COMPUTER NETWORKS

Ashwini M Joshi

Department of Computer Science and Engineering

COMPUTER NETWORKS

Network Layer and Internet Protocol

Ashwini M Joshi (Ph.D.)
Department of Computer Science and Engineering

Unit – 4 Network Layer and Internet Protocol

Chapter Goals:

Understand principles behind network layer services:

- network layer service models
- forwarding versus routing
- how a router works
- routing (path selection)
- broadcast, multicast

Instantiation, implementation in the Internet

Unit – 4 Network Layer and Internet Protocol

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

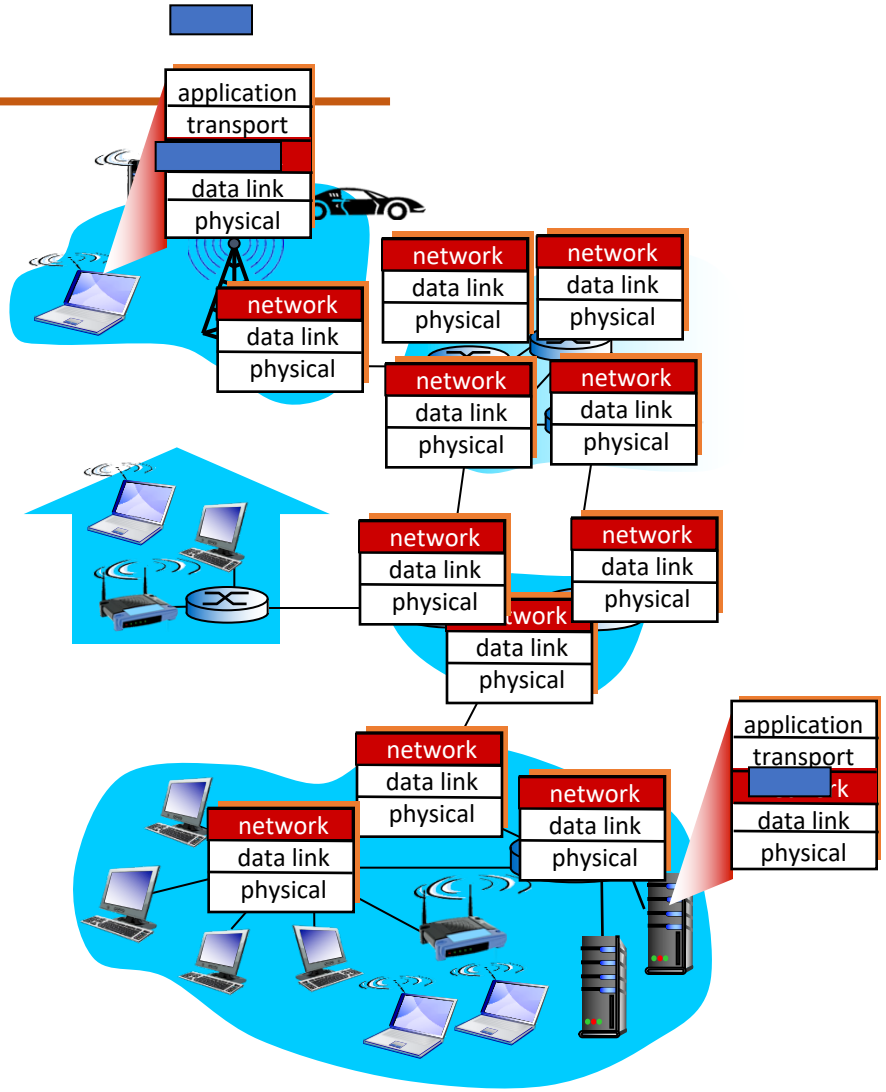
4.6 Routing in the Internet

- RIP
- OSPF
- BGP

4.7 Broadcast and multicast routing

Network Layer and Internet Protocol

- transport segment from sending to receiving host
- on sending side encapsulates segments into datagrams
- on receiving side, delivers segments to transport layer
- network layer protocols in *every* host, router
- router examines header fields in all IP datagrams passing through it



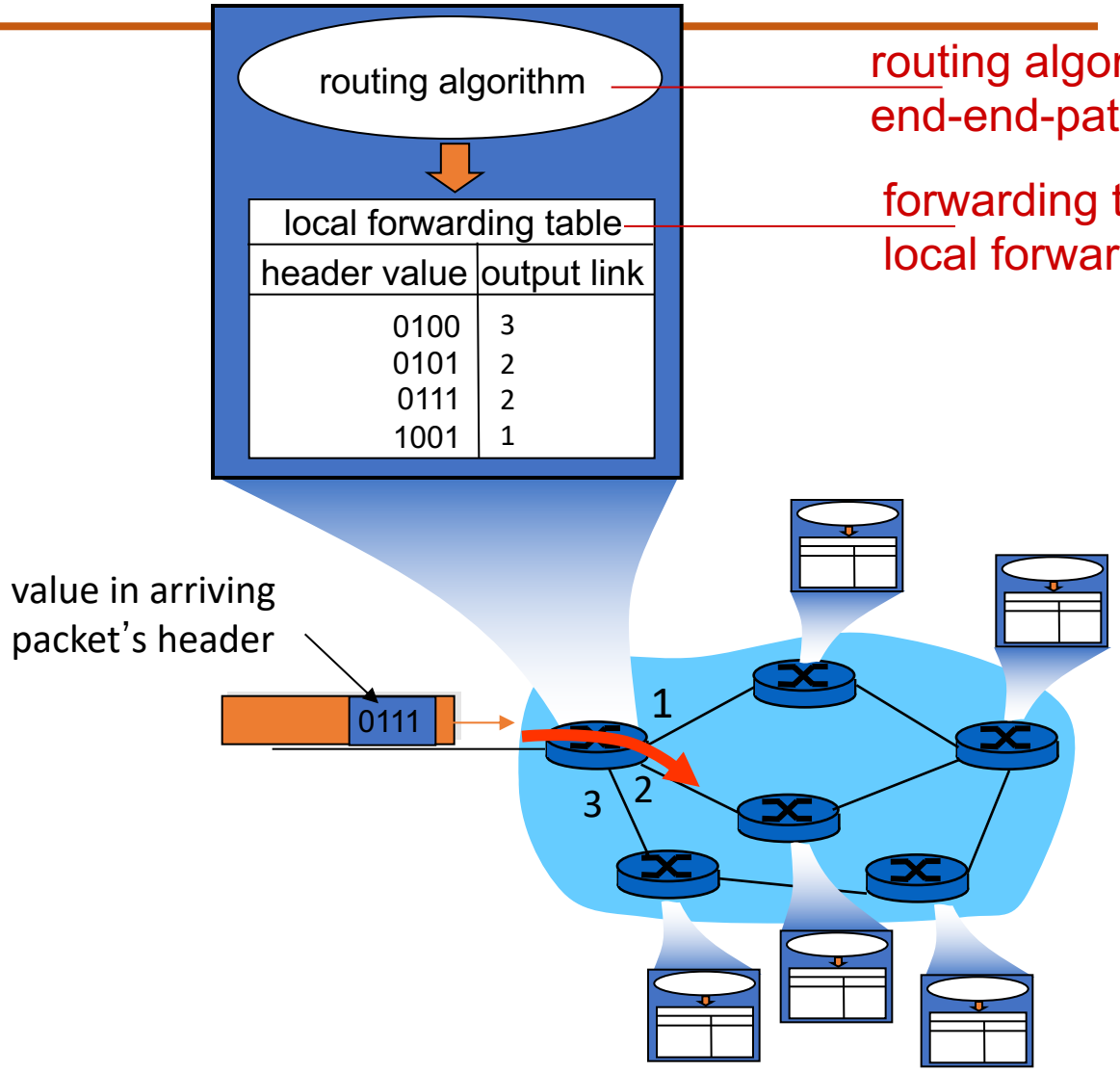
Two key network-layer functions

- *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

Interplay between routing and forwarding



Connection setup

Connection set-up can be the 3rd important function in *some* network architectures:

- ATM, frame relay, X.25

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
- restrictions on changes in inter-packet spacing

Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no

Internet "best effort" service model

No guarantees on:

i. successful datagram delivery to destination
ii. timing or order of delivery
iii. bandwidth available to end-end flow

Network-layer service model

Network Architecture	Service Model	Quality of Service (QoS) Guarantees ?			
		Bandwidth	Loss	Order	Timing
Internet	best effort	none	no	no	no
ATM	Constant Bit Rate	Constant rate	yes	yes	yes
ATM	Available Bit Rate	Guaranteed min	no	yes	no
Internet	Intserv Guaranteed (RFC 1633)	yes	yes	yes	yes
Internet	Diffserv (RFC 2475)	possible	possibly	possibly	no

COMPUTER NETWORKS

Ashwini M Joshi (Ph.D.)
Department of Computer Science and Engineering

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

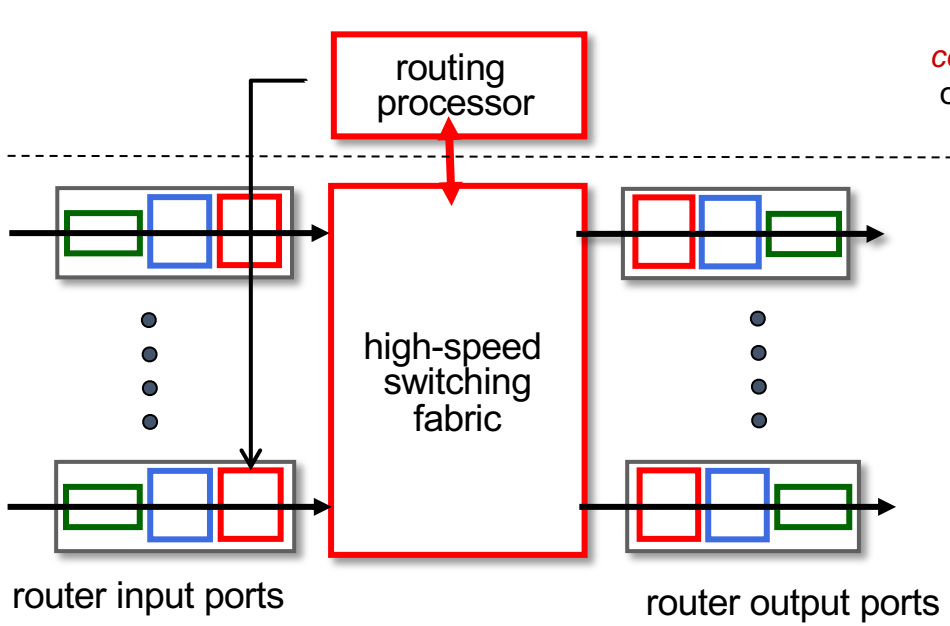
4.6 routing in the Internet

- RIP
- OSPF
- BGP

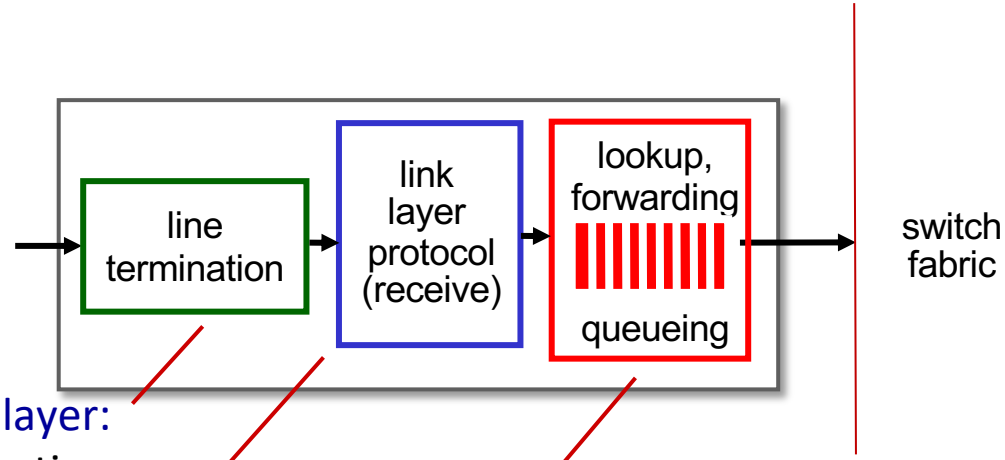
4.7 broadcast and multicast routing

Router architecture overview

high-level view of generic router architecture:



Input port functions



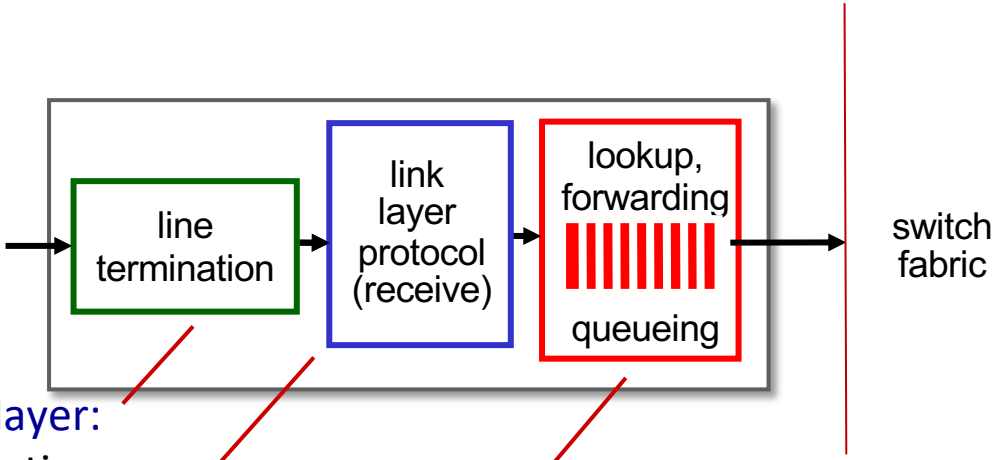
physical layer:
bit-level reception

link layer:
e.g., Ethernet
(chapter 6)

decentralized switching:

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

Input port functions



physical layer:
bit-level reception

link layer:
e.g., Ethernet
(chapter 6)

decentralized switching:

- using header field values, lookup output port using forwarding table in input port memory *("match plus action")*
- destination-based forwarding: forward based only on destination IP address (traditional)
- generalized forwarding: forward based on any set of header field values

Destination-based forwarding

forwarding table

Destination Address Range	Link Interface
11001000 00010111 00010000 00000000 through 11001000 00010111 00010000 00000100	0
11001000 00010111 00010000 00000101 through 11001000 00010111 00010000 00000111	3
11001000 00010111 00011000 11111111	
11001000 00010111 00011001 00000000 through 11001000 00010111 00011111 11111111	2
otherwise	3

Q: but what happens if ranges don't divide up so nicely?

Longest prefix matching

longest prefix match

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011** ********	2
otherwise	3

examples:

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

COMPUTER NETWORKS

Longest prefix matching

longest prefix match

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011*** ********	2
otherwise	3

match!

examples:

11001000 00010111 00010110 10100001 which interface?

11001000 00010111 00011000 10101010 which interface?

COMPUTER NETWORKS

Longest prefix matching

longest prefix match

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011** ********	2
otherwise	3

match!

examples:

11001000 00010110 10100001	which interface?
11001000 00010111 00011000 10101010	which interface?

Longest prefix matching

longest prefix match

when looking for forwarding table entry for given destination address, use *longest* address prefix that matches destination address.

Destination Address Range	Link interface
11001000 00010111 00010** ********	0
11001000 00010111 00011000 ********	1
11001000 00010111 00011** ********	2
otherwise	3

match!

examples:

11001000 00010111 00010110 10100001 which interface?

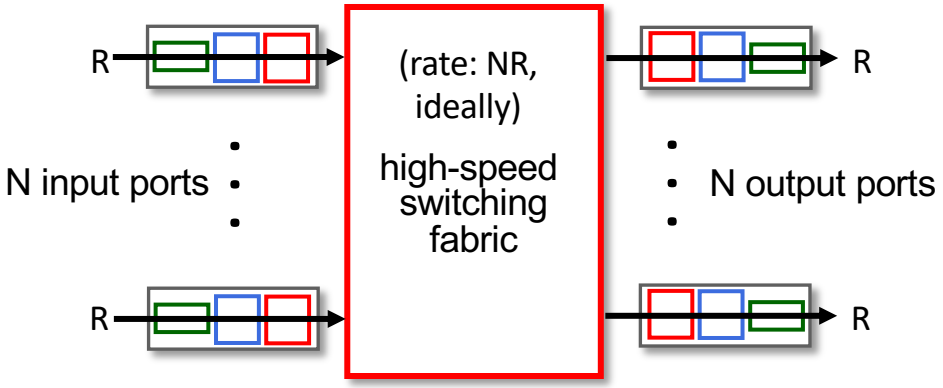
11001000 00010111 00011000 10101010 which interface?

Longest prefix matching

- we'll see *why* longest prefix matching is used shortly, when we study addressing
- longest prefix matching: often performed using ternary content addressable memories (TCAMs)
  - *content addressable:* present address to TCAM: retrieve address in one clock cycle, regardless of table size
  - Cisco Catalyst: ~1M routing table entries in TCAM

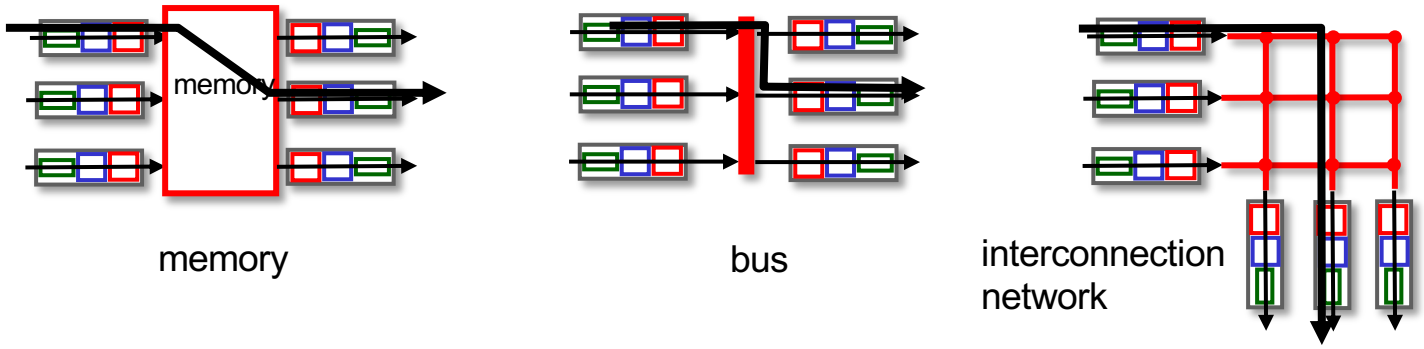
Switching fabrics

- transfer packet from input link to appropriate output link
- 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



Switching fabrics

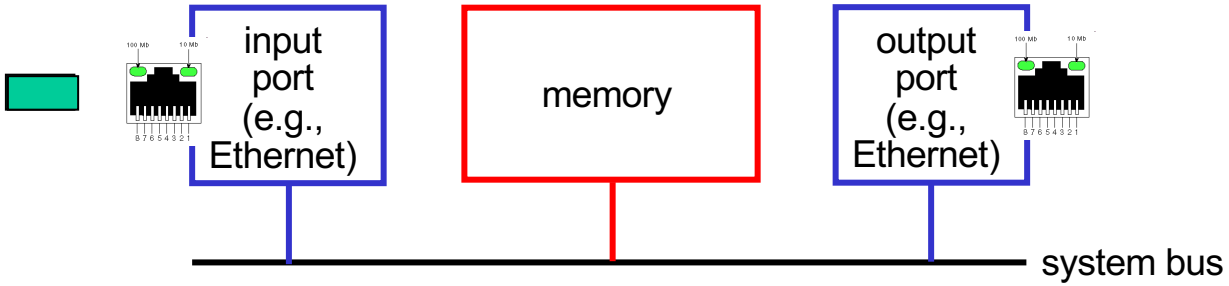
- transfer packet from input link to appropriate output link
- switching rate: rate at which packets can be transfer from inputs to outputs
  - often measured as multiple of input/output line rate
  - N inputs: switching rate N times line rate desirable
- three major types of switching fabrics:



Switching via memory

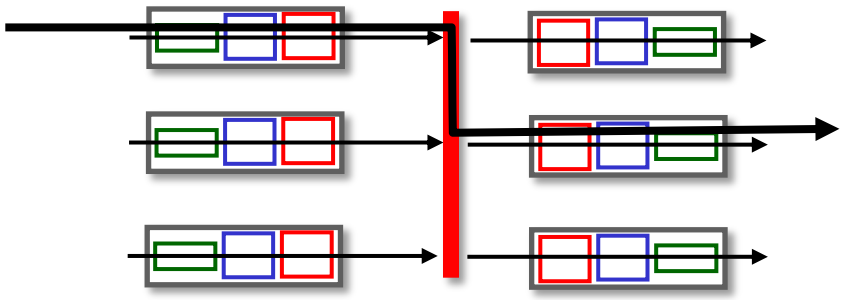
first generation routers:

- 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)



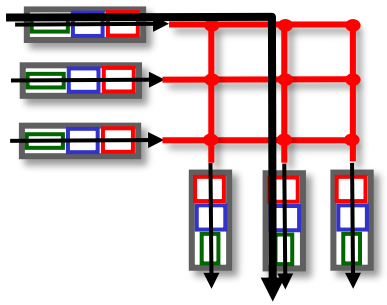
Switching via a bus

- 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 routers

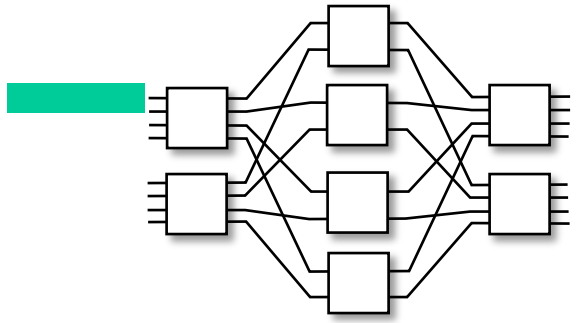


Switching via interconnection network

- Crossbar, Clos networks, other interconnection nets initially developed to connect processors in multiprocessor
- multistage switch: *nxn* switch from multiple stages of smaller switches
- exploiting parallelism:
  - fragment datagram into fixed length cells on entry
  - switch cells through the fabric, reassemble datagram at exit



3x3 crossbar

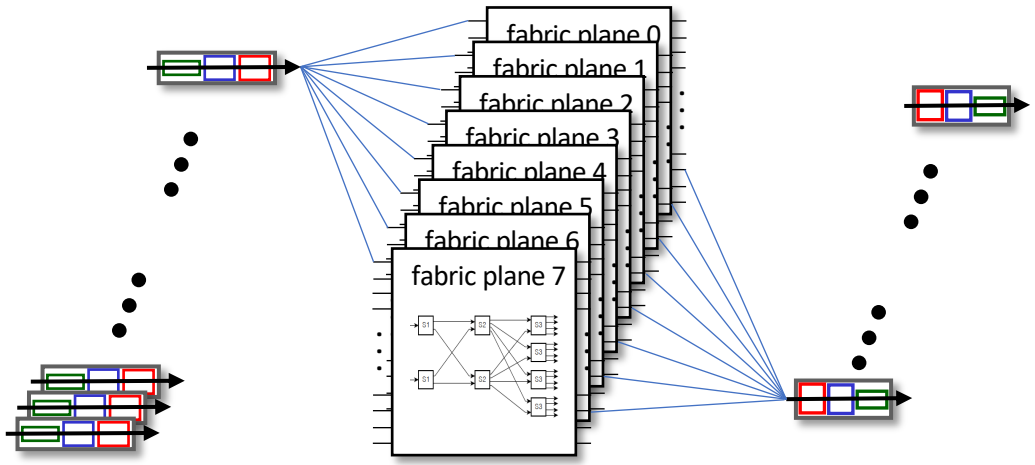


8x8 multistage switch
built from smaller-sized switches

Switching via interconnection network

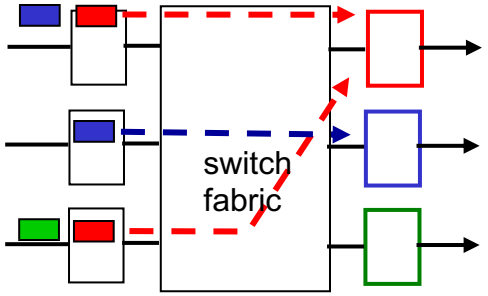
- scaling, using multiple switching “planes” in parallel:
  - speedup, scaleup via parallelism

- Cisco CRS router:
  - basic unit: 8 switching planes
  - each plane: 3-stage interconnection network
  - up to 100’s Tbps switching capacity

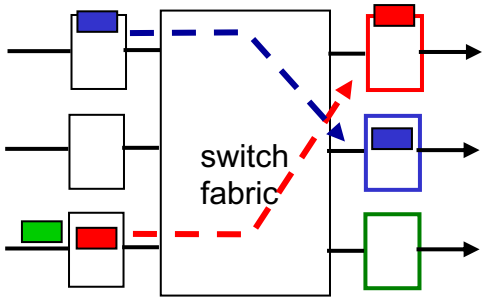


Input port queuing

- If switch 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



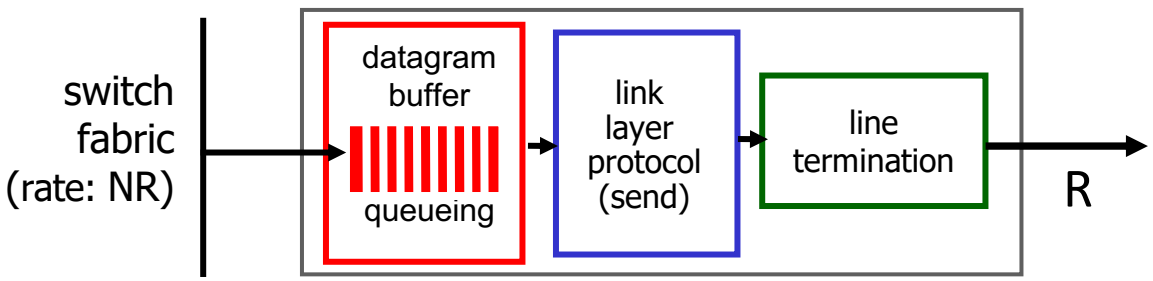
output port contention: only one red datagram can be transferred. lower red packet is *blocked*



one packet time later: green packet experiences HOL blocking

COMPUTER NETWORKS

Output port queuing



This is a really important slide

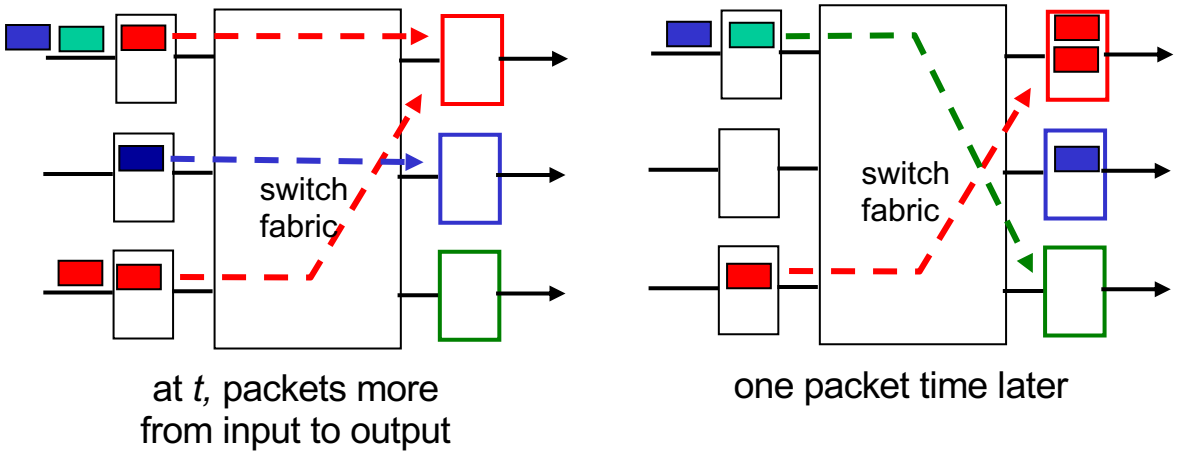
- *Buffering* required when datagrams arrive from fabric faster than link transmission rate. *Drop policy:* which datagrams to drop if no free buffers?

→ Datagrams can be lost due to congestion, lack of buffers

- *Scheduling discipline* chooses among queued datagrams for transmission

→ Priority scheduling – who gets best performance, network neutrality

Output port queuing



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

How much buffering?

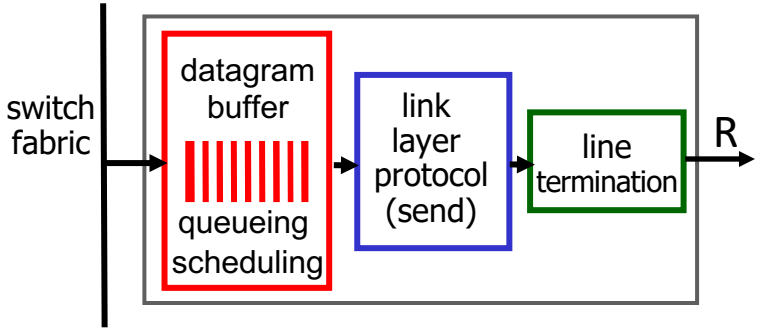
- RFC 3439 rule of thumb: average buffering equal to “typical” RTT (say 250 msec) times link capacity C
  - e.g., C = 10 Gbps link: 2.5 Gbit buffer

- more recent recommendation: with *N* flows, buffering equal to

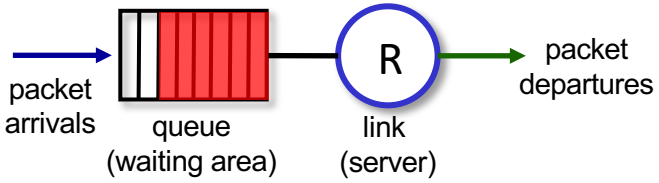
$$\frac{RTT \cdot C}{\sqrt{N}}$$

- but *too* much buffering can increase delays (particularly in home routers)
  - long RTTs: poor performance for realtime apps, sluggish TCP response
  - recall delay-based congestion control: “keep bottleneck link just full enough (busy) but no fuller”

Buffer Management



Abstraction: queue



buffer management:

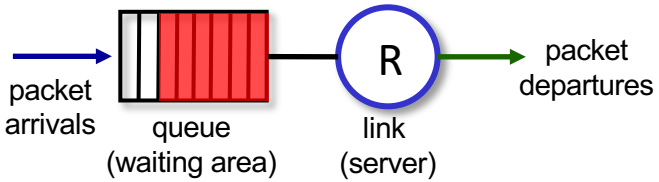
- drop: which packet to add, drop when buffers are full
  - tail drop: drop arriving packet
  - priority: drop/remove on priority basis
- marking: which packets to mark to signal congestion (ECN, RED)

Packet Scheduling: FCFS

packet scheduling:
deciding which packet to send next on link

- first come, first served
- priority
- round robin
- weighted fair queueing

Abstraction: queue



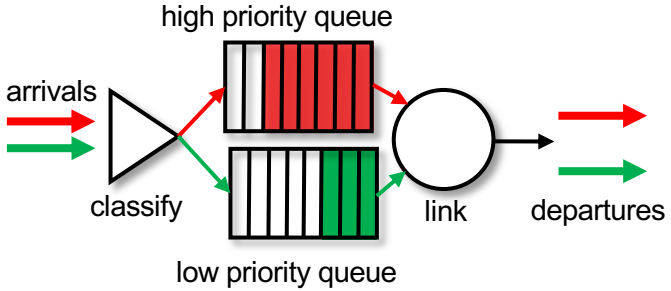
FCFS: packets transmitted in order of arrival to output port

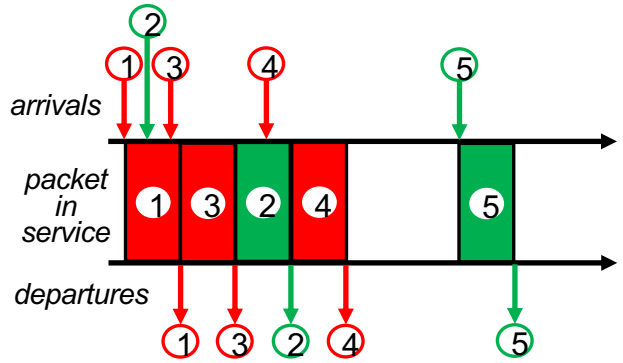
- also known as: First-in-first-out (FIFO)
- real world examples?

Scheduling policies: priority

Priority scheduling:

- arriving traffic classified, queued by class
  - any header fields can be used for classification
- send packet from highest priority queue that has buffered packets
  - FCFS within priority class

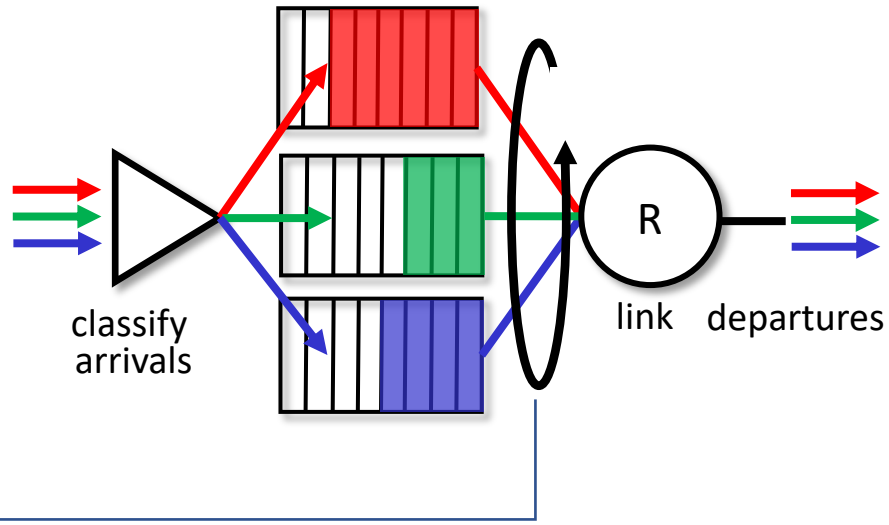




Scheduling policies: round robin

Round Robin (RR) scheduling:

- arriving traffic classified, queued by class
  - any header fields can be used for classification
- server cyclically, repeatedly scans class queues, sending one complete packet from each class (if available) in turn



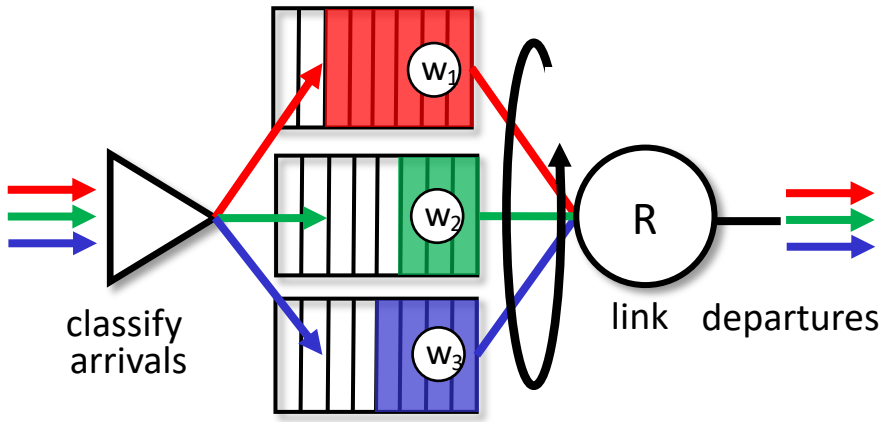
Scheduling policies: weighted fair queueing

Weighted Fair Queuing (WFQ):

- generalized Round Robin
- each class, *i*, has weight, w_i, and gets weighted amount of service in each cycle:

$$\frac{w_i}{\sum_j w_j}$$

- minimum bandwidth guarantee (per-traffic-class)



COMPUTER NETWORKS

Ashwini M Joshi

Department of Computer Science and Engineering

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

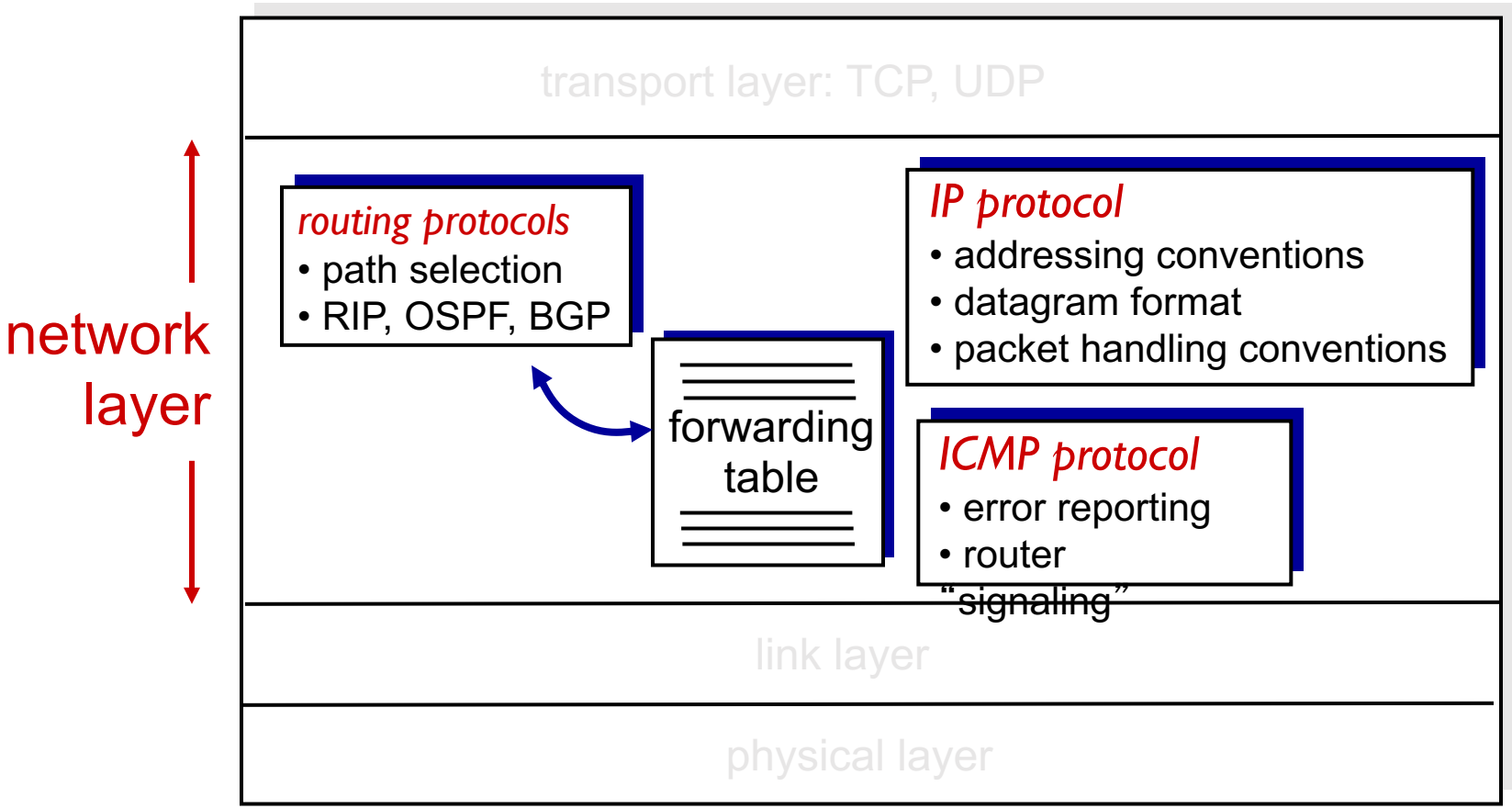
4.6 routing in the Internet

- RIP
- OSPF
- BGP

4.7 broadcast and multicast routing

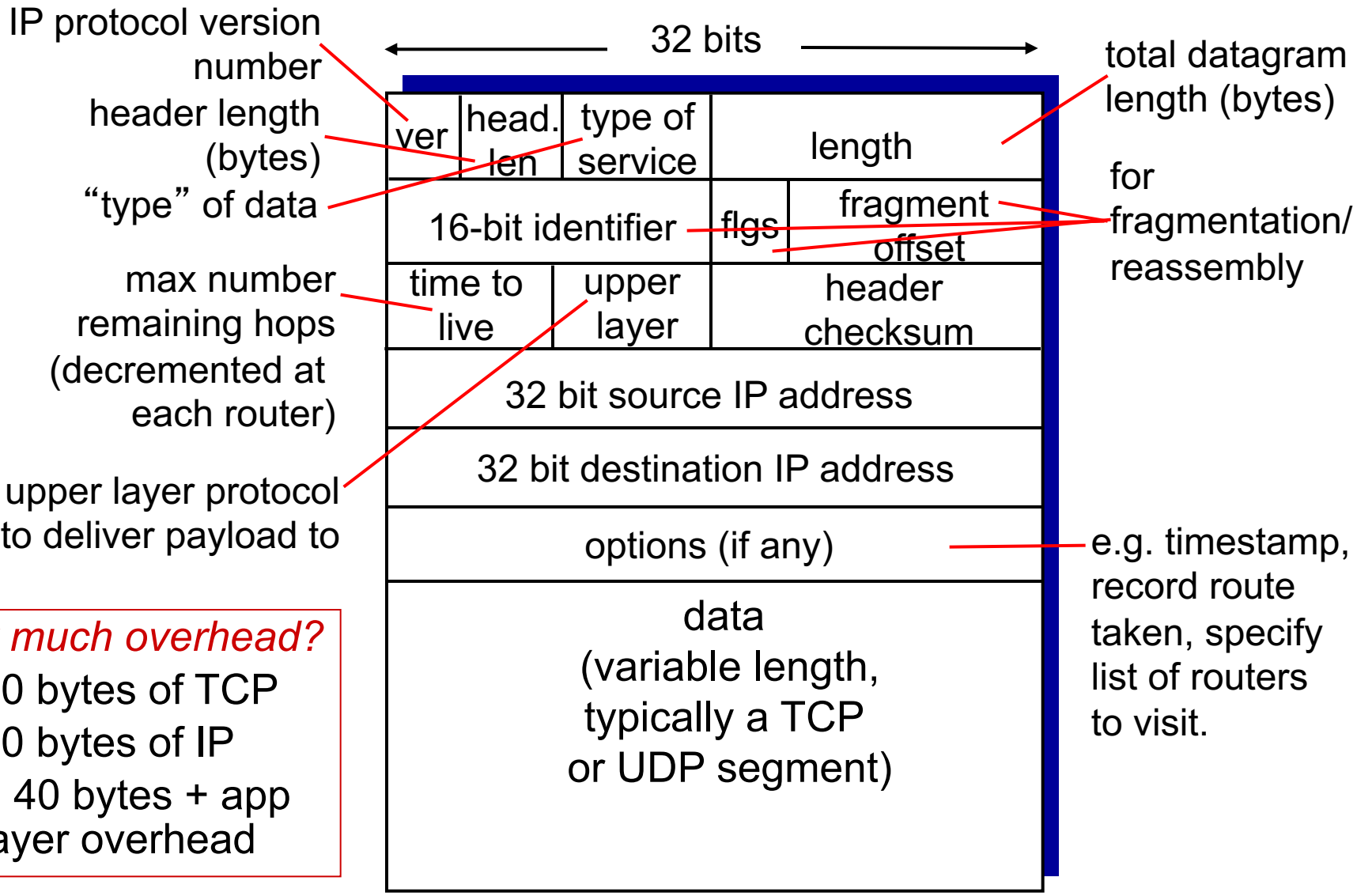
The Internet network layer

host, router network layer functions:



COMPUTER NETWORKS

IP datagram format

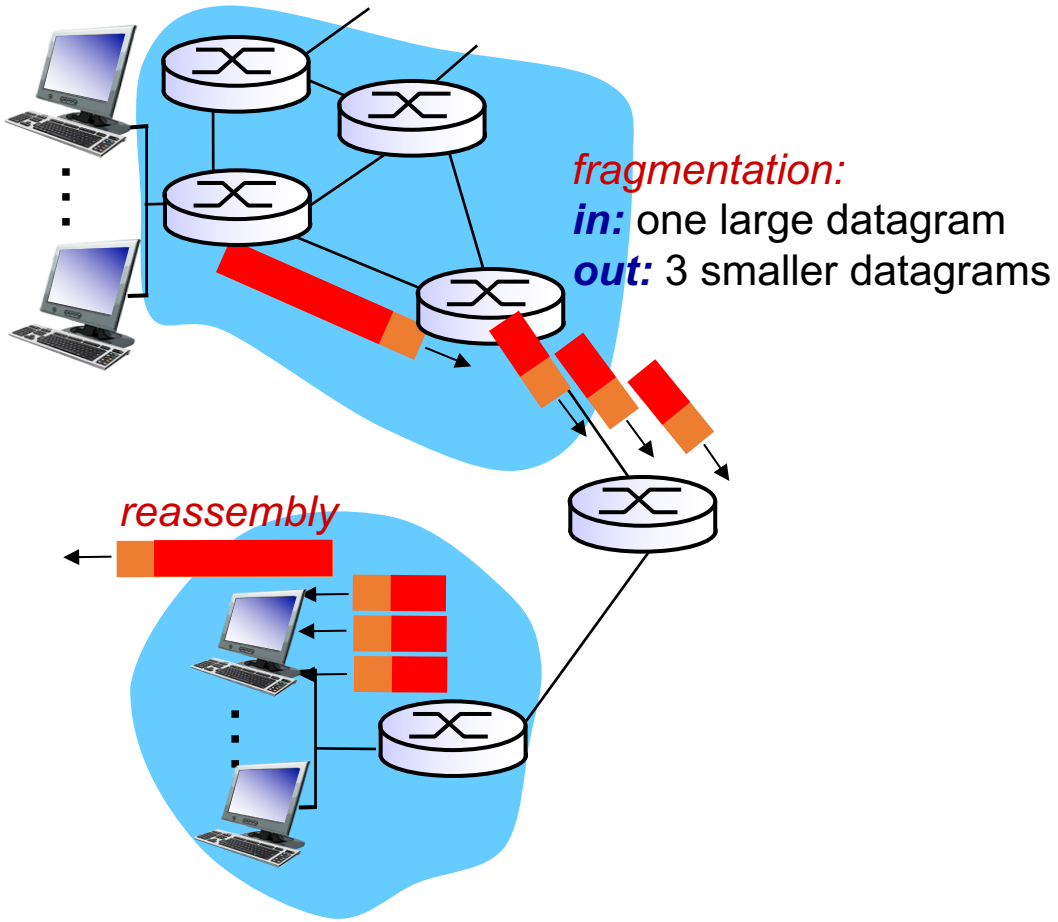


how much overhead?

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

IP fragmentation, reassembly

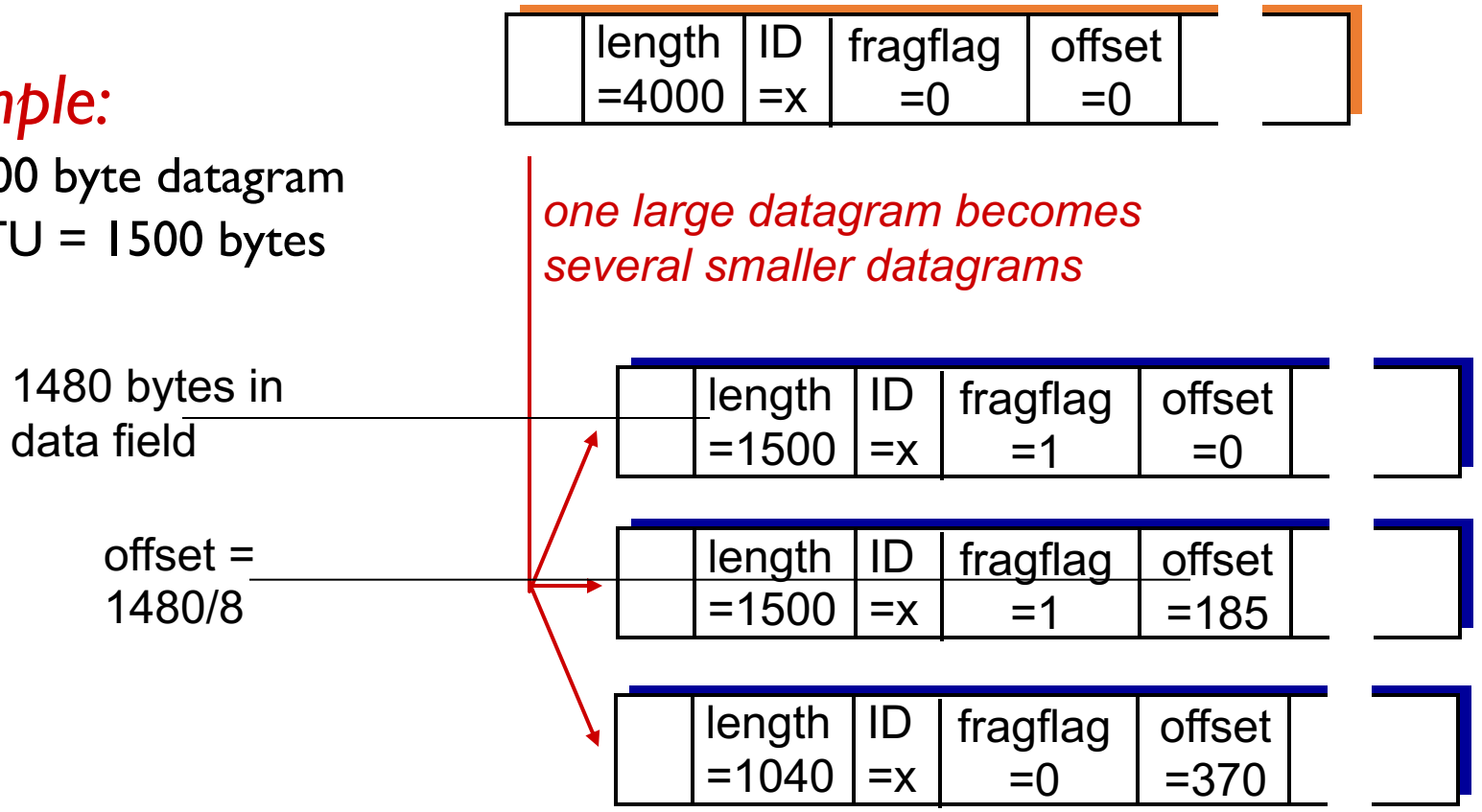
- network links have MTU (max.transfer size) - largest possible link-level frame
  - different link types, different MTUs
- large IP datagram divided ("fragmented") within net
  - one datagram becomes several datagrams
  - "reassembled" only at final destination
  - IP header bits used to identify, order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



IP fragmentation, reassembly

example:

- 4000 byte datagram
- MTU = 1500 bytes



Practise Problems

- Datagram size: 3000 Bytes
- MTU: 500 Bytes
- Header Size: 20 Bytes

Practise Problems

- Datagram size: 520 Bytes
- MTU: 200 Bytes
- Header Size: 20 Bytes

COMPUTER NETWORKS

Ashwini M Joshi

Department of Computer Science and Engineering

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

COMPUTER NETWORKS

IP addressing: Introduction

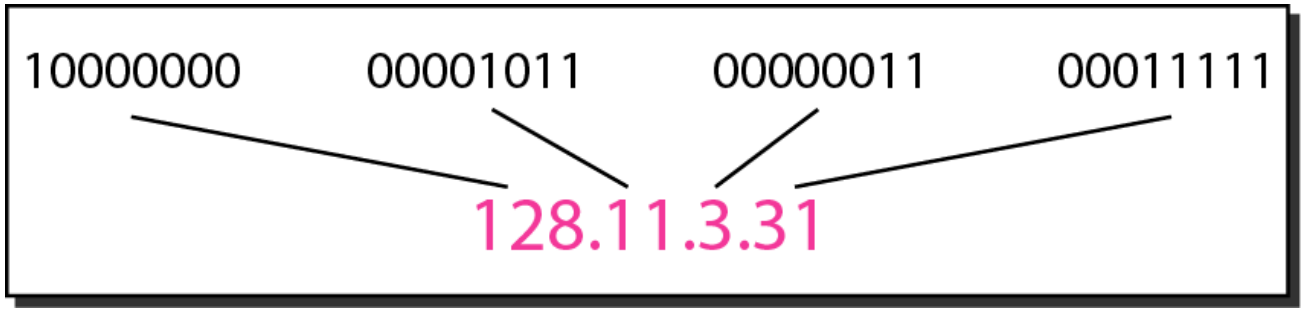
Topics discussed in this section:

Address Space
Notations
Classful/classless Addressing

An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a device (for example, a computer or a router) to the Internet.

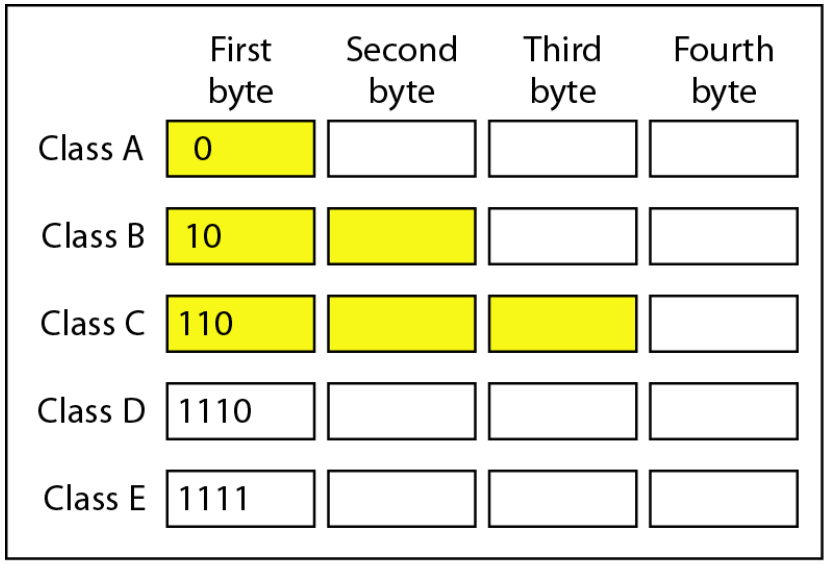
The address space of IPv4 is : 2^{32} or 4,294,967,296.

Dotted-decimal notation and binary notation for an IPv4 address

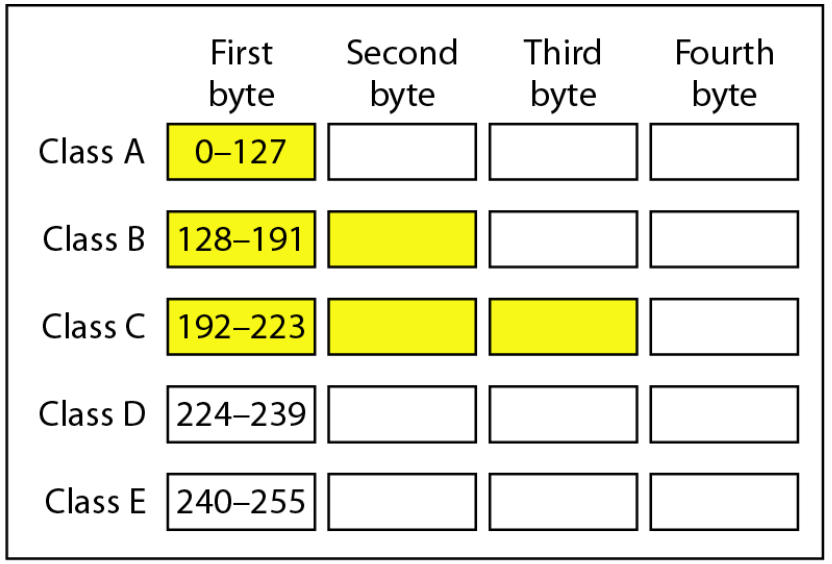


IP addressing: Introduction

In classful addressing, the address space is divided into five classes: A, B, C, D, and E.



a. Binary notation



b. Dotted-decimal notation

Class D: multicast
Class E: reserved

IP addressing: Introduction

The binary, decimal, and hexadecimal conversion table illustrates the three numbering systems.

Binary	Decimal	Hexadecimal
0 0 0 0	0	0
0 0 0 1	1	1
0 0 1 0	2	2
0 0 1 1	3	3
0 1 0 0	4	4
0 1 0 1	5	5
0 1 1 0	6	6
0 1 1 1	7	7
1 0 0 0	8	8
1 0 0 1	9	9
1 0 1 0	10	A
1 0 1 1	11	B
1 1 0 0	12	C
1 1 0 1	13	D
1 1 1 0	14	E
1 1 1 1	15	F

Change the following IPv4 addresses from binary notation to dotted-decimal notation.

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

Solution

We replace each group of 8 bits with its equivalent decimal number (see Appendix B) and add dots for separation.

a. 129.11.11.239

b. 193.131.27.255

Change the following IPv4 addresses from dotted-decimal notation to binary notation.

a. 111.56.45.78

b. 221.34.7.82

Solution

We replace each decimal number with its binary equivalent (see Appendix B).

a. 01101111 00111000 00101101 01001110

b. 11011101 00100010 00000111 01010010

Find the error, if any, in the following IPv4 addresses.

a. 111.56.045.78
b. 221.34.7.8.20
c. 75.45.301.14
d. 11100010.23.14.67

Solution

a. There must be no leading zero (045).
b. There can be no more than four numbers.
c. Each number needs to be less than or equal to 255.
d. A mixture of binary notation and dotted-decimal notation is not allowed.

Find the class of each address.

a. 00000001 00001011 00001011 11101111
b. 11000001 10000011 00011011 11111111
c. 14.23.120.8
d. 252.5.15.111

Solution

a. The first bit is 0. This is a class A address.
b. The first 2 bits are 1; the third bit is 0. This is a class C address.
c. The first byte is 14; the class is A.
d. The first byte is 252; the class is E.

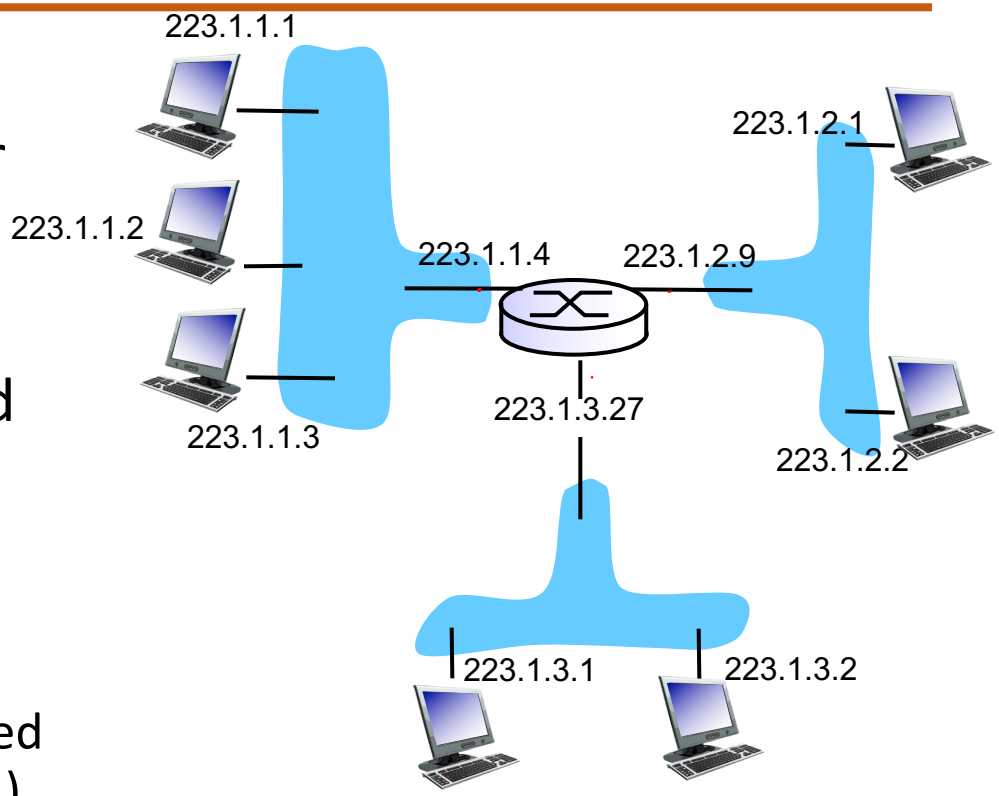
IP addressing: introduction

In classful addressing, a large part of the available addresses were wasted.

Classful addressing, which is almost obsolete, is replaced with classless addressing.

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 active interface (e.g., wired Ethernet, wireless 802.11)



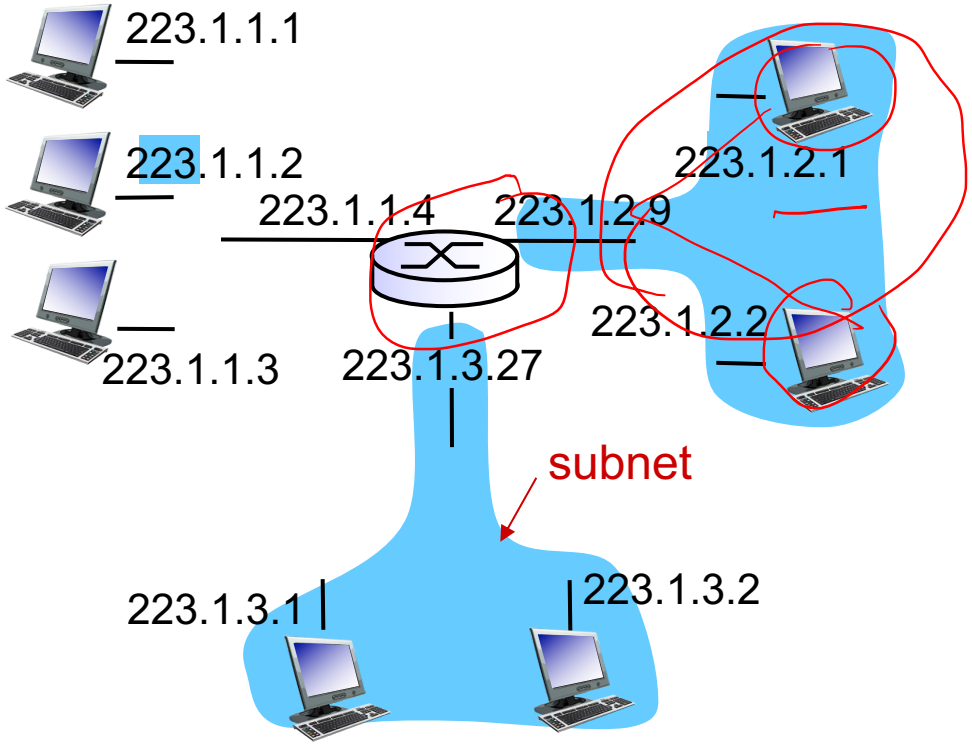
- *one IP address associated with each interface*

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 of IP address
  - can physically reach each other *without intervening router*

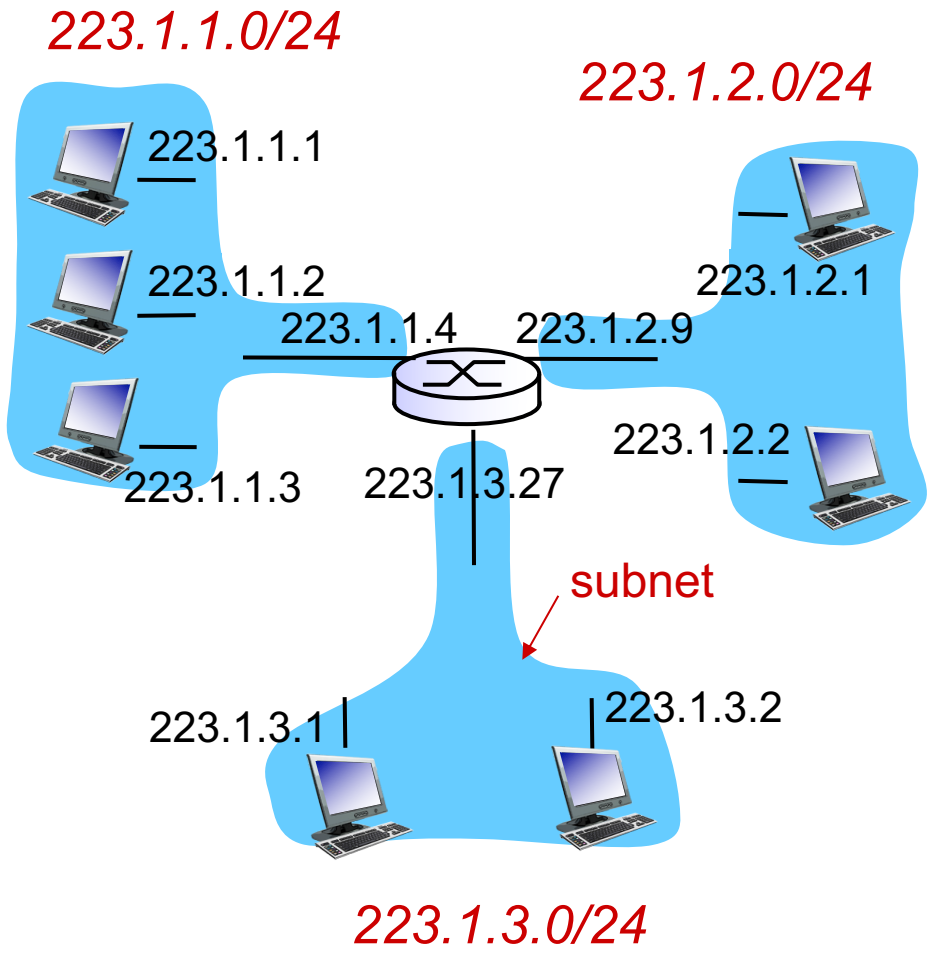


network consisting of 3 subnets

Subnets

recipe

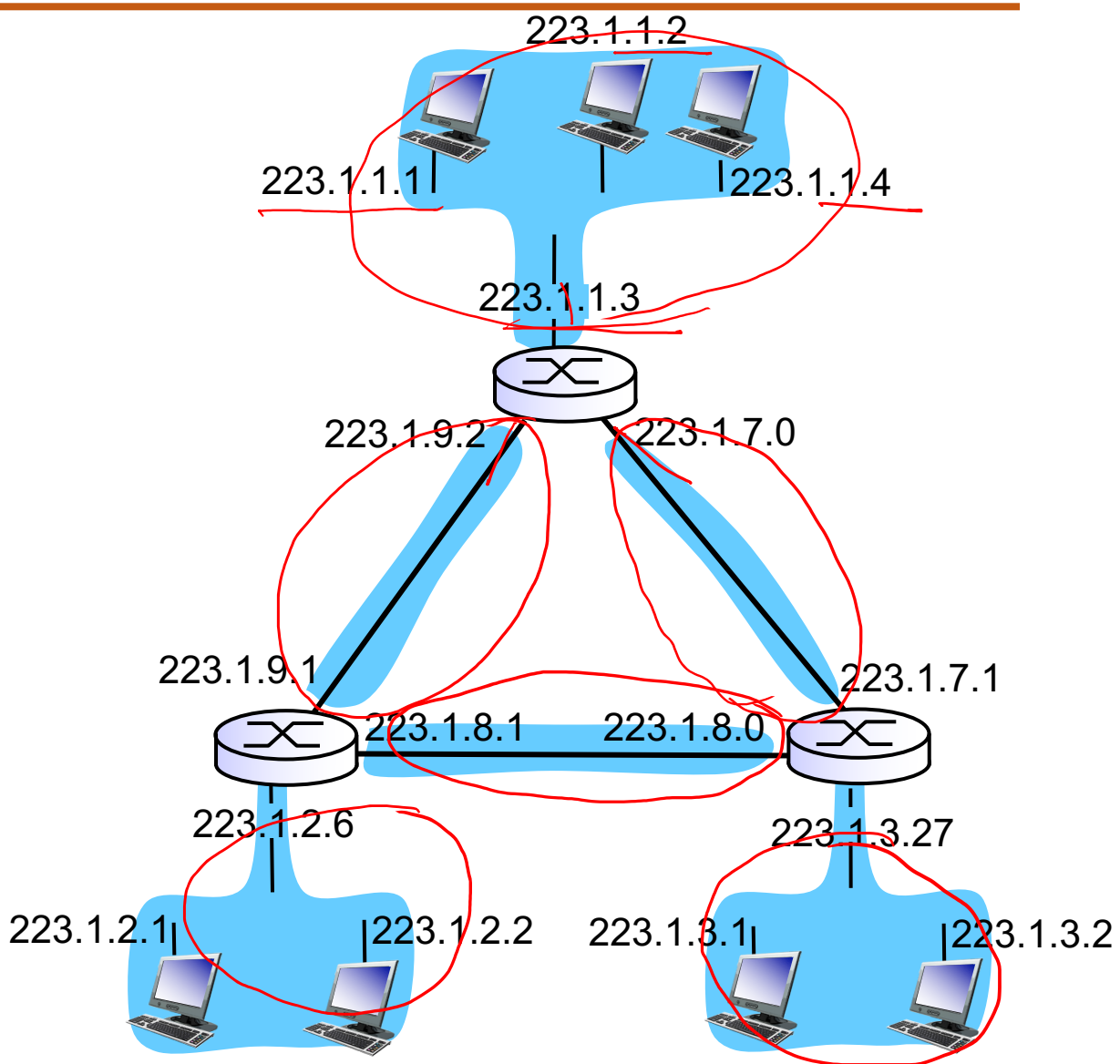
- to determine the subnets, detach each interface from its host or router, creating islands of isolated networks
- each isolated network is called a *subnet*



subnet mask: /24

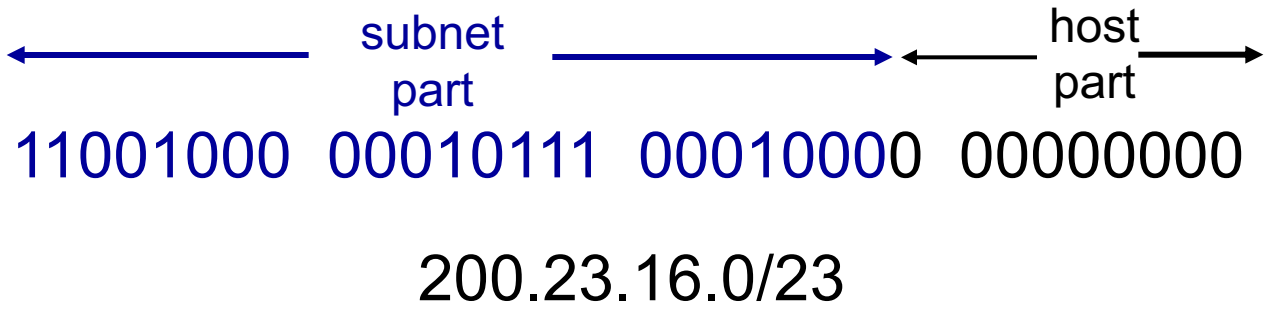
Subnets

how many?



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



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

- An organization is granted the block 214.17.160.0/24. The administrator wants to create 8 subnets.
- a. Find the subnet mask.
- b. Find the number of addresses in each subnet.
- c. Find the last addresses in first subnet.
- d. Find the first addresses in last subnet.
-

- **Solution:**
- **a) log2 8 =3**
- **Extra 1s =3 Mask: /27 (24 + 3)**
-
- **b.32– 27 = 5, 2^5= 32 addresses per subnet**
-
- **c. Subnet 1:**
- **the last address, we need to write 00011111 in last byte**
- **it is 214.17.160.00011111 = 214.17.160. 31**
-
- **d.**
- last subnet:
- 214.17.160.11100000 is the last byte
- So first address of last subnet is: 214.17.160.224

- In a block of addresses, we know the IP address of one host is 25.34.12.56/16.
- What are the first address (network address) and the last address (limited broadcast address) in this block?

- **Solution:**
- With the information given, the first address is found by ANDing the host address
- with the mask 255.255.0.0 (/16).

- Host Address: 25.34.12.56
- Mask (ANDed):255.255.0.0

- Network Address (First):25.34.0.0

- The last address can be found by ORing the host address with the mask complement 0.0.255.255.

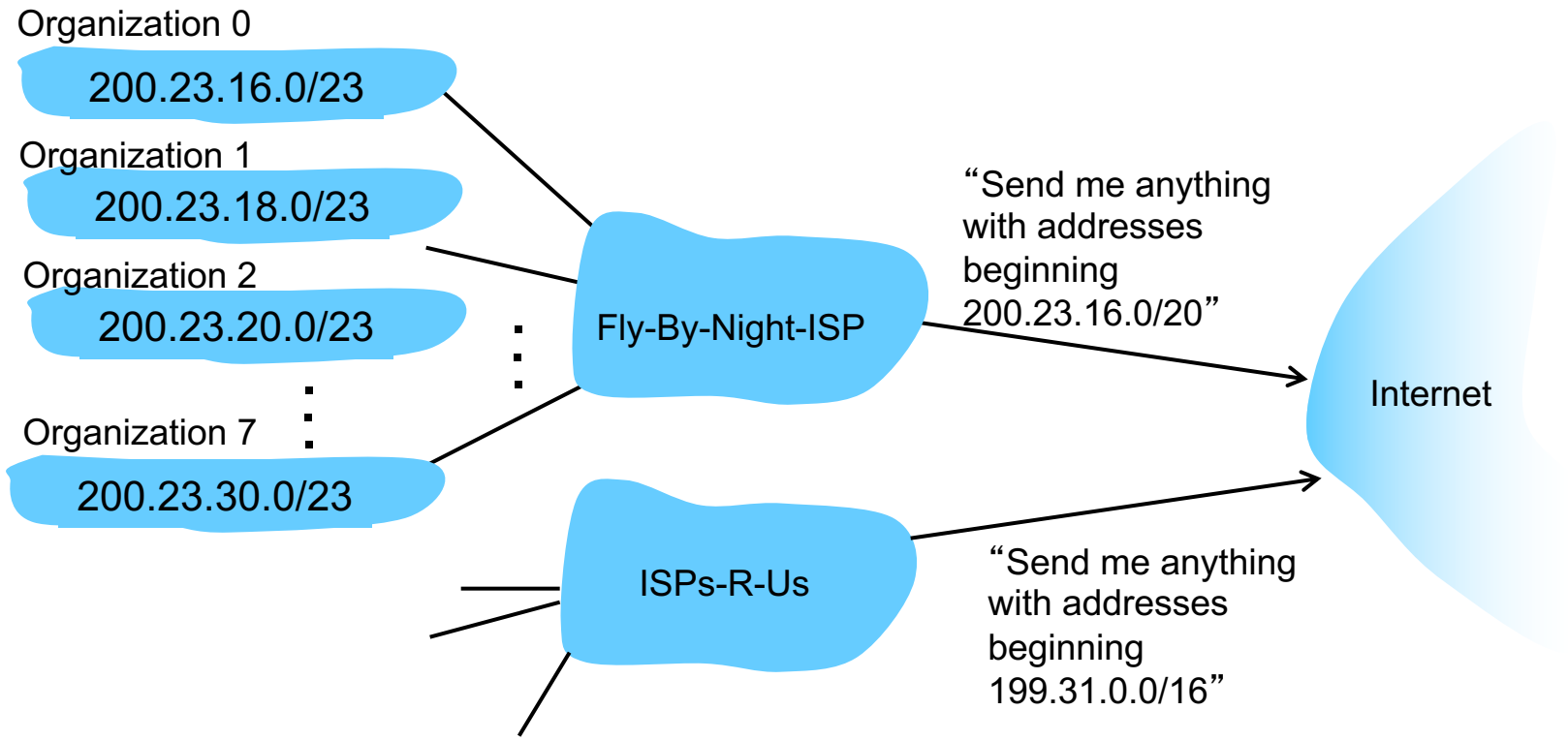
- Host Address:25.34.12.56
- Mask Complement (Ored):0.0.255.255
- Last Address:25.34.255.255

- Find the netid and the hostid of the following IP addresses.
- a. 114.34.2.8
- b. 132.56.8.6

Solution:
netid: 114 hostid: 34.2.8
netid: 132.56 hostid: 8.6

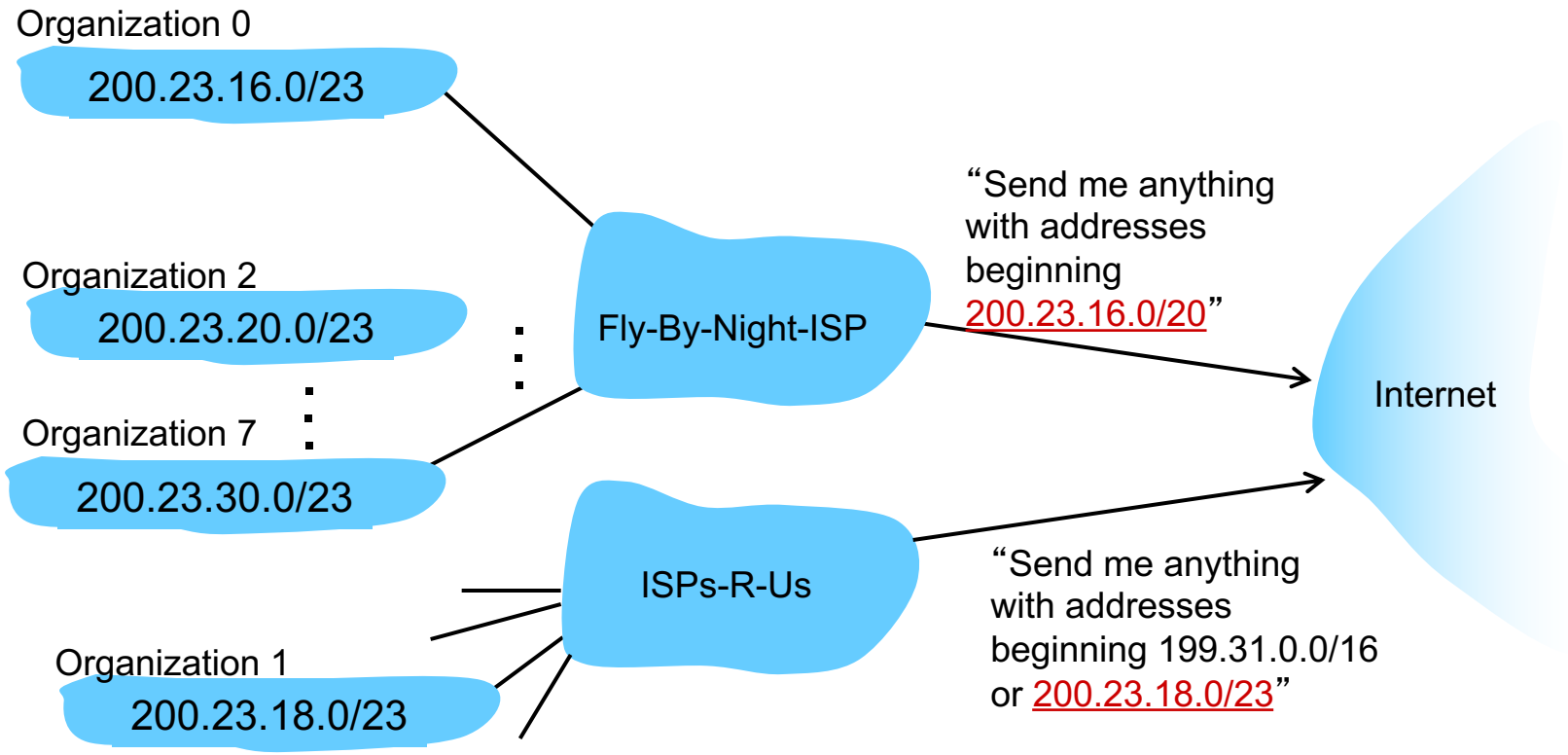
Hierarchical addressing: route aggregation

hierarchical addressing allows efficient advertisement of routing information:



Hierarchical addressing: more specific routes

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



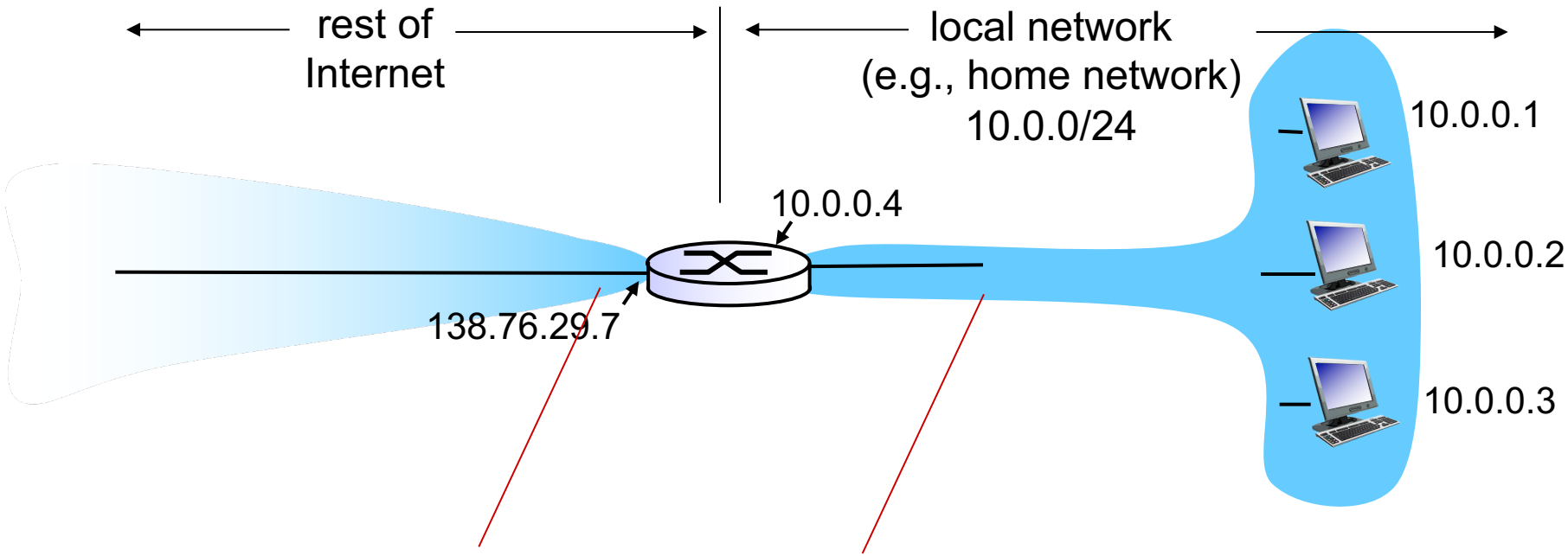
IP addressing: how to get a block?

Q: how does an ISP get block of addresses?

A: ICANN: Internet Corporation for Assigned Names and Numbers http://www.icann.org/

- 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:

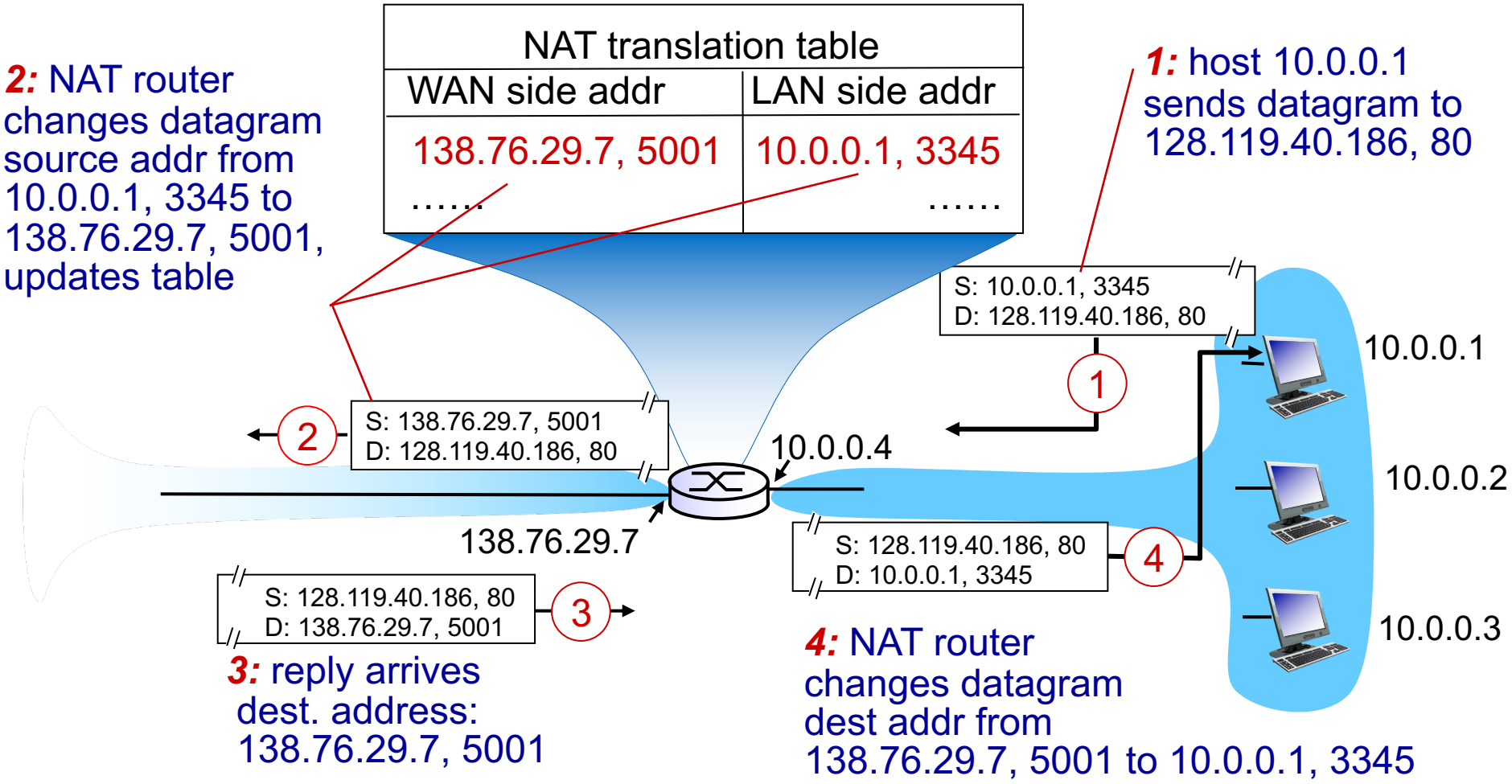
- range of addresses not needed from ISP: just one IP address for all devices
- can change addresses of devices in local network without notifying outside world
- can change ISP without changing addresses of devices in local network
- devices inside local net not explicitly addressable, visible by outside 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: 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

COMPUTER NETWORKS

Ashwini M Joshi

Department of Computer Science and Engineering

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

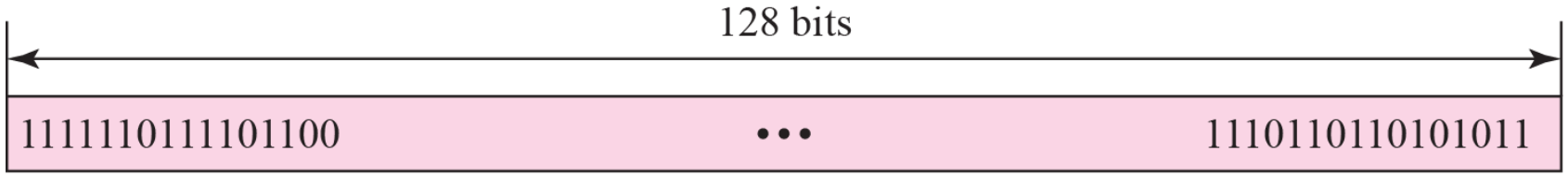
4.7 broadcast and multicast routing

COMPUTER NETWORKS

IPV6 Objectives

- To introduce the IPv6 addressing scheme and different notations used to represent an address in this version.
- To explain the three types of addressing used in IPv6: unicast, anycast, and multicast.
- To show the address space in this version and how it is divided into several blocks.
- To discuss some reserved blocks in the address space and their applications.
- To define the global unicast address block and how it is used for unicast communication.
- To discuss how three levels of hierarchy in addressing are used in IPv6 deploying the global unicast block.
- To discuss autoconfiguration and renumbering of IPv6 addresses.

An IPv6 address is 128 bits or 16 bytes (octet) long as shown in Figure. The address length in IPv6 is four times of the length address in IPv4.



Colon hexadecimal notation

FDEC : BA98 : 7654 : 3210 : ADBF : BBFF : 2922 : FFFF

Zero compression

FDEC : 0 : 0 : 0 : 0 : BBFF : 0 : FFFF → FDEC :: BBFF : 0 : FFFF

Original address | Zero compressed

CIDR address

FDEC :: BBFF : 0 : FFFF/60

Examples

Show the unabbreviated colon hex notation for the following IPv6 addresses:

a. An address with 64 0s followed by 64 1s.
b. An address with 128 0s.
c. An address with 128 1s.
d. An address with 128 alternative 1s and 0s.

Solution

a. 0000:0000:0000:0000:FFFF:FFFF:FFFF:FFFF
b. 0000:0000:0000:0000:0000:0000:0000:0000
c. FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF
d. AAAA:AAAA:AAAA:AAAA:AAAA:AAAA:AAAA:AAAA

Examples

The following shows the zero contraction version of addresses in Example 26.1 (part c and d cannot be abbreviated)

a. :: FFFF:FFFF:FFFF:FFFF
b. ::
c. FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF
d. AAAA:AAAA:AAAA:AAAA:AAAA:AAAA:AAAA:AAAA

Examples

Show abbreviations for the following addresses:

a. 0000:0000:FFFF:0000:0000:0000:0000:0000

b. 1234:2346:0000:0000:0000:0000:0000:1111

c. 0000:0001:0000:0000:0000:0000:1200:1000

d. 0000:0000:0000:0000:0000:FFFF:24.123.12.6

Solution

a. 0:0:FFFF::

b. 1234:2346::1111

c. 0:1::1200:1000

d. ::FFFF:24.123.12.6

Examples

Decompress the following addresses and show the complete unabbreviated IPv6 address:

a. 1111::2222
b. ::
c. 0:1::
d. AAAA:A:AA::1234

Solution

a. 1111:0000:0000:0000:0000:0000:0000:2222
b. 0000:0000:0000:0000:0000:0000:0000:0000
c. 0000:0001:0000:0000:0000:0000:0000:0000
d. AAAA:000A:00AA:0000:0000:0000:0000:1234

Address space allocation

Like the address space of IPv4, the address space of IPv6 is divided into several blocks of varying size and each block is allocated for special purpose.

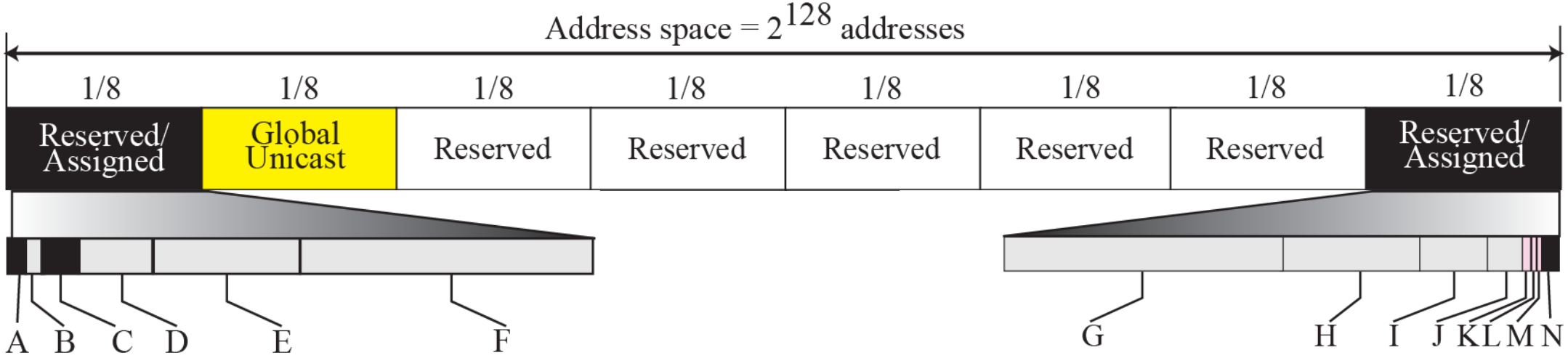
Most of the blocks are still unassigned and have been left aside for future use.

To better understand the allocation and the location of each block in address space, we first divide the whole address space into eight equal ranges.

This division does not show the block allocation, but we believe it shows where each actual block is located (Shown in Figure)

COMPUTER NETWORKS

Address space allocation



A: 1/256 IPv4 Compatible
B: 1/256 Reserverd
C: 1/128 Reserved
D: 1/64 Reserved
E: 1/32 Reserved
F: 1/16 Reserved
G: 1/16 Reserved
H: 1/32 Reserved
I: 1/64 Reserved
J: 1/128 Unique Local Unicast
K: 1/512 Reserved
L: 1/1024 Link Local
M: 1/256 Reserved
N: 1/256 Multicast

Address space allocation

	Block Prefix	CIDR	Block Assignment	Fraction
1	0000 0000	0000::/8	Reserved (IPv4 compatible)	1/256
	0000 0001	0100::/8	Reserved	1/256
	0000 001	0200::/7	Reserved	1/128
	0000 01	0400::/6	Reserved	1/64
	0000 1	0800::/5	Reserved	1/32
	0001	1000::/4	Reserved	1/16
2	**001**	**2000::/3**	**Global unicast**	**1/8**
3	010	4000::/3	Reserved	1/8
4	011	6000::/3	Reserved	1/8
5	100	8000::/3	Reserved	1/8
6	101	A000::/3	Reserved	1/8
7	110	C000::/3	Reserved	1/8
8	1110	E000::/4	Reserved	1/16
	1111 0	F000::/5	Reserved	1/32
	1111 10	F800::/6	Reserved	1/64
	1111 110	FC00::/7	Unique local unicast	1/128
	1111 1110 0	FE00::/9	Reserved	1/512
	1111 1110 10	FE80::/10	Link local addresses	1/1024
	1111 1110 11	FEC0::/10	Reserved	1/1024
	1111 1111	FF00::/8	Multicast addresses	1/256

Figure shows that only a portion of the address space can be used for global unicast communication. How many addresses are in this block?

Solution

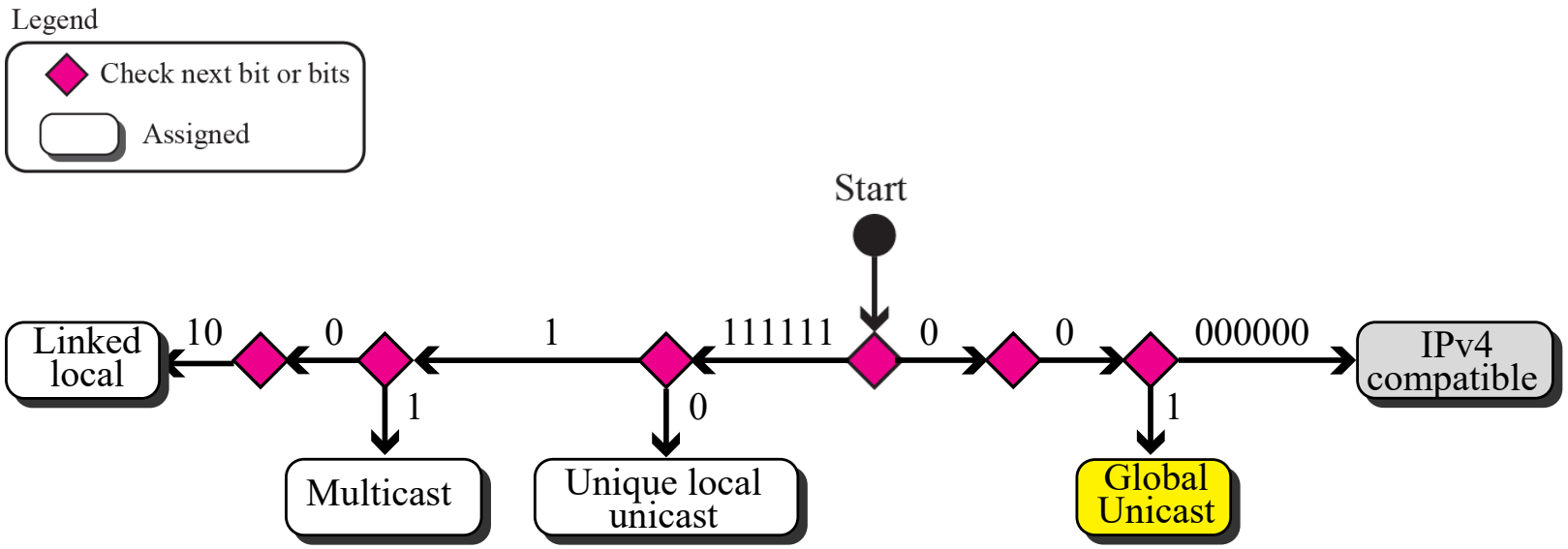
This block occupies only one-eighth of the address spaces. To find the number of addresses, we can divide the total address space by 8 or 2^3 . The result is $(2^{128})/(2^3) = 2^{125}$ —a huge block.

Address space allocation

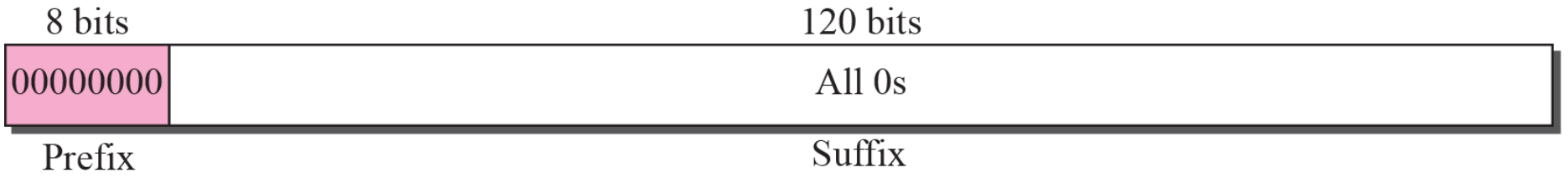
Figure shows that only a portion of the address space can be used for global unicast communication.
How many addresses are in this block?

This block occupies only one-eighth of the address spaces. To find the number of addresses, we can divide the total address space by 8 or 2^3 . The result is $(2^{128})/(2^3) = 2^{125}$ —a huge block.

Algorithm for finding the allocated blocks

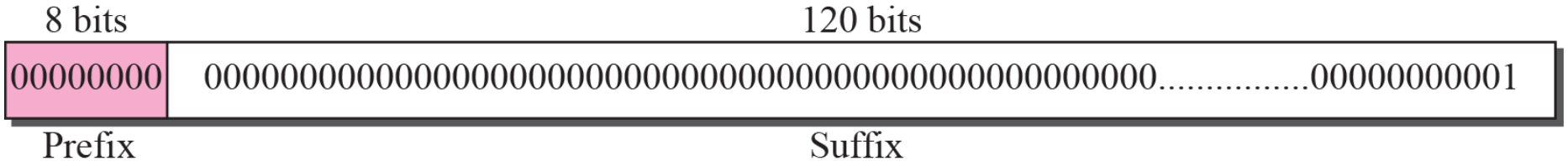


Unspecified address and Loopback address



The unspecified address in IPv6 is ::/128. It should never be used as a destination address.

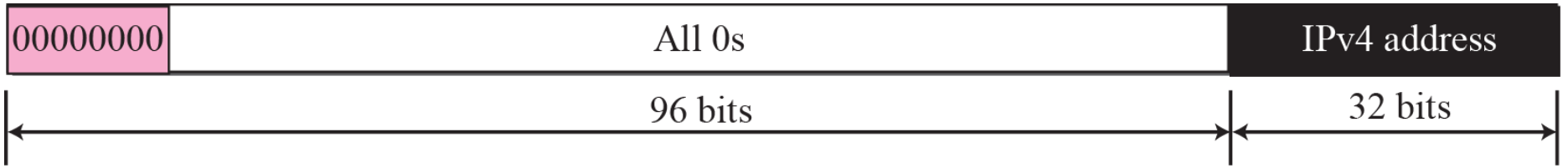
Loopback address



The loopback address in IPv6 is ::1/128. It should never be used as a destination address.

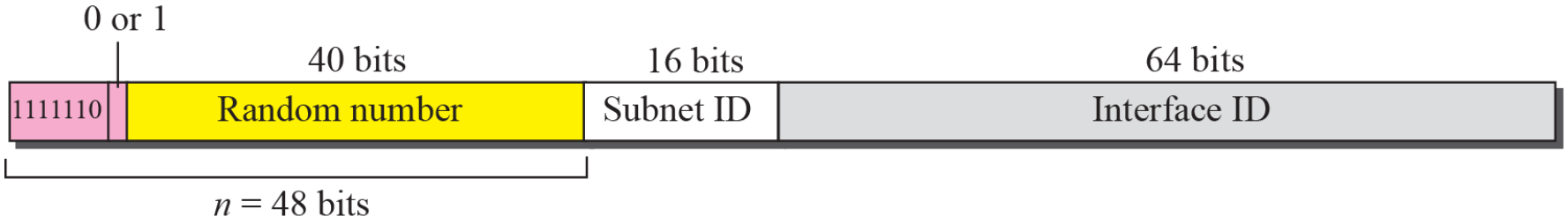
Compatible address, Mapped address and Unique local unicast address

Compatible address



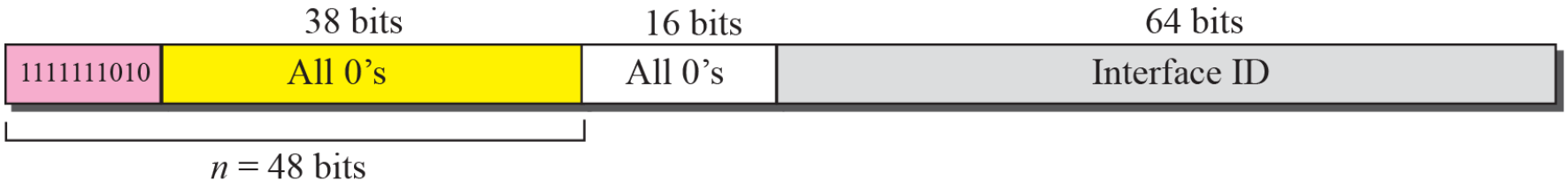
Mapped address

Unique local unicast address

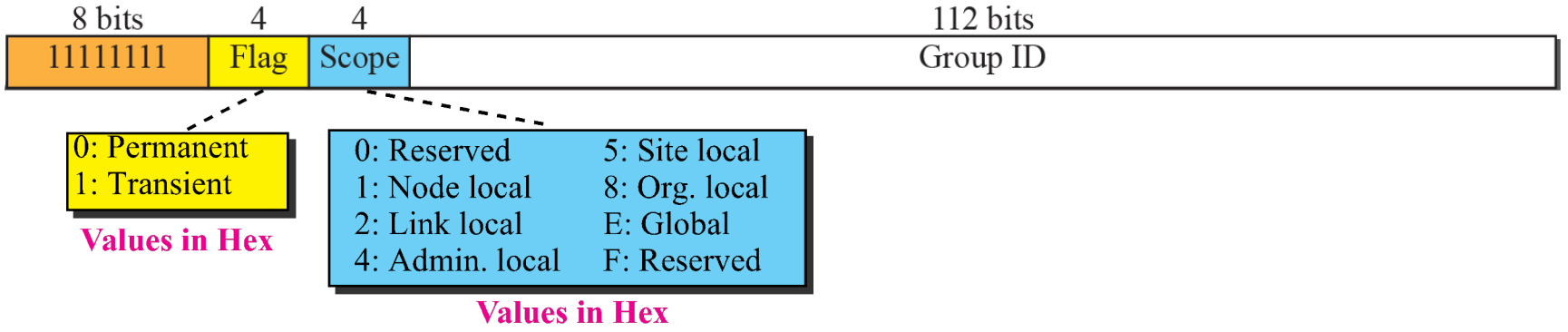


Link local address and Multicast address

Compatible address



Mapped address

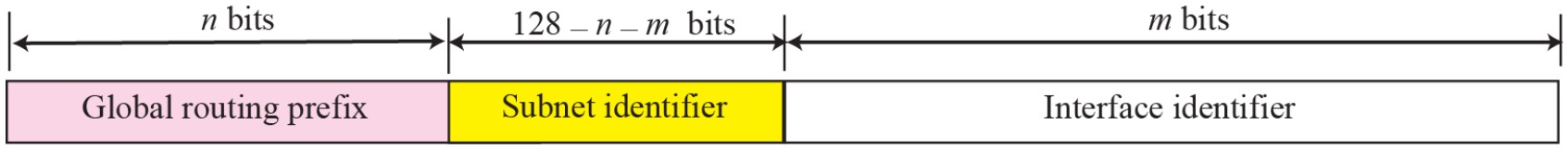


Global Unicast Addresses

This block in the address space that is used for unicast (one-to-one) communication between two hosts in the Internet is called global unicast address block.

CIDR notation for the block is 2000::/3, which means that the three leftmost bits are the same for all addresses in this block (001).

The size of this block is 2125 bits, which is more than enough for the Internet expansion in the many years to come

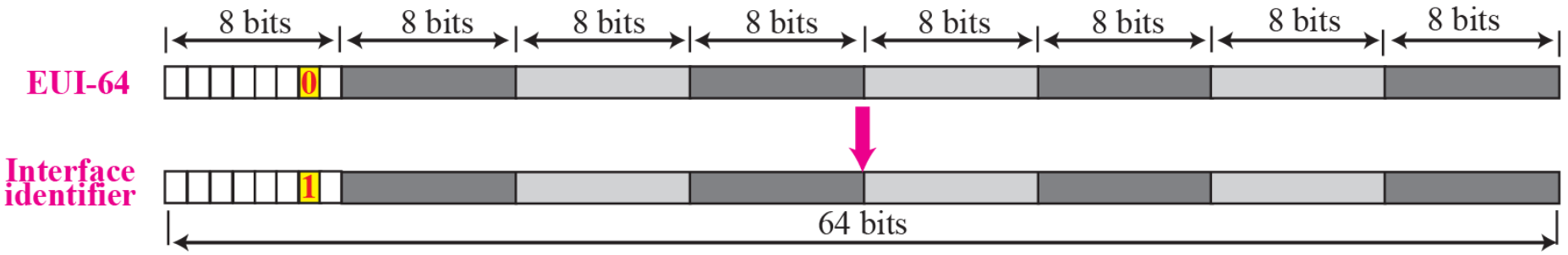


Global Unicast Addresses

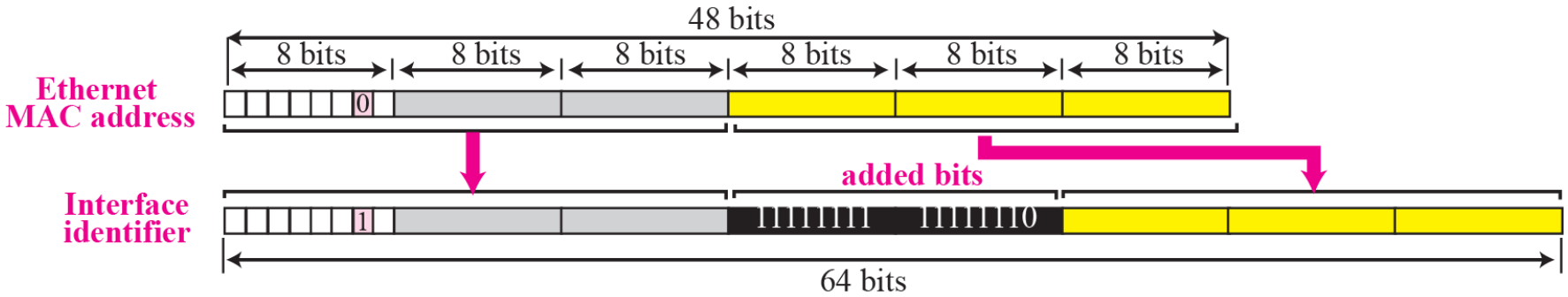
Recommended Lengths of Different Parts in Unicast Addressing

Block Assignment	Length
Global Routing Prefix (n)	48 Bits
Subnet Identifier (128—n—m)	16 Bits
Interface Identifier (m)	64 Bits

Mapping for EUI-64



Mapping for Ethernet MAC



Examples

1. Find the interface identifier if the physical address in the EUI is $(F5\text{-}A9\text{-}23\text{-}EF\text{-}07\text{-}14\text{-}7A\text{-}D2)_{16}$ using the format we defined for Ethernet addresses.

Solution

We only need to change the seventh bit of the first octet from 0 to 1 and change the format to colon hex notation. The result is F7A9:23EF:0714:7AD2.

2. Find the interface identifier if the Ethernet physical address is $(F5\text{-}A9\text{-}23\text{-}14\text{-}7A\text{-}D2)_{16}$ using the format we defined for Ethernet addresses.

Solution

We only need to change the seventh bit of the first octet from 0 to 1, insert two octet FFFE16 and change the format to colon hex notation. The result is F7A9:23FF:FE14:7AD2 in colon hex.

Examples

An organization is assigned the block 2000:1456:2474/48. What is the CIDR notation for the blocks in the first and second subnets in this organization.

Solution

Theoretically, the first and second subnets should use the block with subnet identifier 0001_{16} and 0002_{16}. This means that the blocks are

2000:1456:2474:0000/64

and

2000:1456:2474:0001/64.

An organization is assigned the block 2000:1456:2474/48. What is the IPv6 address of an interface in the third subnet if the IEEE physical address of the computer is $(F5\text{-}A9\text{-}23\text{-}14\text{-}7A\text{-}D2)_{16}$.

Solution

The interface identifier is F7A9:23FF:FE14:7AD2 (see Example 26.12). If we add this identifier to the global prefix and the subnet identifier, we get:

```
2000:1456:2474:0003:F7A9:23FF:FE14:7AD2/128
```

Autoconfiguration

One of the interesting features of IPv6 addressing is the autoconfiguration of hosts.

As we discussed in IPv4, the host and routers are originally configured manually by the network manager.

However, the Dynamic Host Configuration Protocol, DHCP, can be used to allocate an IPv4 address to a host that joins the network.

In IPv6, DHCP protocol can still be used to allocate an IPv6 address to a host, but a host can also configure itself.

Examples

Assume a host with Ethernet address $(F5\text{-}A9\text{-}23\text{-}11\text{-}9B\text{-}E2)_{16}$ has joined the network. What would be its global unicast address if the global unicast prefix of the organization is 3A21:1216:2165 and the subnet identifier is A245:1232.

Solution
The host first creates its interface identifier as

F7A9:23FF:FE11:9BE2

using the Ethernet address read from its card. The host then creates its link-local address as

```
FE80::F7A9:23FF:FE11:9BE2
```

Examples.. Continued

Assuming that this address is unique, the host sends a router solicitation message and receives the router advertisement message that announces the combination of global unicast prefix and the subnet identifier as 3A21:1216:2165:A245:1232. The host then appends its interface identifier to this prefix to find and store its global unicast address as:

```
3A21:1216:2165:A245:1232:F7A9:23FF:FE11:9BE2
```

Renumbering

To allow sites to change the service provider, Renumbering of the address prefix (*n*) was built into IPv6 addressing.

As we discussed before, each site is given a prefix by the service provider to which it is connected.

If the site changes the provider, the address prefix needs to be changed.

A router to which the site is connected can advertise a new prefix and let the site use the old prefix for a short time before disabling it.

In other words, during the transition period, a site has two prefixes.

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:

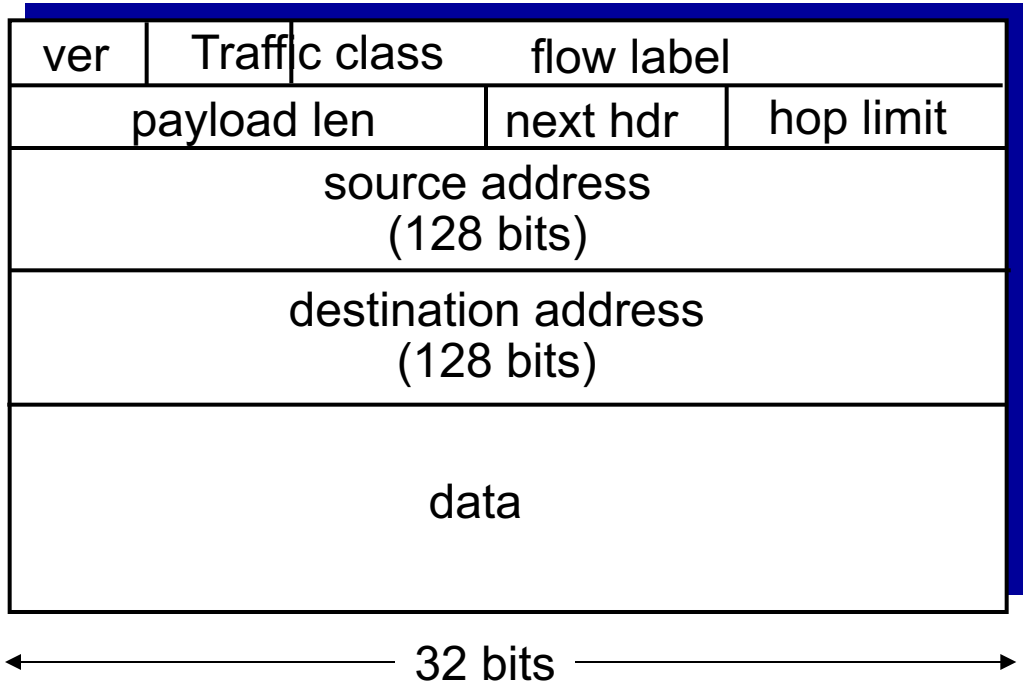
- fixed-length 40 byte header
- no fragmentation allowed

IPv6 datagram format

Traffic class: Similar to TOS

flow Label: identify datagrams in same "flow." (concept of"flow" not well defined).

next header: identify upper layer protocol for data

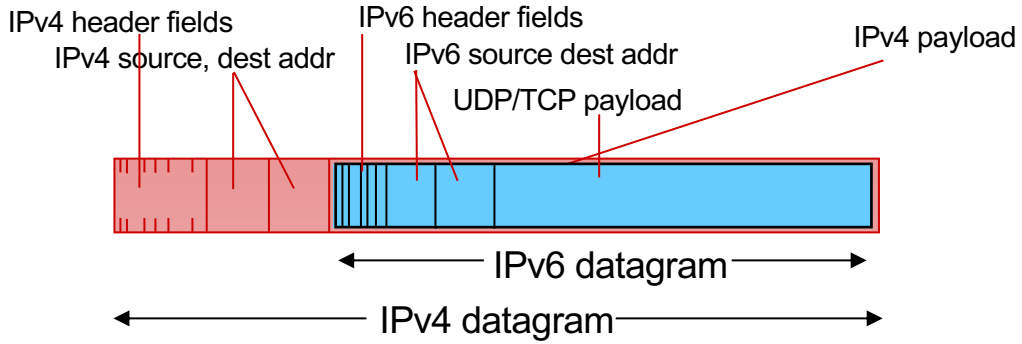


Other changes from IPv4

- *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

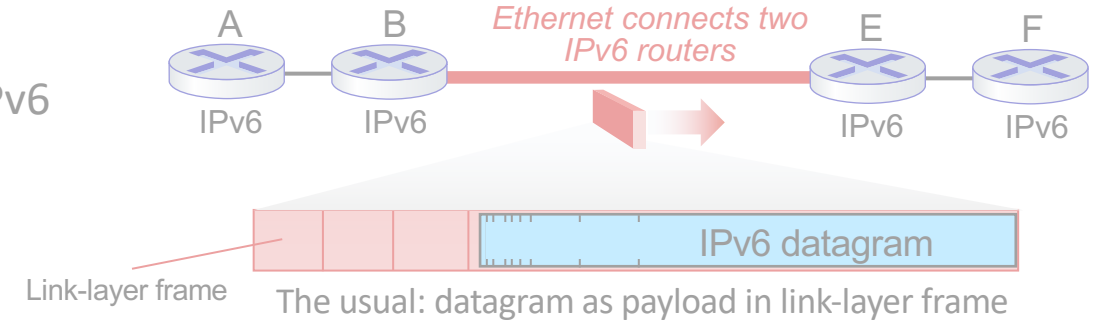
- not all routers can be upgraded simultaneously
  - no “flag days”
  - how will network operate with mixed IPv4 and IPv6 routers?
- tunneling: IPv6 datagram carried as *payload* in IPv4 datagram among IPv4 routers (“packet within a packet”)
  - tunneling used extensively in other contexts (4G/5G)



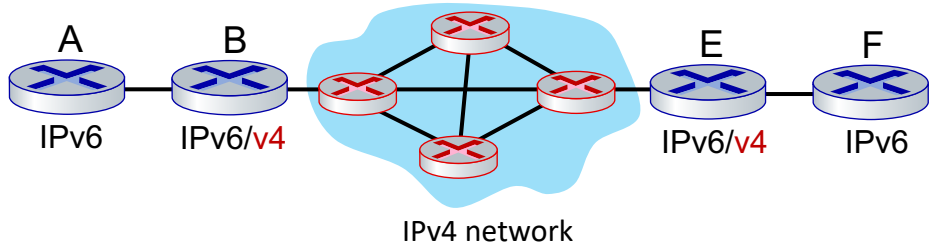
COMPUTER NETWORKS

Tunneling and encapsulation

Ethernet connecting two IPv6 routers:



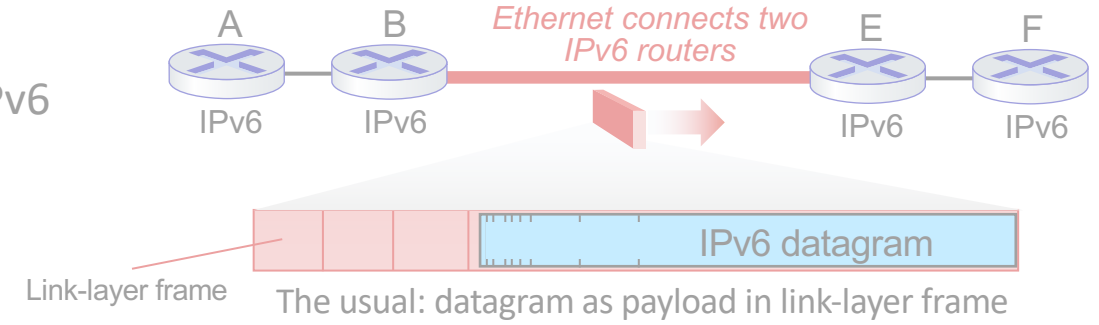
IPv4 network connecting two IPv6 routers



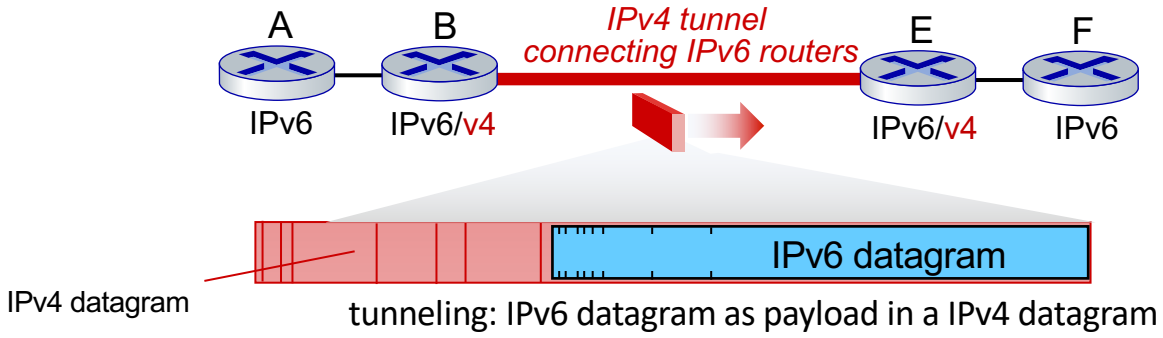
COMPUTER NETWORKS

Tunneling and encapsulation

Ethernet connecting two IPv6 routers:

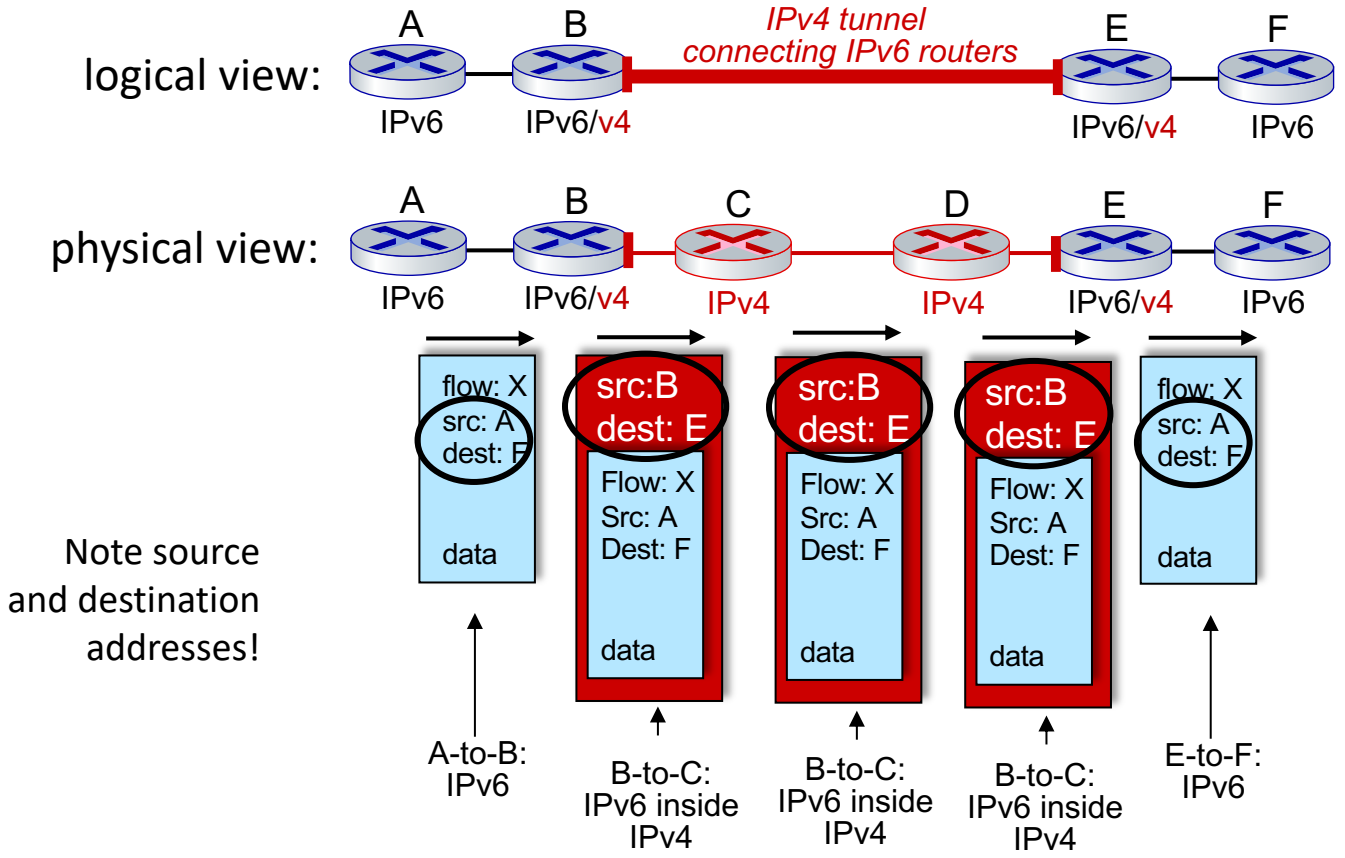


IPv4 tunnel connecting two IPv6 routers



COMPUTER NETWORKS

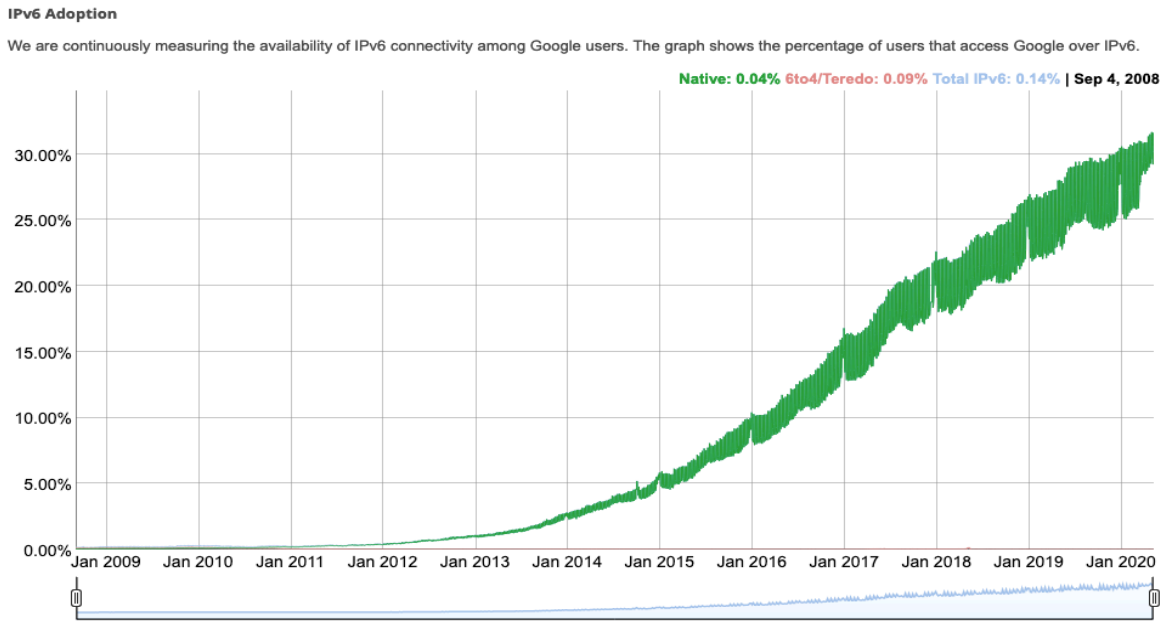
Tunneling



COMPUTER NETWORKS

IPv6: adoption

- Google[1]: ~ 30% of clients access services via IPv6
- NIST: 1/3 of all US government domains are IPv6 capable



1 https://www.google.com/intl/en/ipv6/statistics.html

COMPUTER NETWORKS

IPv6: adoption

- Google[1]: ~ 30% of clients access services via IPv6
- NIST: 1/3 of all US government domains are IPv6 capable
- Long (long!) time for deployment, use
  - 25 years and counting!
  - think of application-level changes in last 25 years: WWW, social media, streaming media, gaming, telepresence, …
  - *Why?*

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 as server
  - "plug-and-play"

COMPUTER NETWORKS

DHCP: Dynamic Host Configuration Protocol

goal: allow host to *dynamically* obtain its IP address from network server when it joins network

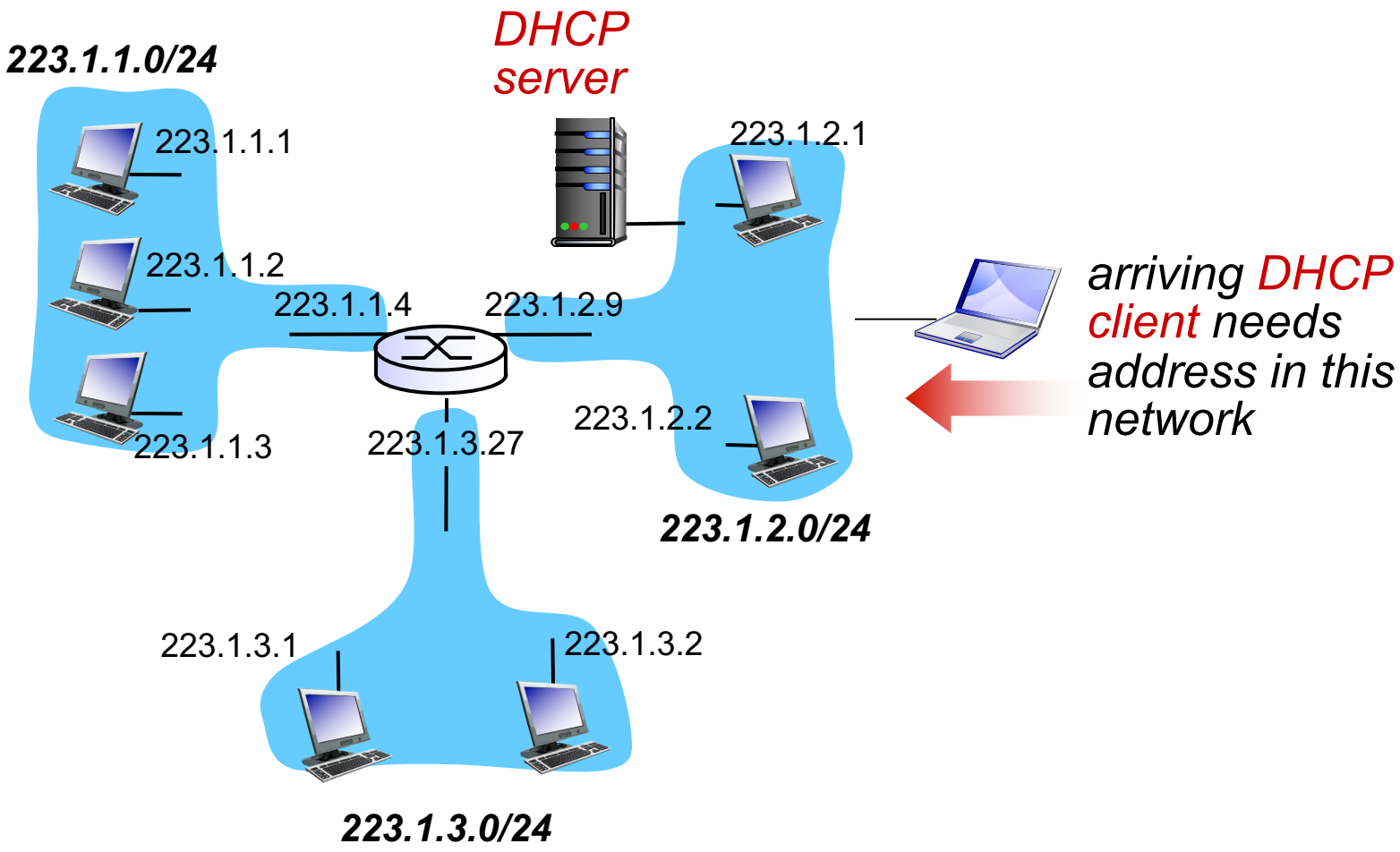
- can renew its lease on address in use
- allows reuse of addresses (only hold address while connected/"on")
- support for mobile users who want to join network (more shortly)

DHCP overview:

- host broadcasts "DHCP discover" msg [optional]
- DHCP server responds with "DHCP offer" msg [optional]
- host requests IP address: "DHCP request" msg
- DHCP server sends address: "DHCP ack" msg

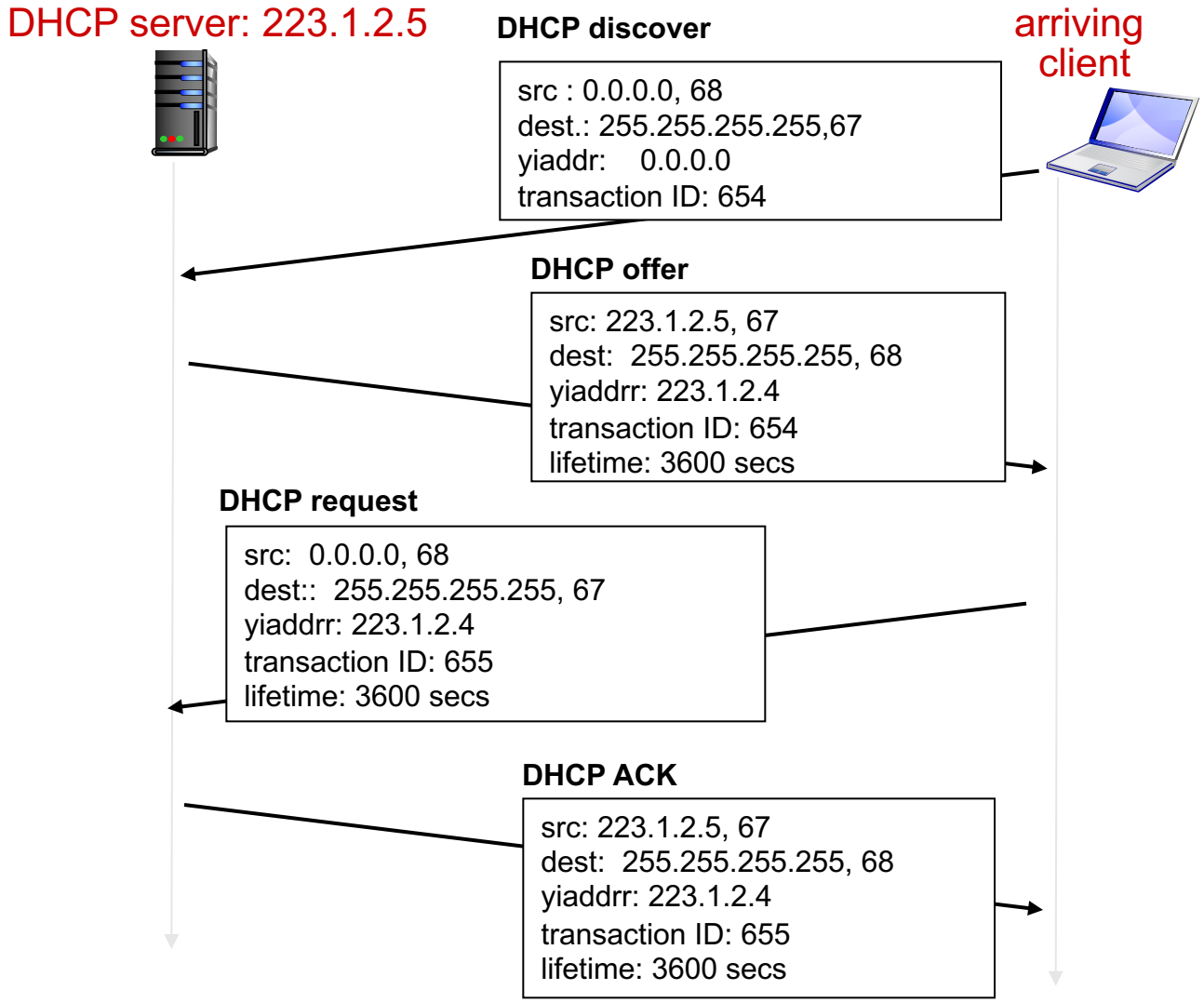
COMPUTER NETWORKS

DHCP client-server scenario



COMPUTER NETWORKS

DHCP client-server scenario

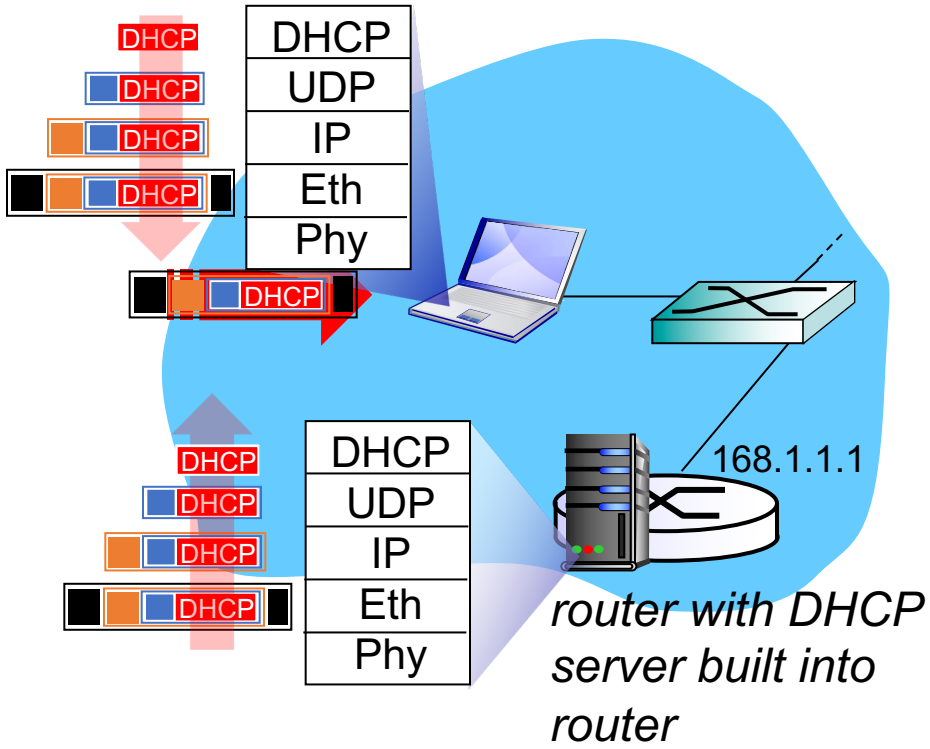


DHCP: more than IP addresses

DHCP returns:

- IP address
- address of first-hop router for client
- name and IP address of DNS sever
- network mask (indicating network versus host portion of address)

DHCP: example

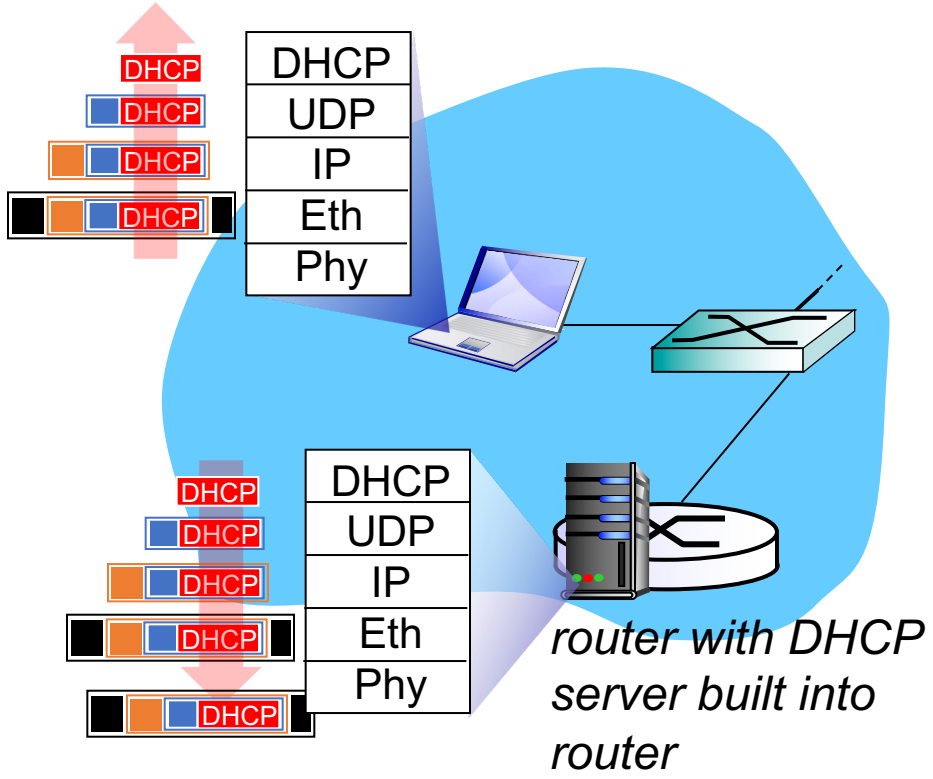


- connecting laptop needs its IP address, addr of first-hop router, addr of DNS server: use DHCP

- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

COMPUTER NETWORKS

DHCP: example



- DCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server

- encapsulation of DHCP server, frame forwarded to client, demuxing up to DHCP at client
- client now knows its IP address, name and IP address of DSN server, IP address of its first-hop router

COMPUTER NETWORKS

DHCP: Wireshark output (home LAN)

request

Message type: **Boot Request (1)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0 (0.0.0.0)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 0.0.0.0 (0.0.0.0)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) **DHCP Message Type = DHCP Request**
Option: (61) Client identifier
    Length: 7; Value: 010016D323688A;
    Hardware type: Ethernet
    Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Option: (t=50,l=4) Requested IP Address = 192.168.1.101
Option: (t=12,l=5) Host Name = "nomad"
Option: (55) Parameter Request List
    Length: 11; Value: 010F03062C2E2F1F21F92B
    1 = Subnet Mask; 15 = Domain Name
    3 = Router; 6 = Domain Name Server
    44 = NetBIOS over TCP/IP Name Server
    ……

reply

Message type: **Boot Reply (2)**
Hardware type: Ethernet
Hardware address length: 6
Hops: 0
Transaction ID: 0x6b3a11b7
Seconds elapsed: 0
Bootp flags: 0x0000 (Unicast)
Client IP address: 192.168.1.101 (192.168.1.101)
Your (client) IP address: 0.0.0.0 (0.0.0.0)
Next server IP address: 192.168.1.1 (192.168.1.1)
Relay agent IP address: 0.0.0.0 (0.0.0.0)
Client MAC address: Wistron_23:68:8a (00:16:d3:23:68:8a)
Server host name not given
Boot file name not given
Magic cookie: (OK)
Option: (t=53,l=1) DHCP Message Type = DHCP ACK
Option: (t=54,l=4) Server Identifier = 192.168.1.1
Option: (t=1,l=4) Subnet Mask = 255.255.255.0
Option: (t=3,l=4) Router = 192.168.1.1
Option: (6) Domain Name Server
    Length: 12; Value: 445747E2445749F244574092;
    IP Address: 68.87.71.226;
    IP Address: 68.87.73.242;
    IP Address: 68.87.64.146
Option: (t=15,l=20) Domain Name = "hsd1.ma.comcast.net."

COMPUTER NETWORKS

Ashwini M Joshi (Ph.D.)
Department of Computer Science and Engineering

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

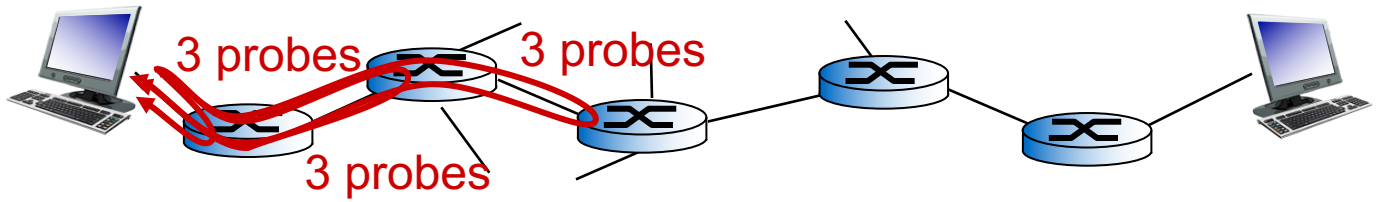
Type	Code	description
0	0	echo reply (ping)
3	0	dest. network unreachable
3	1	dest host unreachable
3	2	dest protocol unreachable
3	3	dest port unreachable
3	6	dest network unknown
3	7	dest host unknown
4	0	source quench (congestion control - not used)
8	0	echo request (ping)
9	0	route advertisement
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



COMPUTER NETWORKS

Ashwini M Joshi (Ph.D.)
Department of Computer Science and Engineering

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

Distance vector v/s Link state

	Distance vector	*Link State*
Primary principle	send larger updates , about the complete network only to neighboring routers	Send smaller updates, about the link state of neighbors, to all routers
updates		

Distance vector v/s Link state

	Distance vector	*Link State*
Primary principle	send larger updates , about the complete network only to neighboring routers	Send smaller updates, about the link state of neighbors, to all routers
Learning about network updates	Learn about network only from neighbors	Learn about network from all routers

Distance vector v/s Link state

	Distance vector	*Link State*
Primary principle	send larger updates , about the complete network only to neighboring routers	Send smaller updates, about the link state of neighbors, to all routers
Learning about network	Learn about network only from neighbors	Learn about network only from all routers
Building the routing table	Based on inputs from only neighbors	Based on complete database collected from all routers
updates		

Distance vector v/s Link state

	Distance vector	**Link State**
Primary principle	send larger updates , about the complete network only to neighboring routers	Send smaller updates, about the link state of neighbors, to all routers
Learning about network	Learn about network only from neighbors	Learn about network only from all routers
Building the routing table	Based on inputs from only neighbors	Based on complete database collected from all routers
Advertisement of updates	Periodically (e.g. every 30 seconds)	Triggered updates, only when there is a change

Distance vector v/s Link state

	Distance vector	**Link State**
Routing loops	More prone ; suffer from problems like count-to-infinity	Less prone to routing loops
		distance vector

Distance vector v/s Link state

	Distance vector	**Link State**
Routing loops	More prone ; suffer from problems like count-to-infinity	Less prone to routing loops
Convergence (stabilisation)	Slow	Fast

Distance vector v/s Link state

	Distance vector	**Link State**
Routing loops	More prone ; suffer from problems like count-to-infinity	Less prone to routing loops
Convergence (stabilisation)	Slow	Fast
Resources	Less CPU power and memory	More CPU power and memory required

Distance vector v/s Link state

	Distance vector	**Link State**
Routing loops	More prone ; suffer from problems like count-to-infinity	Less prone to routing loops
Convergence (stabilisation)	Slow	Fast
Resources	Less CPU power and memory	More CPU power and memory required
Scalability		More scalable than distance vector

Distance vector v/s Link state

	Distance vector	**Link State**
Routing loops	More prone ; suffer from problems like count-to-infinity	Less prone to routing loops
Convergence (stabilisation)	Slow	Fast
Resources	Less CPU power and memory	More CPU power and memory required
Cost		More than Distance vector
Scalability		More scalable than distance vector

THANK YOU

Ashwini M. Joshi
Department of Computer Science and Engineering

ashwinimjoshi@pes.edu

Summary