Computer Networks 2IC15

Network Layer

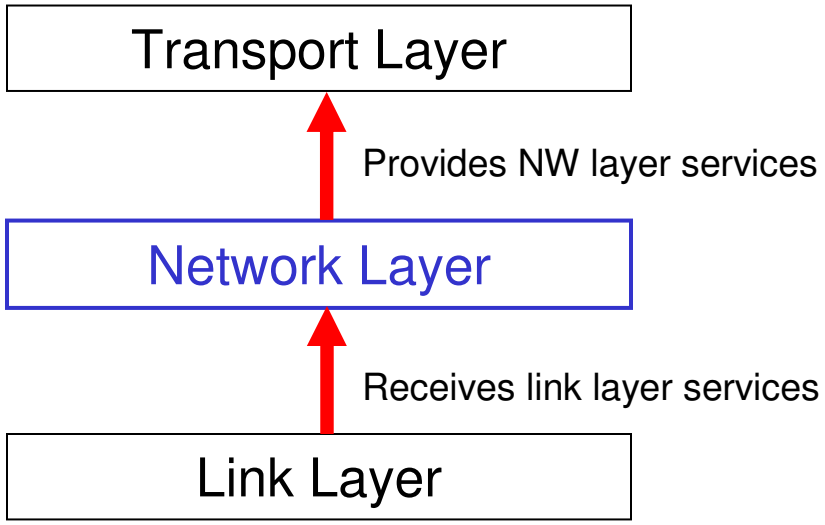
Dr. Tanır Özçelebi

Thanks to
J. F. Kurose
K. W. Ross

Outline

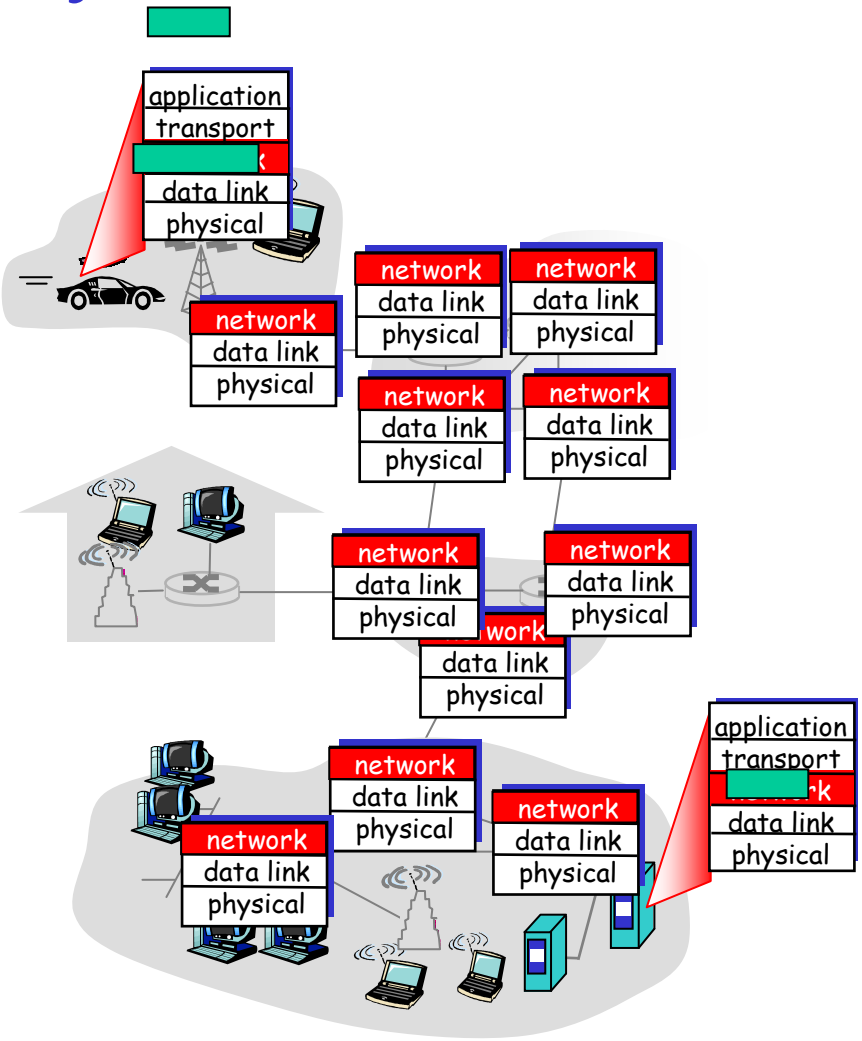
- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
- 4.5 Routing algorithms
- 4.6 Routing protocols in the Internet
- 4.7 Broadcast and multicast routing

Position of the network layer



Network layer

- host-to-host transport
- on sending side encapsulates segments into datagrams
- on rcving side, delivers segments to transport layer
- implemented in *every* host, router
  - unlike upper layers (i.e. application, transport layers)
  - router examines header fields in all IP datagrams passing through it



Tanır Özçelebi, t.ozcelebi@tue.nl
TU/e Computer Science, System Architecture and Networking

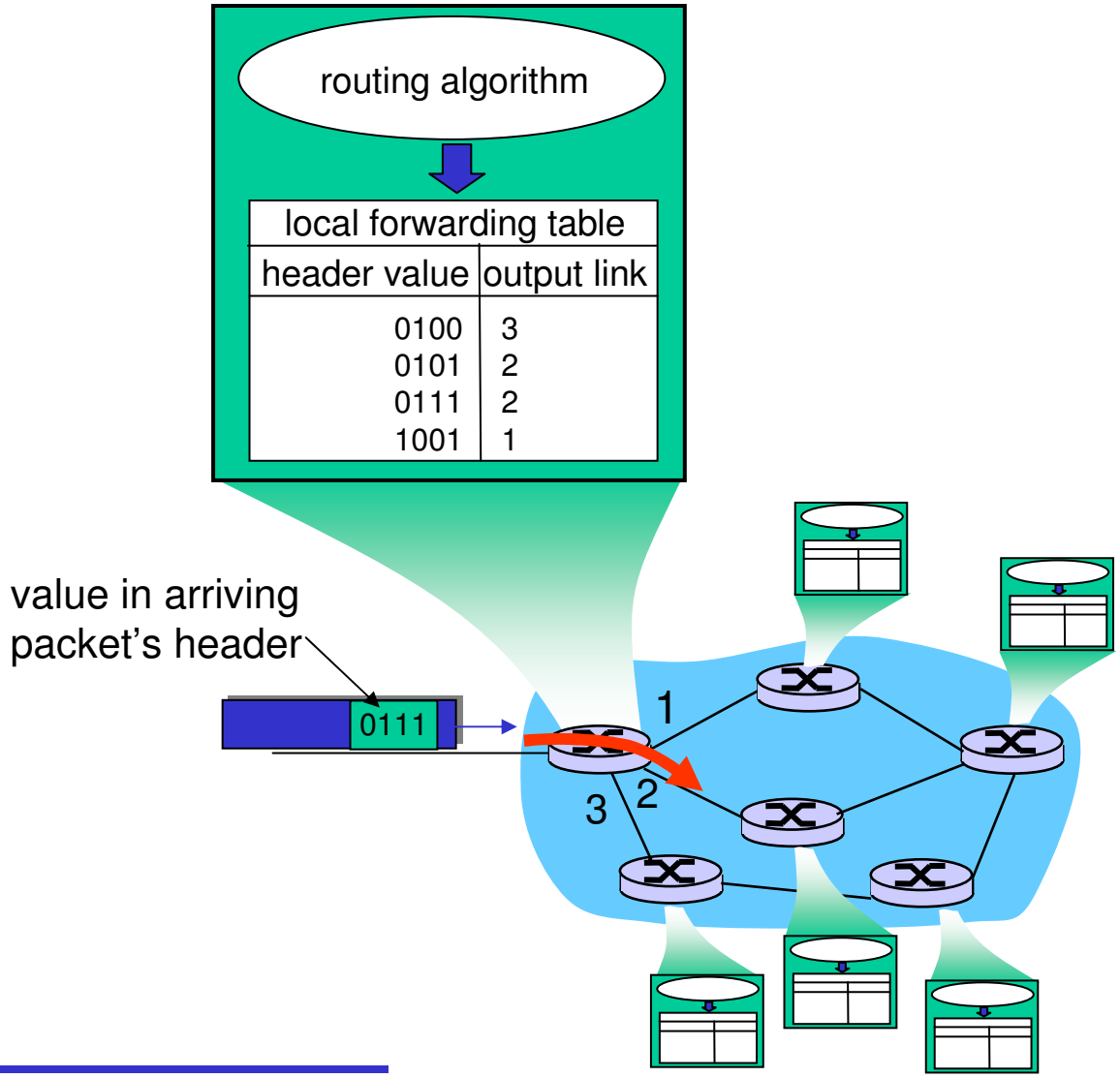
Two Key Network-Layer Functions

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

analogy:

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

Forwarding versus Routing



Connection setup (some networks)

- 3rd important function in *some* network architectures:
  - e.g. ATM, frame-relay, X.25
  - not in the Internet
- before datagrams flow, two end hosts *and* intervening routers establish virtual connection
  - routers get involved
- network vs transport layer connection service:
  - network: between two hosts (may also involve intervening routers in case of VCs)
  - transport: between two processes

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
- guaranteed maximum jitter (changes in inter-packet spacing)

Outline

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
- 4.5 Routing algorithms
- 4.6 Routing protocols in the Internet
- 4.7 Broadcast and multicast routing

Network layer connection and connectionless service

- datagram network: NW-layer connectionless service
- VC network: NW-layer connection service
- analogous to the transport-layer services, but:
  - service: host-to-host
  - no choice: network provides one or the other
  - implementation: in network core

Virtual circuits

"source-to-dest path behaves much like telephone circuit"
- performance-wise
- network actions along source-to-dest path

- call setup, maintenance and teardown for each call
- each packet carries VC identifier (not destination host address)
- *every* router on source-dest path maintains "state" for each passing connection
- link, router resources (bandwidth, buffers) may be *allocated* to VC (dedicated resources = predictable service)

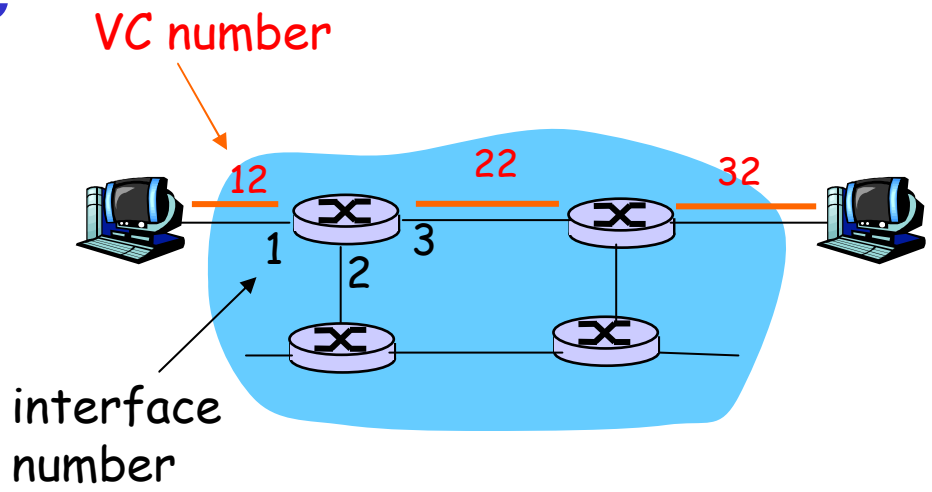
VC implementation

a VC consists of:

1. path from source to destination
2. VC numbers, one number for each link along path
3. entries in forwarding tables in routers along path

- VC number can be changed on each link.
  - New VC number comes from forwarding table

Forwarding table

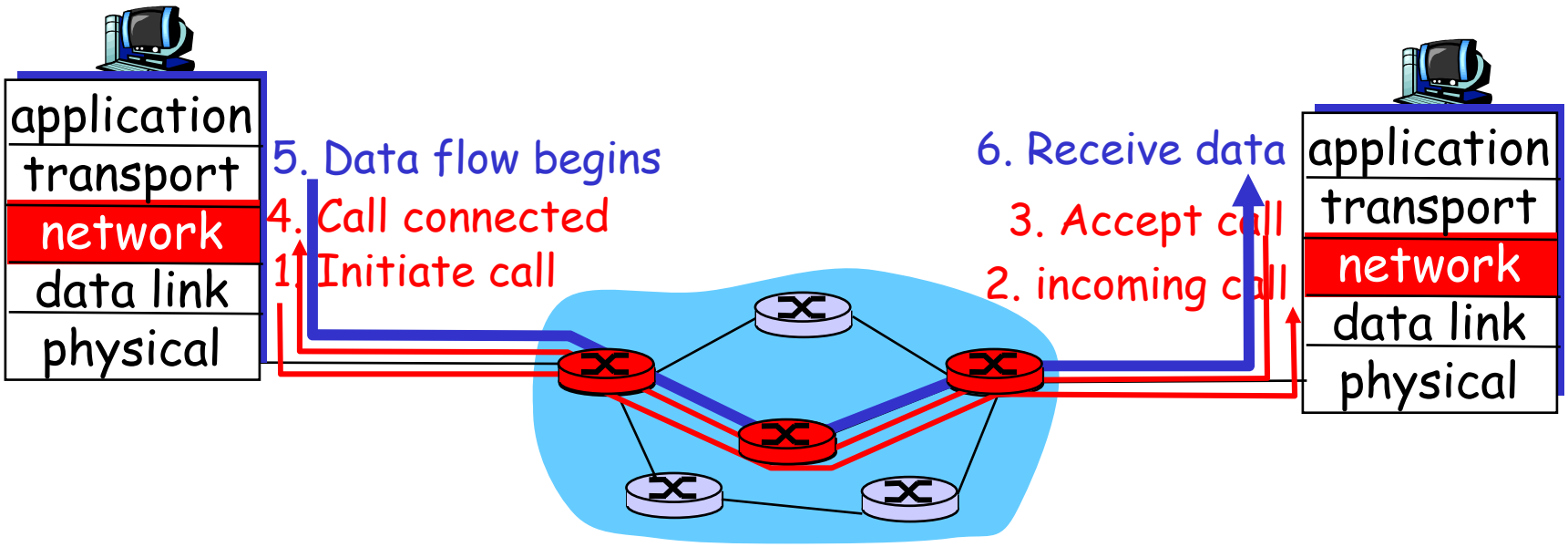


Forwarding table in northwest router:

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

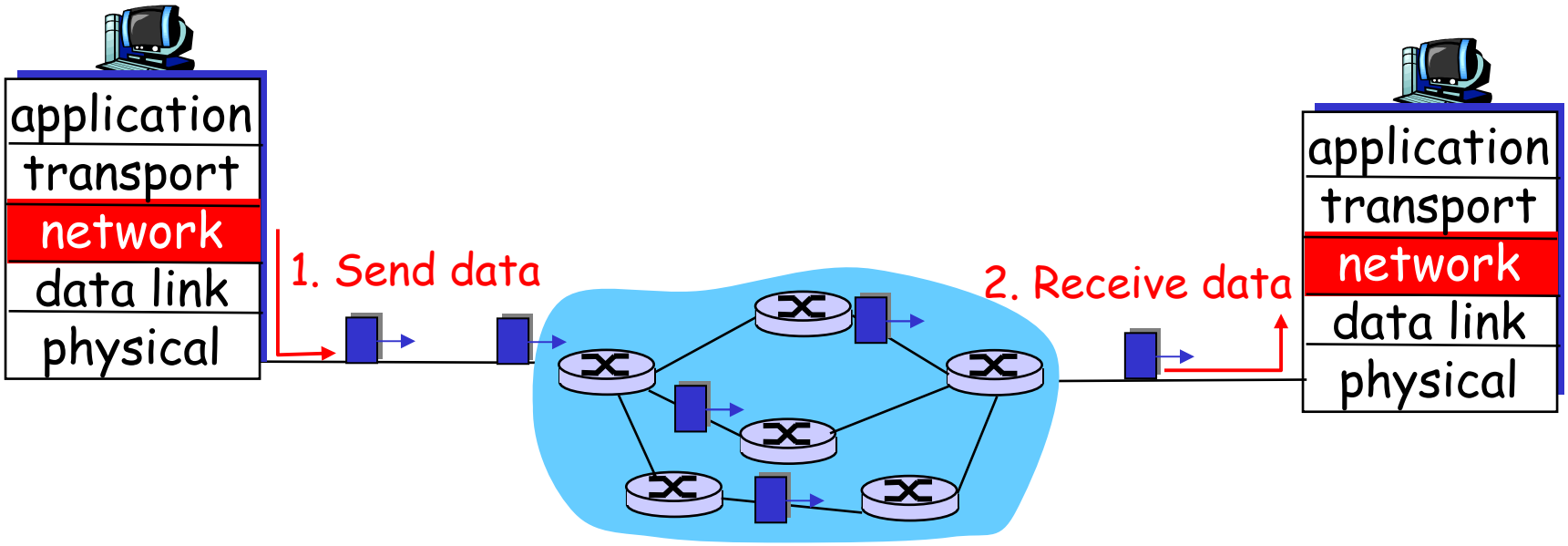
Routers maintain connection state information!

Virtual circuits: signaling protocols



Datagram networks

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



Forwarding table

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

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?

Datagram or VC network: why?

Internet (datagram)

- data exchange among computers
  - "elastic" service, no strict timing req.
- "smart" end systems (computers)
  - adaptation, control, error recovery
  - simplicity at the "core",
  - complexity at the "edge"
- many link types
  - different characteristics
  - uniform service difficult

ATM (VC)

- evolved from telephony
- strict timing, reliability requirements
- guaranteed service
- "dumb" end systems
  - telephones
  - complexity inside network

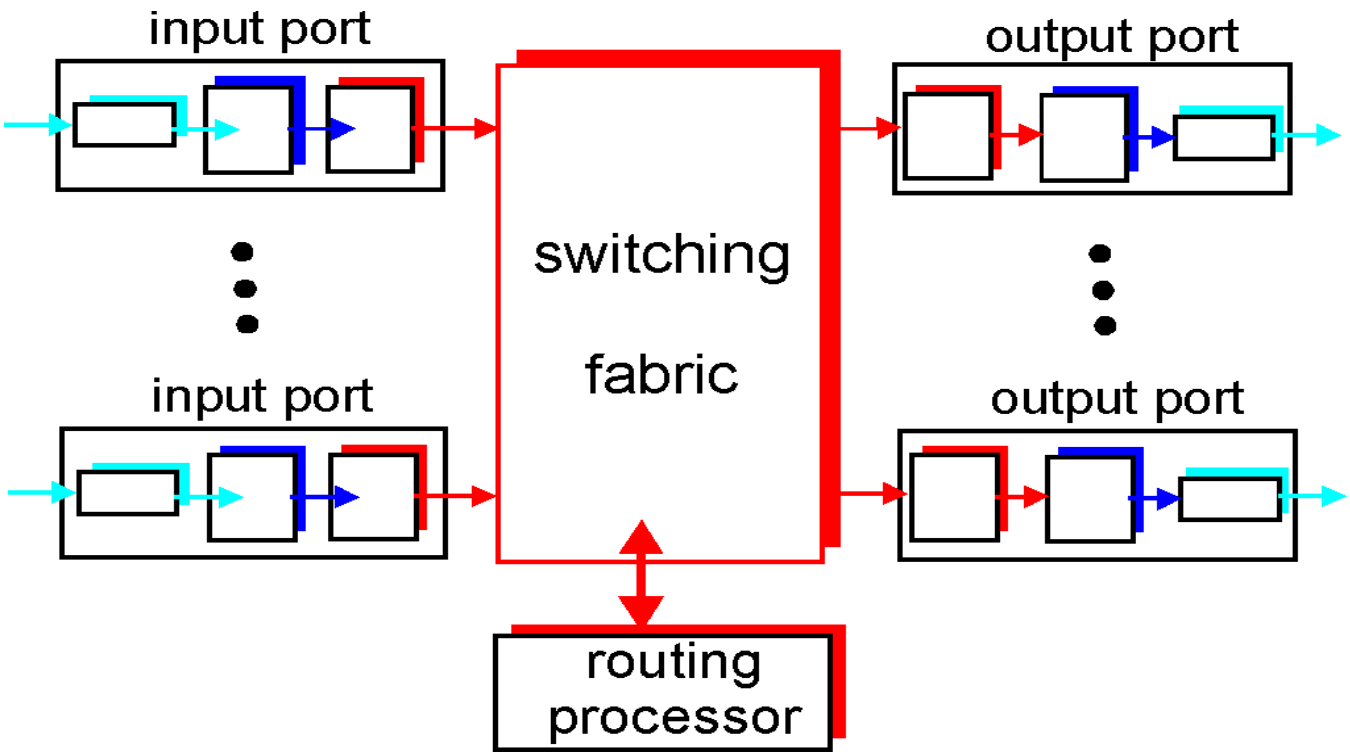
Outline

- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
- 4.5 Routing algorithms
- 4.6 Routing protocols in the Internet
- 4.7 Broadcast and multicast routing

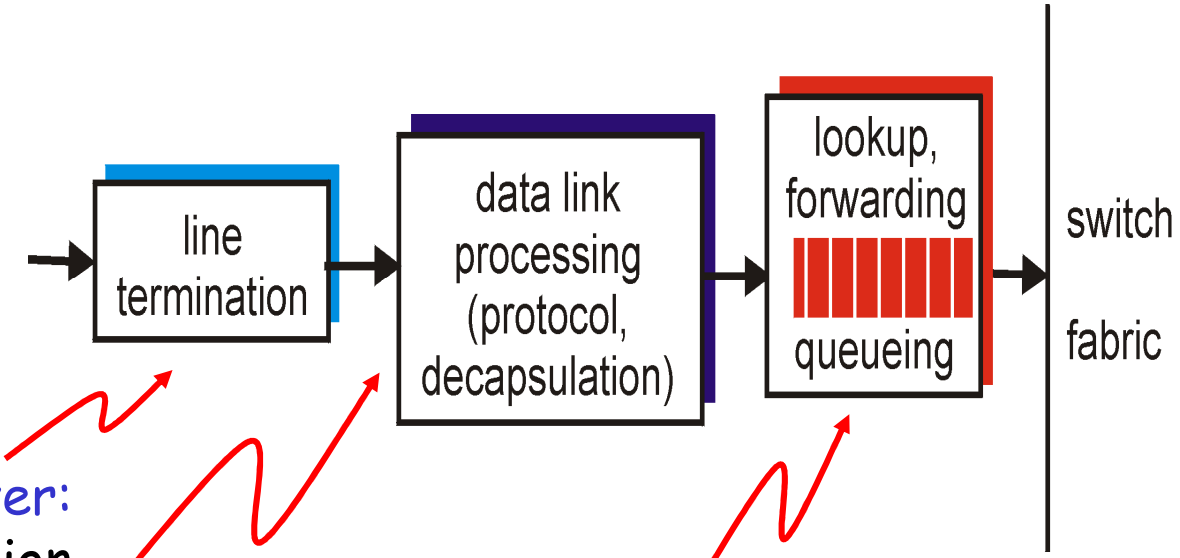
Router Architecture Overview

Two key router functions:

- running routing algorithms
- *forwarding* datagrams from incoming to outgoing link



Input Port Functions



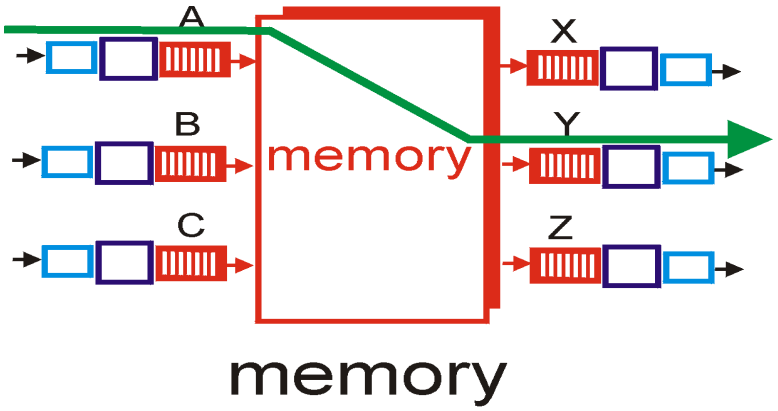
Physical layer:
bit-level reception

Data link layer:
e.g., Ethernet (chapter 5)

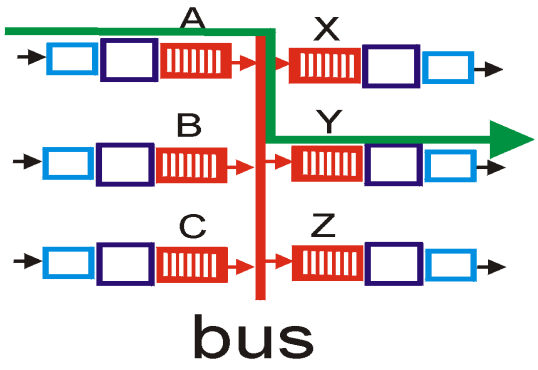
Decentralized switching:

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

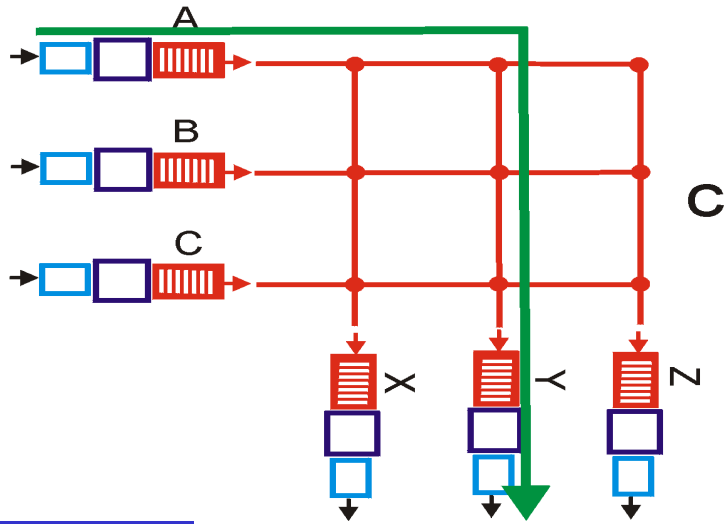
Three types of switching fabrics



memory



bus

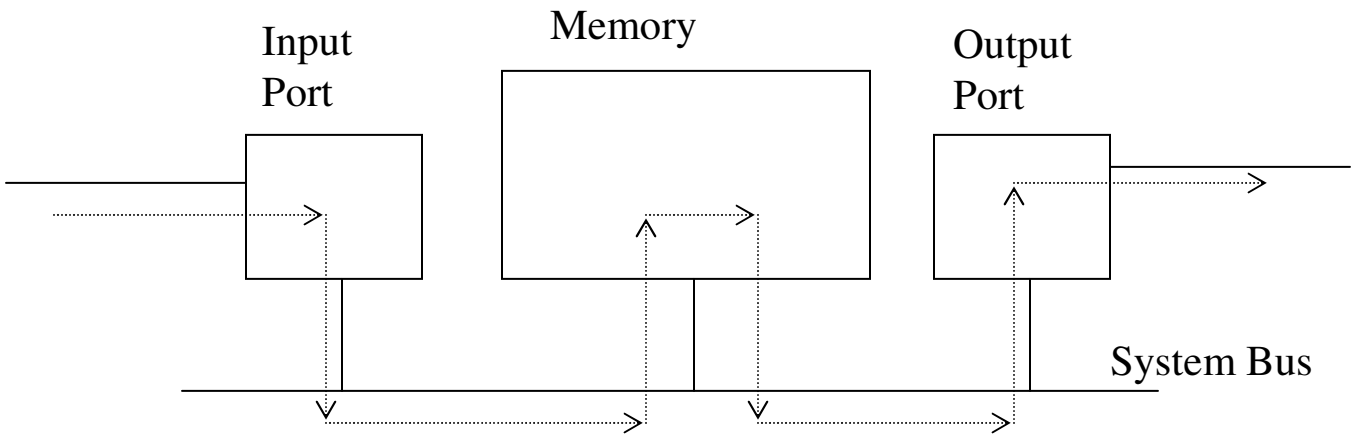


crossbar

Switching Via Memory

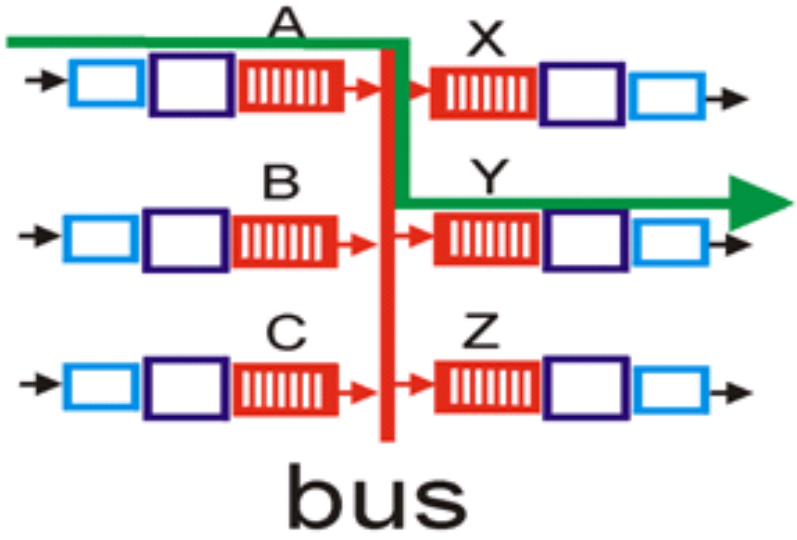
First generation routers:

- 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)



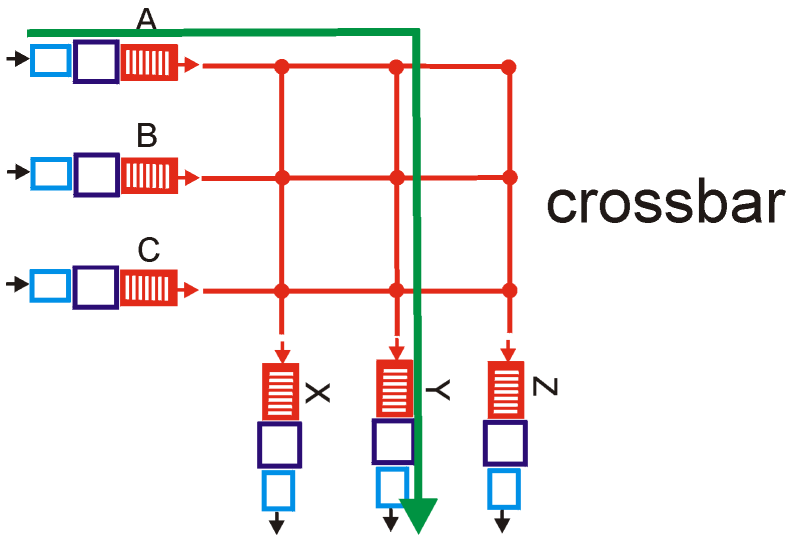
Switching Via a Bus

- datagram from input port memory to output port memory via a shared bus
- switching speed limited by bus bandwidth
- 32 Gbps bus, Cisco 5600: sufficient speed for access and enterprise routers

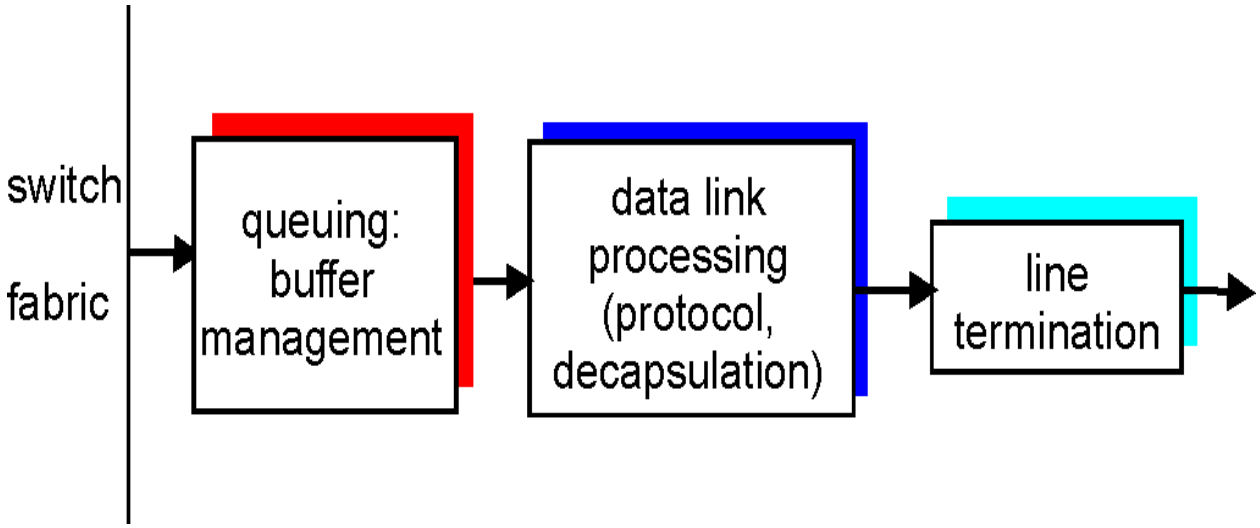


Switching Via An Interconnection Network

- overcome bus bandwidth limitations
- initially developed to connect processors in multiprocessor
- Cisco 12000: switches 60 Gbps through the interconnection network

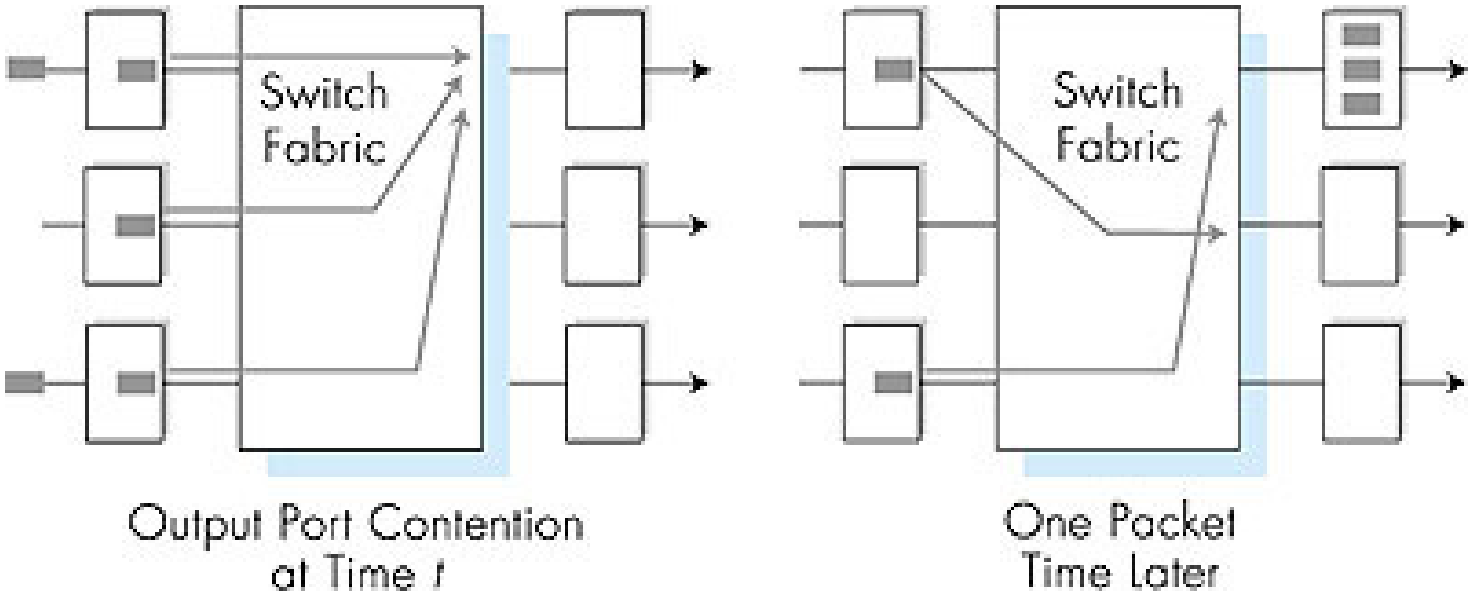


Output Ports



- *Buffering* required when datagrams arrive from fabric faster than the transmission rate
- *Scheduling discipline* chooses among queued datagrams for transmission

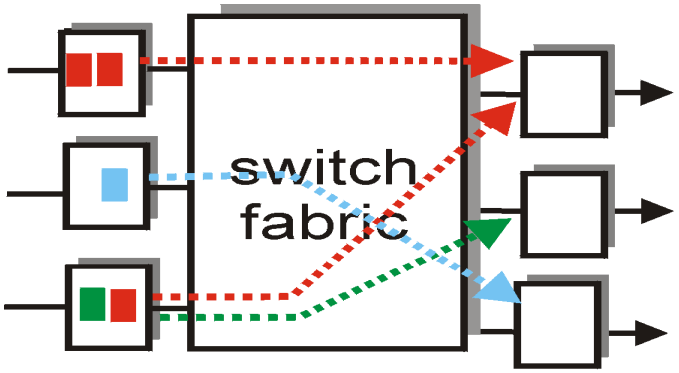
Output port queueing



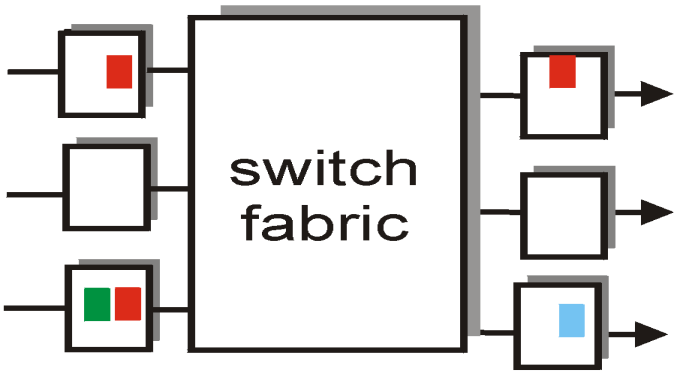
- buffering when arrival rate via switch exceeds output line speed
- *queueing (delay) and loss due to output port buffer overflow!*

Input Port Queuing

- Fabric slower than input ports combined -> queueing may occur at input queues
- Head-of-the-Line (HOL) 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



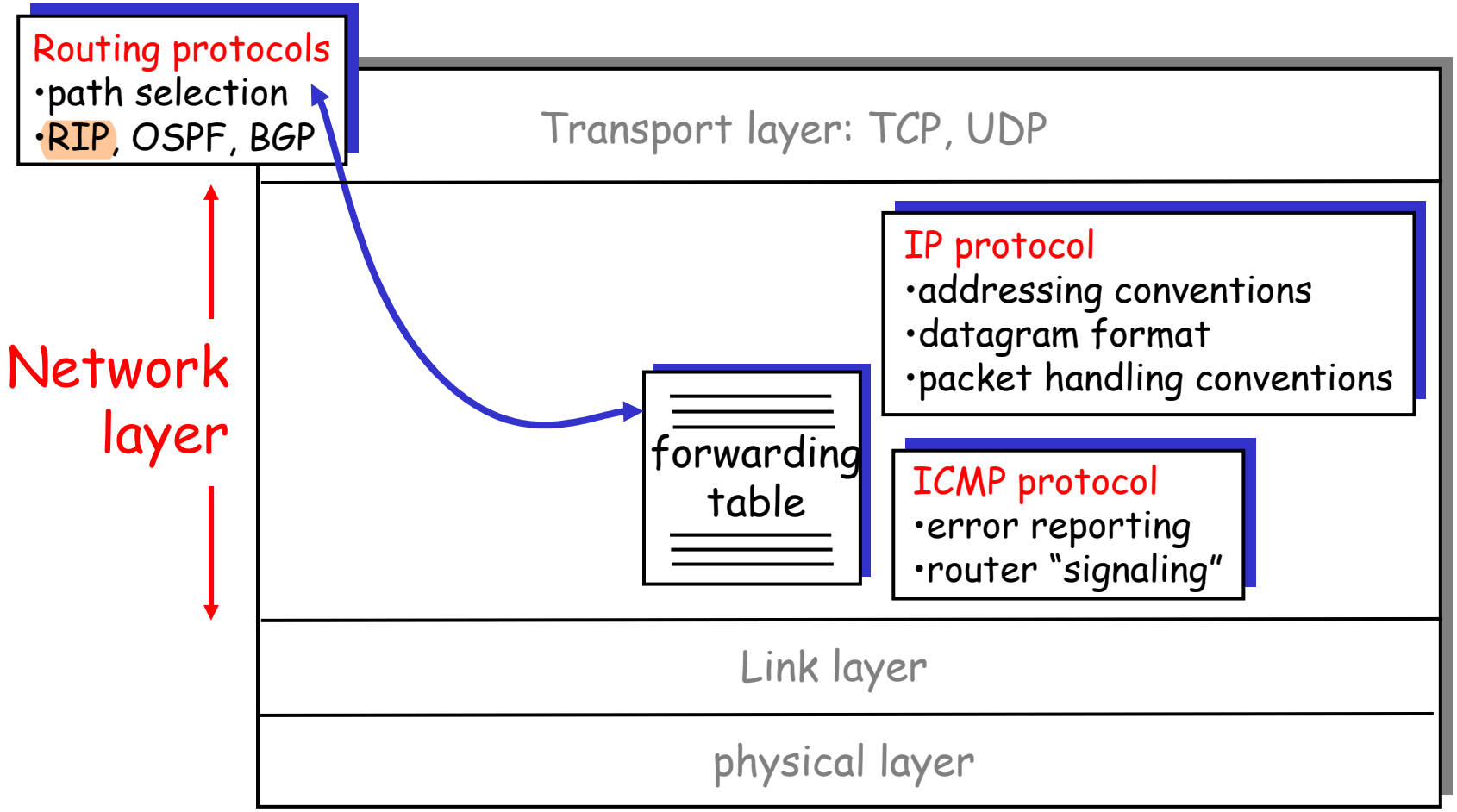
green packet experiences HOL blocking

Outline

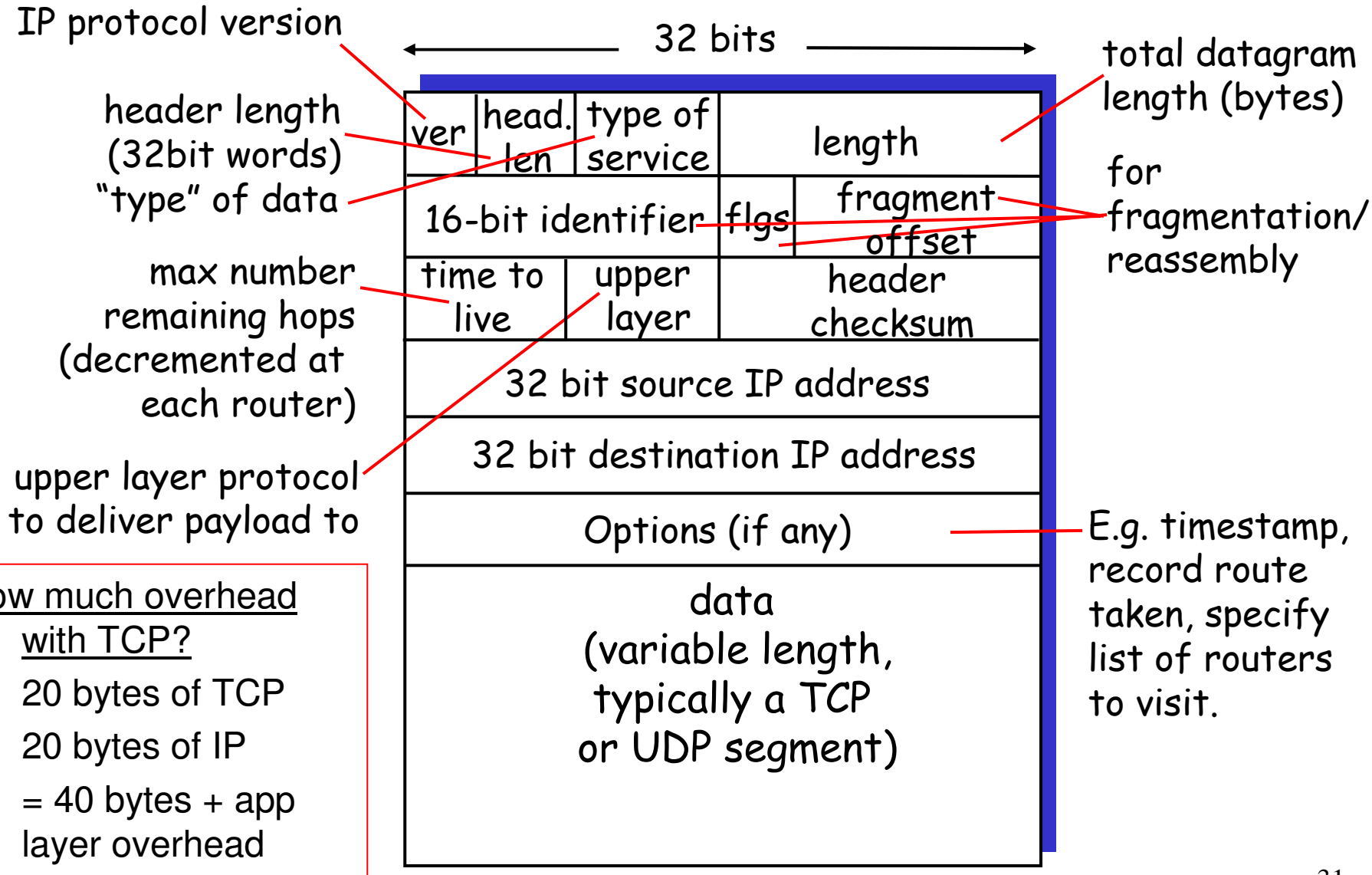
- 4.1 Introduction
- 4.2 Virtual circuit and datagram networks
- 4.3 What's inside a router
- 4.4 IP: Internet Protocol
- 4.5 Routing algorithms
- 4.6 Routing protocols in the Internet
- 4.7 Broadcast and multicast routing

The Internet Network layer

Host, router network layer functions:



IP datagram format

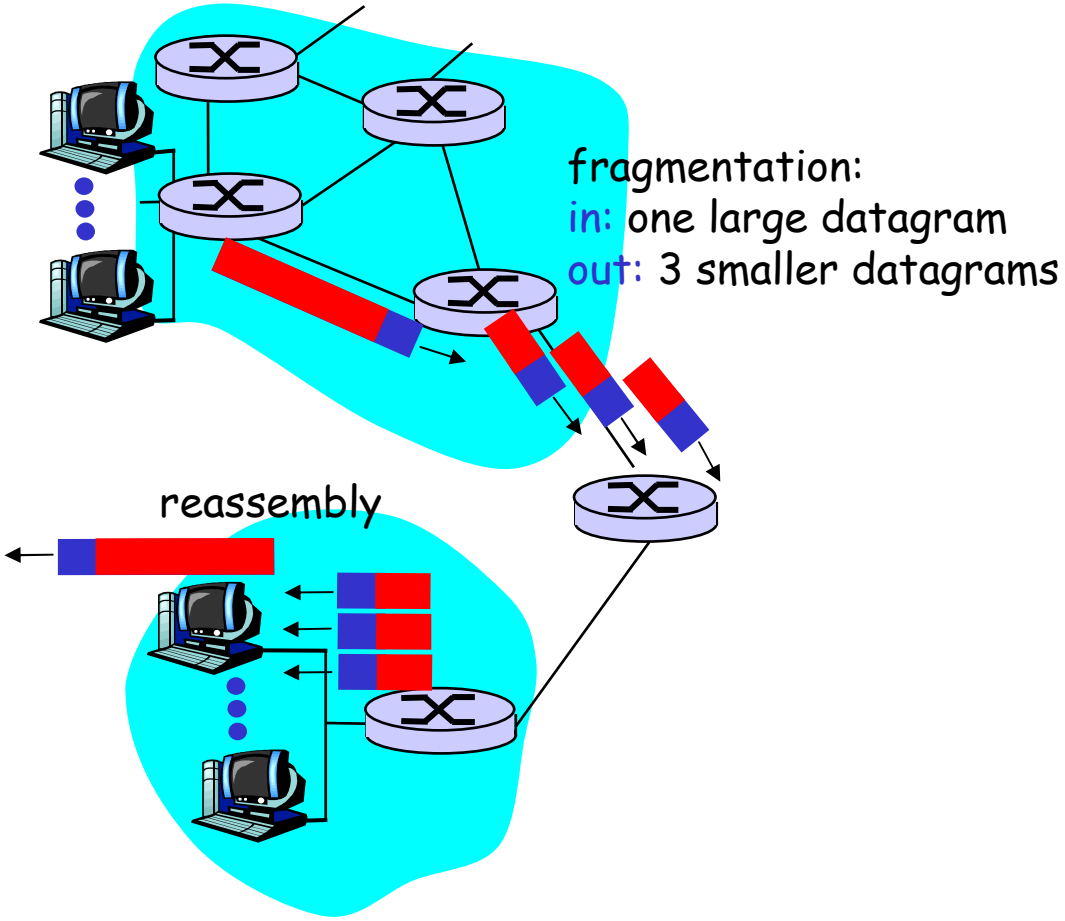


how much overhead with TCP?

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

IP Fragmentation & Reassembly

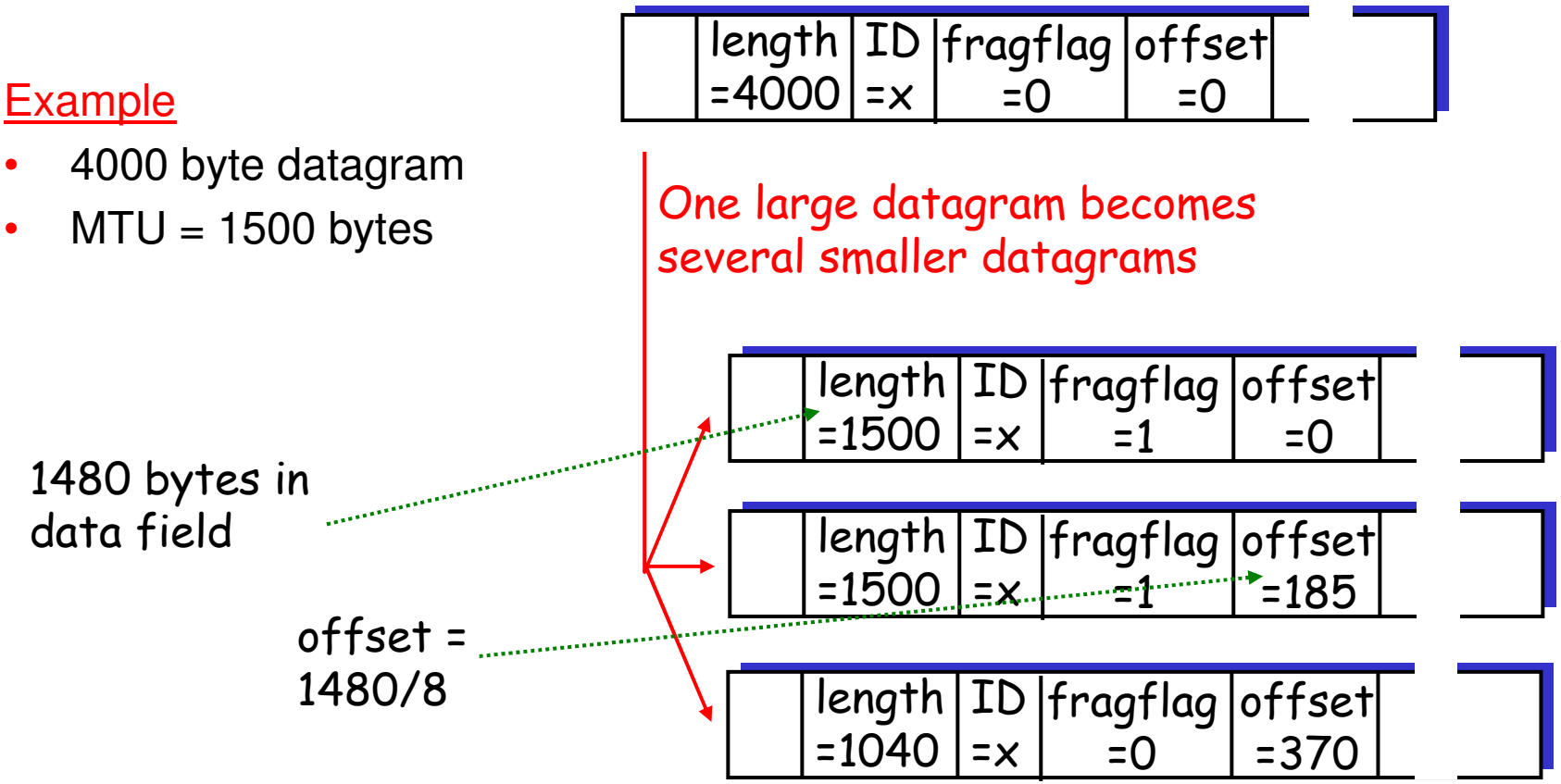
- network links have MTU (max.transfer unit) - largest possible link-level frame.
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order related fragments



IP Fragmentation and Reassembly

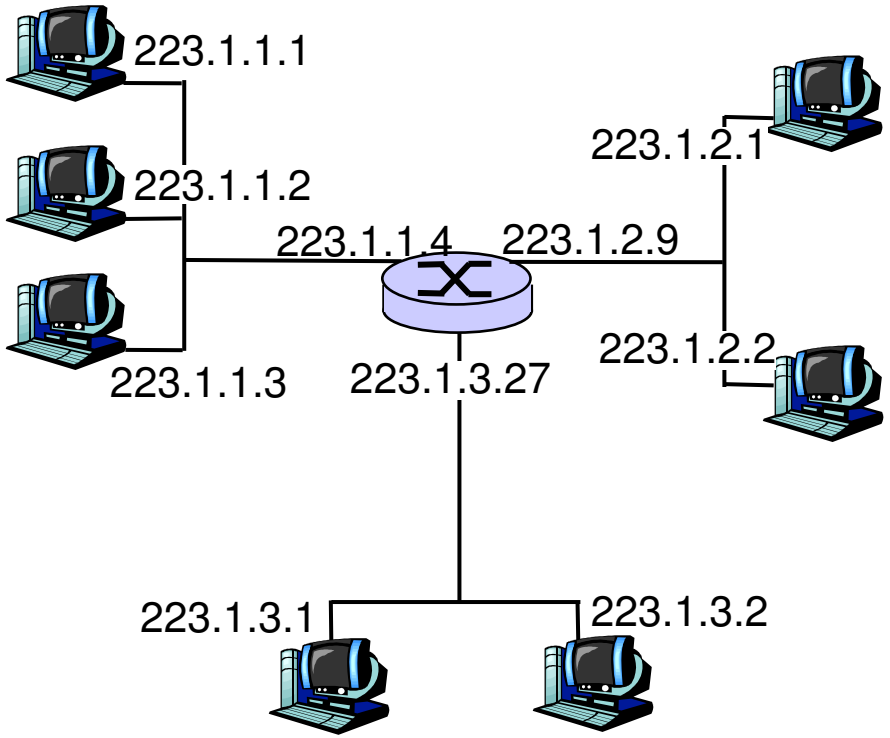
Example

- 4000 byte datagram
- MTU = 1500 bytes



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 interface
  - IP addresses associated with each interface

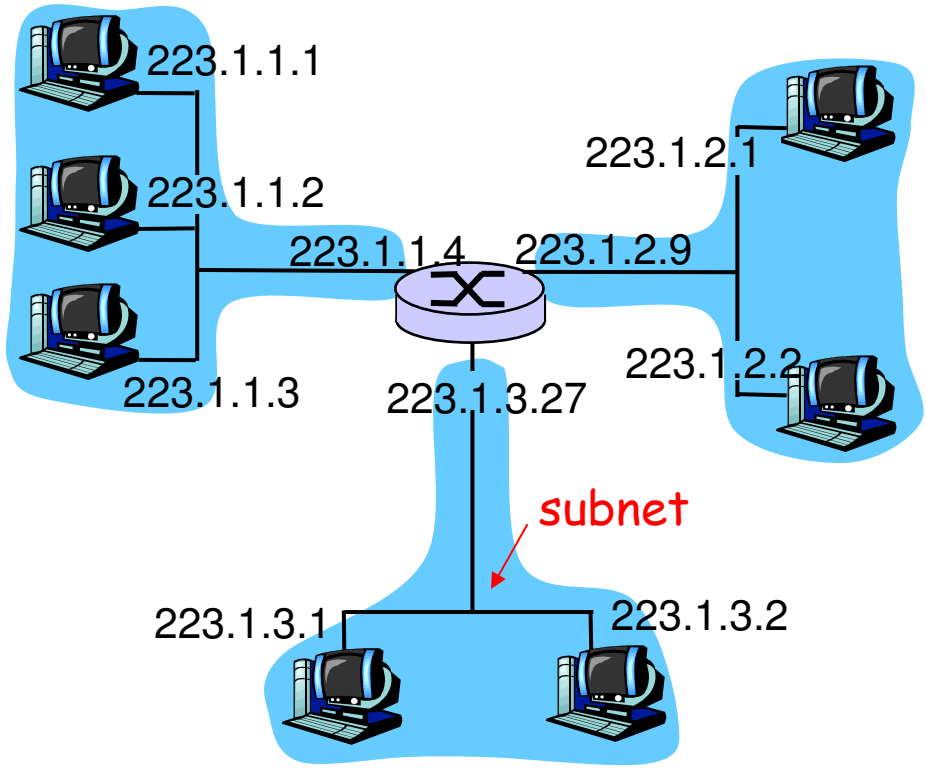


223.1.1.1 = 11011111 00000001 00000001 00000001

223 1 1 1

Subnets

- IP address:
  - subnet part (high order bits)
  - host part (low order bits)
- What is a subnet?
  - device interfaces with same subnet part of IP address
  - can physically reach each other without intervening router

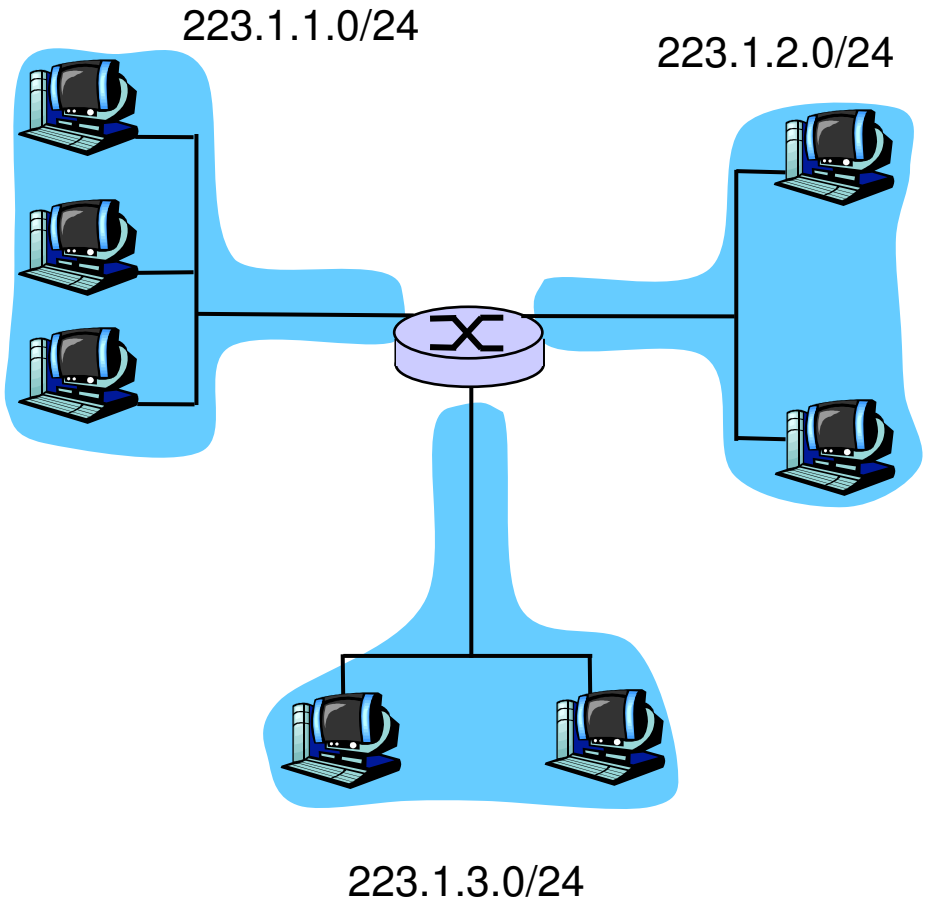


network consisting of 3 subnets

Subnets

Recipe

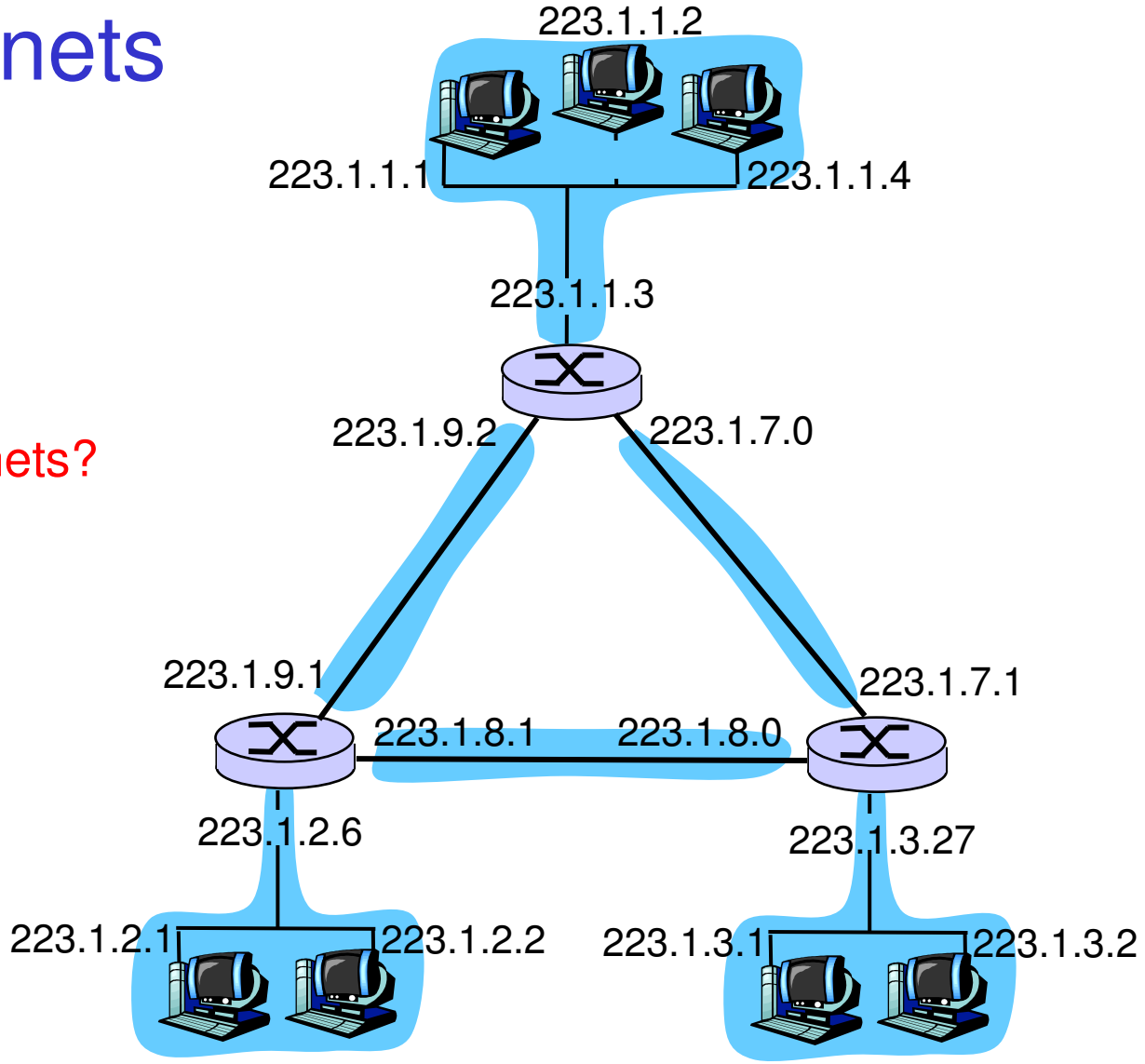
- To determine the subnets, detach each interface from its router, creating islands of isolated networks. Each isolated network is called a subnet.



Subnet mask: /24

Subnets

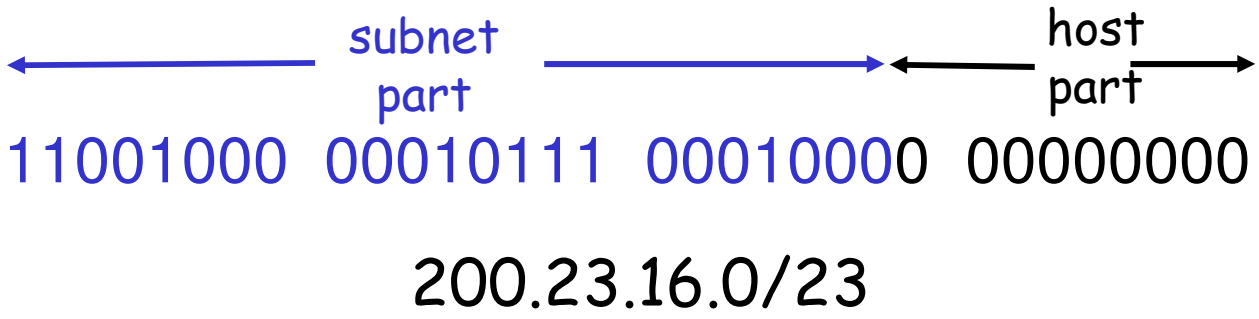
How many subnets?



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



Classful Addressing (less efficient)

- Class A: a.b.c.d/8
- Class B: a.b.c.d/16
- Class C: a.b.c.d/24

IP addresses: how to get one?

Q: How does a *host* get IP address?

- hard-coded by system admin in a file
  - Windows: control-panel->network->configuration->tcp/ip->properties
  - UNIX: /etc/rc.config
- DHCP: Dynamic Host Configuration Protocol:
  - dynamically get address from this server (IP pool)
  - “plug-and-play”

DHCP: Dynamic Host Configuration Protocol

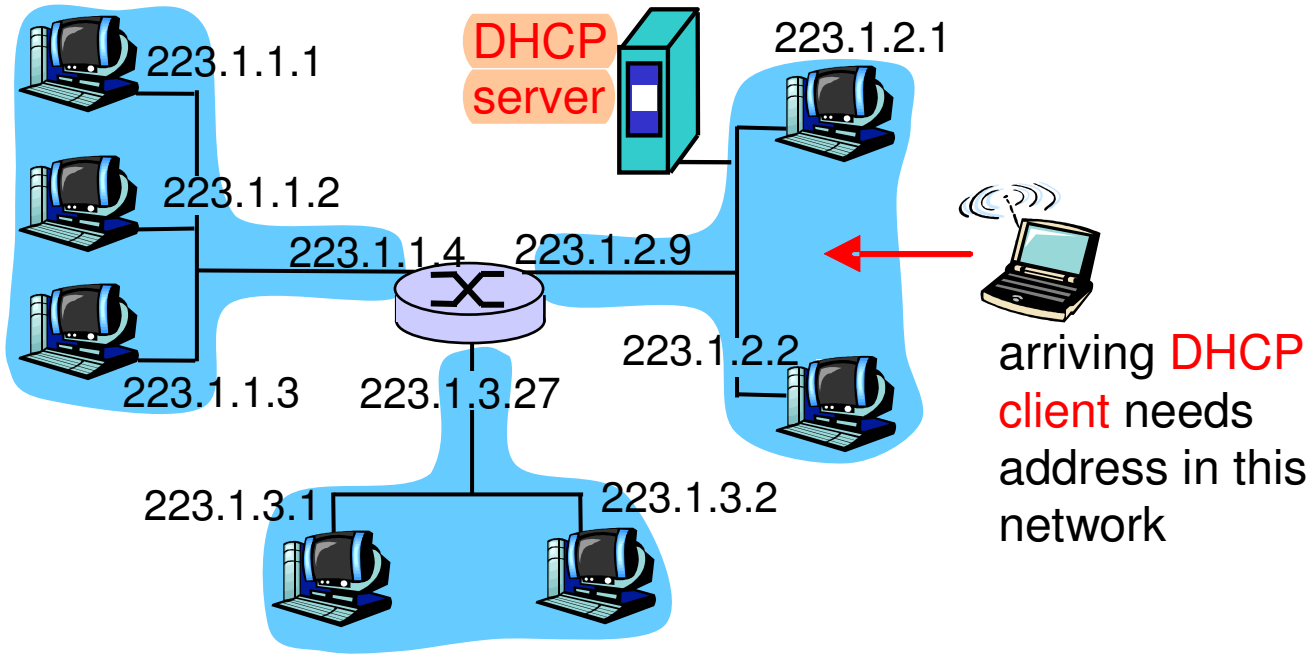
Goal: allow host to *dynamically* obtain its IP address from network server when it joins network (subnet)

- Can renew its lease on address in use
- Allows reuse of addresses (only hold address while connected and "on")
- Additional info:
  - subnet mask, first-hop router address (default gateway), local DNS server address…

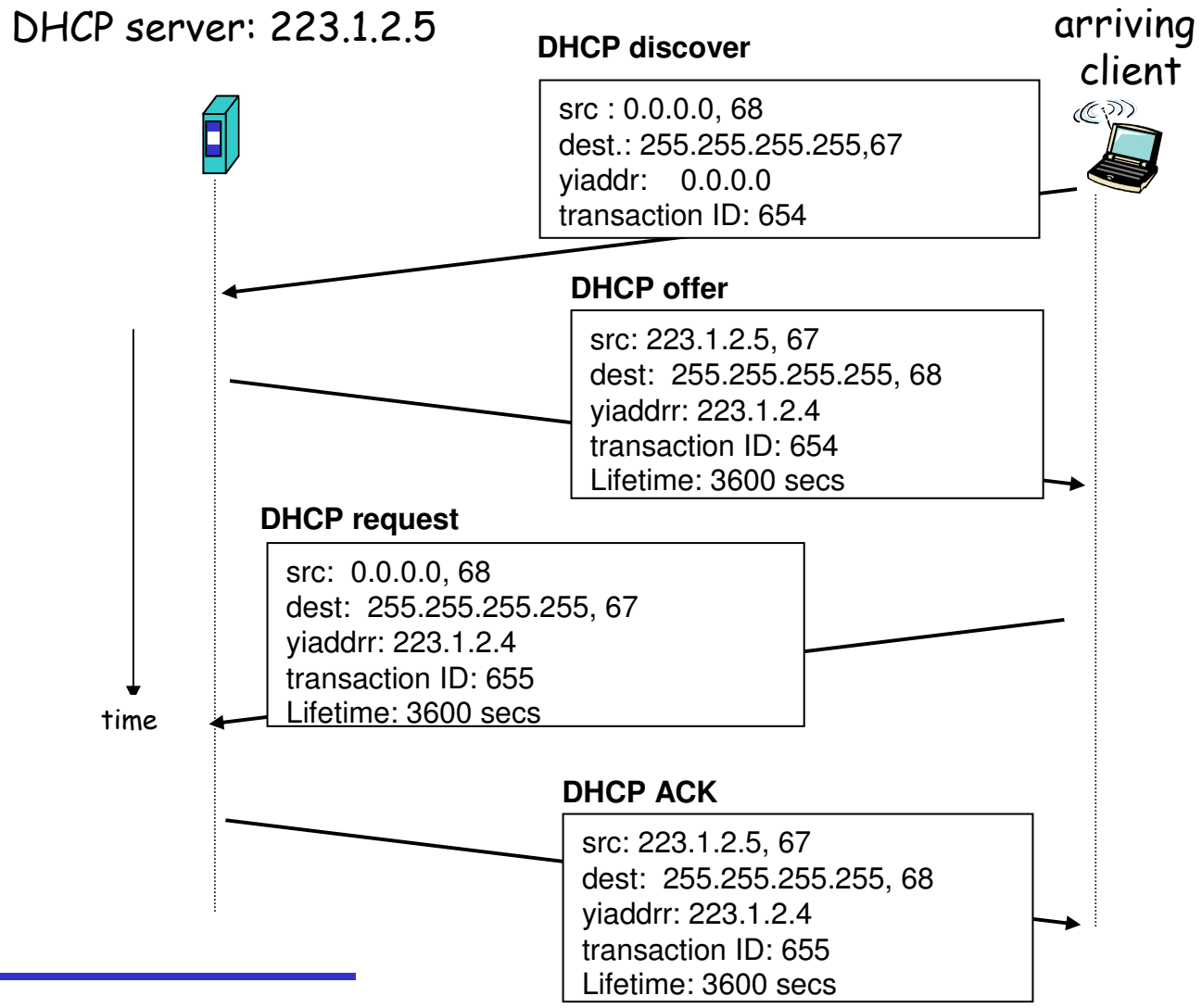
DHCP overview:

- 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

DHCP client-server scenario



DHCP client-server scenario



IP addresses: how to get one?

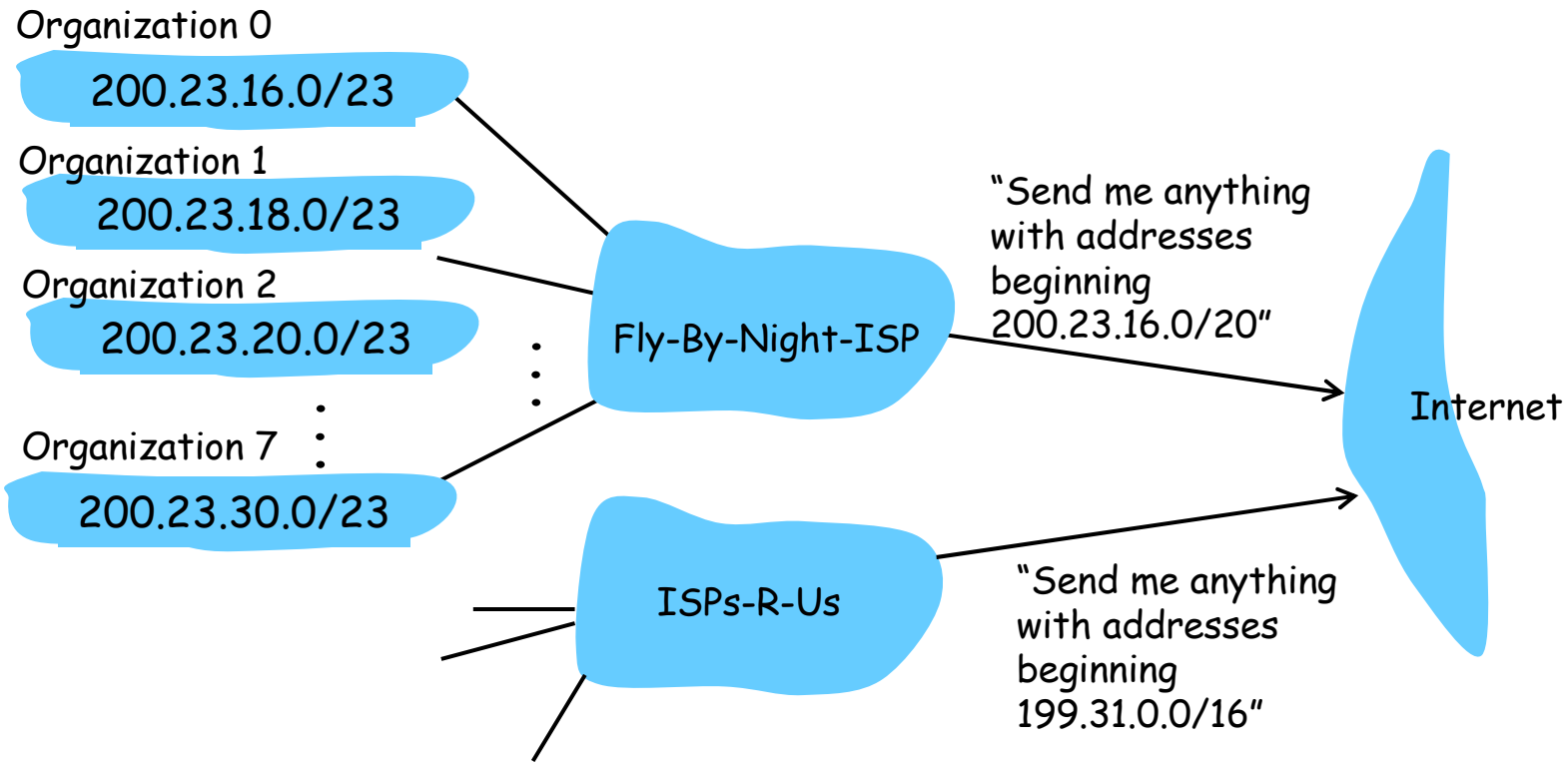
Q: How does *network* get subnet part of IP addr?

A: gets allocated portion of its provider 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

Hierarchical addressing

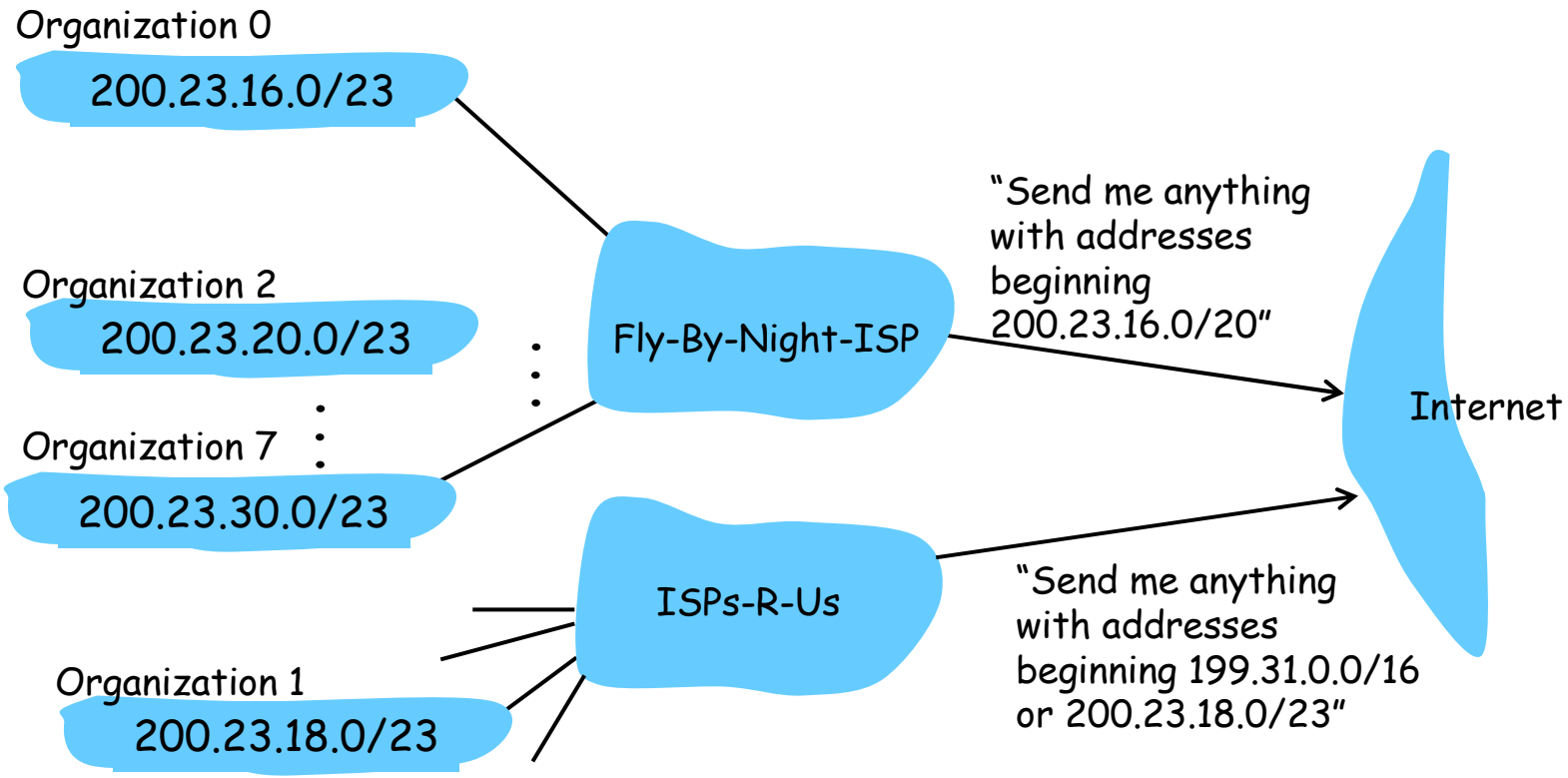
Hierarchical addressing allows efficient advertisement of routing information:



Tanır Özçelebi, t.ozcelebi@tue.nl
TU/e Computer Science, System Architecture and Networking

Hierarchical addressing: more specific routes

ISPs-R-Us has a more specific route to Organization 1



IP addressing: the last word...

Q:

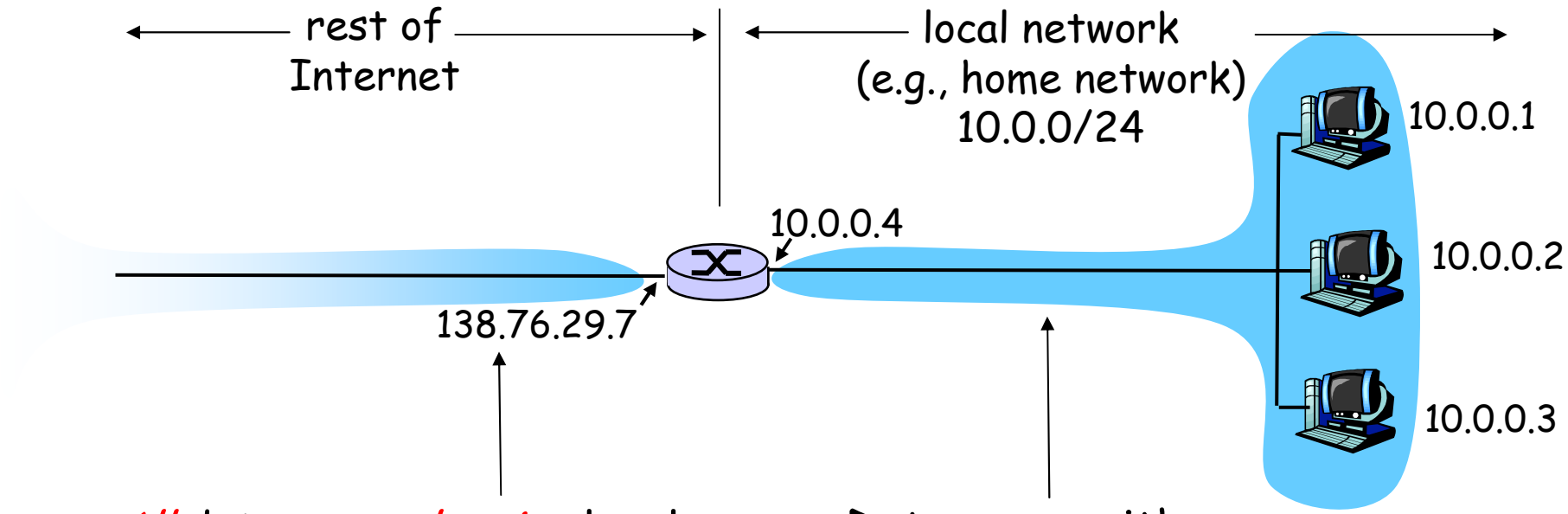
How does an ISP get block of addresses?

A:

ICANN: Internet Corporation for Assigned Names and Numbers

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

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

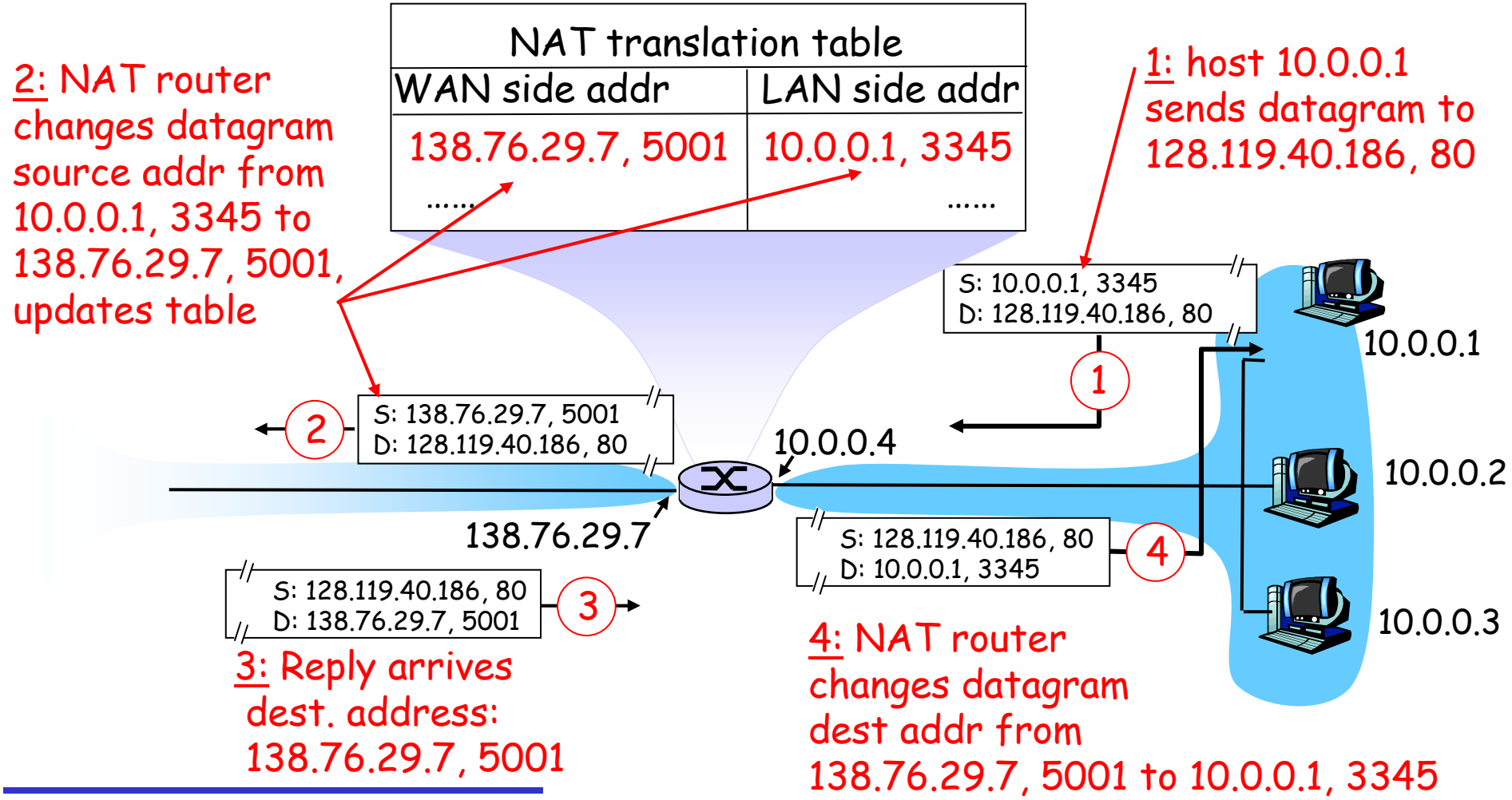
- Motivation: local network uses just one IP address as far as outside world is concerned:
  - range of addresses not needed from ISP:
    - just one IP address for all devices
    - 16-bit port-number field: 60,000 simultaneous connections with a single LAN-side address!
  - can change addresses of 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

- *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

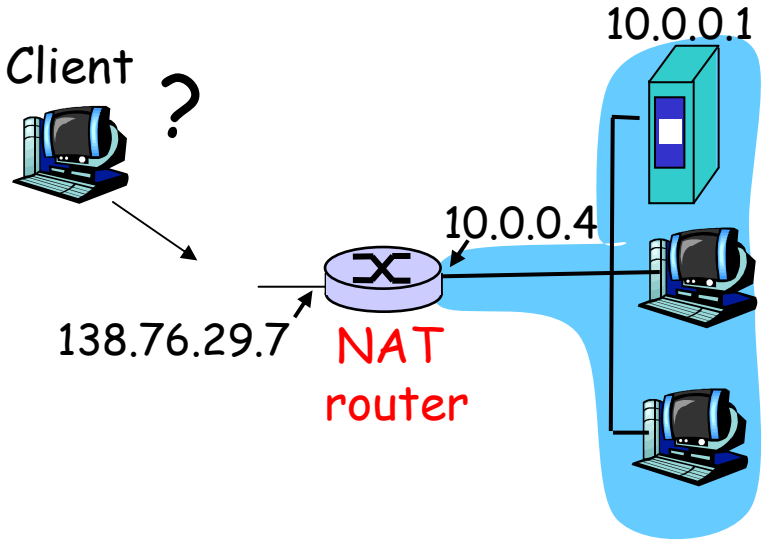


NAT: Network Address Translation

- 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
  - address shortage should instead be solved by IPv6

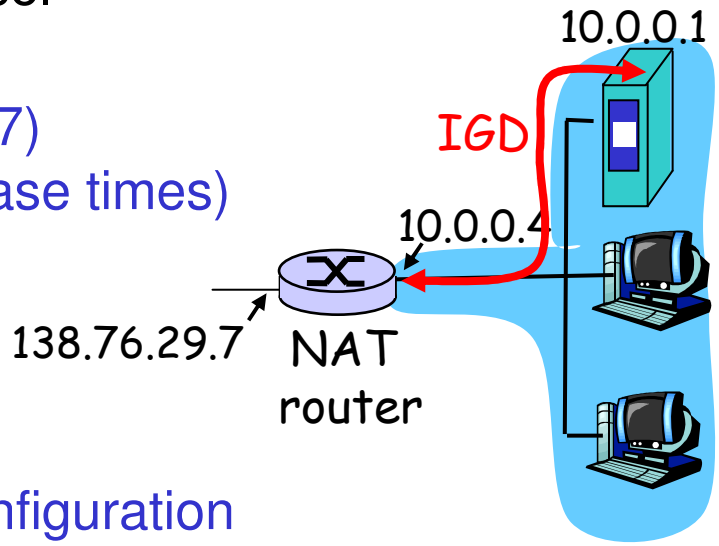
NAT traversal problem

- client wants 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 NATed address: 138.76.29.7
- solution 1: 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 25000



NAT traversal problem

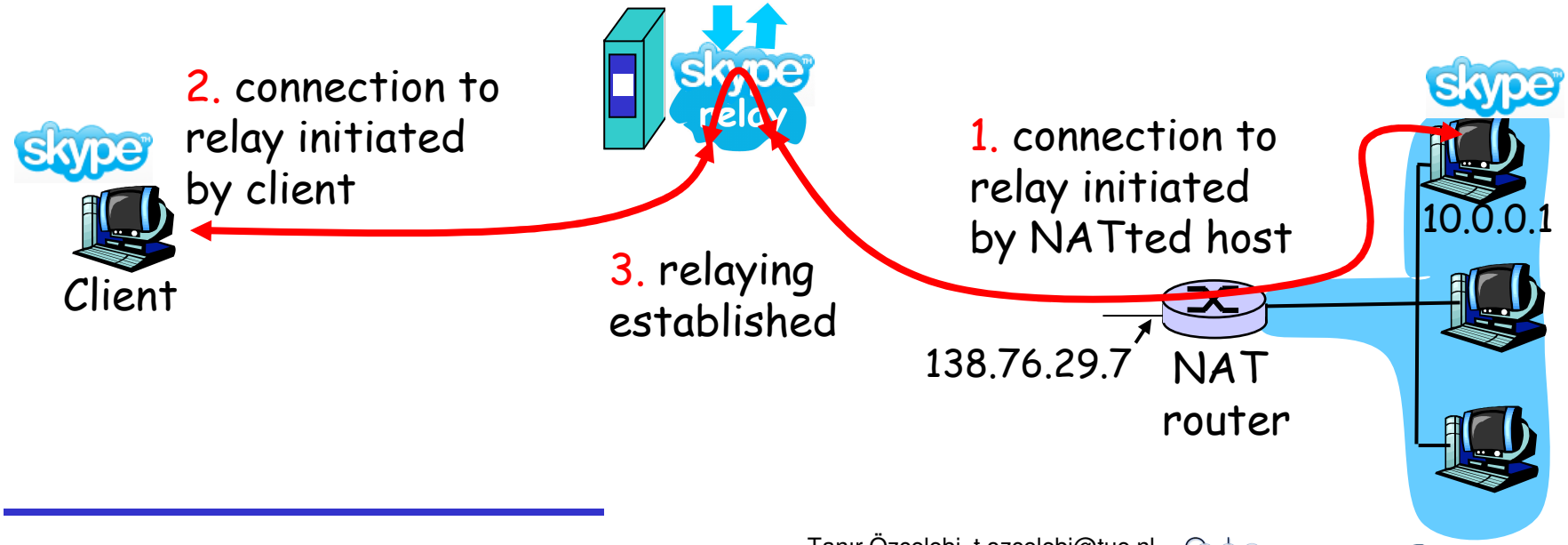
- solution 2:
  Universal Plug and Play (UPnP) Protocol
  allows NATted host to:
  - ❖ learn public IP address (138.76.29.7)
  - ❖ add/remove port mappings (with lease times)



i.e., automate static NAT port map configuration

NAT traversal problem

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



ICMP: Internet Control Message Protocol

- used by hosts & routers to communicate network-level information
  - error reporting: unreachable host, network, port, protocol
  - echo request/reply (used by ping)
- network-layer "above" IP:
  - ICMP msgs carried in IP datagrams
- ICMP message: type, code plus first 8 bytes of IP datagram causing error

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

Traceroute and ICMP

- Source sends series of UDP segments to dest
  - first has TTL =1
  - second has TTL=2, etc.
  - unlikely port number
- When n^{th} datagram arrives to n^{th} router:
  - router discards datagram
  - sends to source an ICMP message (type 11, code 0)
  - message includes name of router & IP address
- When ICMP message arrives, source calculates RTT
- Traceroute does this 3 times

Stopping criterion

- UDP segment eventually arrives at destination host
- Destination returns an ICMP "port unreachable" packet (type 3, code 3)
- When source gets this ICMP, it stops.

IPv6

- Initial motivation: 32-bit address space soon to be completely allocated.

- Additional motivation:
  - header format helps speed up processing/forwarding
  - header changes to facilitate QoS

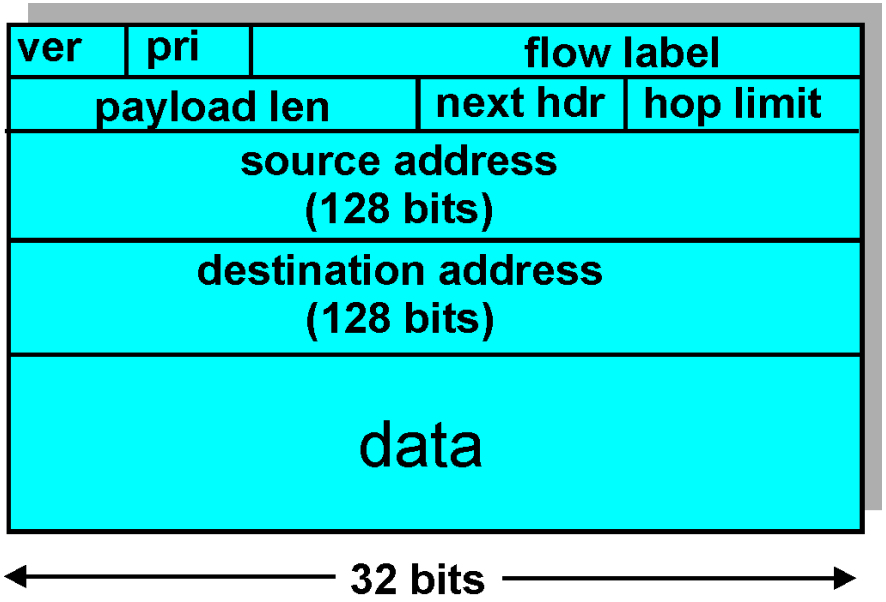
  IPv6 datagram format:
  - fixed-length 40 byte header
  - no fragmentation allowed

IPv6 Header (Cont)

Priority: identify priority among datagrams in flow

Flow Label: identify datagrams in same "flow."

Next header: identify upper layer protocol for data



Other Changes from IPv4

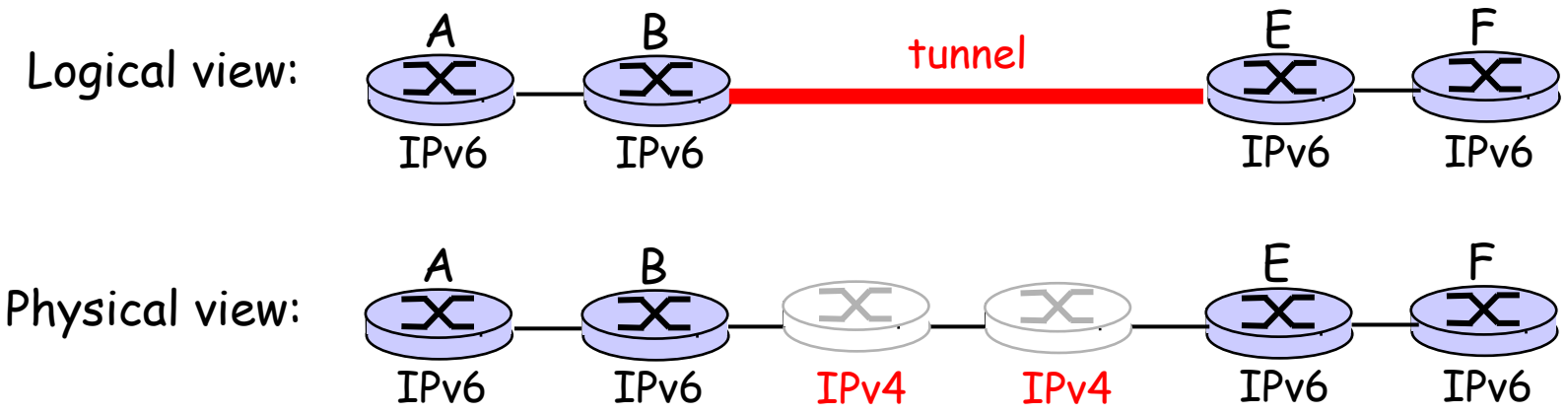
- *Checksum*: removed entirely to reduce processing time at each hop
- No fragmentations!
  - Packet size negotiated initially.
- *Options:* allowed, but outside of header, indicated by “Next Header” field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. “Packet Too Big”
  - multicast group management functions

Transition From IPv4 To IPv6

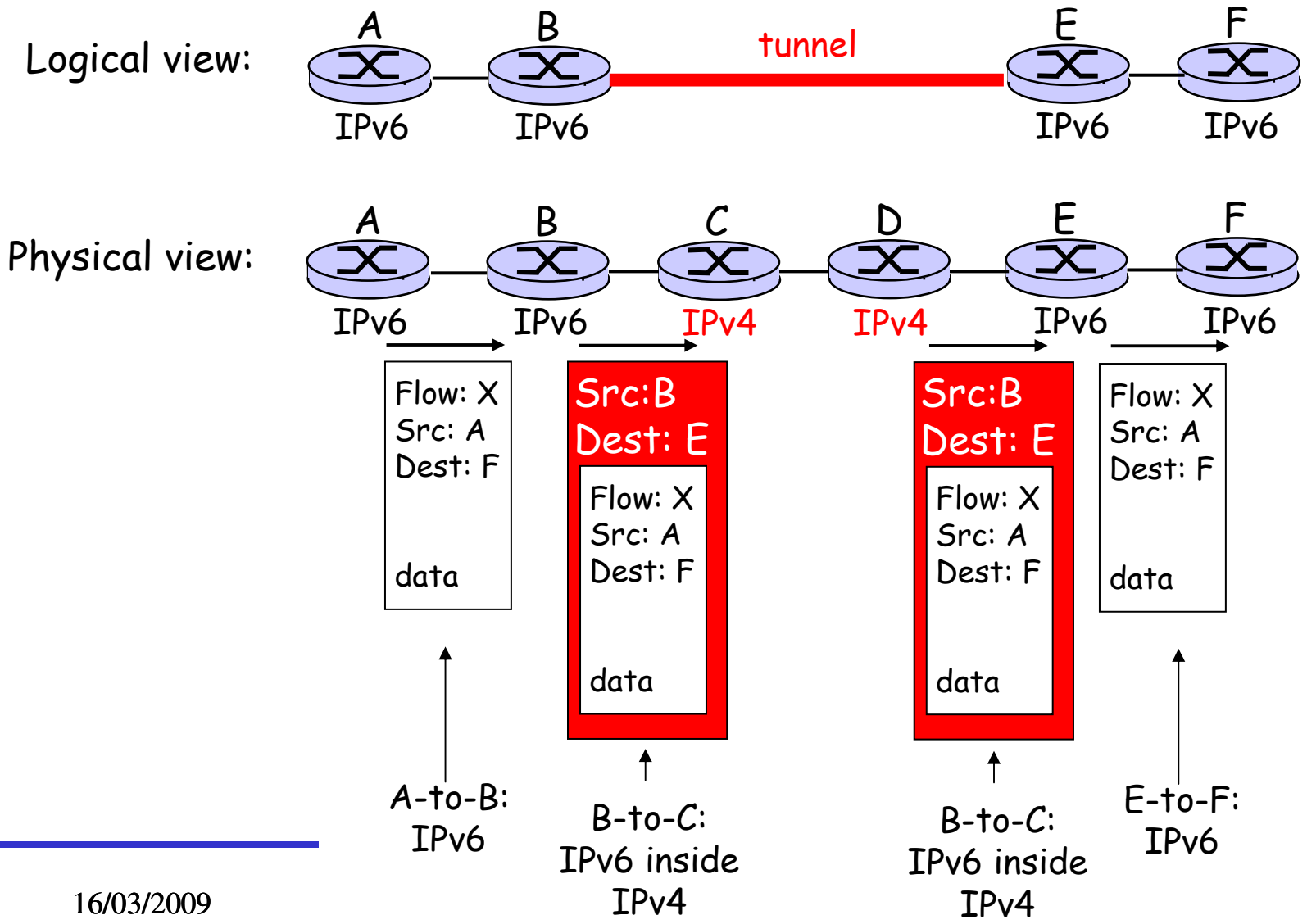
- Not all routers can be upgraded simultaneously: no “flag days”
  - How will the network operate with mixed IPv4 and IPv6 routers?

- *Tunneling:* IPv6 carried as payload in IPv4 datagram among IPv4 routers

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
B-to-C:
IPv6 inside
IPv4
E-to-F:
IPv6