Chapter 4: Network Layer

Chapter goals:

- understand principles behind network layer services:
  - how a router works
  - routing (path selection)
  - dealing with scale
- instantiation and implementation in the Internet (incl. advanced topics: IPv6, multicast)

Overview:

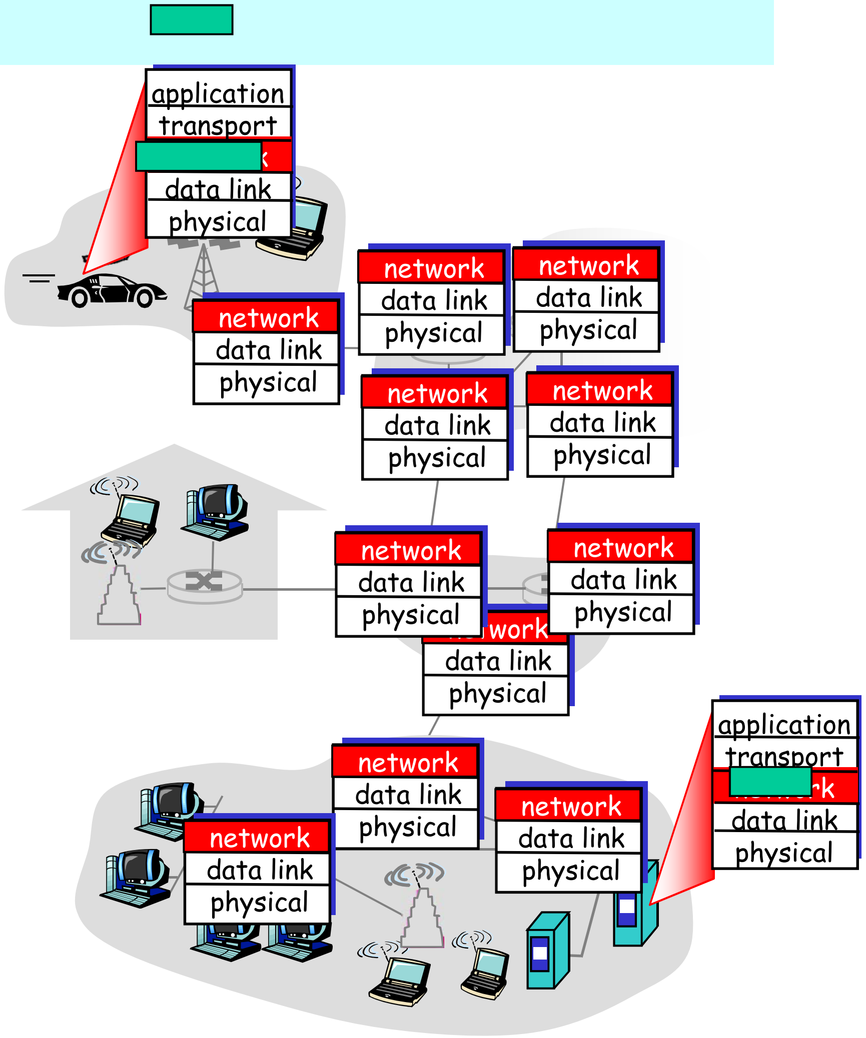
- network layer services
  - VC, datagram
- what's inside a router?
- Addressing, forwarding, IP
- routing principle: path selection
  - hierarchical routing
  - Internet routing protocols

Network layer

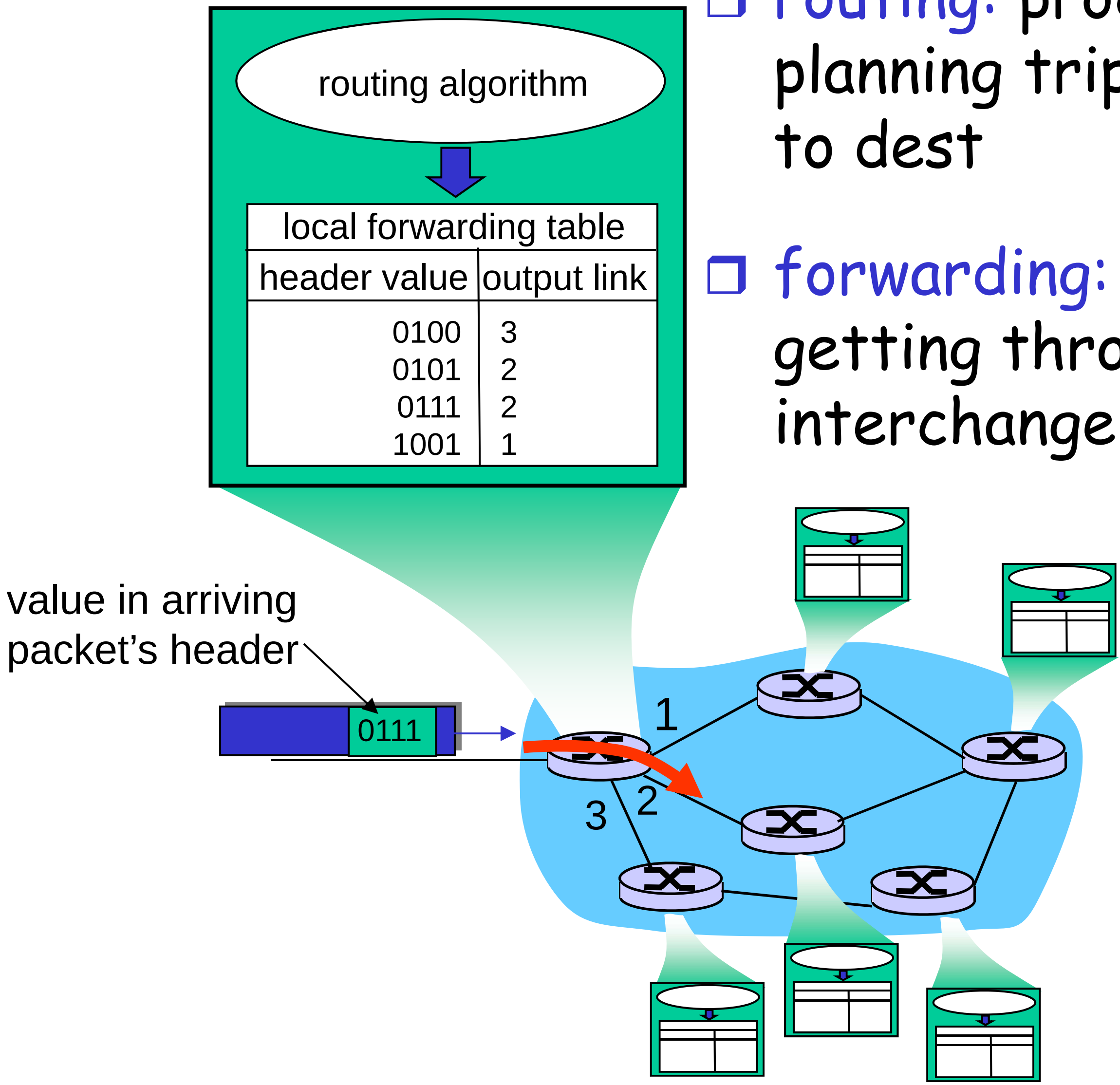
- transport packet from sending to receiving hosts
- network layer protocols in *every* host, router

important functions

- *path determination:* route taken by packets from source to dest. *Routing algorithms*
- *switching:* move packets from router's input to appropriate router output
- *call setup:* (in some some network architectures) along path before data flows
- *congestion control* (in some network architectures)



Interplay between routing and forwarding



- routing: process of planning trip from source to dest
- forwarding: process of getting through single interchange

Network service model

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

service abstraction

- guaranteed bandwidth?
- preservation of inter-packet timing (no jitter)?
- loss-free delivery?
- in-order delivery?
- congestion feedback to sender?

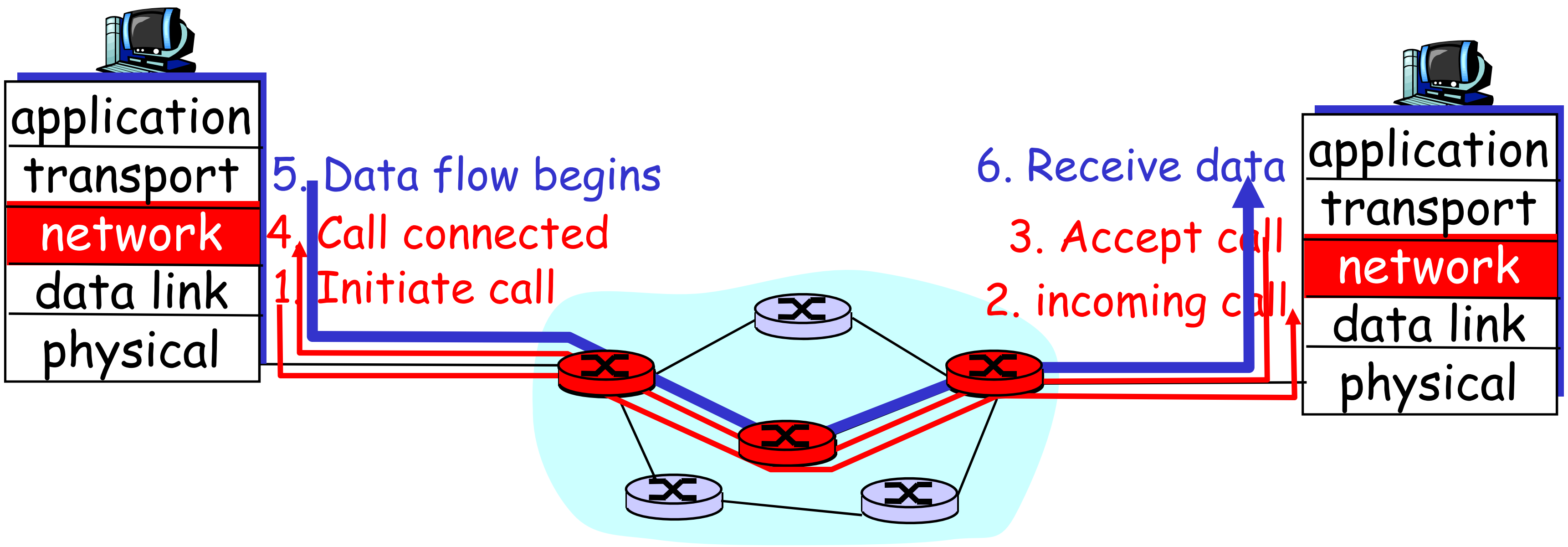
The most important abstraction provided by network layer:

virtual circuit
or
datagram?

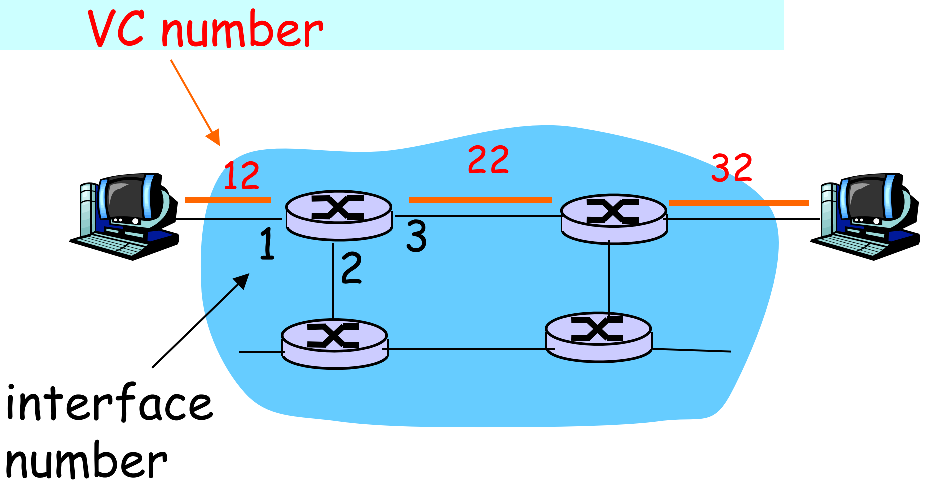
Virtual circuits:

"source-to-dest path behaves almost like telephone circuit"

- call setup, teardown for each call *before* data can flow
  - signaling protocols to setup, maintain teardown VC (ATM, frame-relay, X.25; not in IP)
- each packet carries VC identifier (not destination host)
- *every* router maintains "state" for each passing connection
- resources (bandwidth, buffers) may be *allocated* to VC



Forwarding table in a VC network



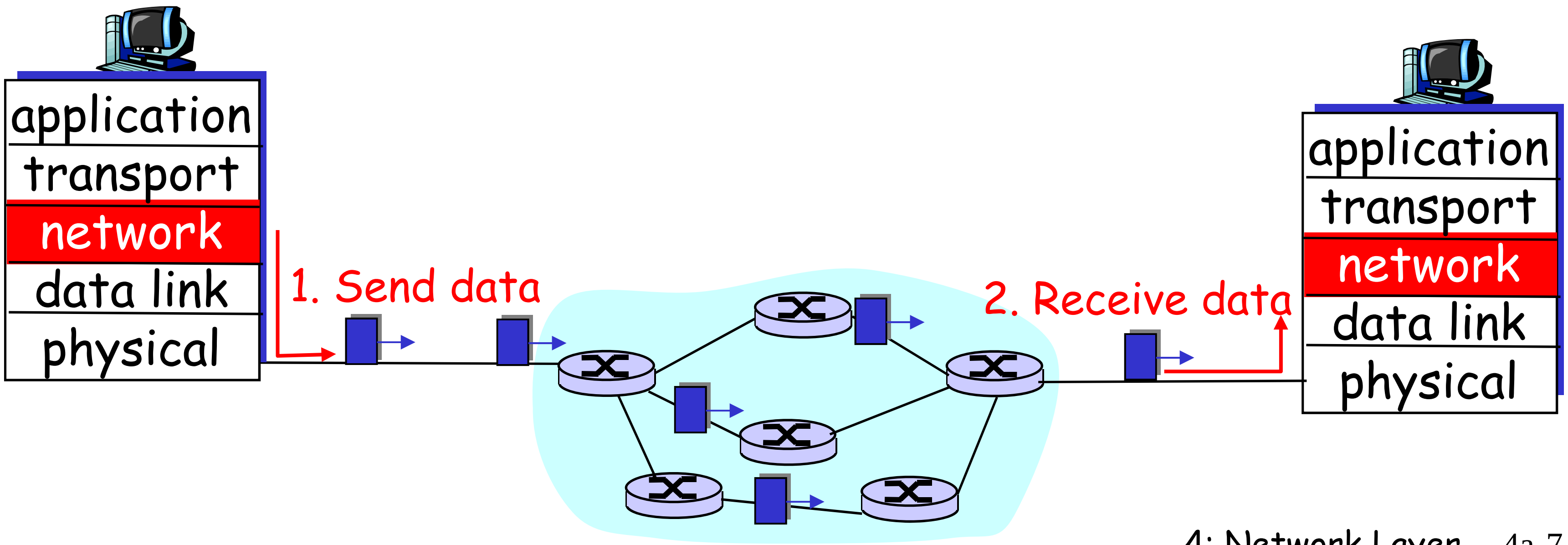
Forwarding table in northwest router:

Incoming interface	Incoming VC #	Outgoing interface	Outgoing VC #
1	12	3	22
2	63	1	18
3	7	2	17
1	97	3	87
…	…	…	…

Routers maintain connection state information!

Datagram networks: the Internet model

- no call setup at network layer
- routers: no state about end-to-end connections
  - no network-level concept of “connection”
- packets typically routed using destination host ID
  - packets between same source-dest pair may take different paths



Forwarding table in a datagram network

4 billion possible entries

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 in datagram NWs:
in practice by masking: Longest prefix matching

Prefix Match	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?

Chapter 4: Network Layer

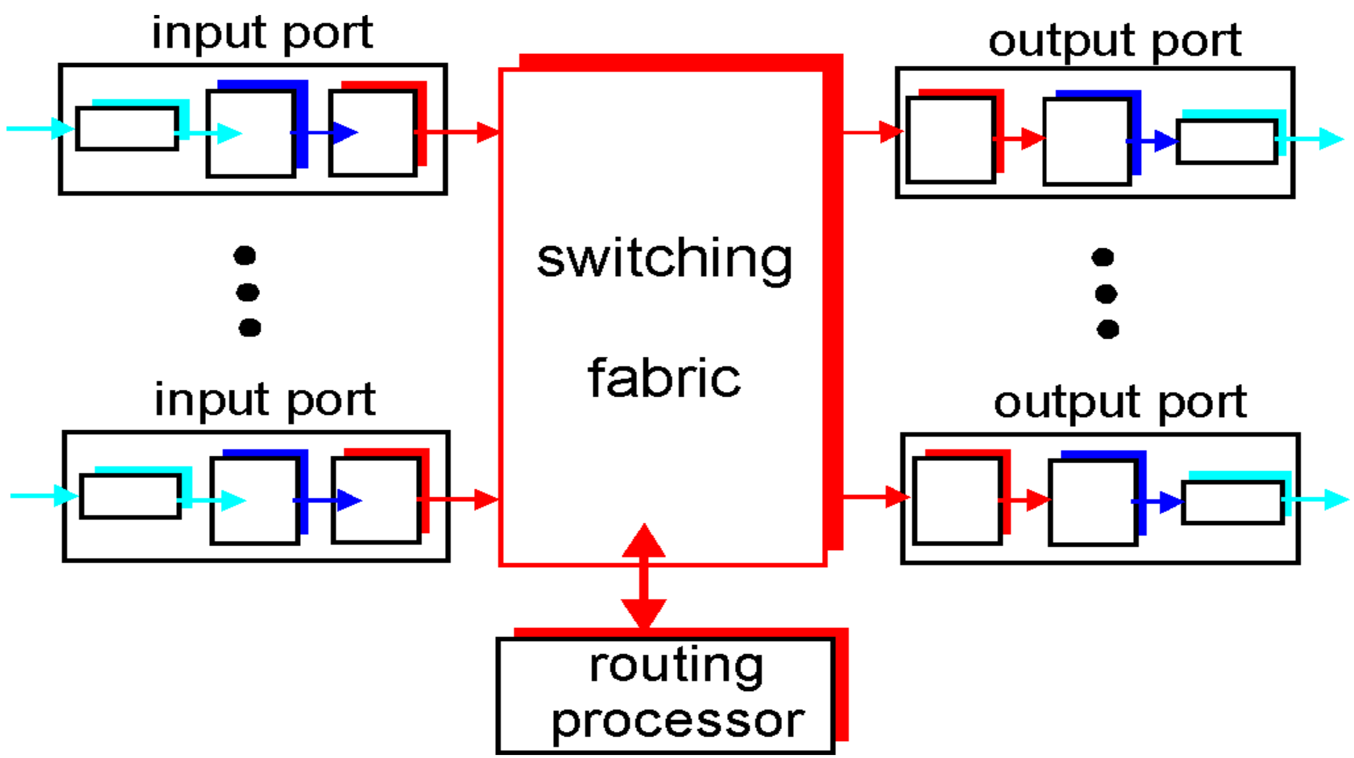
- 4. 1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.4 IP: Internet Protocol
  - Datagram format
  - IPv4 addressing
  - ICMP
  - IPv6
- 4.3 What's inside a router
- 4.5 Routing algorithms
  - Link state
  - Distance Vector
  - Hierarchical routing
- 4.6 Routing in the Internet
  - RIP
  - OSPF
  - BGP
- 4.7 Broadcast and multicast routing

Router Architecture Overview

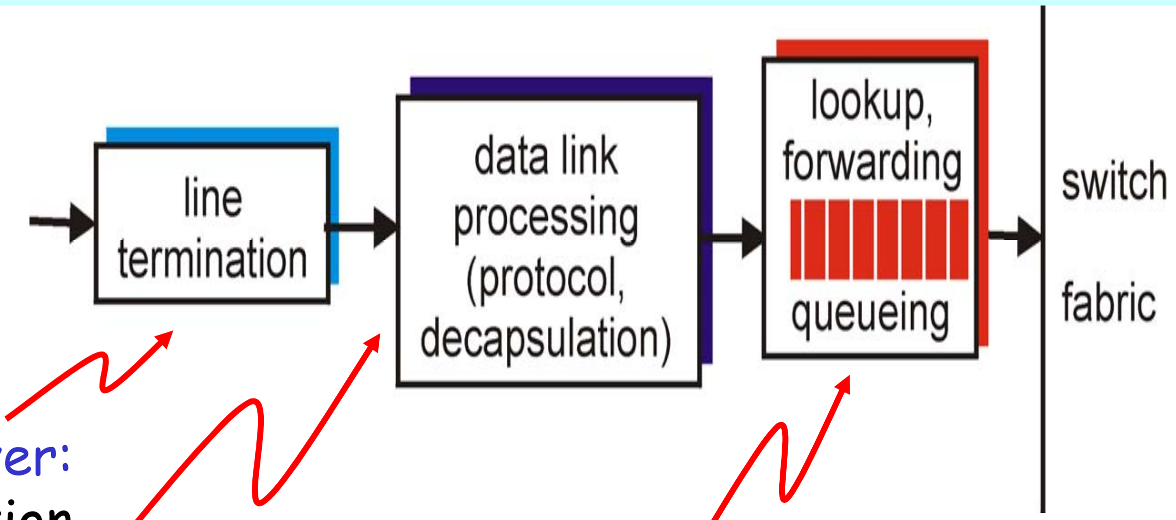
Router Architecture Overview

Two key router functions:

- run routing algorithms/protocol
- *switching packets* from incoming to outgoing link



Input Port Functions



Physical layer:
bit-level reception

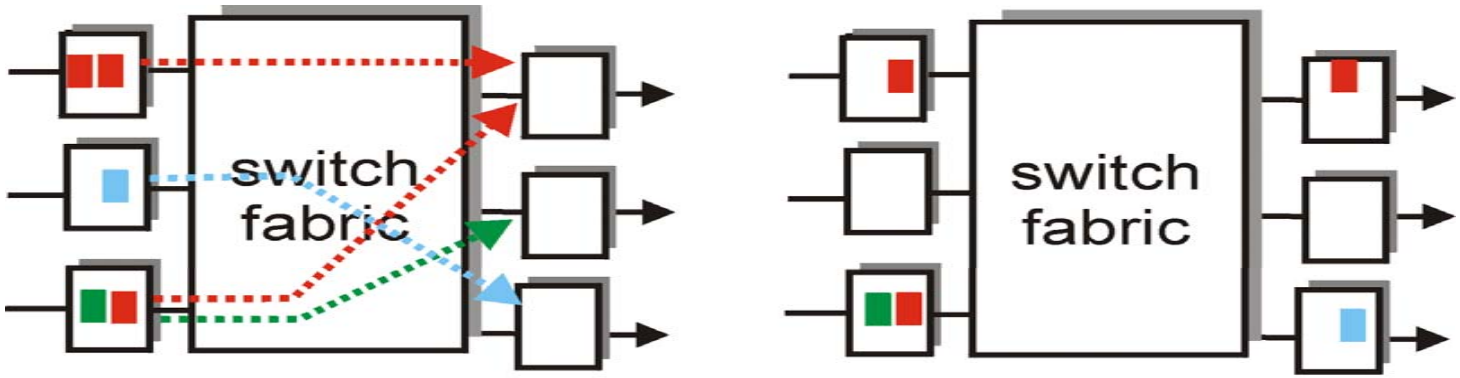
Data link layer:
e.g., Ethernet
see chapter 5

Decentralized switching:

- given datagram dest., lookup output port using routing table in input port memory
- goal: complete input port processing at 'line speed'
- queuing: if datagrams arrive faster than forwarding rate into switch fabric

Input Port Queuing

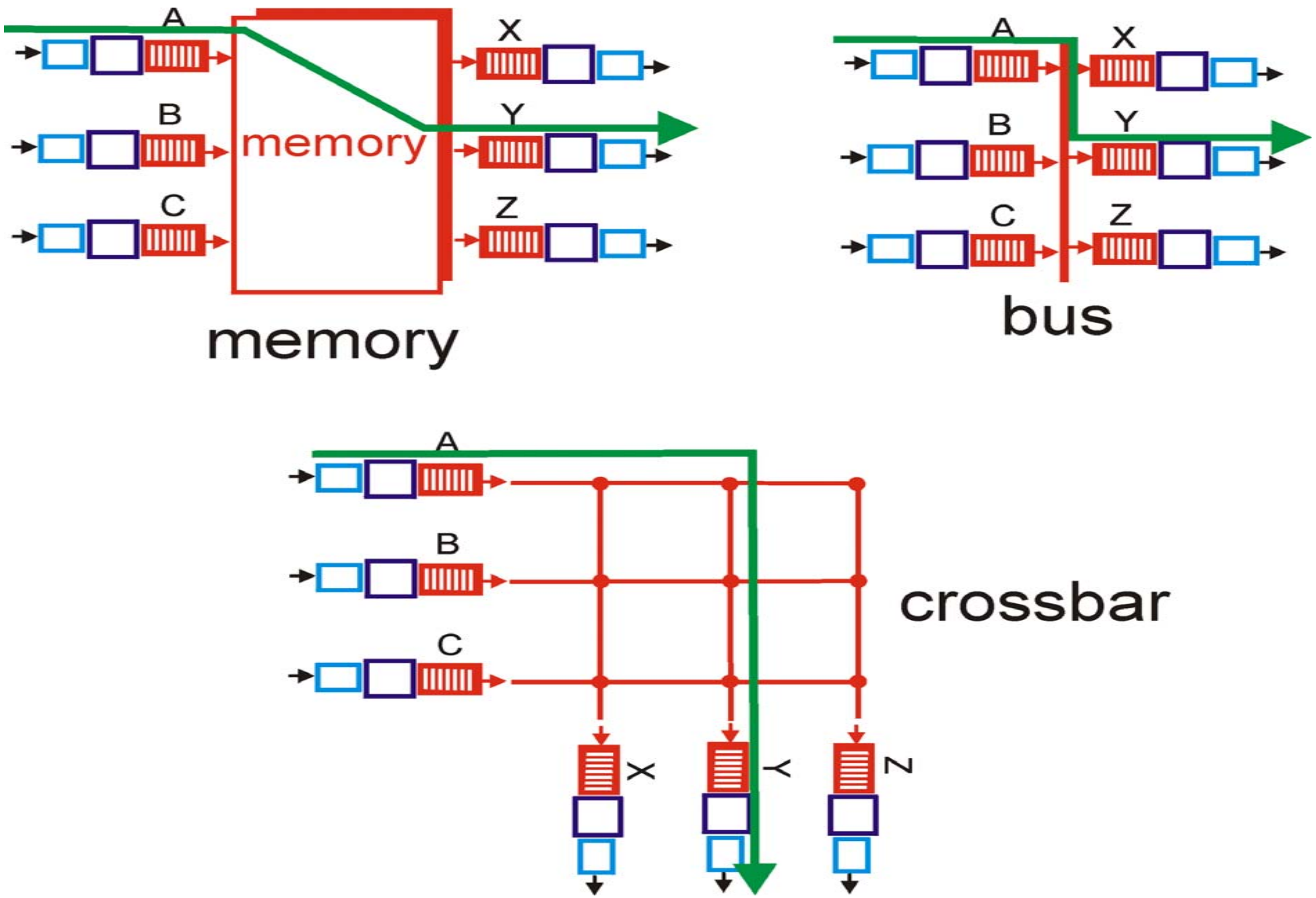
- Fabric slower that input ports combined -> queueing may occur at input queues
- Head-of-the-Line blocking: queued datagram at front of queue prevents others in queue from moving forward
- *queueing delay and loss due to input buffer overflow!*



output port contention at time t - only one red packet can be transferred

green packet experiences HOL blocking

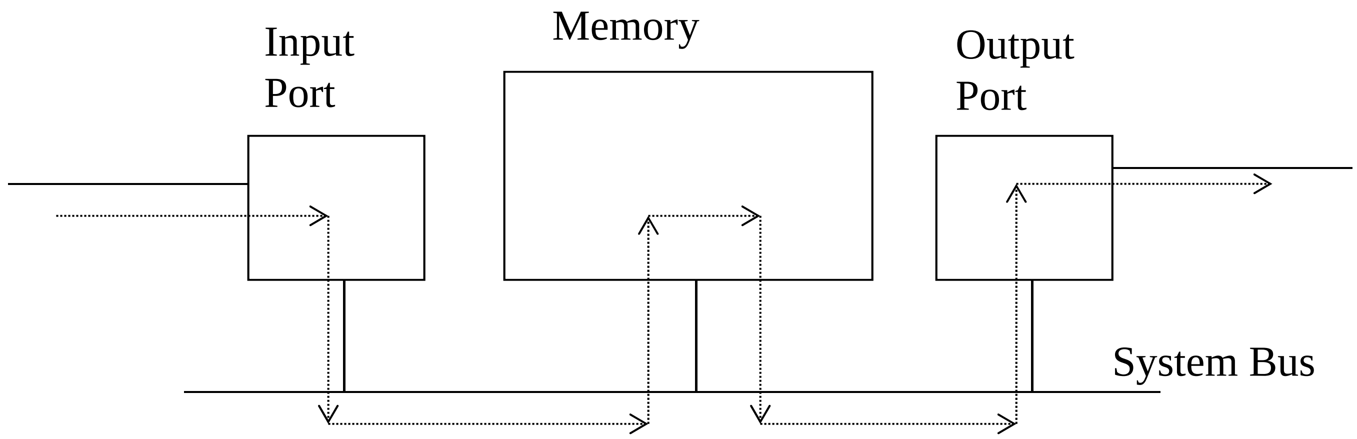
Three types of switching fabrics



Switching Via Memory

First generation routers:

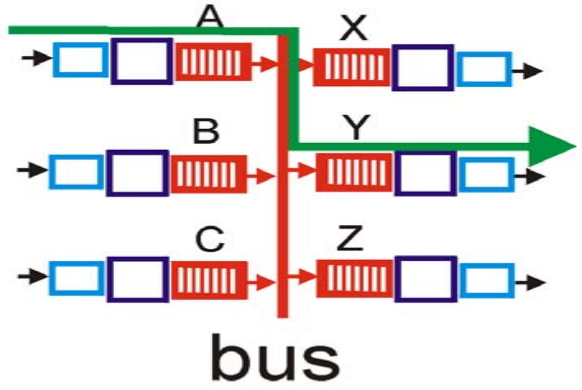
- packet copied by system's (single) CPU
- speed limited by memory bandwidth (2 bus crossings per datagram)



Modern routers:

- input port processor performs lookup, copy into memory
- Cisco Catalyst 8500

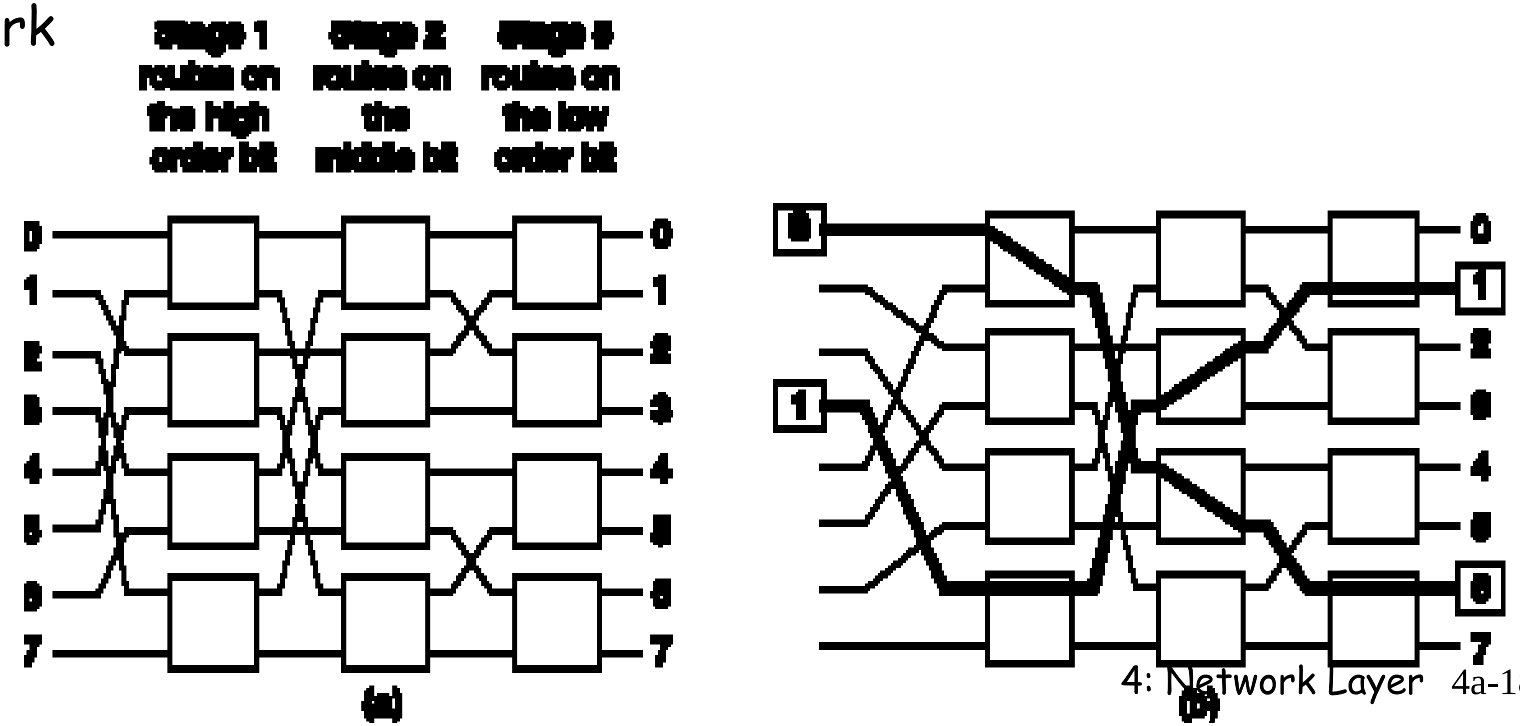
Switching Via Bus



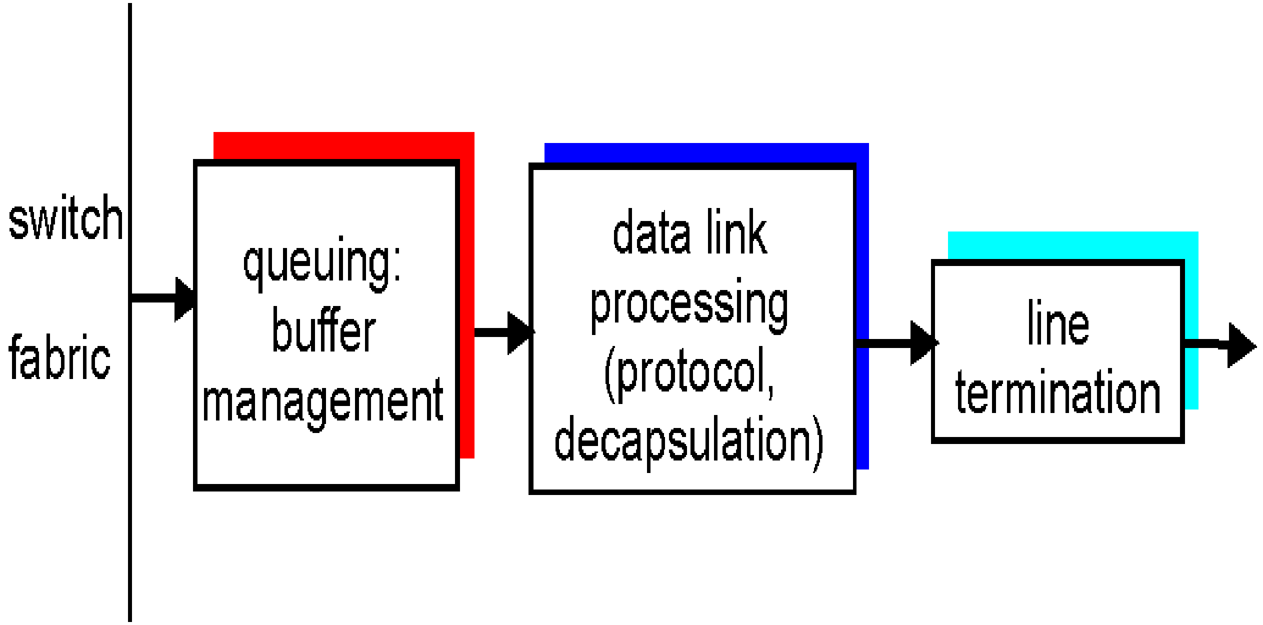
- datagram from input port memory to output port memory via a shared bus
- bus contention: switching speed limited by bus bandwidth
- 1 Gbps bus, Cisco 1900: sufficient speed for access and enterprise routers (not regional or backbone)

Switching Via An Interconnection Network

- Overcome bus bandwidth limitations
- Banyan networks, other interconnection nets (also used in processors-memory interconnects in multiprocessors), see eg
- Advanced design: fragmenting datagram into fixed length cells, switch cells through the fabric (ATM-network principle).
- Cisco 12000: switches 60 Gbps through the interconnection network

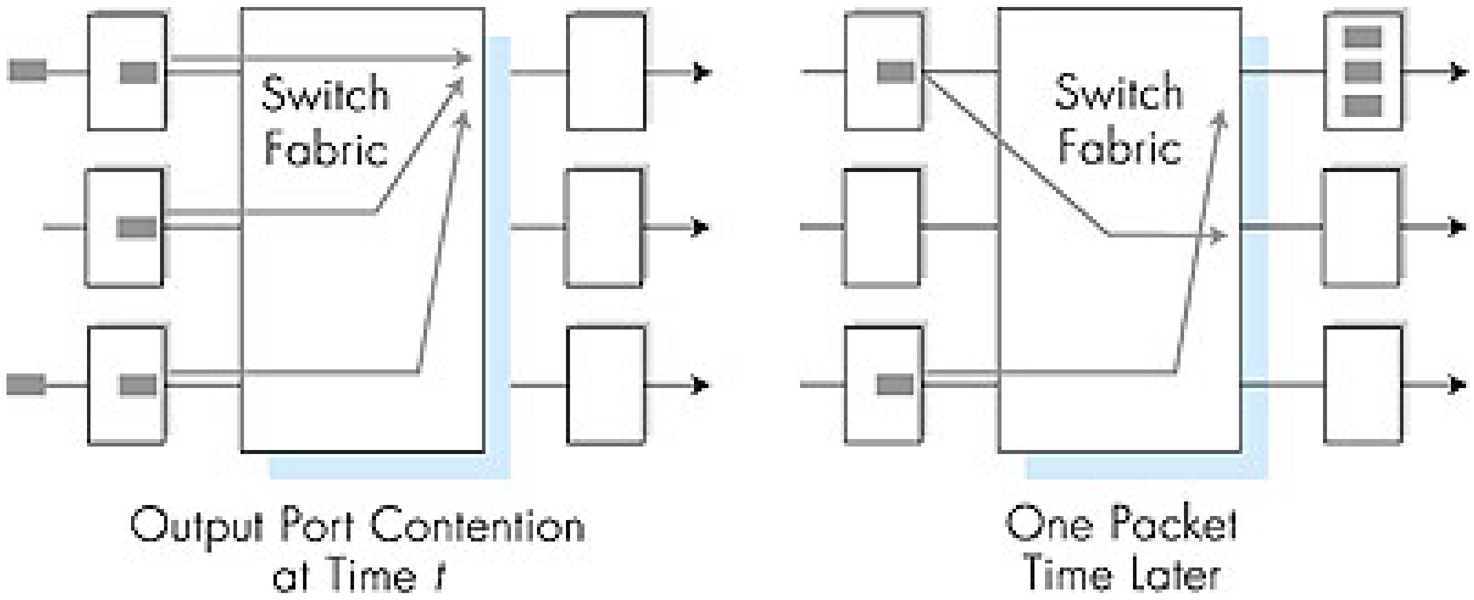


Output Ports



- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Scheduling discipline* chooses among queued datagrams for transmission (cf. QoS guarantees, to be discussed in multimedia context)

Output port queueing



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

Roadmap

Chapter goals:

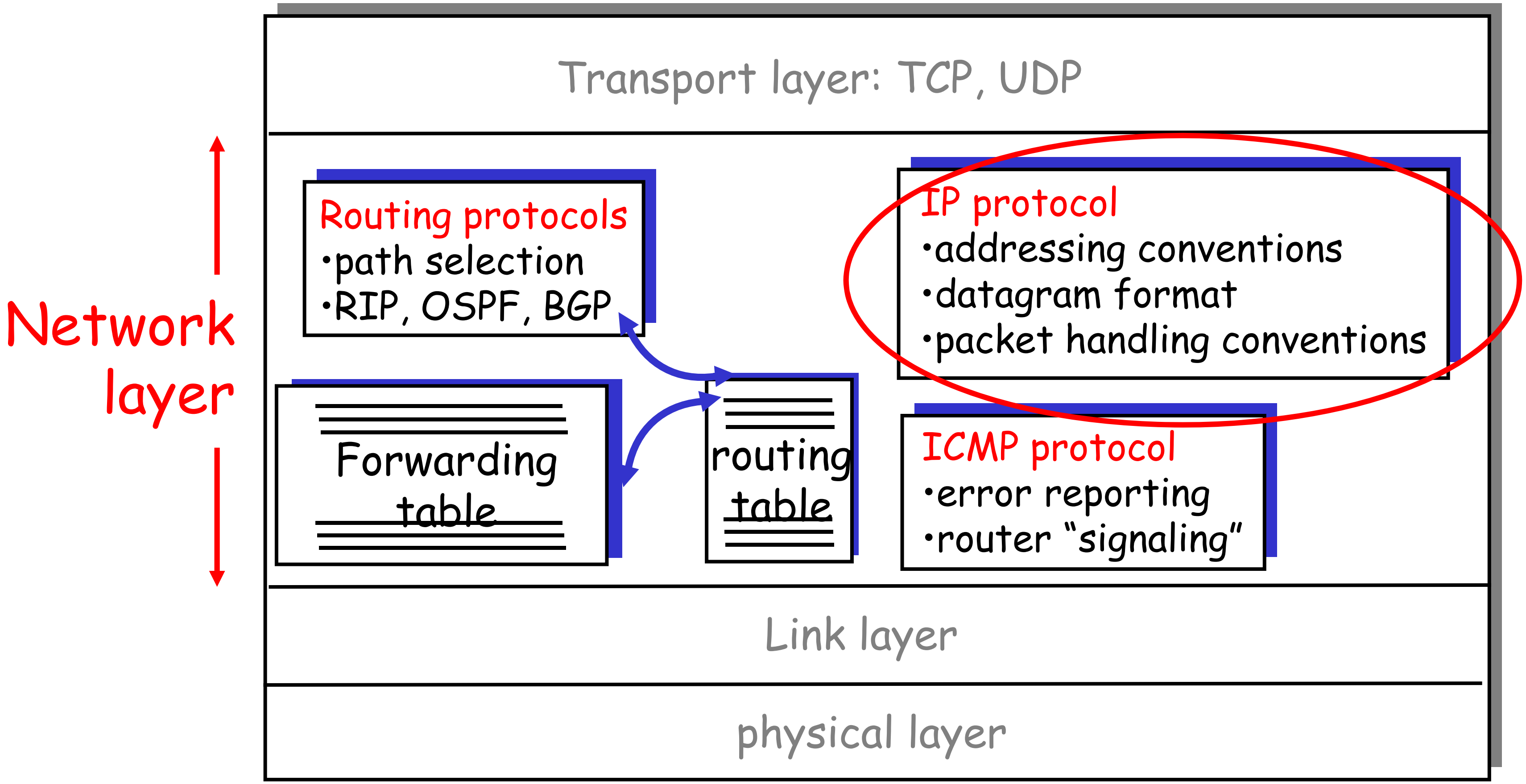
- understand principles behind network layer services:
  - how a router works
  - routing (path selection)
  - dealing with scale
- instantiation and implementation in the Internet (incl. IPv6, multicast)

Overview:

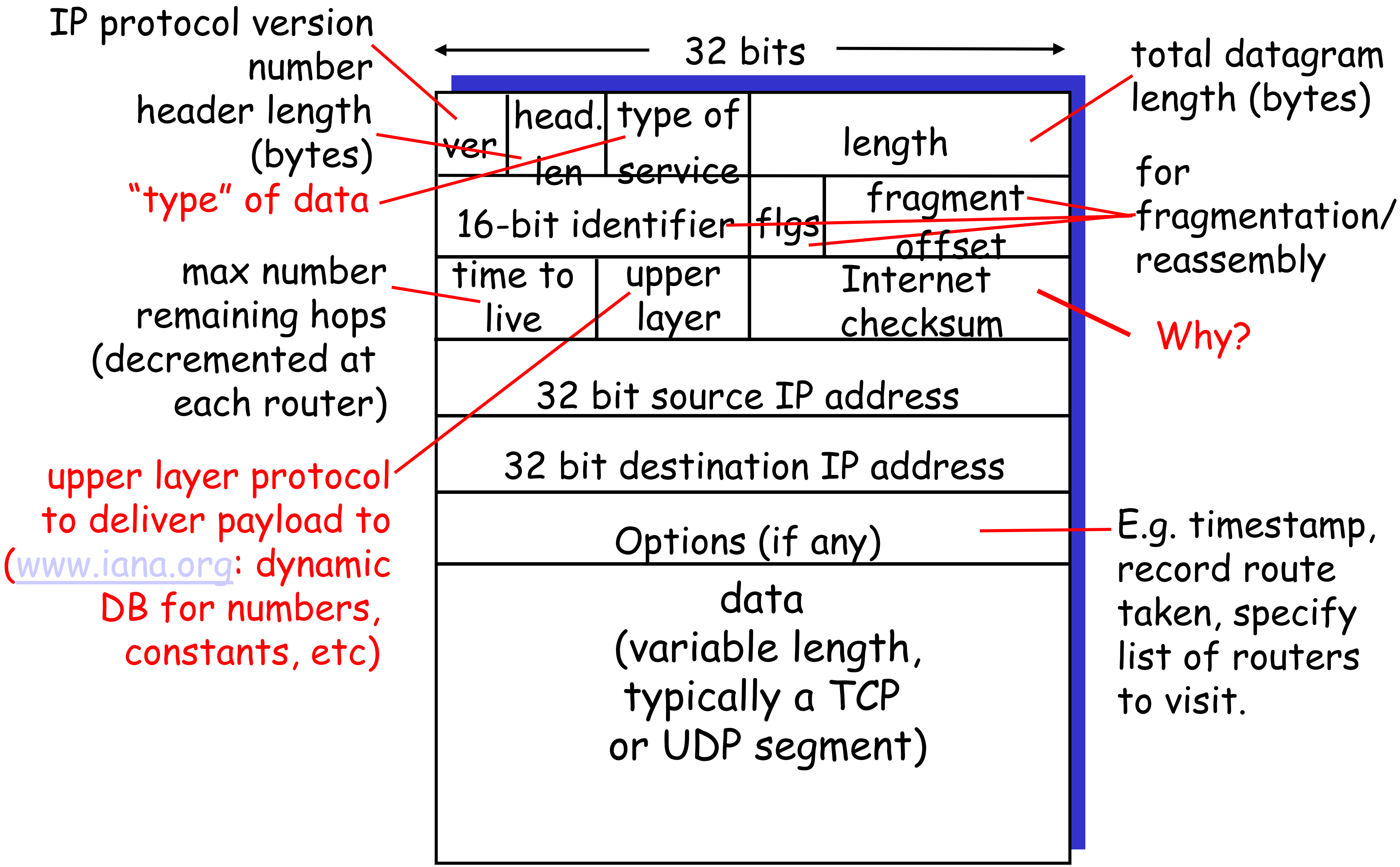
- network layer services
  - VC, datagram
- what's inside a router?
- **Addressing, forwarding, IP**
- routing principle: path selection
  - hierarchical routing
  - Internet routing protocols

The Internet Network layer

(Host or router) network layer functions:

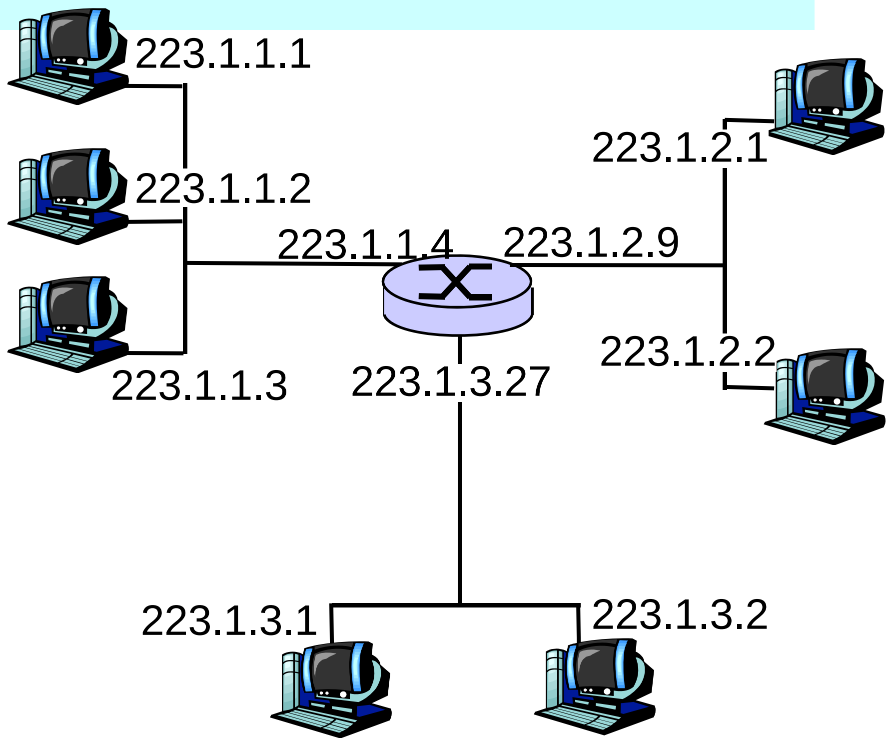


IPv4 datagram format



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 interface
  - IP addresses associated with each interface

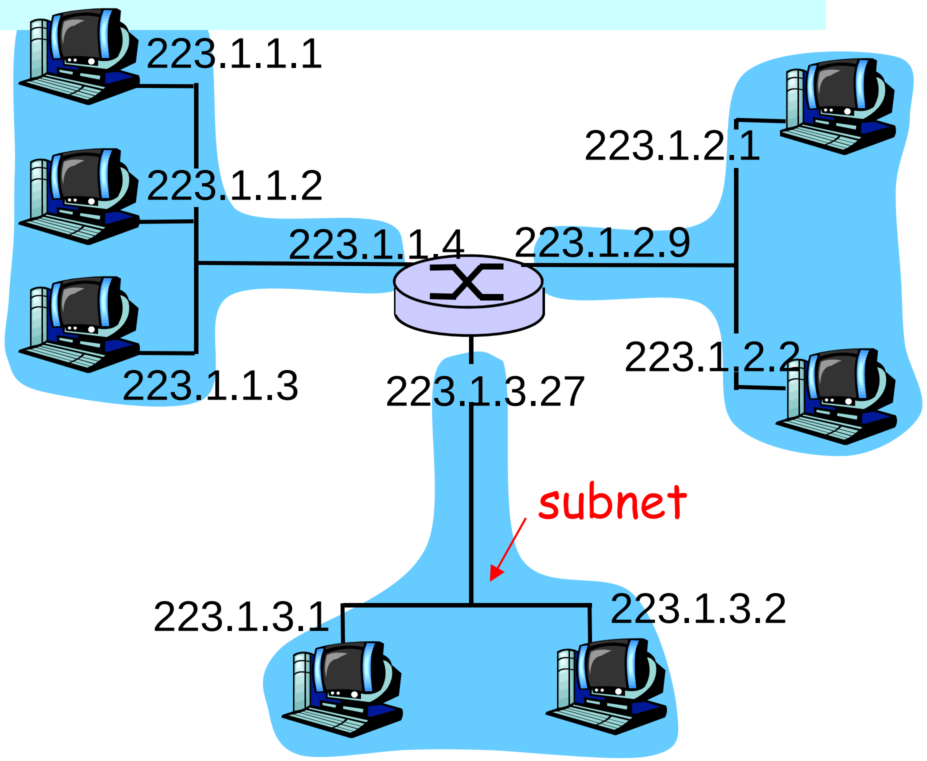


223.1.1.1 = 11011111 00000001 00000001 00000001

223	1	1	1

Subnets

- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- *What's a subnet ?*
  - device interfaces with same subnet-part in their IP addresses
  - can physically reach each other without intervening router

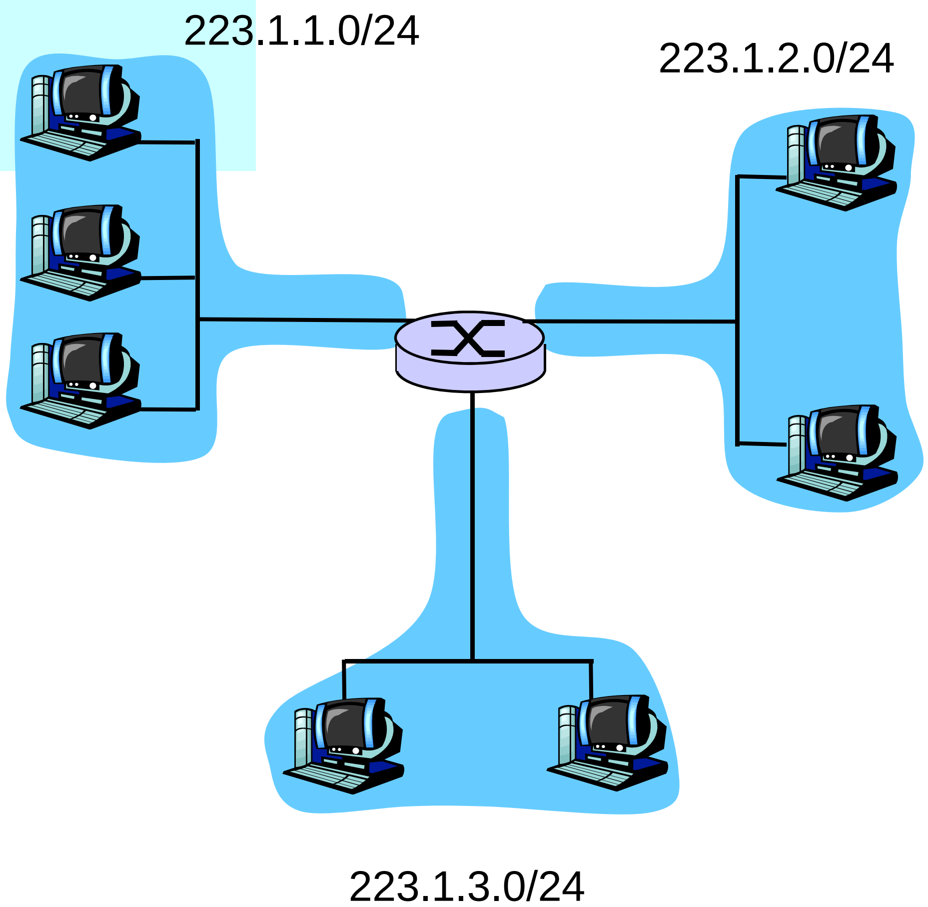


network consisting of 3 subnets

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: /24

IP addressing: CIDR

CIDR: Classless InterDomain Routing

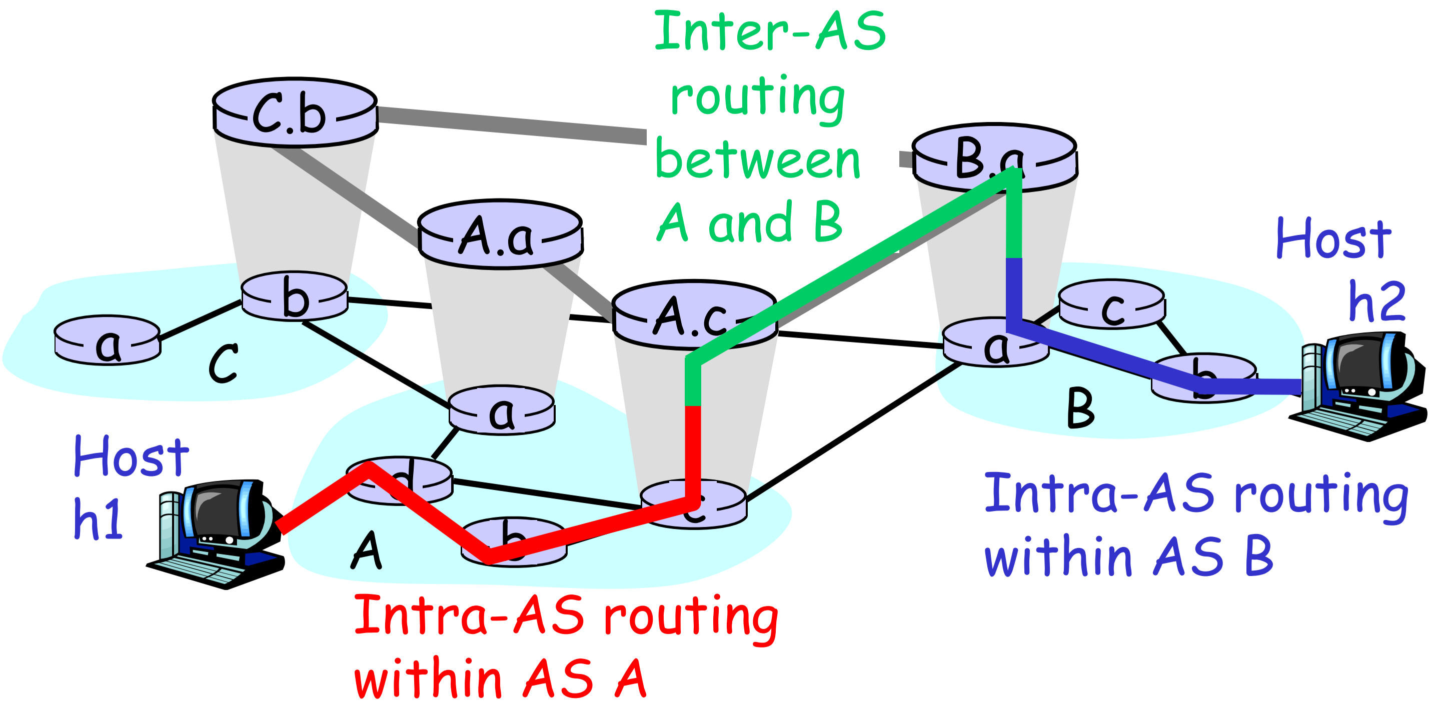
- subnet portion of address of arbitrary length
- address format: a.b.c.d/x, where x is # bits in subnet portion of address

subnet part | host part

11001000 00010111 00010000 00000000

200.23.16.0/23

Internet hierarchical routing



scale: with 50 million destinations:

- can't store all dest's in routing tables!
- routing table exchange would swamp links!

- We'll examine Internet routing algorithms and protocols shortly

IP addresses: how to get one?

Host portion:

- hard-coded by system admin in a file; or
- DHCP: Dynamic Host Configuration Protocol: dynamically get address:
  - host broadcasts "DHCP discover" msg
  - DHCP server responds with "DHCP offer" msg
  - host requests IP address: "DHCP request" msg
  - DHCP server sends address: "DHCP ack" msg

IP addresses: how to get one?

Network portion:

- get allocated portion of ISP's address space:

ISP's block	11001000 00010111 00010000 00000000	200.23.16.0/20
Organization 0	11001000 00010111 00010000 00000000	200.23.16.0/23
Organization 1	11001000 00010111 00010010 00000000	200.23.18.0/23
Organization 2	11001000 00010111 00010100 00000000	200.23.20.0/23
...	….. ….	….
Organization 7	11001000 00010111 00011110 00000000	200.23.30.0/23

IP addressing: the last word...

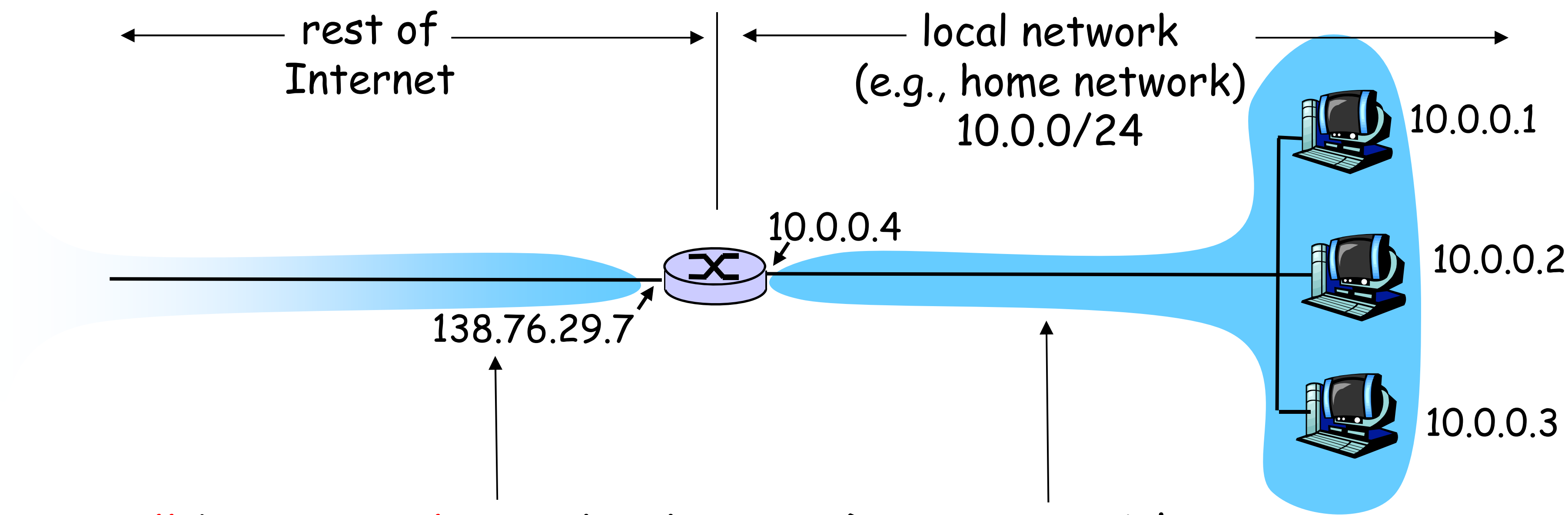
Q: How does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers

- allocates addresses
- manages DNS
- assigns domain names, resolves disputes

Well, it was not really the last word…

NAT: Network Address Translation



All datagrams *leaving* local network have same single source NAT IP address: 138.76.29.7, different source port numbers

Datagrams with source or destination in this network have 10.0.0/24 address for source, destination (as usual)

NAT: Network Address Translation

- Motivation: local network uses just one IP address as far as outside world is concerned:
  - range of addresses not needed from ISP: just one IP address for all devices
  - can change addresses of devices in local network without notifying outside world
  - can change ISP without changing addresses of devices in local network
  - devices inside local net not explicitly addressable, visible by outside world (a security plus).

NAT: Network Address Translation

Implementation: NAT router must:

- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)

  . . . remote clients/servers will respond using (NAT IP address, new port #) as destination addr.

- *remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, new port #) translation pair

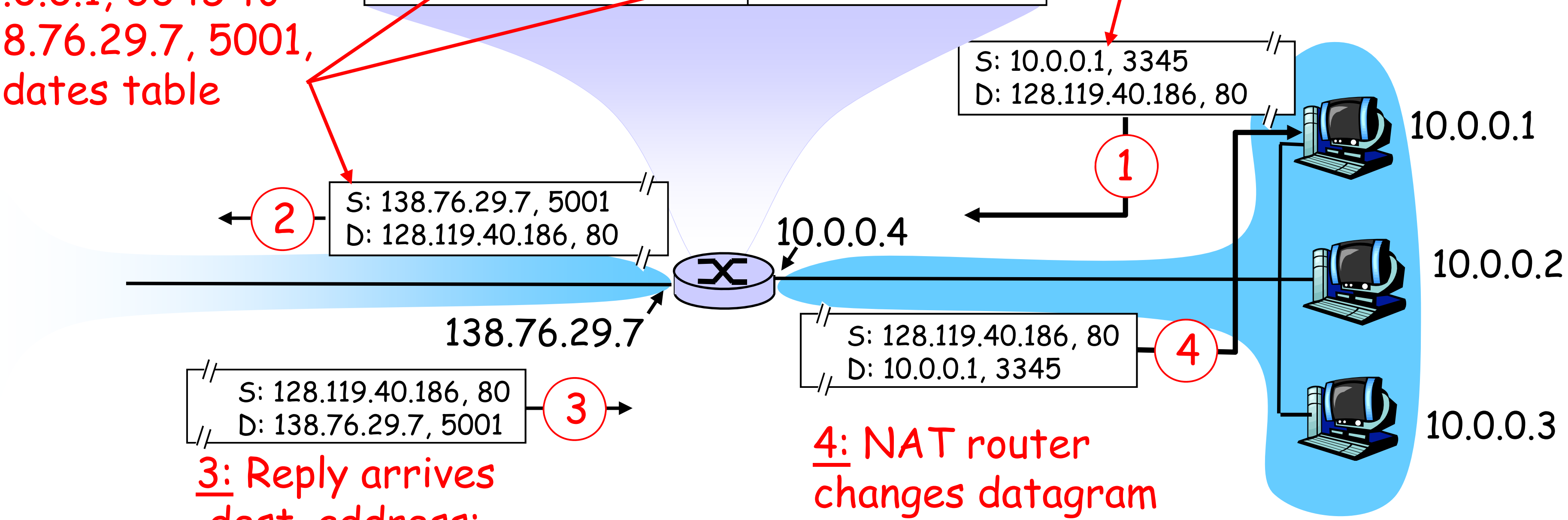
- *incoming datagrams: replace* (NAT IP address, new port #) in dest fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT table

NAT: Network Address Translation

NAT translation table	
WAN side addr	LAN side addr
138.76.29.7, 5001	10.0.0.1, 3345
……	……

1: host 10.0.0.1 sends datagram to 128.119.40.186, 80

2: NAT router changes datagram source addr from 10.0.0.1, 3345 to 138.76.29.7, 5001, updates table



3: Reply arrives dest. address: 138.76.29.7, 5001

4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

NAT: Network Address Translation

- 16-bit port-number field:
  - 60,000 simultaneous connections with a single LAN-side address!
- NAT is controversial:
  - routers should only process up to layer 3
  - violates end-to-end argument
    - NAT possibility must be taken into account by app designers, eg, P2P applications

Getting a datagram from source to dest.

Getting a datagram from source to dest.

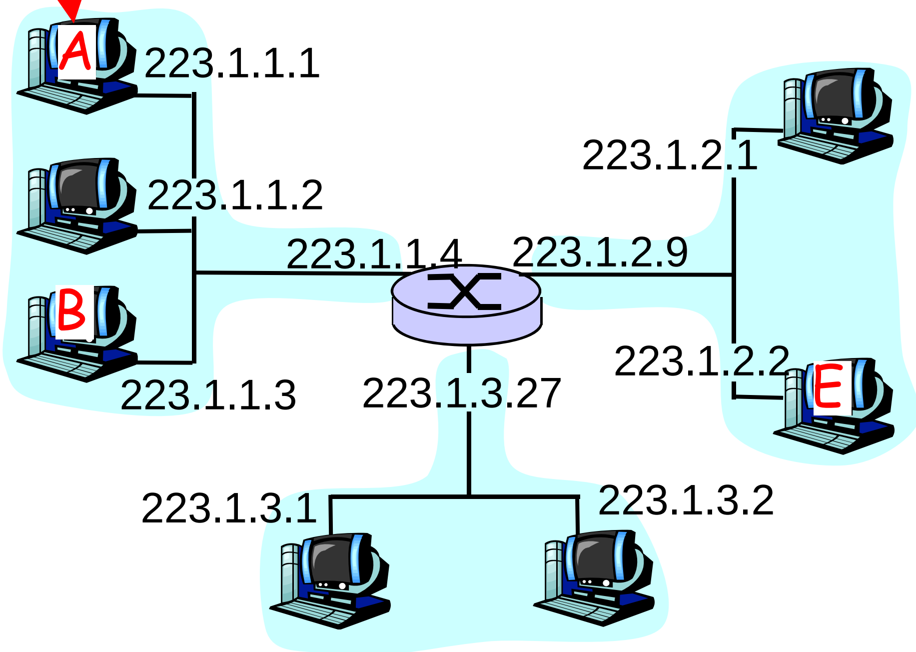
forwarding table in A

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2

IP datagram:

misc fields	source IP addr	dest IP addr	data

- datagram remains unchanged, as it travels source to destination
- addr fields of interest here



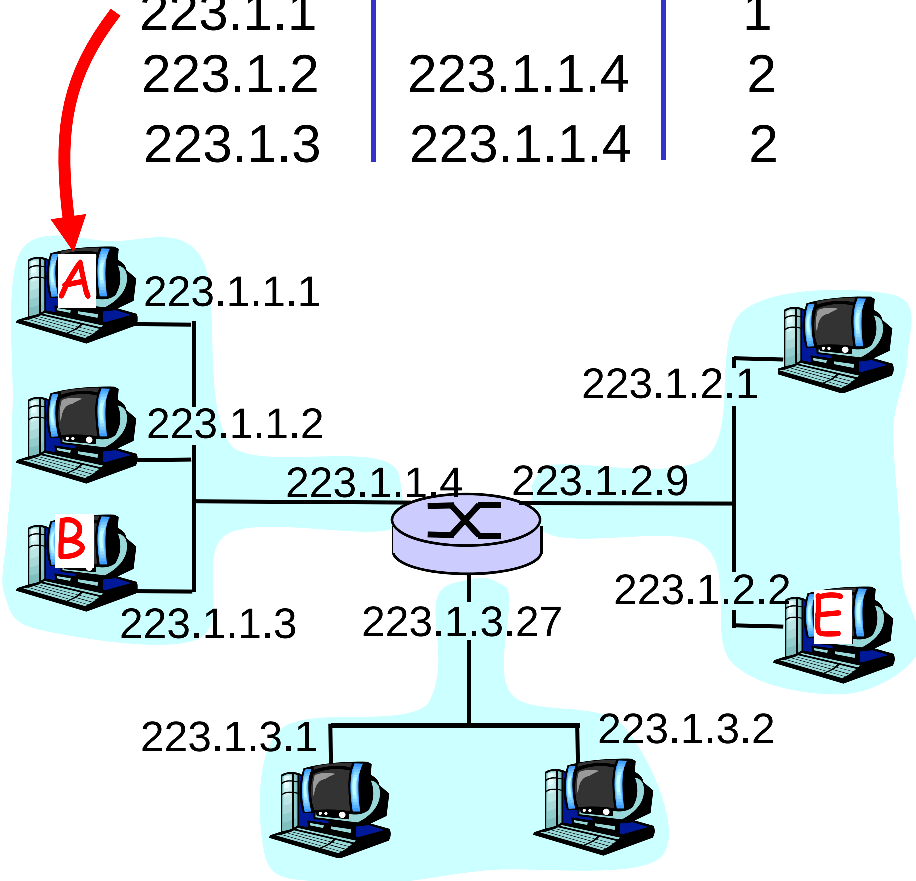
Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.1.3	data

Starting at A, given IP datagram addressed to B:

- look up net. address of B
- find B is on same net. as A (B and A are directly connected)
- link layer will send datagram directly to B (inside link-layer frame)

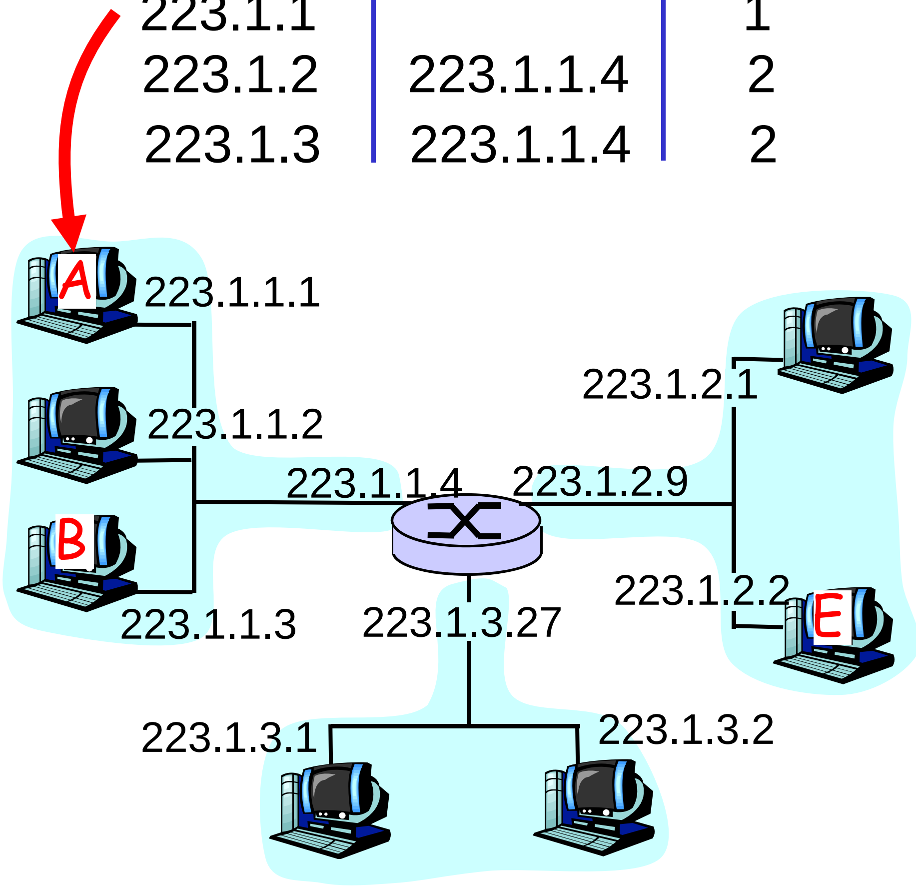
Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.2.3	data

Dest. Net.	next router	Nhops
223.1.1		1
223.1.2	223.1.1.4	2
223.1.3	223.1.1.4	2



Starting at A, dest. E:

- look up network address of E
- E on *different* network
- routing table: next hop router to E is 223.1.1.4
- link layer is asked to send datagram to router 223.1.1.4 (inside link-layer frame)
- datagram arrives at 223.1.1.4
- continued…..

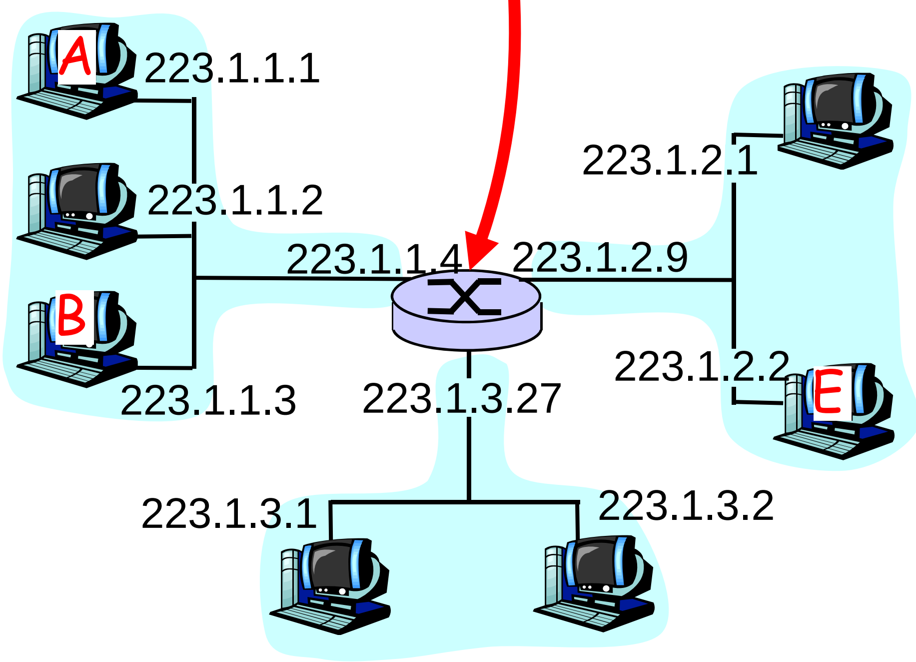
Getting a datagram from source to dest.

misc fields	223.1.1.1	223.1.2.3	data

Dest. network	next router	Nhops	interface
223.1.1	-	1	223.1.1.4
223.1.2	-	1	223.1.2.9
223.1.3	-	1	223.1.3.27

Arriving at 223.1.4, destined for 223.1.2.2

- look up network address of E
- E on *same* network as router's interface 223.1.2.9
  - router, E directly attached
- link layer sends datagram to 223.1.2.2 (inside link-layer frame) via interface 223.1.2.9
- datagram arrives at 223.1.2.2!!! (hooray!)

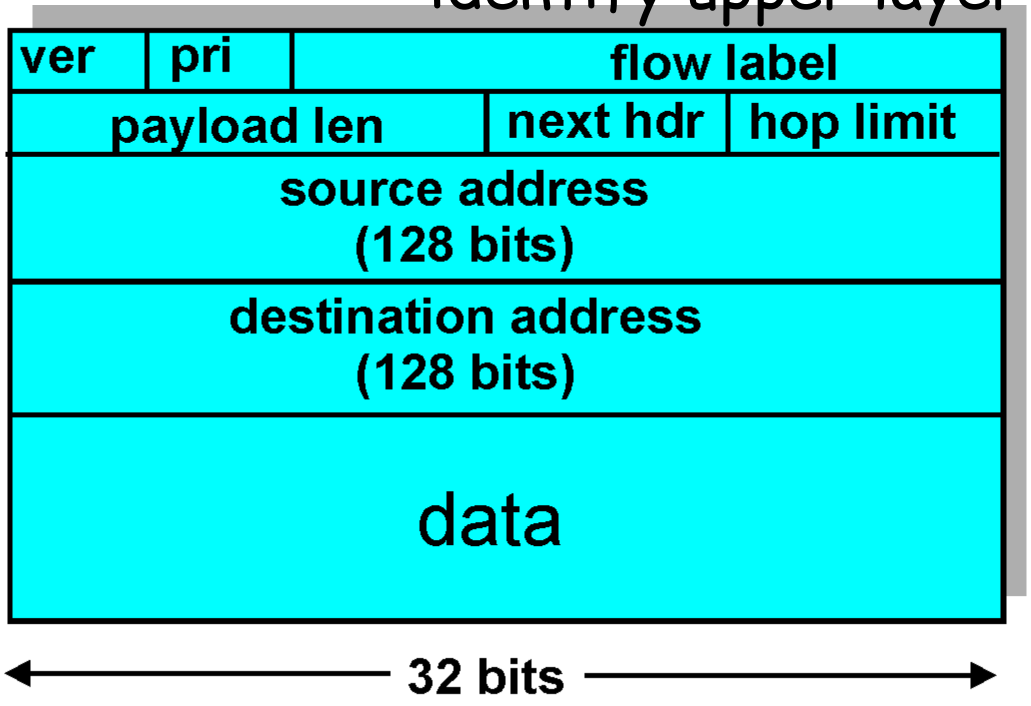


IPv6

- Initial motivation: *prediction:* 32-bit address space completely allocated by approx. 2008.
- Additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate provisioning of services that could guarantee timing, bandwidth
  - new "anycast" address: route to "best" of several replicated servers
- IPv6 datagram format (to speed-up pkt-processing):
  - fixed-length 40 byte header
  - no (intermediate) fragmentation allowed
  - no checksum

IPv6 Header (Cont)

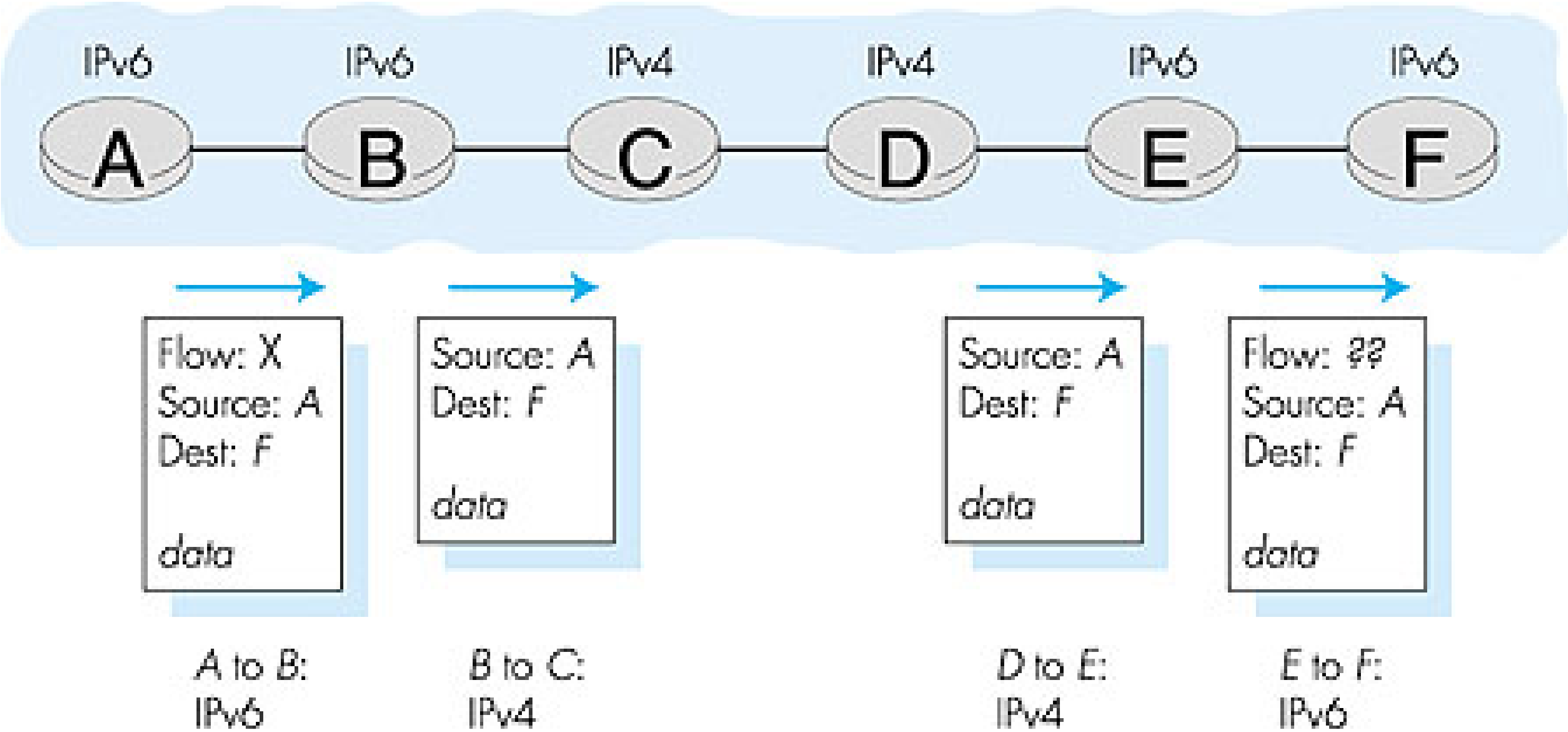
Priority: identify priority among datagrams in flow
Flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).
Next header: (e.g. extend header with info such as identify upper layer protocol for data)



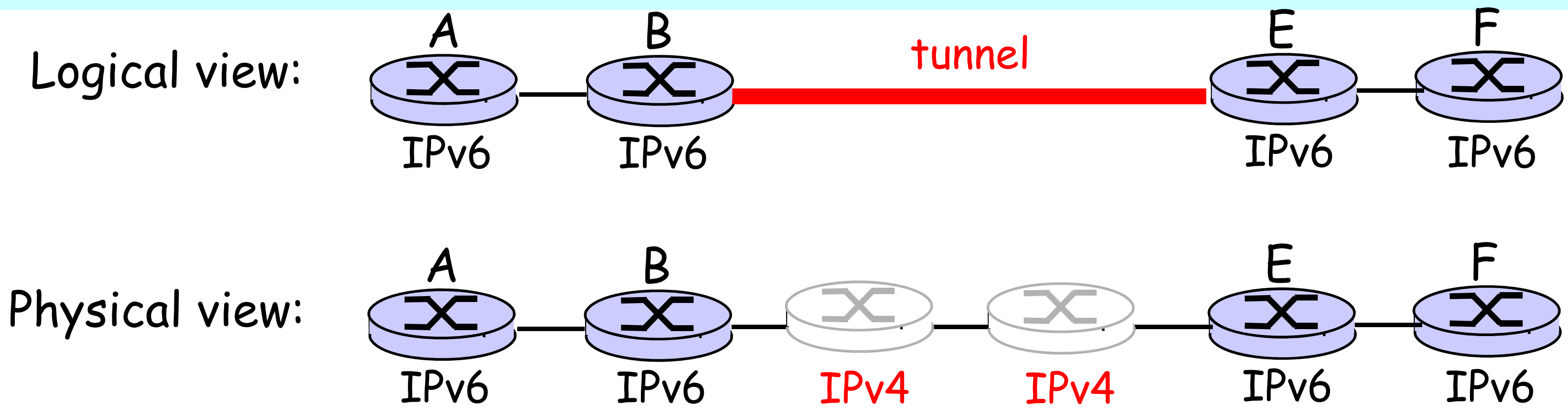
Transition From IPv4 To IPv6

- Not all routers can be upgraded simultaneous
  - no "flag days"
  - How will the network operate with mixed IPv4 and IPv6 routers?
- Two proposed approaches:
  - *Dual Stack*: some routers with dual stack (v6, v4) can "translate" between formats
  - *Tunneling:* IPv6 carried as payload n IPv4 datagram among IPv4 routers

Dual Stack Approach

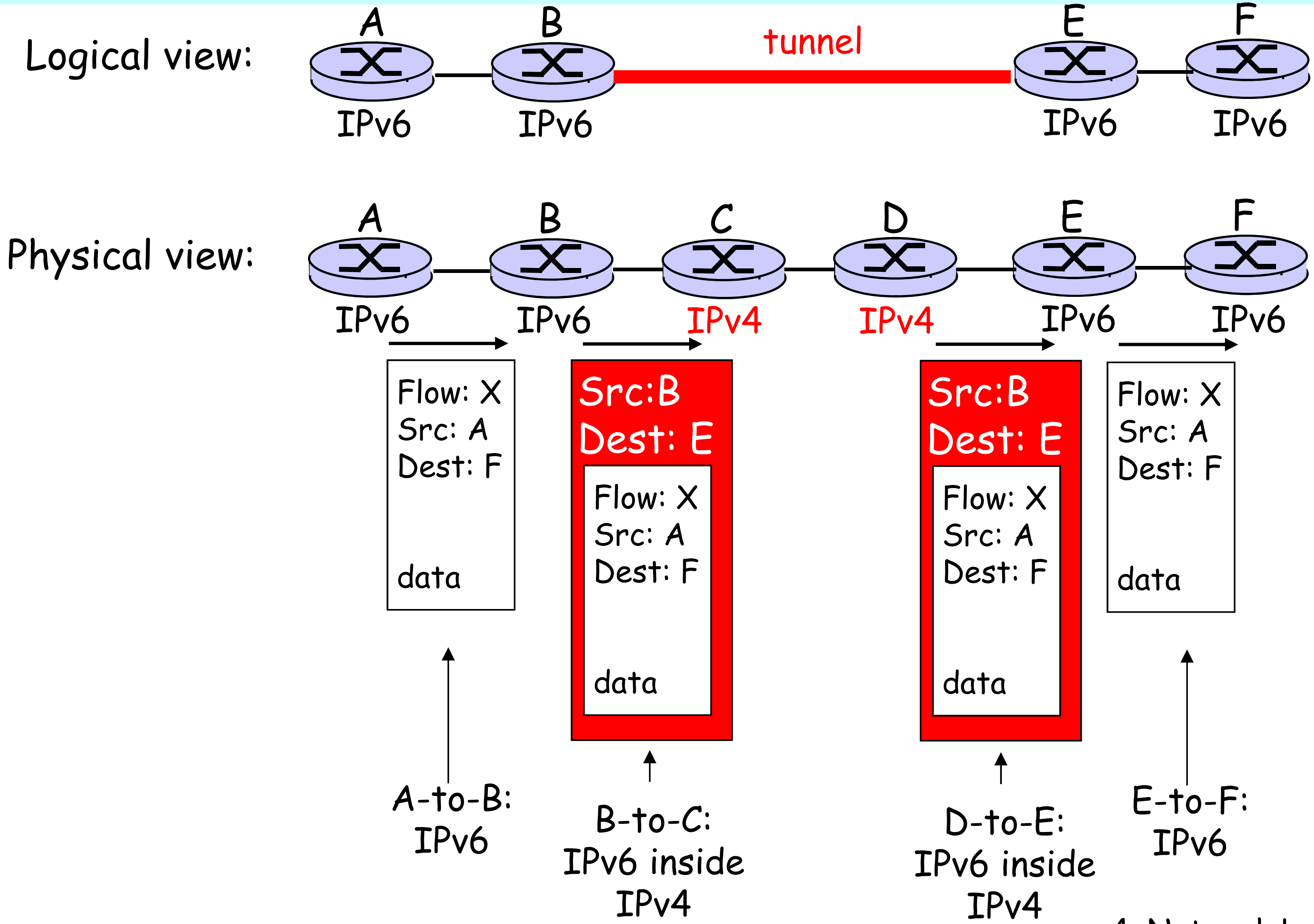


Tunneling



Logical view:
A
B
tunnel
E
F
IPv6
IPv6
IPv6
IPv6
Physical view:
A
B
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6

Tunneling



Logical view:
A
B
tunnel
E
F
IPv6
IPv6
IPv6
IPv6
Physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
Flow: X
Src: A
Dest: F
data
Src:B
Dest: E
Flow: X
Src: A
Dest: F
data
Src:B
Dest: E
Flow: X
Src: A
Dest: F
data
Flow: X
Src: A
Dest: F
data
A-to-B:
IPv6
B-to-C:
IPv6 inside
IPv4
D-to-E:
IPv6 inside
IPv4
E-to-F:
IPv6

ICMP: Internet Control Message Protocol

- used by hosts, routers, gateways to communicate network-level information:
  - error reporting:
  - control: echo request/reply (used by ping), cong. Control (tentative)
- ICMP message: type, code plus first 8 bytes of IP datagram causing error
- network-layer-protocol "above" IP:
  - ICMP msgs carried in IP datagrams
- What if an ICMP message gets lost?

Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
10	0	router discovery
11	0	TTL expired
12	0	bad IP header

Roadmap

Chapter goals:

- understand principles behind network layer services:
  - how a router works
  - routing (path selection)
  - dealing with scale
- instantiation and implementation in the Internet (incl. IPv6, multicast)

Overview:

- network layer services
  - VC, datagram
- what's inside a router?
- **Addressing, forwarding, IP**
- NEXT: routing principle: path selection
  - hierarchical routing
  - Internet routing protocols

Review questions for this part

- Contrast virtual circuit and datagram routing (simplicity, cost, purposes, what service types they may enable)
- Explain the interplay between routing and forwarding
- What is inside a router? How/where do queueing delays happen inside a router? Where/why can packets be dropped at a router?
- What is subnet? What is subnet masking?
- Explain how to get an IP packet from source to destination
- Explain how NAT works.

Extra slides

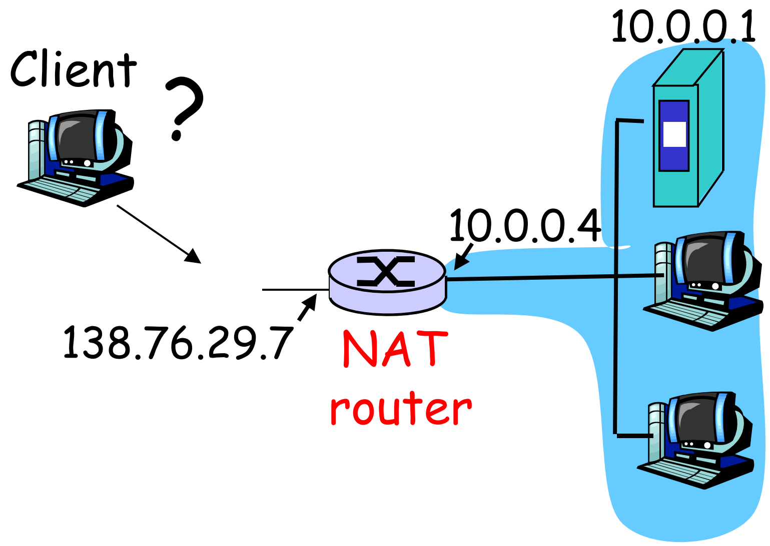
Network layer service models:

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

- Internet model being extented: Intserv, Diffserv
  - (will study these later on)

NAT traversal problem

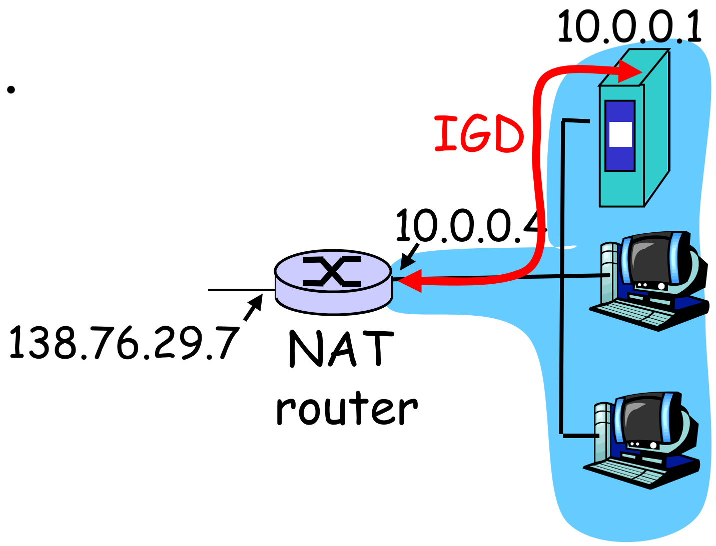
- client want to connect to server with address 10.0.0.1
  - server address 10.0.0.1 local to LAN (client can't use it as destination addr)
  - only one externally visible NATted address: 138.76.29.7
- solution 1 (manual): statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 10.0.0.1 port 2500



NAT traversal problem

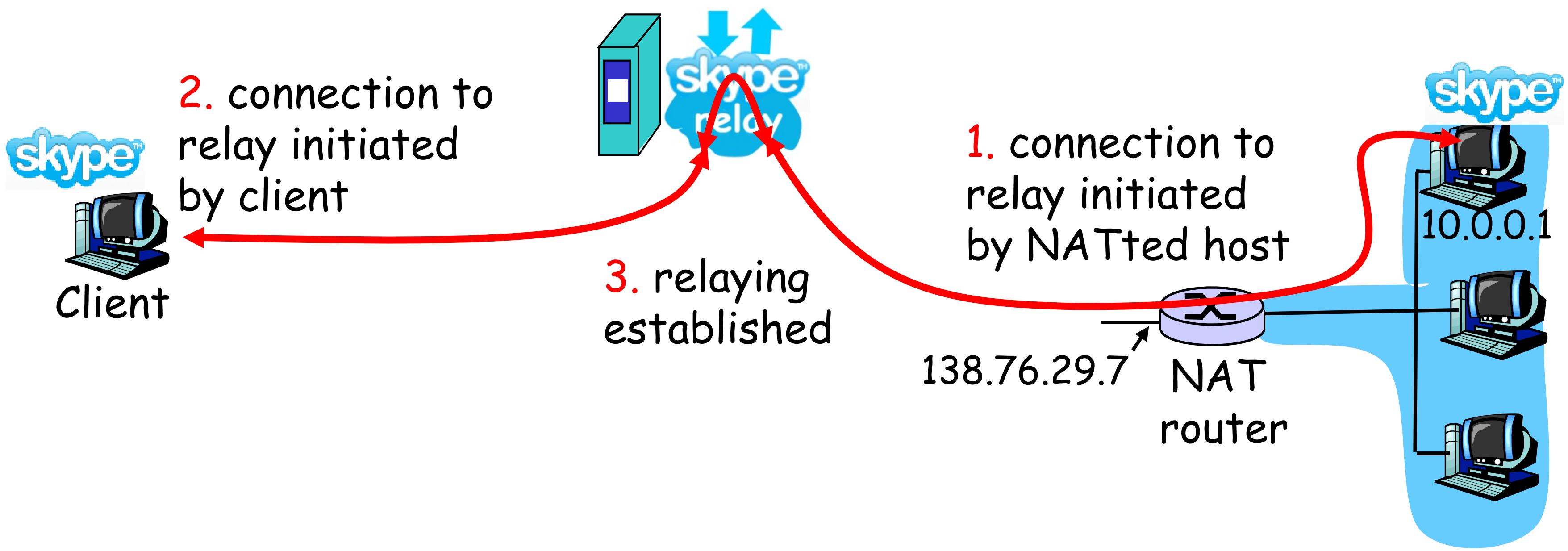
- solution 2 (protocol) : Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATted host to:
  - learn public IP address (138.76.29.7)
  - enumerate existing port mappings
  - add/remove port mappings (with lease times)

i.e., automate static NAT port map configuration



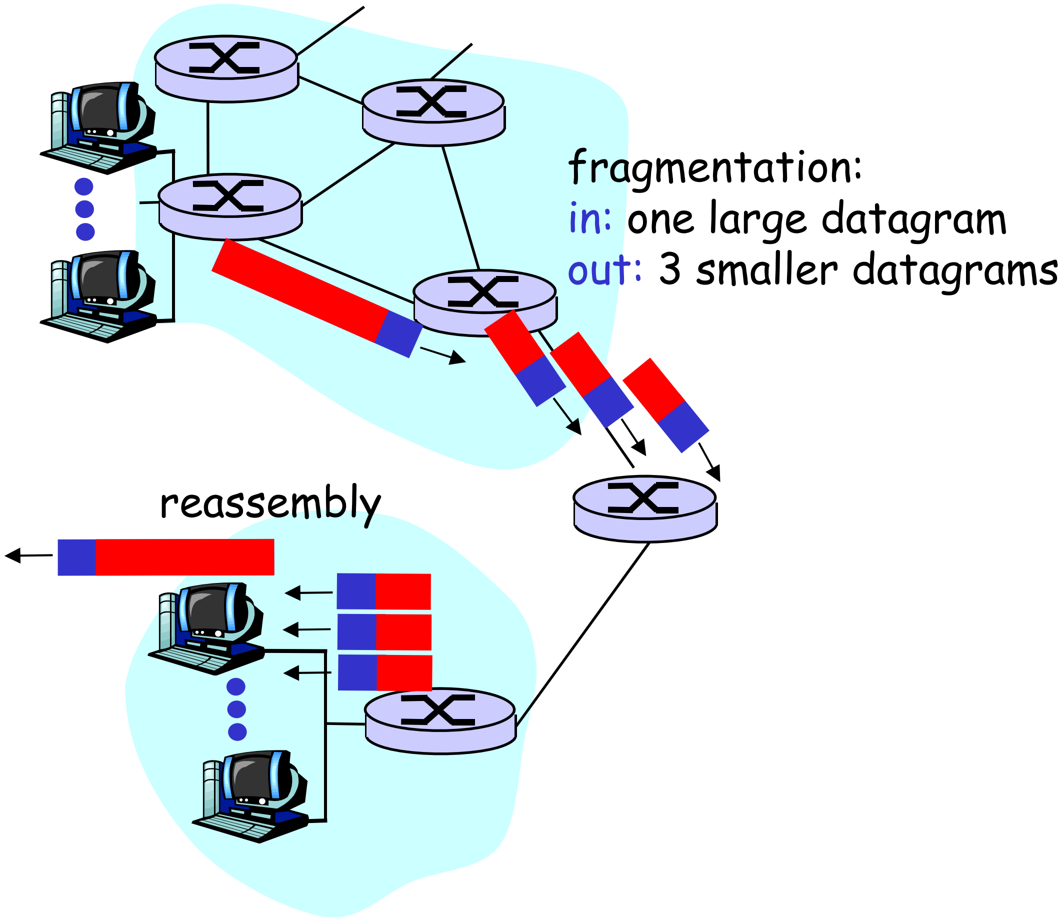
NAT traversal problem

- solution 3 (application): relaying (used in Skype)
  - NATed server establishes connection to relay
  - External client connects to relay
  - relay bridges packets between two connections



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 and Reassembly