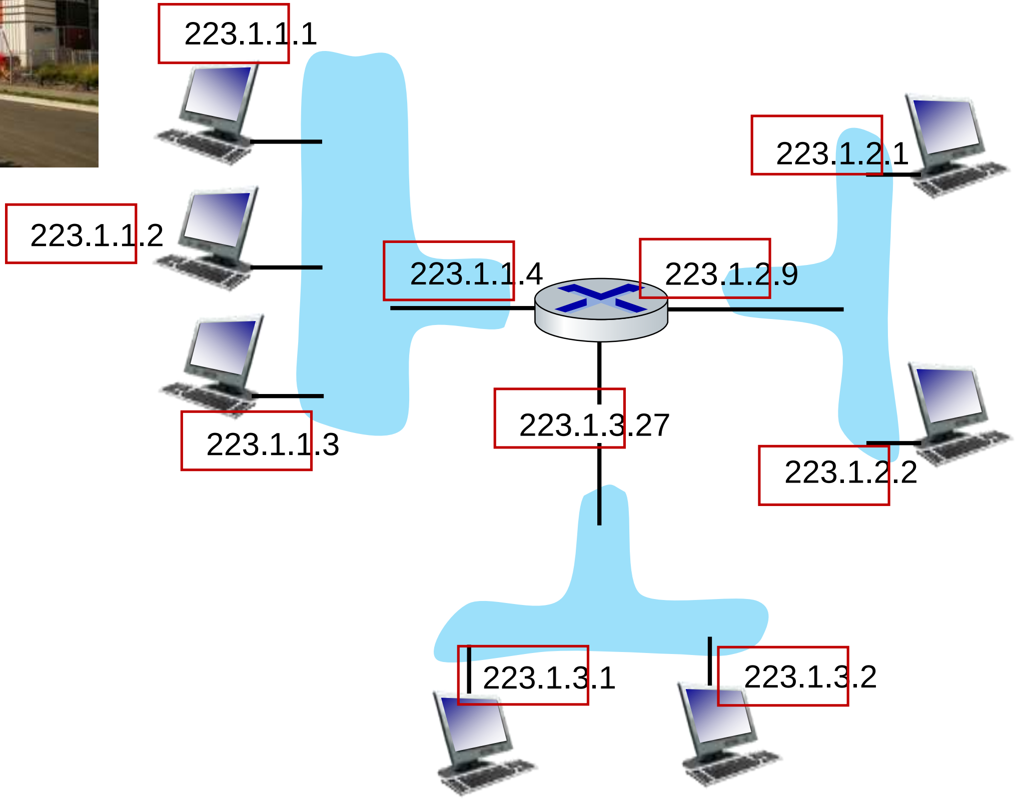
A: wired Ethernet interfaces connected by Ethernet switches

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
  - Analogy: Houses in a Street
- IP addresses have structure:
  - subnet part: common high order bits
  - host part: low order bits

- IP address: subnet.host#
  223.1.1.3
- Street addr: street.house#
  Smith St.3

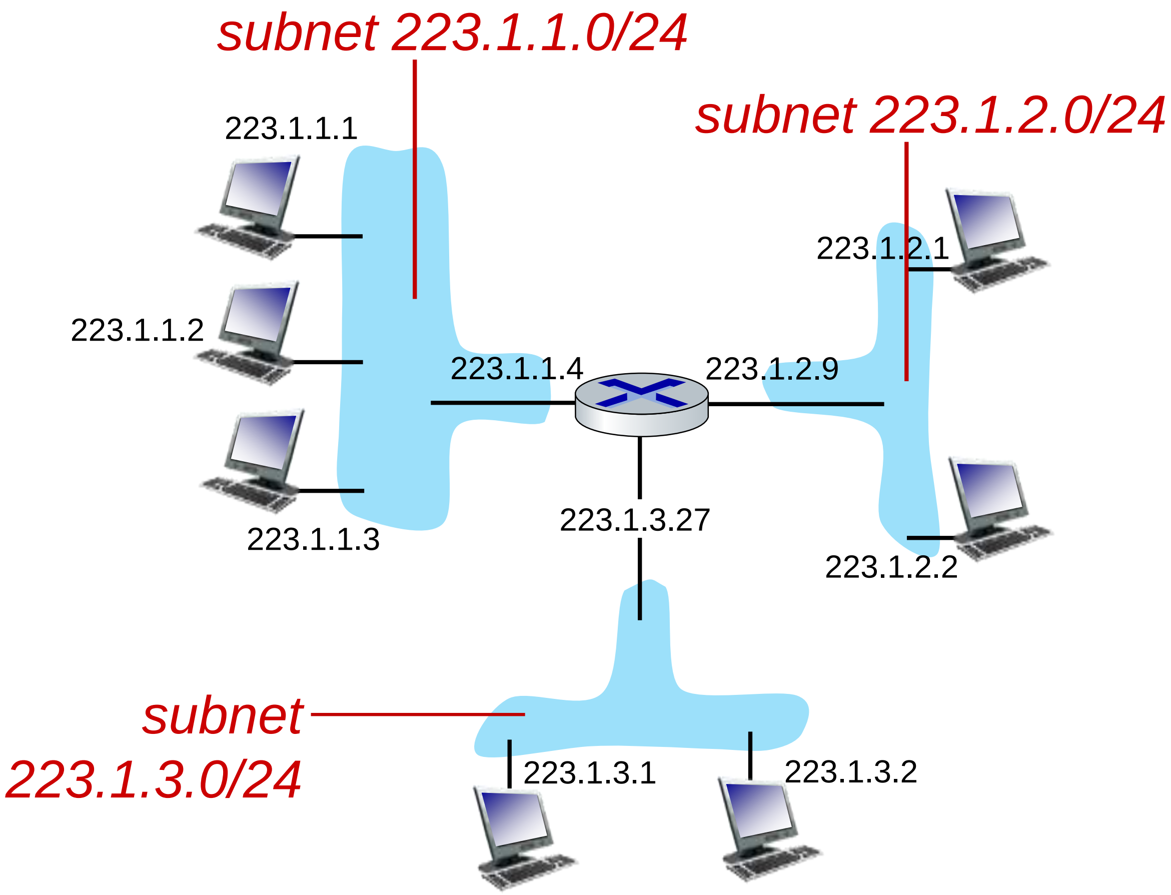


network consisting of 3 subnets

Subnets

Recipe for defining subnets:

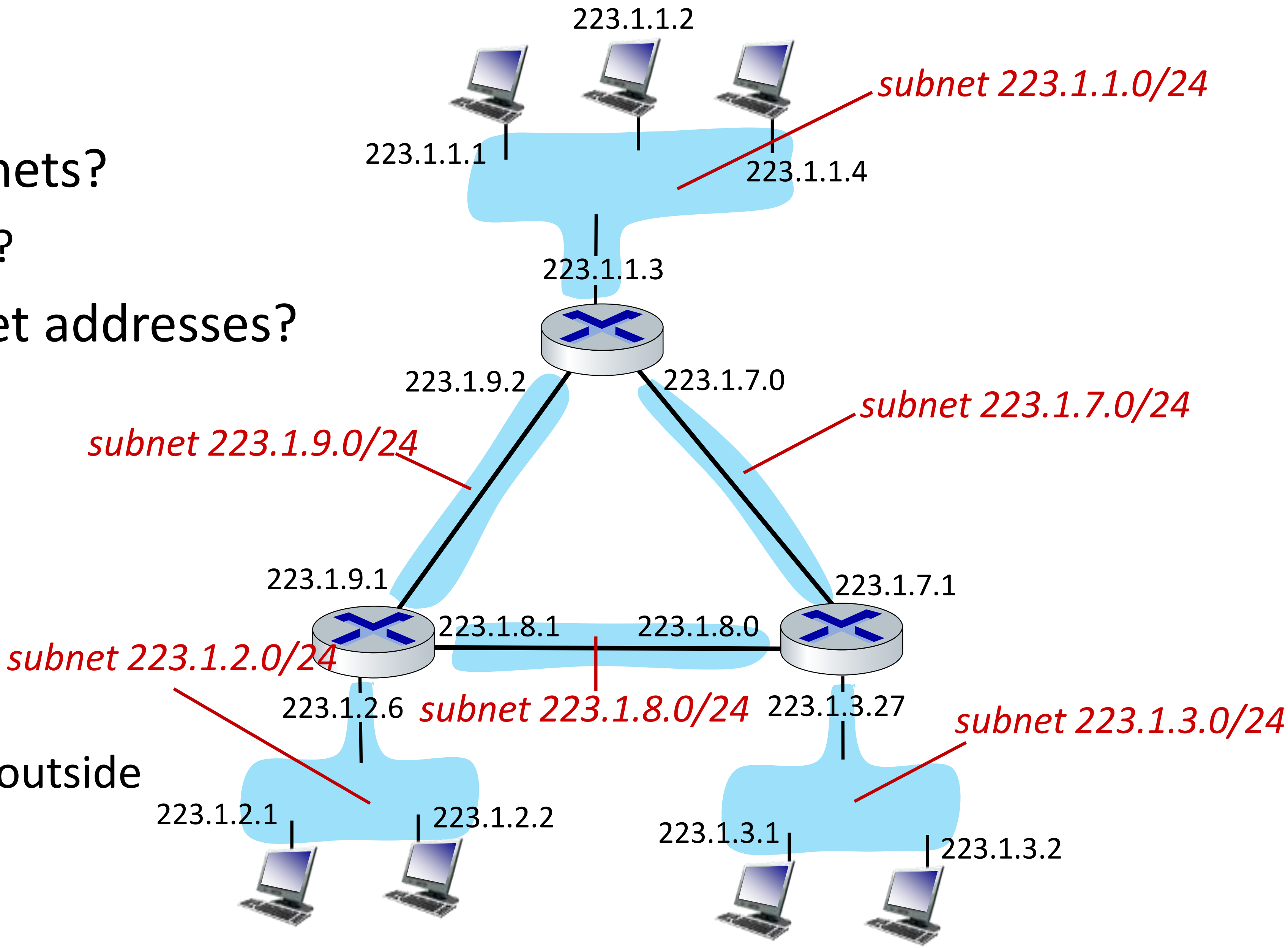
- group IP addresses with the same high order part
- creating “islands” of isolated networks
- each isolated network is called a *subnet*
- Write IP address
  - IP address: 223.1.1.3
    subnet mask: 255.255.255.0
  - shorthand: 223.1.1.3/24



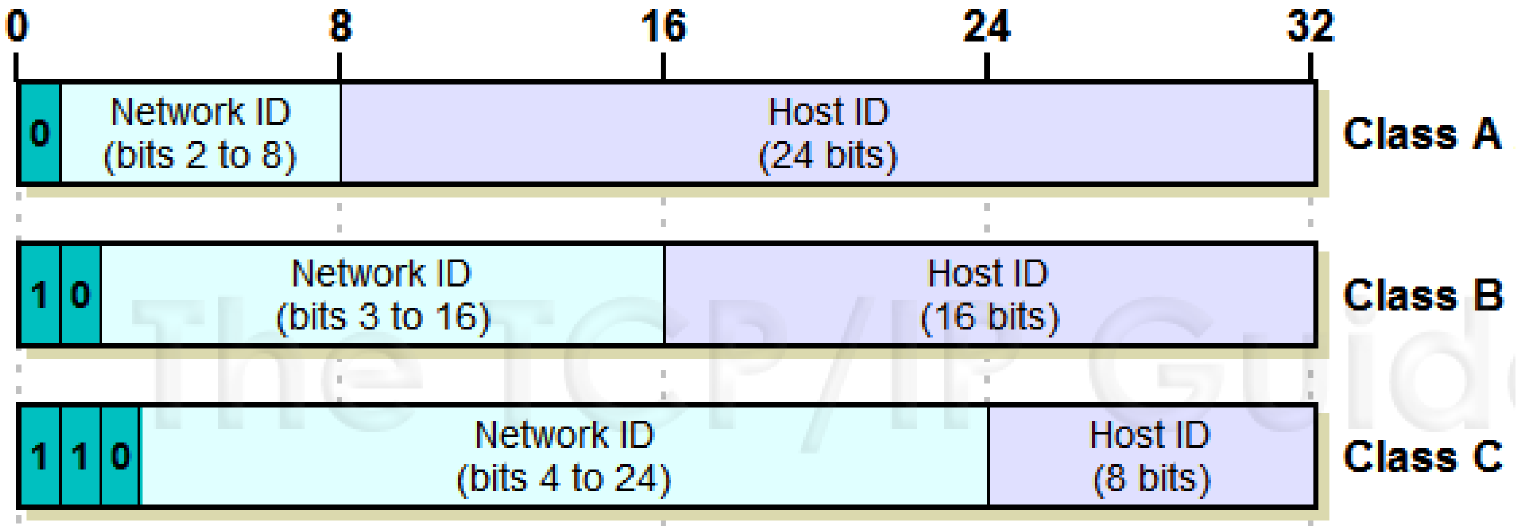
subnet mask: /24
(high-order 24 bits: subnet part of IP address)

Subnets

- where are the subnets?
  - how many subnets?
- what are the subnet addresses?
  - 223.1.1.0/24
  - ...
  - 223.1.9.0/24
- host IP address?
- router IP address?
  - router: gateway to outside



IP Address - Classful

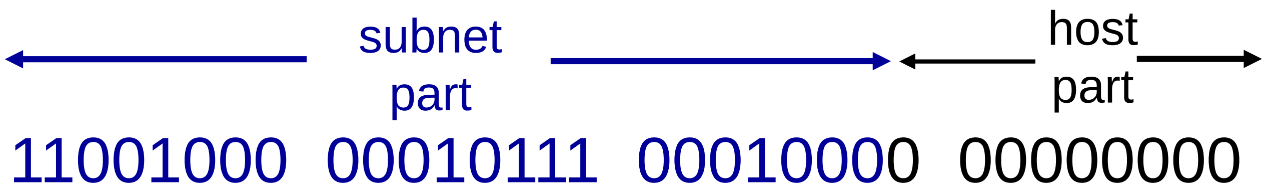


Class	Start Address	End Address	Subnet Mask	shorthand
A	0.0.0.0	127.255.255.255	255.0.0.0	/8
B	128.0.0.0	191.255.255.255	255.255.0.0	/16
C	192.0.0.0	223.255.255.255	255.255.255.0	/24

IP addressing: CIDR

CIDR: Classless InterDomain Routing (pronounced “cider”)

- subnet portion of address of arbitrary length
- 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

Chapter 4 – part1: Summary

- Network layer: overview
- What's inside a router
- IP addressing, subnet
- Forwarding/routing
- Middleboxes

Question: how are forwarding tables (destination-based forwarding) or flow tables (generalized forwarding) computed?

Answer: by the control plane (next chapter)

Lecture done ✓

- Q & A