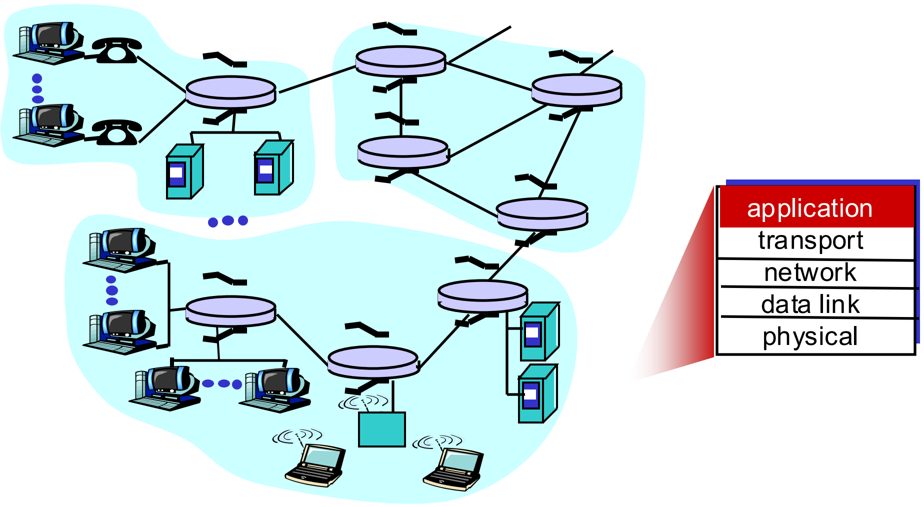
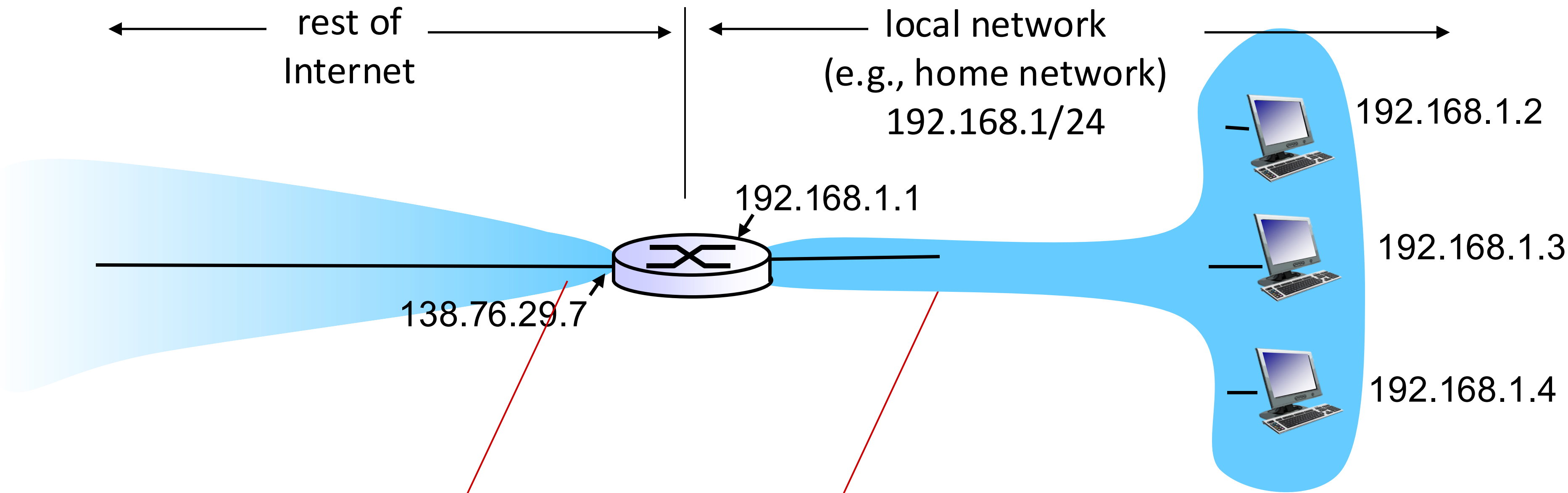
CS 4390
Computer Networks



Network Layer
The Internet Protocol – Part II

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 192.168.1/24 address for source, destination (as usual)

NAT – 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 – alleviate IP addresses shortage!
- can change addresses of devices in local network without notifying outside world – ease of maintenance
- 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 – Implementation

NAT router must:

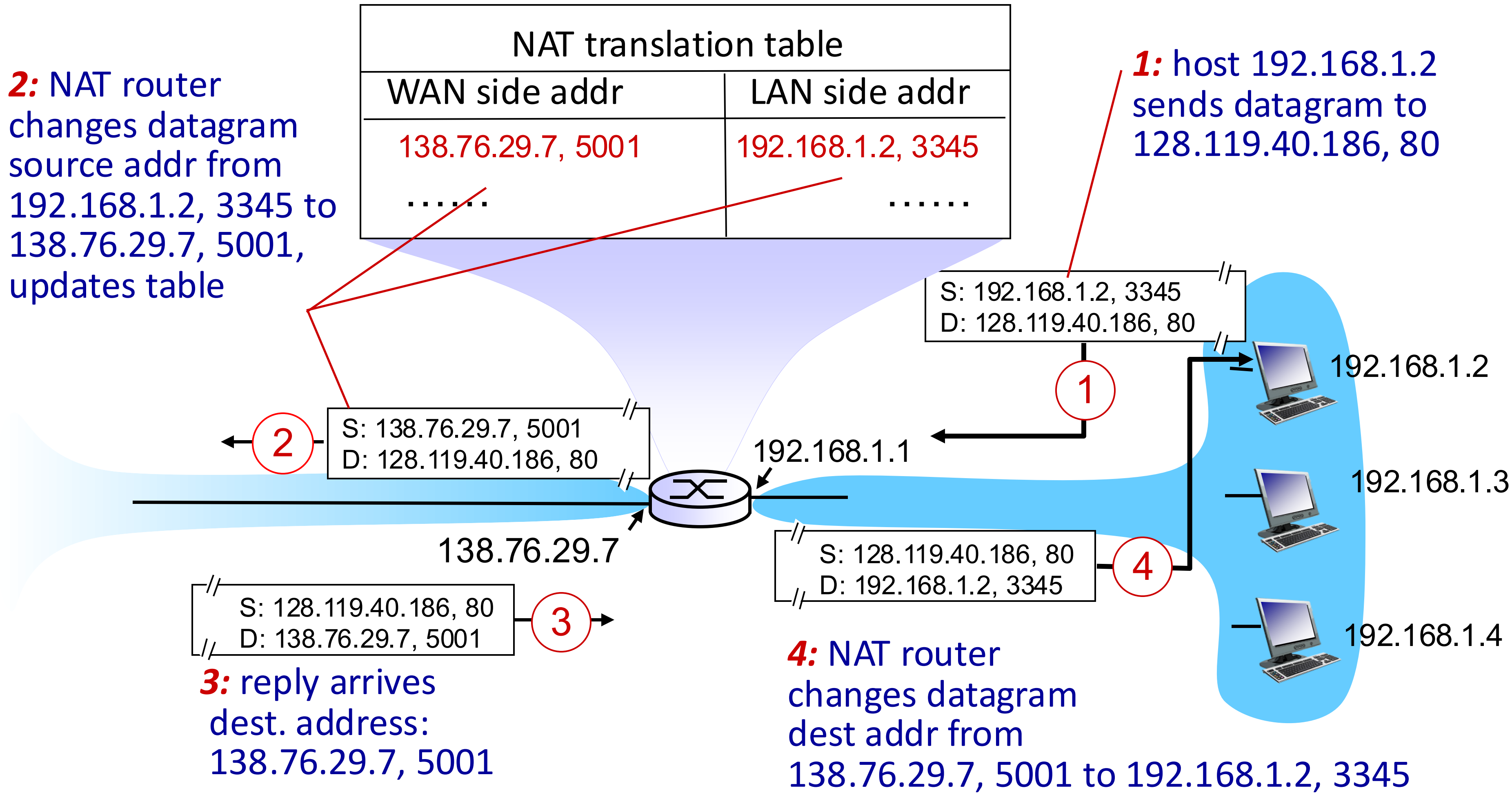
– *for outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)

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

– *for 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

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

NAT - Example

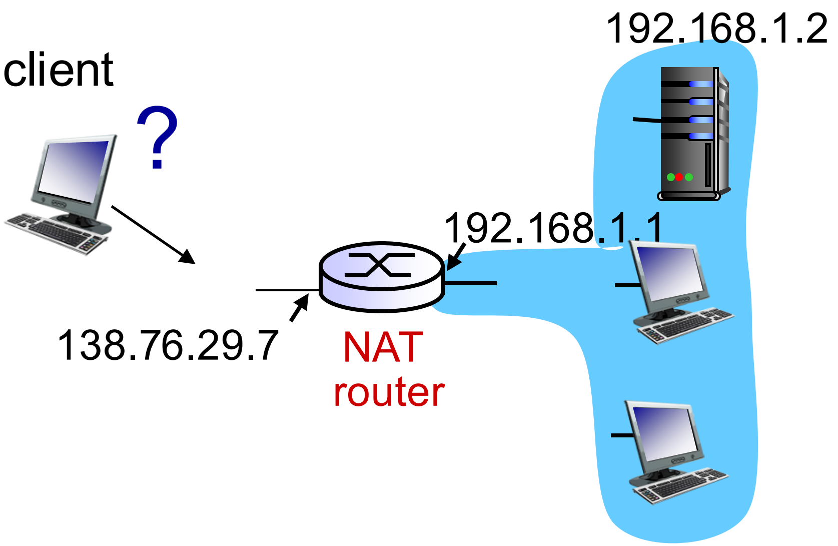

NAT translation table
WAN side addr
LAN side addr
138.76.29.7, 5001
192.168.1.2, 3345
……
……
2: NAT router changes datagram source addr from 192.168.1.2, 3345 to 138.76.29.7, 5001, updates table
1: host 192.168.1.2 sends datagram to 128.119.40.186, 80
S: 192.168.1.2, 3345
D: 128.119.40.186, 80
1
192.168.1.2
192.168.1.3
192.168.1.4
192.168.1.1
S: 138.76.29.7, 5001
D: 128.119.40.186, 80
2
138.76.29.7
S: 128.119.40.186, 80
D: 138.76.29.7, 5001
3
3: reply arrives dest. address: 138.76.29.7, 5001
S: 128.119.40.186, 80
D: 192.168.1.2, 3345
4
4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 192.168.1.2, 3345

NAT

- 16-bit port-number field:
  - ~60K simultaneous mappings with a single LAN-side address, after excluding the well known port numbers etc…!
- NAT is *controversial*:
  - routers should only process up to layer 3
  - address shortage should instead be solved by IPv6

NAT Traversal Problem

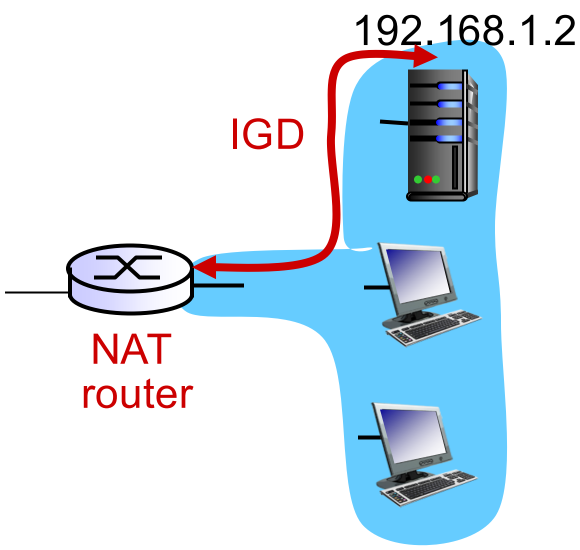
- client wants to connect to server with address 192.168.1.2
  - server address 192.168.1.2 local to LAN (client can't use it as destination addr)
  - only one externally visible NATed address: 138.76.29.7
- *solution1:* statically configure NAT to forward incoming connection requests at given port to server
  - e.g., (123.76.29.7, port 2500) always forwarded to 192.168.1.2 port 25000



NAT Traversal Problem

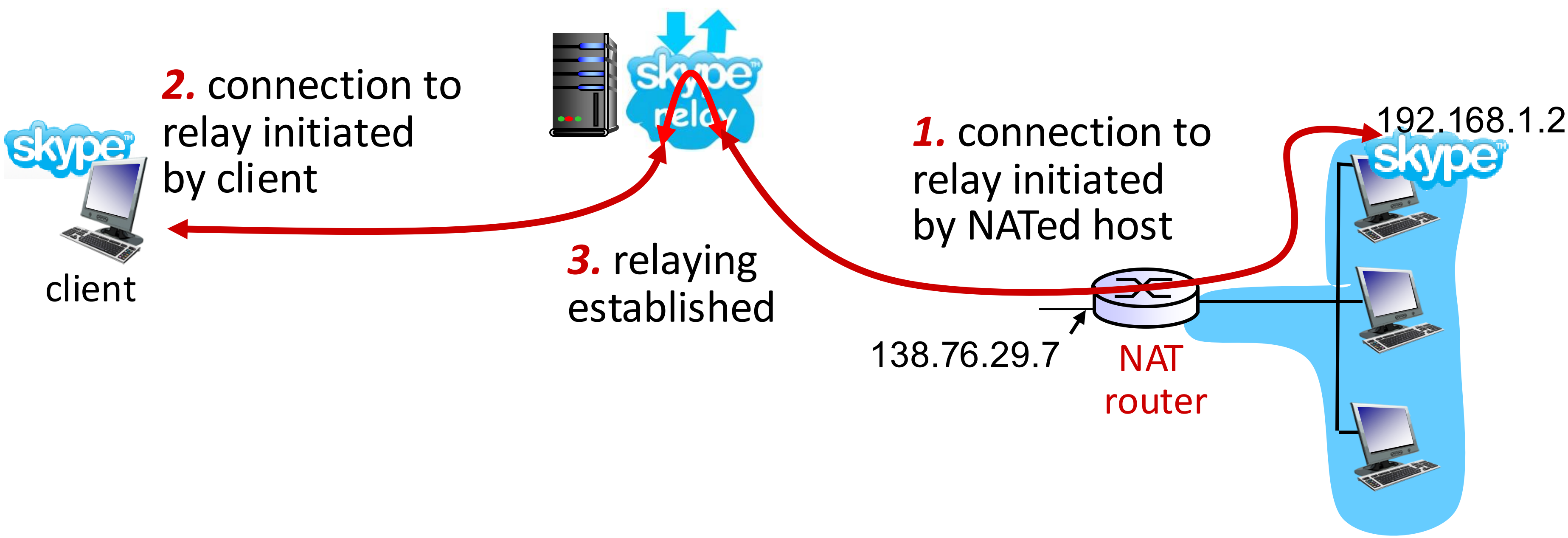
- *solution 2:* Universal Plug and Play (UPnP) Internet Gateway Device (IGD) Protocol. Allows NATed host to:
  - learn public IP address (138.76.29.7)
  - add/remove port mappings (with lease times)

  i.e., automate static NAT port map configuration



NAT Traversal Problem

- *solution 3:* relaying (used in Skype)
  - NATed 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*

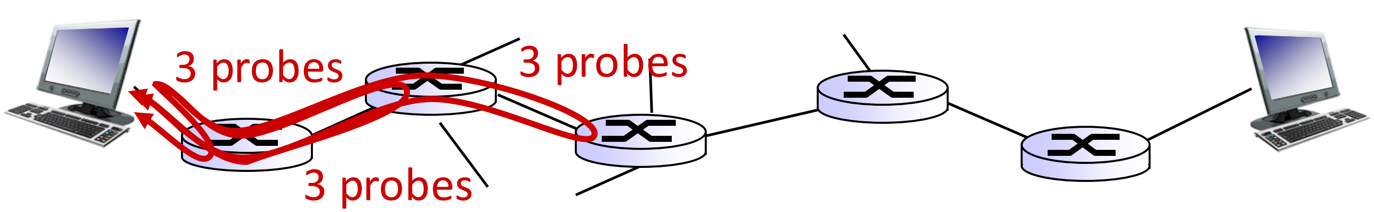
Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
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 set has TTL =1
  - second set has TTL=2, etc.
  - unlikely port number
- when *n*th set of datagrams arrives to nth router:
  - router discards datagrams
  - and sends source ICMP messages (type 11, code 0)
  - ICMP messages includes name of router & IP address
- when ICMP messages arrives, source records RTTs

stopping criteria:

- UDP segment eventually arrives at destination host
- destination returns ICMP "port unreachable" message (type 3, code 3)
- source stops



IPv6 – Motivation

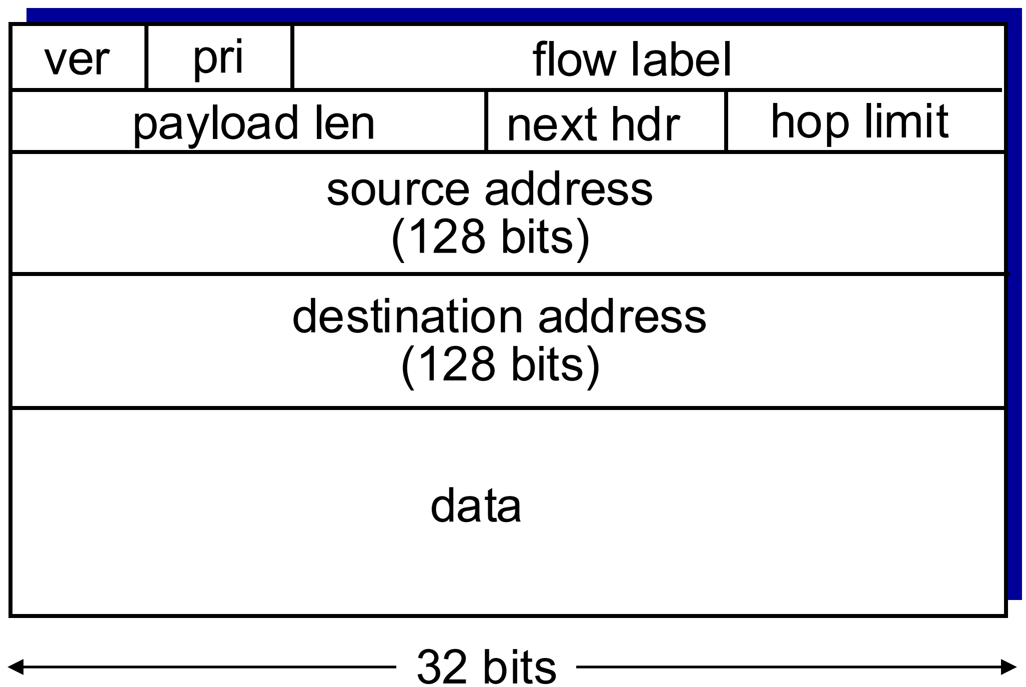
- *initial motivation:* 32-bit address space soon to be completely allocated.
- additional motivation:
  - header format helps speed processing/forwarding
  - header changes to facilitate QoS

IPv6 datagram format:

- fixed-length 40 byte header
- no fragmentation allowed

IPv6 Datagram Format

priority: identify priority among datagrams in flow
flow Label: identify datagrams in same "flow."
(concept of "flow" not well defined).
next header: identify upper layer protocol for data

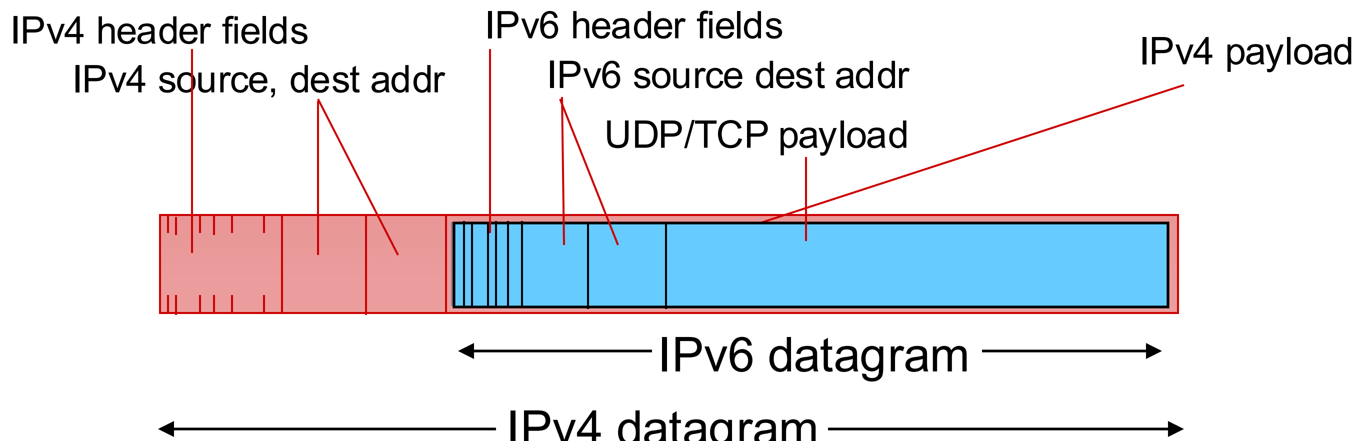


IPv6 Other Changes from IPv4

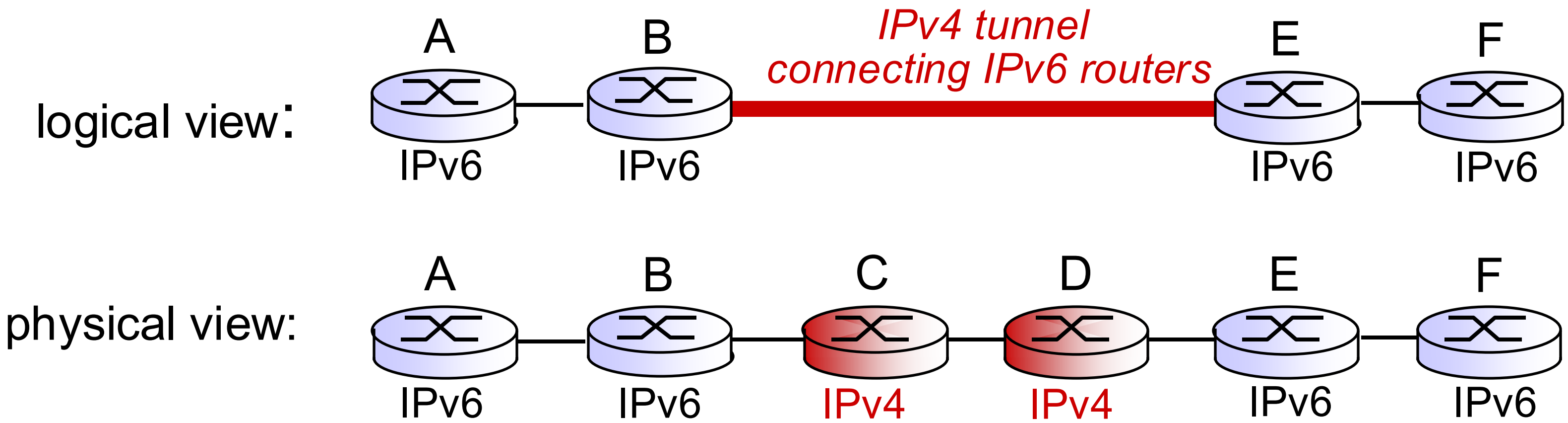
- *checksum*: removed entirely to reduce processing time at each hop
- *options:* allowed, but outside of header, indicated by "Next Header" field
- *ICMPv6:* new version of ICMP
  - additional message types, e.g. "Packet Too Big"
  - multicast group management functions

Transition from IPv4 to IPv6

- not all routers can be upgraded simultaneously
  - no "flag days"
  - how will network operate with mixed IPv4 and IPv6 routers?
- *tunneling:* IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers

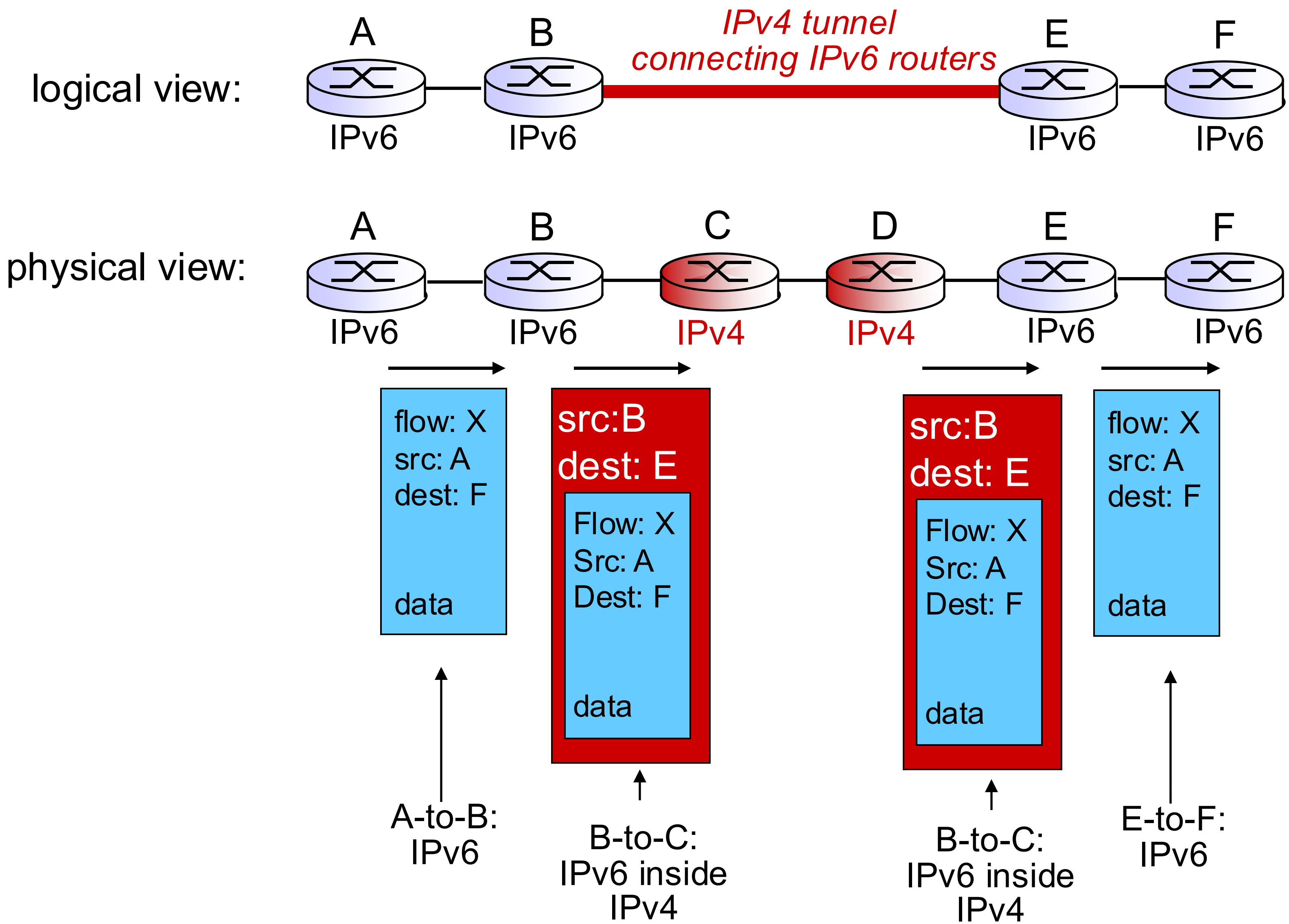


Tunneling



logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6

Tunneling

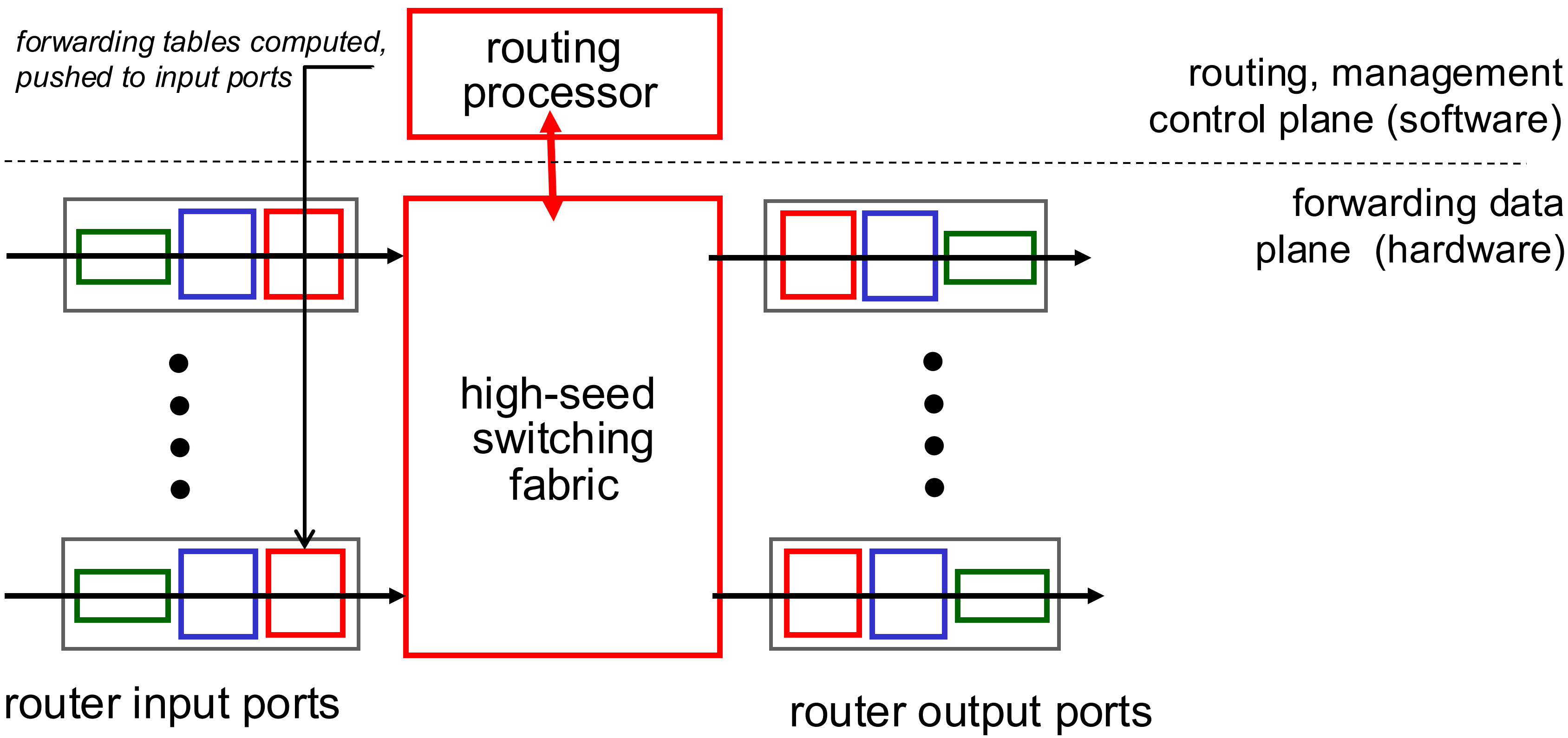


logical view:
A
B
IPv4 tunnel
connecting IPv6 routers
E
F
IPv6
IPv6
IPv6
IPv6
physical view:
A
B
C
D
E
F
IPv6
IPv6
IPv4
IPv4
IPv6
IPv6
flow: X
src: A
dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
src:B
dest: E
Flow: X
Src: A
Dest: F
data
flow: X
src: A
dest: F
data
A-to-B:
IPv6
B-to-C:
IPv6 inside
IPv4
B-to-C:
IPv6 inside
IPv4
E-to-F:
IPv6

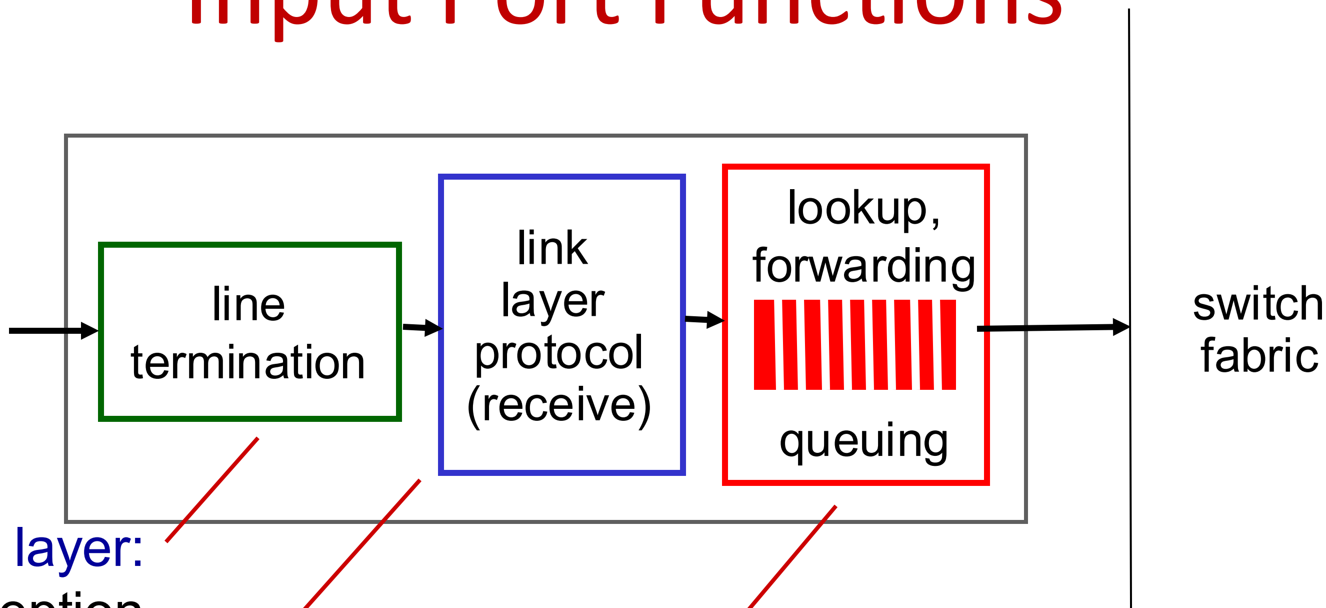
Router Architecture Overview

two key router functions:

- run routing algorithms/protocol (RIP, OSPF, BGP)
- *forwarding* datagrams from incoming to outgoing link



Input Port Functions



physical layer:
bit-level reception

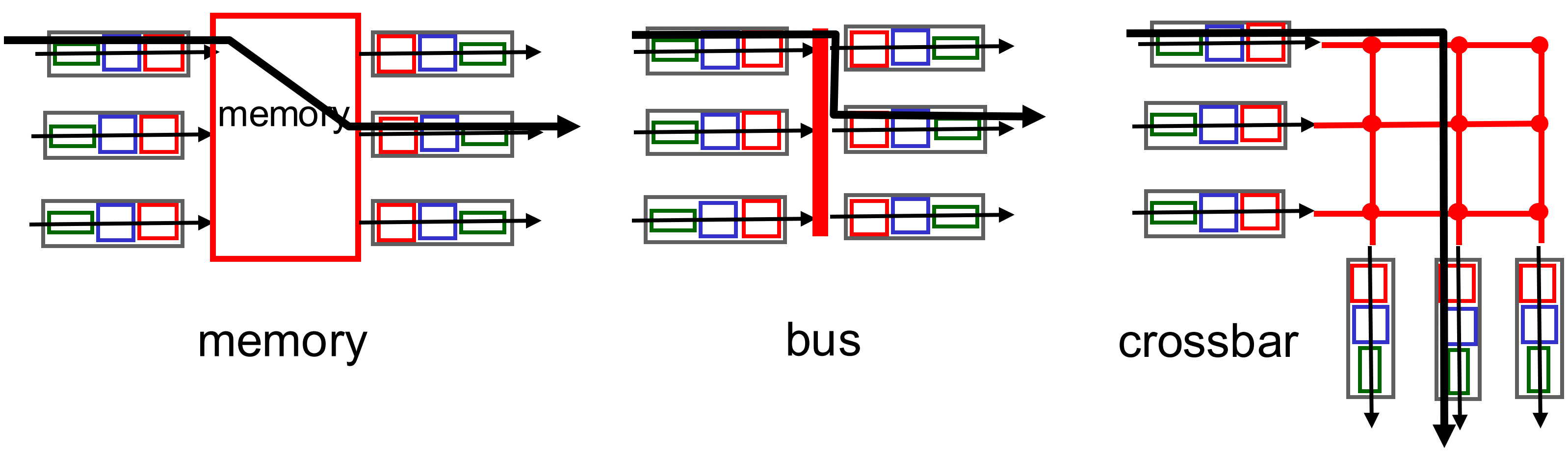
data link layer:
e.g., Ethernet
see chapter 5

decentralized switching:

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

Switching Fabrics

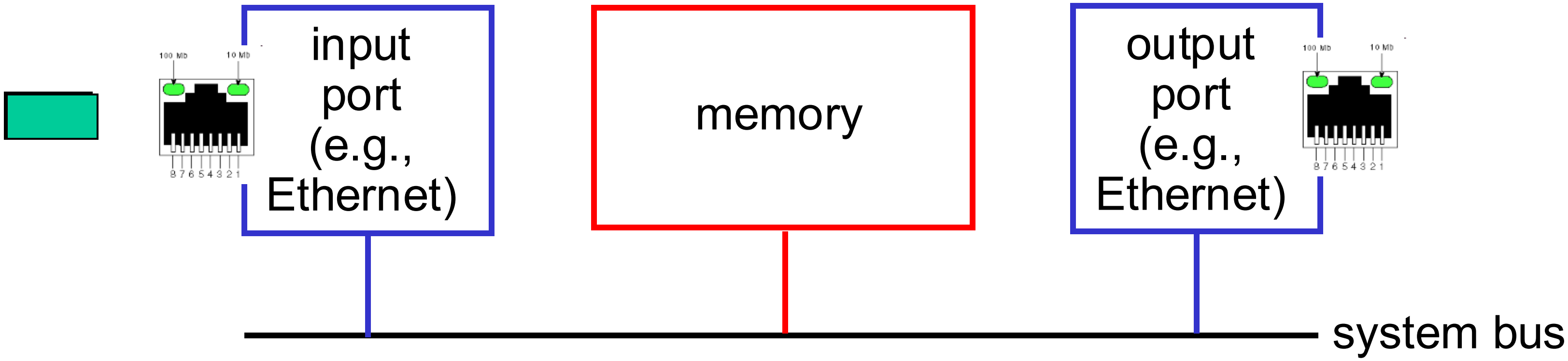
- transfer packet from input buffer to appropriate output buffer
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three types of switching fabrics



Switching via Memory

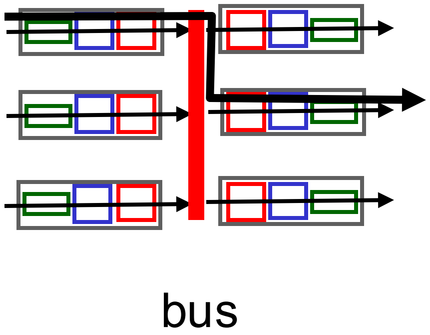
first generation routers:

- 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)



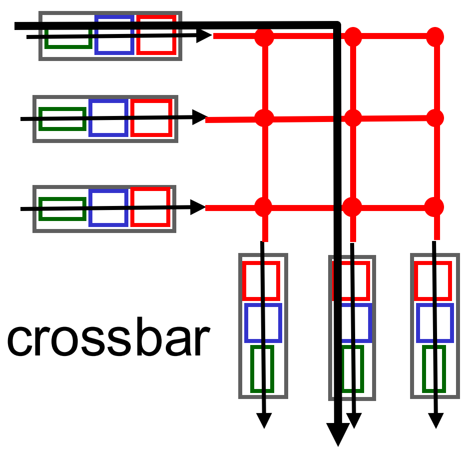
Switching via a Bus

- 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

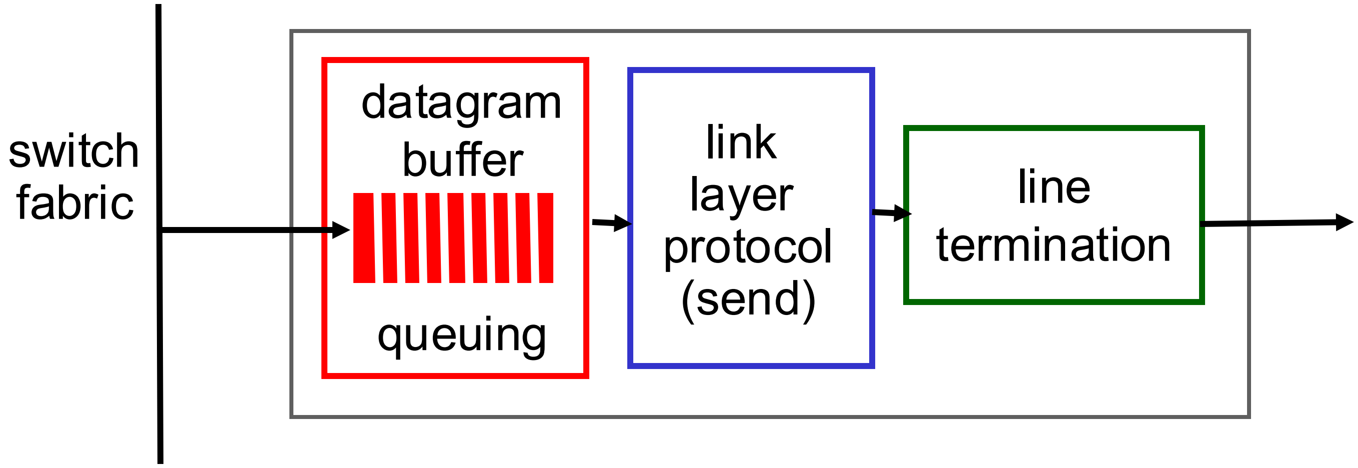


Switching via Interconnection Network

- overcome bus bandwidth limitations
- 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

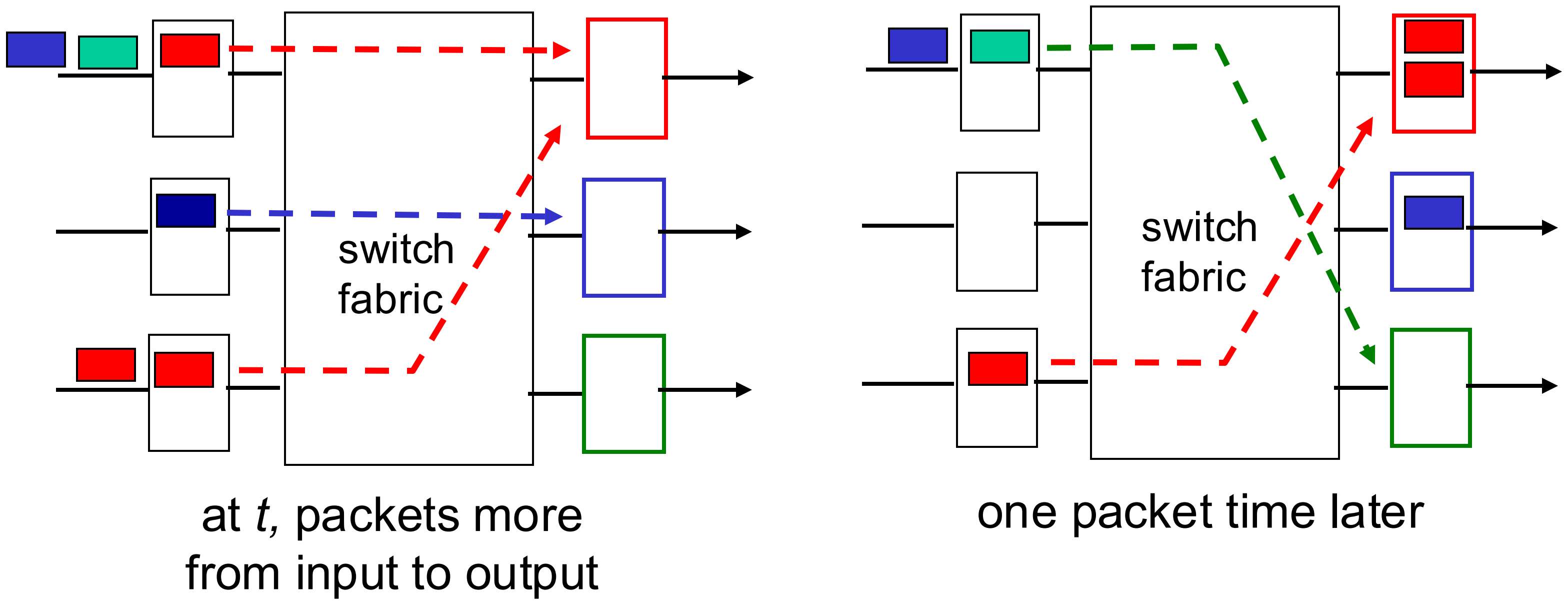


Output Ports



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

Output Port Queuing



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

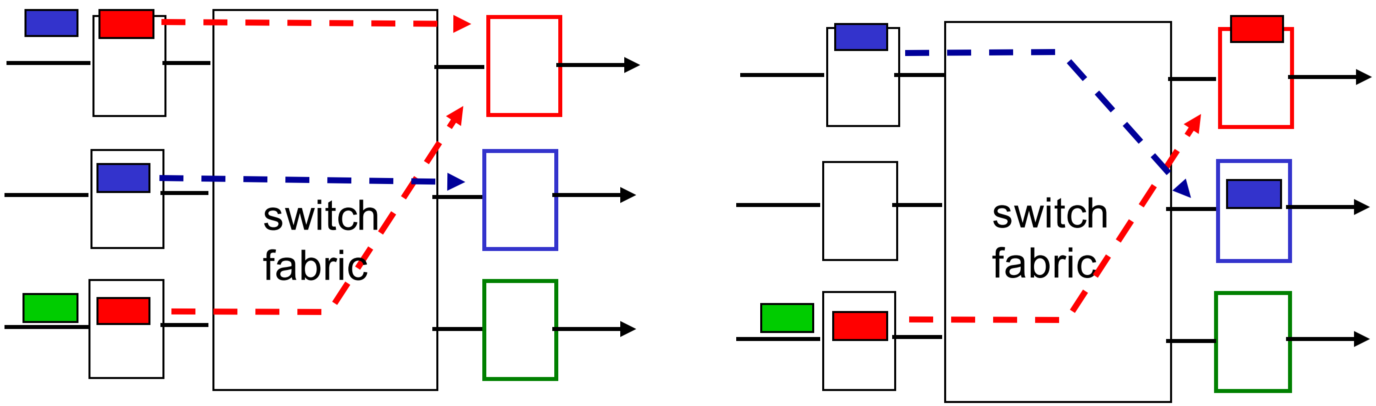
How much Buffering?

- 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

$$\frac{RTT \cdot C}{\sqrt{N}}$$

Input Port Queuing

- fabric slower than input ports combined -> queuing may occur at input queues
  - *queuing 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