

Chapter 4

Network Layer

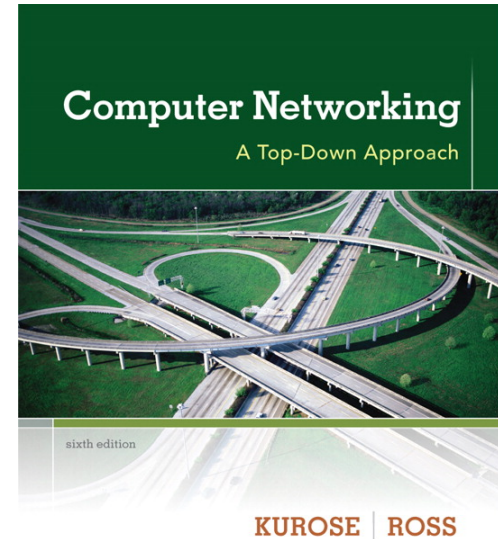
A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you see the animations; and can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- ❖ If you use these slides (e.g., in a class) that you mention their source (after all, we'd like people to use our book!)
- ❖ If you post any slides on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

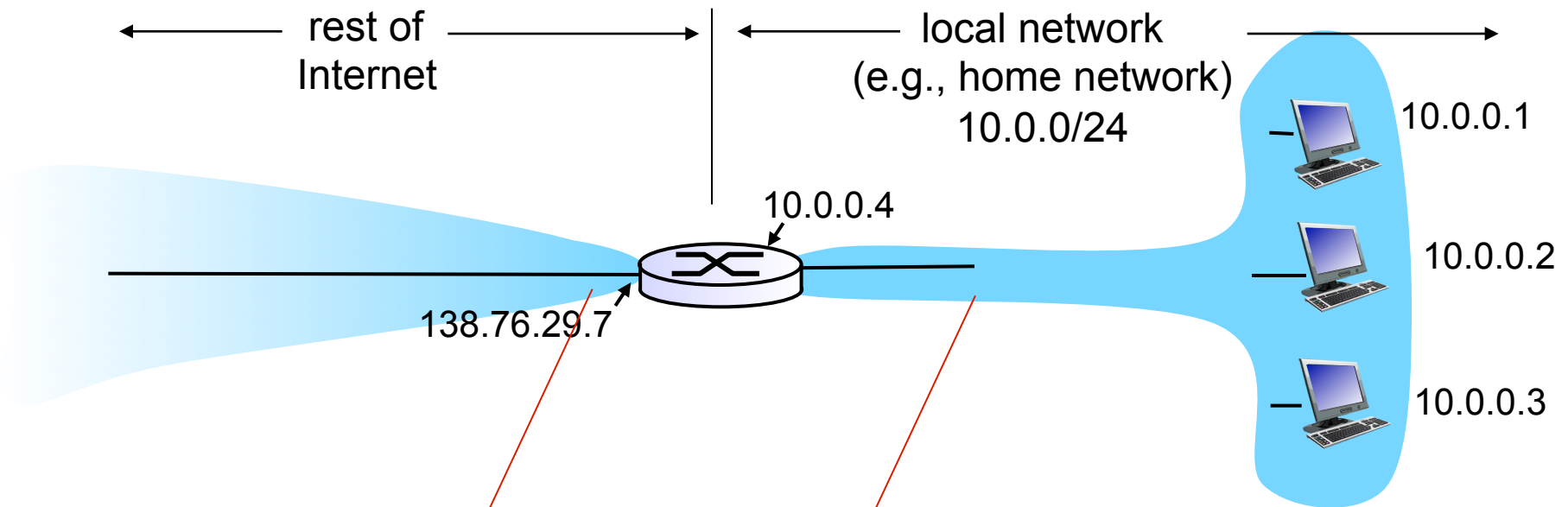
Thanks and enjoy! JFK/KWR

© All material copyright 1996-2012
J.F. Kurose and K.W. Ross, All Rights Reserved



**Computer
Networking: A Top
Down Approach**
6th edition
Jim Kurose, Keith Ross
Addison-Wesley
March 2012

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:

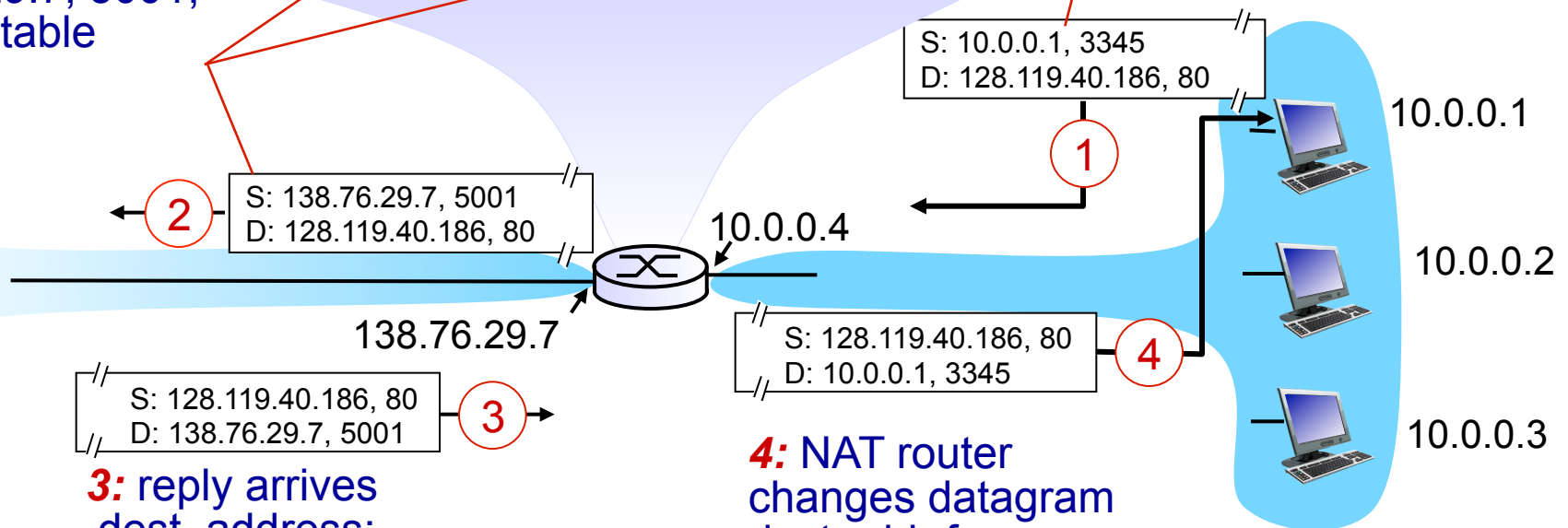
- *outgoing datagrams: replace* (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #)
... remote clients/servers will respond using (NAT IP address, new port #) as destination addr
- *remember (in NAT translation table)* every (source IP address, port #) to (NAT IP address, 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
 - *If no matching table entry, drop the datagram!*

NAT: network address translation

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

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



3: reply arrives
dest. address:
138.76.29.7, 5001

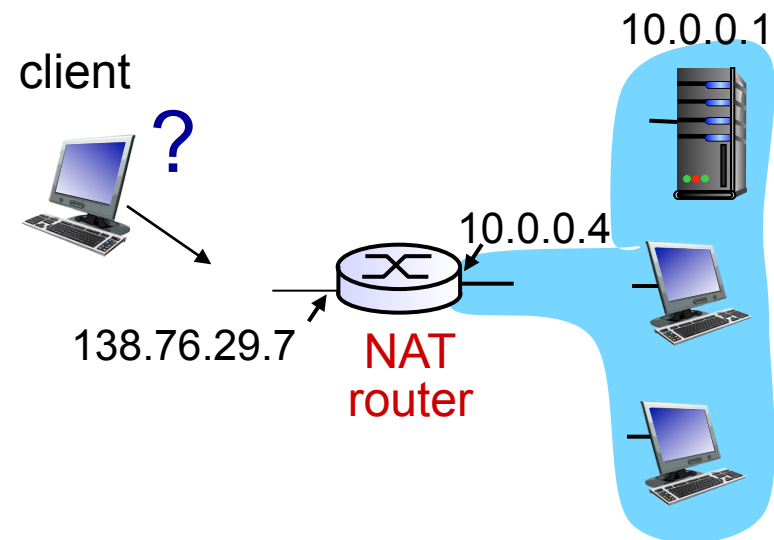
4: NAT router changes datagram dest addr from 138.76.29.7, 5001 to 10.0.0.1, 3345

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, e.g., P2P applications
 - Address shortage should instead be solved by IPv6
- ❖ BUT: NAT provides a measure of security for most home networks (why?)

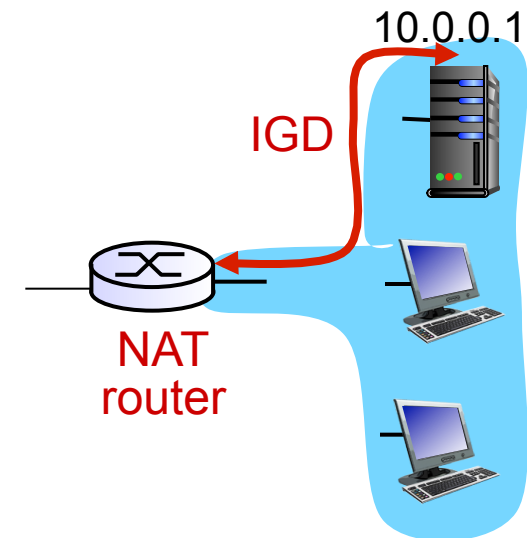
NAT traversal problem

- ❖ 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., (138.76.29.7, port 2500) always forwarded to 10.0.0.1 port 25000
 - **Ugly**: most folks don't know how to do this



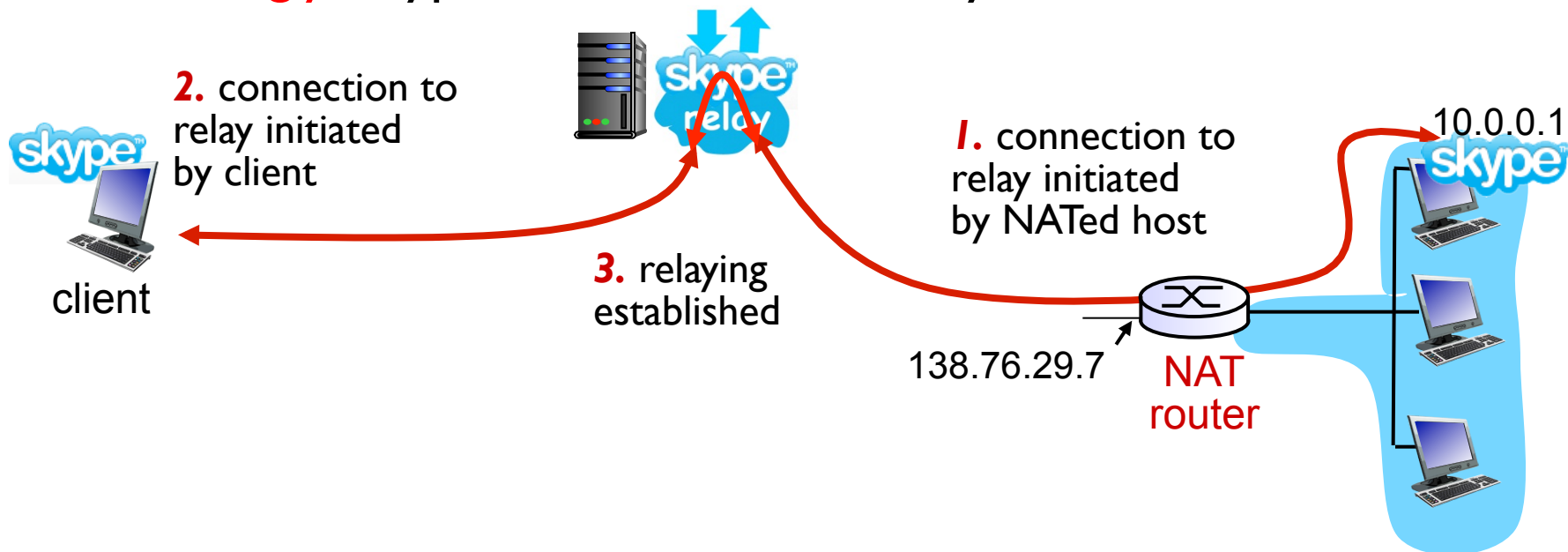
NAT traversal problem

- ❖ **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
 - **ugly:** skype knows where/when you are



Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol

- datagram format
- IPv4 addressing
- **ICMP**
- **IPv6**

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

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

<u>Type</u>	<u>Code</u>	<u>description</u>
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

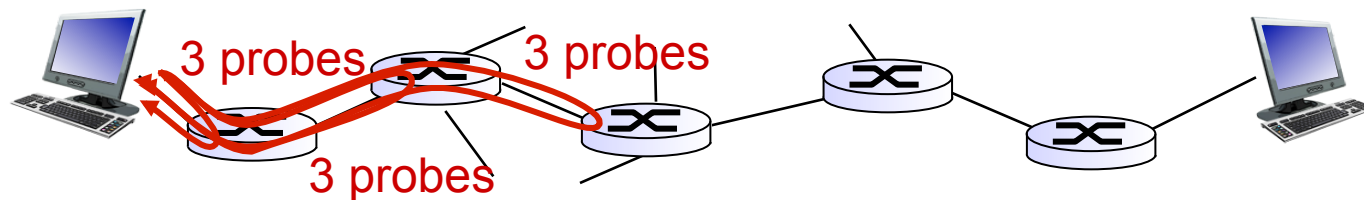
Traceroute and ICMP

- ❖ 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 n th 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



Exercise: Path MTU Discovery

- ❖ MTU = “Maximum Transmission Unit”
 - Size of largest datagram that can be transmitted over a channel (e.g., Ethernet is 1500 bytes, WiFi is 7981)
- ❖ Given the ICMP message types, and what you know about IP’s fragmentation function, **devise a procedure to determine the “Path MTU”, i.e., the smallest MTU along a path from source to destination**

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

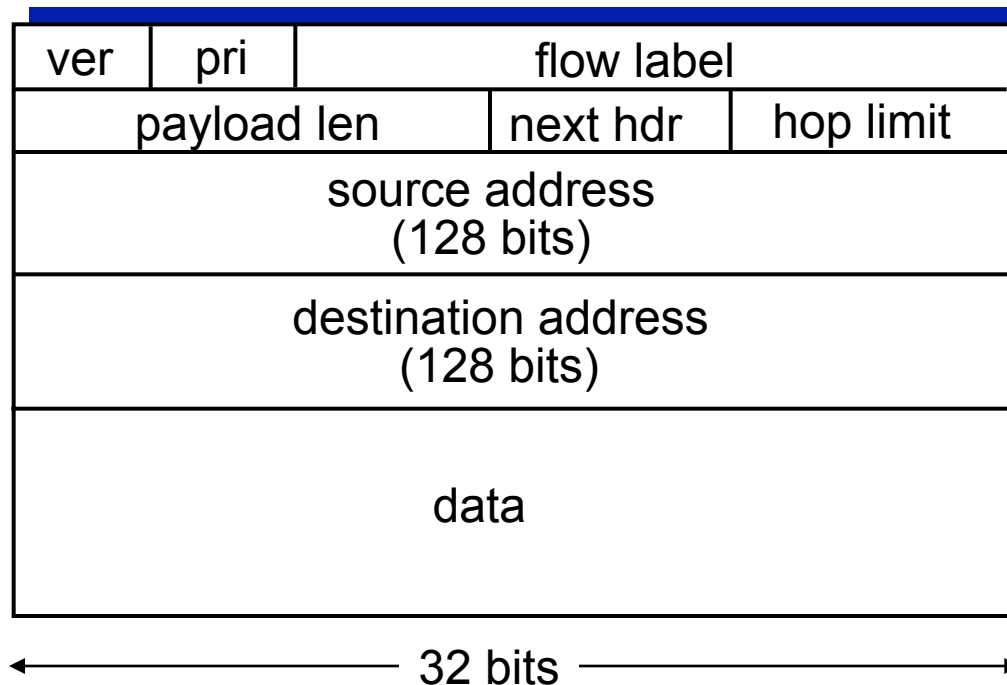
IPv6 datagram format

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

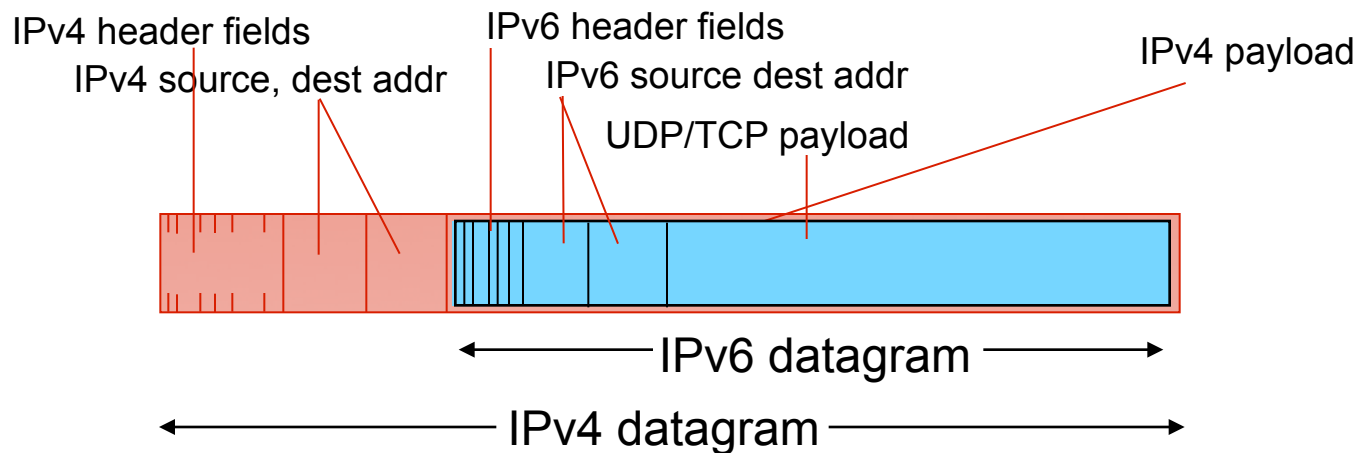


Other changes from IPv4

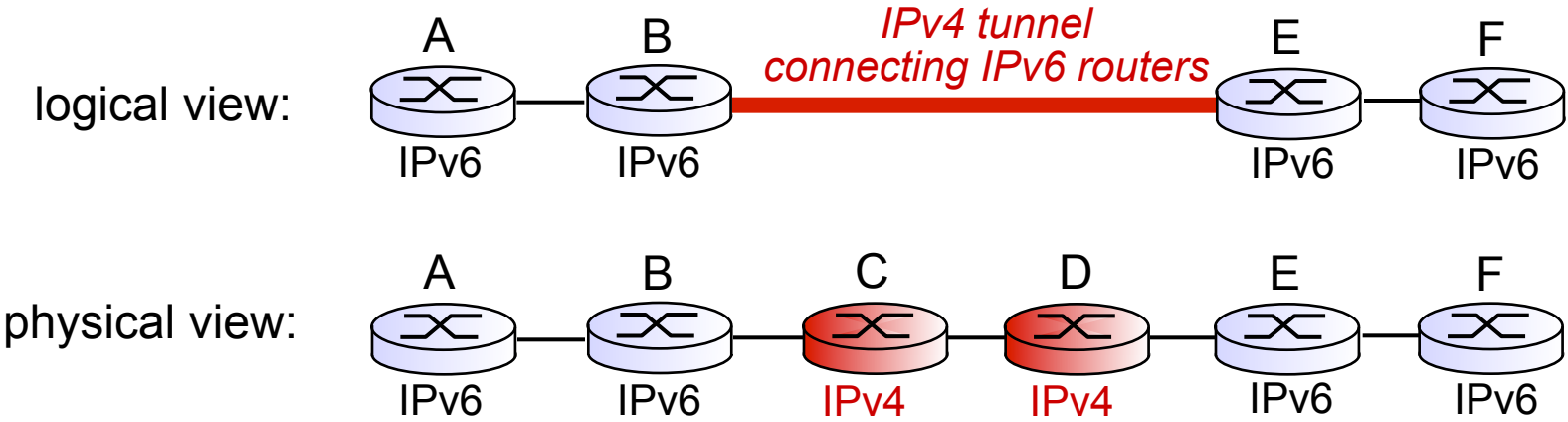
- ❖ *checksum*: removed entirely to reduce processing time at each hop
- ❖ *options*: allowed, but outside of header, indicated by “Next Header” field
- ❖ *ICMPv6*: new version of ICMP
 - additional message types, e.g. “Packet Too Big”
 - multicast group management functions

Transition from IPv4 to IPv6

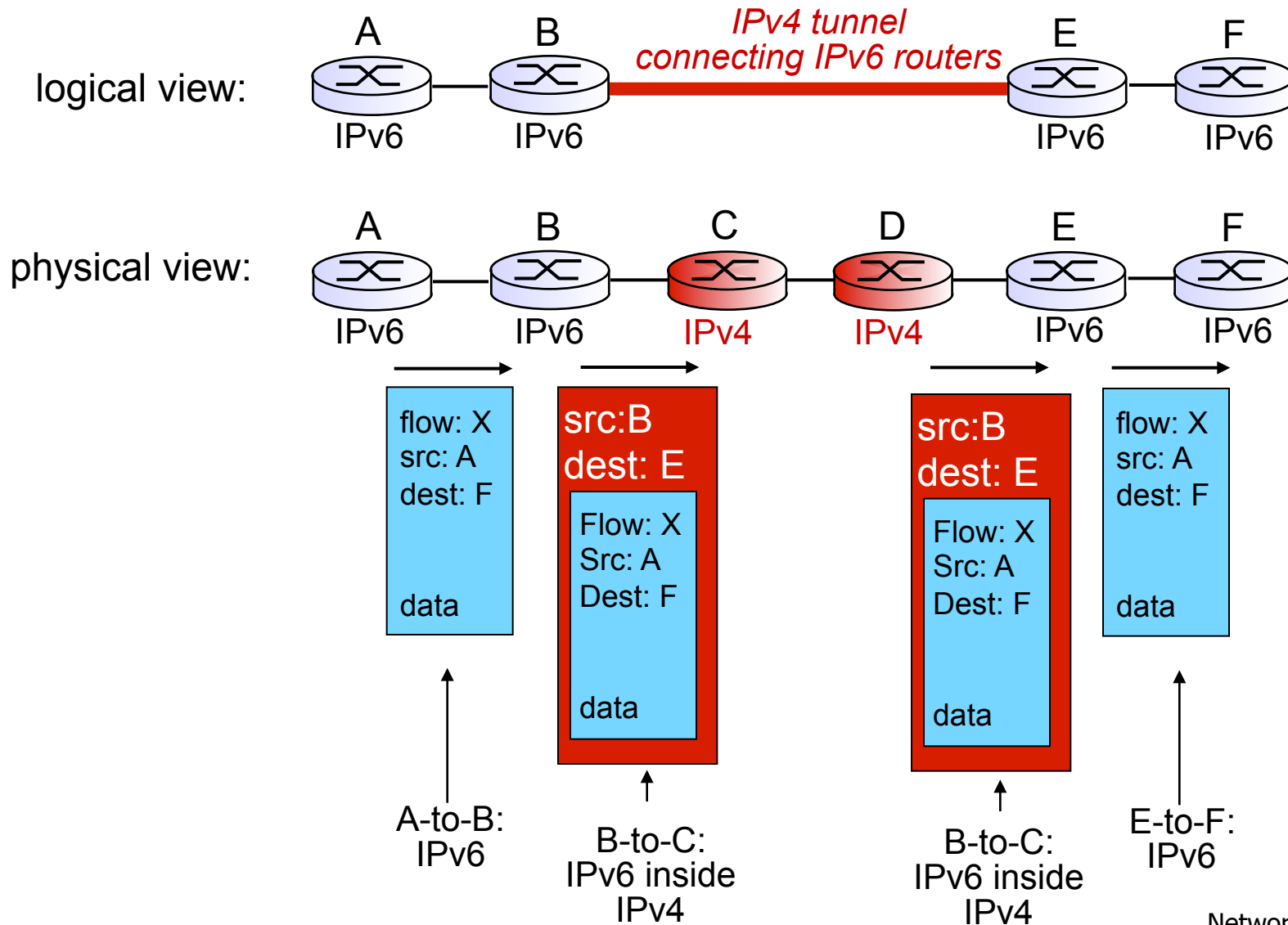
- ❖ not all routers can be upgraded simultaneously
 - no “flag days”
 - how will network operate with mixed IPv4 and IPv6 routers?
- ❖ **tunneling**: IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers



Tunneling



Tunneling



Chapter 4: outline

4.1 introduction

4.2 virtual circuit and datagram networks

4.3 what's inside a router

4.4 IP: Internet Protocol

- datagram format
- IPv4 addressing
- ICMP
- IPv6

4.5 routing algorithms

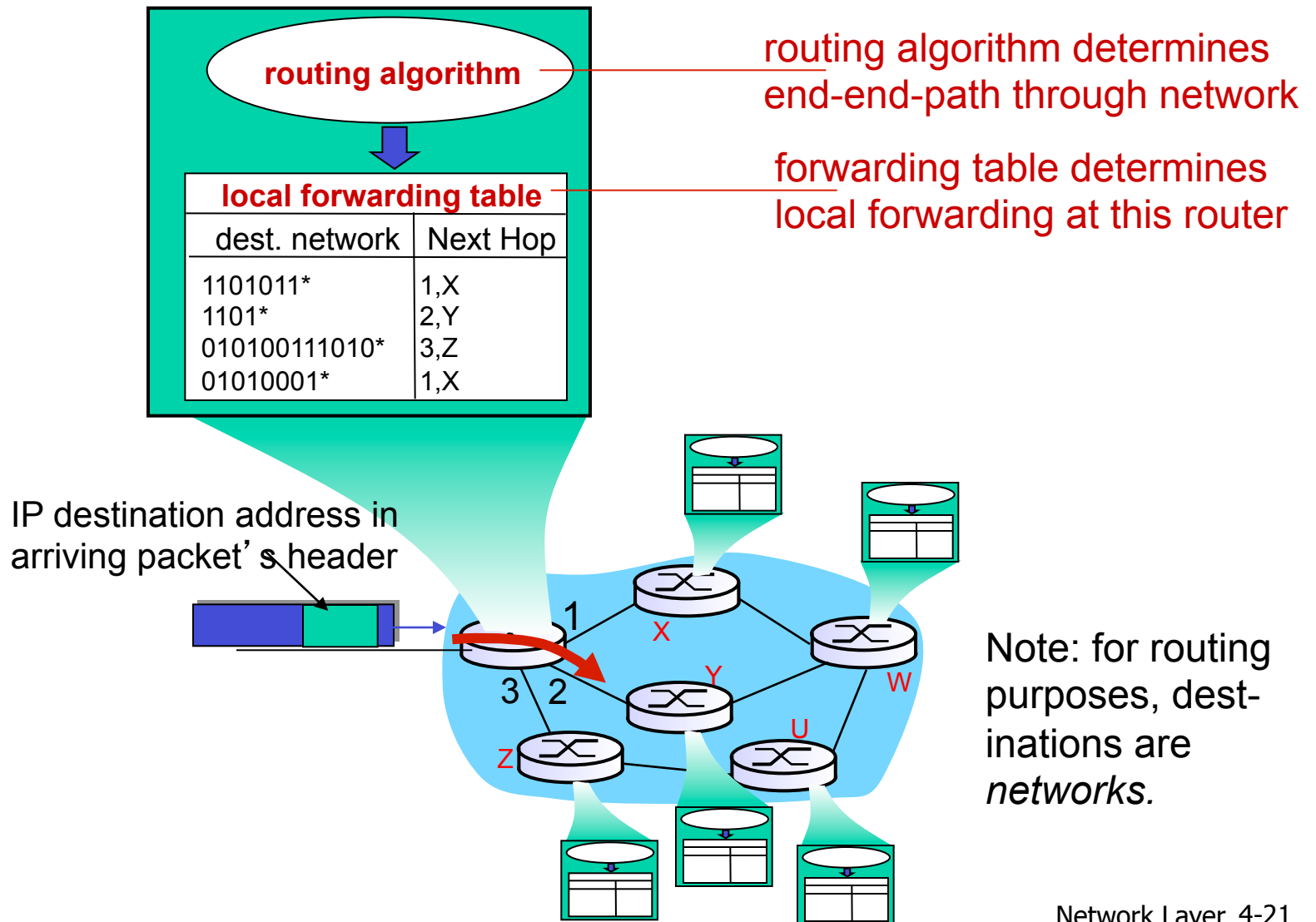
- link state
- distance vector
- hierarchical routing

4.6 routing in the Internet

- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

Interplay between routing, forwarding



Exercise: Routing Algorithms

- ❖ Imagine that you are routers
 - Each of your interfaces is connected to a network
 - Each network has an IP network number (prefix)
- ❖ You already know:
 - Your own interfaces' IP addresses
 - The prefixes assigned to your attached networks
- ❖ How will you (routers) **work together to construct a forwarding table at each router** containing an entry for every attached network?

