RECAP:

Network Layer (Router)

Local fn → forwarding of pkts.

non local / global fn → Routing

- Network Layer Architecture
- Routing Table (generate globally)
- Control fn (plane)
- Data plane (forwarding)
- Addressing → longest prefix search.

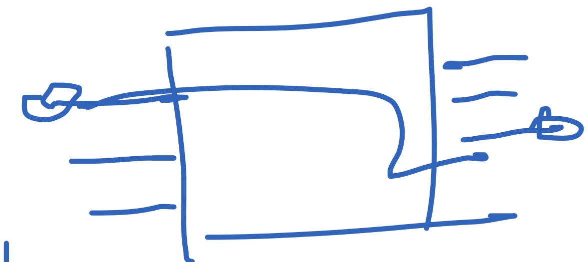
TCP/IP stack

↳ Std Network layer prot.

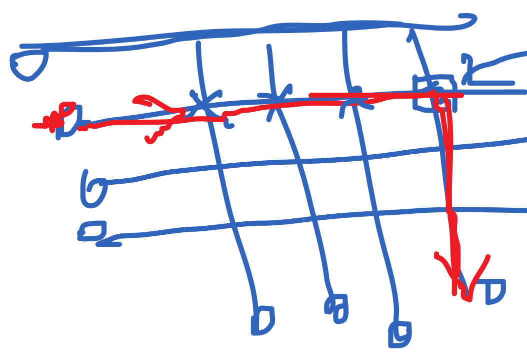
Router (forwarding)

↳ switching

input ⟶ output



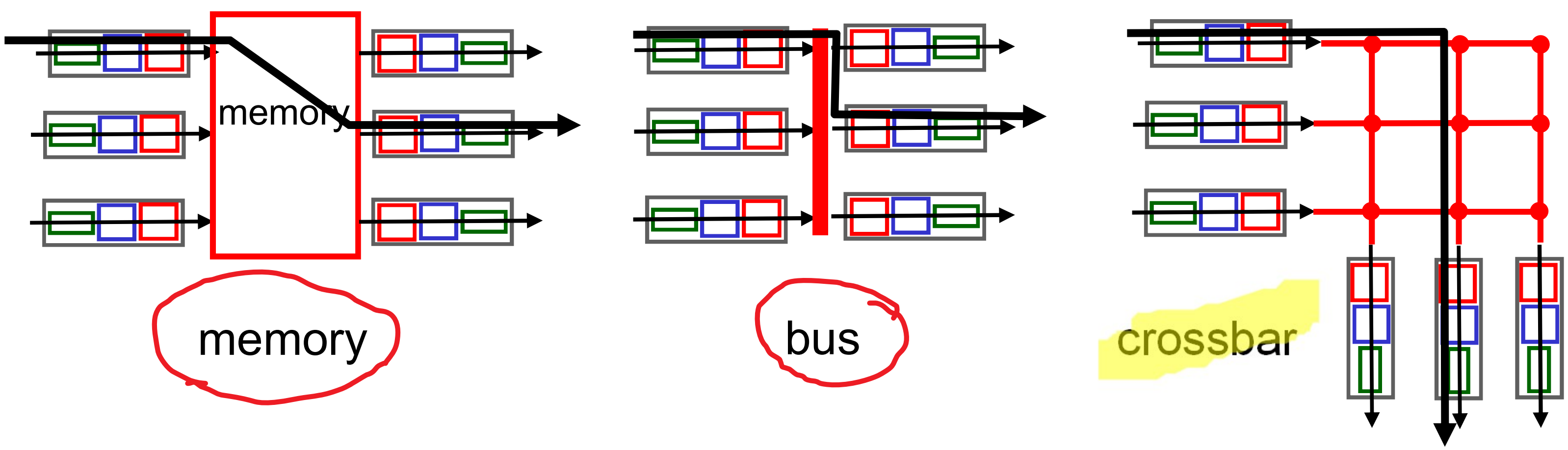
[Crossbar]



close/open

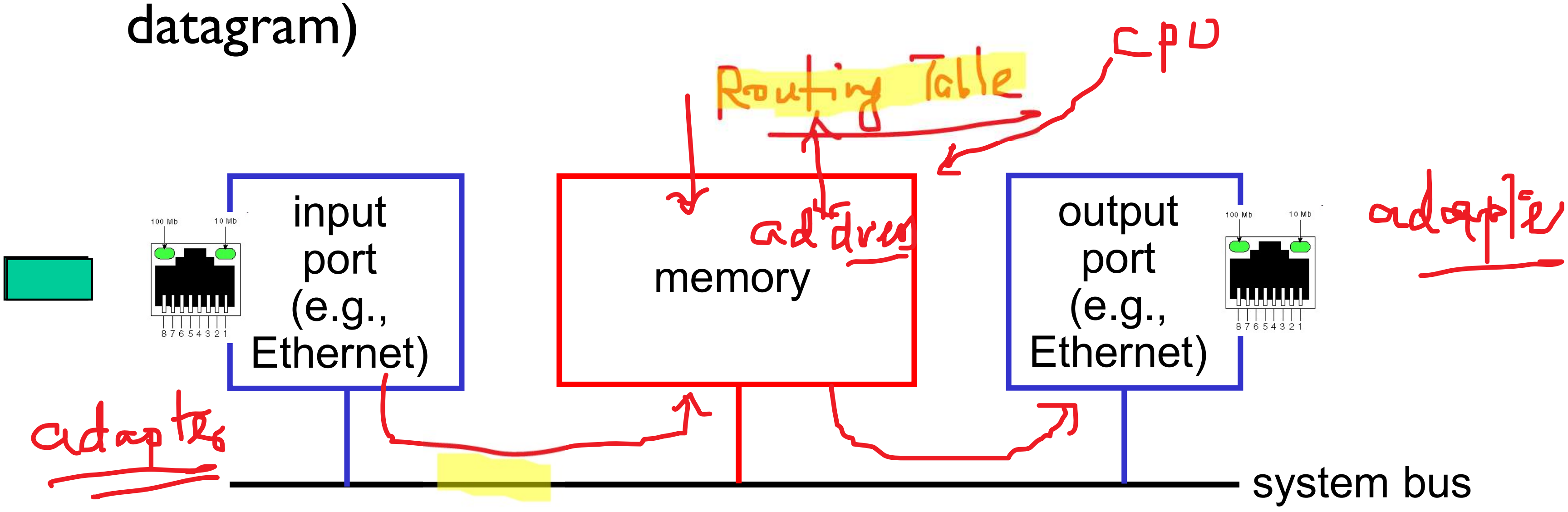
Switching fabrics

- 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

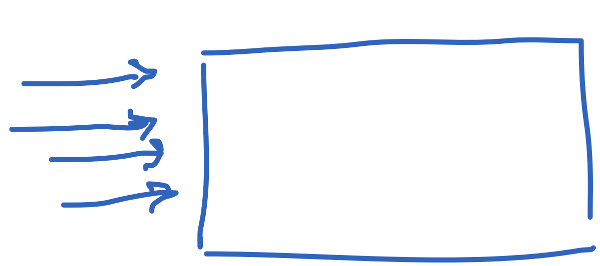
first generation routers:
- traditional computers with switching under direct control of CPU
- packet copied to system's memory
- speed limited by memory bandwidth (2 bus crossings per datagram)



Software based switching

Routing

N ports.



1 pkt at a time.

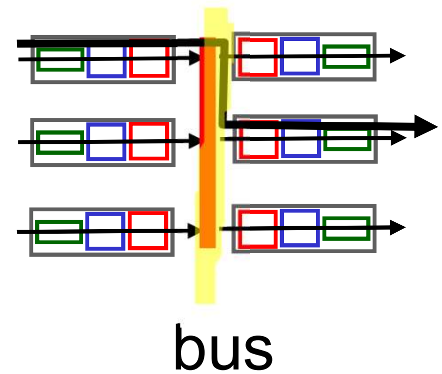
multiple memory read/write op.

Bottleneck.

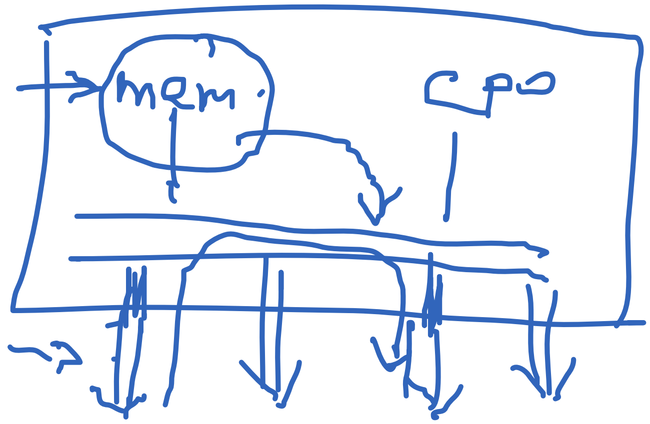
Switching via a bus

- 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



Routing Table



Network proc

← adapters

PC/Router

Bus

PCI Bus

↳ bottleneck

Bus Mastering Capabilities

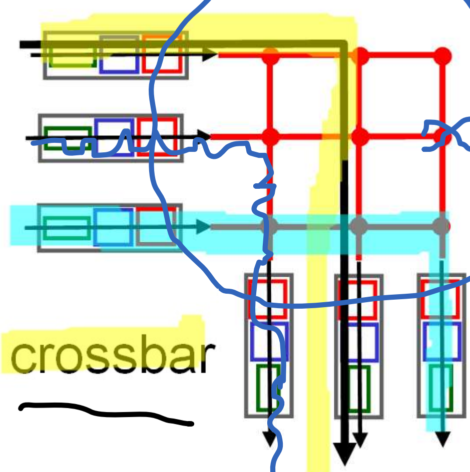
Switching via interconnection network

- overcome bus bandwidth limitations
  Batcher-Banyan Switch
- 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

Interconnects = N^2

Gates, Control

N



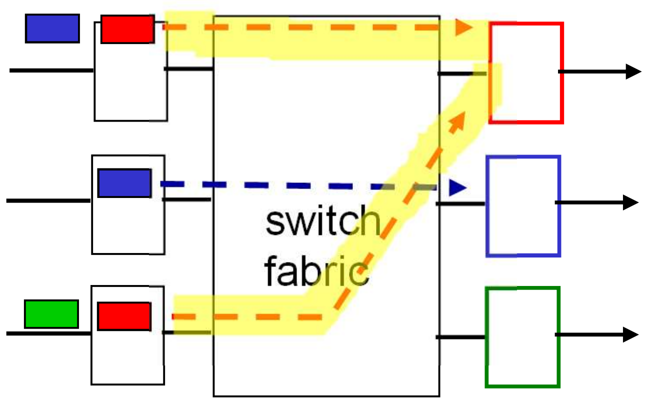
crossbar

N

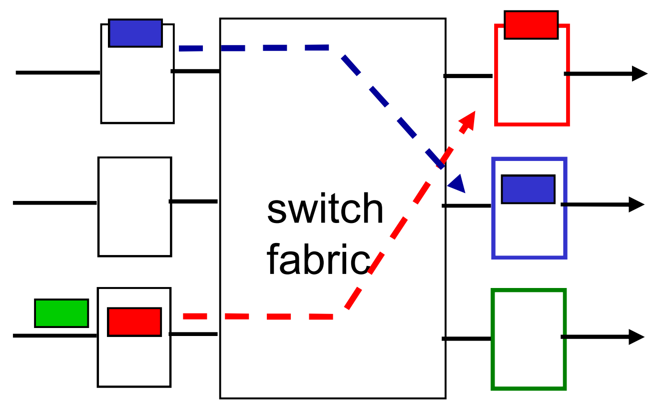
1 Tbps.

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



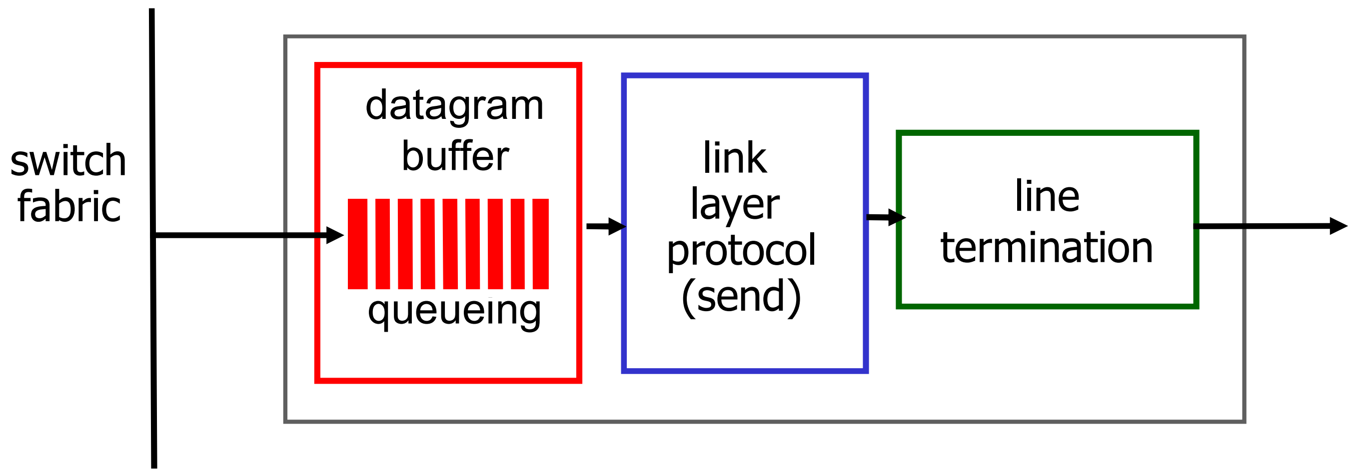
output port contention:
only one red datagram can be transferred.
lower red packet is blocked



one packet time later:
green packet experiences HOL blocking

Output ports

buffer -



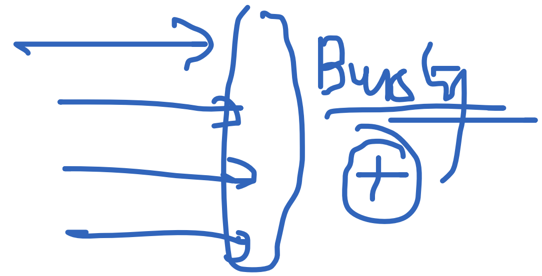
- *buffering* required from fabric faster rate

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

- *scheduling* datagrams

Priority scheduling – who gets best performance, network neutrality

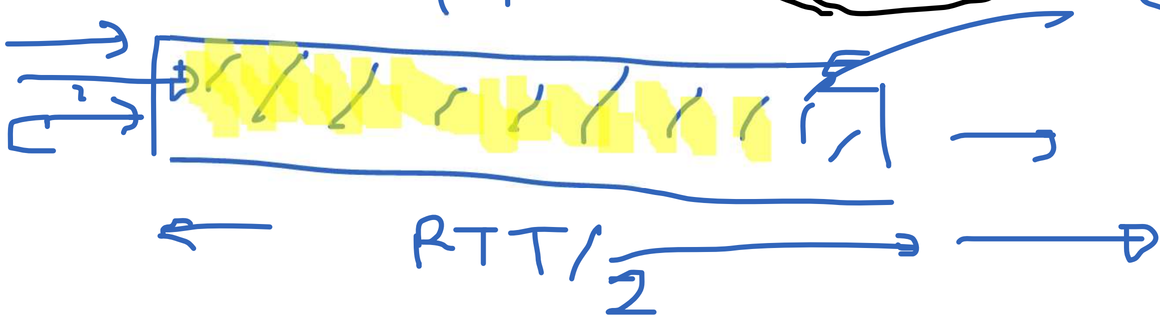
How much buffering?

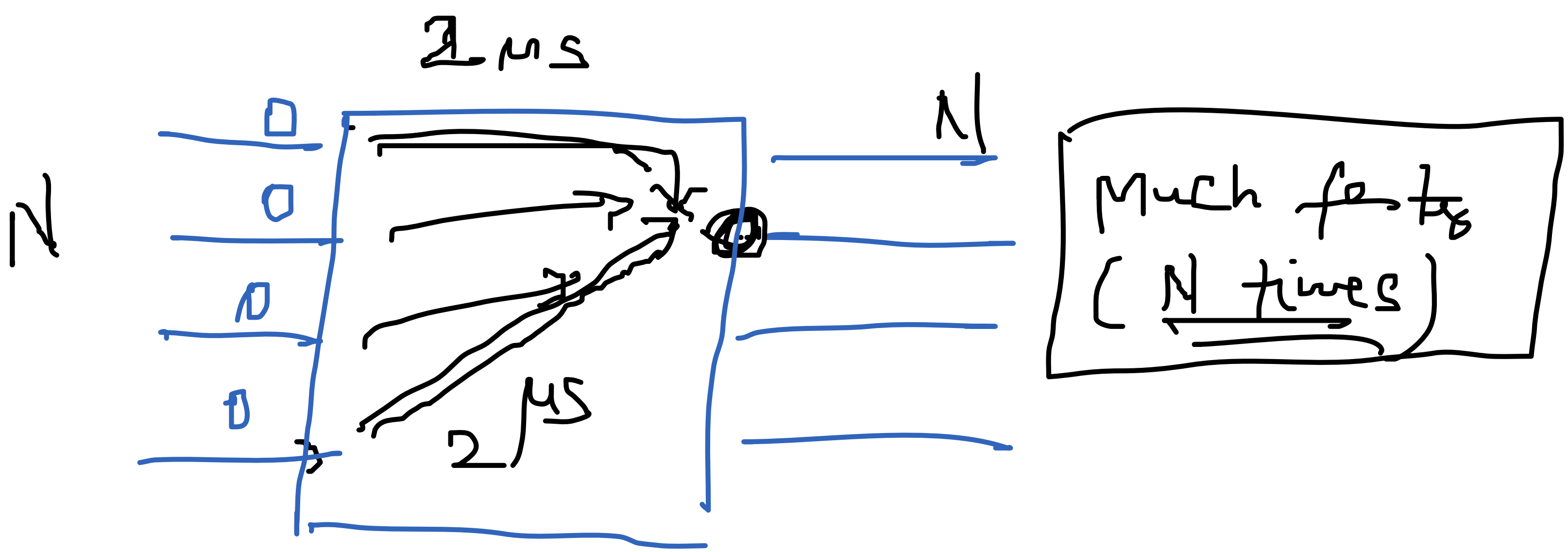


- RFC 3439 rule of thumb: average buffering equal to "typical" RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gpbs link: 2.5 Gbit buffer
- recent recommendation: with N flows, buffering equal to $\propto \left[\frac{RTT \cdot C}{\sqrt{N}}\right]$

Multiple srcs → bursty pipe

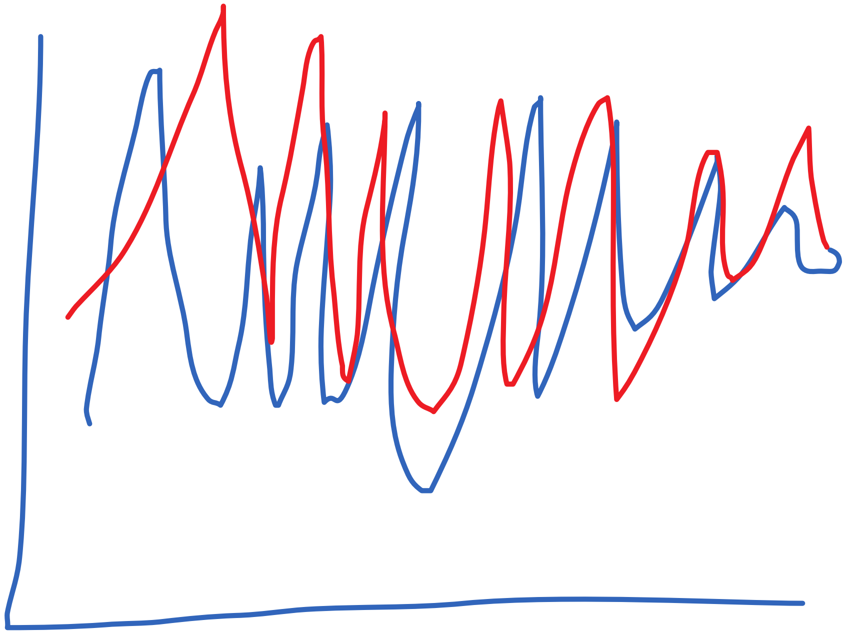
$(C * RTT)/2$

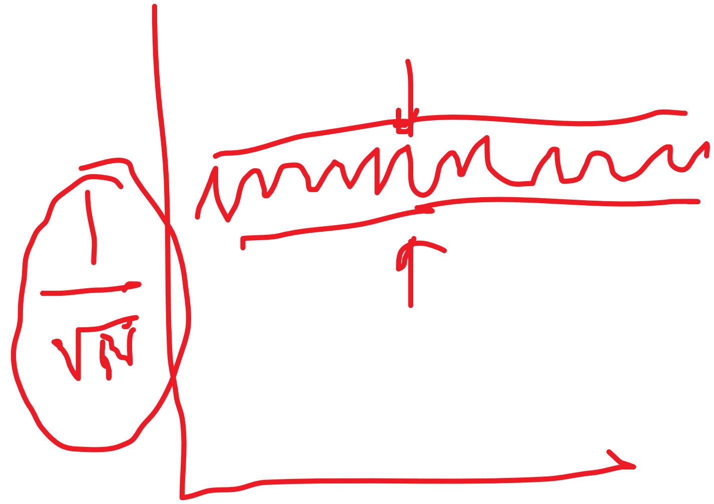




output buffer operates at
N time the switching rate

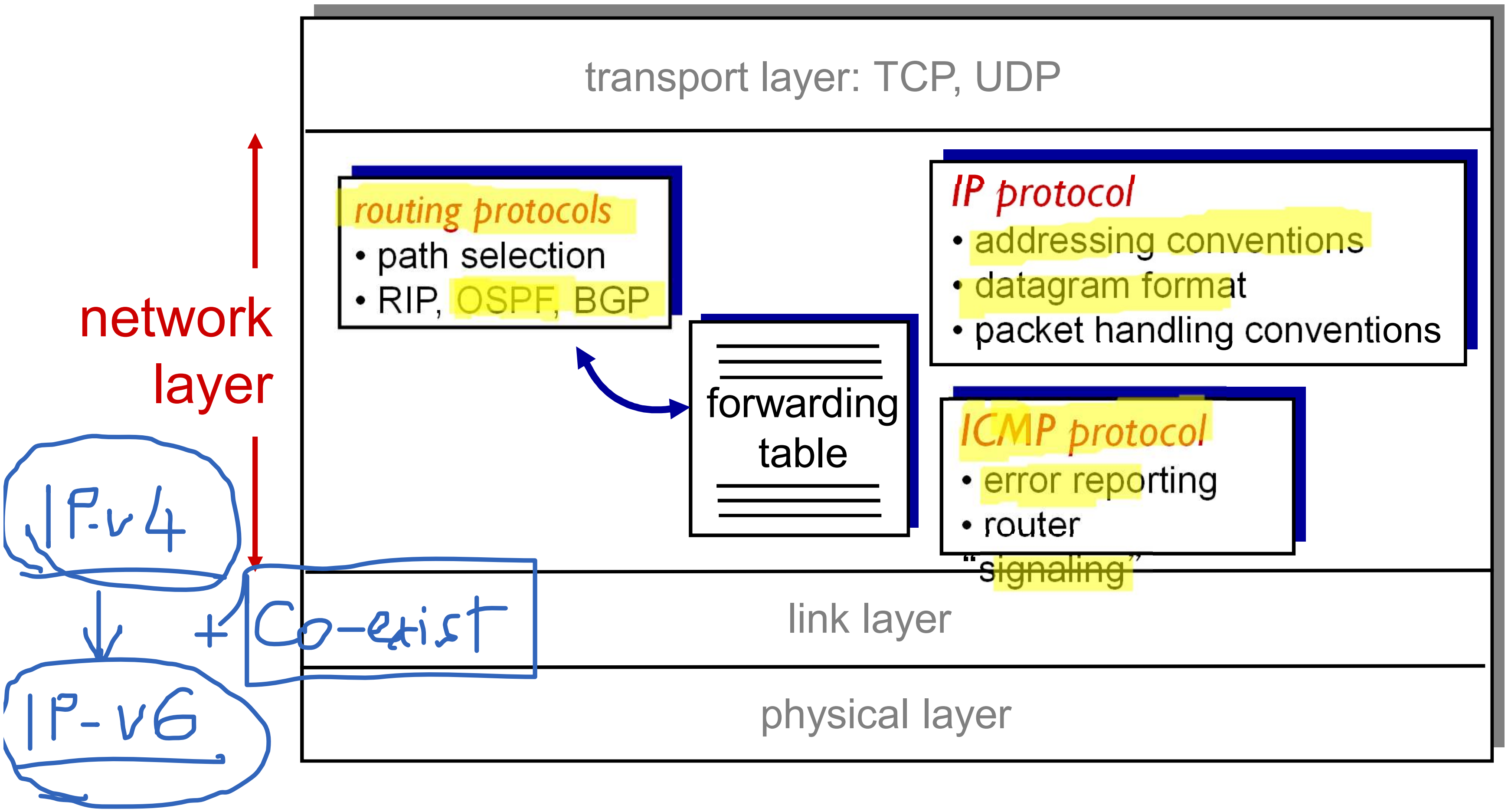
switch time = 2 μs
buffer write time = 1 μs
↳ 2 packets can be written in
the memory in 2 μs.



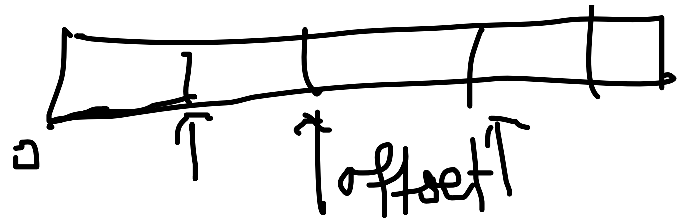

1
√N

The Internet network layer

host, router network layer functions:



IP datagram format



IP protocol version number

header length (bytes)

"type" of data

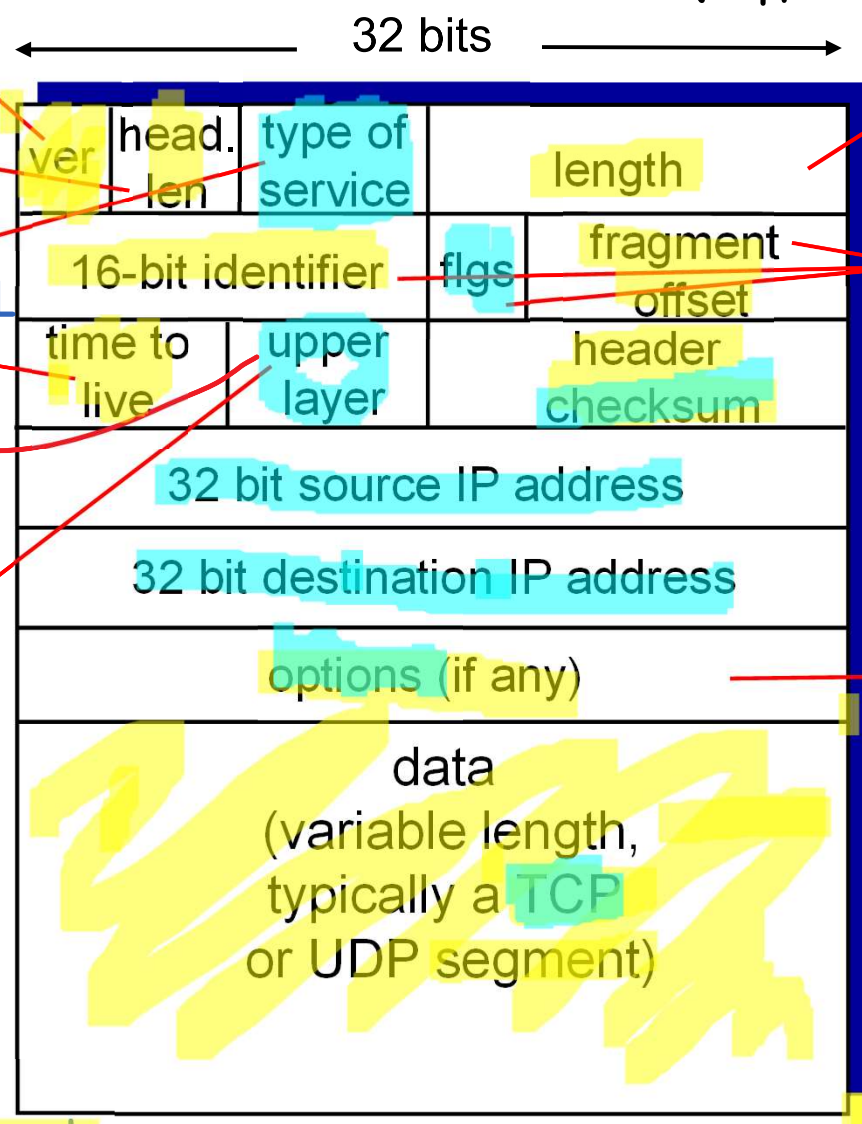
TCP/UDP

OSPF

TTL

max number remaining hops (decremented at each router)

upper layer protocol to deliver payload to



total datagram length (bytes)

for fragmentation/ reassembly

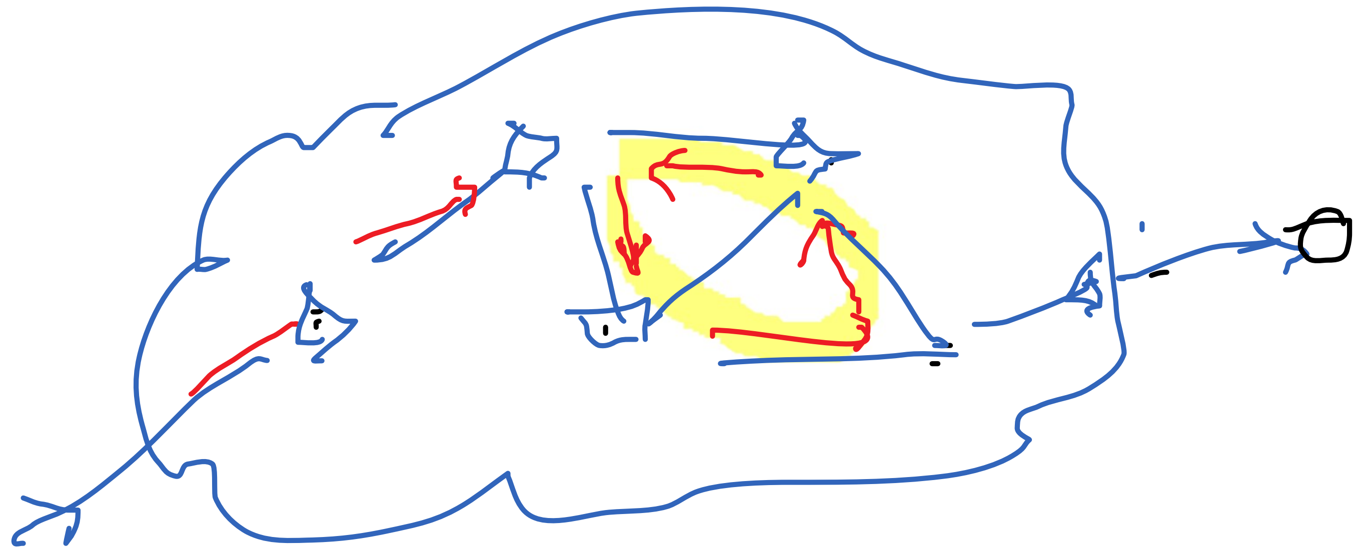
e.g. timestamp, record route taken, specify list of routers to visit.

how much overhead?

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

TTL : Max Hop Count.

TTL:



pkt misrouted → loop

[memory leakage] : { – malloc() – dont have free() } loop

Counter

TTL = 100

every router decrements counter

when count = 0, drop the pkt

IPv4

32 bit addresses

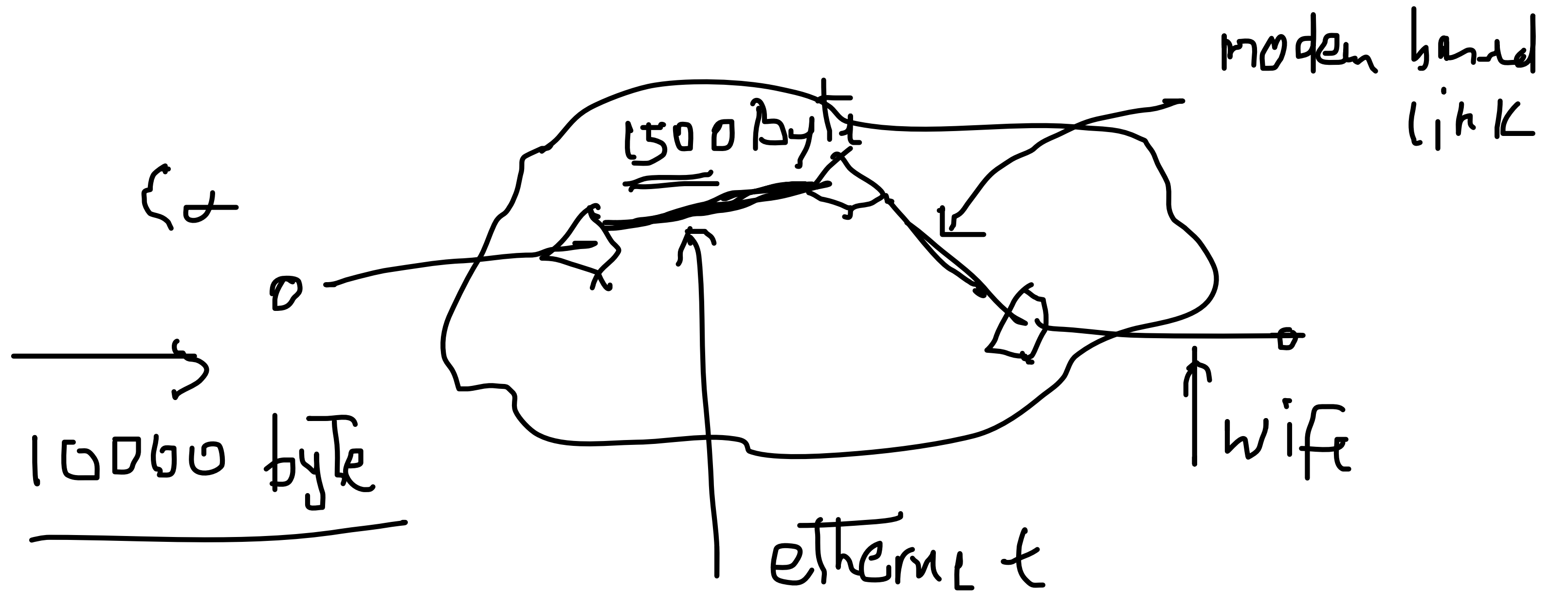
102.202.3.115

Routing Table

dest IP addr	Output port
102.202.3.15	3.
⋮	⋮

Match

longest prefix match



diff max frame size

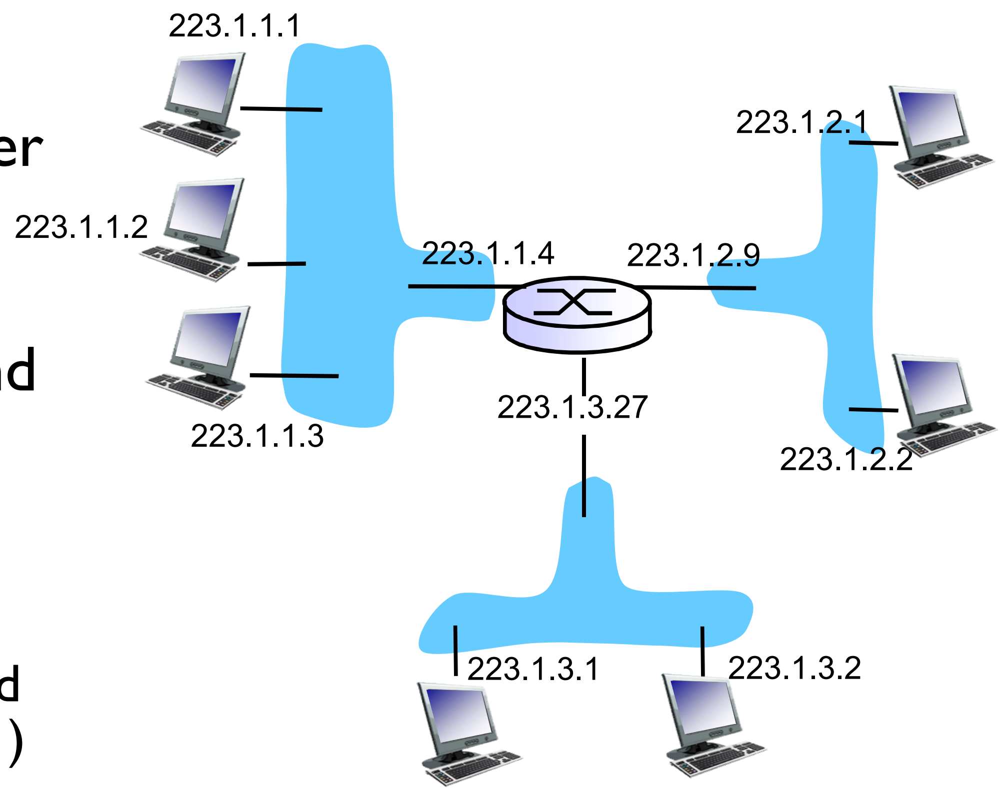
10000 B → fragment

1500 B → 7 fragments

end combine fragments.

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*

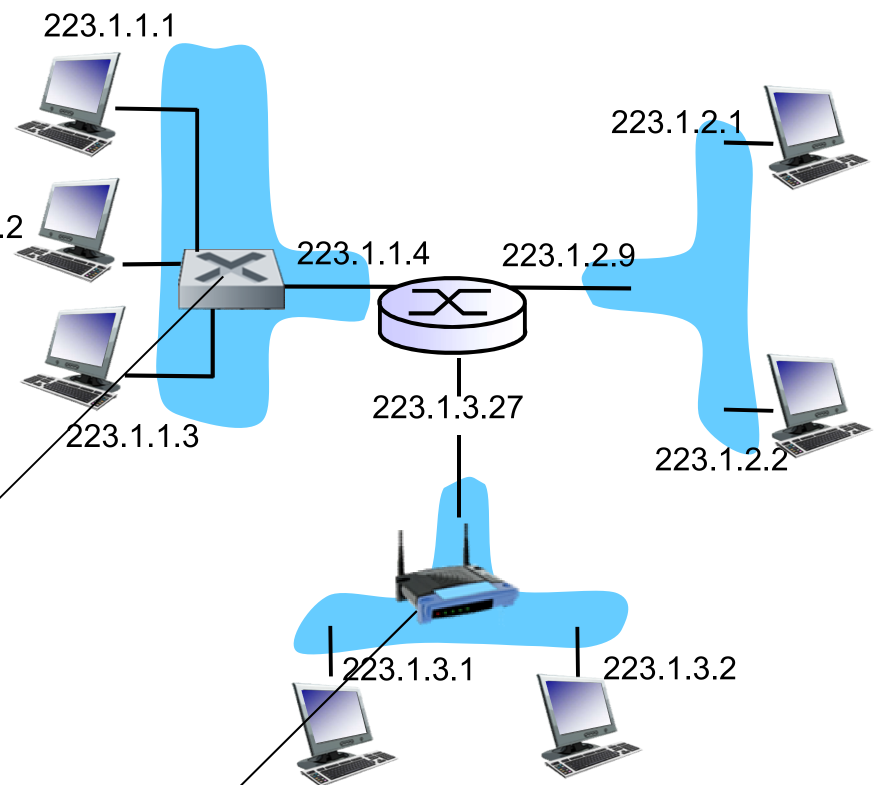


$$223.1.1.1 = \underbrace{11011111}_{223}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}\ \underbrace{00000001}_{1}$$

IP addressing: introduction

Q: how are interfaces actually connected?

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



A: wired Ethernet interfaces connected by Ethernet switches

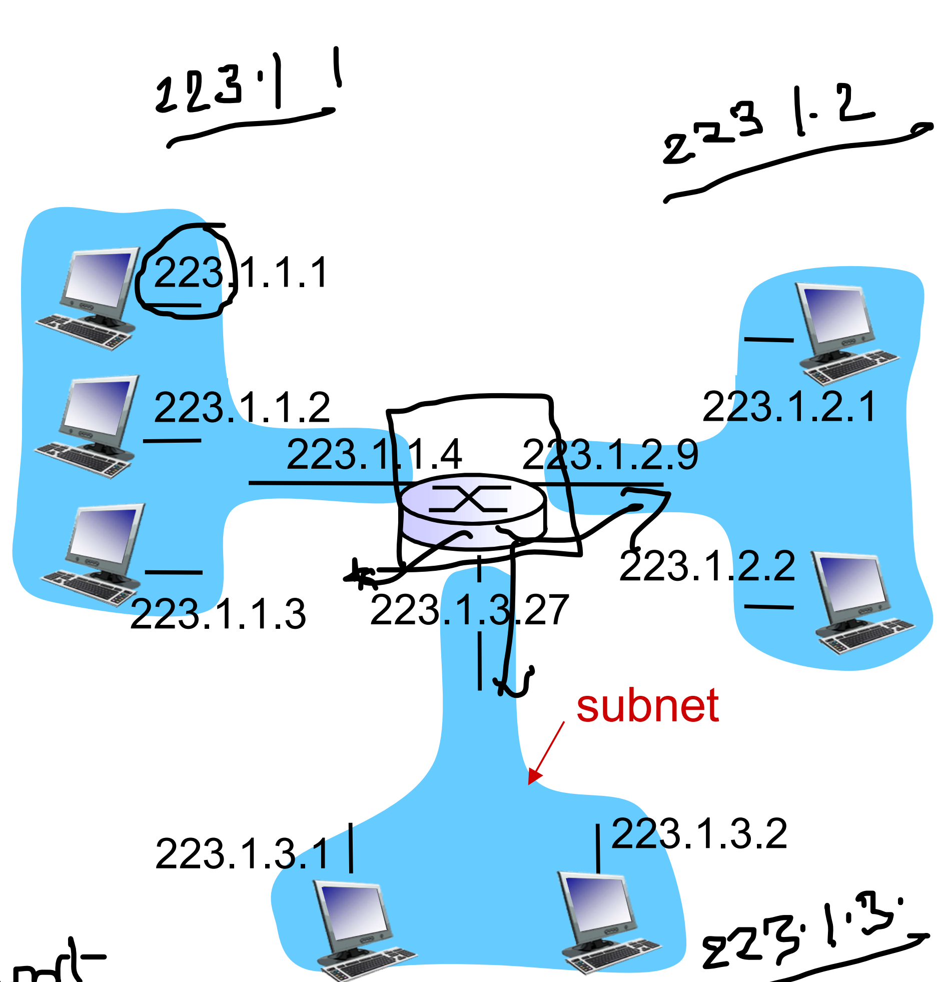
A: wireless WiFi interfaces connected by WiFi base station

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

Subnets

LAN

- 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*



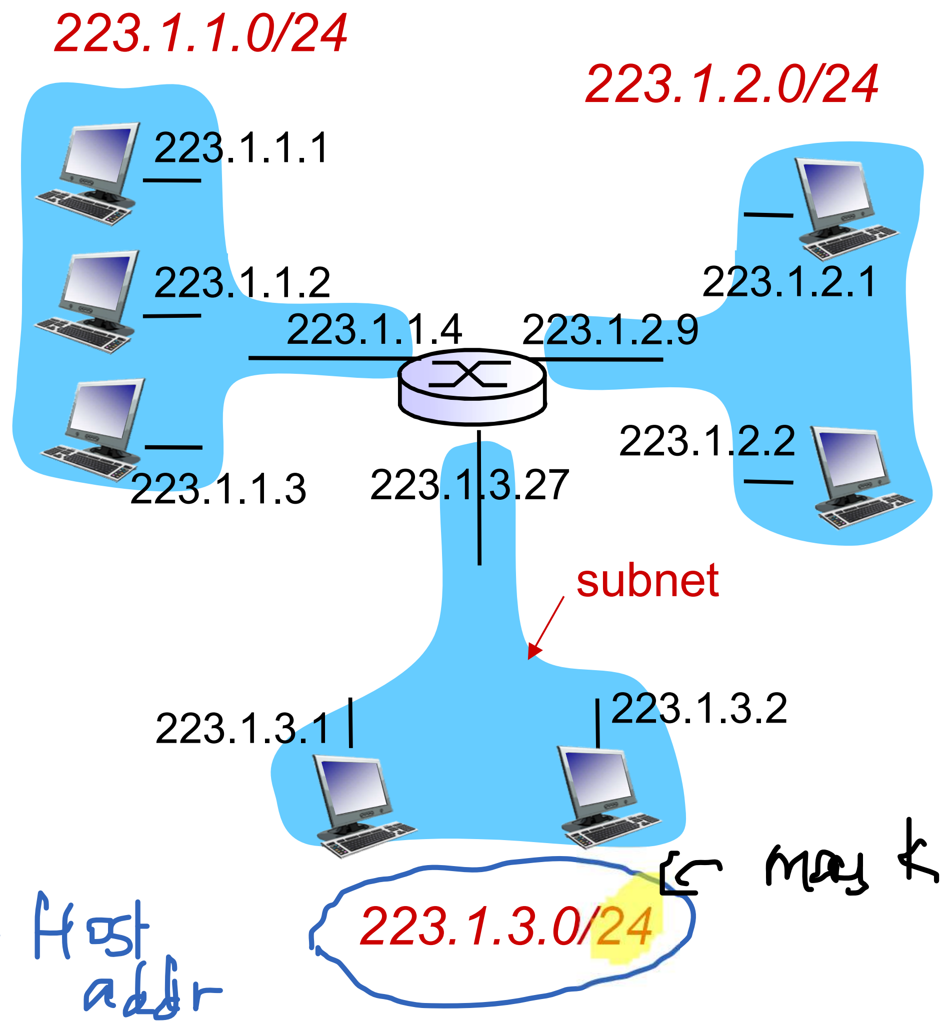
network consisting of 3 subnets

Network
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: /24

Addr → Network addr + Host addr

223.1.3.0/24

IP addressing: CIDR

CIDR: Classless InterDomain Routing

- 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