



Data Communications and Networking

Fourth Edition

Forouzan

Chapter 19

Network Layer: Logical Addressing

19-1 IPv4 ADDRESSES

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.

Topics discussed in this section:

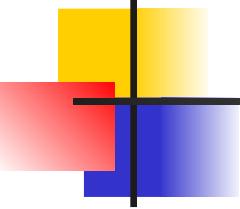
Address Space

Notations

Classful Addressing

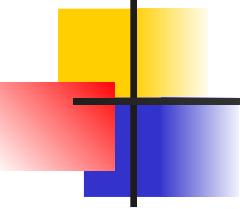
Classless Addressing

Network Address Translation (NAT)



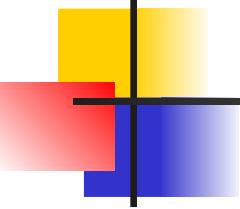
Note

An IPv4 address is 32 bits long.



Note

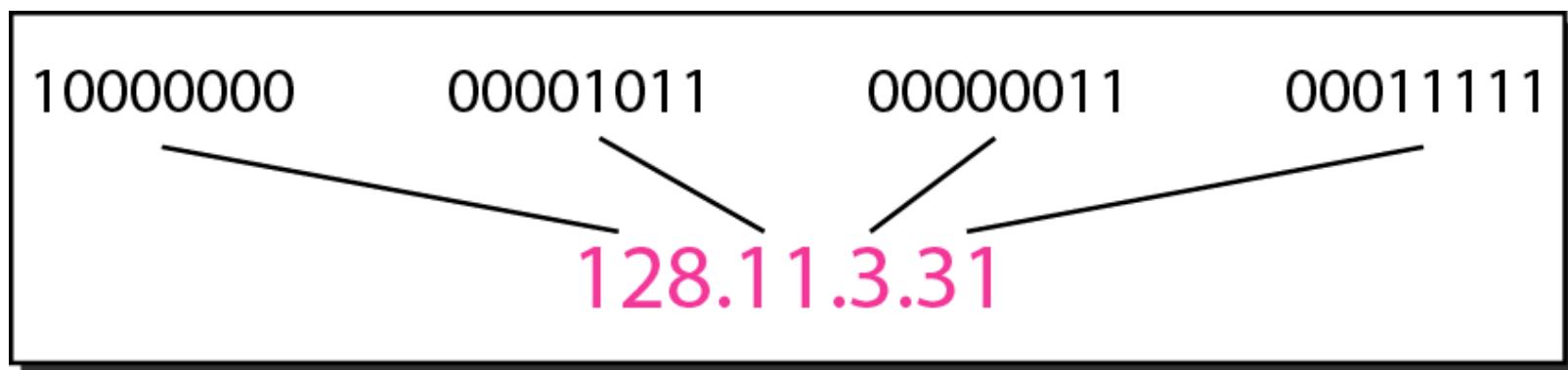
**The IPv4 addresses are unique
and universal.**

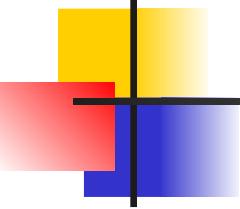


Note

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

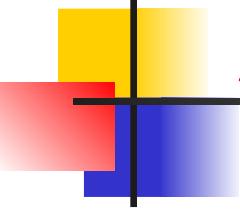
Figure 19.1 Dotted-decimal notation and binary notation for an IPv4 address





Note

**Numbering systems are reviewed in
Appendix B.**



Example 19.1

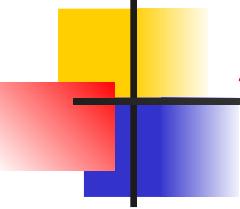
Change the following IPv4 addresses from binary notation to dotted-decimal notation.

- 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



Example 19.2

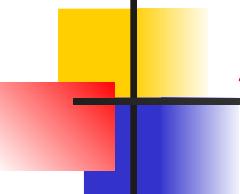
Change the following IPv4 addresses from dotted-decimal notation to binary notation.

- 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



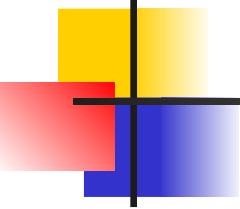
Example 19.3

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.**



Note

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

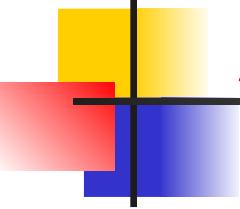
Figure 19.2 Finding the classes in binary and dotted-decimal notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0			
Class B	10			
Class C	110			
Class D	1110			
Class E	1111			

a. Binary notation

	First byte	Second byte	Third byte	Fourth byte
Class A	0–127			
Class B	128–191			
Class C	192–223			
Class D	224–239			
Class E	240–255			

b. Dotted-decimal notation



Example 19.4

Find the class of each address.

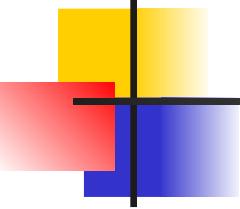
- a.** 00000001 00001011 00001011 11101111
- b.** 11000001 10000011 00011011 11111111
- c.** 14.23.120.8
- d.** 252.5.15.111

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.*

Table 19.1 *Number of blocks and block size in classful IPv4 addressing*

<i>Class</i>	<i>Number of Blocks</i>	<i>Block Size</i>	<i>Application</i>
A	128	16,777,216	Unicast
B	16,384	65,536	Unicast
C	2,097,152	256	Unicast
D	1	268,435,456	Multicast
E	1	268,435,456	Reserved

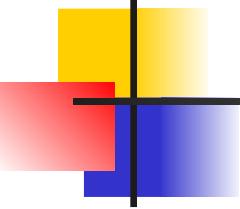


Note

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

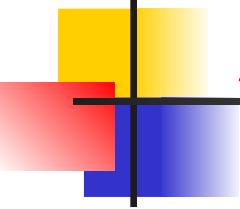
Table 19.2 *Default masks for classful addressing*

Class	Binary	Dotted-Decimal	CIDR
A	11111111 00000000 00000000 00000000	255.0.0.0	/8
B	11111111 11111111 00000000 00000000	255.255.0.0	/16
C	11111111 11111111 11111111 00000000	255.255.255.0	/24



Note

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

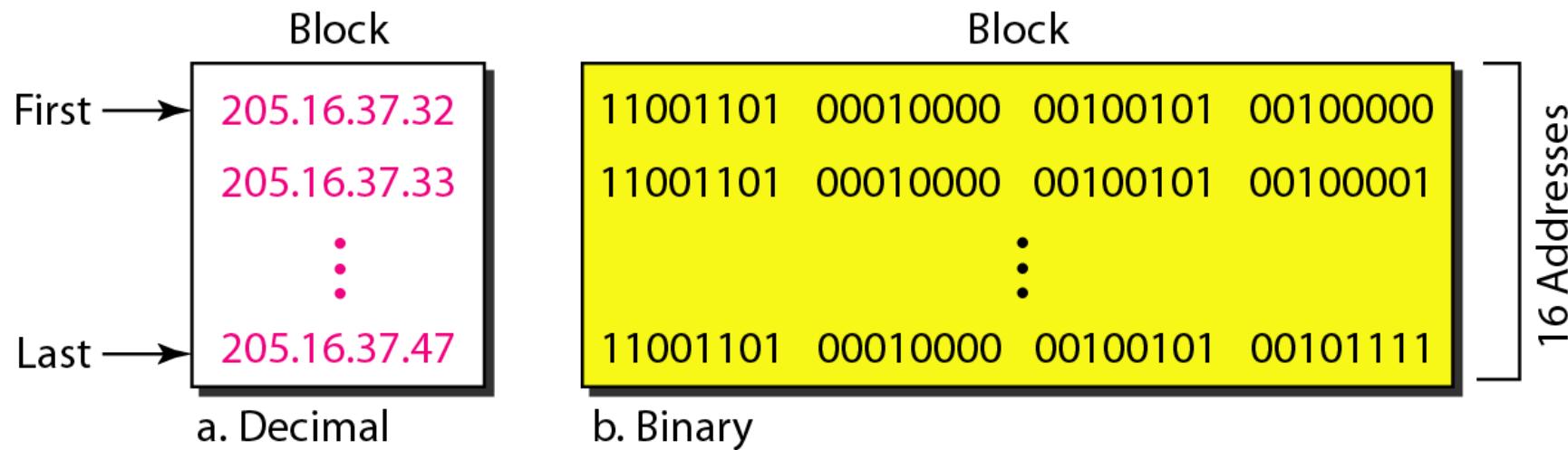


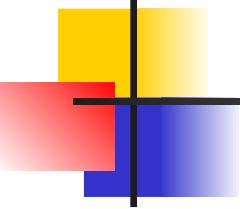
Example 19.5

Figure 19.3 shows a block of addresses, in both binary and dotted-decimal notation, granted to a small business that needs 16 addresses.

We can see that the restrictions are applied to this block. The addresses are contiguous. The number of addresses is a power of 2 ($16 = 2^4$), and the first address is divisible by 16. The first address, when converted to a decimal number, is 3,440,387,360, which when divided by 16 results in 215,024,210.

Figure 19.3 A block of 16 addresses granted to a small organization



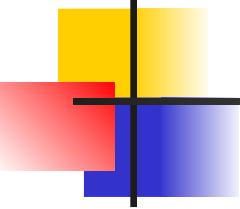


Note

In IPv4 addressing, a block of addresses can be defined as

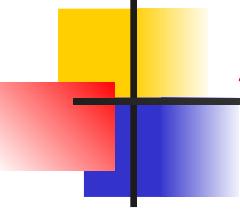
x.y.z.t /n

in which x.y.z.t defines one of the addresses and the /n defines the mask.



Note

The first address in the block can be found by setting the rightmost $32 - n$ bits to 0s.



Example 19.6

A block of addresses is granted to a small organization. We know that one of the addresses is 205.16.37.39/28. What is the first address in the block?

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

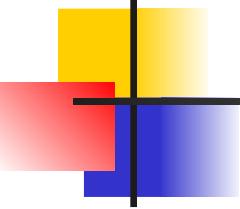
If we set 32–28 rightmost bits to 0, we get

11001101 00010000 00100101 0010000

or

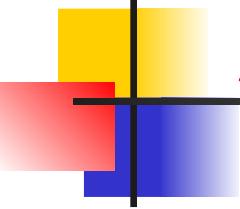
205.16.37.32.

This is actually the block shown in Figure 19.3.



Note

The last address in the block can be found by setting the rightmost $32 - n$ bits to 1s.



Example 19.7

Find the last address for the block in Example 19.6.

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

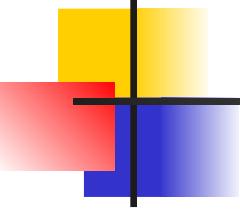
If we set 32 – 28 rightmost bits to 1, we get

11001101 00010000 00100101 00101111

or

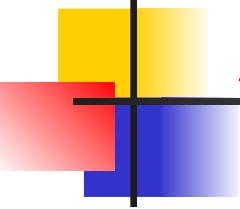
205.16.37.47

This is actually the block shown in Figure 19.3.



Note

**The number of addresses in the block
can be found by using the formula
 2^{32-n} .**

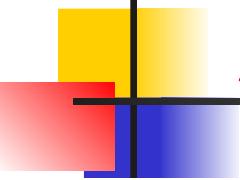


Example 19.8

Find the number of addresses in Example 19.6.

Solution

The value of n is 28, which means that number of addresses is 2^{32-28} or 16.



Example 19.9

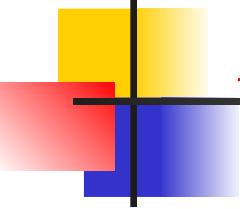
Another way to find the first address, the last address, and the number of addresses is to represent the mask as a 32-bit binary (or 8-digit hexadecimal) number. This is particularly useful when we are writing a program to find these pieces of information. In Example 19.5 the /28 can be represented as

11111111 11111111 11111111 11110000

(twenty-eight 1s and four 0s).

Find

- a. The first address*
- b. The last address*
- c. The number of addresses.*



Example 19.9 (continued)

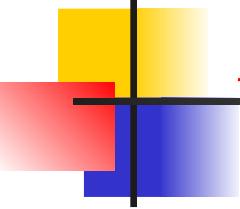
Solution

a. The first address can be found by ANDing the given addresses with the mask. ANDing here is done bit by bit. The result of ANDing 2 bits is 1 if both bits are 1s; the result is 0 otherwise.

Address:	11001101	00010000	00100101	00100111
----------	----------	----------	----------	----------

Mask:	11111111	11111111	11111111	11110000
-------	-----------------	-----------------	-----------------	-----------------

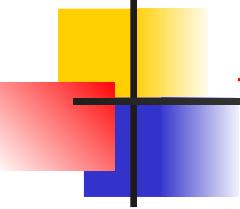
First address:	11001101	00010000	00100101	00100000
----------------	----------	----------	----------	----------



Example 19.9 (continued)

b. *The last address can be found by ORing the given addresses with the complement of the mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.*

Address:	11001101	00010000	00100101	00100111
Mask complement:	00000000	00000000	00000000	00001111
Last address:	11001101	00010000	00100101	00101111



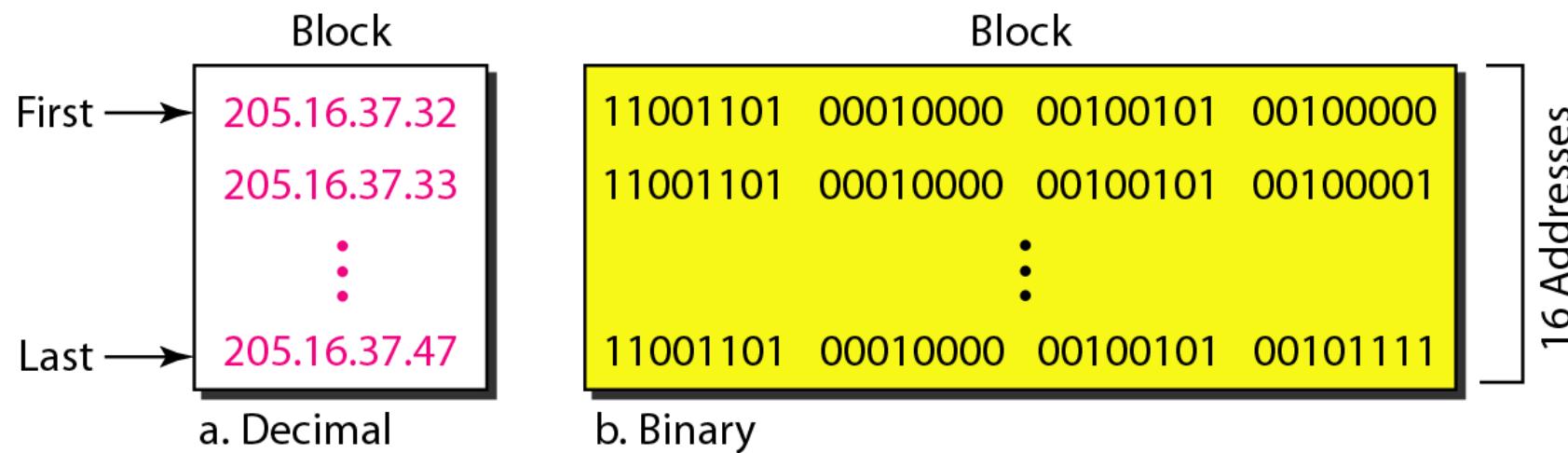
Example 19.9 (continued)

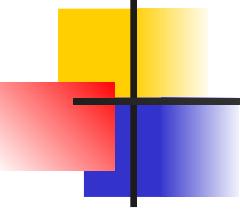
- c. The number of addresses can be found by complementing the mask, interpreting it as a decimal number, and adding 1 to it.

Mask complement: 00000000 00000000 00000000 00001111

Number of addresses: $15 + 1 = 16$

Figure 19.4 A network configuration for the block 205.16.37.32/28





Note

The first address in a block is normally not assigned to any device; it is used as the network address that represents the organization to the rest of the world.

Figure 19.5 *Two levels of hierarchy in an IPv4 address*

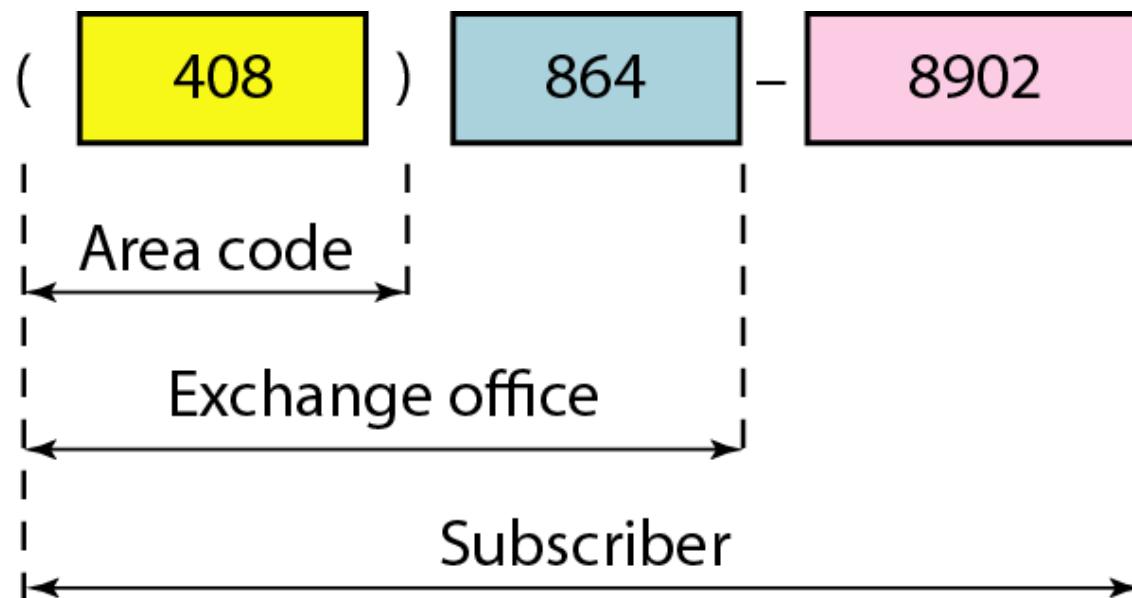
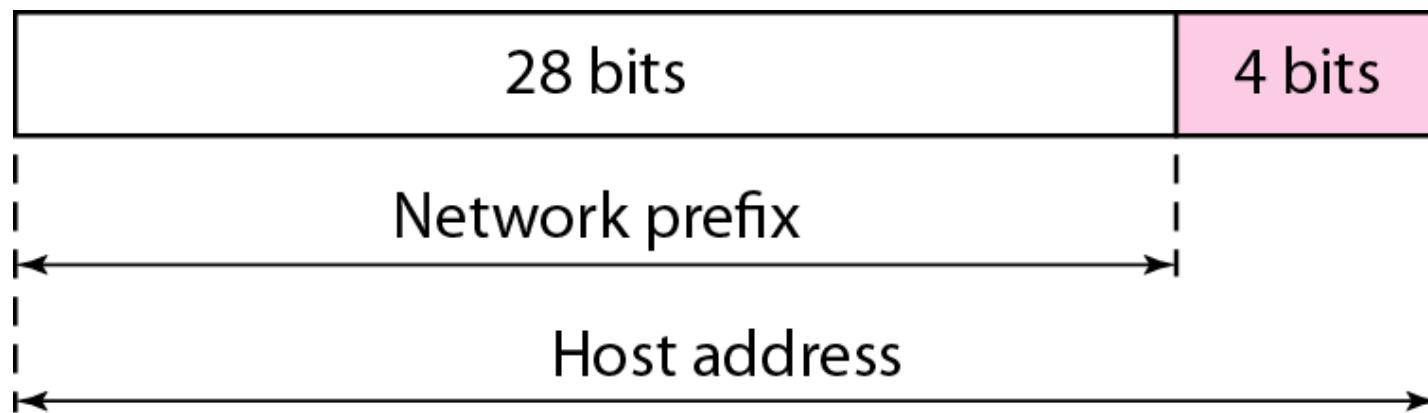
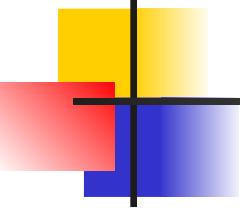


Figure 19.6 A frame in a character-oriented protocol





Note

Each address in the block can be considered as a two-level hierarchical structure:
the leftmost n bits (prefix) define the network;
the rightmost $32 - n$ bits define the host.

Figure 19.7 Configuration and addresses in a subnetted network

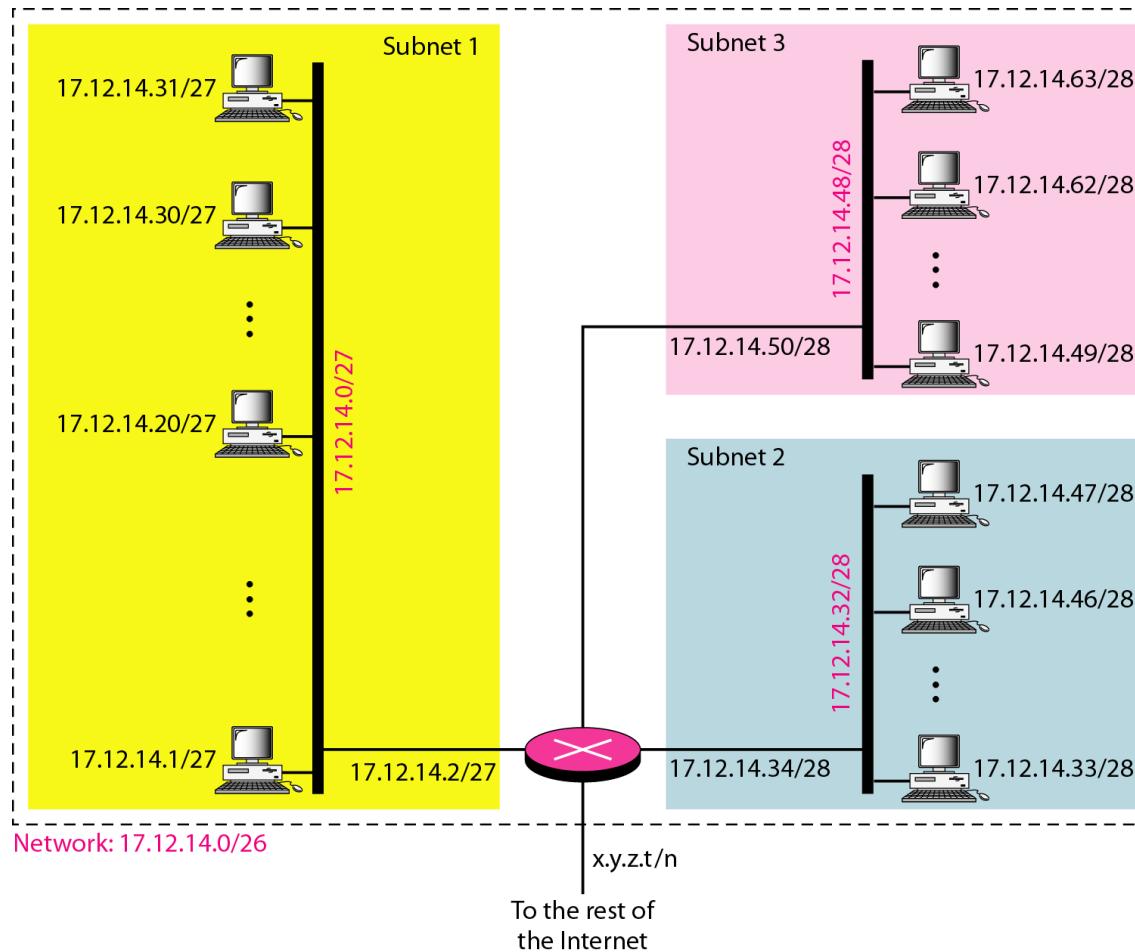
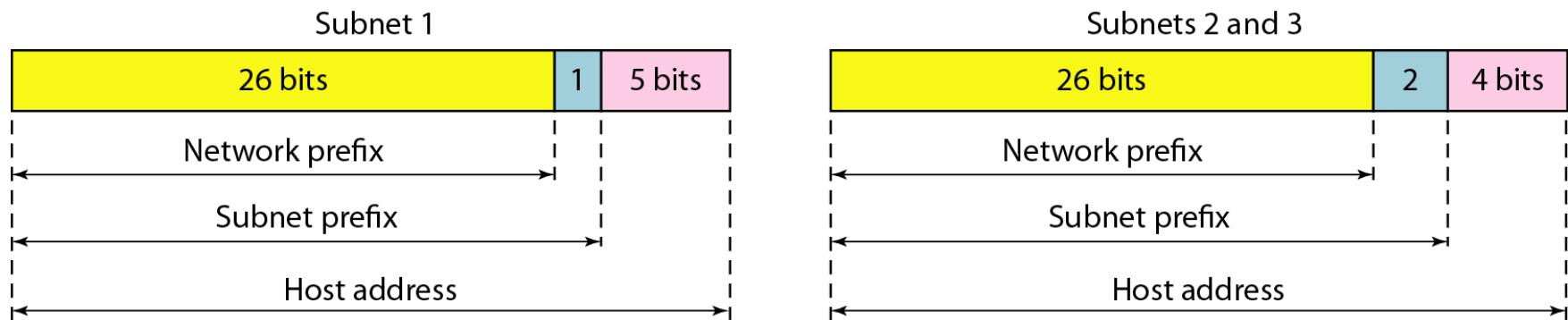
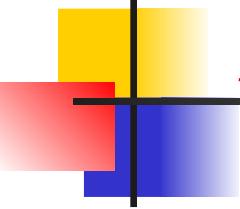


Figure 19.8 Three-level hierarchy in an IPv4 address





Example 19.10

An ISP is granted a block of addresses starting with 190.100.0.0/16 (65,536 addresses). The ISP needs to distribute these addresses to three groups of customers as follows:

- a. The first group has 64 customers; each needs 256 addresses.*
- b. The second group has 128 customers; each needs 128 addresses.*
- c. The third group has 128 customers; each needs 64 addresses.*

Design the subblocks and find out how many addresses are still available after these allocations.

Example 19.10 (continued)

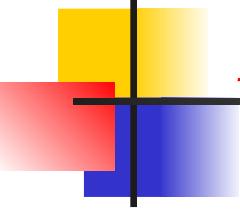
Solution

Figure 19.9 shows the situation.

Group 1

For this group, each customer needs 256 addresses. This means that 8 ($\log_2 256$) bits are needed to define each host. The prefix length is then $32 - 8 = 24$. The addresses are

1st Customer:	190.100.0.0/24	190.100.0.255/24
2nd Customer:	190.100.1.0/24	190.100.1.255/24
...		
64th Customer:	190.100.63.0/24	190.100.63.255/24
$Total = 64 \times 256 = 16,384$		



Example 19.10 (continued)

Group 2

For this group, each customer needs 128 addresses. This means that 7 ($\log_2 128$) bits are needed to define each host. The prefix length is then $32 - 7 = 25$. The addresses are

<i>1st Customer:</i>	190.100.64.0/25	190.100.64.127/25
<i>2nd Customer:</i>	190.100.64.128/25	190.100.64.255/25
...		
<i>128th Customer:</i>	190.100.127.128/25	190.100.127.255/25
<i>Total = $128 \times 128 = 16,384$</i>		

Example 19.10 (continued)

Group 3

For this group, each customer needs 64 addresses. This means that 6 ($\log_2 64$) bits are needed to each host. The prefix length is then $32 - 6 = 26$. The addresses are

<i>1st Customer:</i>	190.100.128.0/26	190.100.128.63/26
<i>2nd Customer:</i>	190.100.128.64/26	190.100.128.127/26
...		
<i>128th Customer:</i>	190.100.159.192/26	190.100.159.255/26
<i>Total = $128 \times 64 = 8192$</i>		

Number of granted addresses to the ISP: 65,536

Number of allocated addresses by the ISP: 40,960

Number of available addresses: 24,576

Figure 19.9 An example of address allocation and distribution by an ISP

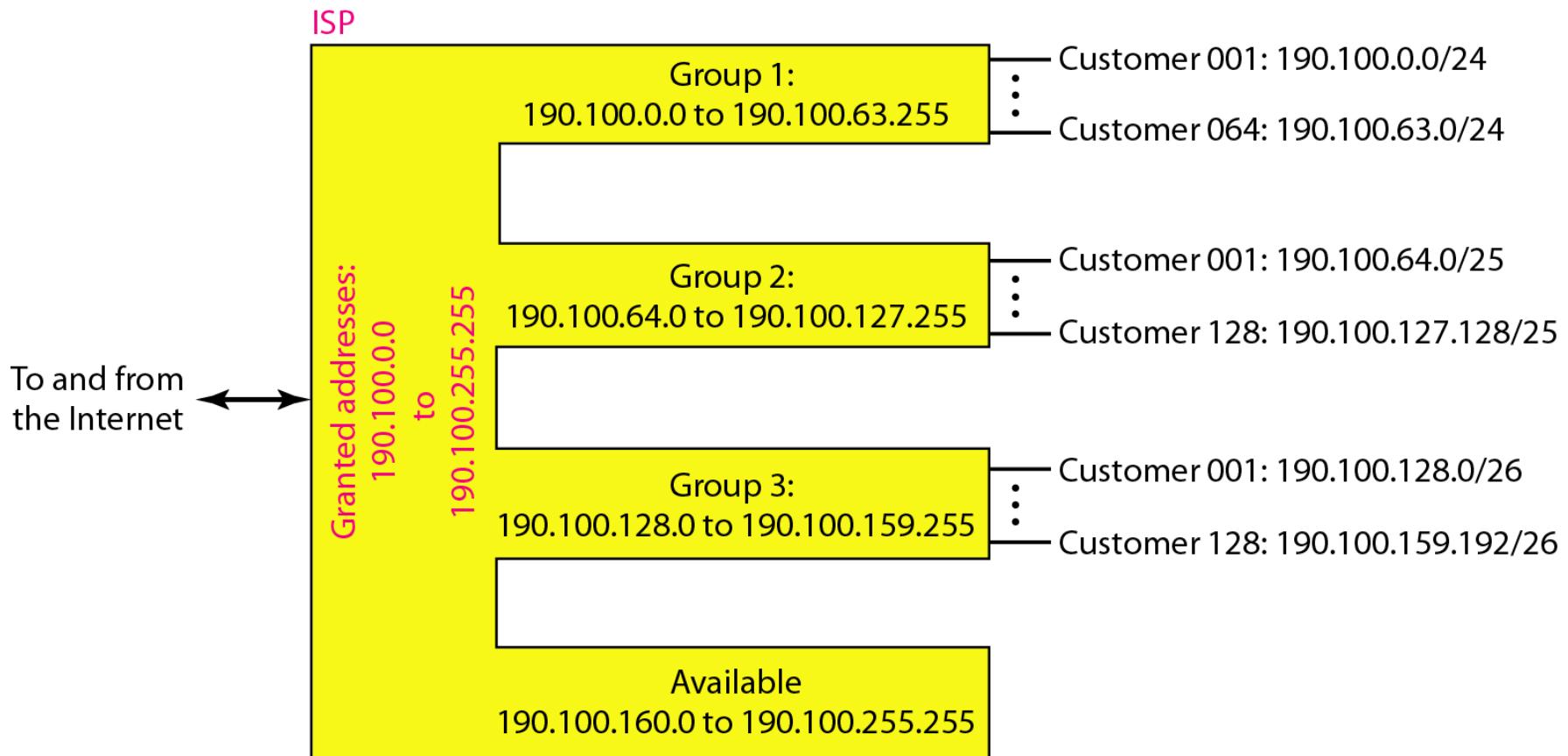


Table 19.3 Addresses for private networks

<i>Range</i>		<i>Total</i>
10.0.0.0	to	2^{24}
172.16.0.0	to	2^{20}
192.168.0.0	to	2^{16}

Figure 19.10 A NAT implementation

Site using private addresses

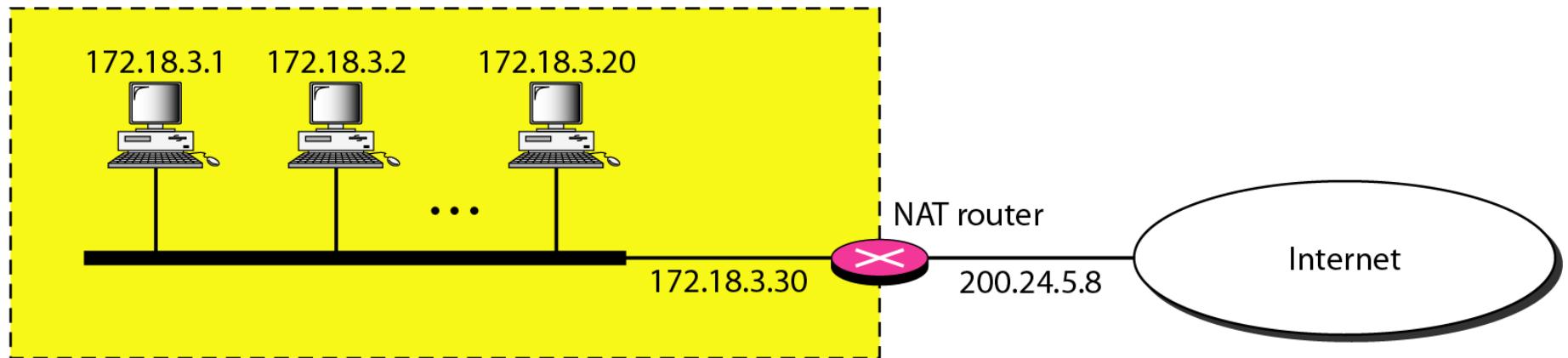


Figure 19.11 Addresses in a NAT

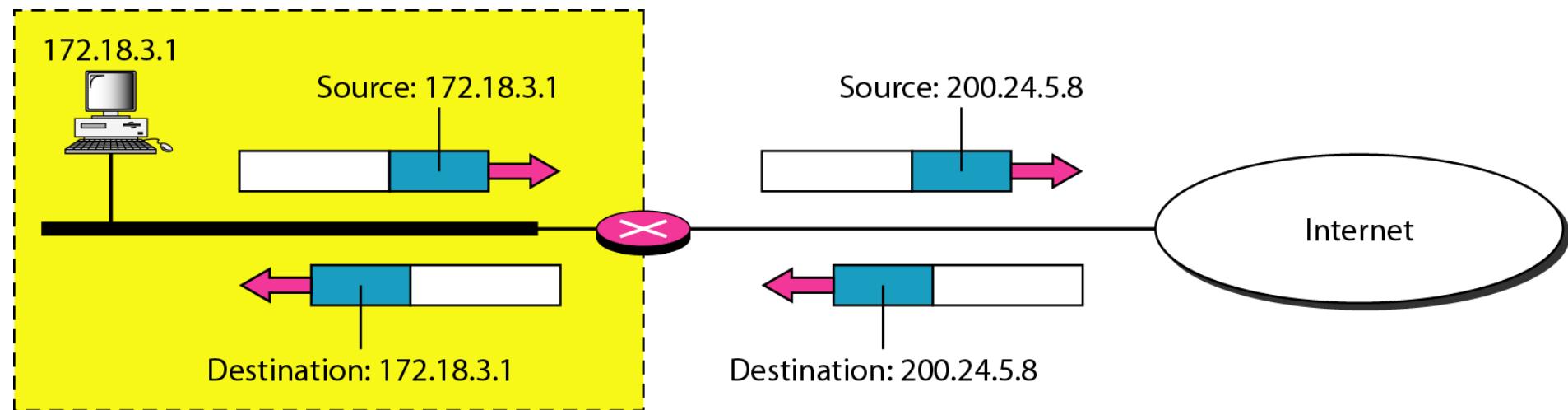


Figure 19.12 NAT address translation

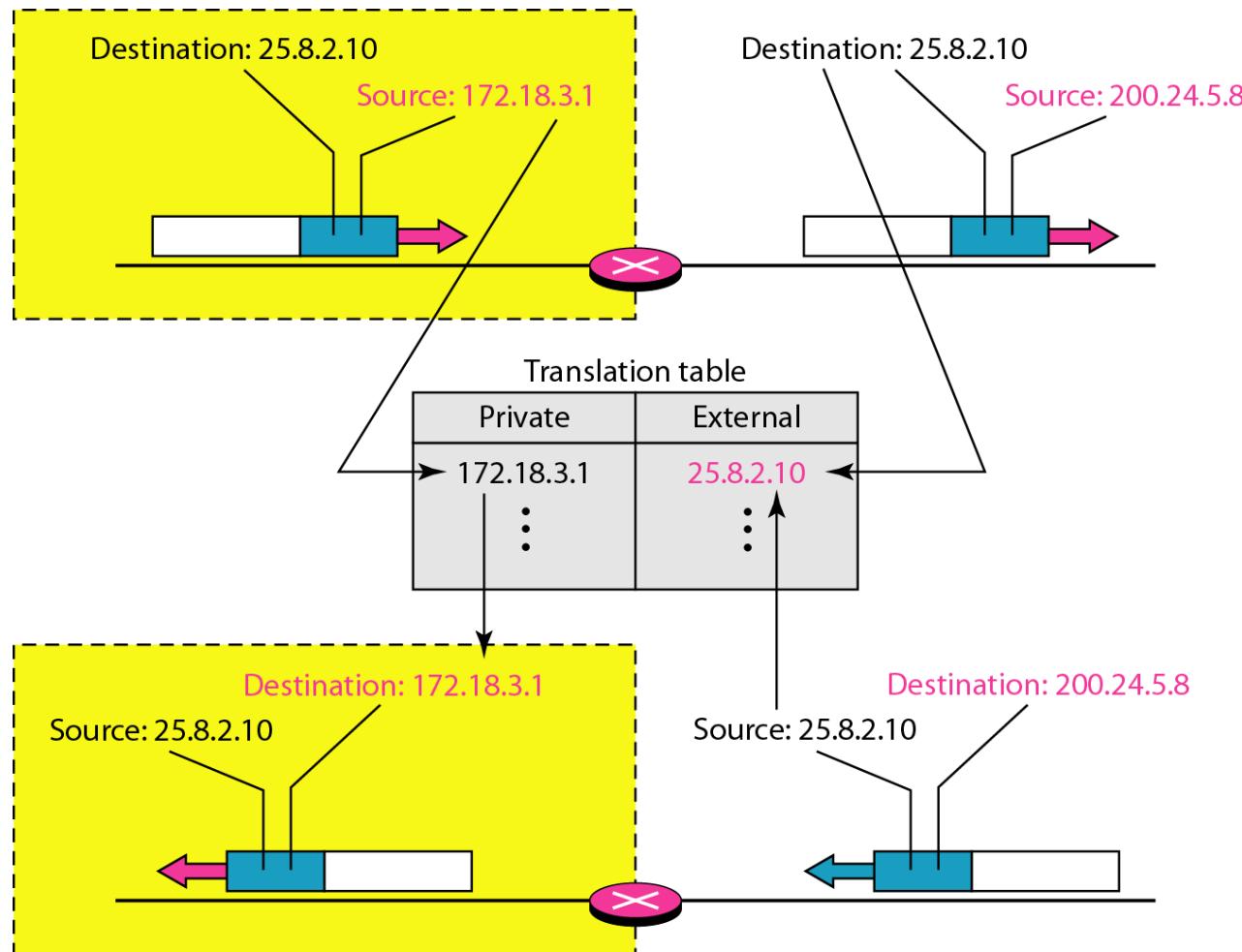
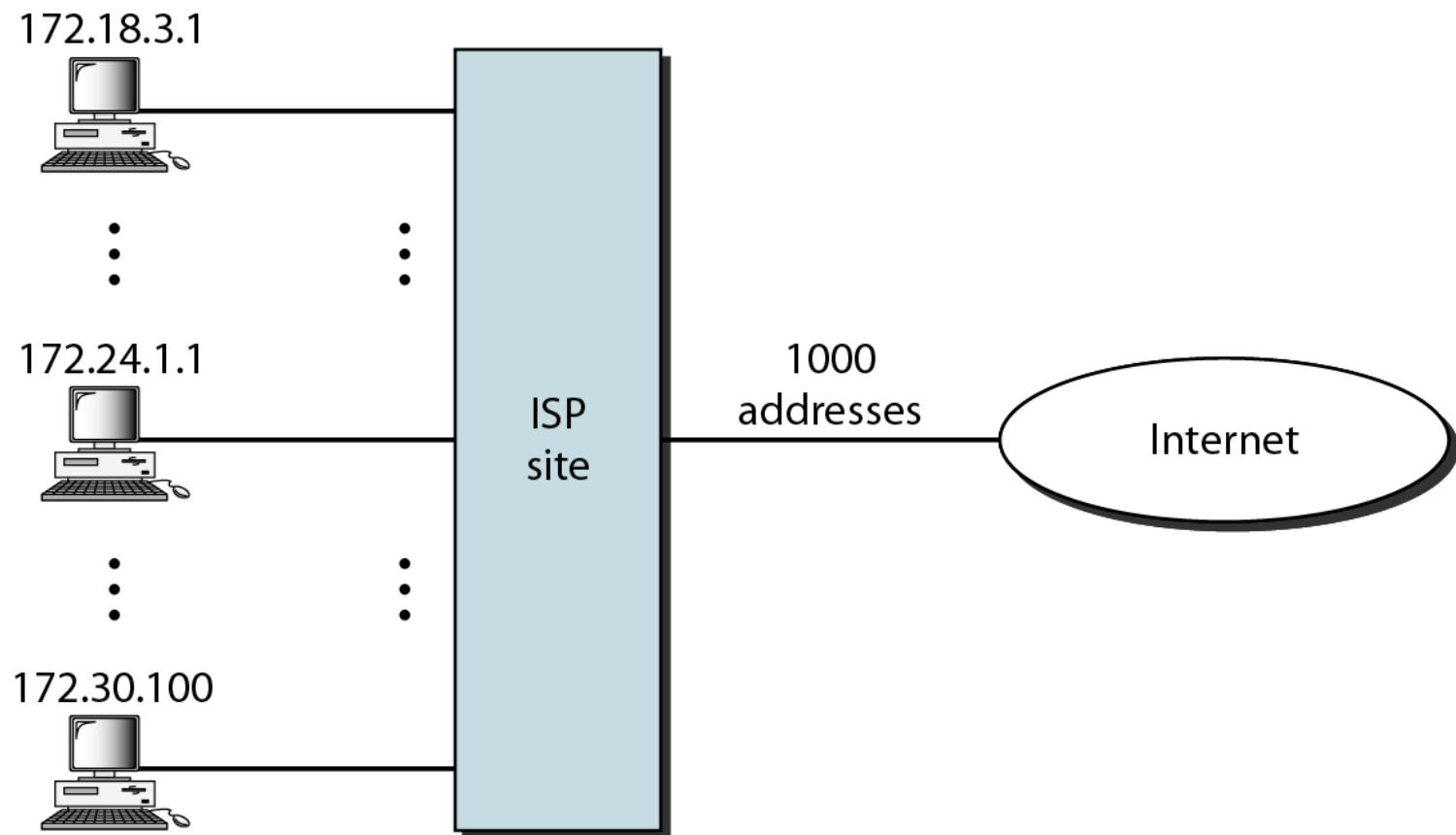


Table 19.4 Five-column translation table

<i>Private Address</i>	<i>Private Port</i>	<i>External Address</i>	<i>External Port</i>	<i>Transport Protocol</i>
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
...

Figure 19.13 An ISP and NAT



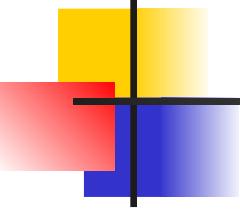
19-2 IPv6 ADDRESSES

Despite all short-term solutions, address depletion is still a long-term problem for the Internet. This and other problems in the IP protocol itself have been the motivation for IPv6.

Topics discussed in this section:

Structure

Address Space



Note

An IPv6 address is 128 bits long.

Figure 19.14 IPv6 address in binary and hexadecimal colon notation

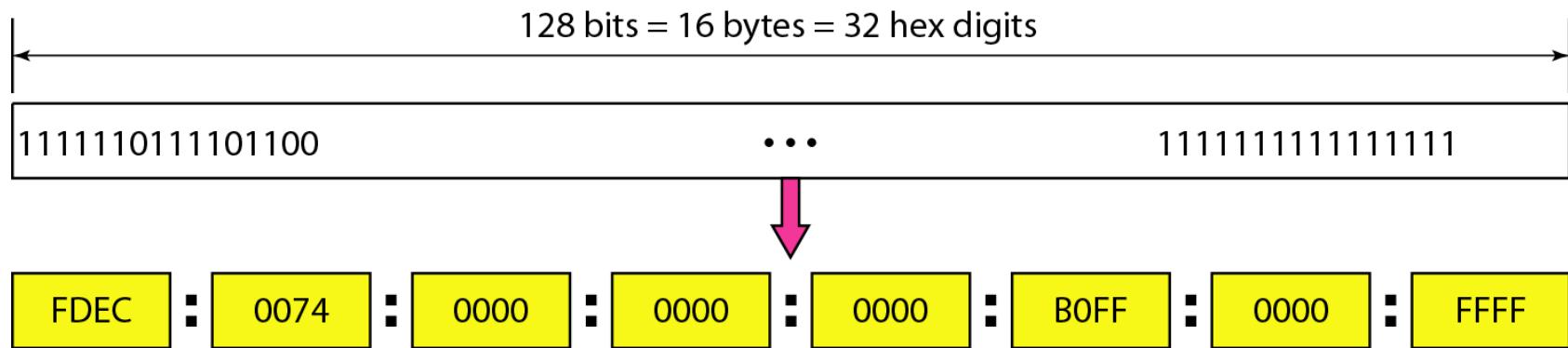
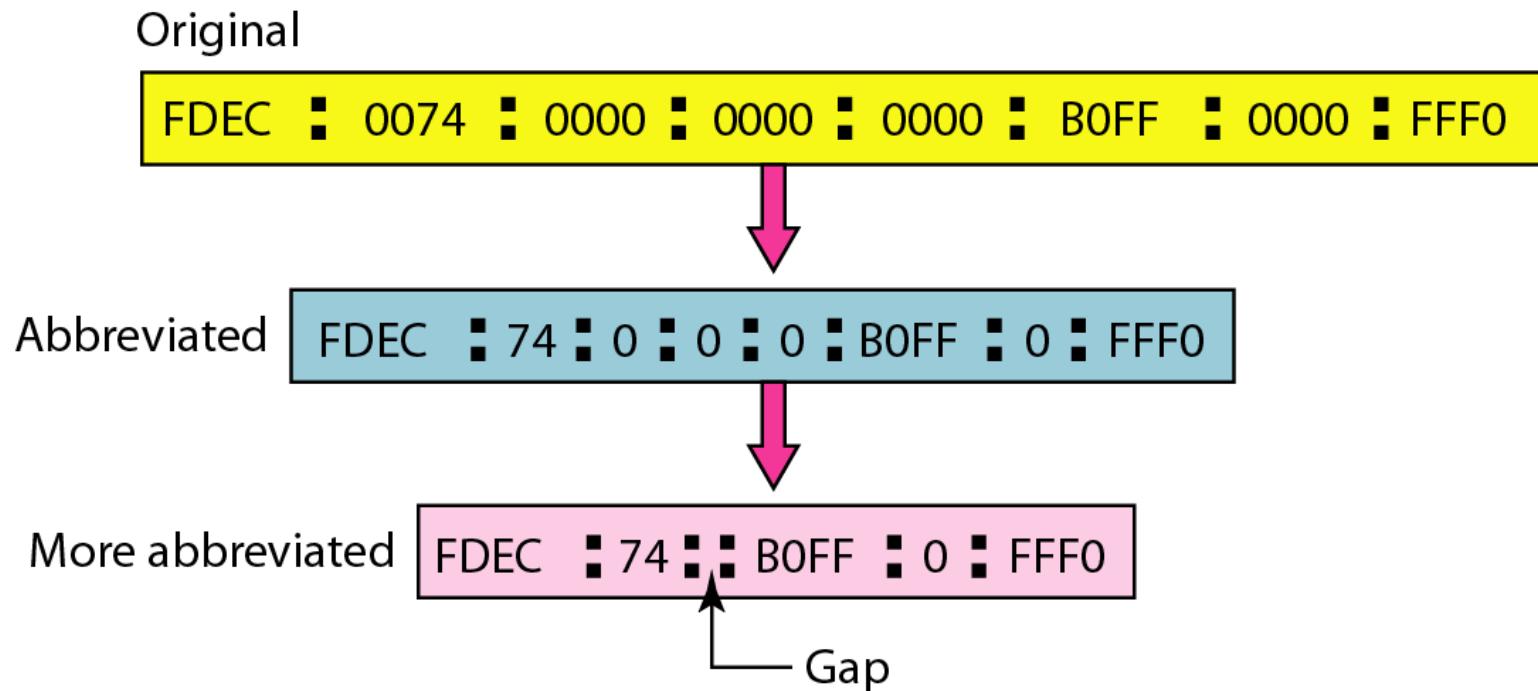
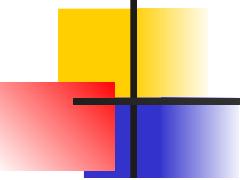


Figure 19.15 Abbreviated IPv6 addresses





Example 19.11

Expand the address 0:15::1:12:1213 to its original.

Solution

We first need to align the left side of the double colon to the left of the original pattern and the right side of the double colon to the right of the original pattern to find how many 0s we need to replace the double colon.

XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX:XXXX
0: 15: : 1: 12:1213

This means that the original address is.

0000:0015:0000:0000:0000:0001:0012:1213

Table 19.5 Type prefixes for IPv6 addresses

Type Prefix	Type	Fraction
0000 0000	Reserved	1/256
0000 0001	Unassigned	1/256
0000 001	ISO network addresses	1/128
0000 010	IPX (Novell) network addresses	1/128
0000 011	Unassigned	1/128
0000 1	Unassigned	1/32
0001	Reserved	1/16
001	Reserved	1/8
010	Provider-based unicast addresses	1/8

Table 19.5 Type prefixes for IPv6 addresses (continued)

Type Prefix	Type	Fraction
011	Unassigned	1/8
100	Geographic-based unicast addresses	1/8
101	Unassigned	1/8
110	Unassigned	1/8
1110	Unassigned	1/16
1111 0	Unassigned	1/32
1111 10	Unassigned	1/64
1111 110	Unassigned	1/128
1111 1110 0	Unassigned	1/512
1111 1110 10	Link local addresses	1/1024
1111 1110 11	Site local addresses	1/1024
1111 1111	Multicast addresses	1/256

Figure 19.16 Prefixes for provider-based unicast address

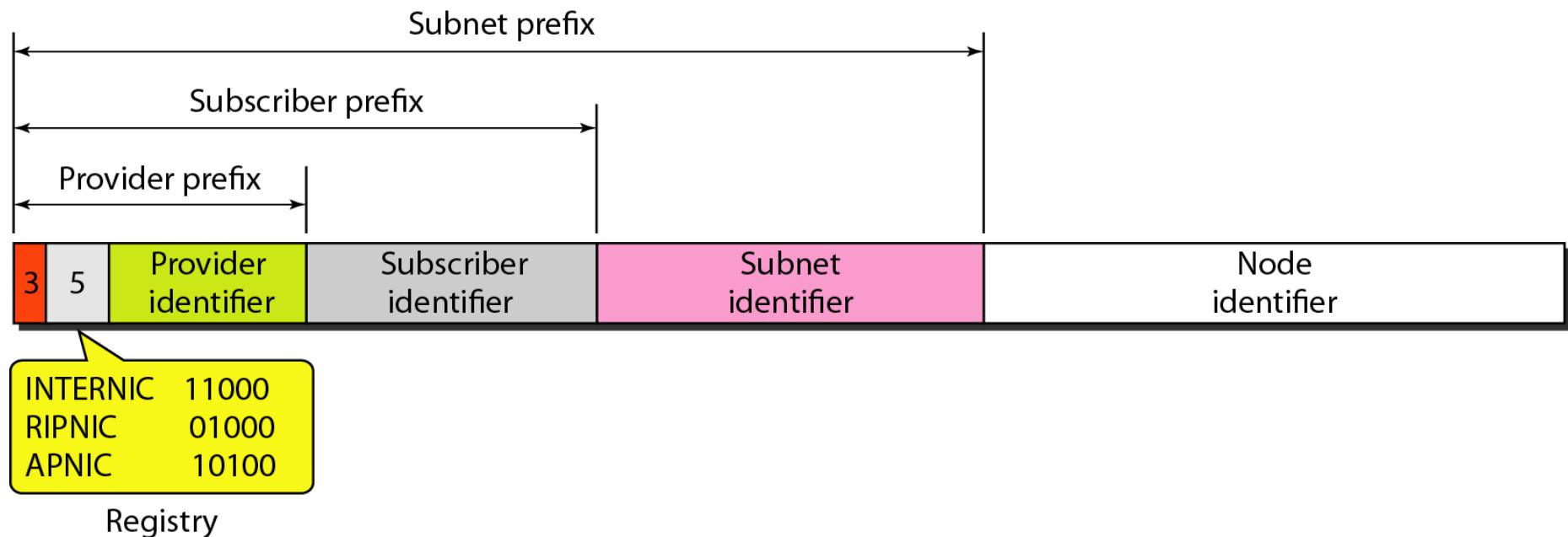


Figure 19.17 Multicast address in IPv6

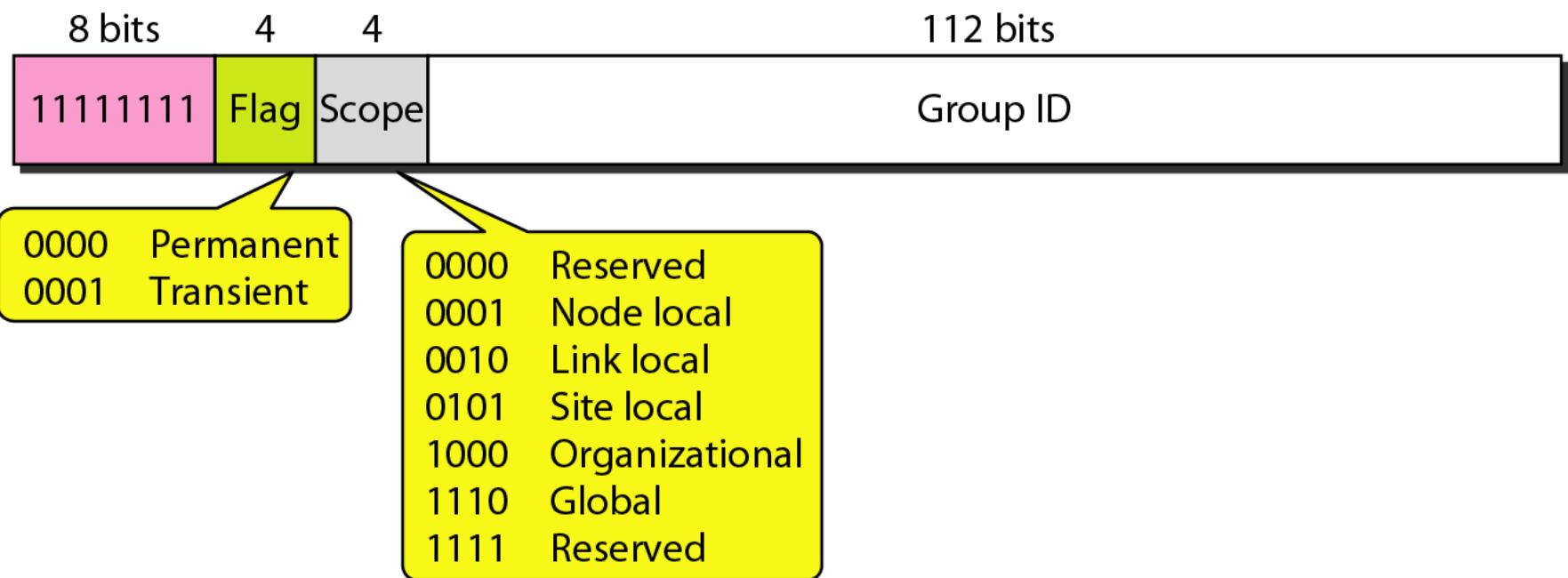


Figure 19.18 Reserved addresses in IPv6

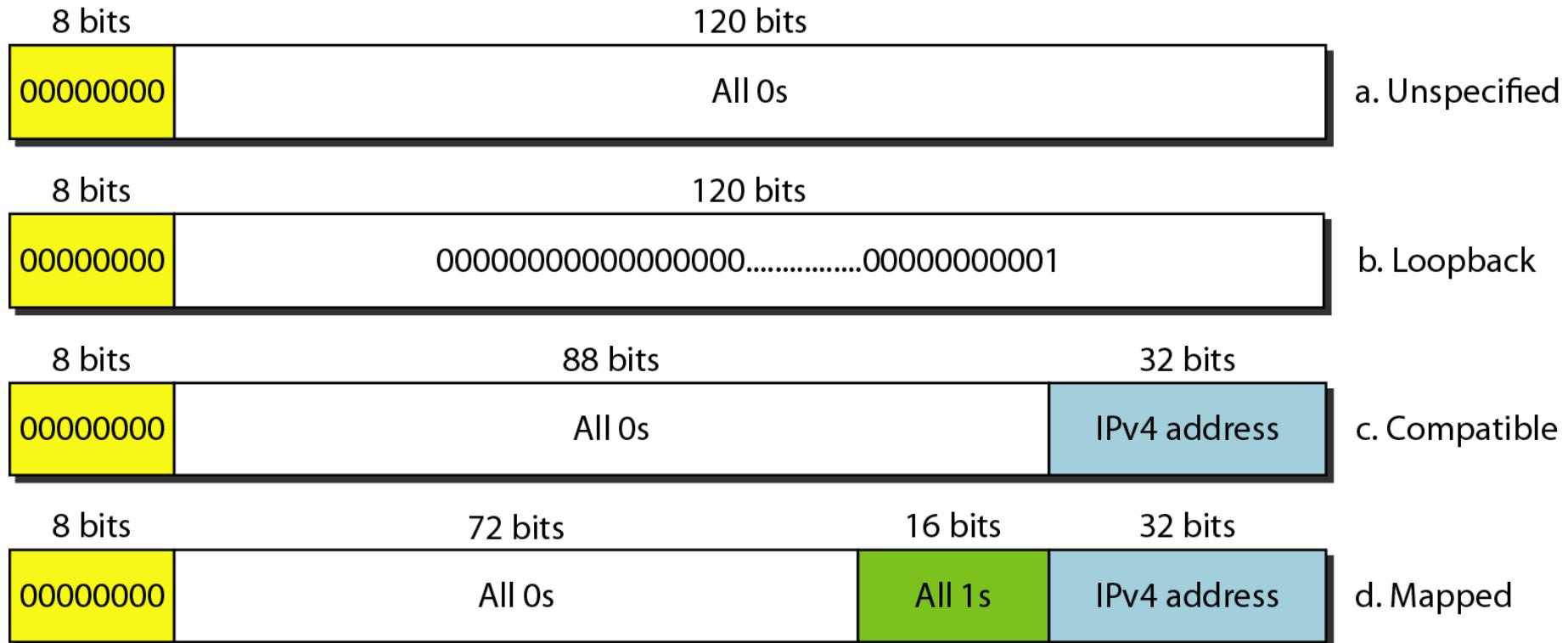
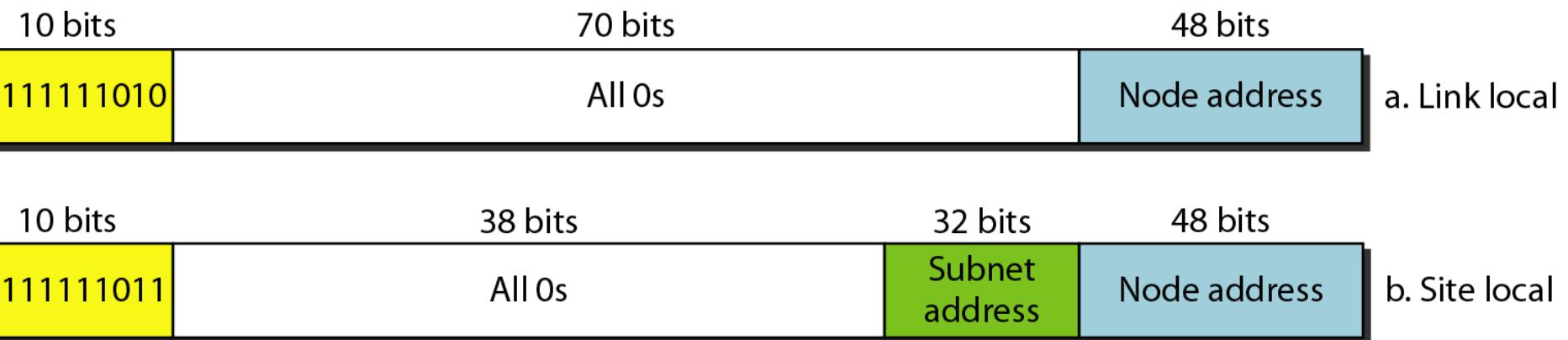


Figure 19.19 Local addresses in IPv6





Data Communications and Networking

Fourth Edition

Forouzan

Chapter 20

Network Layer: Internet Protocol

20-1 INTERNETWORKING

In this section, we discuss internetworking, connecting networks together to make an internetwork or an internet.

Topics discussed in this section:

Need for Network Layer

Internet as a Datagram Network

Internet as a Connectionless Network

Figure 20.1 *Links between two hosts*

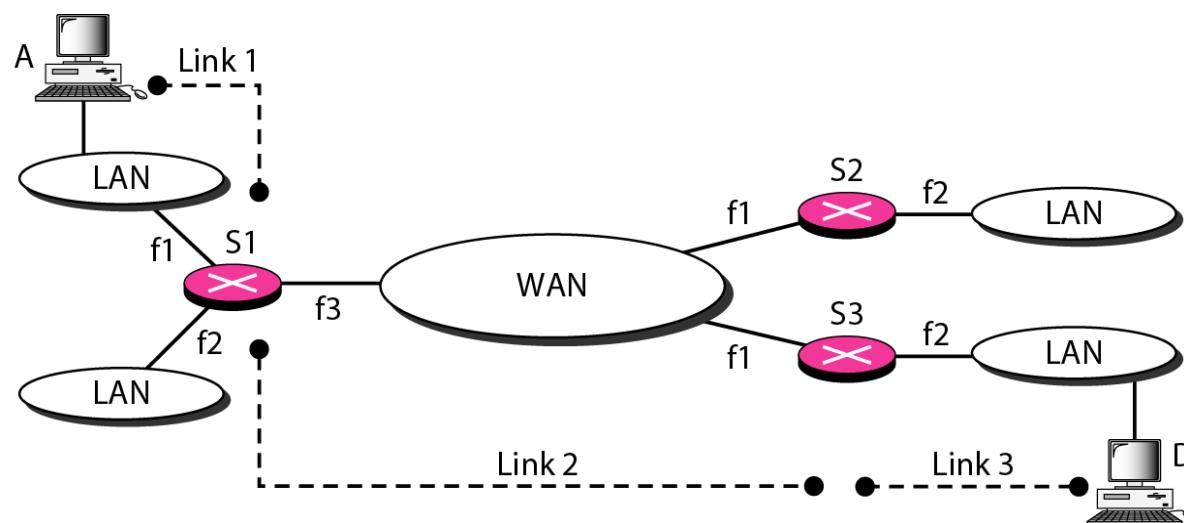


Figure 20.2 Network layer in an internetwork

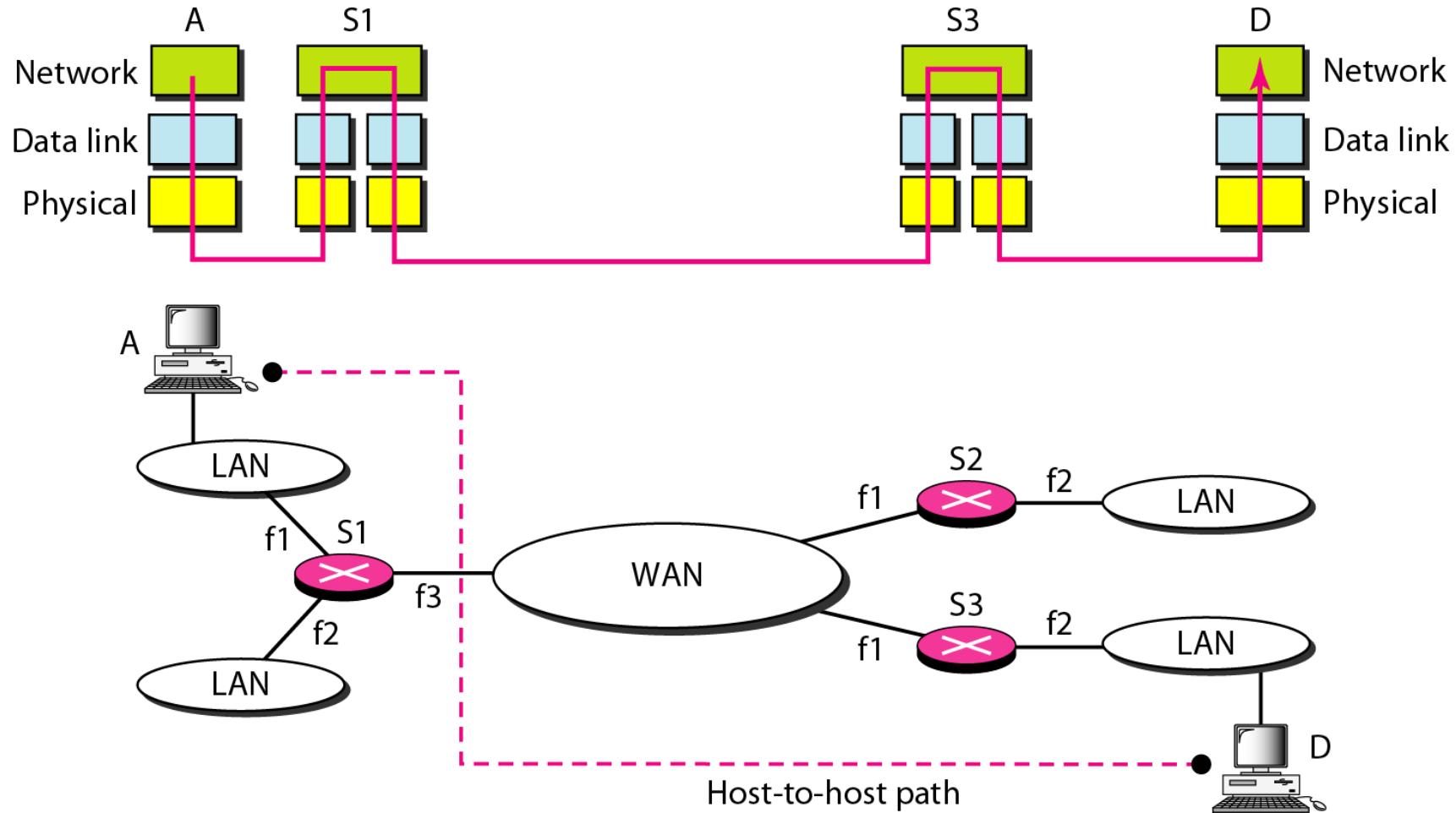


Figure 20.3 Network layer at the source, router, and destination

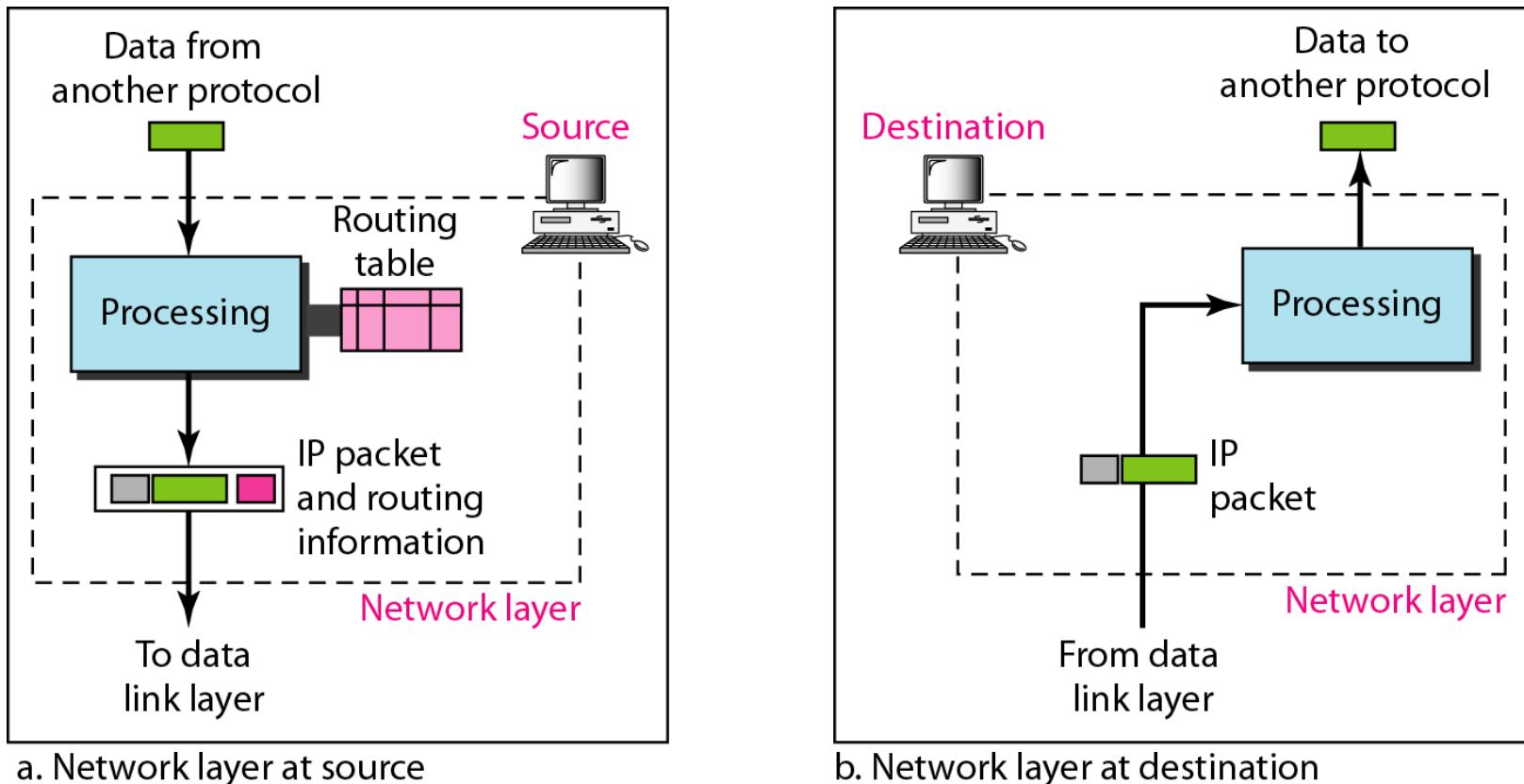
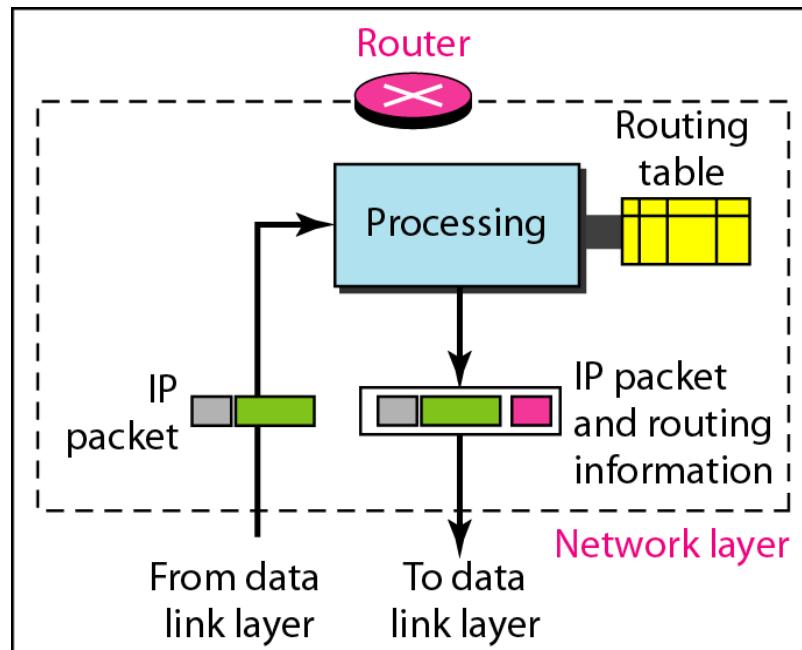
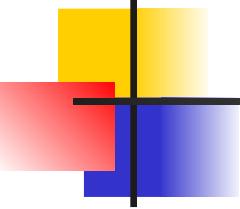


Figure 20.3 Network layer at the source, router, and destination (continued)

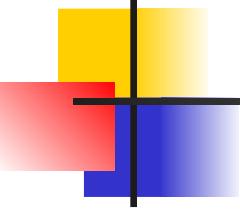


c. Network layer at a router



Note

Switching at the network layer in the Internet uses the datagram approach to packet switching.



Note

**Communication at the network layer in
the Internet is connectionless.**

20-2 IPv4

The Internet Protocol version 4 (IPv4) is the delivery mechanism used by the TCP/IP protocols.

Topics discussed in this section:

Datagram

Fragmentation

Checksum

Options

Figure 20.4 Position of IPv4 in TCP/IP protocol suite

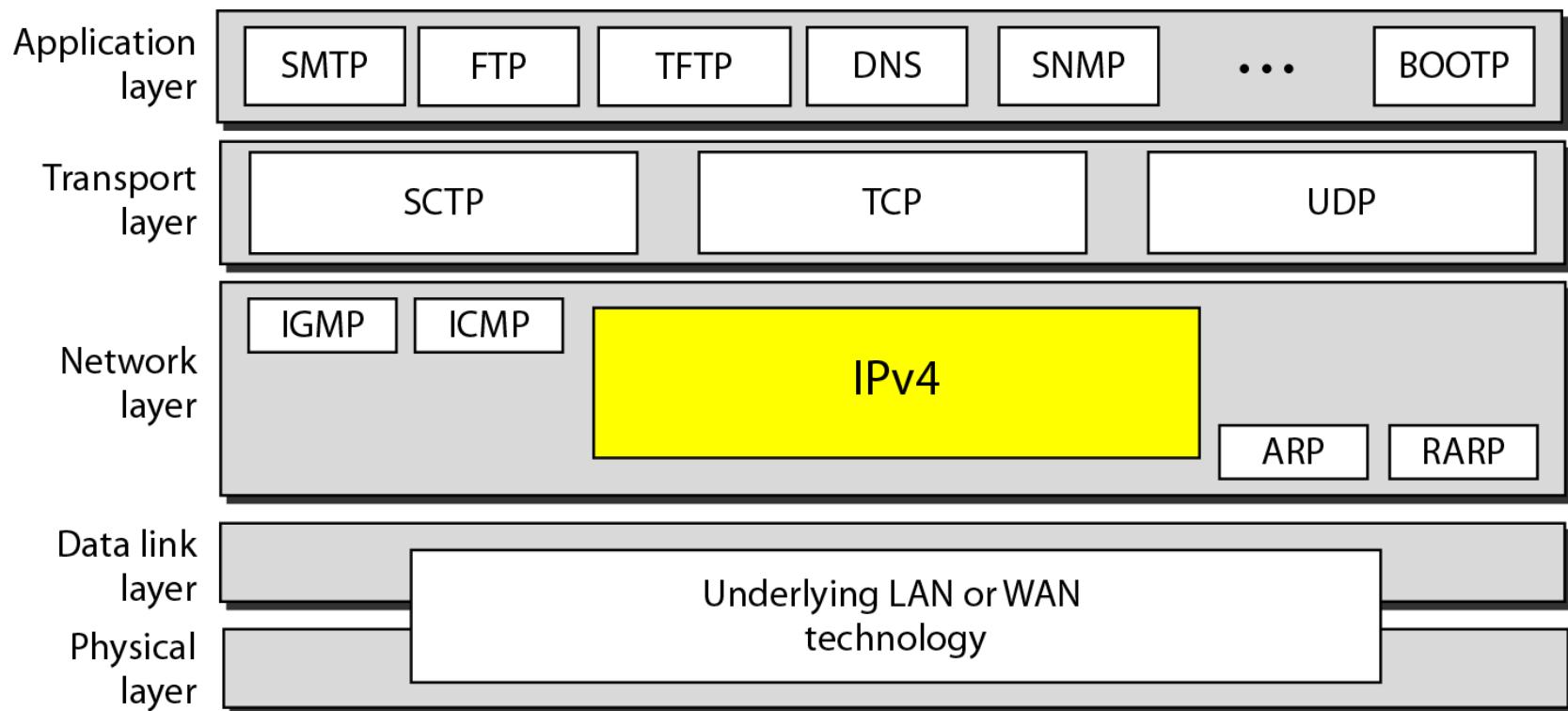


Figure 20.5 IPv4 datagram format

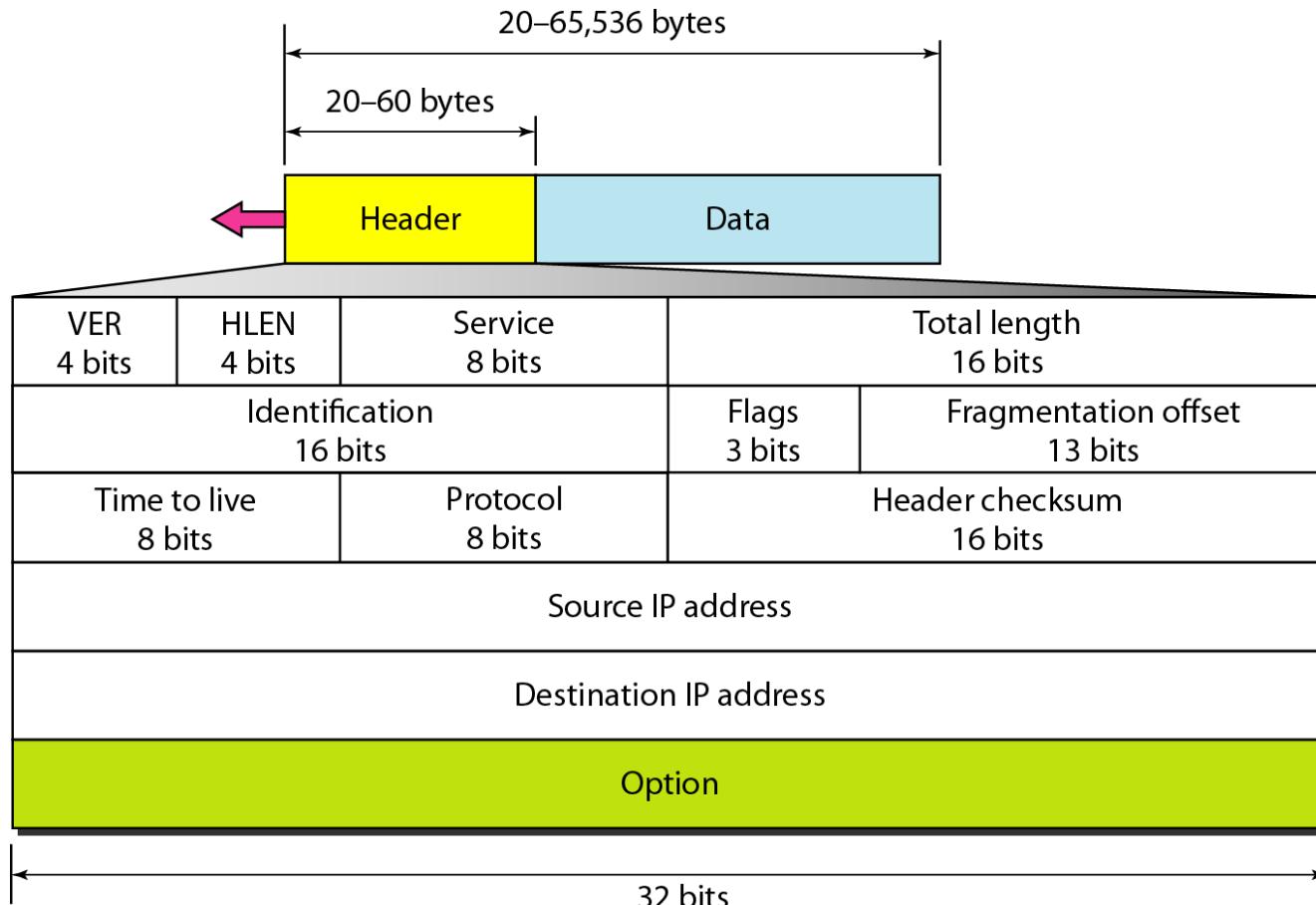
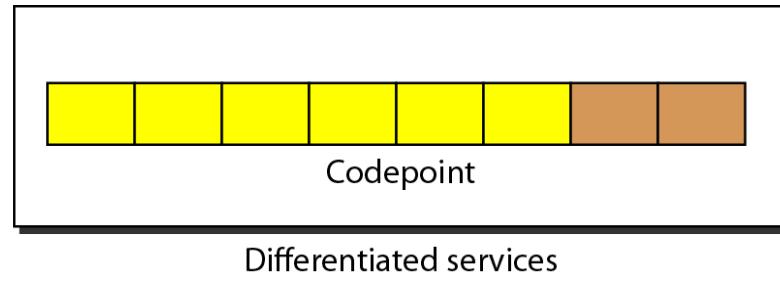
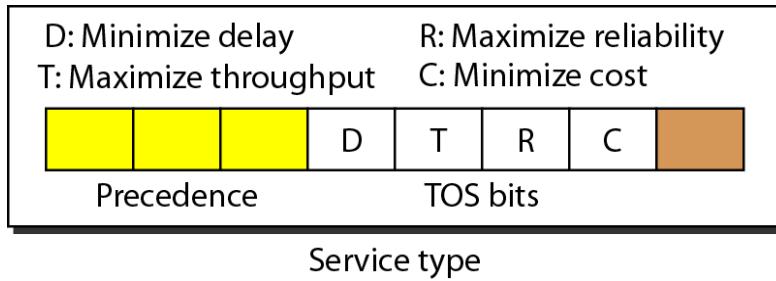
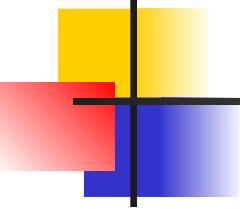


Figure 20.6 *Service type or differentiated services*





Note

The precedence subfield was part of version 4, but never used.

Table 20.1 *Types of service*

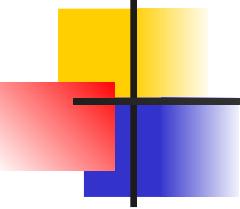
<i>TOS Bits</i>	<i>Description</i>
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Table 20.2 *Default types of service*

<i>Protocol</i>	<i>TOS Bits</i>	<i>Description</i>
ICMP	0000	Normal
BOOTP	0000	Normal
NNTP	0001	Minimize cost
IGP	0010	Maximize reliability
SNMP	0010	Maximize reliability
TELNET	1000	Minimize delay
FTP (data)	0100	Maximize throughput
FTP (control)	1000	Minimize delay
TFTP	1000	Minimize delay
SMTP (command)	1000	Minimize delay
SMTP (data)	0100	Maximize throughput
DNS (UDP query)	1000	Minimize delay
DNS (TCP query)	0000	Normal
DNS (zone)	0100	Maximize throughput

Table 20.3 *Values for codepoints*

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF



Note

The total length field defines the total length of the datagram including the header.

Figure 20.7 *Encapsulation of a small datagram in an Ethernet frame*

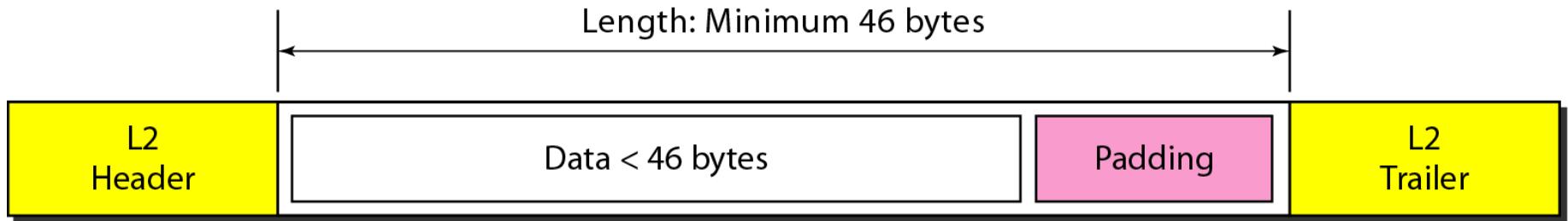


Figure 20.8 *Protocol field and encapsulated data*

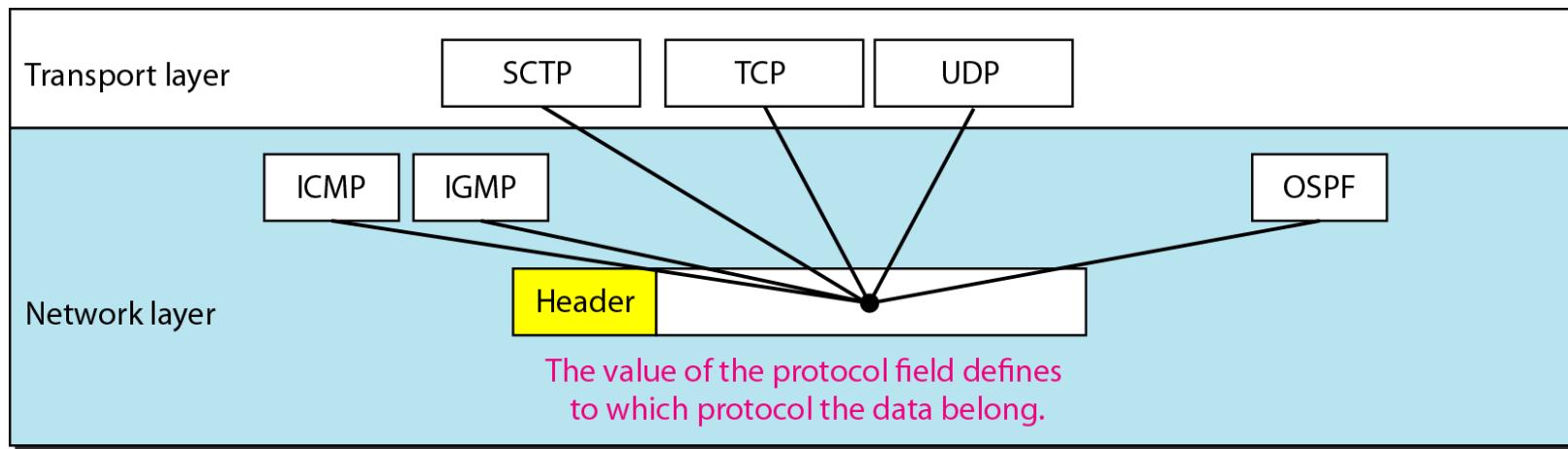
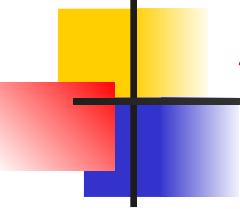


Table 20.4 *Protocol values*

<i>Value</i>	<i>Protocol</i>
1	ICMP
2	IGMP
6	TCP
17	UDP
89	OSPF



Example 20.1

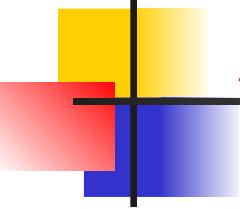
An IPv4 packet has arrived with the first 8 bits as shown:

01000010

The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits (0100) show the version, which is correct. The next 4 bits (0010) show an invalid header length ($2 \times 4 = 8$). The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

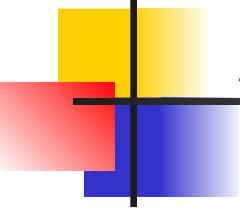


Example 20.2

In an IPv4 packet, the value of HLEN is 1000 in binary. How many bytes of options are being carried by this packet?

Solution

The HLEN value is 8, which means the total number of bytes in the header is 8×4 , or 32 bytes. The first 20 bytes are the base header, the next 12 bytes are the options.

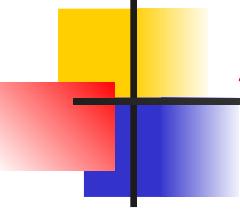


Example 20.3

In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is 0x0028. How many bytes of data are being carried by this packet?

Solution

The HLEN value is 5, which means the total number of bytes in the header is 5×4 , or 20 bytes (no options). The total length is 40 bytes, which means the packet is carrying 20 bytes of data ($40 - 20$).



Example 20.4

An IPv4 packet has arrived with the first few hexadecimal digits as shown.

0x45000028000100000102 ...

How many hops can this packet travel before being dropped? The data belong to what upper-layer protocol?

Solution

To find the time-to-live field, we skip 8 bytes. The time-to-live field is the ninth byte, which is 01. This means the packet can travel only one hop. The protocol field is the next byte (02), which means that the upper-layer protocol is IGMP.

Figure 20.9 Maximum transfer unit (MTU)

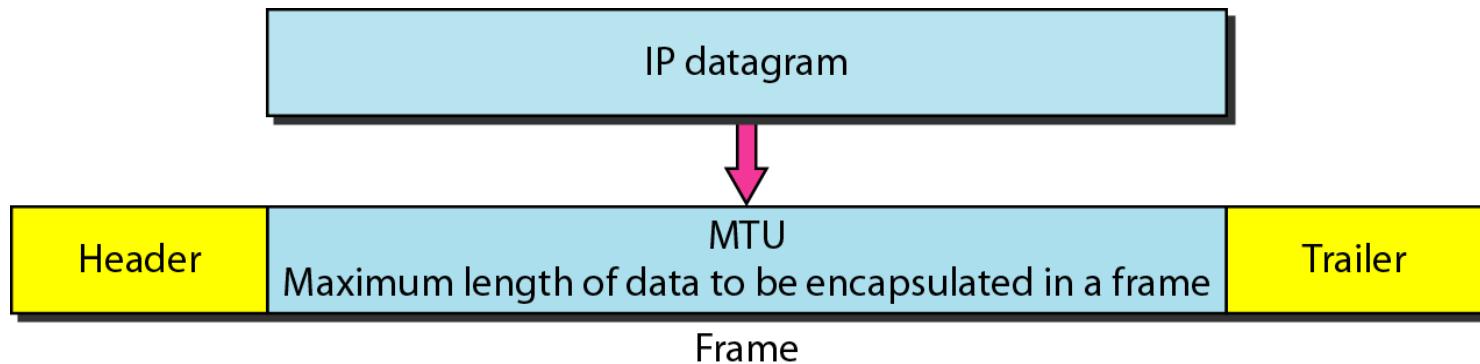


Table 20.5 *MTUs for some networks*

<i>Protocol</i>	<i>MTU</i>
Hyperchannel	65,535
Token Ring (16 Mbps)	17,914
Token Ring (4 Mbps)	4,464
FDDI	4,352
Ethernet	1,500
X.25	576
PPP	296

Figure 20.10 *Flags used in fragmentation*



Figure 20.11 Fragmentation example

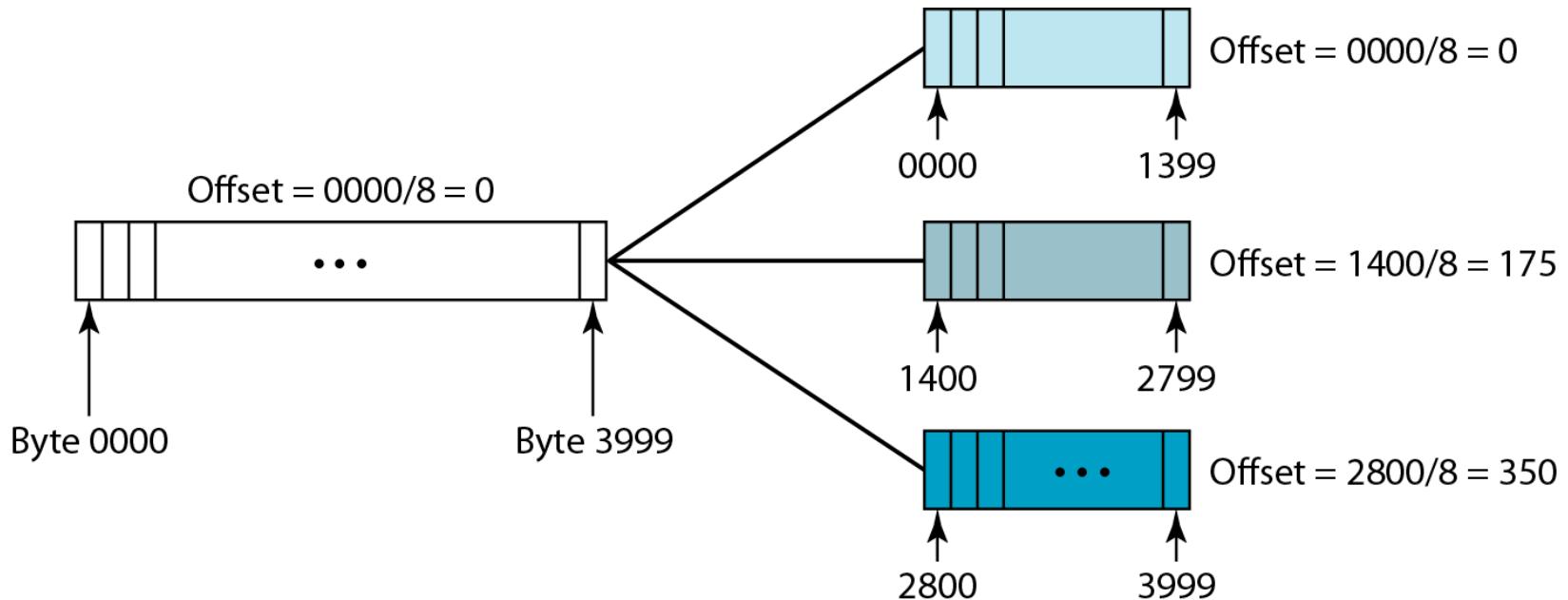
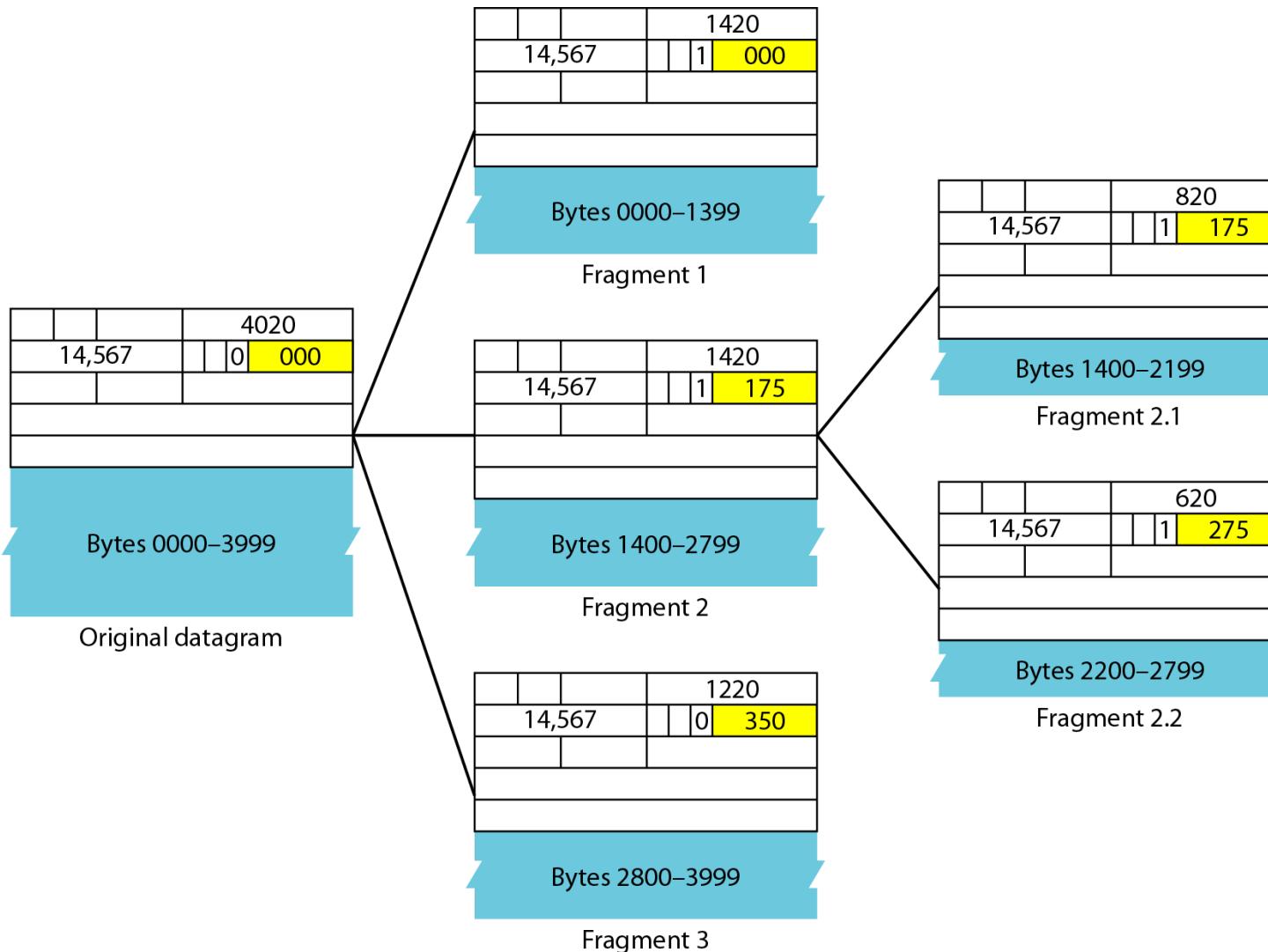
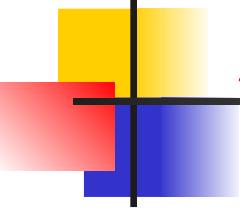


Figure 20.12 Detailed fragmentation example



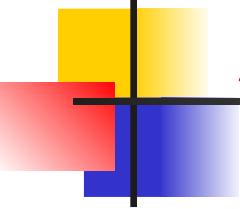


Example 20.5

A packet has arrived with an M bit value of 0. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 0, it means that there are no more fragments; the fragment is the last one. However, we cannot say if the original packet was fragmented or not. A non-fragmented packet is considered the last fragment.

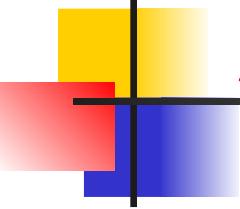


Example 20.6

A packet has arrived with an M bit value of 1. Is this the first fragment, the last fragment, or a middle fragment? Do we know if the packet was fragmented?

Solution

If the M bit is 1, it means that there is at least one more fragment. This fragment can be the first one or a middle one, but not the last one. We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

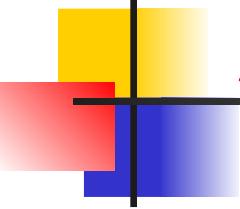


Example 20.7

A packet has arrived with an M bit value of 1 and a fragmentation offset value of 0. Is this the first fragment, the last fragment, or a middle fragment?

Solution

Because the M bit is 1, it is either the first fragment or a middle one. Because the offset value is 0, it is the first fragment.

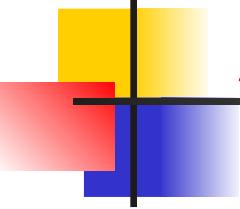


Example 20.8

A packet has arrived in which the offset value is 100. What is the number of the first byte? Do we know the number of the last byte?

Solution

To find the number of the first byte, we multiply the offset value by 8. This means that the first byte number is 800. We cannot determine the number of the last byte unless we know the length.

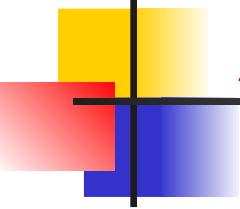


Example 20.9

A packet has arrived in which the offset value is 100, the value of HLEN is 5, and the value of the total length field is 100. What are the numbers of the first byte and the last byte?

Solution

The first byte number is $100 \times 8 = 800$. The total length is 100 bytes, and the header length is 20 bytes (5×4), which means that there are 80 bytes in this datagram. If the first byte number is 800, the last byte number must be 879.



Example 20.10

Figure 20.13 shows an example of a checksum calculation for an IPv4 header without options. The header is divided into 16-bit sections. All the sections are added and the sum is complemented. The result is inserted in the checksum field.

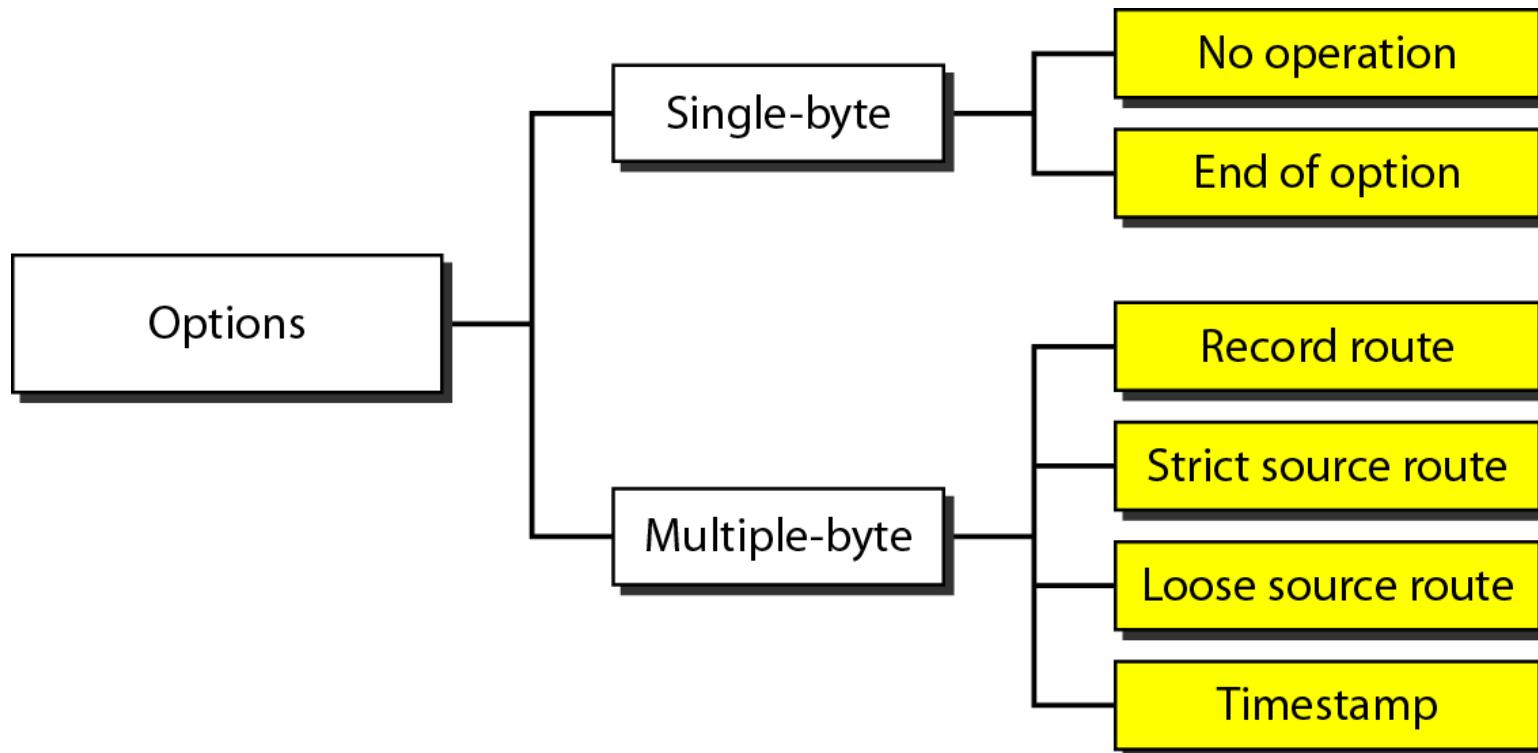
Figure 20.13 Example of checksum calculation in IPv4

4	5	0	28								
1			0	0							
4	17	0				↑					
10.12.14.5											
12.6.7.9											

4, 5, and 0 → 4 5 0 0
28 → 0 0 1 C
1 → 0 0 0 1
0 and 0 → 0 0 0 0
4 and 17 → 0 4 1 1
0 → 0 0 0 0
10.12 → 0 A 0 C
14.5 → 0 E 0 5
12.6 → 0 C 0 6
7.9 → 0 7 0 9

Sum → 7 4 4 E
Checksum → 8 B B 1

Figure 20.14 *Taxonomy of options in IPv4*



20-3 IPv6

The network layer protocol in the TCP/IP protocol suite is currently IPv4. Although IPv4 is well designed, data communication has evolved since the inception of IPv4 in the 1970s. IPv4 has some deficiencies that make it unsuitable for the fast-growing Internet.

Topics discussed in this section:

Advantages

Packet Format

Extension Headers

Figure 20.15 IPv6 datagram header and payload

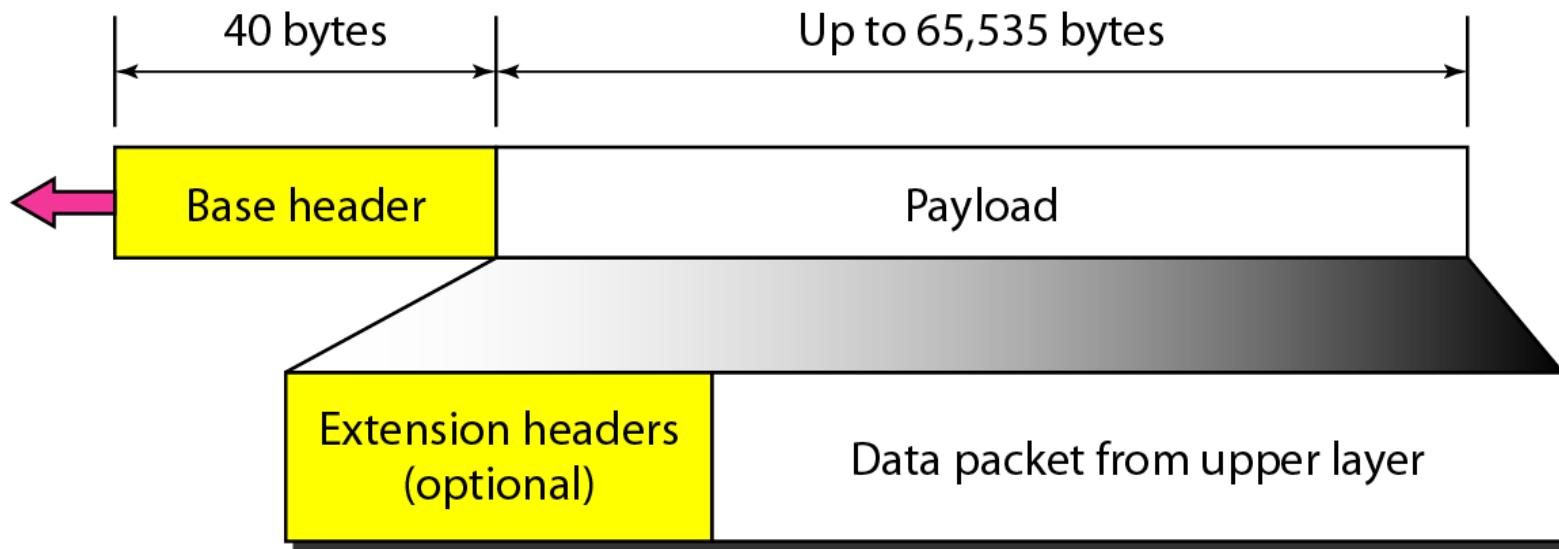


Figure 20.16 Format of an IPv6 datagram

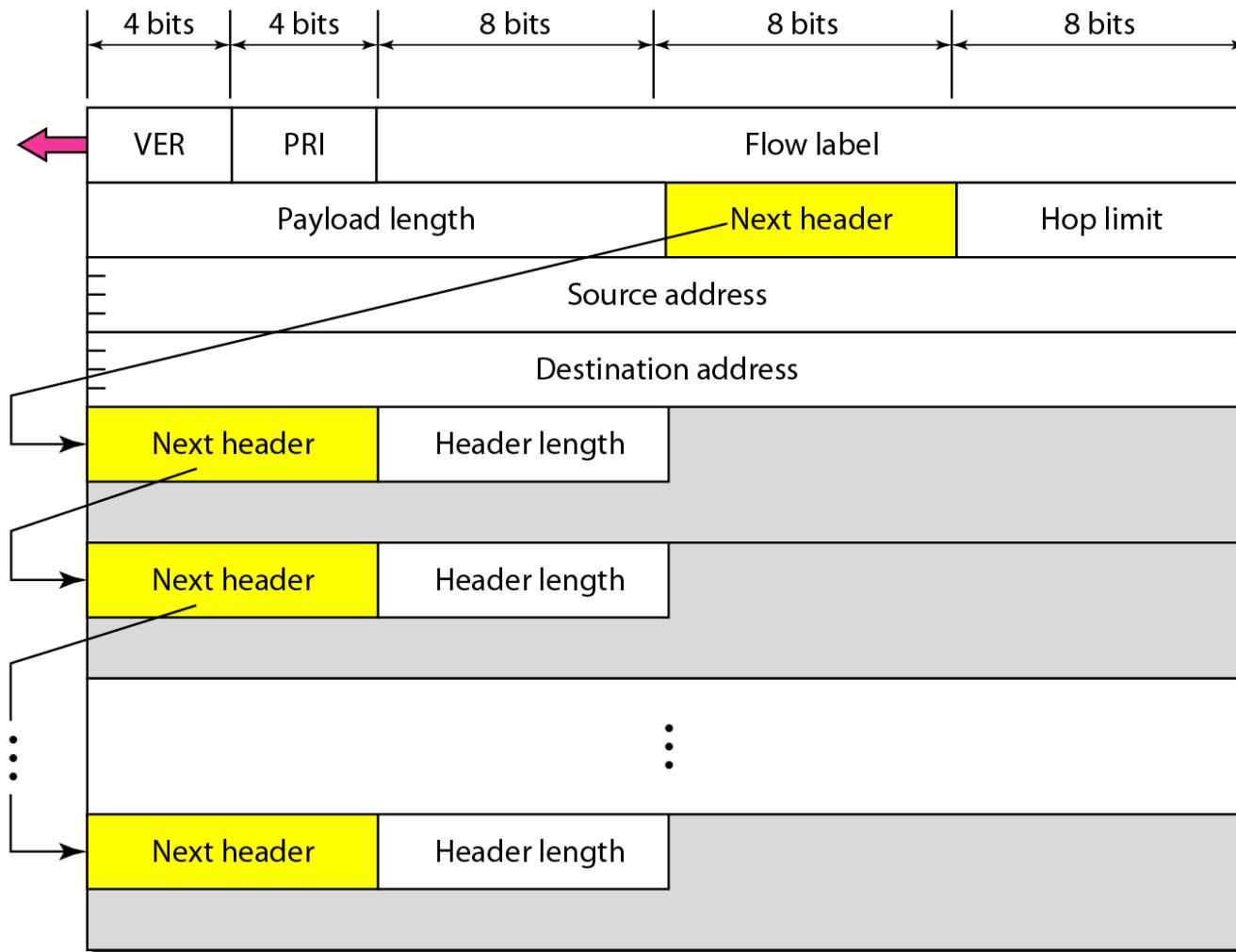


Table 20.6 *Next header codes for IPv6*

<i>Code</i>	<i>Next Header</i>
0	Hop-by-hop option
2	ICMP
6	TCP
17	UDP
43	Source routing
44	Fragmentation
50	Encrypted security payload
51	Authentication
59	Null (no next header)
60	Destination option

Table 20.7 *Priorities for congestion-controlled traffic*

<i>Priority</i>	<i>Meaning</i>
0	No specific traffic
1	Background data
2	Unattended data traffic
3	Reserved
4	Attended bulk data traffic
5	Reserved
6	Interactive traffic
7	Control traffic

Table 20.8 *Priorities for noncongestion-controlled traffic*

<i>Priority</i>	<i>Meaning</i>
8	Data with greatest redundancy
...	...
15	Data with least redundancy

Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

<i>Comparison</i>
1. The header length field is eliminated in IPv6 because the length of the header is fixed in this version.
2. The service type field is eliminated in IPv6. The priority and flow label fields together take over the function of the service type field.
3. The total length field is eliminated in IPv6 and replaced by the payload length field.
4. The identification, flag, and offset fields are eliminated from the base header in IPv6. They are included in the fragmentation extension header.
5. The TTL field is called hop limit in IPv6.
6. The protocol field is replaced by the next header field.
7. The header checksum is eliminated because the checksum is provided by upper-layer protocols; it is therefore not needed at this level.
8. The option fields in IPv4 are implemented as extension headers in IPv6.

Figure 20.17 *Extension header types*

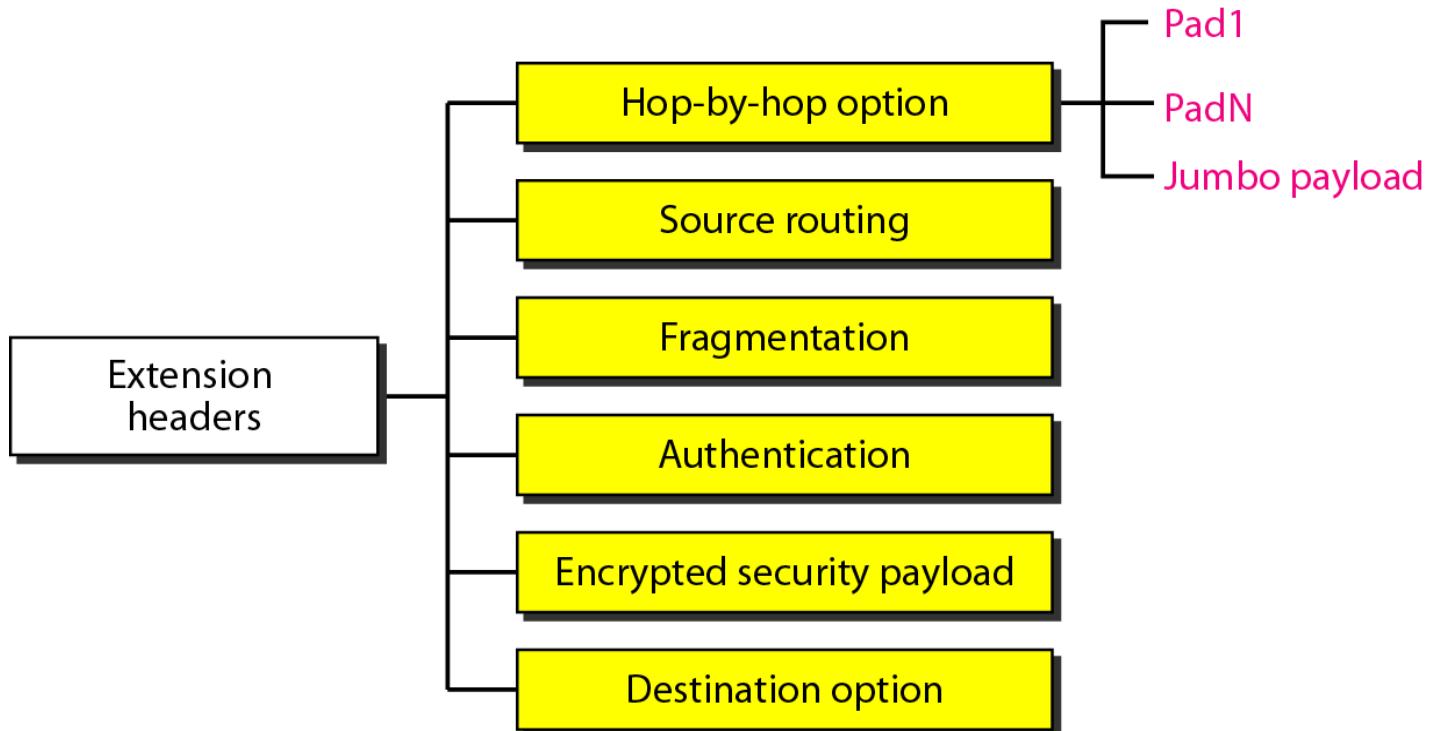


Table 20.10 *Comparison between IPv4 options and IPv6 extension headers*

<i>Comparison</i>
1. The no-operation and end-of-option options in IPv4 are replaced by Pad1 and PadN options in IPv6.
2. The record route option is not implemented in IPv6 because it was not used.
3. The timestamp option is not implemented because it was not used.
4. The source route option is called the source route extension header in IPv6.
5. The fragmentation fields in the base header section of IPv4 have moved to the fragmentation extension header in IPv6.
6. The authentication extension header is new in IPv6.
7. The encrypted security payload extension header is new in IPv6.

20-4 TRANSITION FROM IPv4 TO IPv6

Because of the huge number of systems on the Internet, the transition from IPv4 to IPv6 cannot happen suddenly. It takes a considerable amount of time before every system in the Internet can move from IPv4 to IPv6. The transition must be smooth to prevent any problems between IPv4 and IPv6 systems.

Topics discussed in this section:

Dual Stack

Tunneling

Header Translation

Figure 20.18 *Three transition strategies*

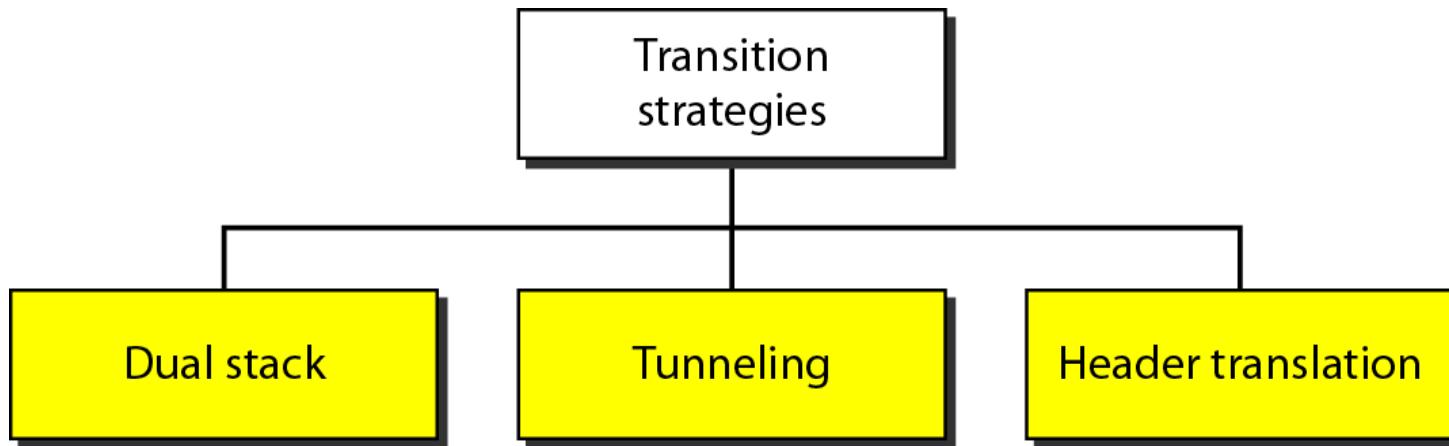


Figure 20.19 Dual stack

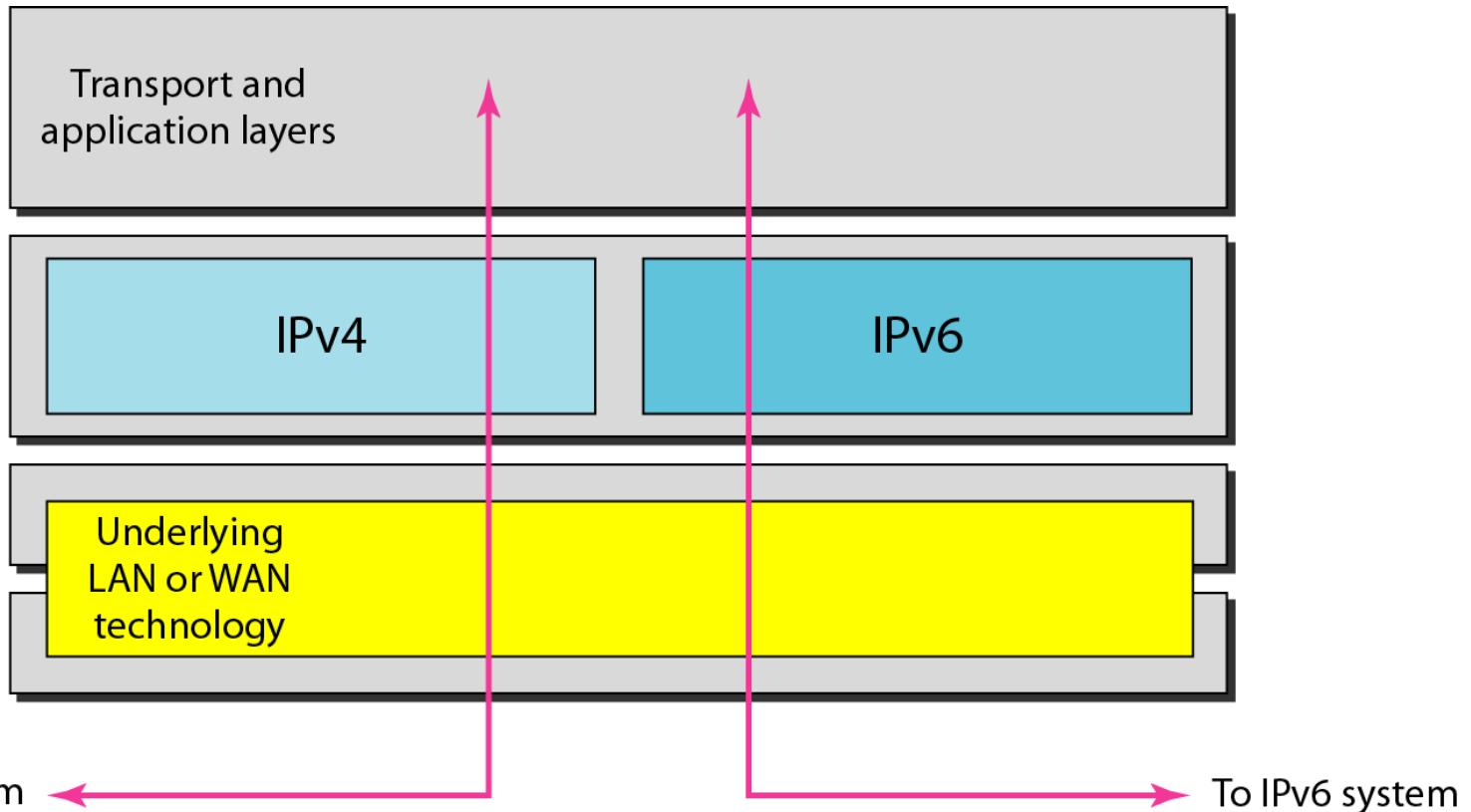


Figure 20.20 Tunneling strategy

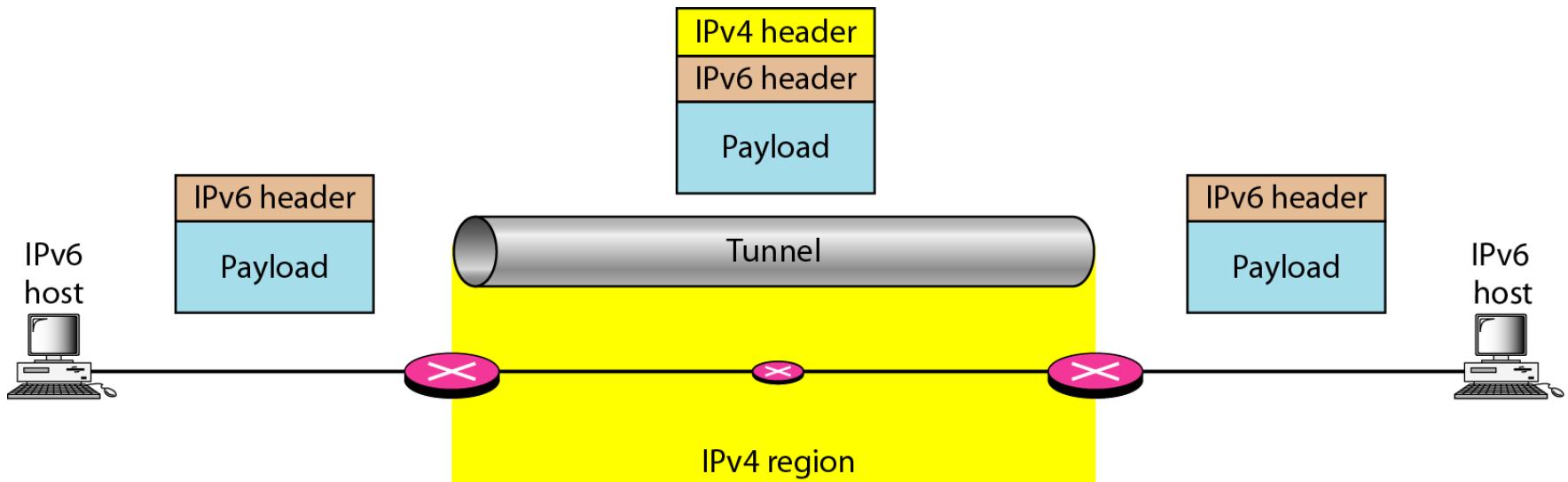


Figure 20.21 Header translation strategy

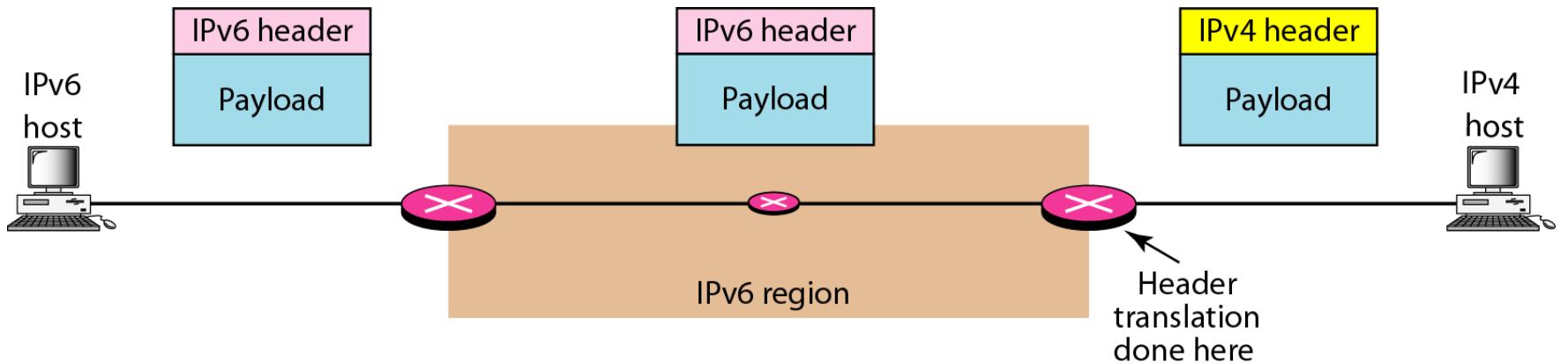


Table 20.11 *Header translation*

<i>Header Translation Procedure</i>
1. The IPv6 mapped address is changed to an IPv4 address by extracting the rightmost 32 bits.
2. The value of the IPv6 priority field is discarded.
3. The type of service field in IPv4 is set to zero.
4. The checksum for IPv4 is calculated and inserted in the corresponding field.
5. The IPv6 flow label is ignored.
6. Compatible extension headers are converted to options and inserted in the IPv4 header. Some may have to be dropped.
7. The length of IPv4 header is calculated and inserted into the corresponding field.
8. The total length of the IPv4 packet is calculated and inserted in the corresponding field.



Chapter 21

Network Layer: Address Mapping, Error Reporting, and Multicasting

21-1 ADDRESS MAPPING

*The delivery of a packet to a host or a router requires two levels of addressing: **logical** and **physical**. We need to be able to map a logical address to its corresponding physical address and vice versa. This can be done by using either static or dynamic mapping.*

Topics discussed in this section:

Mapping Logical to Physical Address

Mapping Physical to Logical Address

Figure 21.1 ARP operation

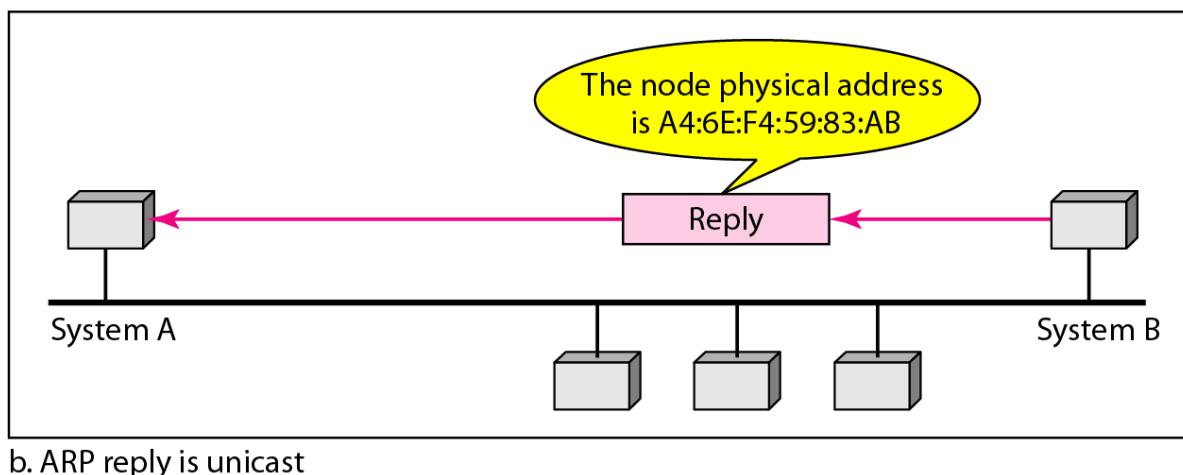
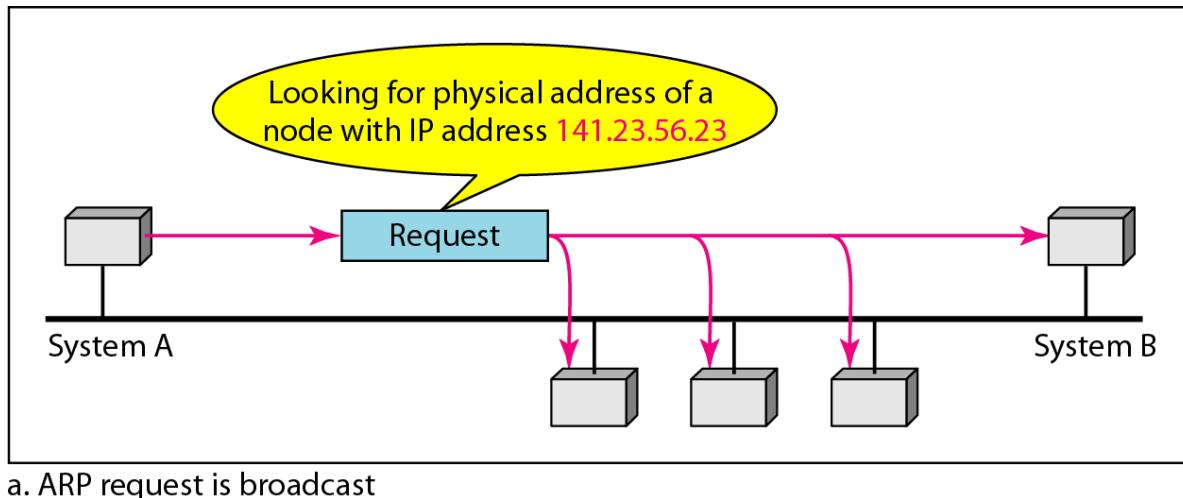


Figure 21.2 ARP packet

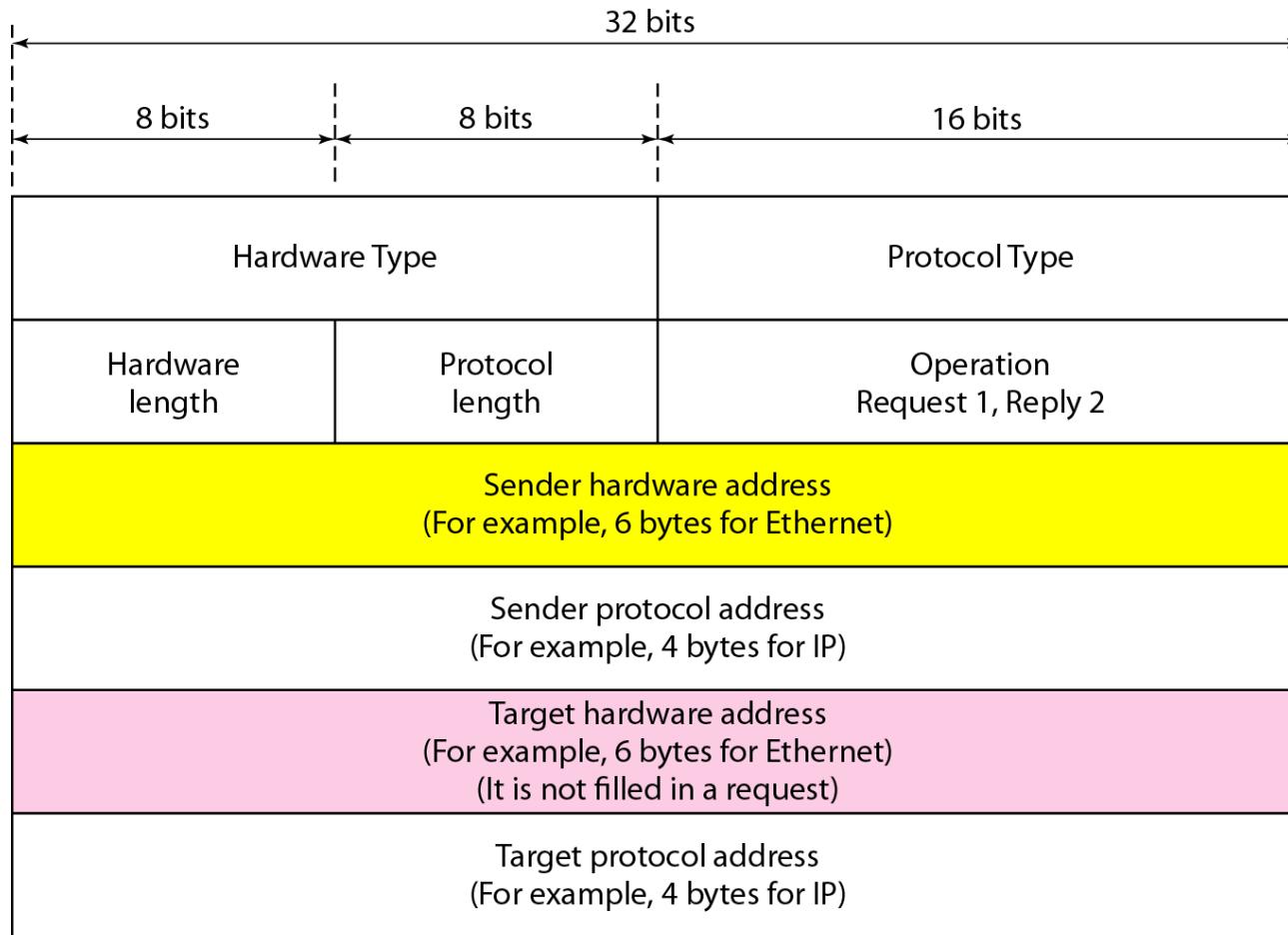


Figure 21.3 *Encapsulation of ARP packet*

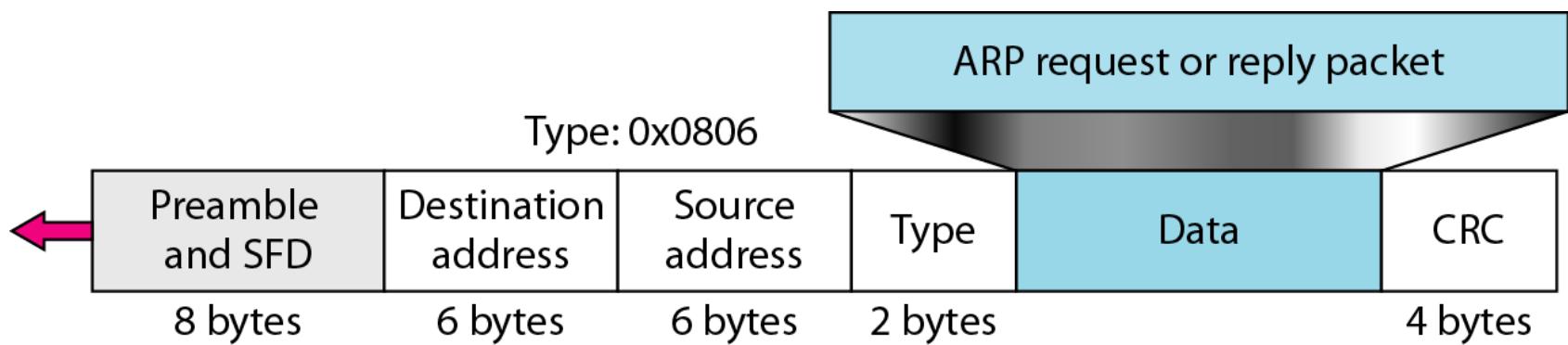
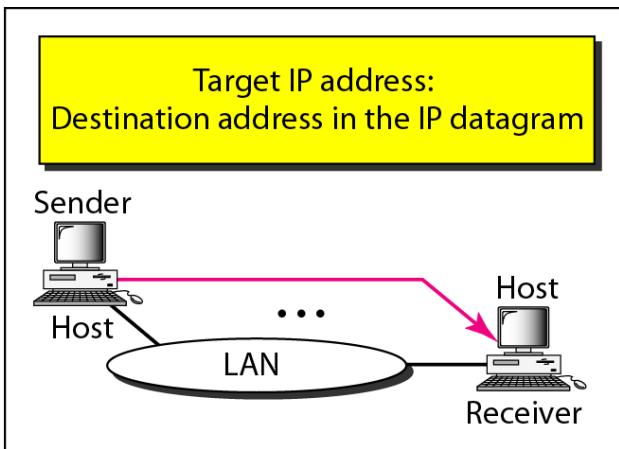
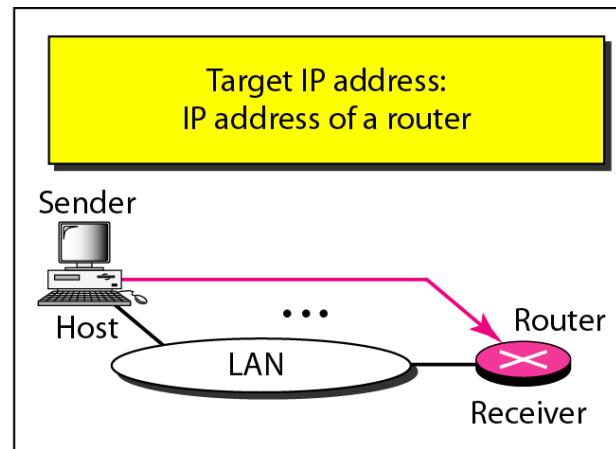


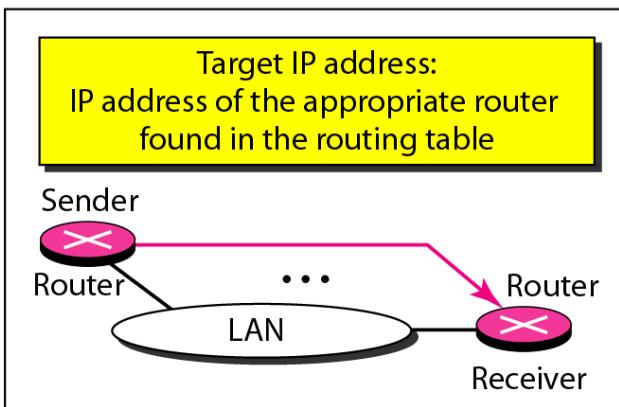
Figure 21.4 Four cases using ARP



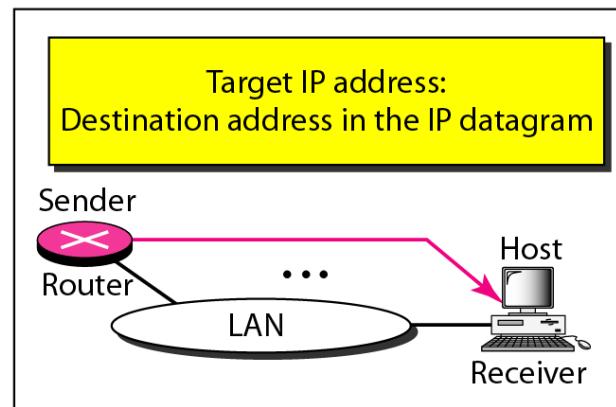
Case 1. A host has a packet to send to another host on the same network.



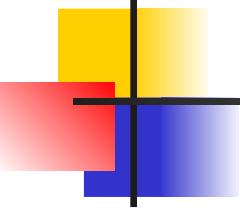
Case 2. A host wants to send a packet to another host on another network.
It must first be delivered to a router.



Case 3. A router receives a packet to be sent to a host on another network. It must first be delivered to the appropriate router.



Case 4. A router receives a packet to be sent to a host on the same network.



Note

**An ARP request is broadcast;
an ARP reply is unicast.**

Example 21.1

A host with IP address 130.23.43.20 and physical address B2:34:55:10:22:10 has a packet to send to another host with IP address 130.23.43.25 and physical address A4:6E:F4:59:83:AB. The two hosts are on the same Ethernet network. Show the ARP request and reply packets encapsulated in Ethernet frames.

Solution

Figure 21.5 shows the ARP request and reply packets. Note that the ARP data field in this case is 28 bytes, and that the individual addresses do not fit in the 4-byte boundary. That is why we do not show the regular 4-byte boundaries for these addresses.

Figure 21.5 Example 21.1, an ARP request and reply

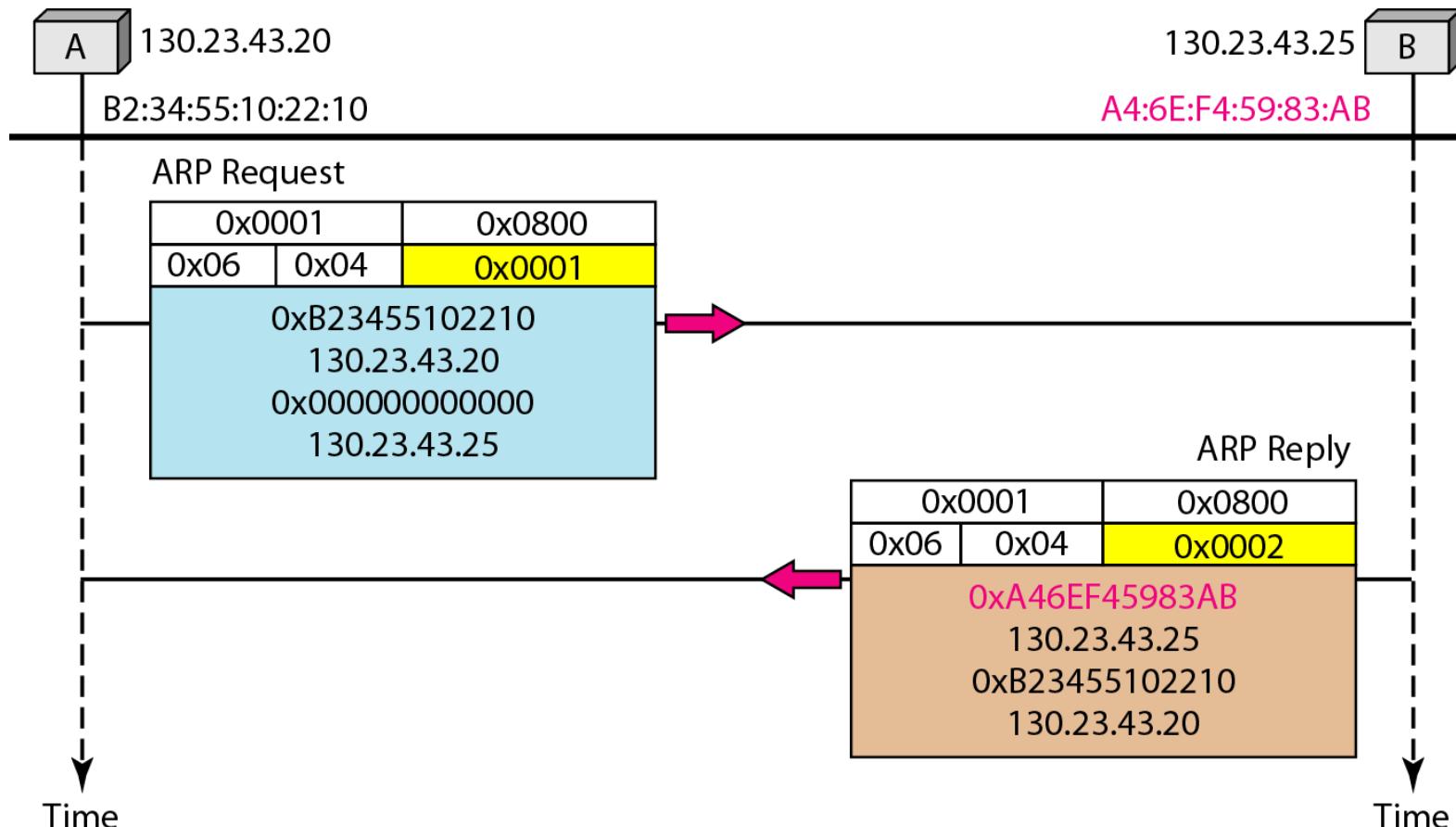


Figure 21.6 Proxy ARP

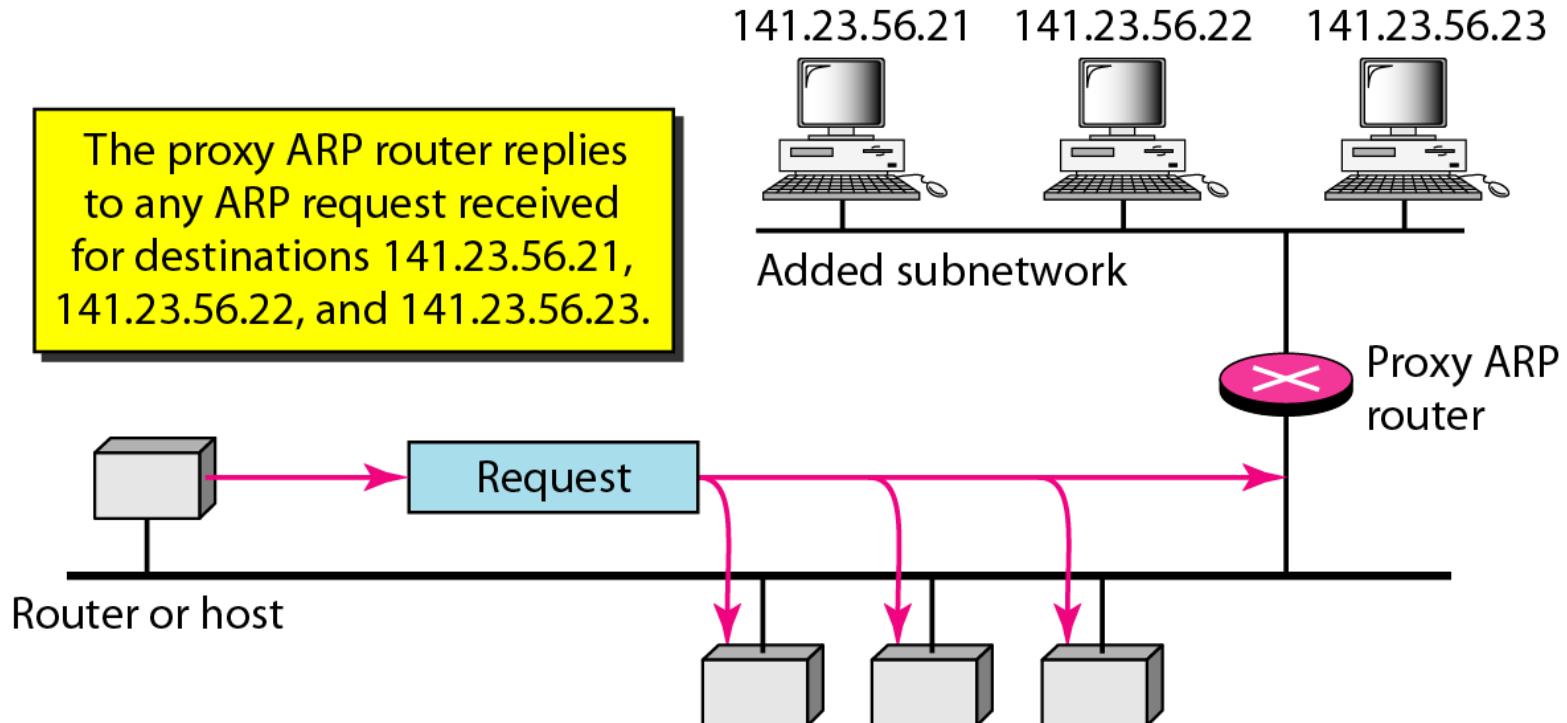
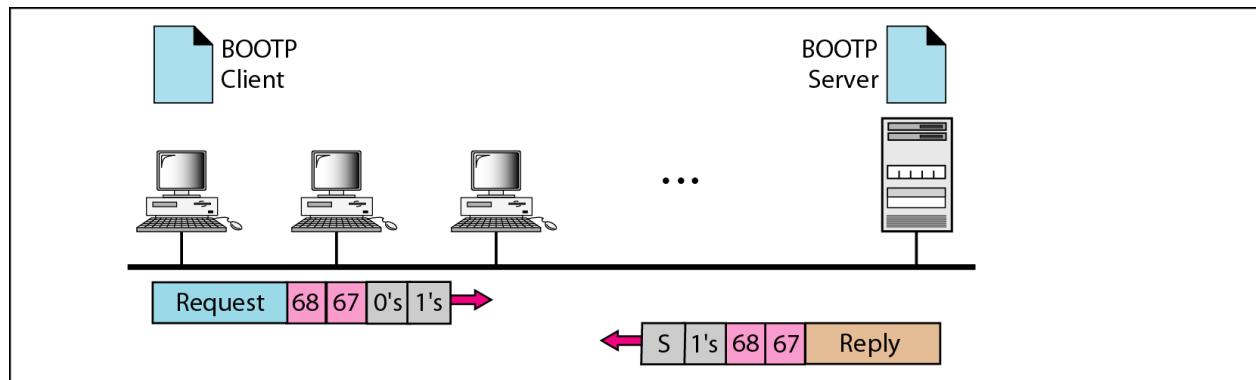
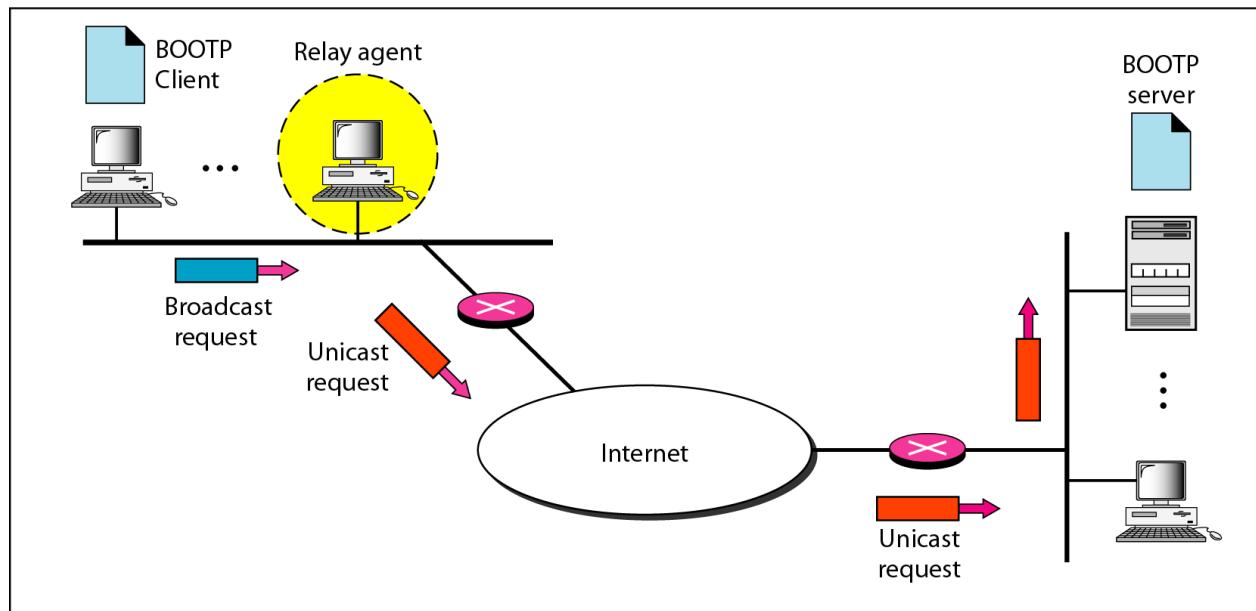


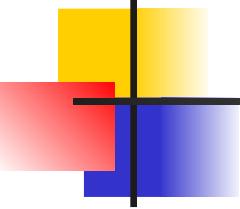
Figure 21.7 *BOOTP client and server on the same and different networks*



a. Client and server on the same network



b. Client and server on different networks



Note

DHCP provides static and dynamic address allocation that can be manual or automatic.

21-2 ICMP

The IP protocol has no error-reporting or error-correcting mechanism. The IP protocol also lacks a mechanism for host and management queries. The Internet Control Message Protocol (ICMP) has been designed to compensate for the above two deficiencies. It is a companion to the IP protocol.

Topics discussed in this section:

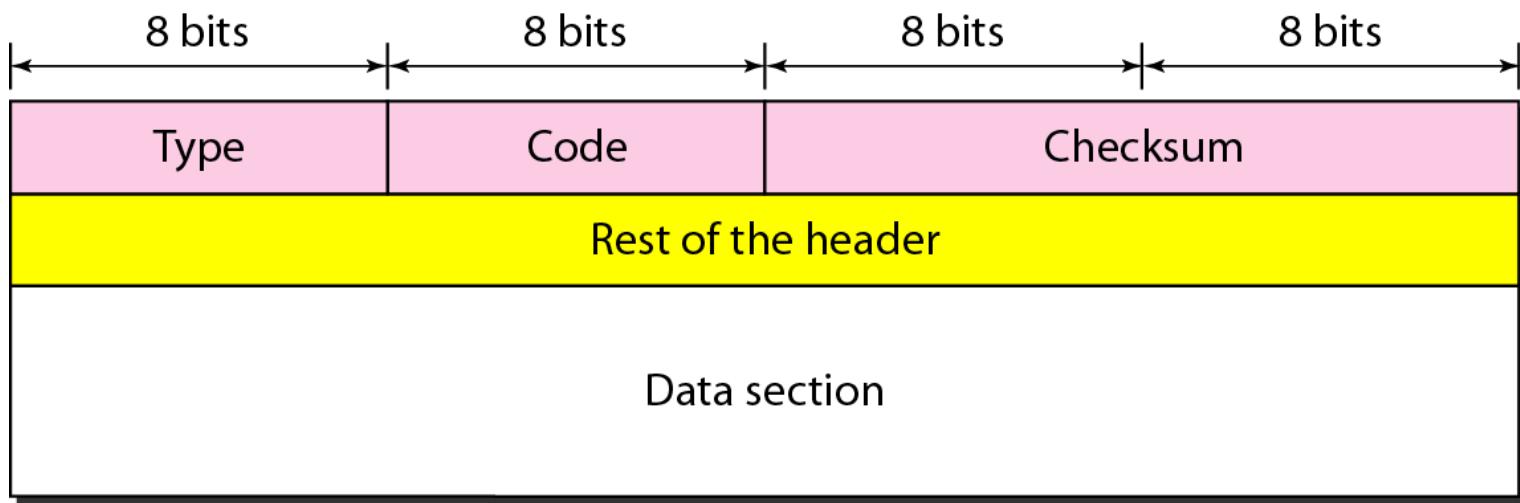
Types of Messages

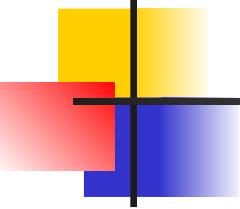
Message Format

Error Reporting and Query

Debugging Tools

Figure 21.8 *General format of ICMP messages*

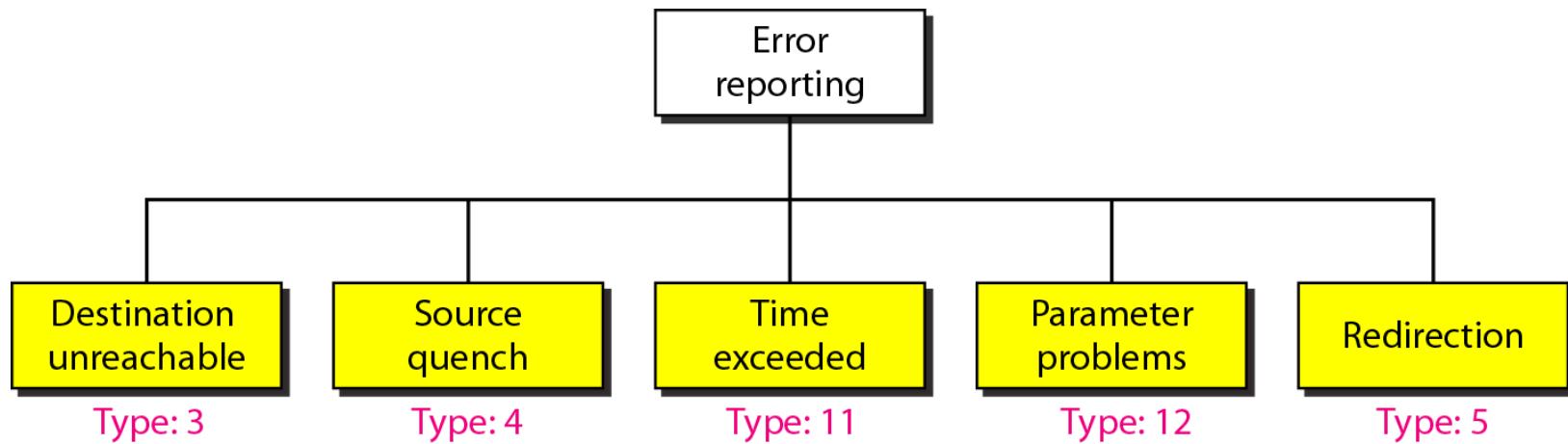


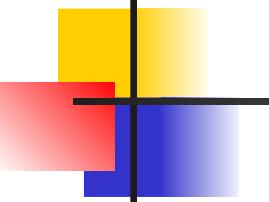


Note

ICMP always reports error messages to the original source.

Figure 21.9 *Error-reporting messages*





Note

Important points about ICMP error messages:

- No ICMP error message will be generated in response to a datagram carrying an ICMP error message.
- No ICMP error message will be generated for a fragmented datagram that is not the first fragment.
- No ICMP error message will be generated for a datagram having a multicast address.
- No ICMP error message will be generated for a datagram having a special address such as 127.0.0.0 or 0.0.0.0.

Figure 21.10 *Contents of data field for the error messages*

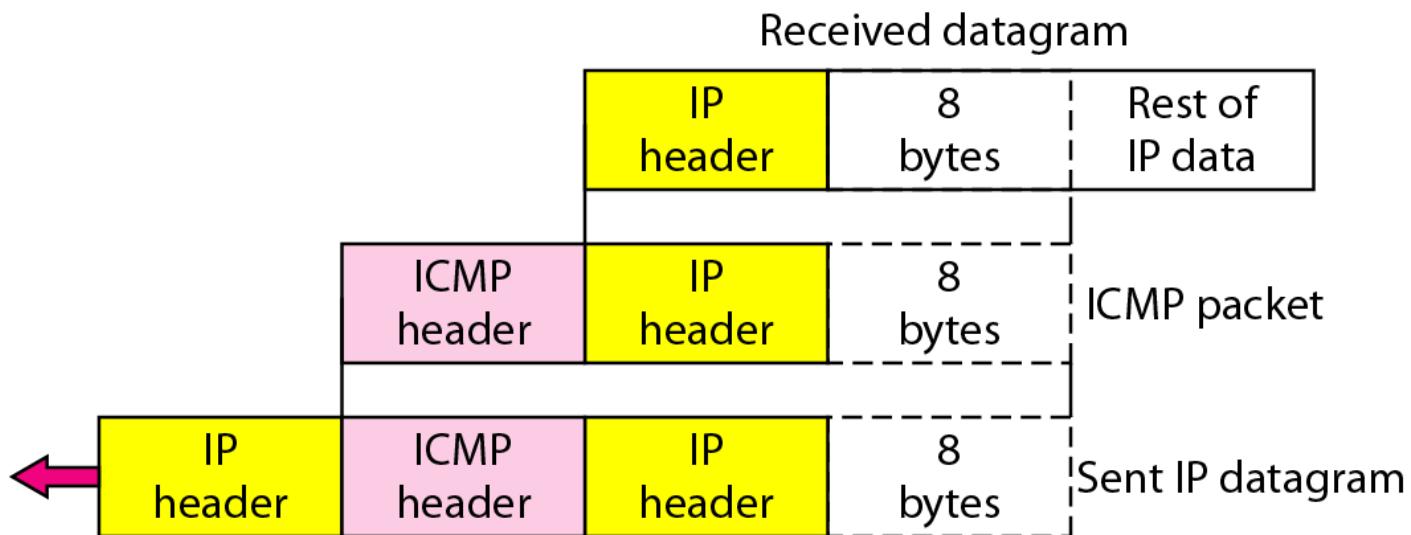


Figure 21.11 *Redirection concept*

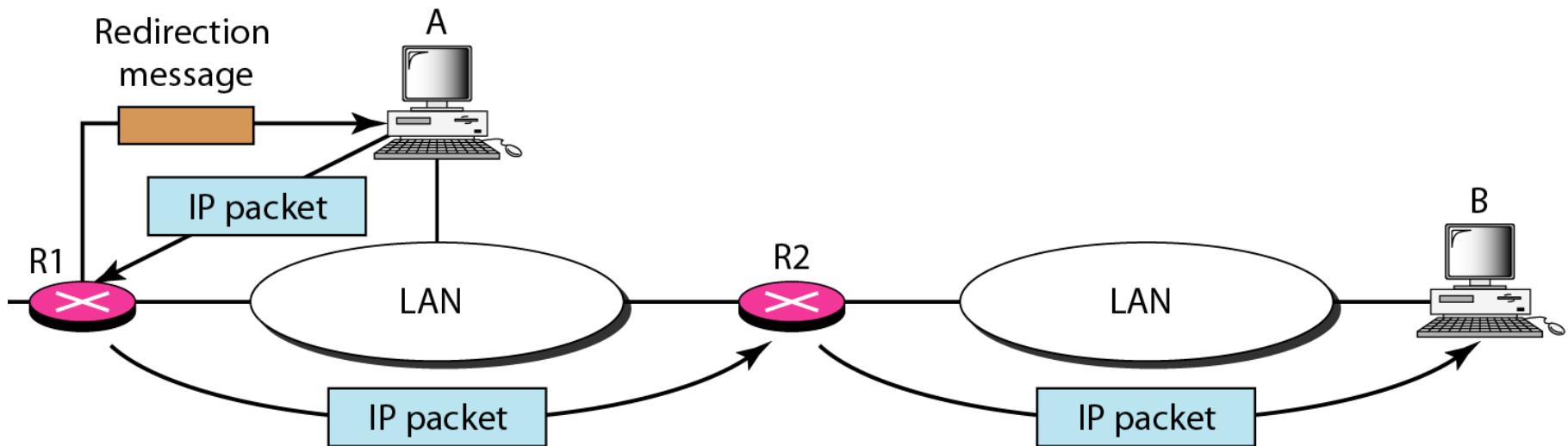


Figure 21.12 *Query messages*

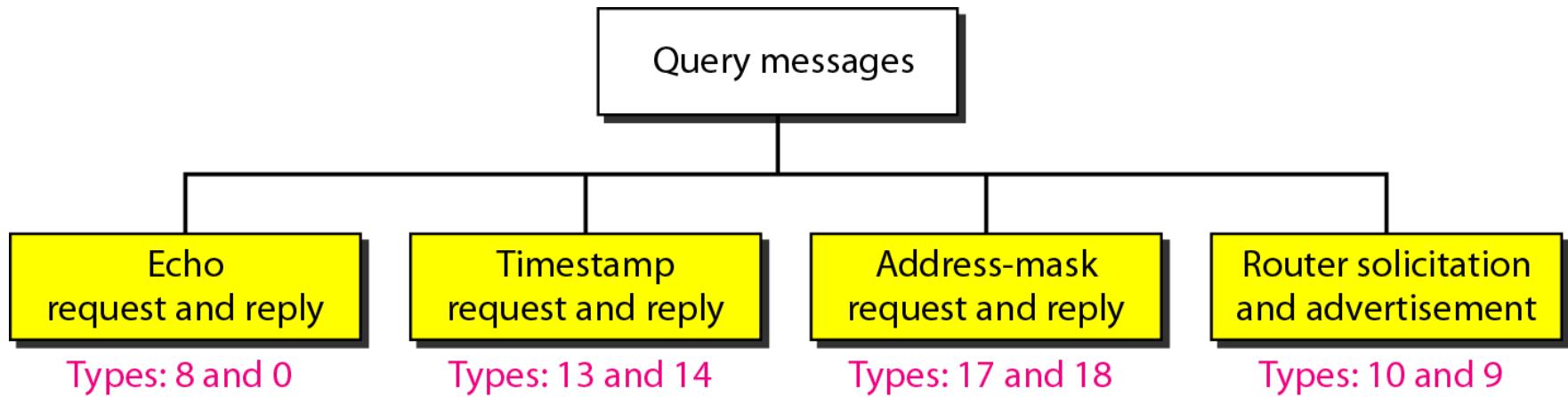
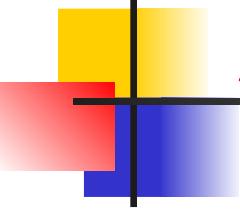


Figure 21.13 *Encapsulation of ICMP query messages*

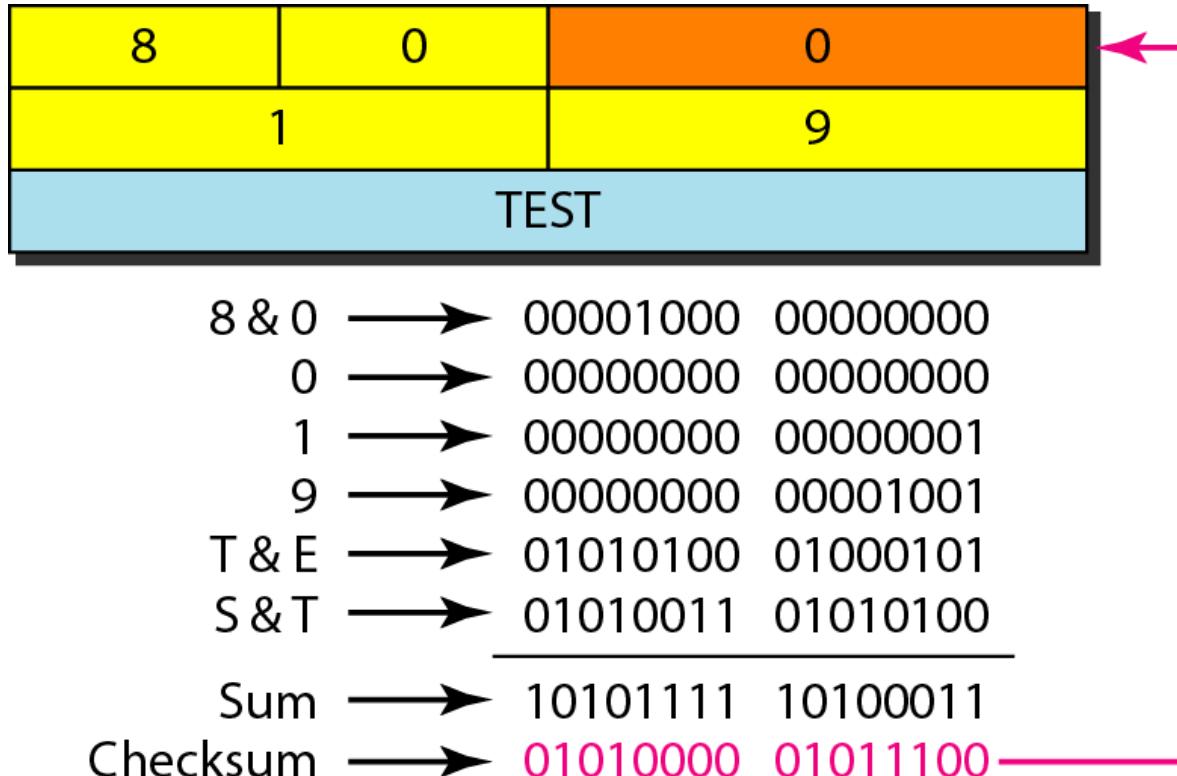


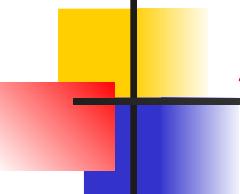


Example 21.2

Figure 21.14 shows an example of checksum calculation for a simple echo-request message. We randomly chose the identifier to be 1 and the sequence number to be 9. The message is divided into 16-bit (2-byte) words. The words are added and the sum is complemented. Now the sender can put this value in the checksum field.

Figure 21.14 Example of checksum calculation





Example 21.3

We use the ping program to test the server fhda.edu. The result is shown on the next slide. The ping program sends messages with sequence numbers starting from 0. For each probe it gives us the RTT time. The TTL (time to live) field in the IP datagram that encapsulates an ICMP message has been set to 62. At the beginning, ping defines the number of data bytes as 56 and the total number of bytes as 84. It is obvious that if we add 8 bytes of ICMP header and 20 bytes of IP header to 56, the result is 84. However, note that in each probe ping defines the number of bytes as 64. This is the total number of bytes in the ICMP packet ($56 + 8$).

Example 21.3 (continued)

```
$ ping fhda.edu
```

PING fhda.edu (153.18.8.1) 56 (84) bytes of data.

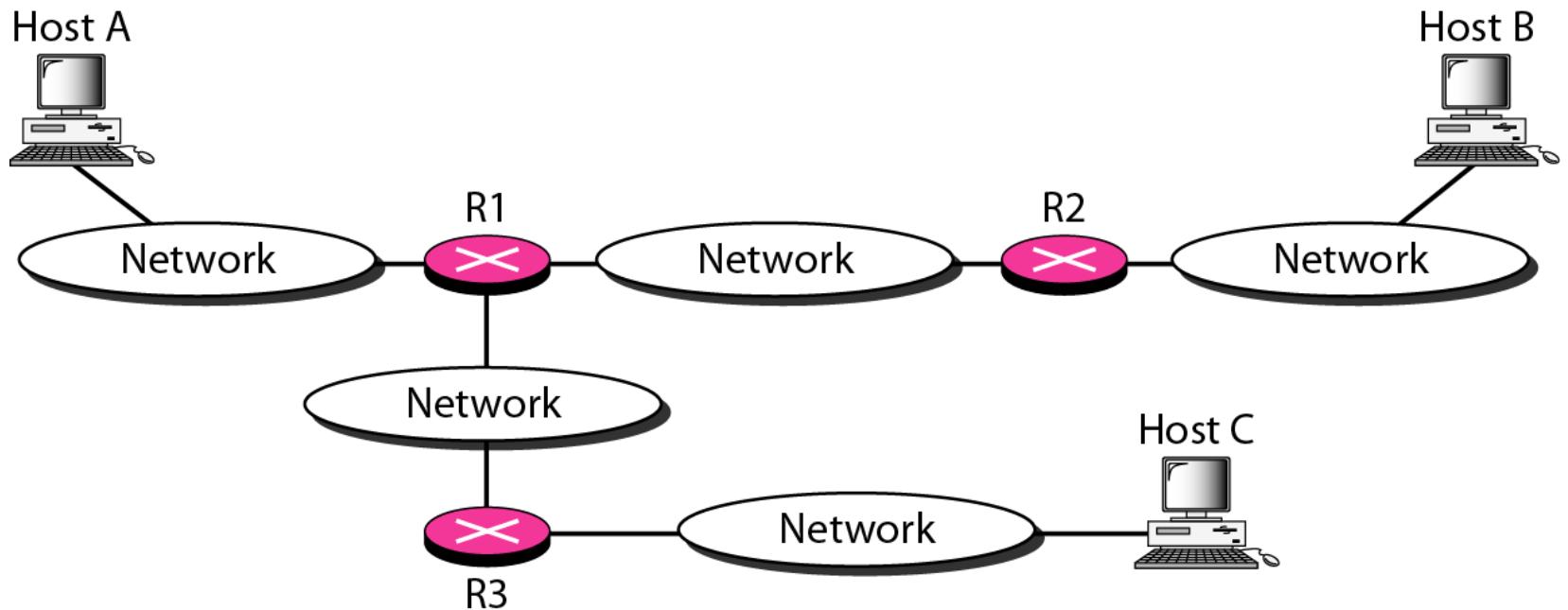
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=0	ttl=62	time=1.91 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=1	ttl=62	time=2.04 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=2	ttl=62	time=1.90 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=3	ttl=62	time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=4	ttl=62	time=1.93 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=5	ttl=62	time=2.00 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=6	ttl=62	time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=7	ttl=62	time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=8	ttl=62	time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=9	ttl=62	time=1.89 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=10	ttl=62	time=1.98 ms

--- fhda.edu ping statistics ---

11 packets transmitted, 11 received, 0% packet loss, time 10103ms

rtt min/avg/max = 1.899/1.955/2.041 ms

Figure 21.15 *The traceroute program operation*

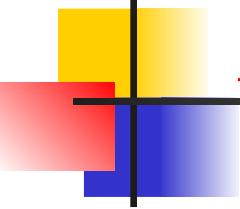


Example 21.4

We use the traceroute program to find the route from the computer voyager.deanza.edu to the server fhda.edu. The following shows the result:

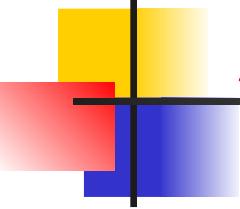
```
$ traceroute fhda.edu
traceroute to fhda.edu      (153.18.8.1), 30 hops max, 38 byte packets
 1 Dcore.fhda.edu          (153.18.31.254)   0.995 ms   0.899 ms   0.878 ms
 2 Dbackup.fhda.edu        (153.18.251.4)    1.039 ms   1.064 ms   1.083 ms
 3 tiptoe.fhda.edu         (153.18.8.1)       1.797 ms   1.642 ms   1.757 ms
```

The unnumbered line after the command shows that the destination is 153.18.8.1. The packet contains 38 bytes: 20 bytes of IP header, 8 bytes of UDP header, and 10 bytes of application data. The application data are used by traceroute to keep track of the packets.



Example 21.4 (continued)

The first line shows the first router visited. The router is named Dcore.fhda.edu with IP address 153.18.31.254. The first round-trip time was 0.995 ms, the second was 0.899 ms, and the third was 0.878 ms. The second line shows the second router visited. The router is named Dbackup.fhda.edu with IP address 153.18.251.4. The three round-trip times are also shown. The third line shows the destination host. We know that this is the destination host because there are no more lines. The destination host is the server fhda.edu, but it is named tiptoe.fhda.edu with the IP address 153.18.8.1. The three round-trip times are also shown.



Example 21.5

In this example, we trace a longer route, the route to xerox.com (see next slide). Here there are 17 hops between source and destination. Note that some round-trip times look unusual. It could be that a router was too busy to process the packet immediately.

Example 21.5 (continued)

```
$ traceroute xerox.com
```

traceroute to xerox.com (13.1.64.93), 30 hops max, 38 byte packets

1	Dcore.fhda.edu	(153.18.31.254)	0.622 ms	0.891 ms	0.875 ms
2	Ddmz.fhda.edu	(153.18.251.40)	2.132 ms	2.266 ms	2.094 ms
3	Cinic.fhda.edu	(153.18.253.126)	2.110 ms	2.145 ms	1.763 ms
4	cenic.net	(137.164.32.140)	3.069 ms	2.875 ms	2.930 ms
5	cenic.net	(137.164.22.31)	4.205 ms	4.870 ms	4.197 ms
...
14	snfc21.pbi.net	(151.164.191.49)	7.656 ms	7.129 ms	6.866 ms
15	sbcglobal.net	(151.164.243.58)	7.844 ms	7.545 ms	7.353 ms
16	pacbell.net	(209.232.138.114)	9.857 ms	9.535 ms	9.603 ms
17	209.233.48.223	(209.233.48.223)	10.634 ms	10.771 ms	10.592 ms
18	alpha.Xerox.COM	(13.1.64.93)	11.172 ms	11.048 ms	10.922 ms

21-3 IGMP

The IP protocol can be involved in two types of communication: unicasting and multicasting. The Internet Group Management Protocol (IGMP) is one of the necessary, but not sufficient, protocols that is involved in multicasting. IGMP is a companion to the IP protocol.

Topics discussed in this section:

Group Management

IGMP Messages and IGMP Operation

Encapsulation

Netstat Utility

Figure 21.16 *IGMP message types*

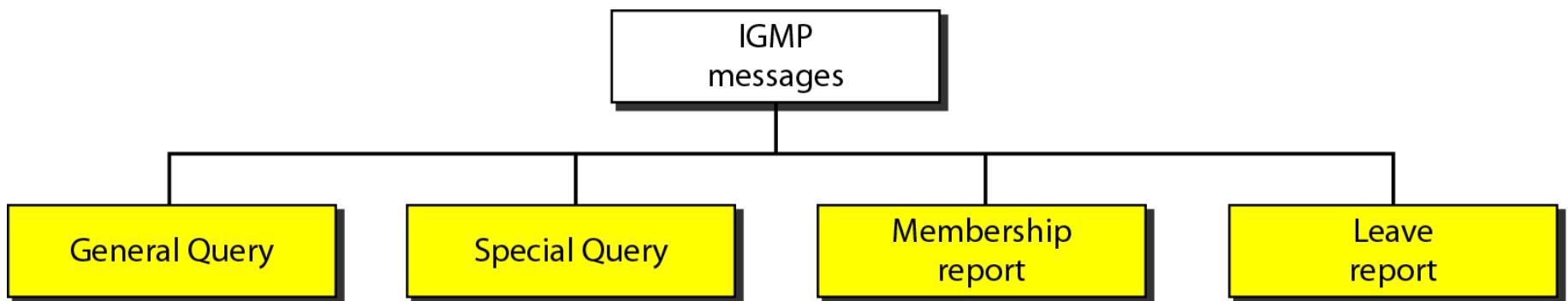


Figure 21.17 *IGMP message format*

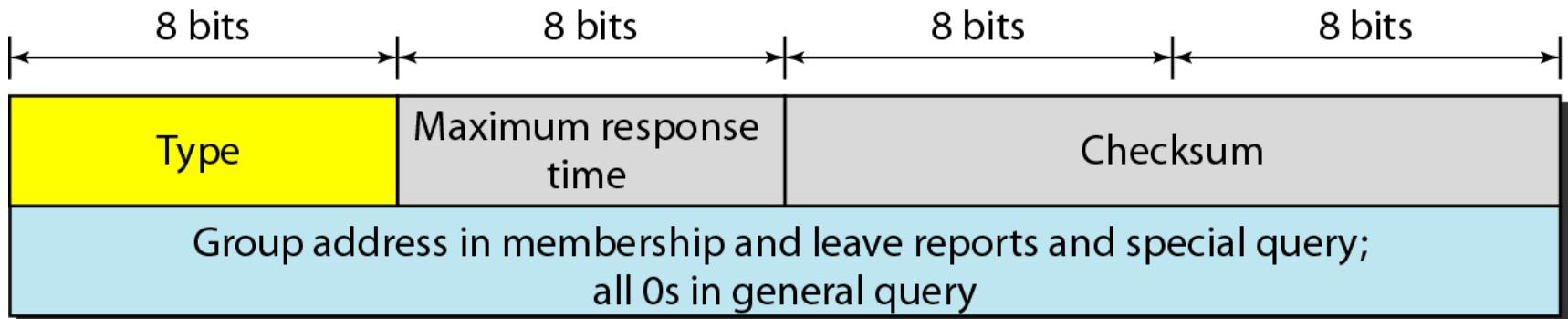
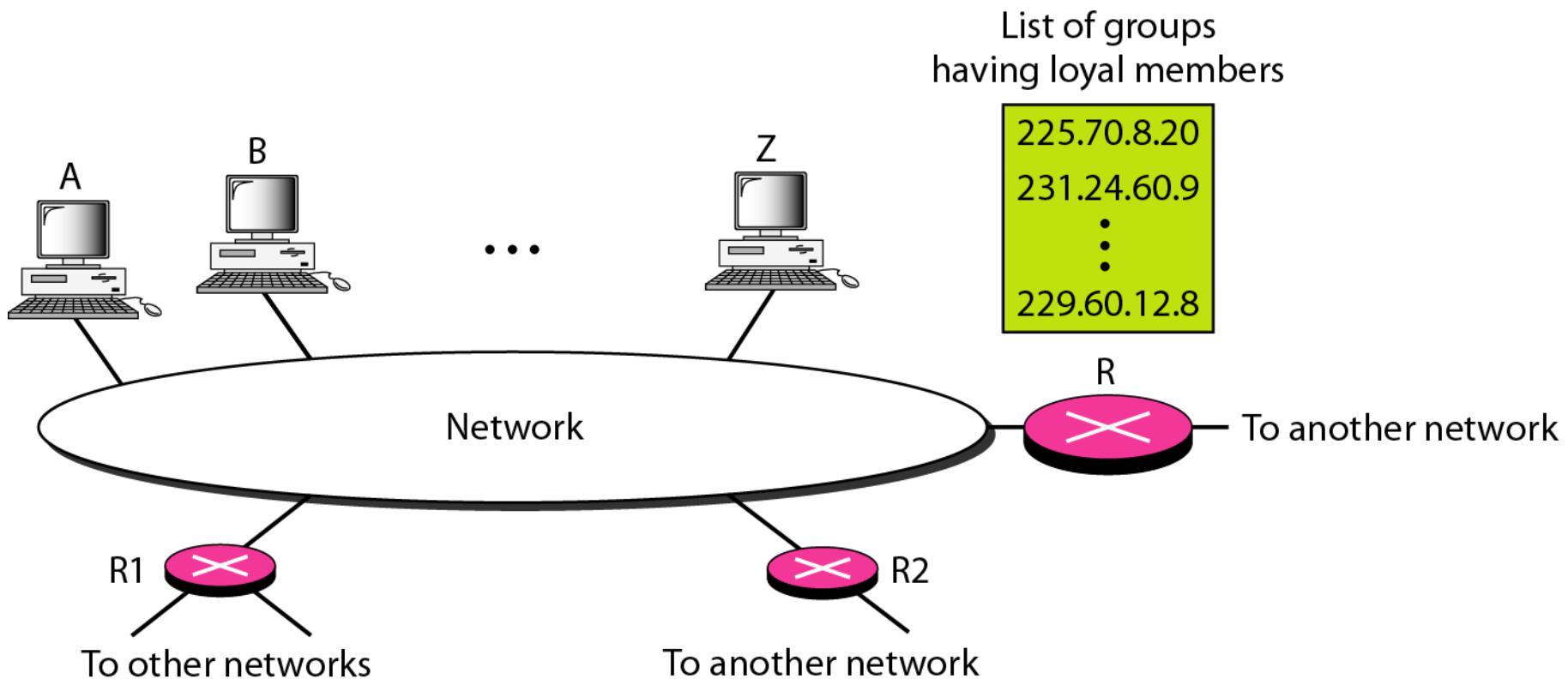
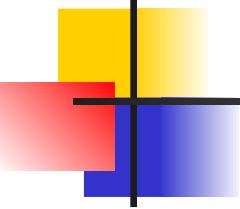


Table 21.1 IGMP type field

Type	Value
General or special query	0x11 or 00010001
Membership report	0x16 or 00010110
Leave report	0x17 or 00010111

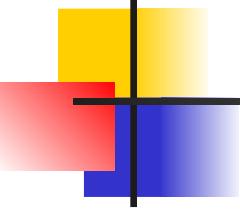
Figure 21.18 IGMP operation





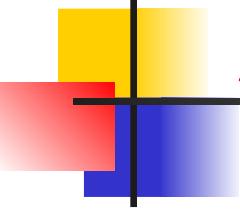
Note

In IGMP, a membership report is sent twice, one after the other.



Note

The general query message does not define a particular group.



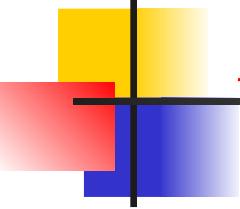
Example 21.6

Imagine there are three hosts in a network, as shown in Figure 21.19. A query message was received at time 0; the random delay time (in tenths of seconds) for each group is shown next to the group address. Show the sequence of report messages.

Solution

The events occur in this sequence:

- a. *Time 12: The timer for 228.42.0.0 in host A expires, and a membership report is sent, which is received by the router and every host including host B which cancels its timer for 228.42.0.0.*



Example 21.6 (continued)

- b. Time 30:** The timer for 225.14.0.0 in host A expires, and a membership report is sent which is received by the router and every host including host C which cancels its timer for 225.14.0.0.
- c. Time 50:** The timer for 238.71.0.0 in host B expires, and a membership report is sent, which is received by the router and every host.
- d. Time 70:** The timer for 230.43.0.0 in host C expires, and a membership report is sent, which is received by the router and every host including host A which cancels its timer for 230.43.0.0.

Figure 21.19 Example 21.6

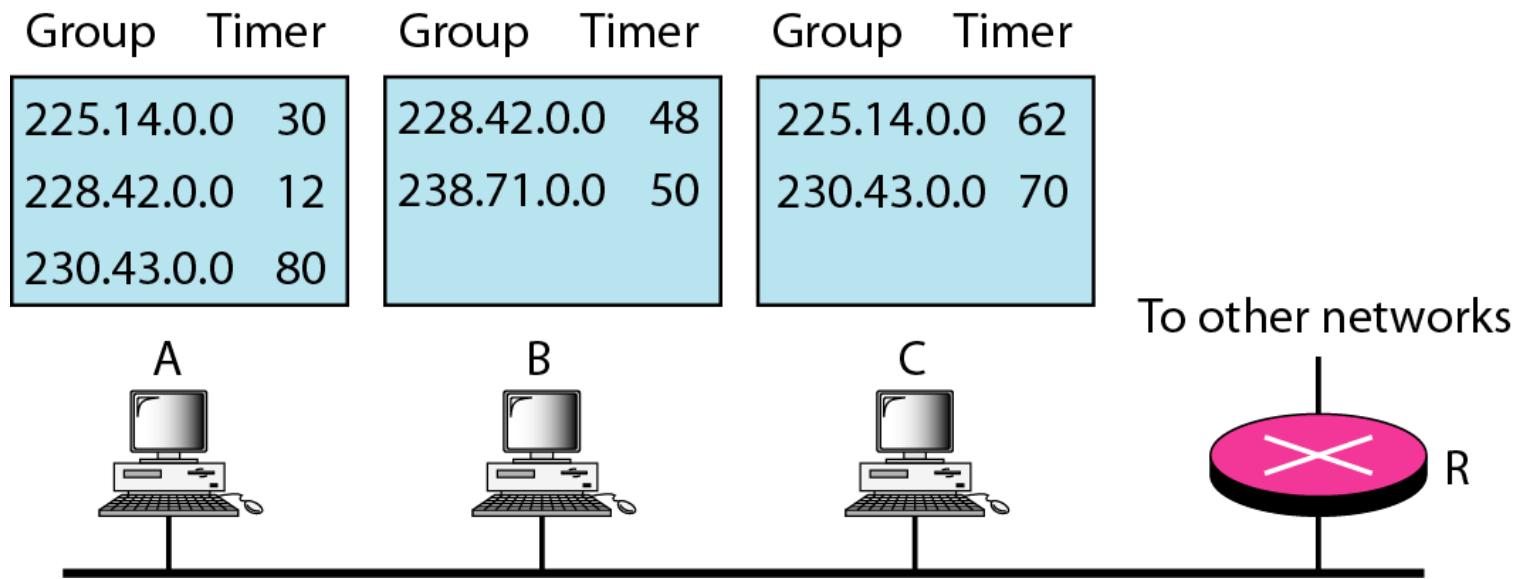
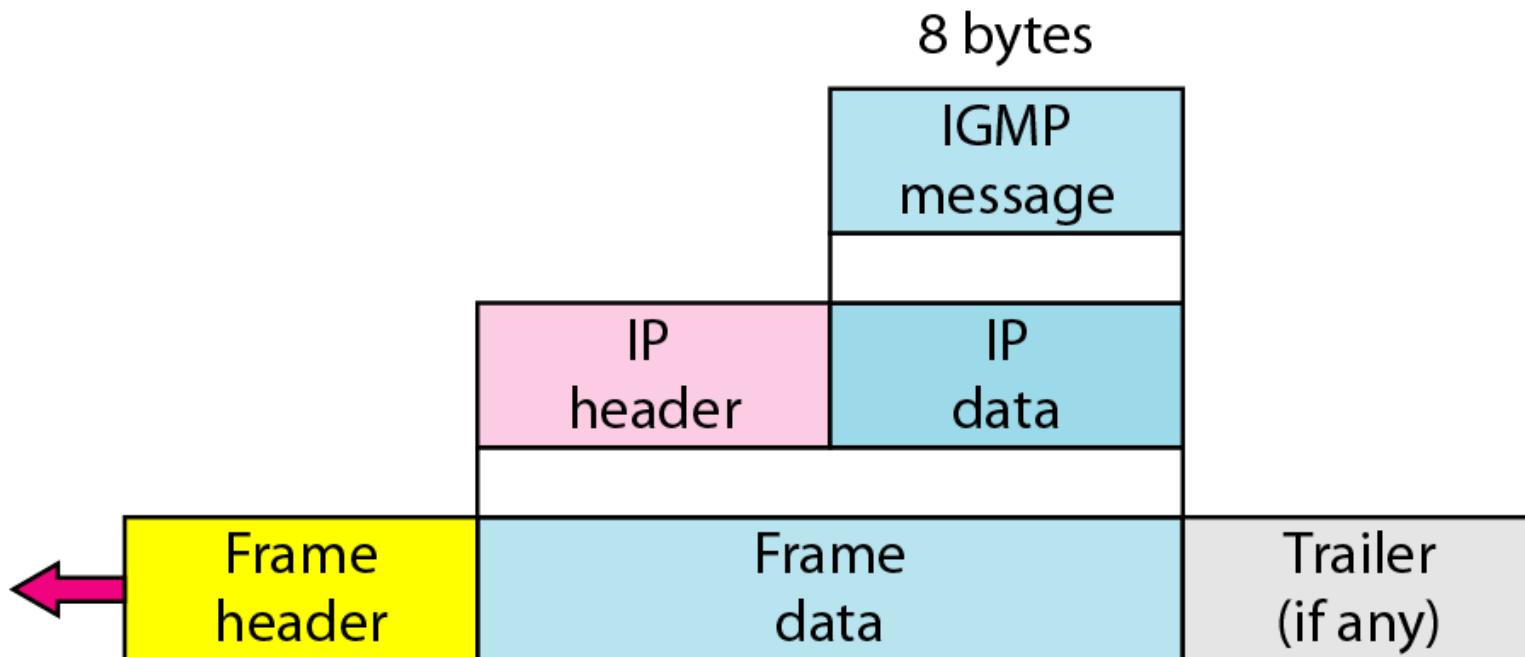
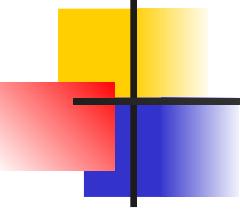


Figure 21.20 *Encapsulation of IGMP packet*





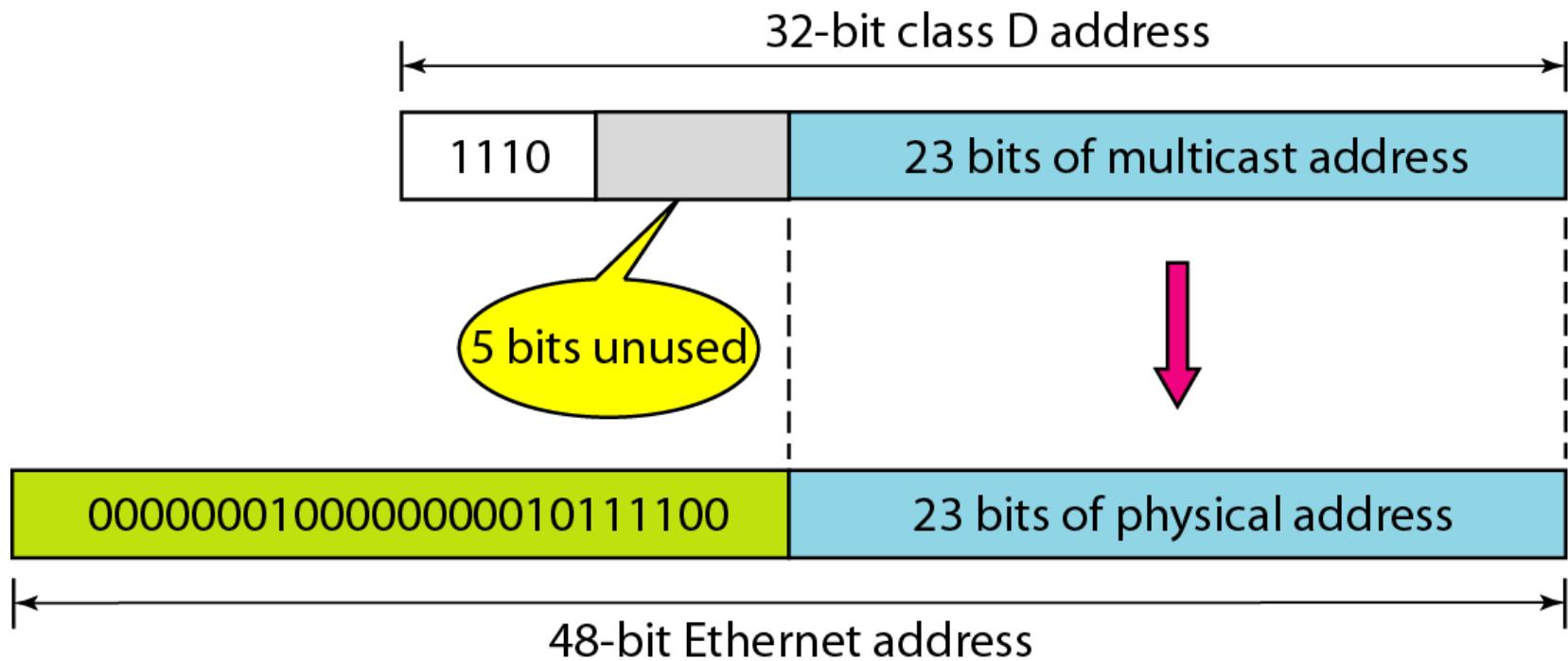
Note

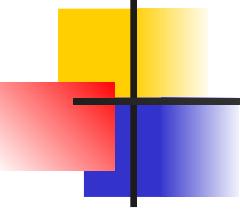
The IP packet that carries an IGMP packet has a value of 1 in its TTL field.

Table 21.2 *Destination IP addresses*

<i>Type</i>	<i>IP Destination Address</i>
Query	224.0.0.1 All systems on this subnet
Membership report	The multicast address of the group
Leave report	224.0.0.2 All routers on this subnet

Figure 21.21 *Mapping class D to Ethernet physical address*

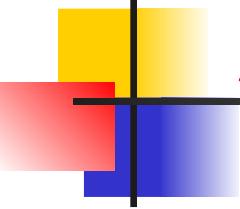




Note

**An Ethernet multicast physical address
is in the range**

01:00:5E:00:00:00 to 01:00:5E:7F:FF:FF.



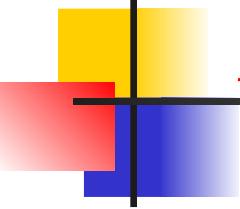
Example 21.7

Change the multicast IP address 230.43.14.7 to an Ethernet multicast physical address.

Solution

We can do this in two steps:

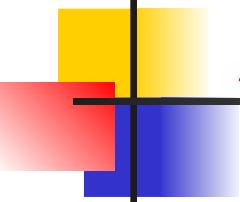
- a. *We write the rightmost 23 bits of the IP address in hexadecimal. This can be done by changing the rightmost 3 bytes to hexadecimal and then subtracting 8 from the leftmost digit if it is greater than or equal to 8. In our example, the result is 2B:0E:07.*



Example 21.7 (continued)

- b.** We add the result of part a to the starting Ethernet multicast address, which is 01:00:5E:00:00:00. The result is

01:00:5E:2B:0E:07



Example 21.8

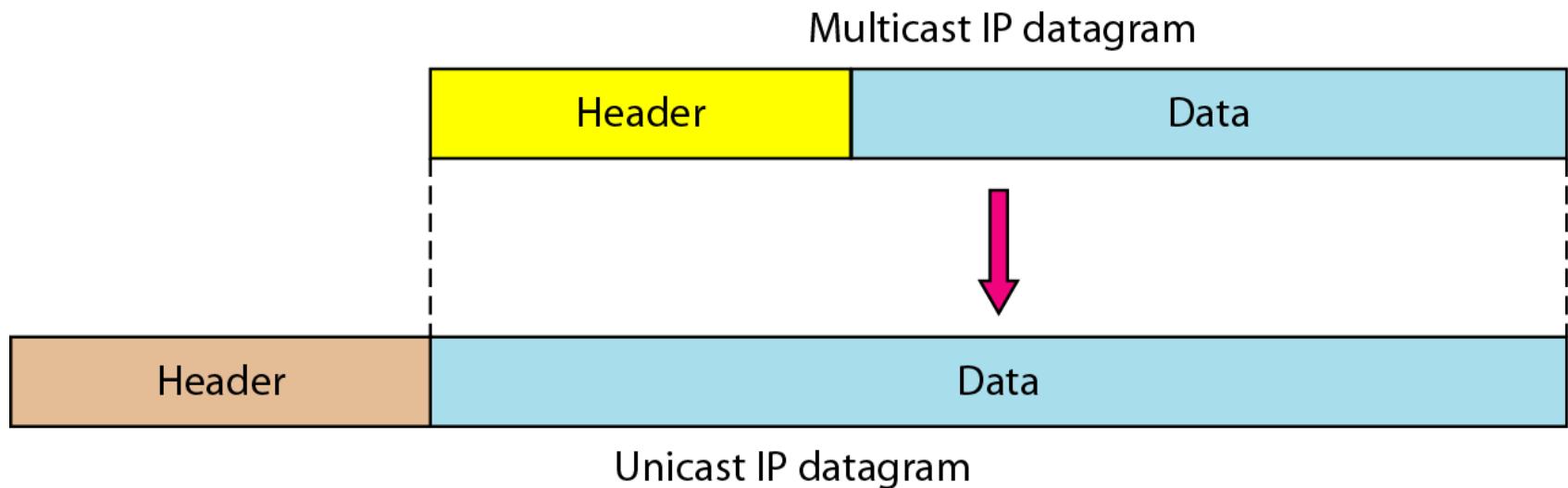
Change the multicast IP address 238.212.24.9 to an Ethernet multicast address.

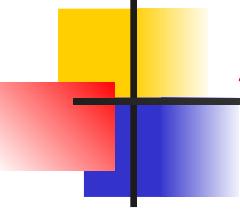
Solution

- a. The rightmost 3 bytes in hexadecimal is D4:18:09. We need to subtract 8 from the leftmost digit, resulting in 54:18:09.**
- b. We add the result of part a to the Ethernet multicast starting address. The result is**

01:00:5E:54:18:09

Figure 21.22 *Tunneling*





Example 21.9

We use *netstat* (see next slide) with three options: *-n*, *-r*, and *-a*. The *-n* option gives the numeric versions of IP addresses, the *-r* option gives the routing table, and the *-a* option gives all addresses (unicast and multicast). Note that we show only the fields relative to our discussion. “Gateway” defines the router, “Iface” defines the interface.

Note that the multicast address is shown in color. Any packet with a multicast address from 224.0.0.0 to 239.255.255.255 is masked and delivered to the Ethernet interface.

Example 21.9 (continued)

```
$ netstat -nra
```

Kernel IP routing table

Destination	Gateway	Mask	Flags	Iface
153.18.16.0	0.0.0.0	255.255.240.0	U	eth0
169.254.0.0	0.0.0.0	255.255.0.0	U	eth0
127.0.0.0	0.0.0.0	255.0.0.0	U	lo
224.0.0.0	0.0.0.0	224.0.0.0	U	eth0
0.0.0.0	153.18.31.254	0.0.0.0	UG	eth0

21-4 ICMPv6

We discussed IPv6 in Chapter 20. Another protocol that has been modified in version 6 of the TCP/IP protocol suite is ICMP (ICMPv6). This new version follows the same strategy and purposes of version 4.

Topics discussed in this section:

Error Reporting

Query

Figure 21.23 Comparison of network layers in version 4 and version 6

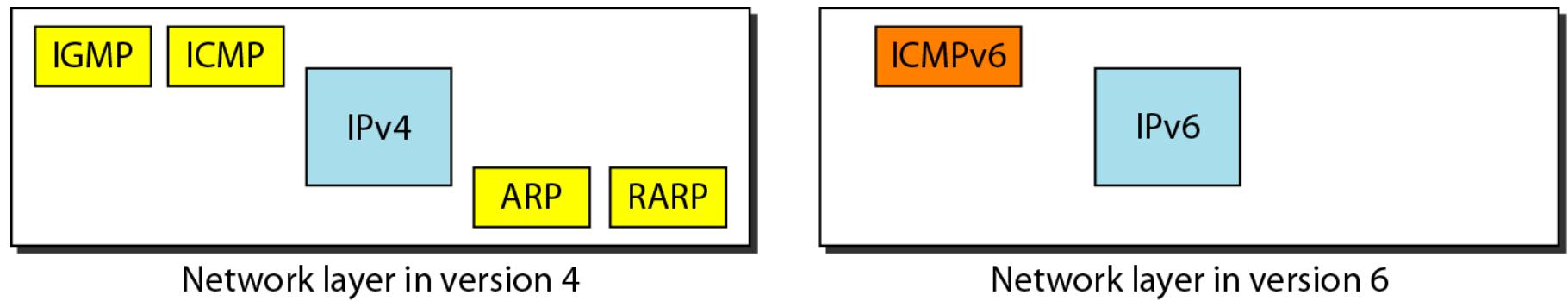


Table 21.3 Comparison of error-reporting messages in ICMPv4 and ICMPv6

Type of Message	Version 4	Version 6
Destination unreachable	Yes	Yes
Source quench	Yes	No
Packet too big	No	Yes
Time exceeded	Yes	Yes
Parameter problem	Yes	Yes
Redirection	Yes	Yes

Table 21.4 *Comparison of query messages in ICMPv4 and ICMPv6*

Type of Message	Version 4	Version 6
Echo request and reply	Yes	Yes
Timestamp request and reply	Yes	No
Address-mask request and reply	Yes	No
Router solicitation and advertisement	Yes	Yes
Neighbor solicitation and advertisement	ARP	Yes
Group membership	IGMP	Yes



Data Communications and Networking

Fourth Edition

Forouzan

Chapter 22

Network Layer: Delivery, Forwarding, and Routing

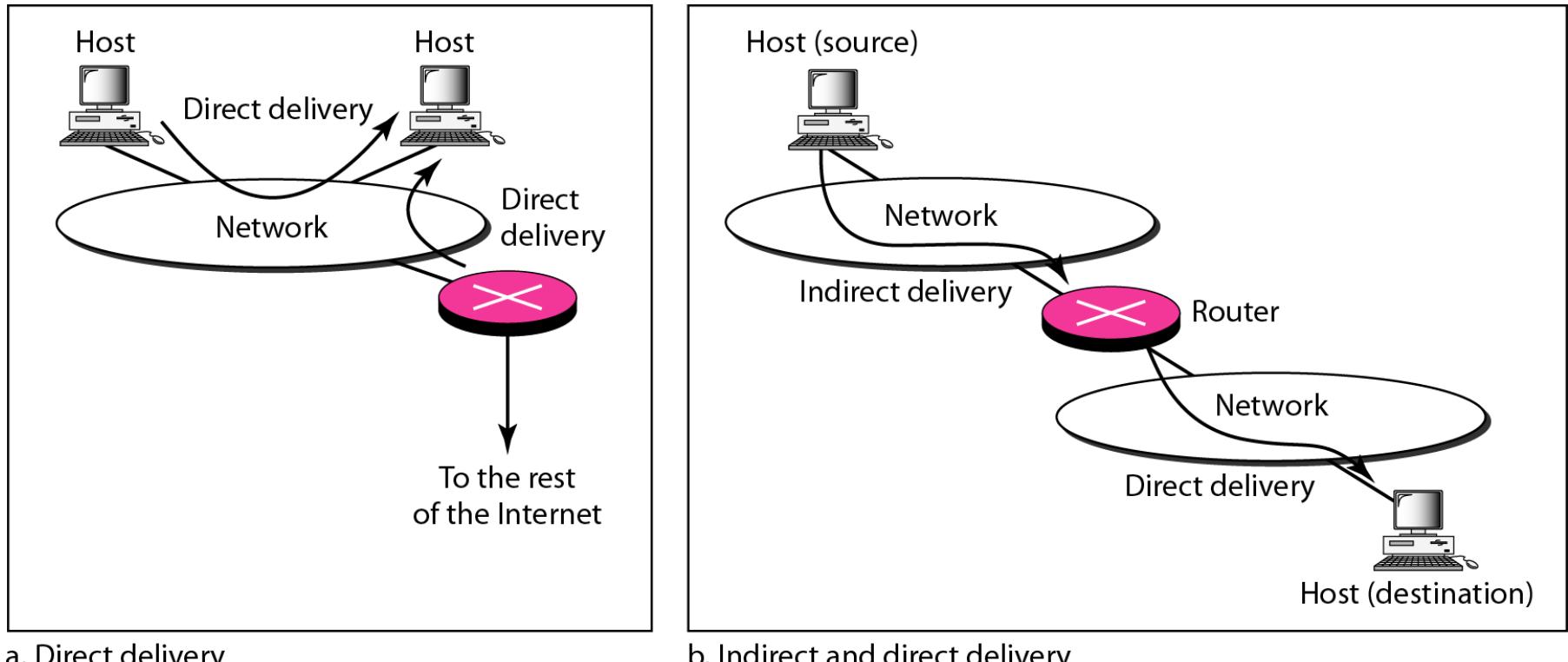
22-1 DELIVERY

The network layer supervises the handling of the packets by the underlying physical networks. We define this handling as the delivery of a packet.

Topics discussed in this section:

Direct Versus Indirect Delivery

Figure 22.1 Direct and indirect delivery



22-2 FORWARDING

Forwarding means to place the packet in its route to its destination. Forwarding requires a host or a router to have a routing table. When a host has a packet to send or when a router has received a packet to be forwarded, it looks at this table to find the route to the final destination.

Topics discussed in this section:

Forwarding Techniques

Forwarding Process

Routing Table

Figure 22.2 Route method versus next-hop method

a. Routing tables based on route

Destination	Route
Host B	R1, R2, host B

Routing table
for host A

Destination	Route
Host B	R2, host B

Routing table
for R1

Destination	Route
Host B	Host B

Routing table
for R2

b. Routing tables based on next hop

Destination	Next hop
Host B	R1

Destination	Next hop
Host B	R2

Destination	Next hop
Host B	---

Host A



Network

R1

Host B



Network

R2

Network

Figure 22.3 Host-specific versus network-specific method

Routing table for host S based on host-specific method

Destination	Next hop
A	R1
B	R1
C	R1
D	R1

Routing table for host S based on network-specific method

Destination	Next hop
N2	R1

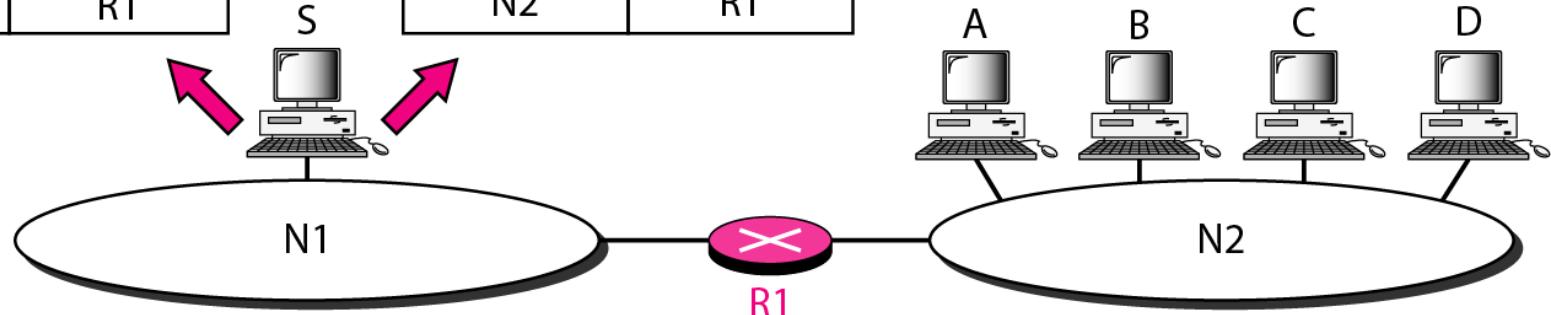


Figure 22.4 Default method

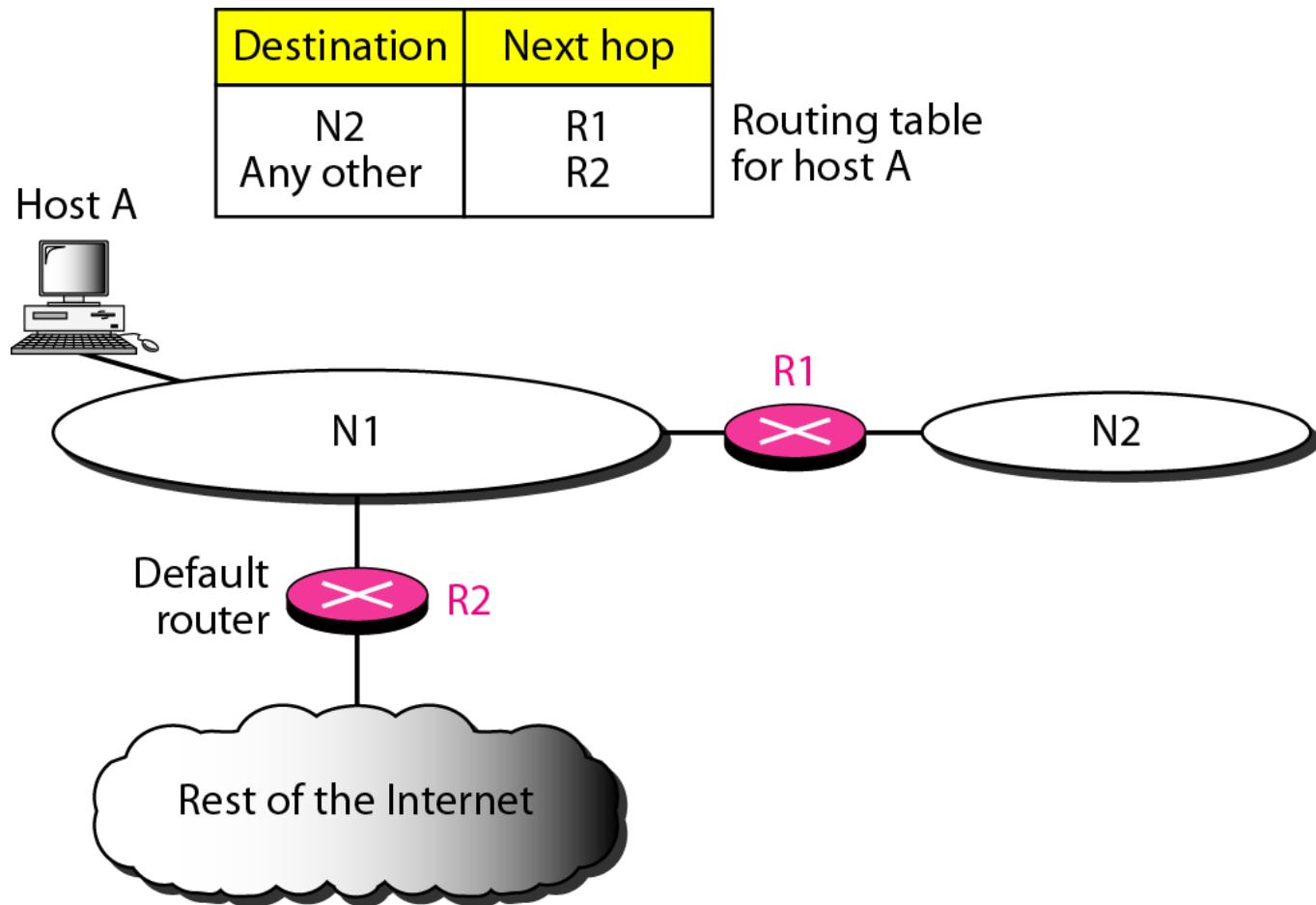
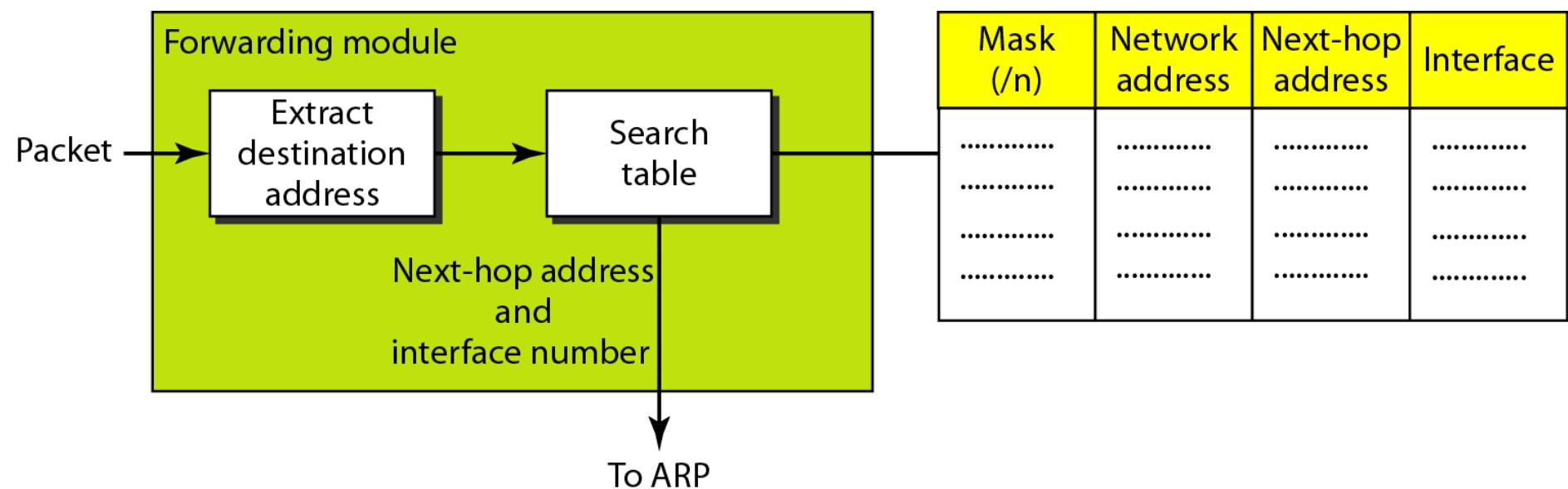
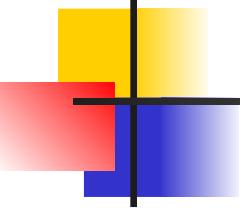


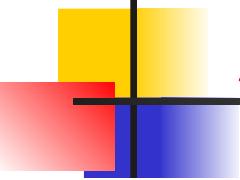
Figure 22.5 Simplified forwarding module in classless address





Note

In classless addressing, we need at least four columns in a routing table.



Example 22.1

Make a routing table for router R1, using the configuration in Figure 22.6.

Solution

Table 22.1 shows the corresponding table.

Figure 22.6 Configuration for Example 22.1

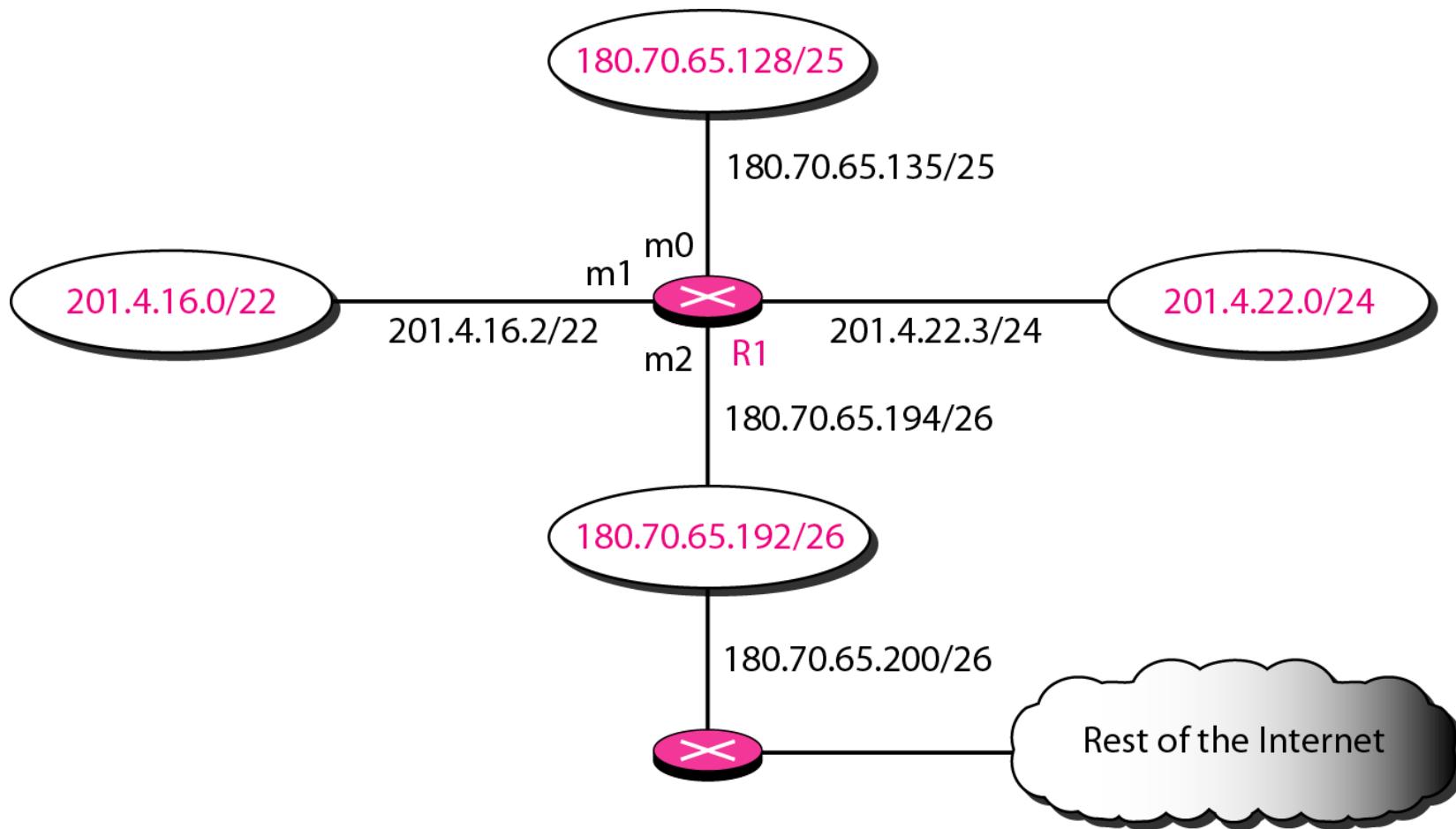
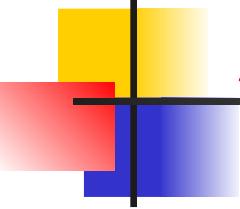


Table 22.1 *Routing table for router R1 in Figure 22.6*

<i>Mask</i>	<i>Network Address</i>	<i>Next Hop</i>	<i>Interface</i>
/26	180.70.65.192	—	m2
/25	180.70.65.128	—	m0
/24	201.4.22.0	—	m3
/22	201.4.16.0	m1
Any	Any	180.70.65.200	m2



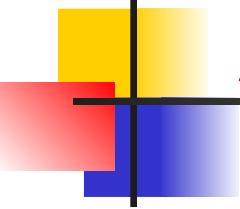
Example 22.2

Show the forwarding process if a packet arrives at R1 in Figure 22.6 with the destination address 180.70.65.140.

Solution

The router performs the following steps:

- 1. The first mask (/26) is applied to the destination address. The result is 180.70.65.128, which does not match the corresponding network address.***
- 2. The second mask (/25) is applied to the destination address. The result is 180.70.65.128, which matches the corresponding network address. The next-hop address and the interface number m0 are passed to ARP for further processing.***



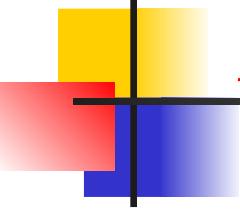
Example 22.3

Show the forwarding process if a packet arrives at R1 in Figure 22.6 with the destination address 201.4.22.35.

Solution

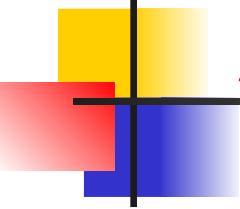
The router performs the following steps:

- 1. The first mask (/26) is applied to the destination address. The result is 201.4.22.0, which does not match the corresponding network address.*
- 2. The second mask (/25) is applied to the destination address. The result is 201.4.22.0, which does not match the corresponding network address (row 2).*



Example 22.3 (continued)

3. The third mask (/24) is applied to the destination address. The result is 201.4.22.0, which matches the corresponding network address. The destination address of the packet and the interface number m3 are passed to ARP.



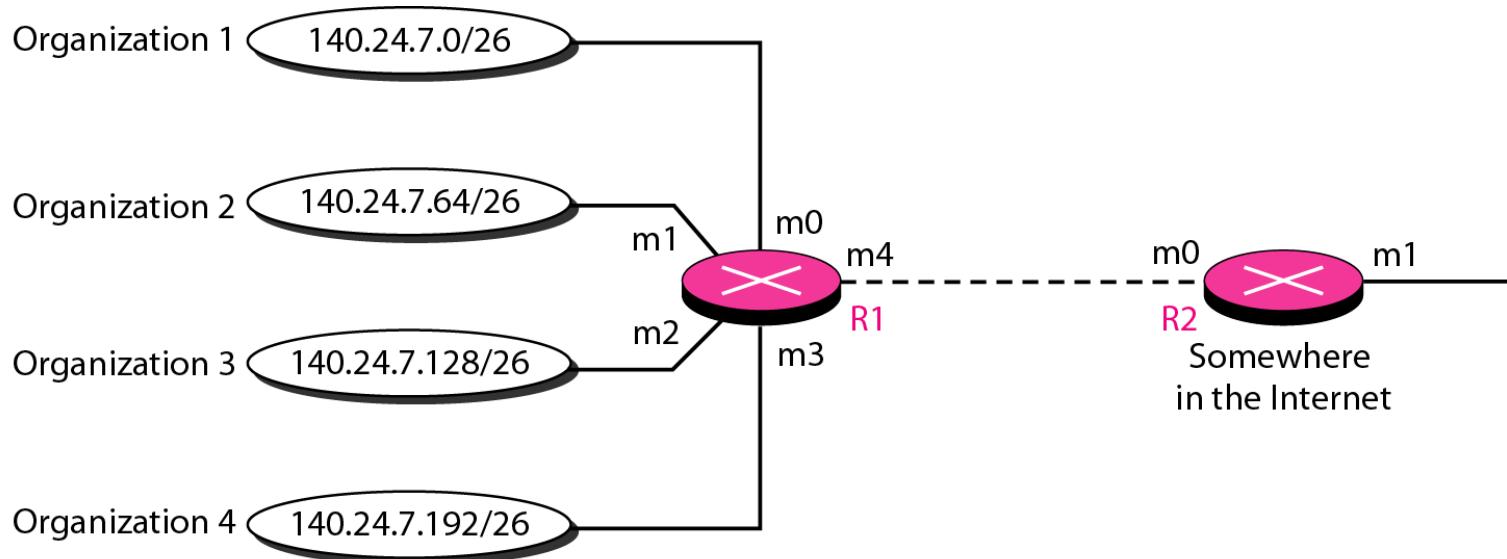
Example 22.4

Show the forwarding process if a packet arrives at R1 in Figure 22.6 with the destination address 18.24.32.78.

Solution

This time all masks are applied, one by one, to the destination address, but no matching network address is found. When it reaches the end of the table, the module gives the next-hop address 180.70.65.200 and interface number m2 to ARP. This is probably an outgoing package that needs to be sent, via the default router, to someplace else in the Internet.

Figure 22.7 Address aggregation



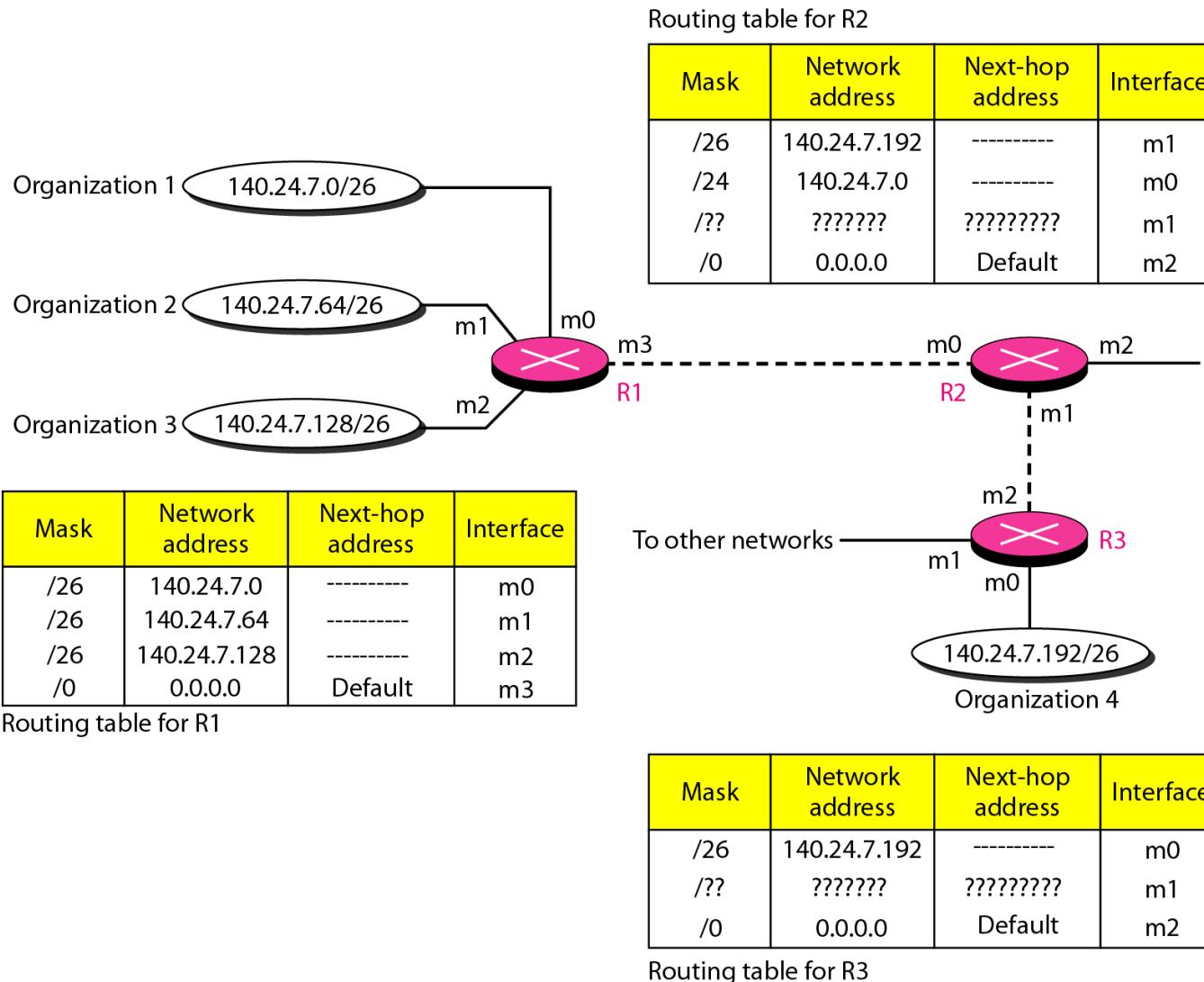
Mask	Network address	Next-hop address	Interface
/26	140.24.7.0	-----	m0
/26	140.24.7.64	-----	m1
/26	140.24.7.128	-----	m2
/26	140.24.7.192	-----	m3
/0	0.0.0.0	Default	m4

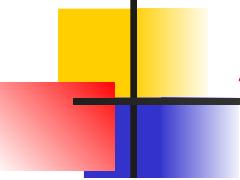
Routing table for R1

Mask	Network address	Next-hop address	Interface
/24	140.24.7.0	-----	m0
/0	0.0.0.0	Default	m1

Routing table for R2

Figure 22.8 Longest mask matching

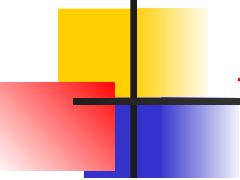




Example 22.5

As an example of hierarchical routing, let us consider Figure 22.9. A regional ISP is granted 16,384 addresses starting from 120.14.64.0. The regional ISP has decided to divide this block into four subblocks, each with 4096 addresses. Three of these subblocks are assigned to three local ISPs; the second subblock is reserved for future use. Note that the mask for each block is /20 because the original block with mask /18 is divided into 4 blocks.

The first local ISP has divided its assigned subblock into 8 smaller blocks and assigned each to a small ISP. Each small ISP provides services to 128 households, each using four addresses.



Example 22.5 (continued)

The second local ISP has divided its block into 4 blocks and has assigned the addresses to four large organizations.

The third local ISP has divided its block into 16 blocks and assigned each block to a small organization. Each small organization has 256 addresses, and the mask is /24.

There is a sense of hierarchy in this configuration. All routers in the Internet send a packet with destination address 120.14.64.0 to 120.14.127.255 to the regional ISP.

Figure 22.9 Hierarchical routing with ISPs

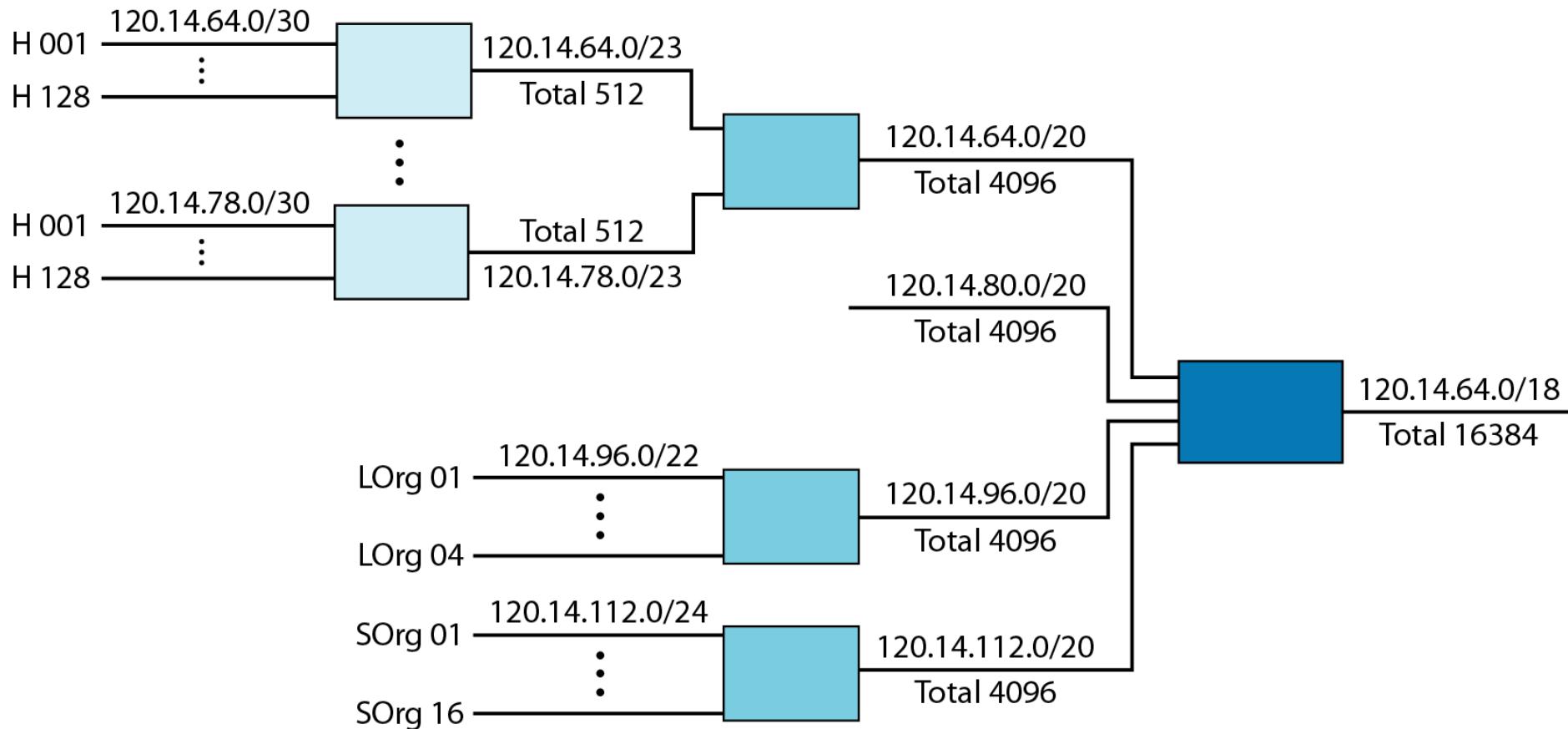
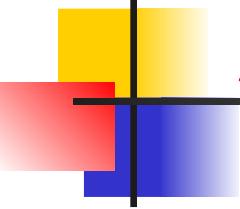


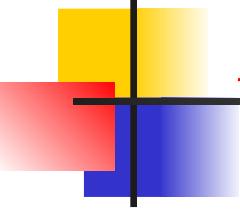
Figure 22.10 *Common fields in a routing table*

Mask	Network address	Next-hop address	Interface	Flags	Reference count	Use
.....



Example 22.6

*One utility that can be used to find the contents of a routing table for a host or router is **netstat** in UNIX or LINUX. The next slide shows the list of the contents of a default server. We have used two options, **r** and **n**. The option **r** indicates that we are interested in the routing table, and the option **n** indicates that we are looking for numeric addresses. Note that this is a routing table for a host, not a router. Although we discussed the routing table for a router throughout the chapter, a host also needs a routing table.*



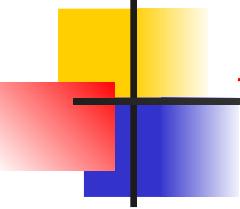
Example 22.6 (continued)

```
$ netstat -rn
```

Kernel IP routing table

Destination	Gateway	Mask	Flags	Iface
153.18.16.0	0.0.0.0	255.255.240.0	U	eth0
127.0.0.0	0.0.0.0	255.0.0.0	U	lo
0.0.0.0	153.18.31.254	0.0.0.0	UG	eth0

The destination column here defines the network address. The term gateway used by UNIX is synonymous with router. This column actually defines the address of the next hop. The value 0.0.0.0 shows that the delivery is direct. The last entry has a flag of G, which means that the destination can be reached through a router (default router). The Iface defines the interface.



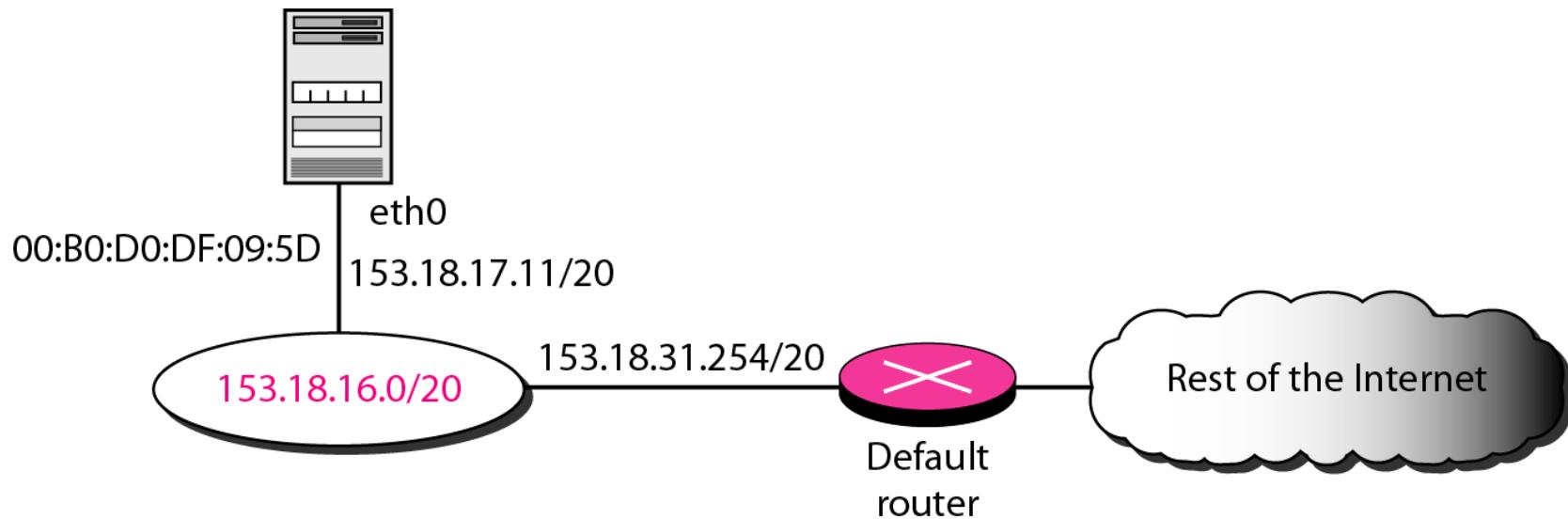
Example 22.6 (continued)

*More information about the IP address and physical address of the server can be found by using the **ifconfig** command on the given interface (eth0).*

```
$ ifconfig eth0
```

```
eth0 Link encap:Ethernet HWaddr 00:B0:D0:DF:09:5D  
inet addr:153.18.17.11 Bcast:153.18.31.255 Mask:255.255.240.0  
...
```

Figure 22.11 Configuration of the server for Example 22.6



22-3 UNICAST ROUTING PROTOCOLS

A routing table can be either static or dynamic. A static table is one with manual entries. A dynamic table is one that is updated automatically when there is a change somewhere in the Internet. A routing protocol is a combination of rules and procedures that lets routers in the Internet inform each other of changes.

Topics discussed in this section:

Optimization

Intra- and Interdomain Routing

Distance Vector Routing and RIP

Link State Routing and OSPF

Path Vector Routing and BGP

Figure 22.12 Autonomous systems

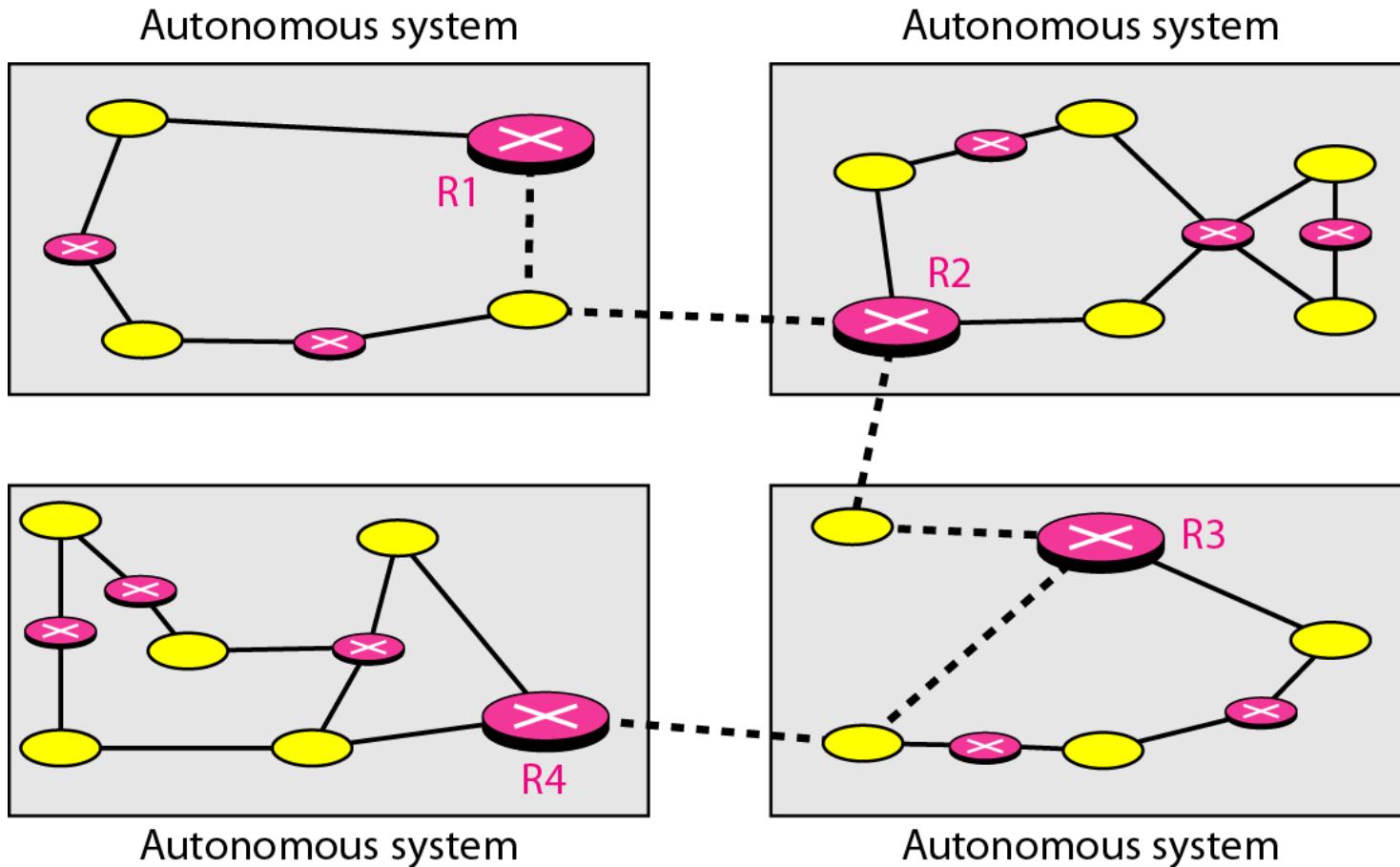


Figure 22.13 Popular routing protocols

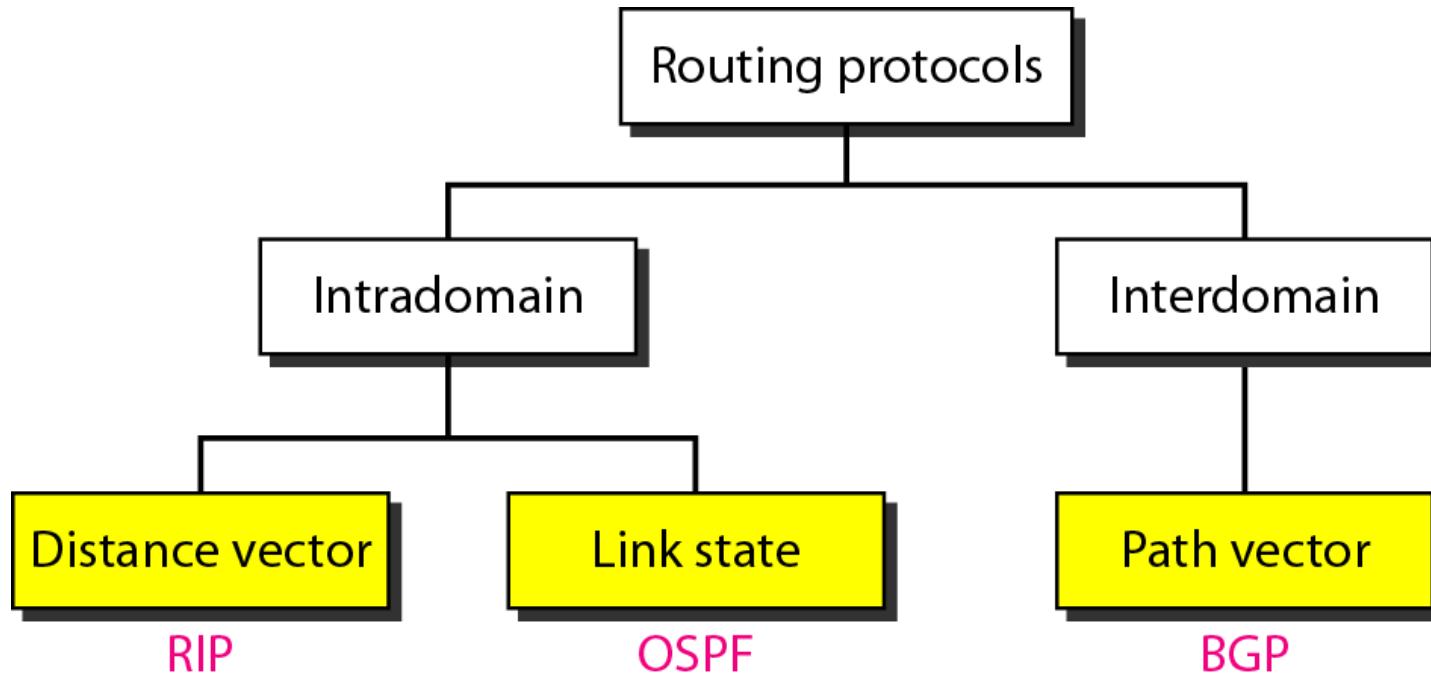


Figure 22.14 Distance vector routing tables

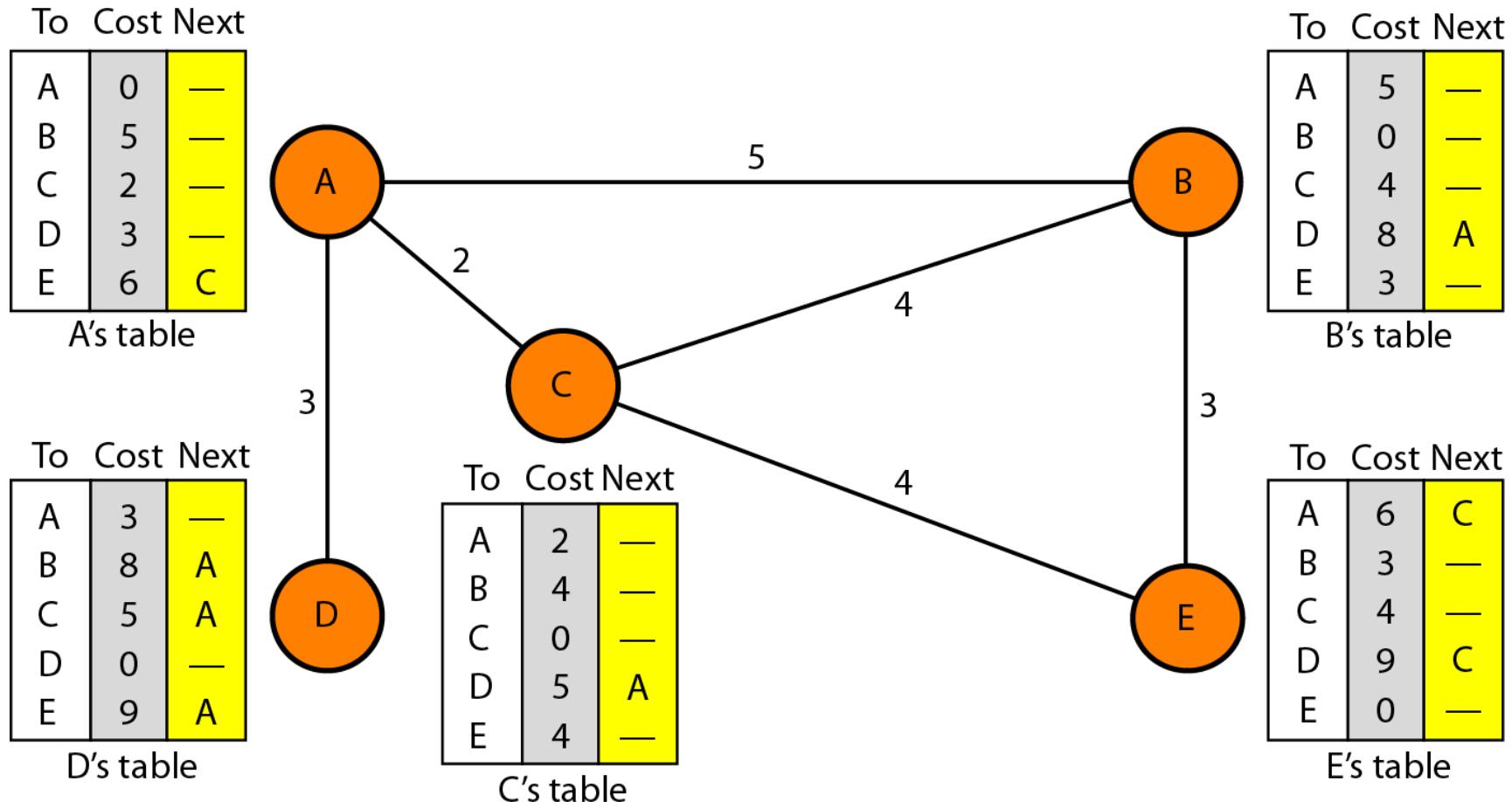
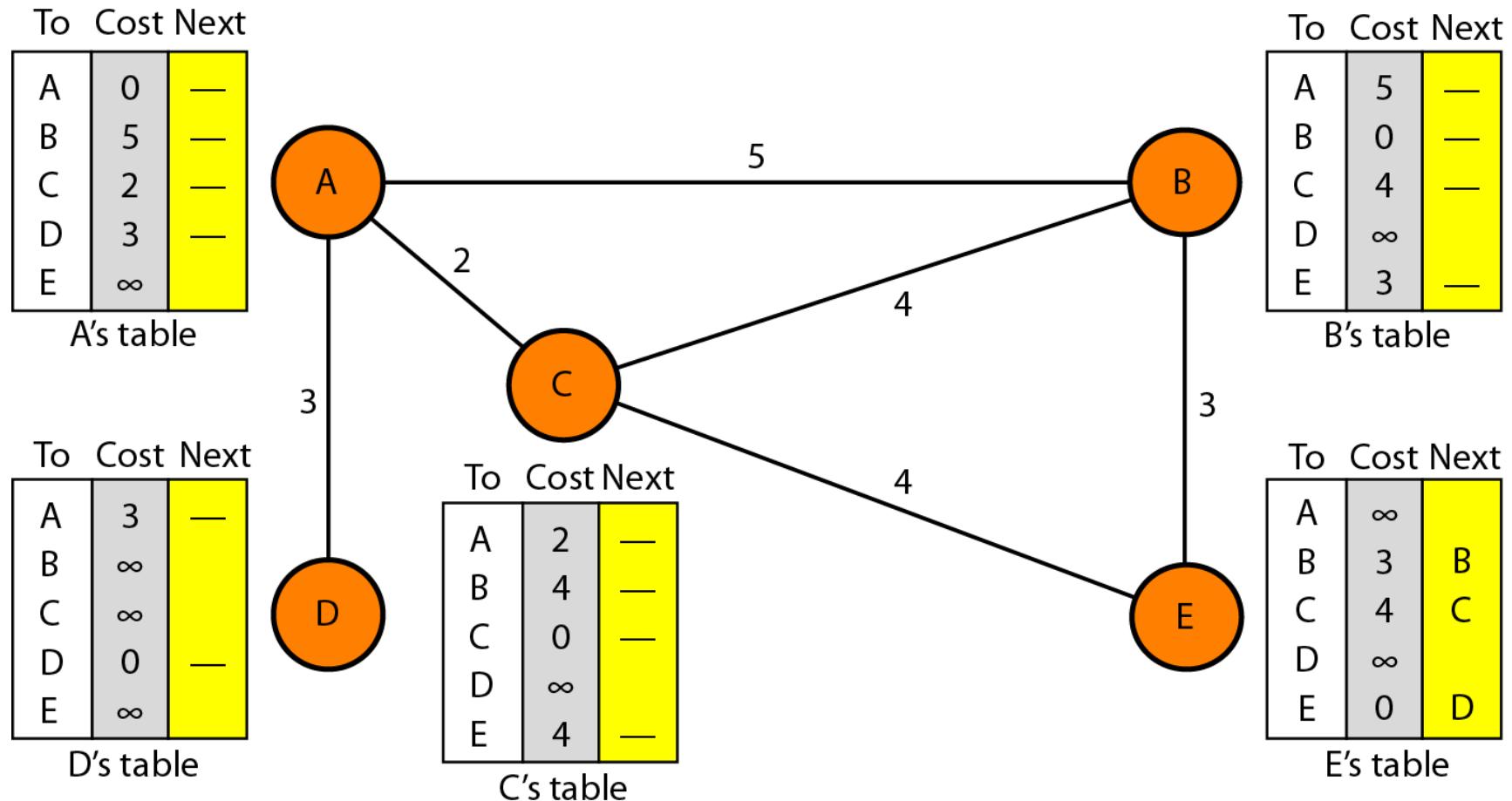
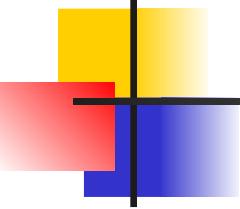


Figure 22.15 Initialization of tables in distance vector routing





Note

In distance vector routing, each node shares its routing table with its immediate neighbors periodically and when there is a change.

Figure 22.16 Updating in distance vector routing

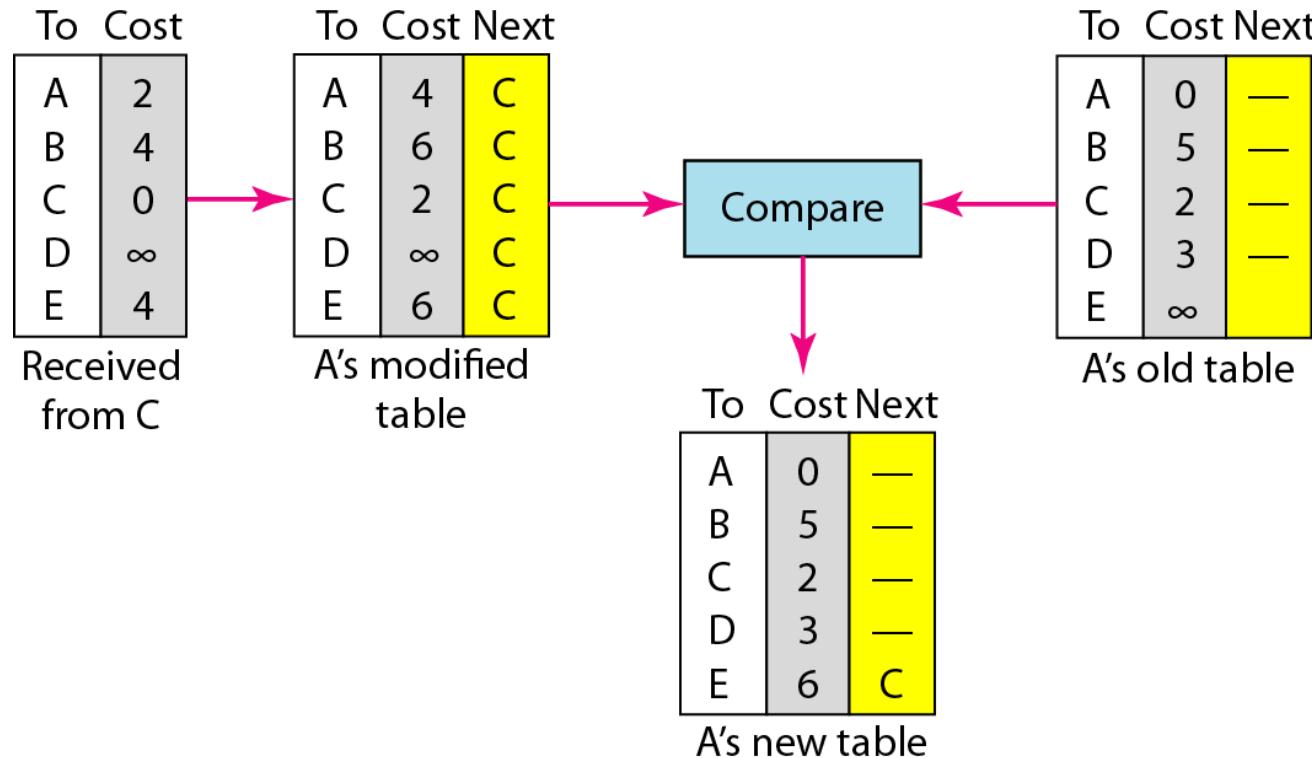


Figure 22.17 Two-node instability

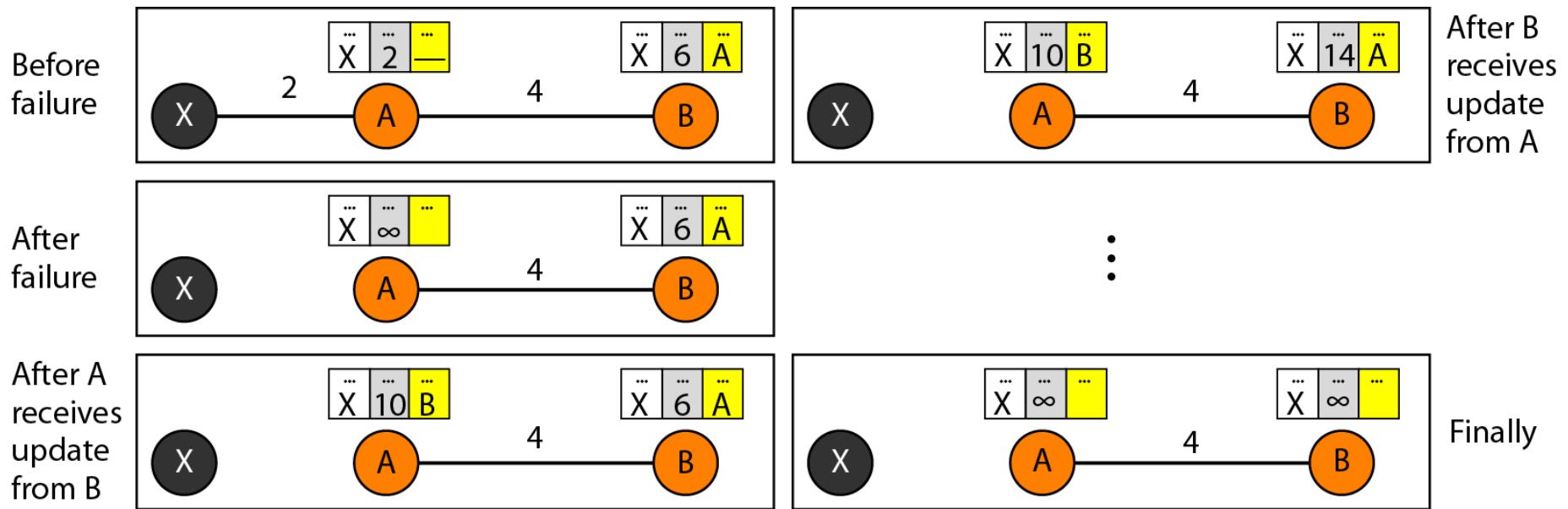


Figure 22.18 Three-node instability

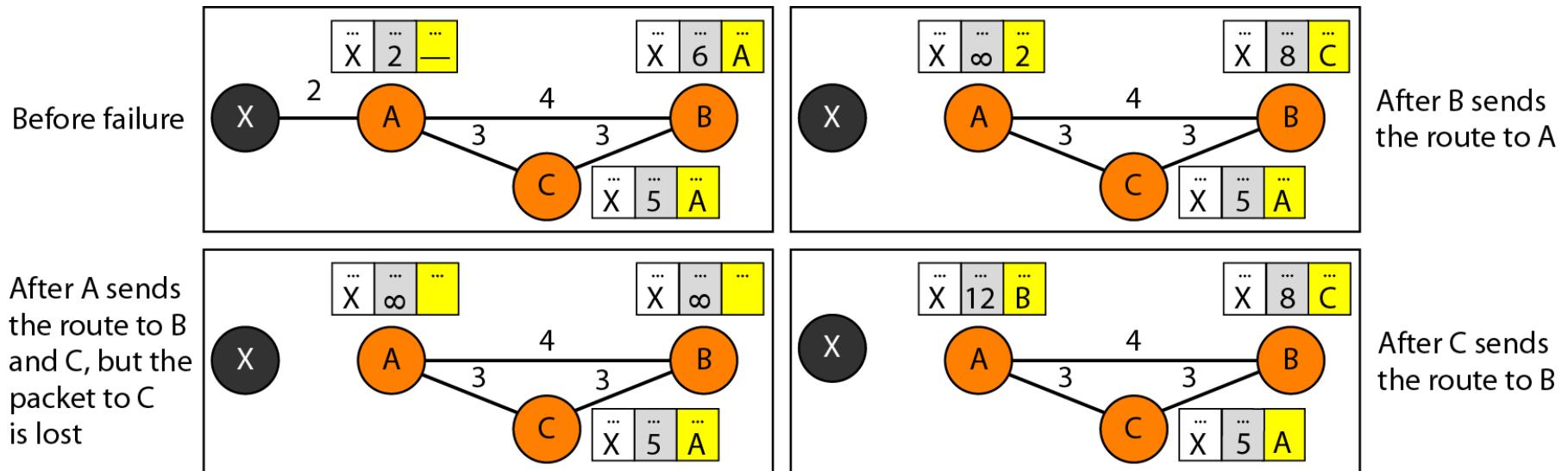


Figure 22.19 Example of a domain using RIP

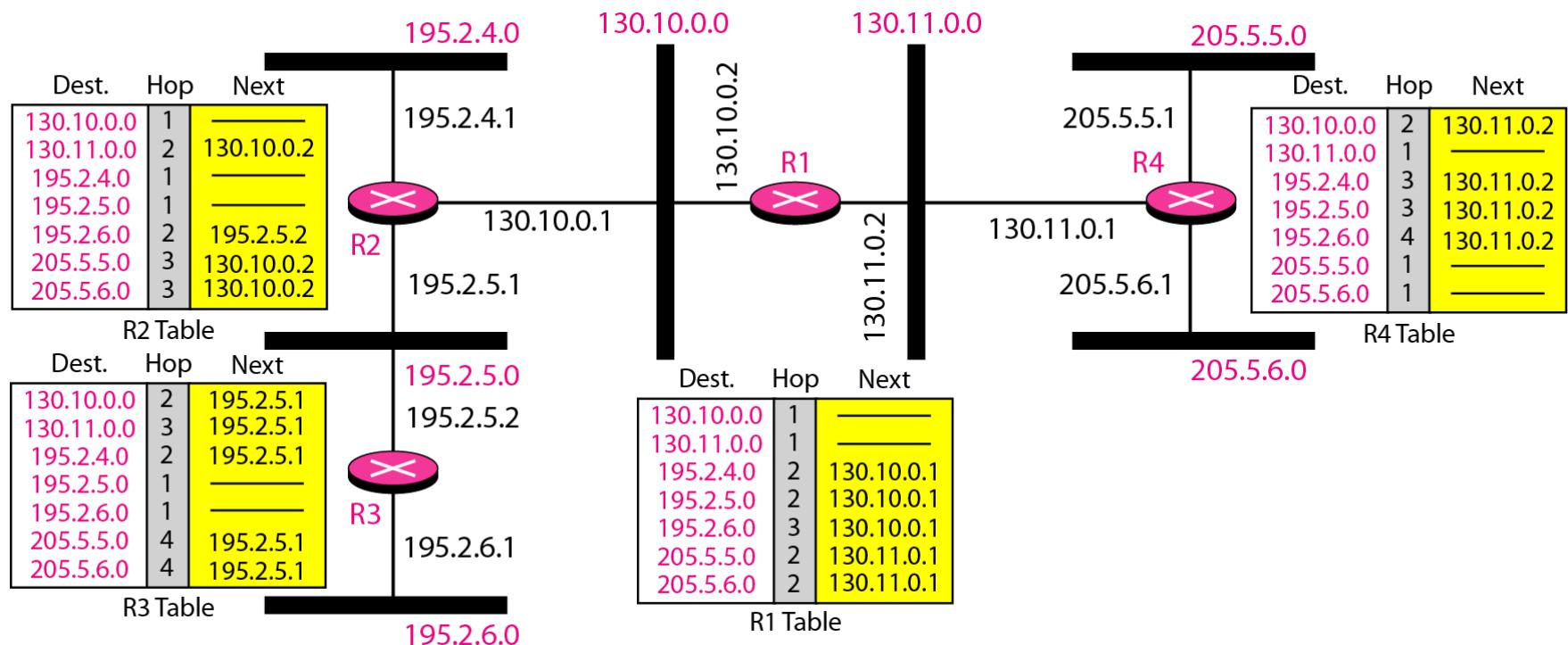


Figure 22.20 Concept of link state routing

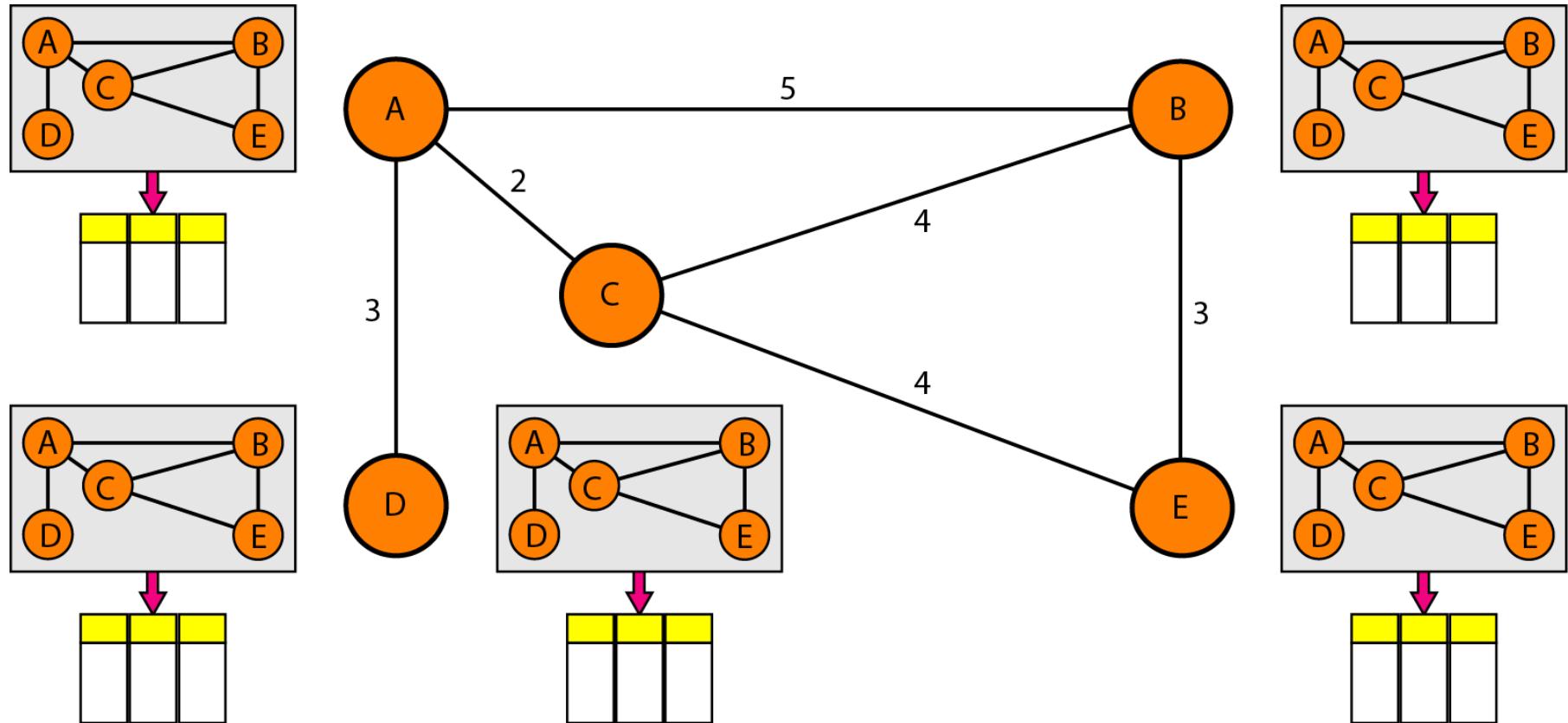


Figure 22.21 *Link state knowledge*

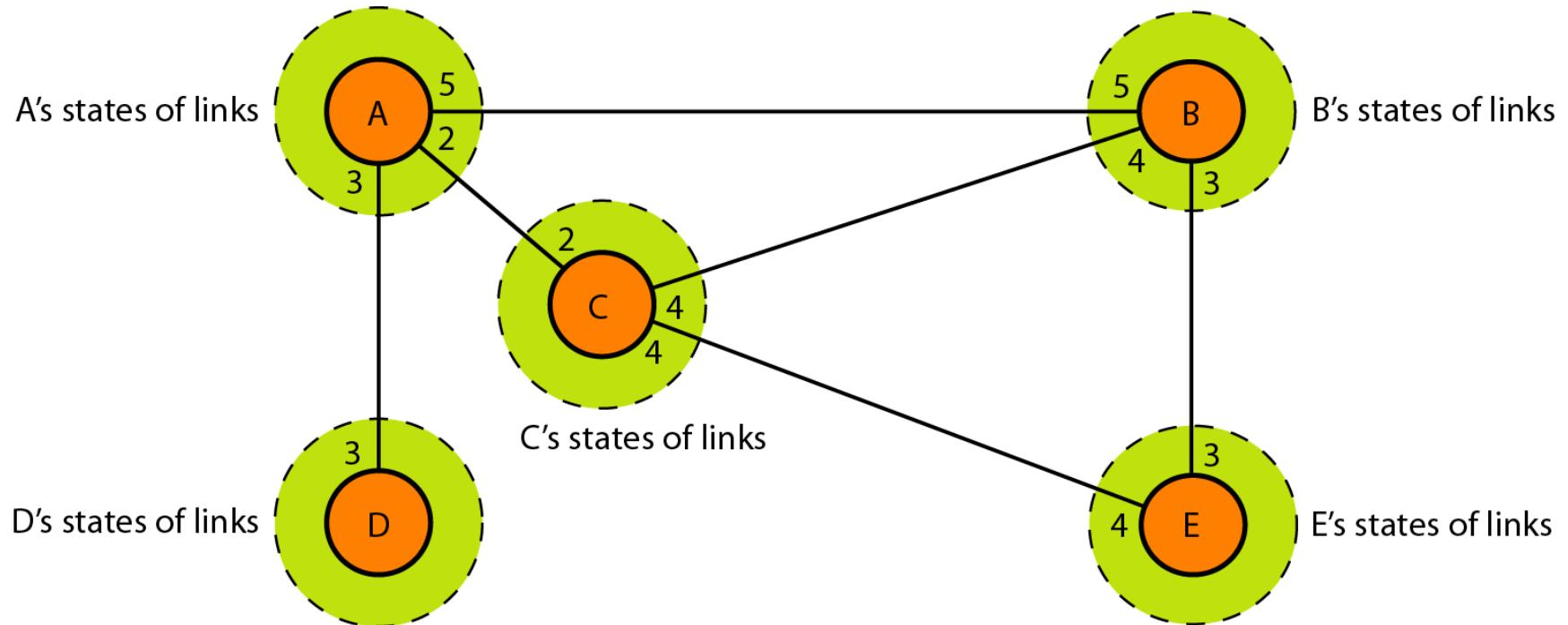


Figure 22.22 Dijkstra algorithm

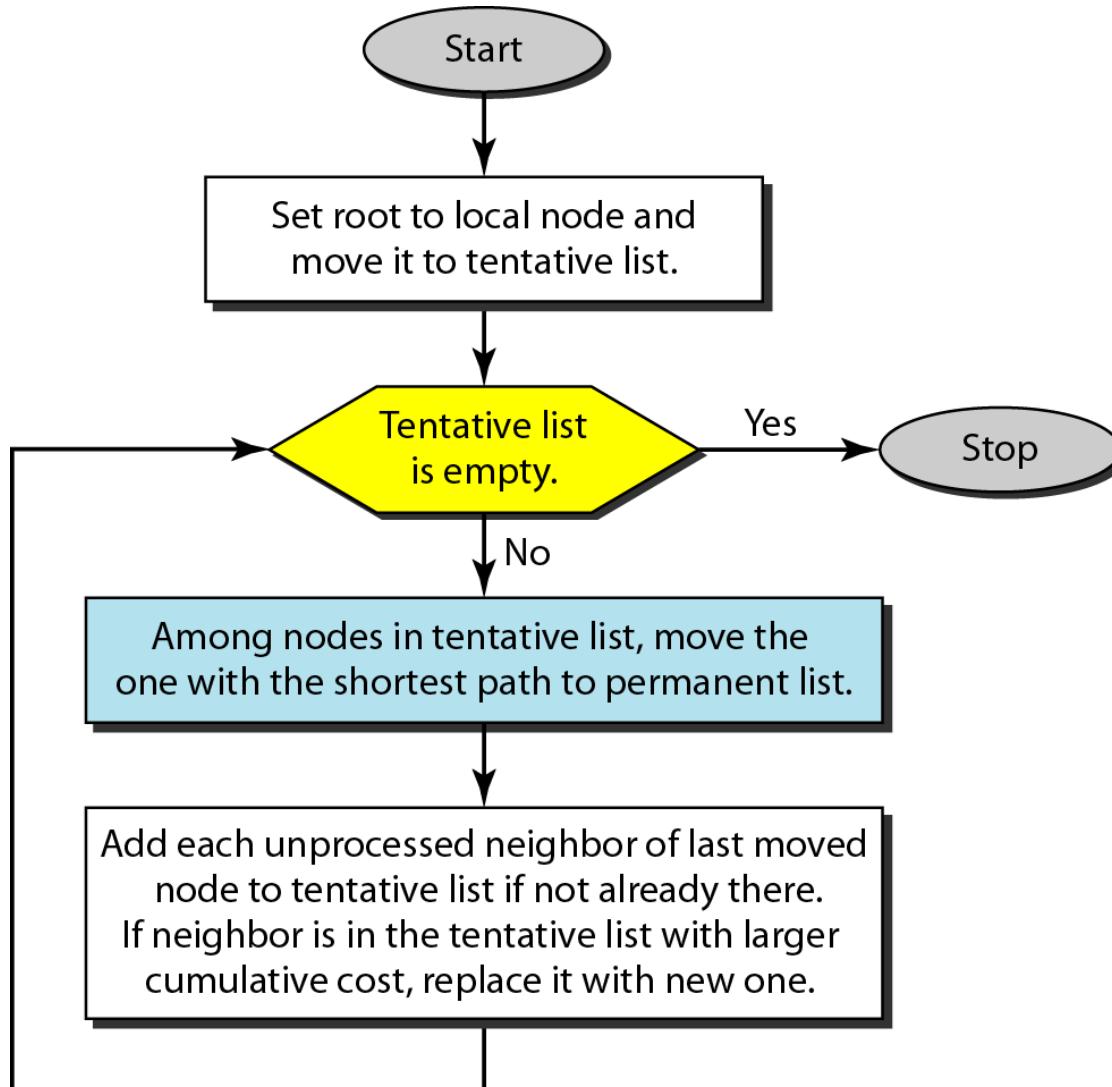


Figure 22.23 Example of formation of shortest path tree

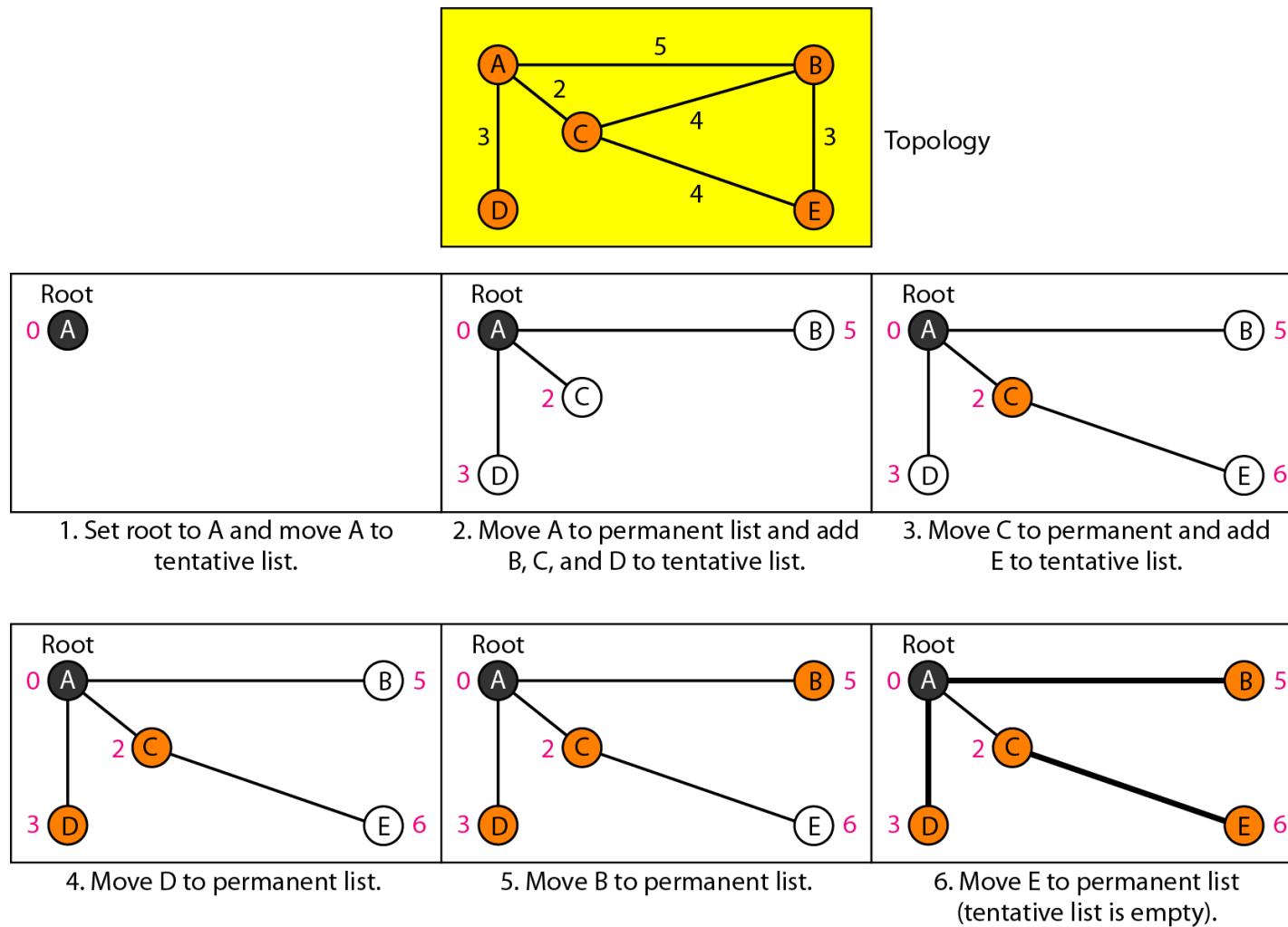


Table 22.2 *Routing table for node A*

<i>Node</i>	<i>Cost</i>	<i>Next Router</i>
A	0	—
B	5	—
C	2	—
D	3	—
E	6	C

Figure 22.24 Areas in an autonomous system

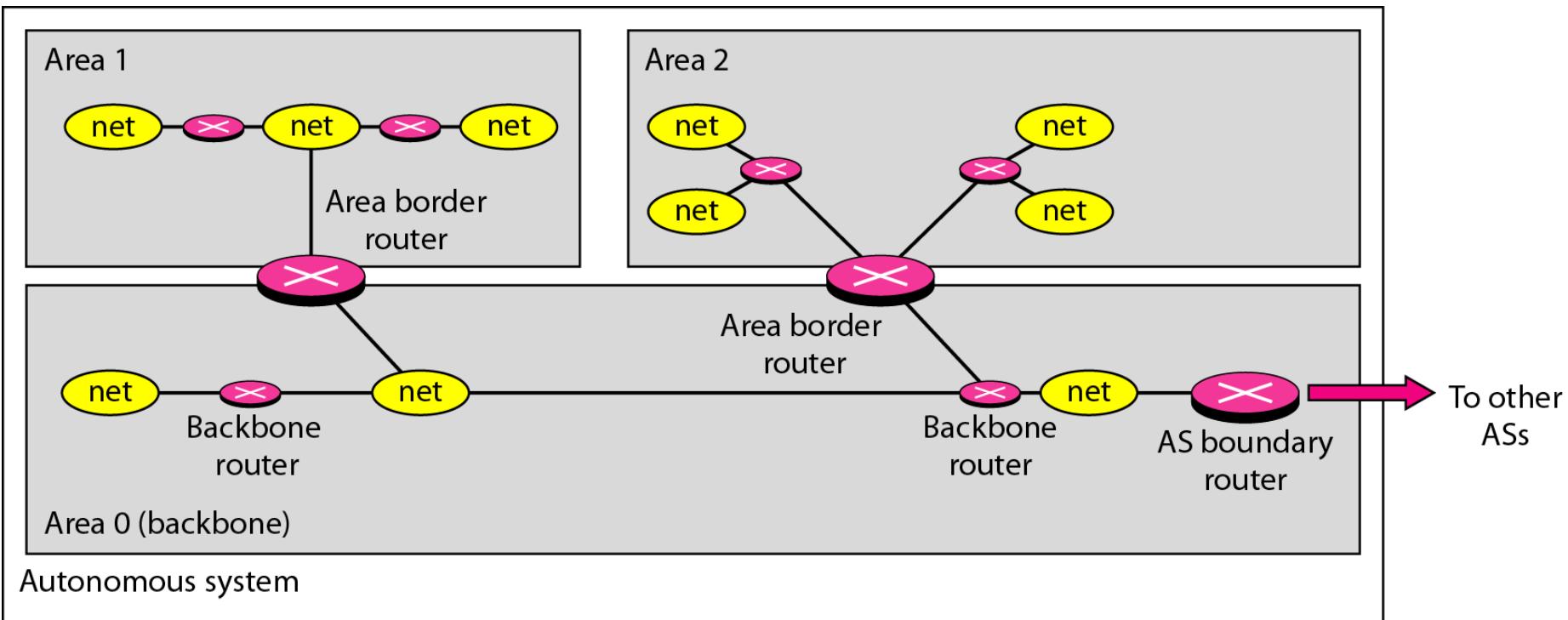


Figure 22.25 *Types of links*

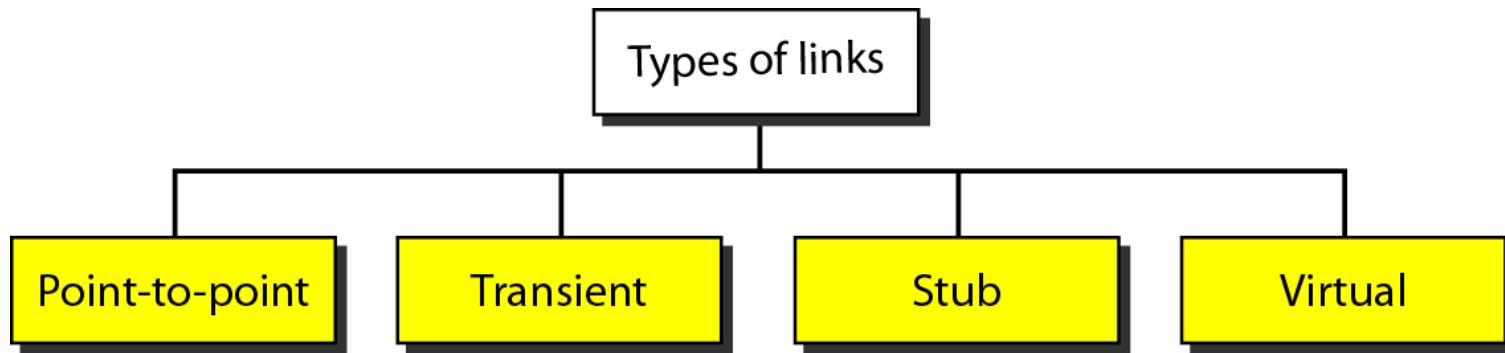


Figure 22.26 Point-to-point link

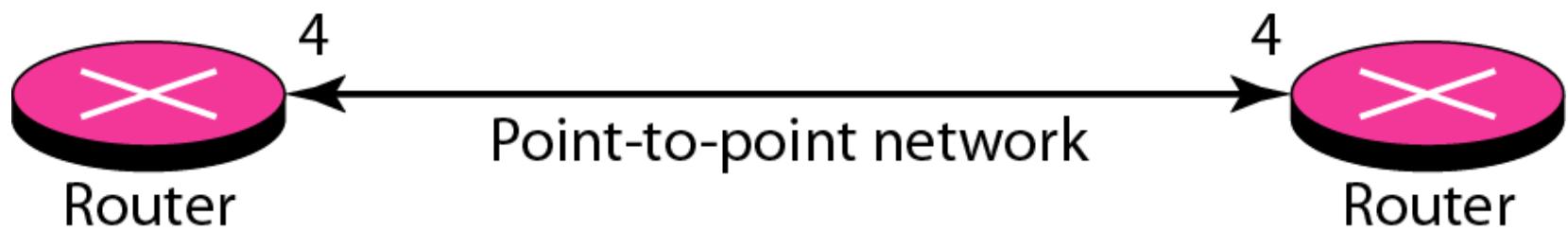
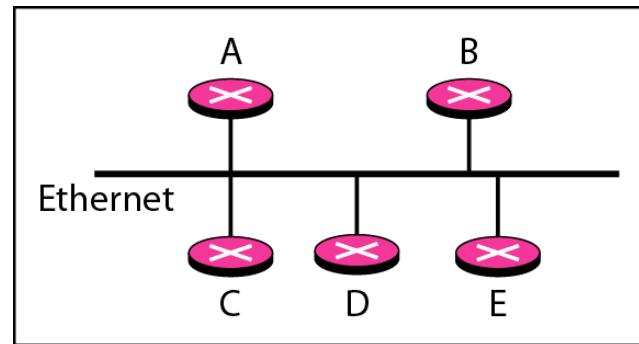
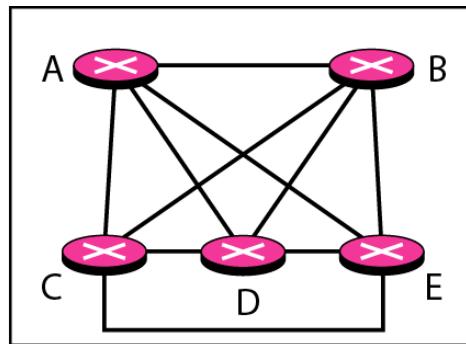


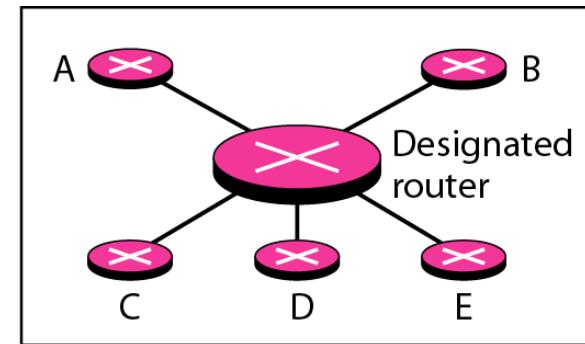
Figure 22.27 Transient link



a. Transient network

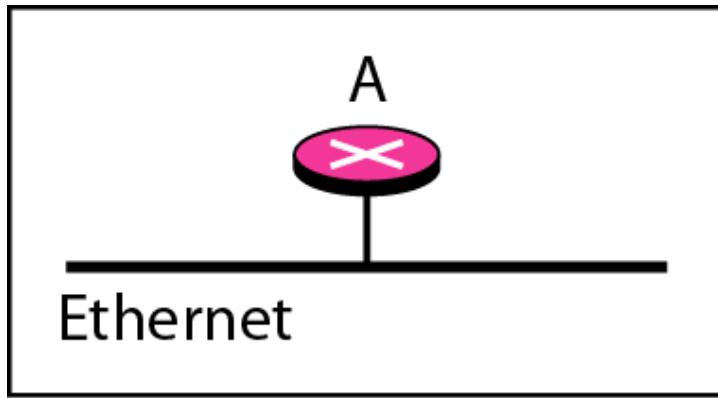


b. Unrealistic representation



c. Realistic representation

Figure 22.28 *Stub link*

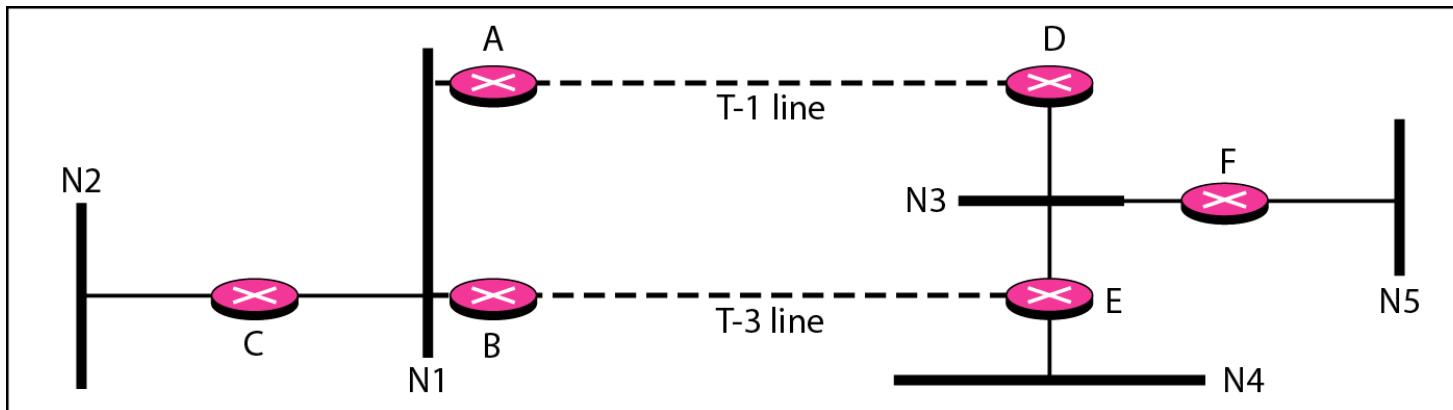


a. Stub network

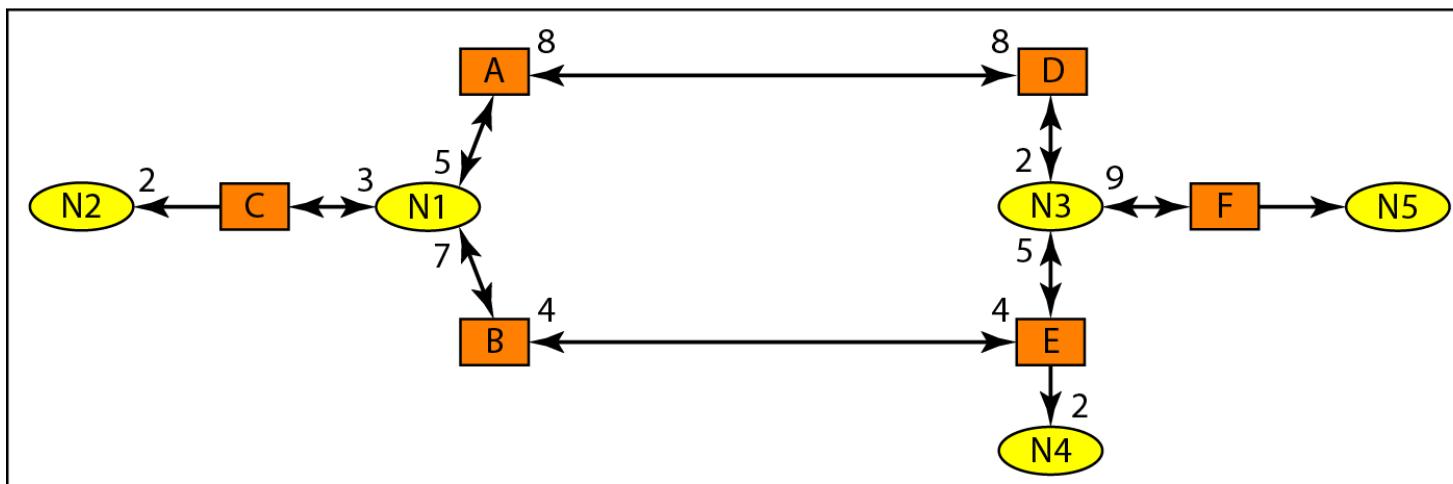


b. Representation

Figure 22.29 Example of an AS and its graphical representation in OSPF



a. Autonomous system



b. Graphical representation

Figure 22.30 Initial routing tables in path vector routing

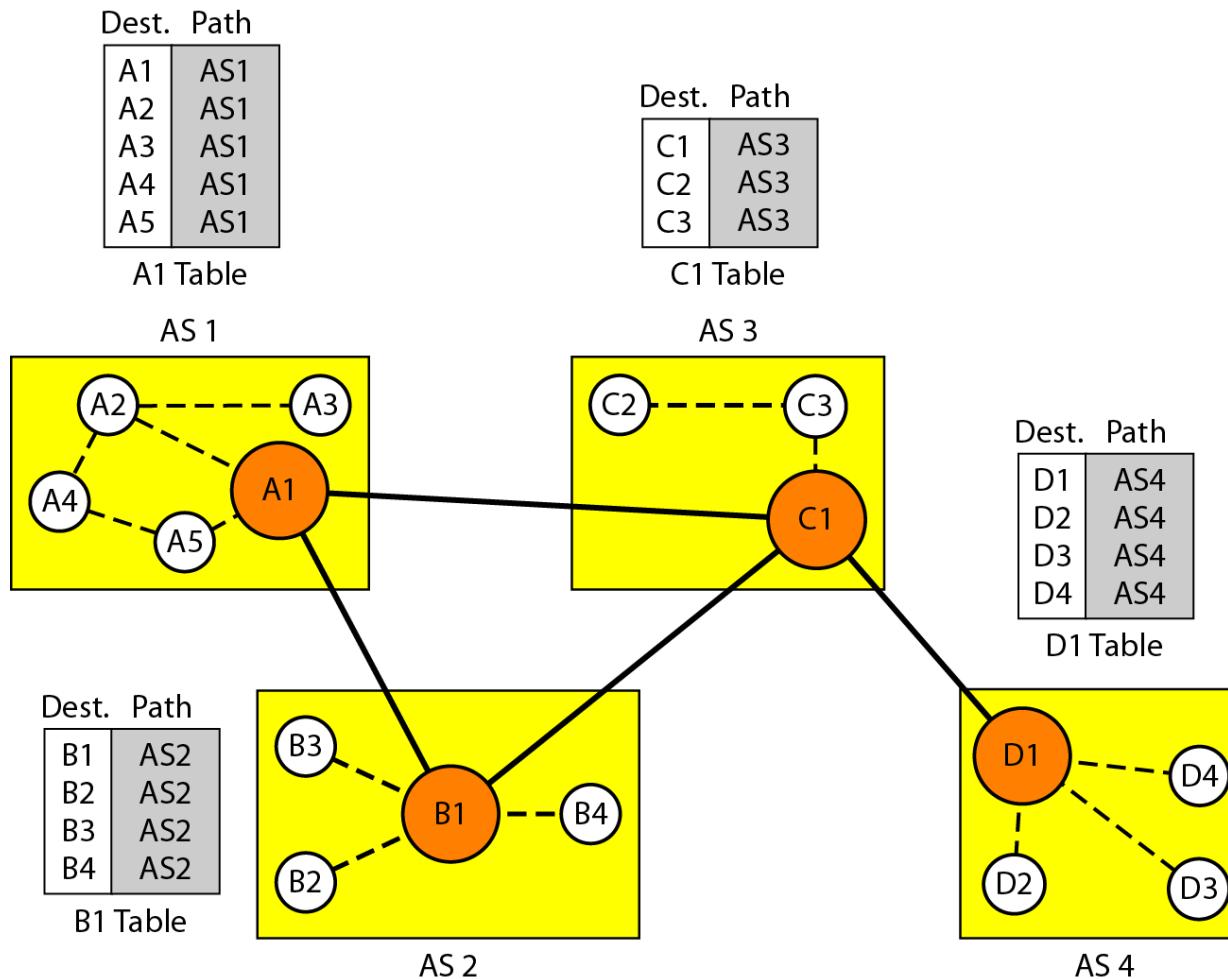


Figure 22.31 Stabilized tables for three autonomous systems

Dest.	Path
A1	AS1
...	
A5	AS1
B1	AS1-AS2
...	
B4	AS1-AS2
C1	AS1-AS3
...	
C3	AS1-AS3
D1	AS1-AS2-AS4
...	
D4	AS1-AS2-AS4

A1 Table

Dest.	Path
A1	AS2-AS1
...	
A5	AS2-AS1
B1	AS2
...	
B4	AS2
C1	AS2-AS3
...	
C3	AS2-AS3
D1	AS2-AS3-AS4
...	
D4	AS2-AS3-AS4

B1 Table

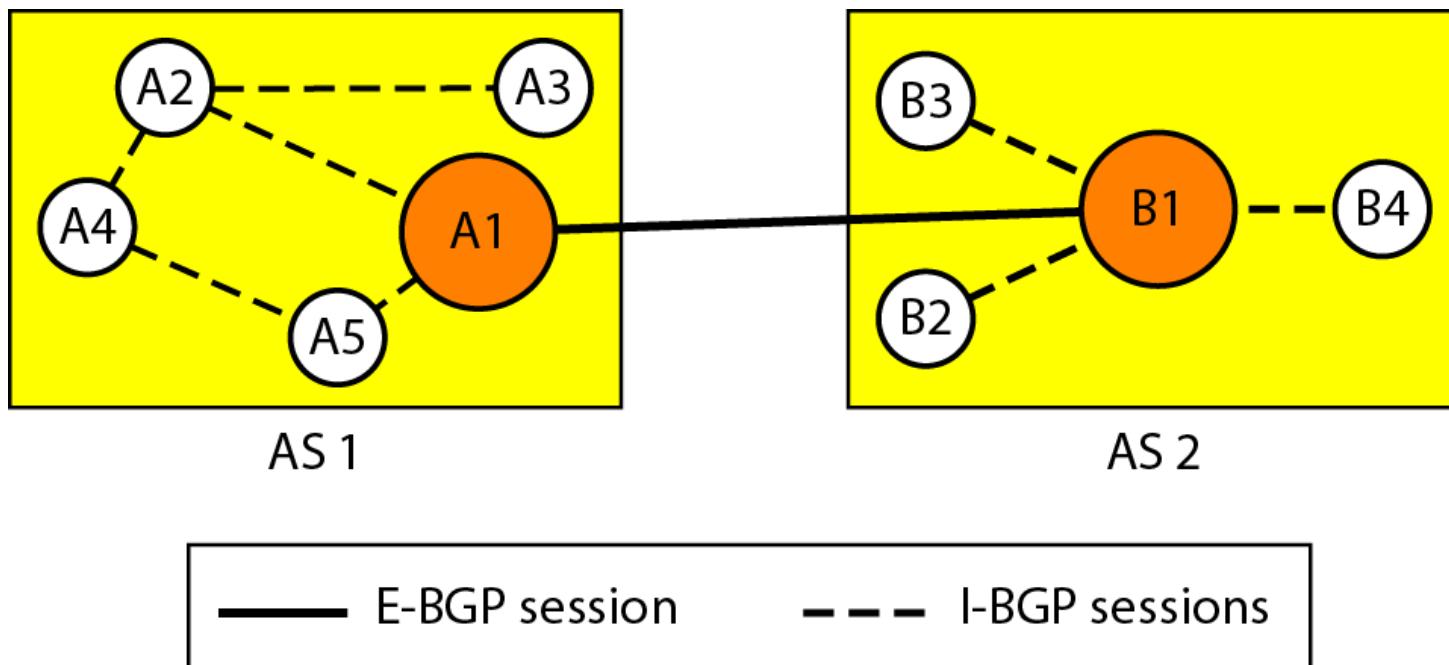
Dest.	Path
A1	AS3-AS1
...	
A5	AS3-AS1
B1	AS3-AS2
...	
B4	AS3-AS2
C1	AS3
...	
C3	AS3
D1	AS3-AS4
...	
D4	AS3-AS4

C1 Table

Dest.	Path
A1	AS4-AS3-AS1
...	
A5	AS4-AS3-AS1
B1	AS4-AS3-AS2
...	
B4	AS4-AS3-AS2
C1	AS4-AS3
...	
C3	AS4-AS3
D1	AS4
...	
D4	AS4

D1 Table

Figure 22.32 Internal and external BGP sessions



22-4 MULTICAST ROUTING PROTOCOLS

In this section, we discuss multicasting and multicast routing protocols.

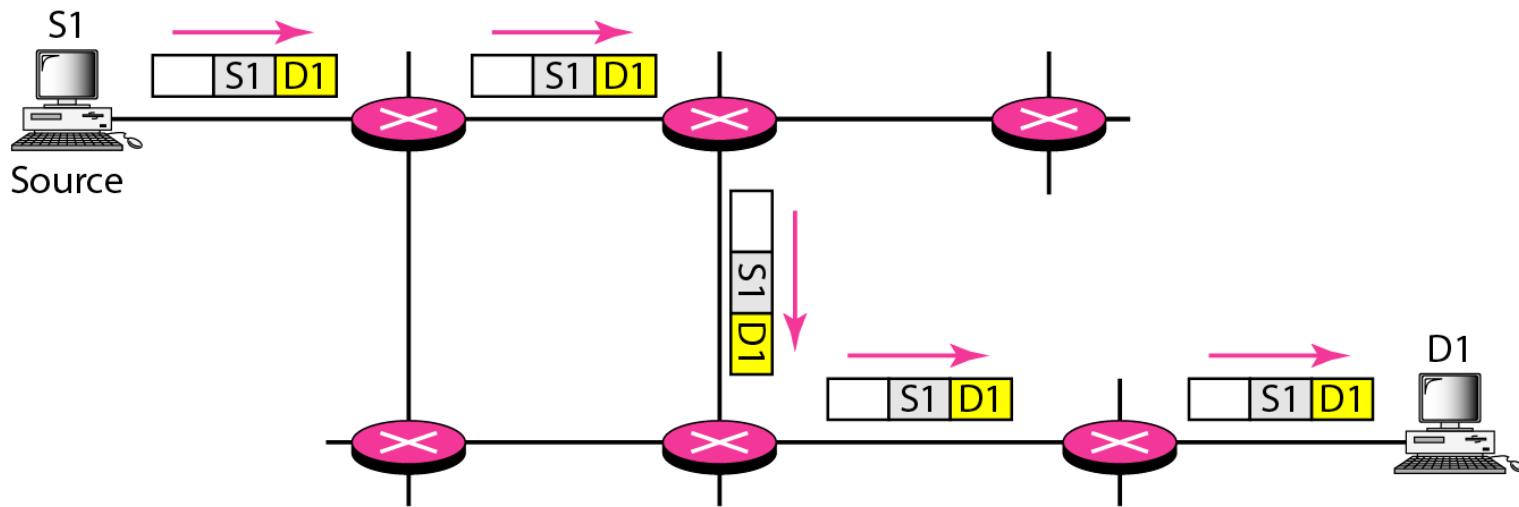
Topics discussed in this section:

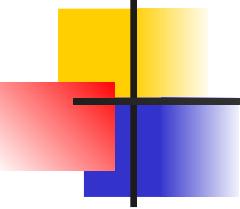
Unicast, Multicast, and Broadcast

Applications

**Multicast Routing
Routing Protocols**

Figure 22.33 Unicasting

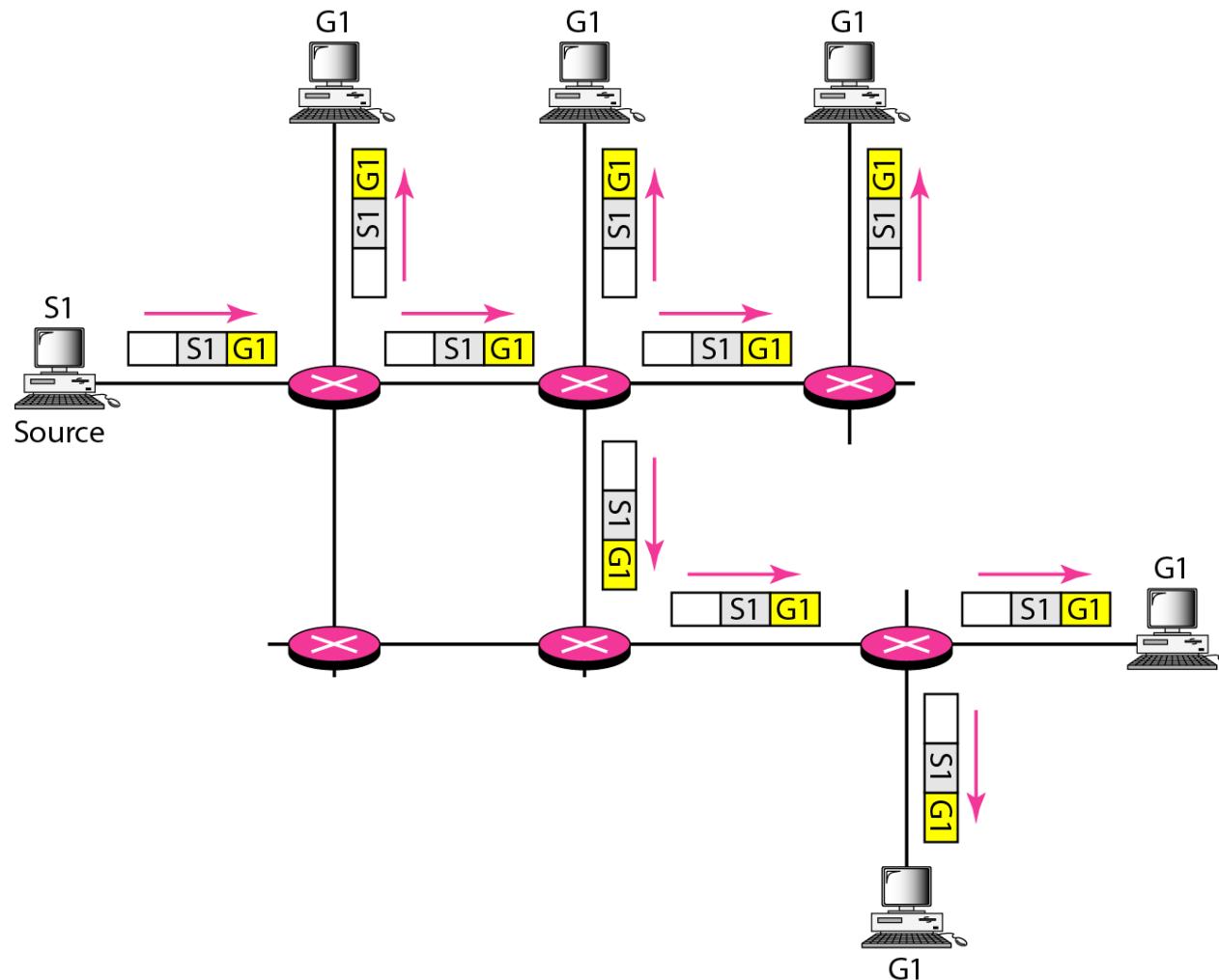


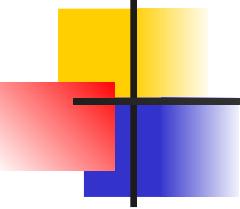


Note

In unicasting, the router forwards the received packet through only one of its interfaces.

Figure 22.34 Multicasting

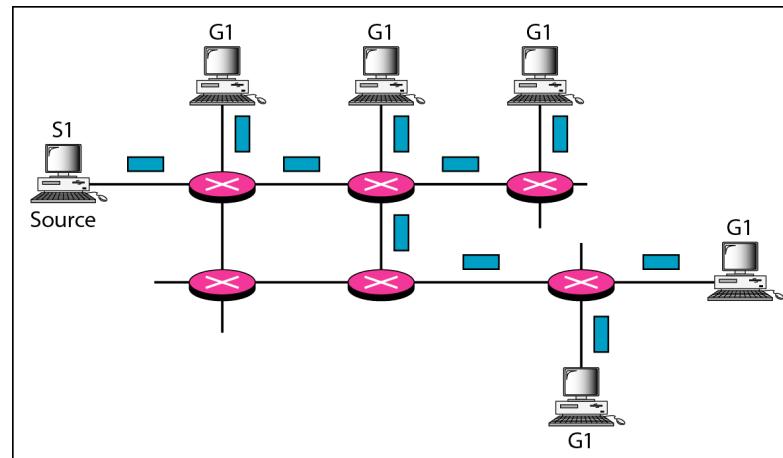




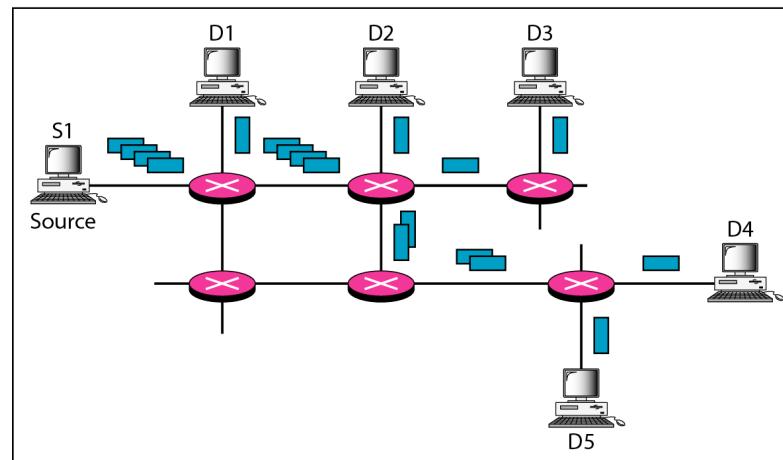
Note

In multicasting, the router may forward the received packet through several of its interfaces.

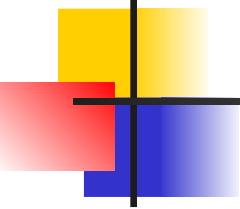
Figure 22.35 *Multicasting versus multiple unicasting*



a. Multicasting

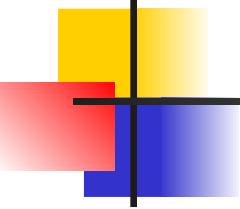


b. Multiple unicasting



Note

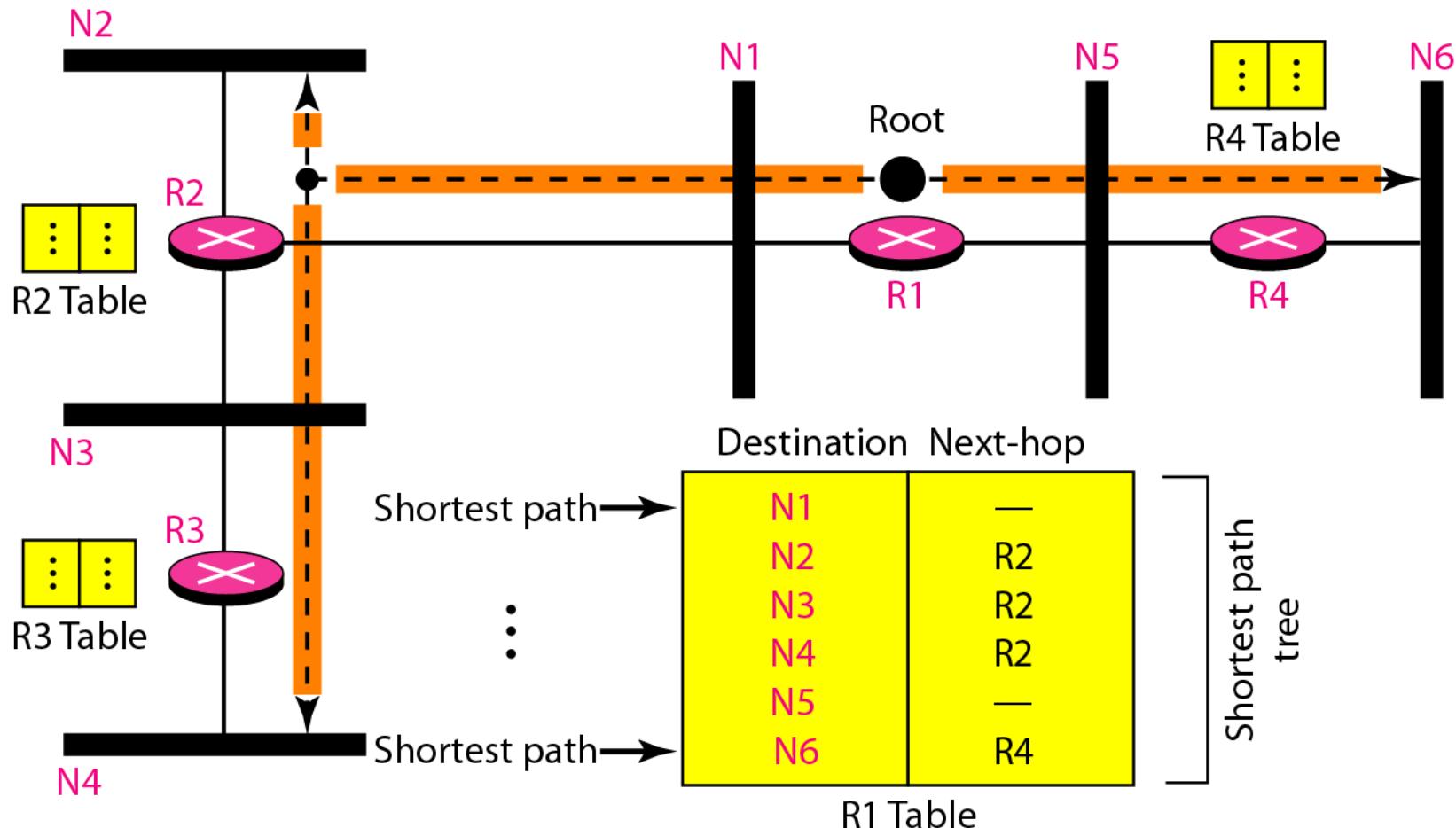
Emulation of multicasting through multiple unicasting is not efficient and may create long delays, particularly with a large group.

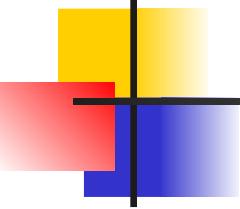


Note

In unicast routing, each router in the domain has a table that defines a shortest path tree to possible destinations.

Figure 22.36 Shortest path tree in unicast routing

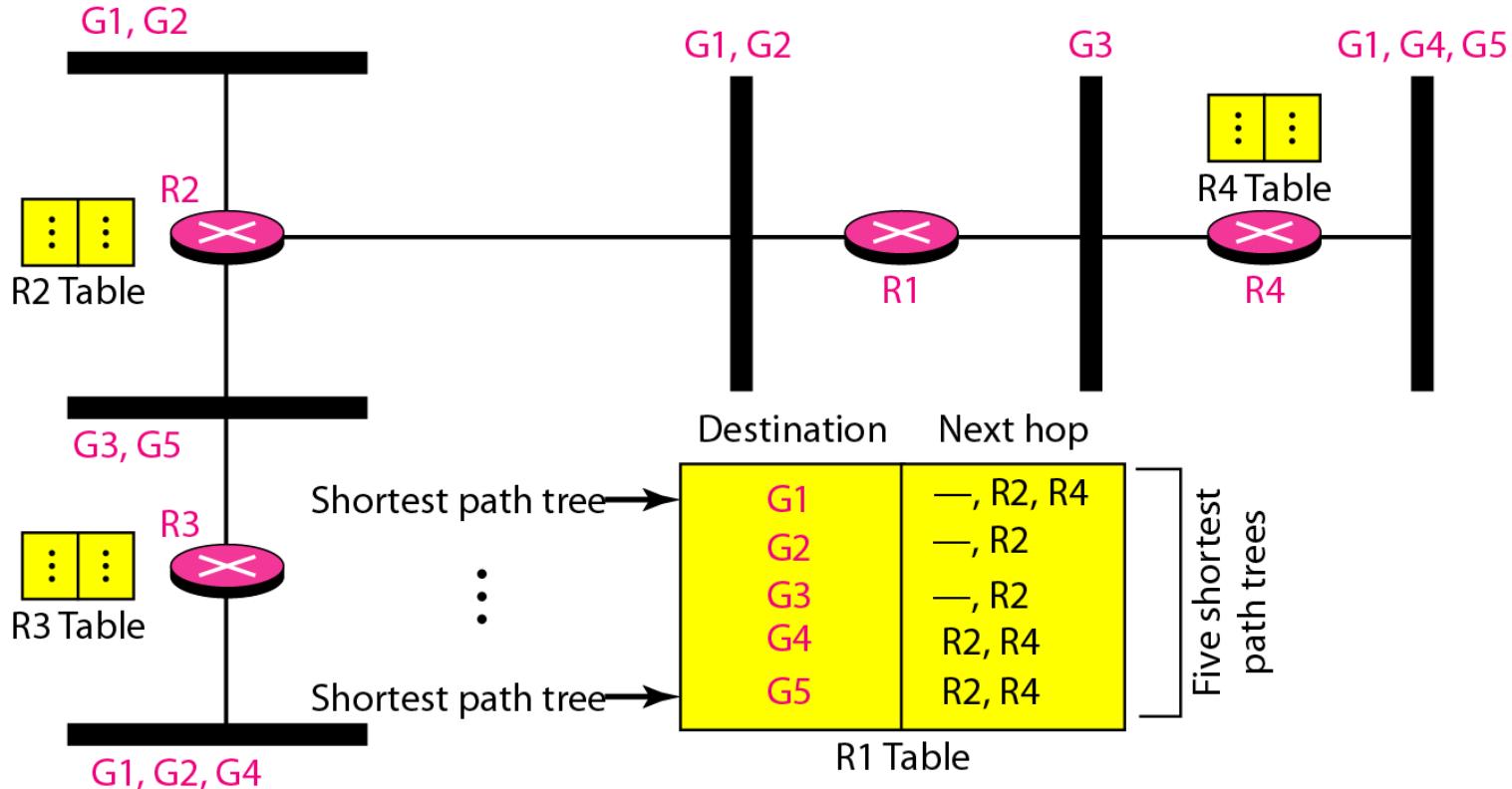


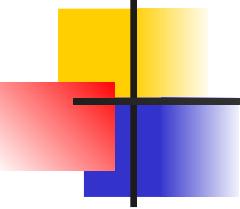


Note

In multicast routing, each involved router needs to construct a shortest path tree for each group.

Figure 22.37 Source-based tree approach

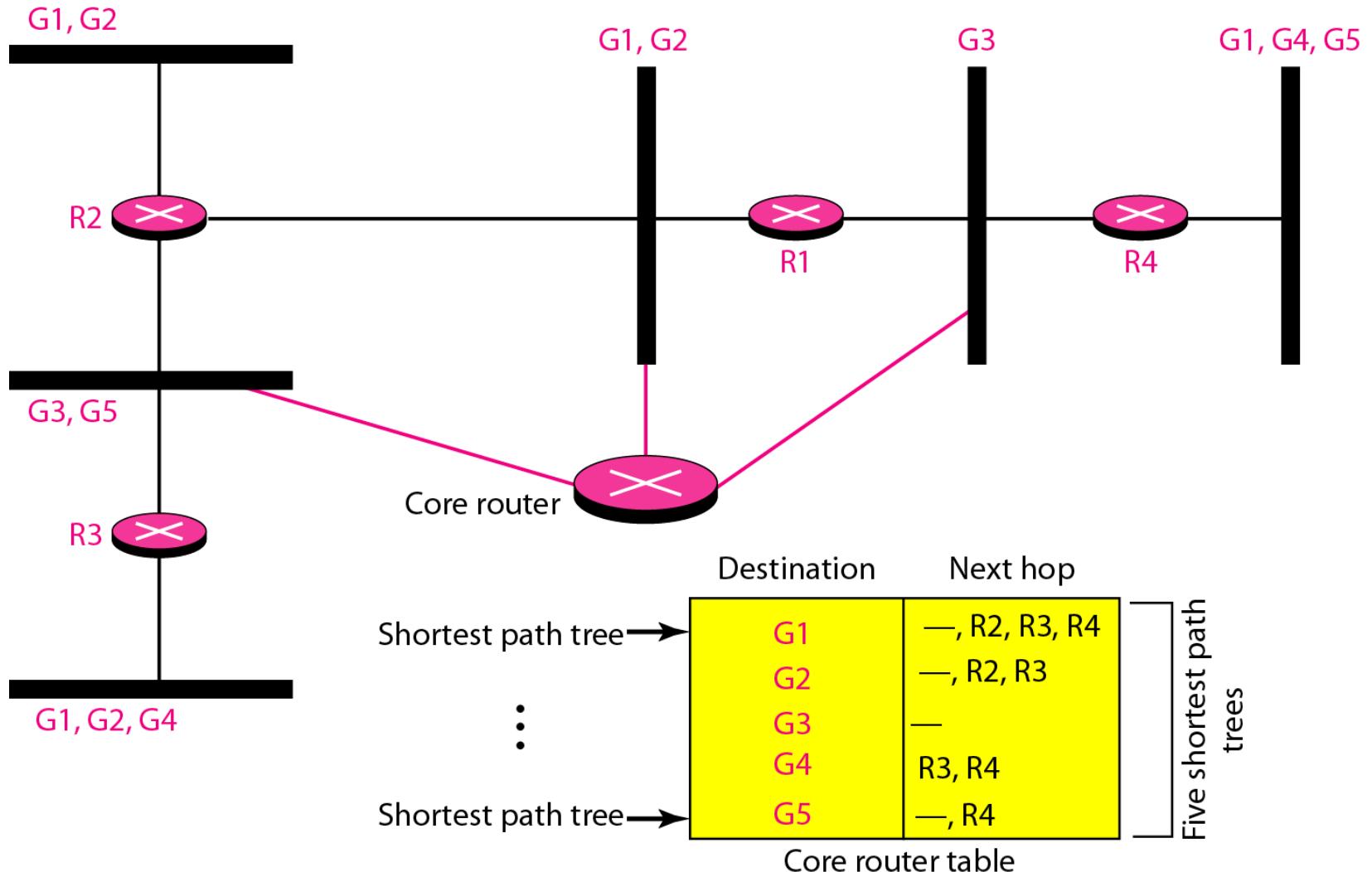


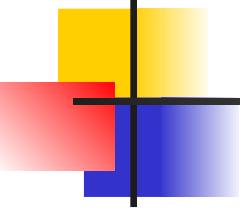


Note

In the source-based tree approach, each router needs to have one shortest path tree for each group.

Figure 22.38 Group-shared tree approach

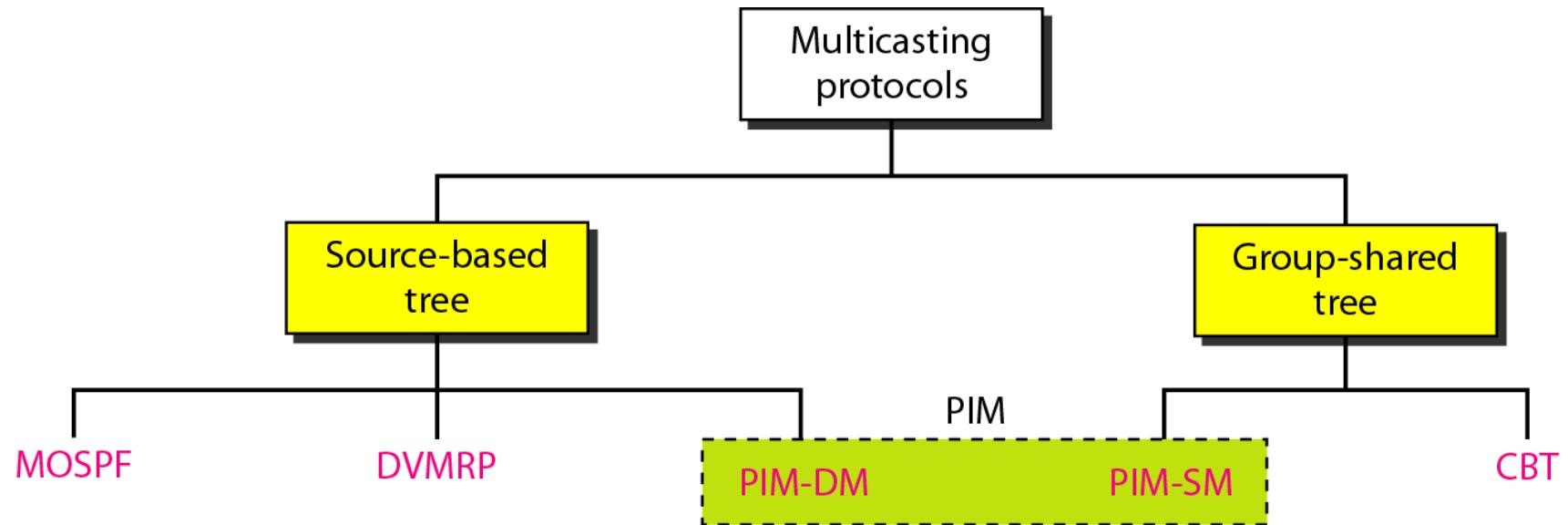


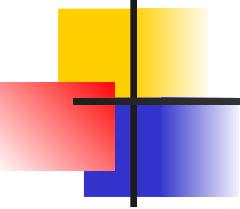


Note

In the group-shared tree approach, only the core router, which has a shortest path tree for each group, is involved in multicasting.

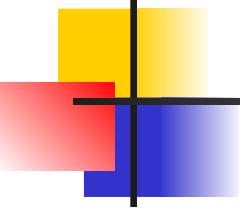
Figure 22.39 Taxonomy of common multicast protocols





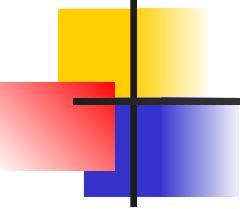
Note

Multicast link state routing uses the source-based tree approach.



Note

Flooding broadcasts packets, but creates loops in the systems.



Note

RPF eliminates the loop in the flooding process.

Figure 22.40 Reverse path forwarding (RPF)

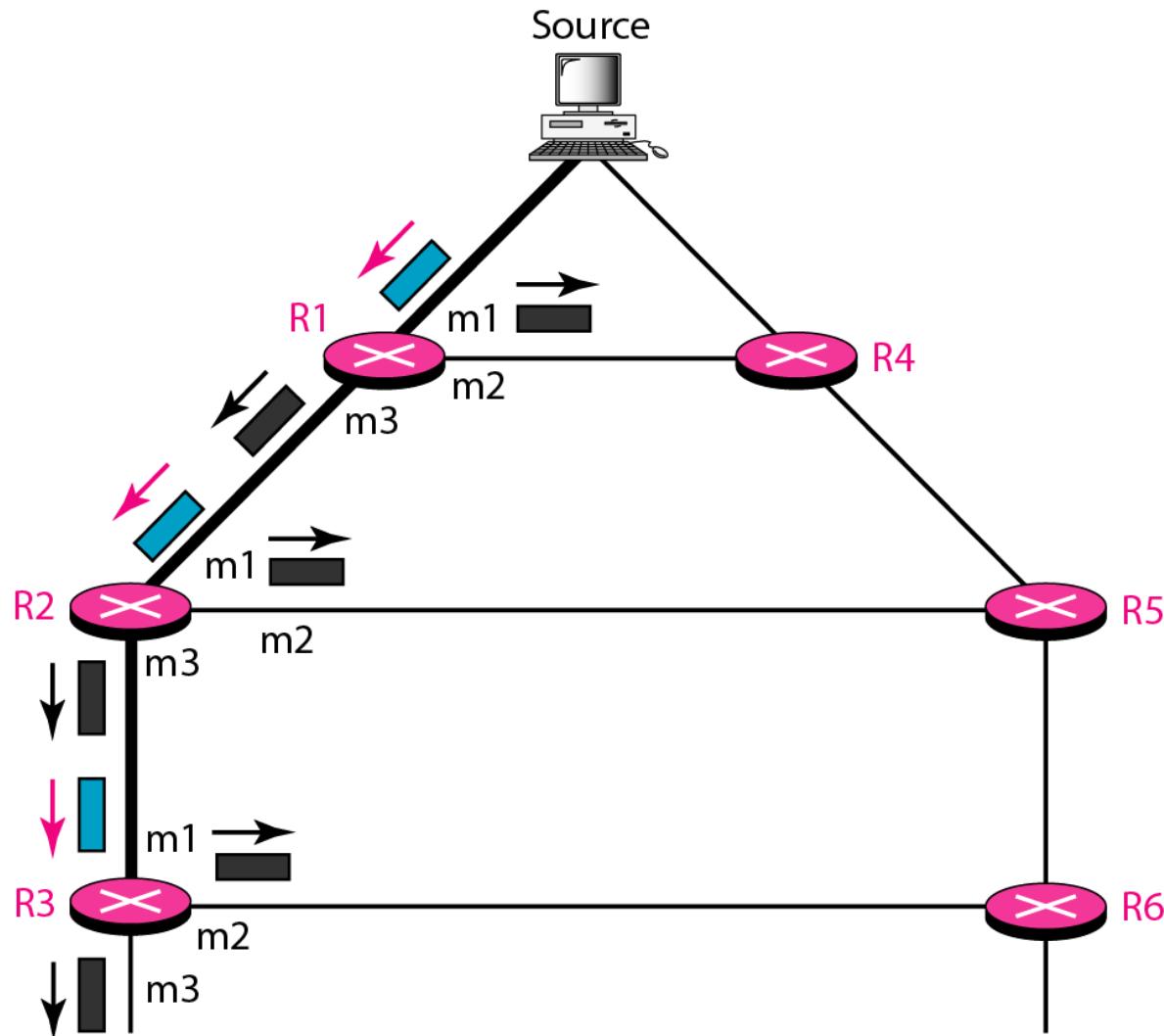


Figure 22.41 Problem with RPF

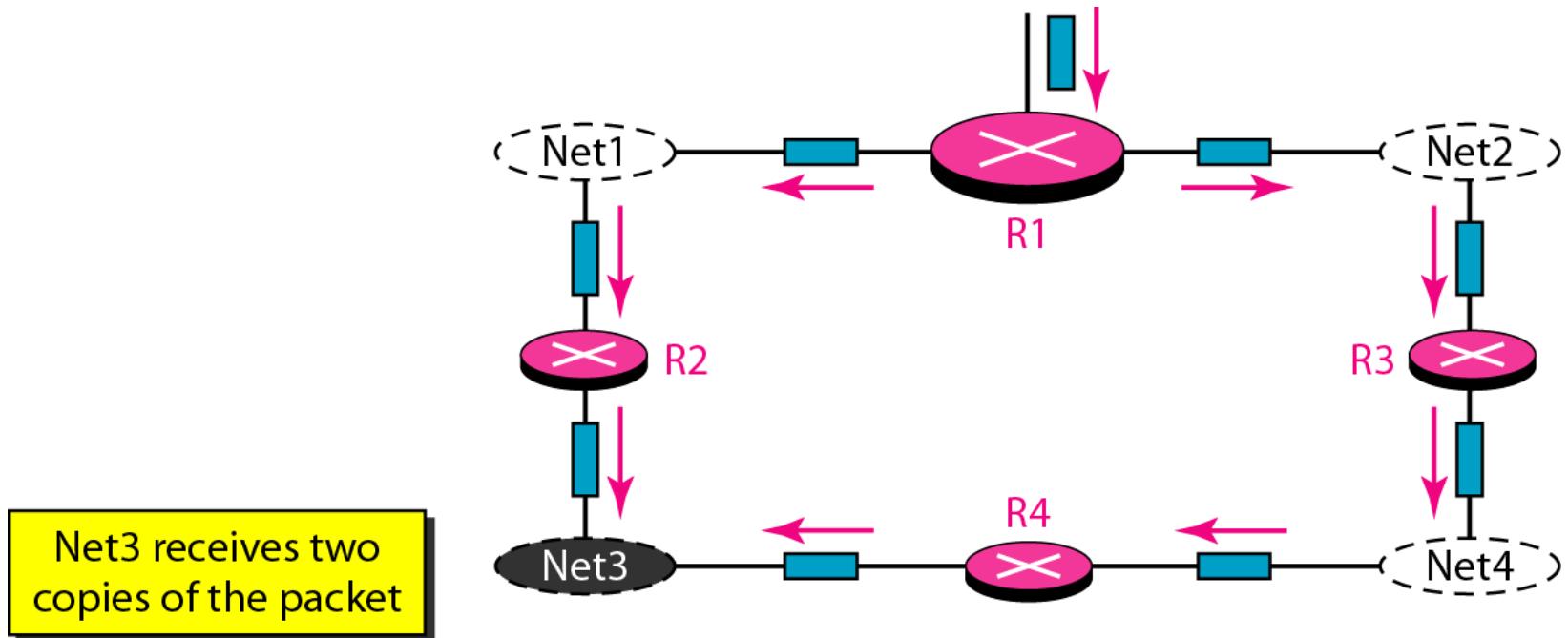
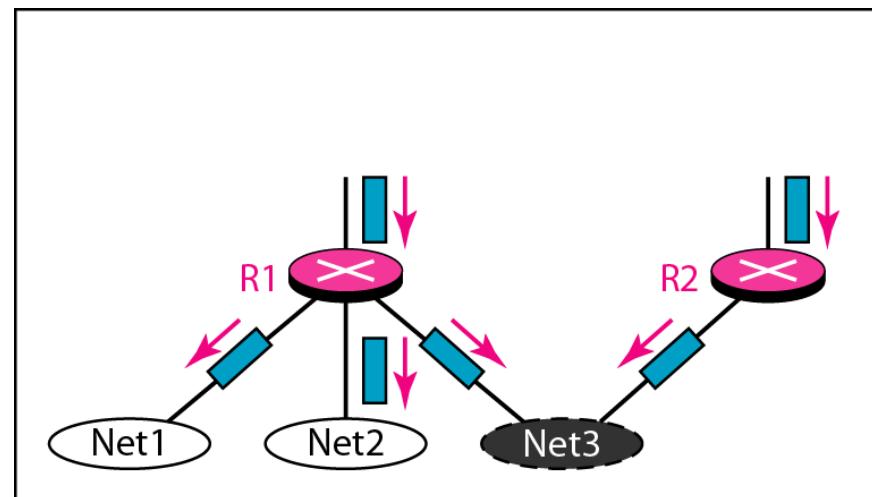
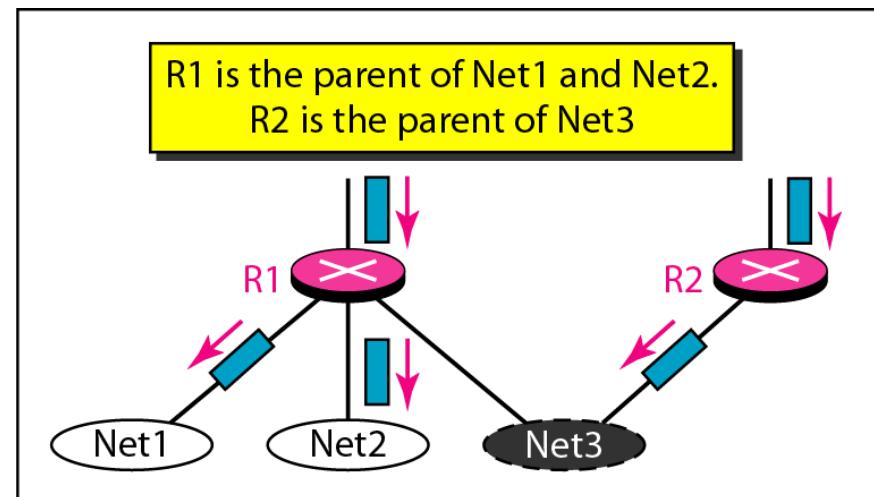


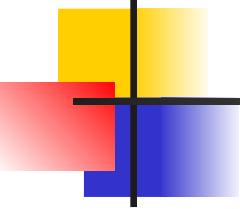
Figure 22.42 RPF Versus RPB



a. RPF



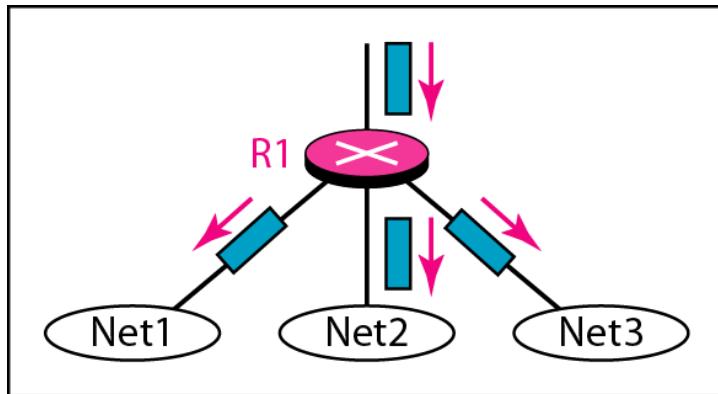
b. RPB



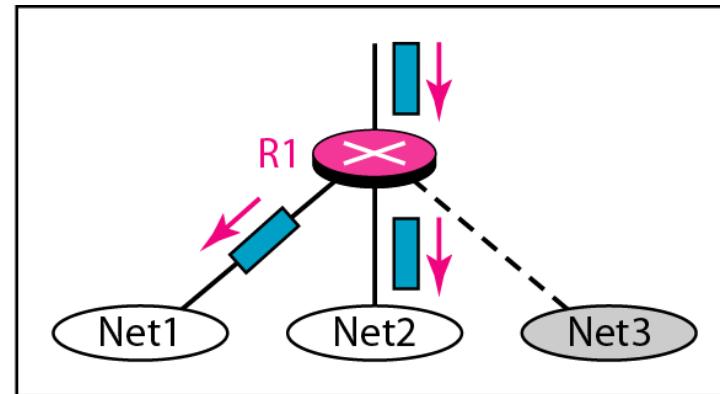
Note

RPB creates a shortest path broadcast tree from the source to each destination. It guarantees that each destination receives one and only one copy of the packet.

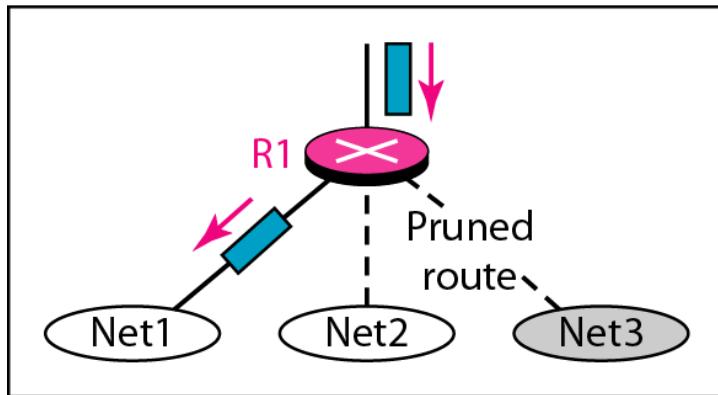
Figure 22.43 RPF, RPB, and RPM



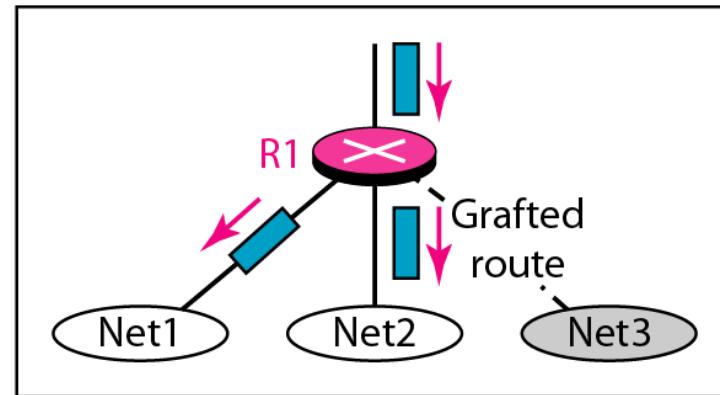
a. RPF



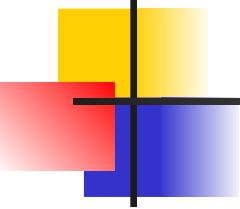
b. RPB



c. RPM (after pruning)



d. RPM (after grafting)



Note

**RPM adds pruning and grafting to RPB
to create a multicast shortest
path tree that supports dynamic
membership changes.**

Figure 22.44 Group-shared tree with rendezvous router

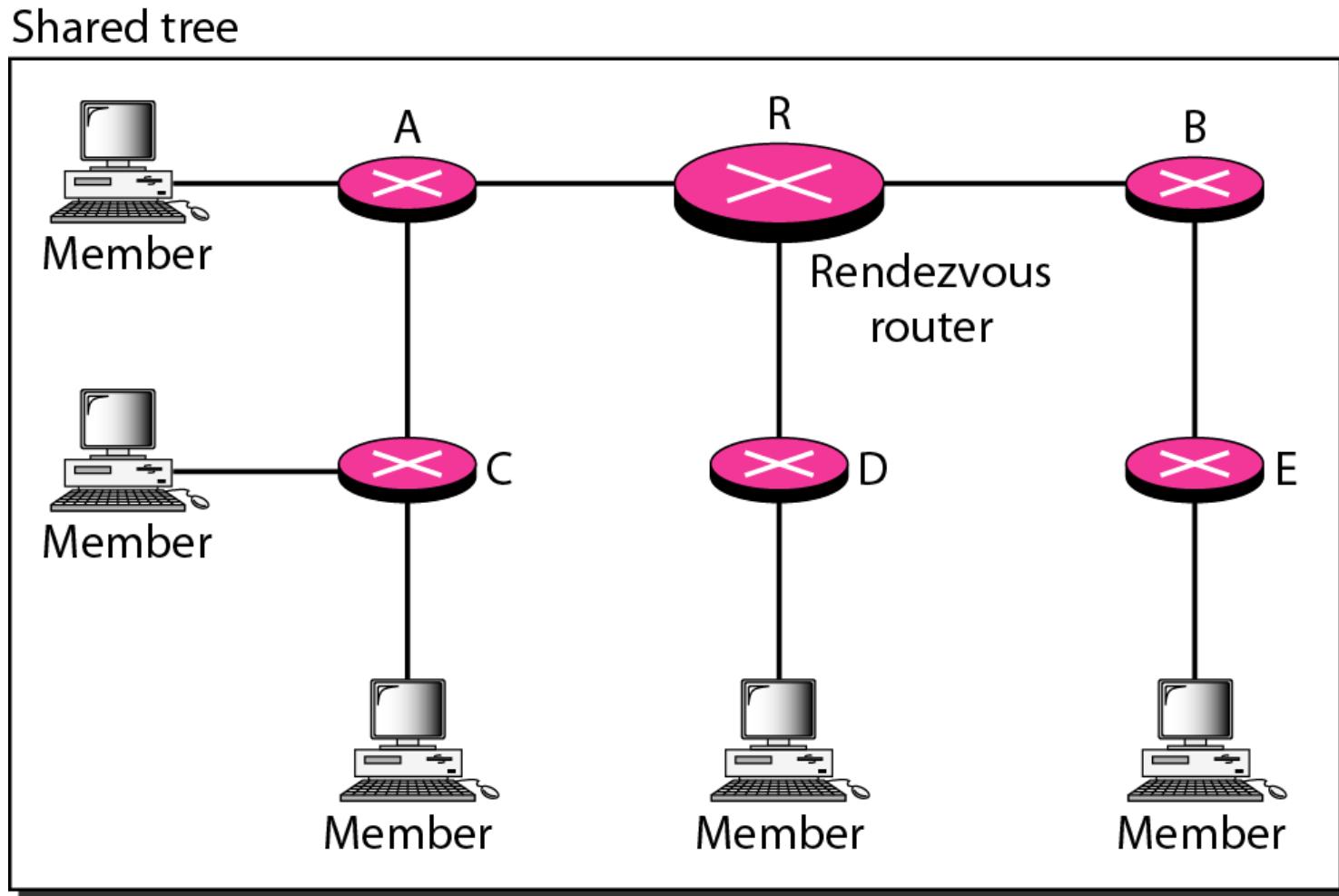
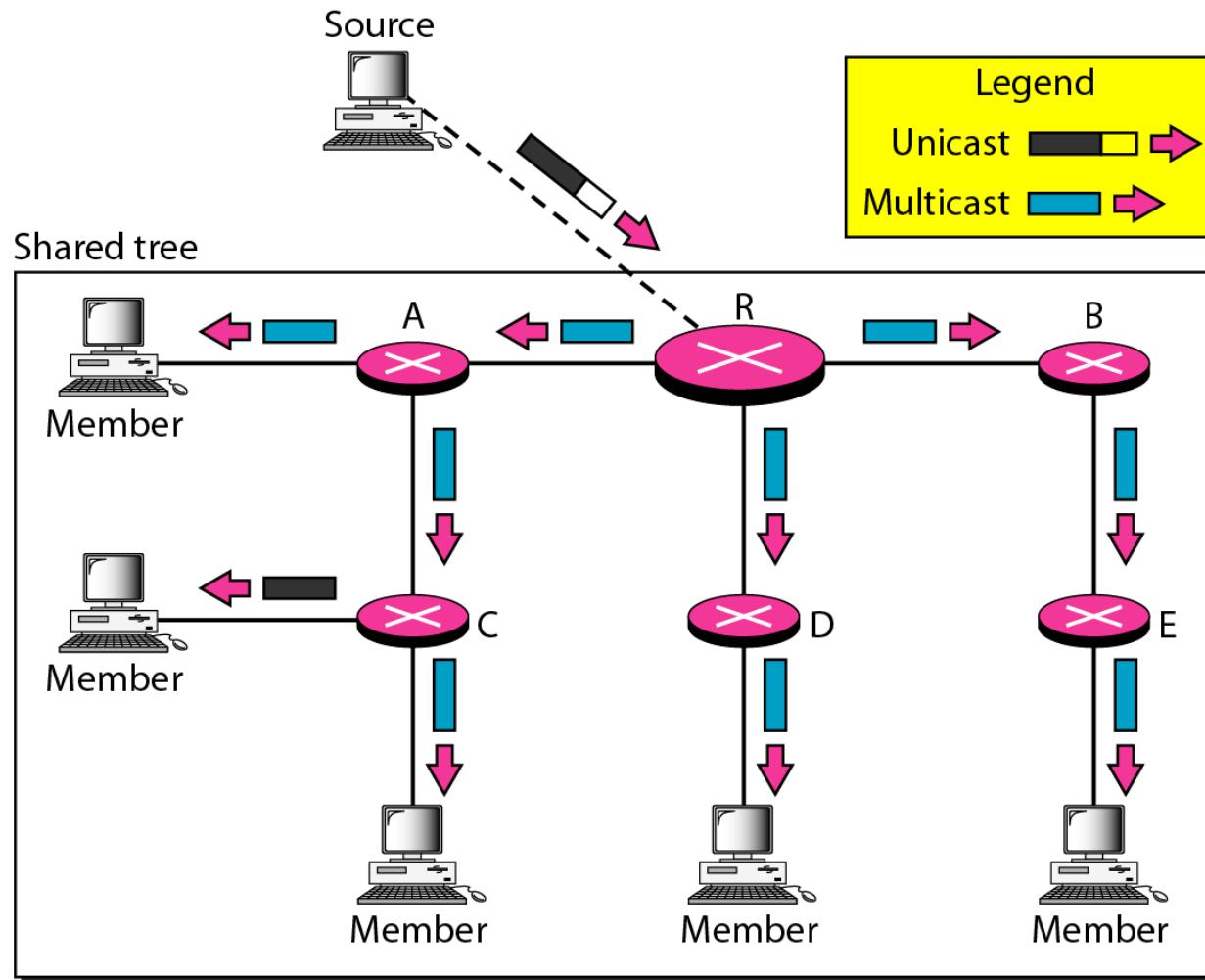
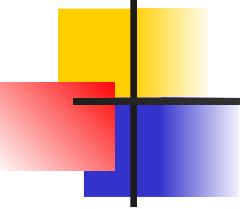


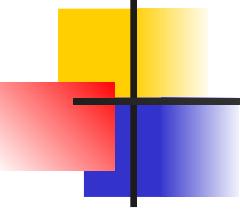
Figure 22.45 *Sending a multicast packet to the rendezvous router*





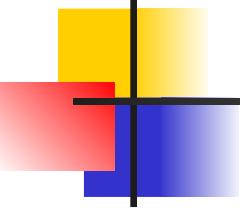
Note

In CBT, the source sends the multicast packet (encapsulated in a unicast packet) to the core router. The core router decapsulates the packet and forwards it to all interested interfaces.



Note

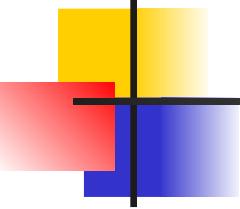
PIM-DM is used in a dense multicast environment, such as a LAN.



Note

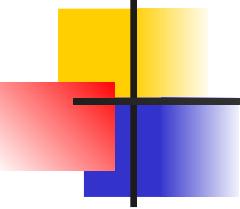
PIM-DM uses RPF and pruning and grafting strategies to handle multicasting.

However, it is independent of the underlying unicast protocol.



Note

PIM-SM is used in a sparse multicast environment such as a WAN.



Note

PIM-SM is similar to CBT but uses a simpler procedure.

Figure 22.46 Logical tunneling

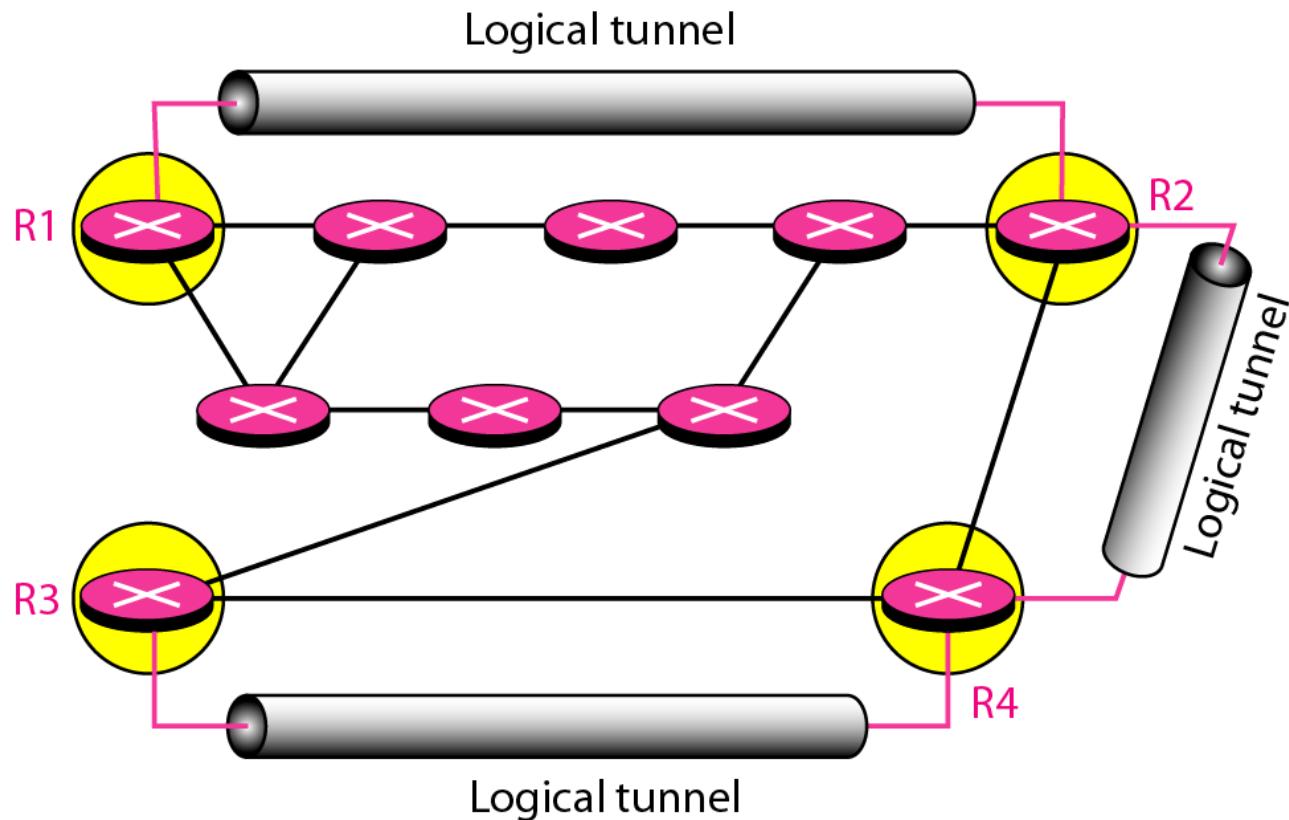
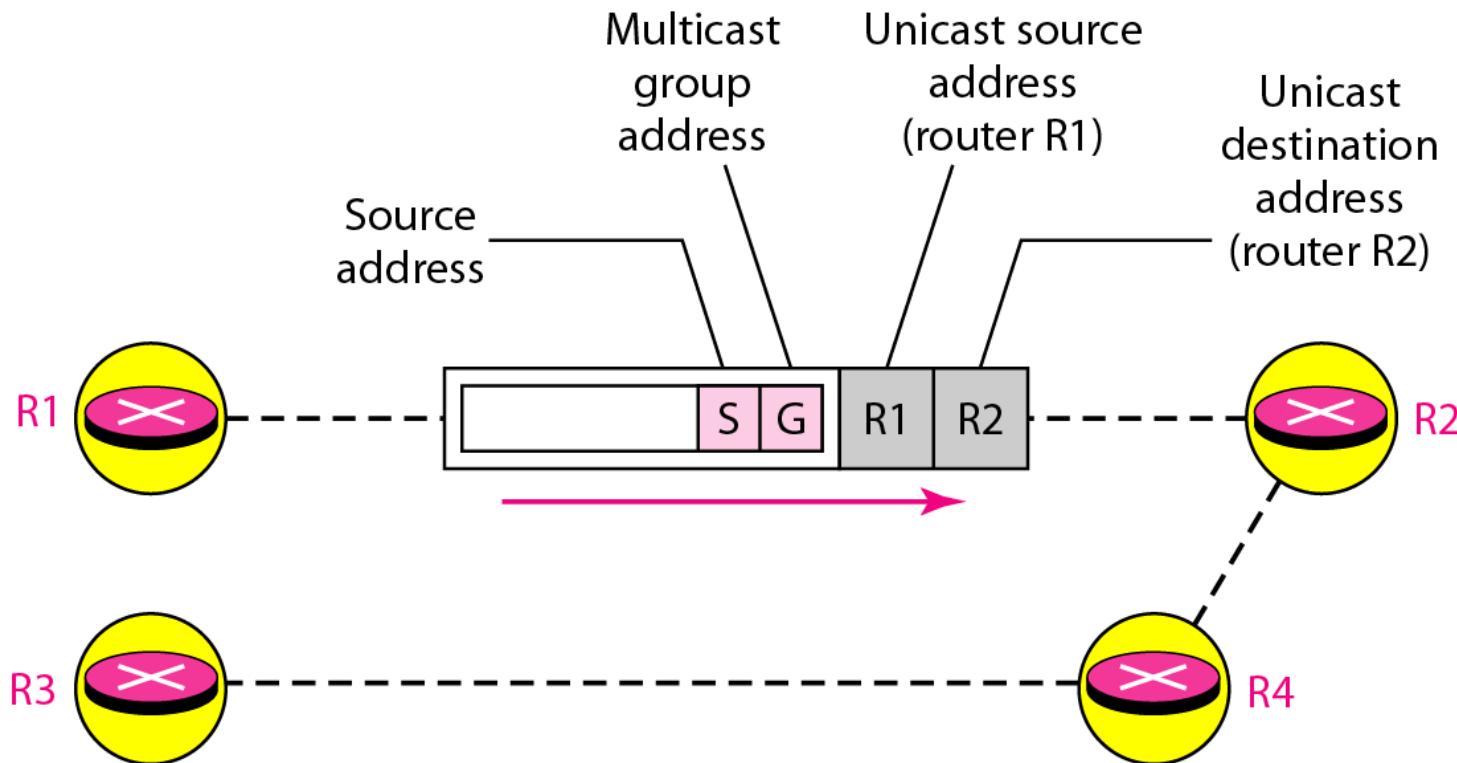


Figure 22.47 MBONE





Data Communications and Networking

Fourth Edition

Forouzan

Chapter 23

Process-to-Process Delivery: UDP, TCP, and SCTP

23-1 PROCESS-TO-PROCESS DELIVERY

The transport layer is responsible for process-to-process delivery—the delivery of a packet, part of a message, from one process to another. Two processes communicate in a client/server relationship, as we will see later.

Topics discussed in this section:

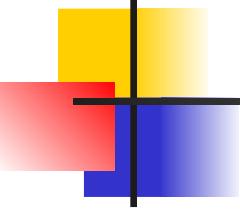
Client/Server Paradigm

Multiplexing and Demultiplexing

Connectionless Versus Connection-Oriented Service

Reliable Versus Unreliable

Three Protocols



Note

The transport layer is responsible for process-to-process delivery.

Figure 23.1 Types of data deliveries

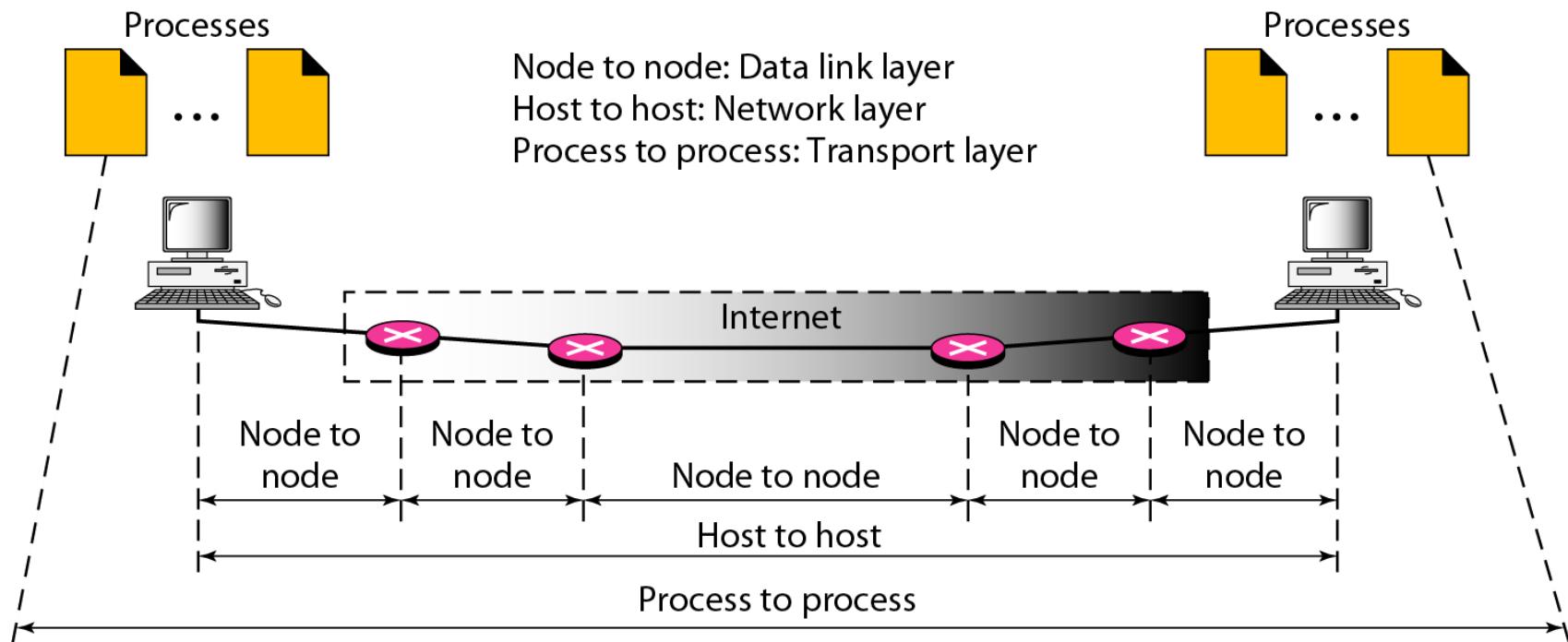


Figure 23.2 Port numbers

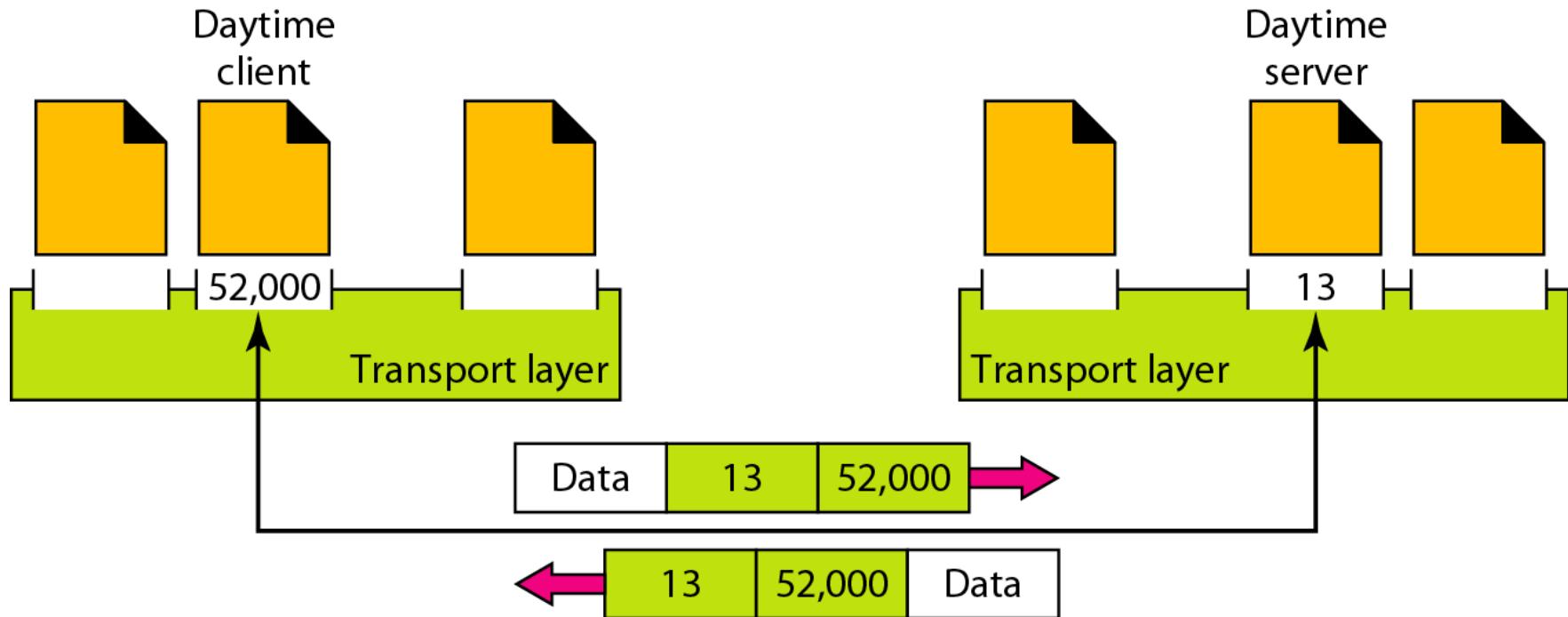


Figure 23.3 IP addresses versus port numbers

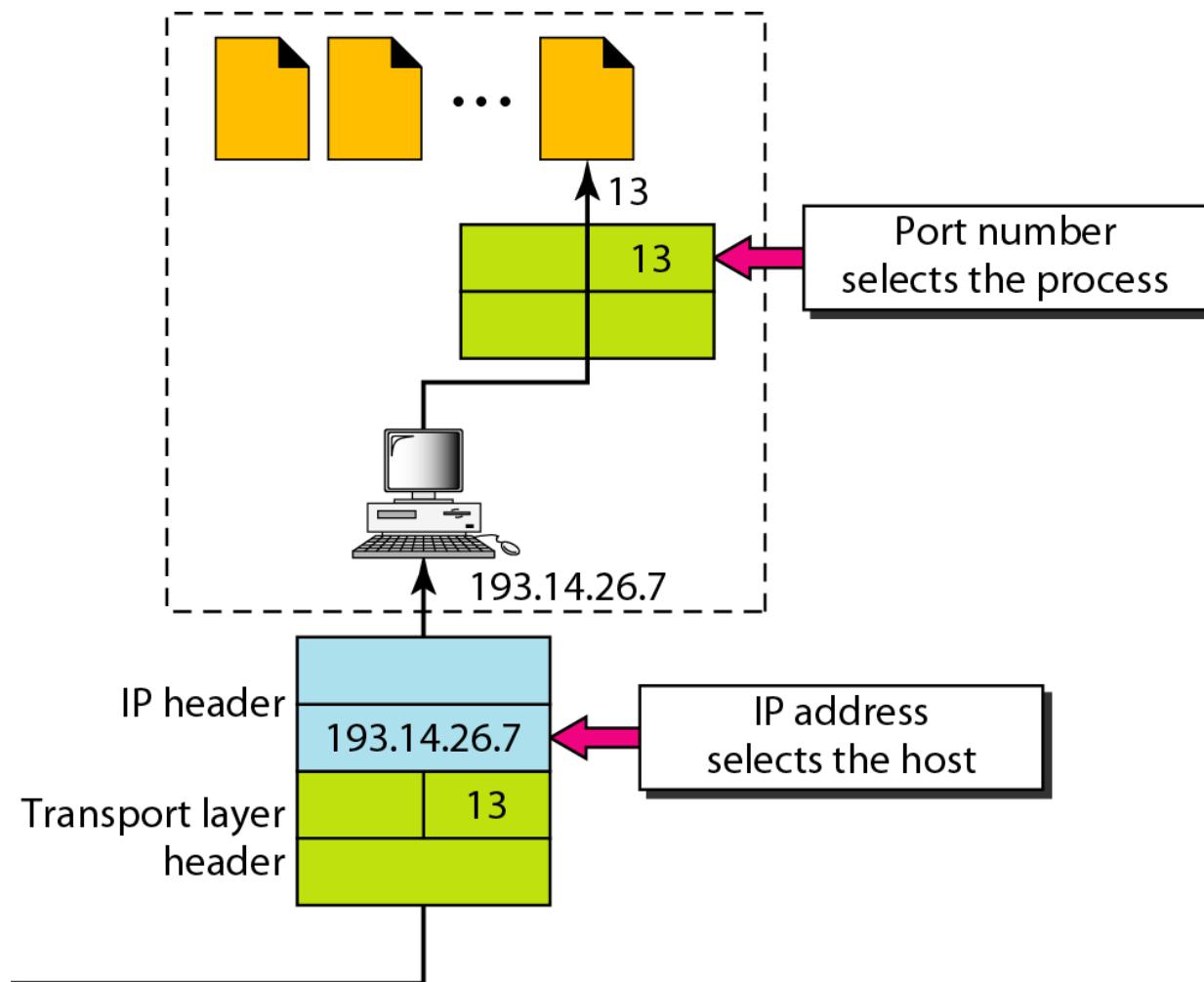


Figure 23.4 IANA ranges

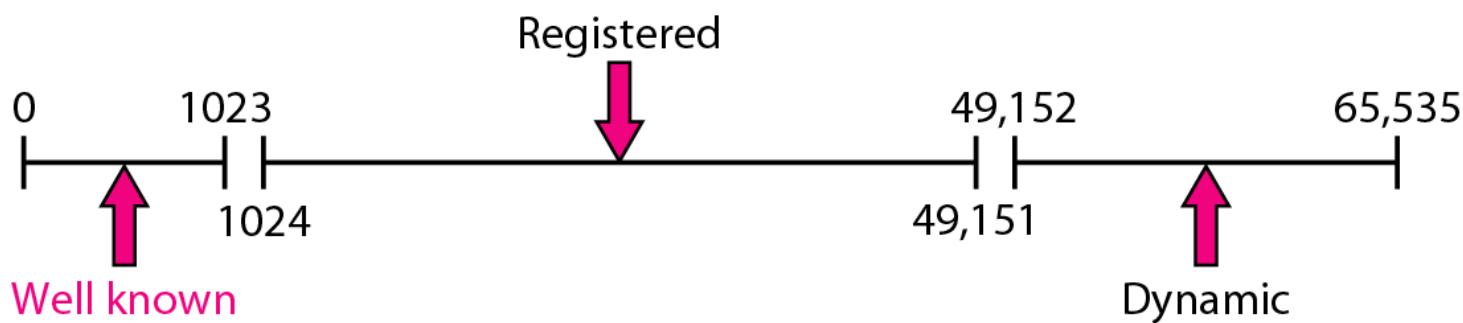


Figure 23.5 *Socket address*

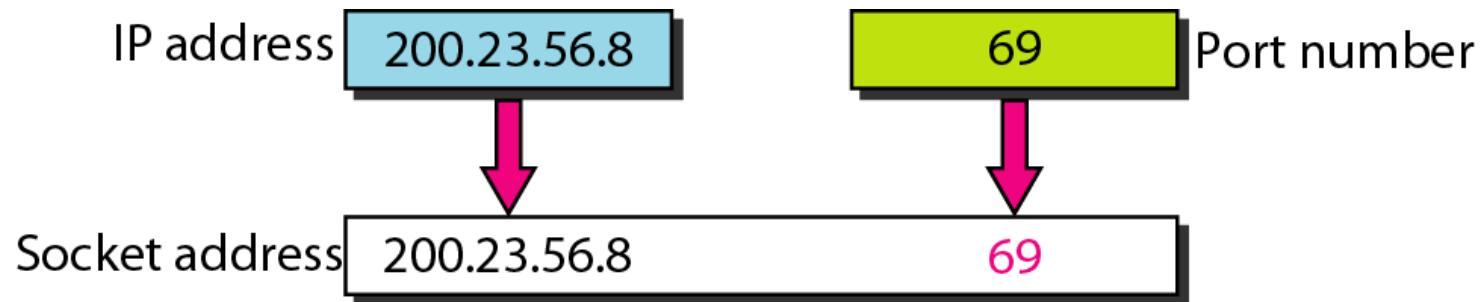


Figure 23.6 Multiplexing and demultiplexing

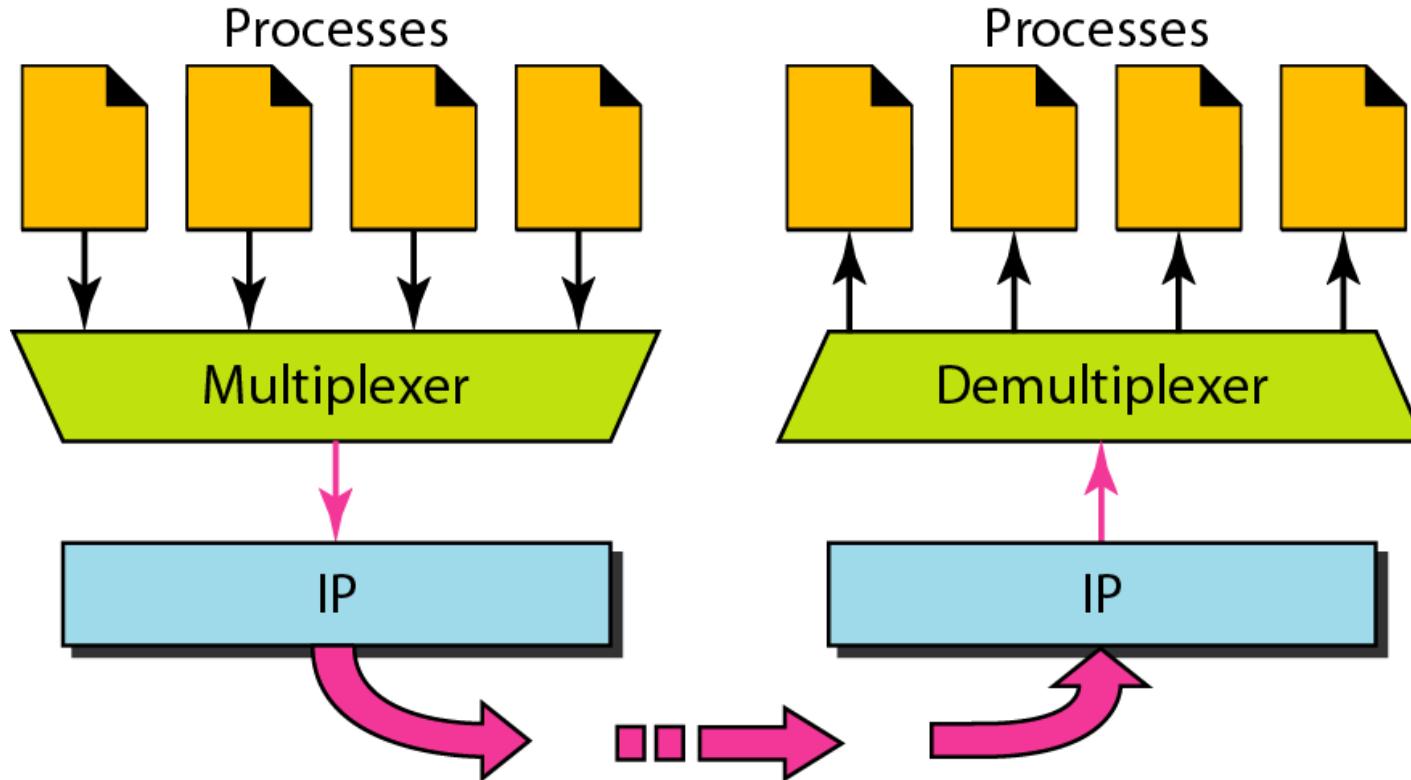


Figure 23.7 Error control

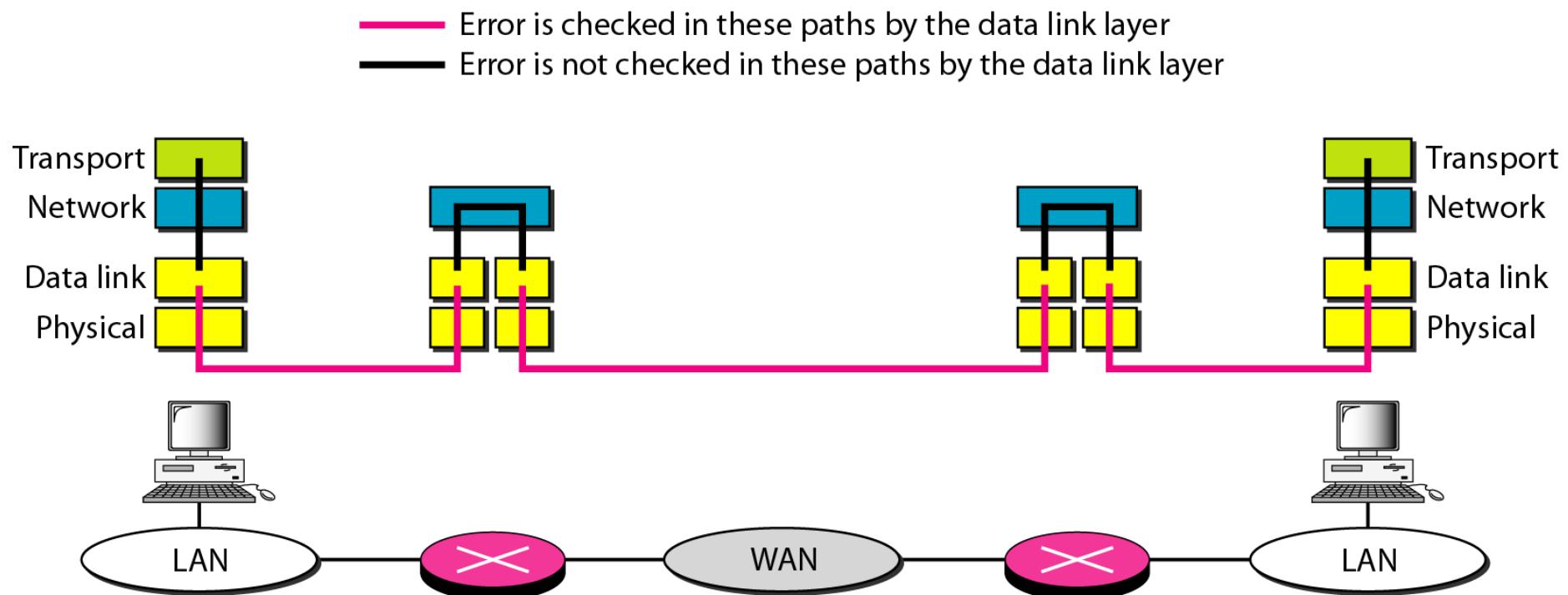
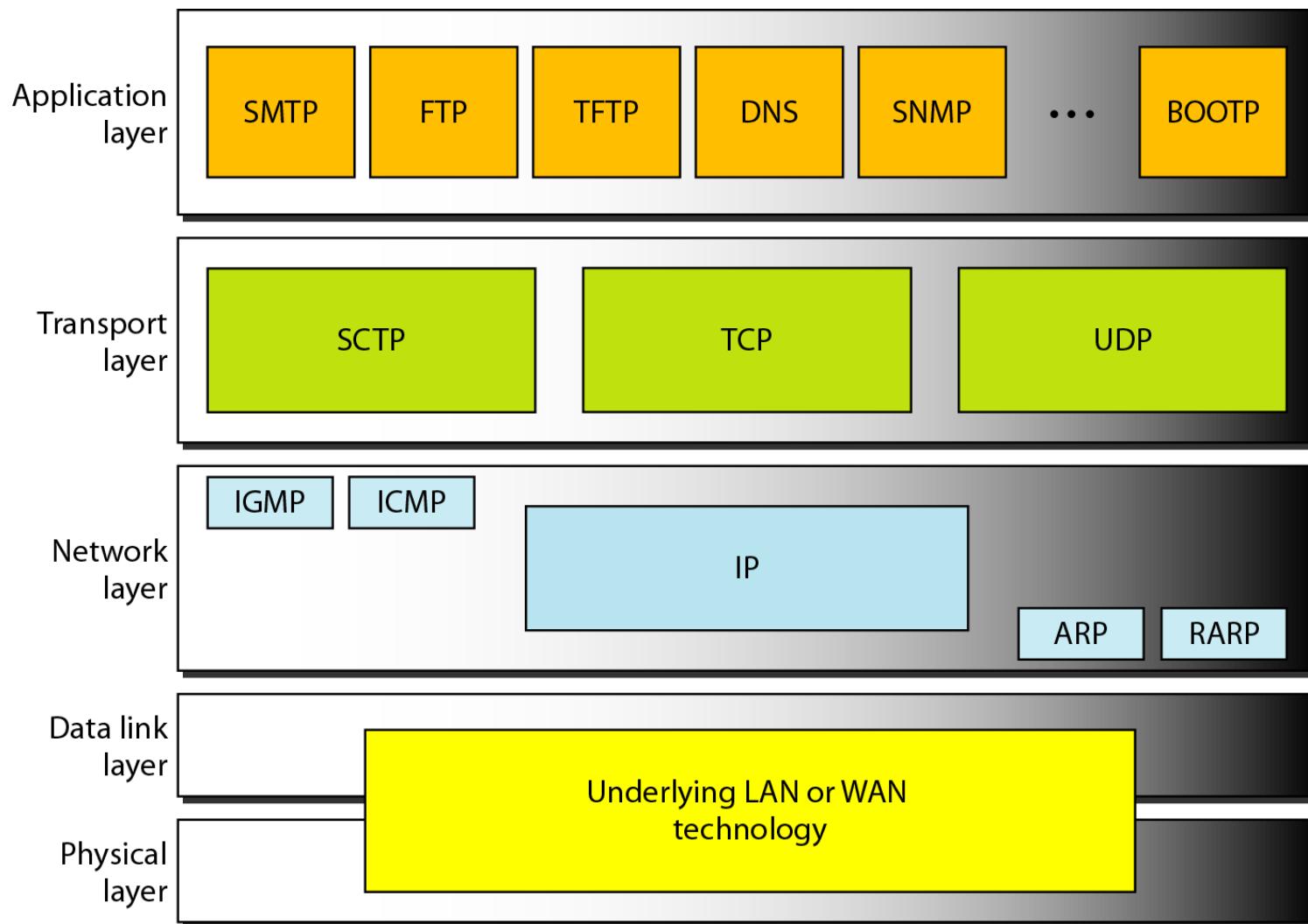


Figure 23.8 Position of UDP, TCP, and SCTP in TCP/IP suite



23-2 USER DATAGRAM PROTOCOL (UDP)

The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.

Topics discussed in this section:

Well-Known Ports for UDP

User Datagram

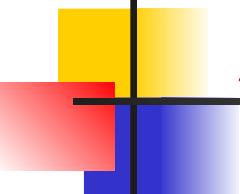
Checksum

UDP Operation

Use of UDP

Table 23.1 Well-known ports used with UDP

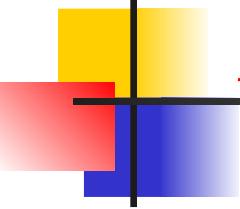
<i>Port</i>	<i>Protocol</i>	<i>Description</i>
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	BOOTPs	Server port to download bootstrap information
68	BOOTPc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)



Example 23.1

In UNIX, the well-known ports are stored in a file called /etc/services. Each line in this file gives the name of the server and the well-known port number. We can use the grep utility to extract the line corresponding to the desired application. The following shows the port for FTP. Note that FTP can use port 21 with either UDP or TCP.

```
$ grep ftp /etc/services
ftp          21/tcp
ftp          21/udp
```

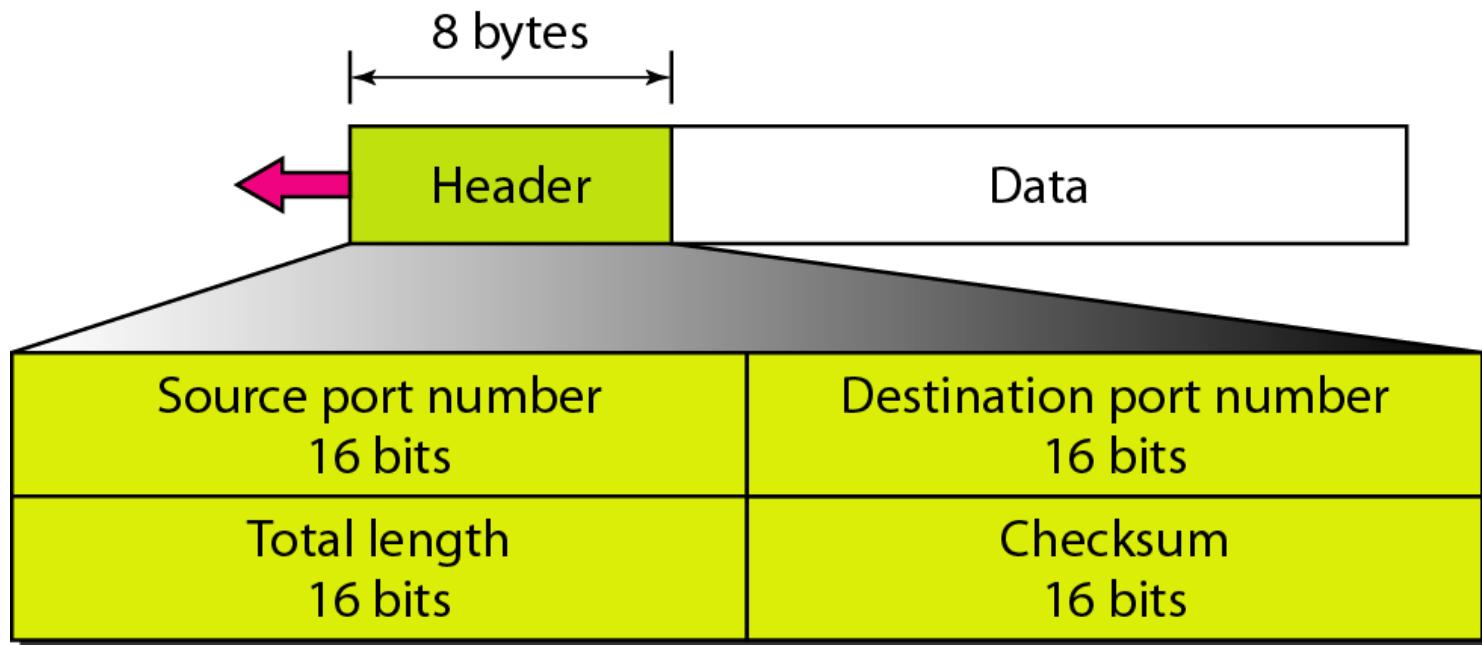


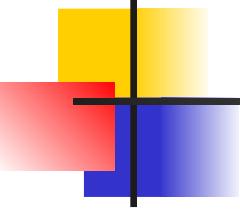
Example 23.1 (continued)

SNMP uses two port numbers (161 and 162), each for a different purpose, as we will see in Chapter 28.

```
$ grep snmp /etc/services
snmp          161/tcp      #Simple Net Mgmt Proto
snmp          161/udp      #Simple Net Mgmt Proto
snmptrap      162/udp      #Traps for SNMP
```

Figure 23.9 *User datagram format*

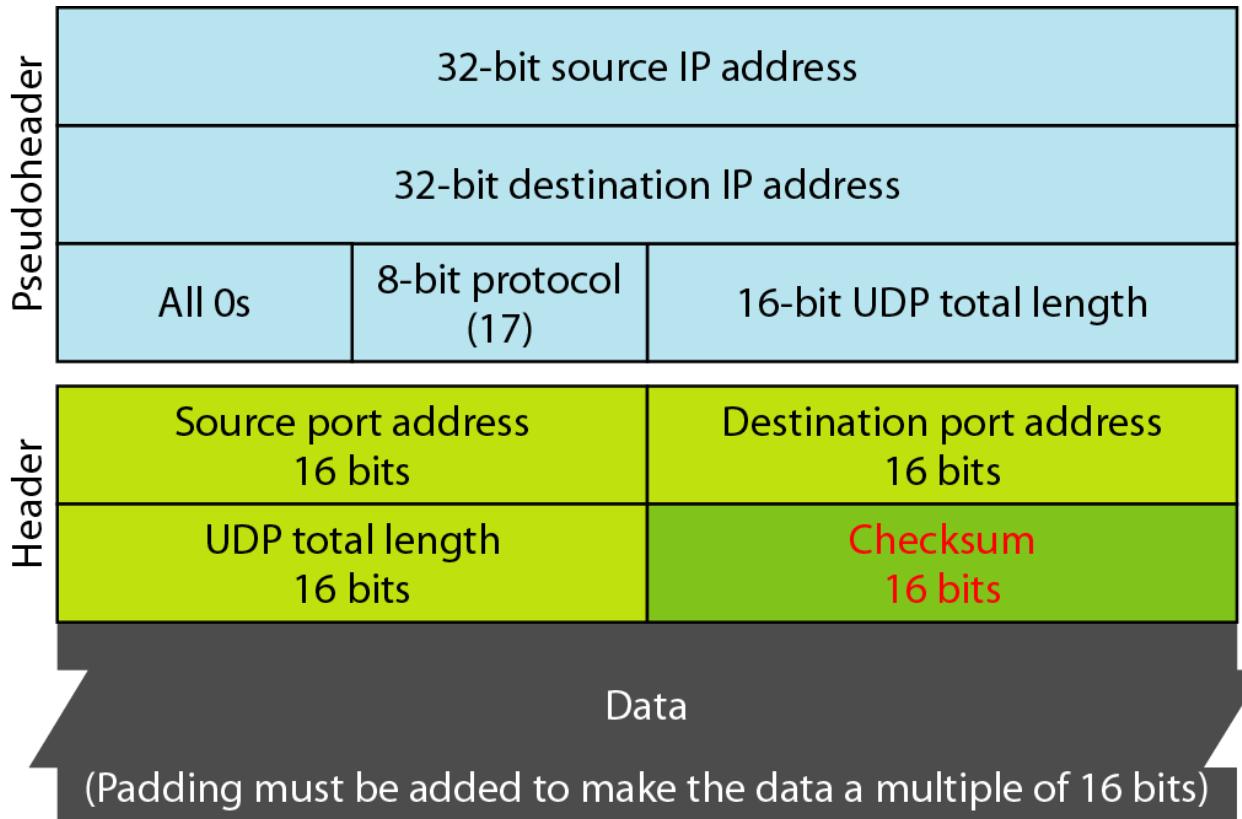


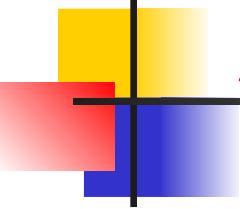


Note

**UDP length
= IP length – IP header's length**

Figure 23.10 Pseudoheader for checksum calculation





Example 23.2

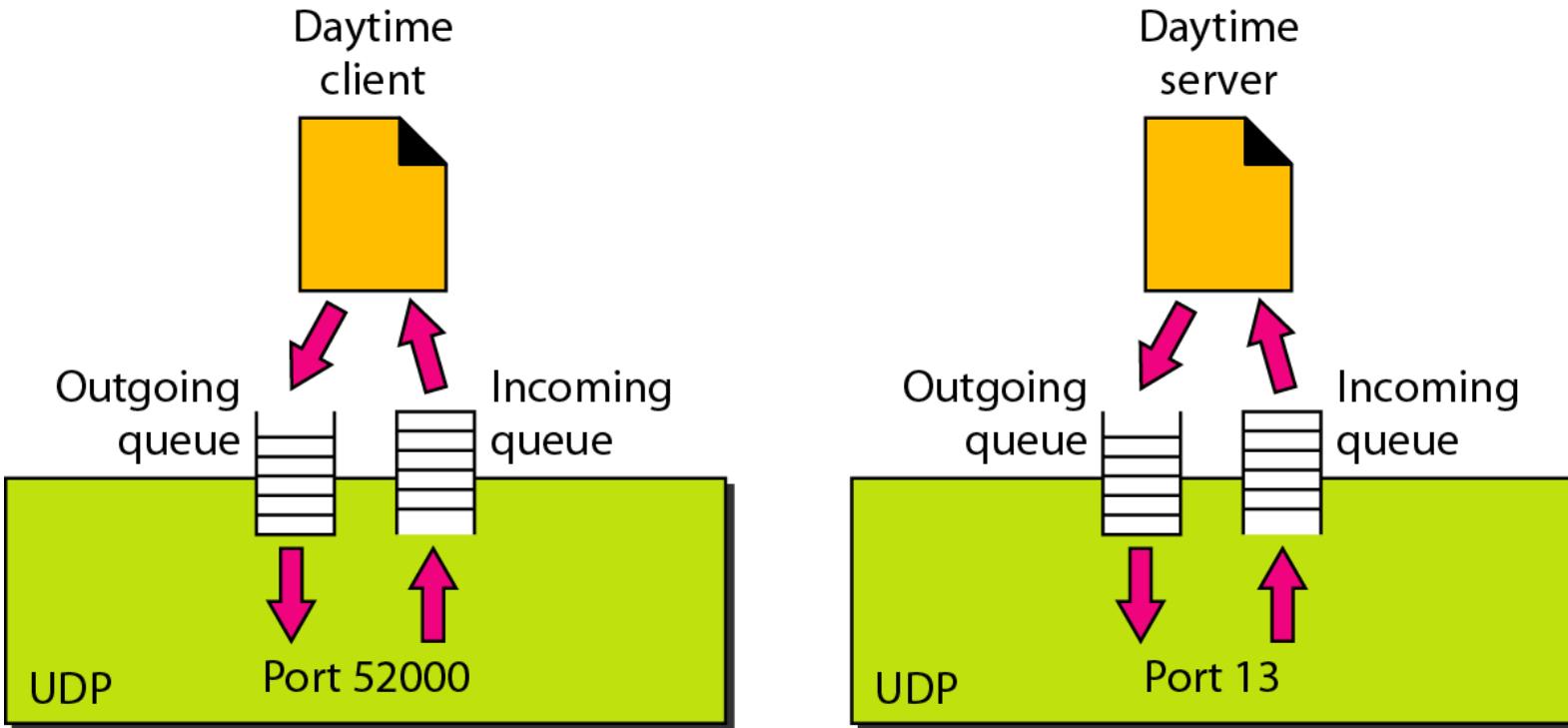
Figure 23.11 shows the checksum calculation for a very small user datagram with only 7 bytes of data. Because the number of bytes of data is odd, padding is added for checksum calculation. The pseudoheader as well as the padding will be dropped when the user datagram is delivered to IP.

Figure 23.11 Checksum calculation of a simple UDP user datagram

153.18.8.105		
171.2.14.10		
All 0s	17	15
1087		13
15		All 0s
T	E	S
I	N	G
All 0s		

10011001 00010010	→	153.18
00001000 01101001	→	8.105
10101011 00000010	→	171.2
00001110 00001010	→	14.10
00000000 00010001	→	0 and 17
00000000 00001111	→	15
00000100 00111111	→	1087
00000000 00001101	→	13
00000000 00001111	→	15
00000000 00000000	→	0 (checksum)
01010100 01000101	→	T and E
01010011 01010100	→	S and T
01001001 01001110	→	I and N
01000111 00000000	→	G and 0 (padding)
<hr/>		
10010110 11101011	→	Sum
01101001 00010100	→	Checksum

Figure 23.12 Queues in UDP



23-3 TCP

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPS to send data. In addition, TCP uses flow and error control mechanisms at the transport level.

Topics discussed in this section:

TCP Services

TCP Features

Segment

A TCP Connection

Flow Control

Error Control

Table 23.2 Well-known ports used by TCP

<i>Port</i>	<i>Protocol</i>	<i>Description</i>
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
20	FTP, Data	File Transfer Protocol (data connection)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	Bootstrap Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

Figure 23.13 Stream delivery

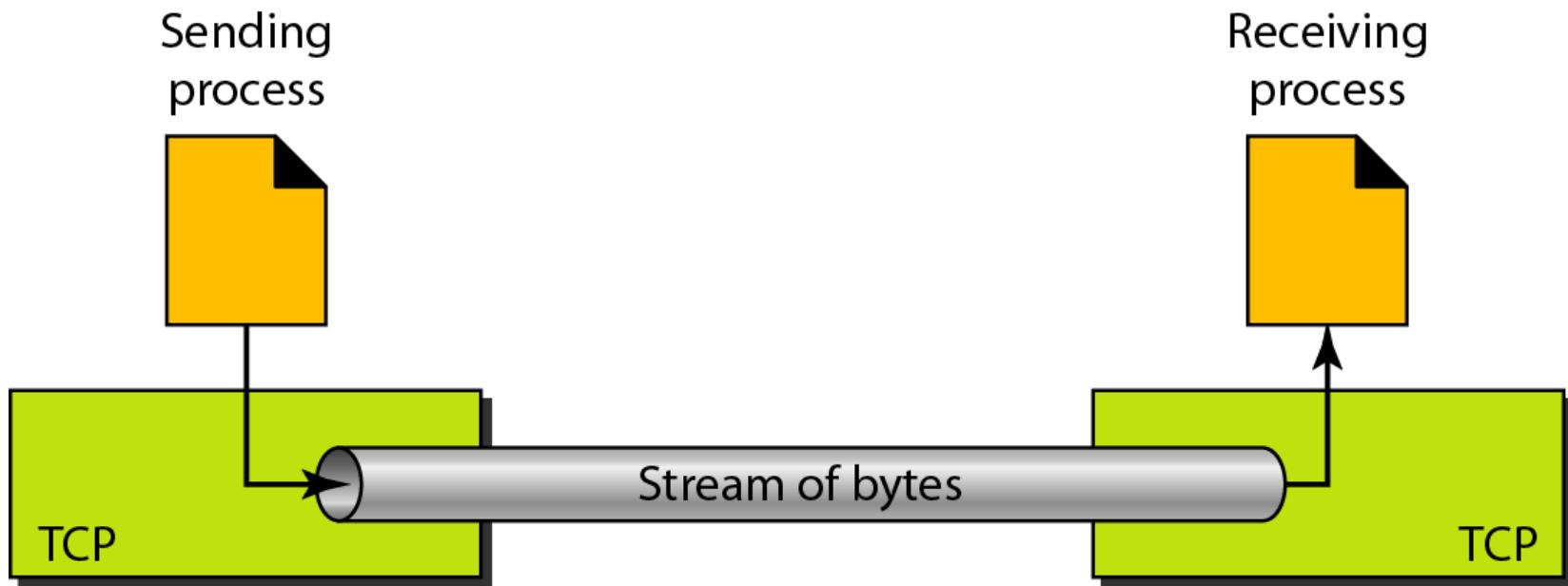


Figure 23.14 *Sending and receiving buffers*

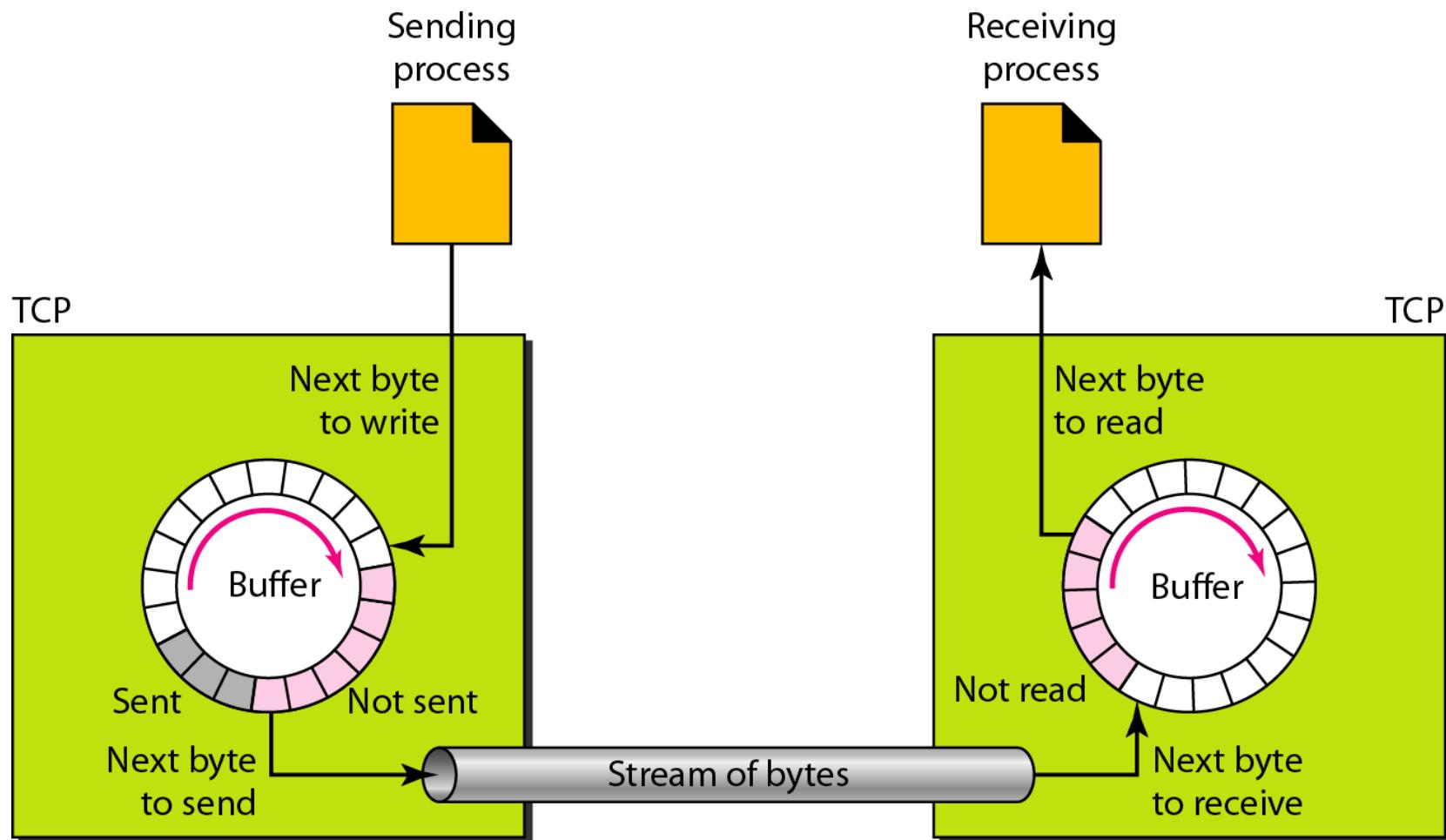
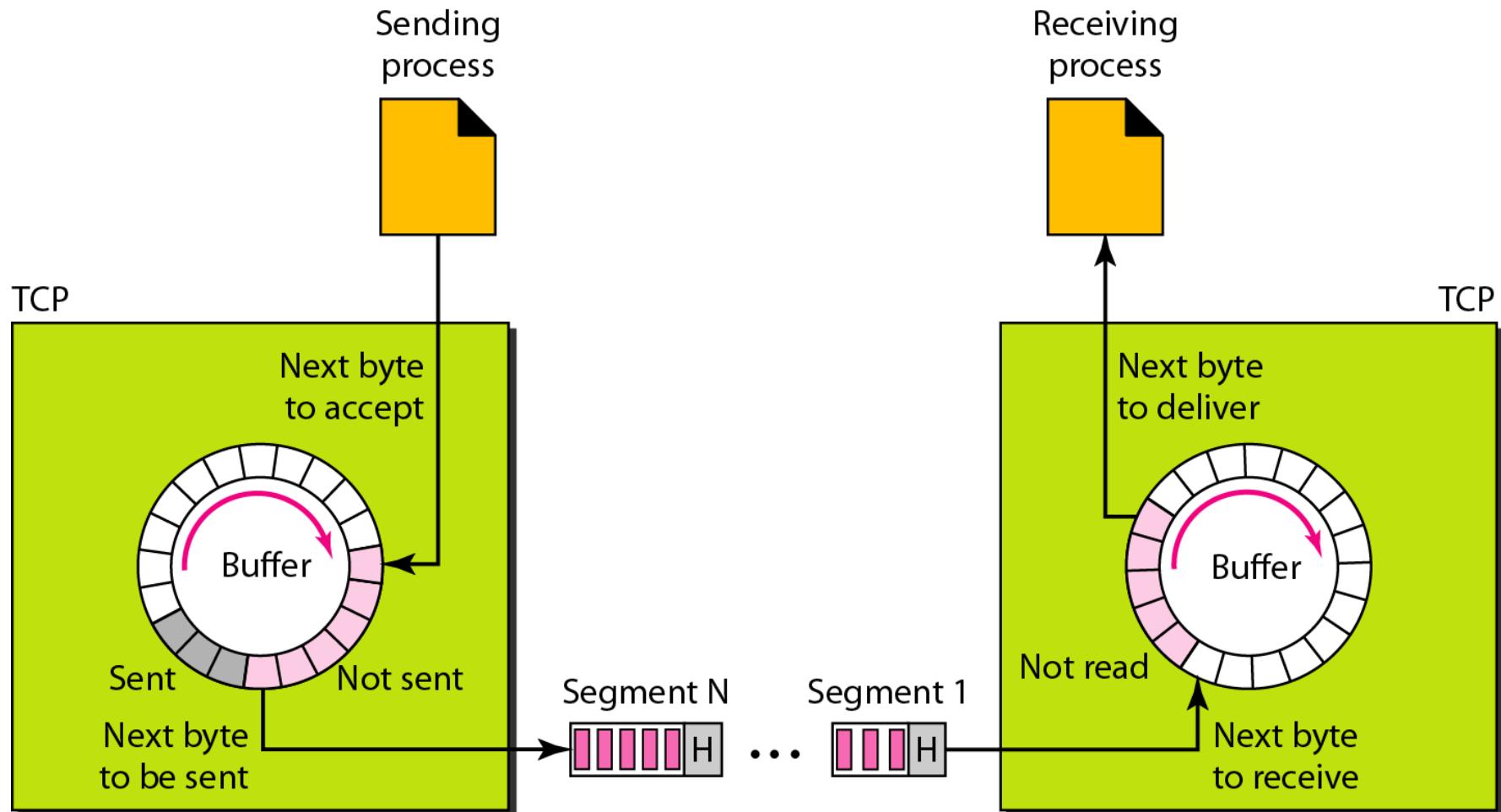
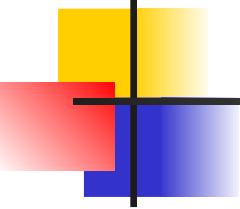


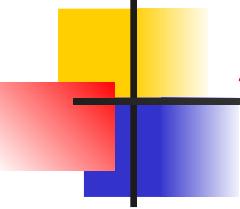
Figure 23.15 TCP segments





Note

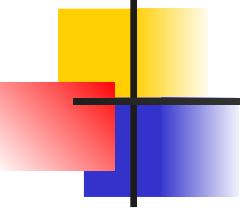
The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.



Example 23.3

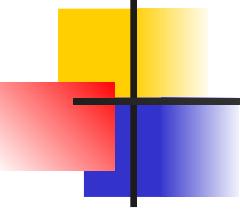
The following shows the sequence number for each segment:

Segment 1	➡	Sequence Number: 10,001 (range: 10,001 to 11,000)
Segment 2	➡	Sequence Number: 11,001 (range: 11,001 to 12,000)
Segment 3	➡	Sequence Number: 12,001 (range: 12,001 to 13,000)
Segment 4	➡	Sequence Number: 13,001 (range: 13,001 to 14,000)
Segment 5	➡	Sequence Number: 14,001 (range: 14,001 to 15,000)



Note

The value in the sequence number field of a segment defines the number of the first data byte contained in that segment.



Note

The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive.

The acknowledgment number is cumulative.

Figure 23.16 TCP segment format

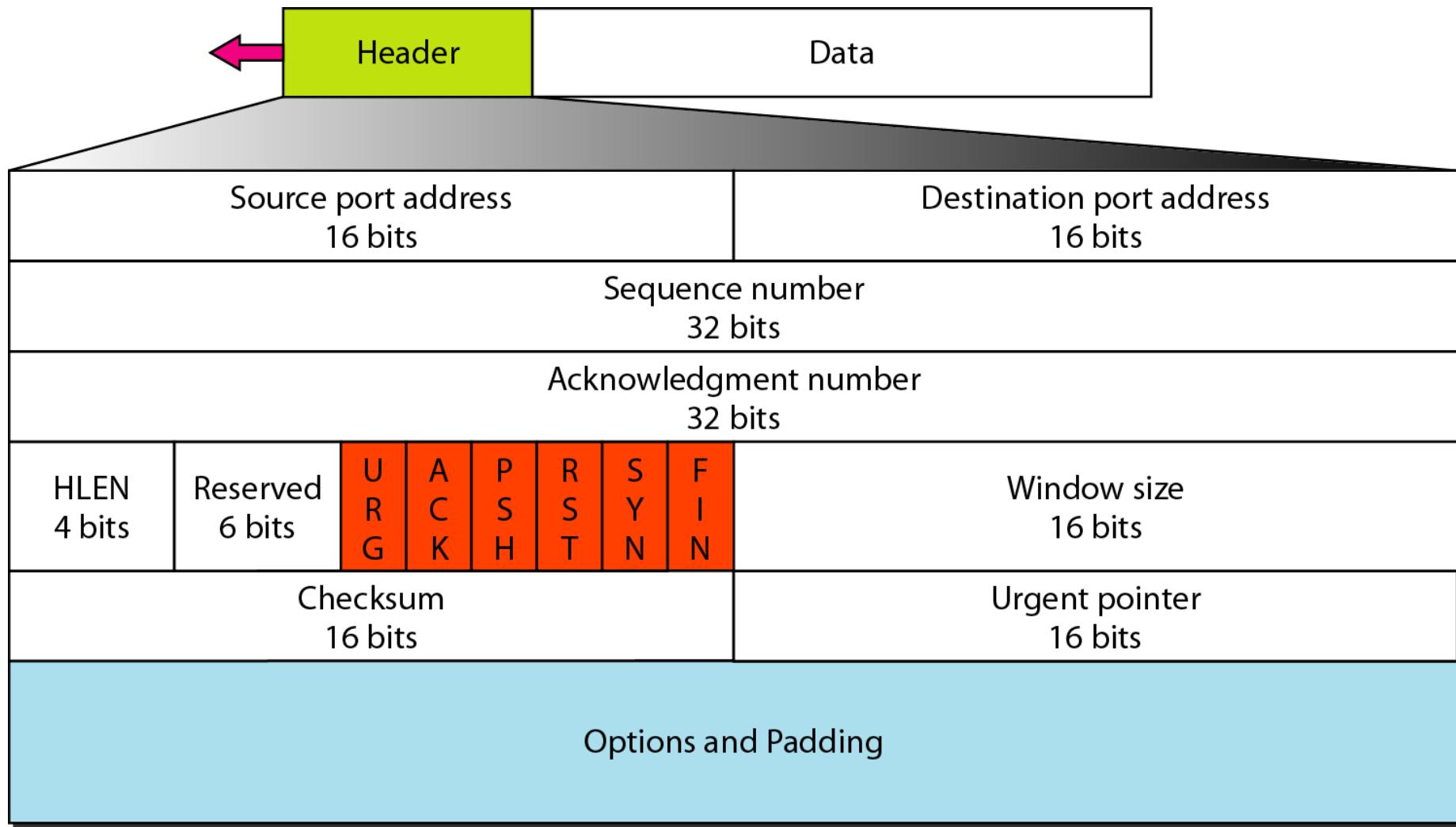


Figure 23.17 Control field

URG: Urgent pointer is valid

ACK: Acknowledgment is valid

PSH: Request for push

RST: Reset the connection

SYN: Synchronize sequence numbers

FIN: Terminate the connection

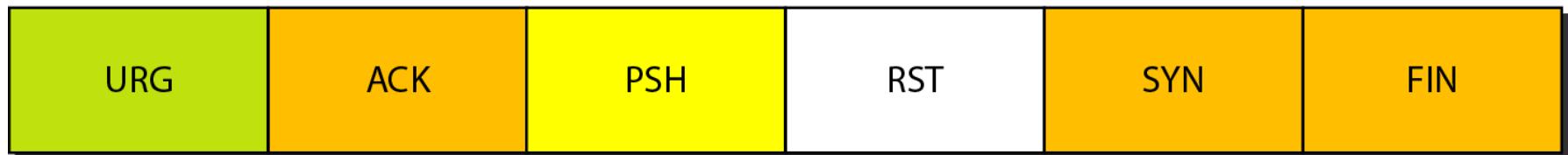
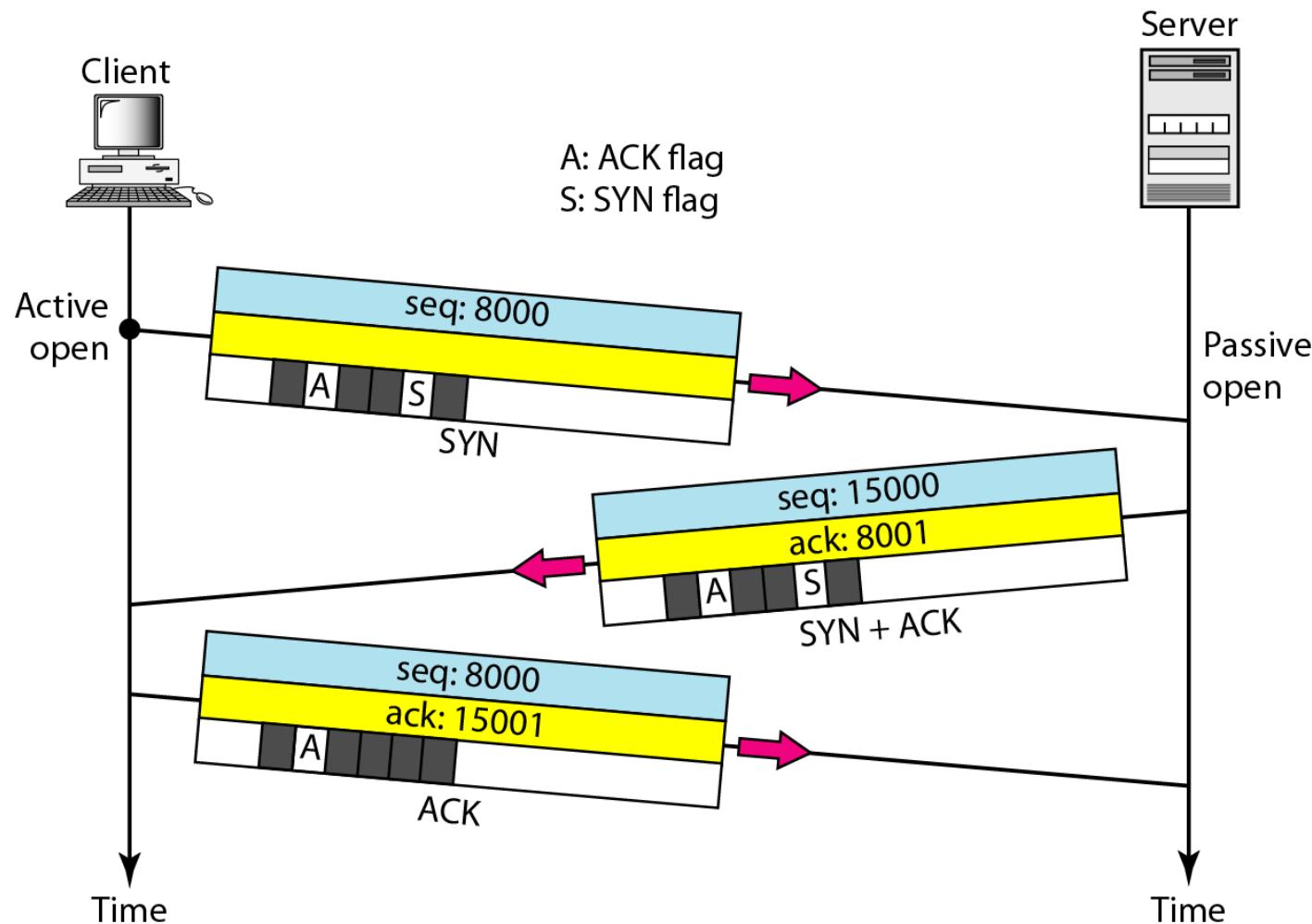
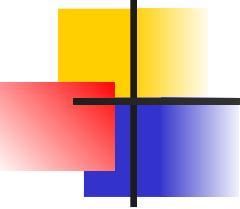


Table 23.3 *Description of flags in the control field*

<i>Flag</i>	<i>Description</i>
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledgment field is valid.
PSH	Push the data.
RST	Reset the connection.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

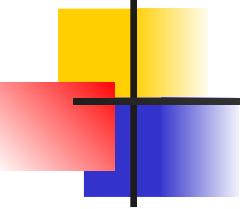
Figure 23.18 Connection establishment using three-way handshaking





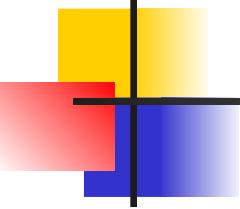
Note

A SYN segment cannot carry data, but it consumes one sequence number.



Note

A SYN + ACK segment cannot carry data, but does consume one sequence number.



Note

**An ACK segment, if carrying no data,
consumes no sequence number.**

Figure 23.19 Data transfer

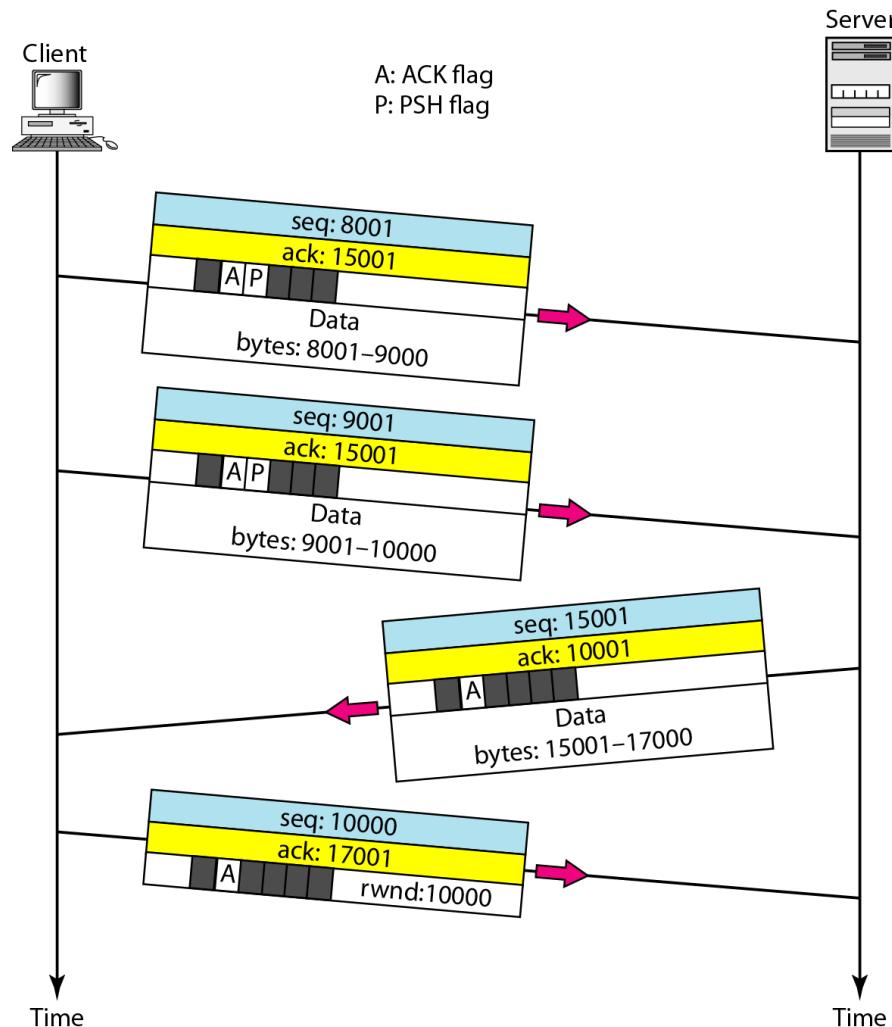
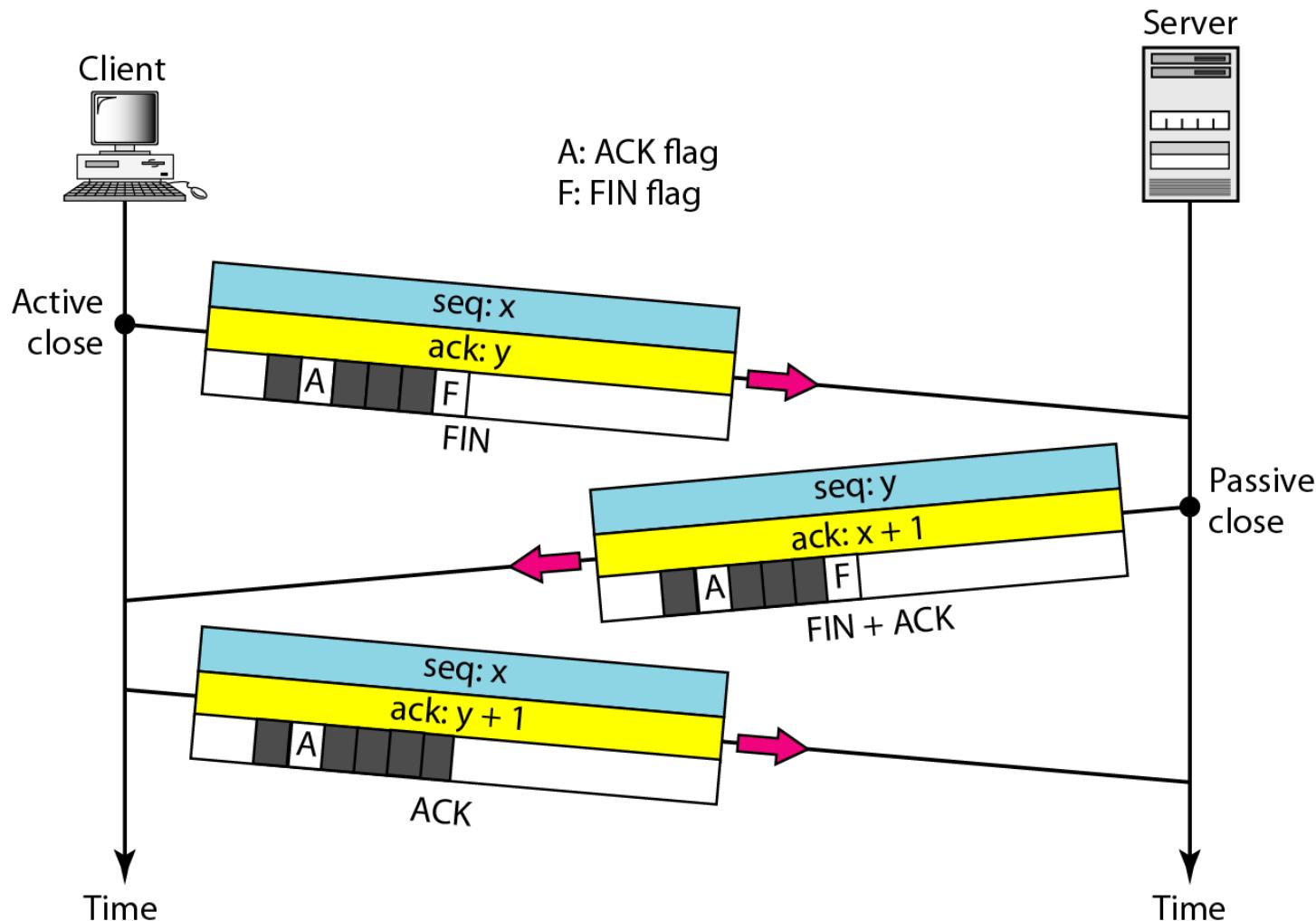
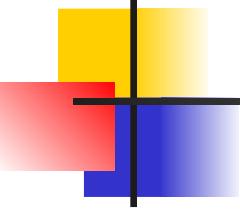


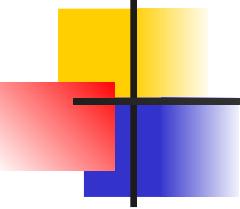
Figure 23.20 Connection termination using three-way handshaking





Note

The FIN segment consumes one sequence number if it does not carry data.



Note

**The FIN + ACK segment consumes
one sequence number if it
does not carry data.**

Figure 23.21 Half-close

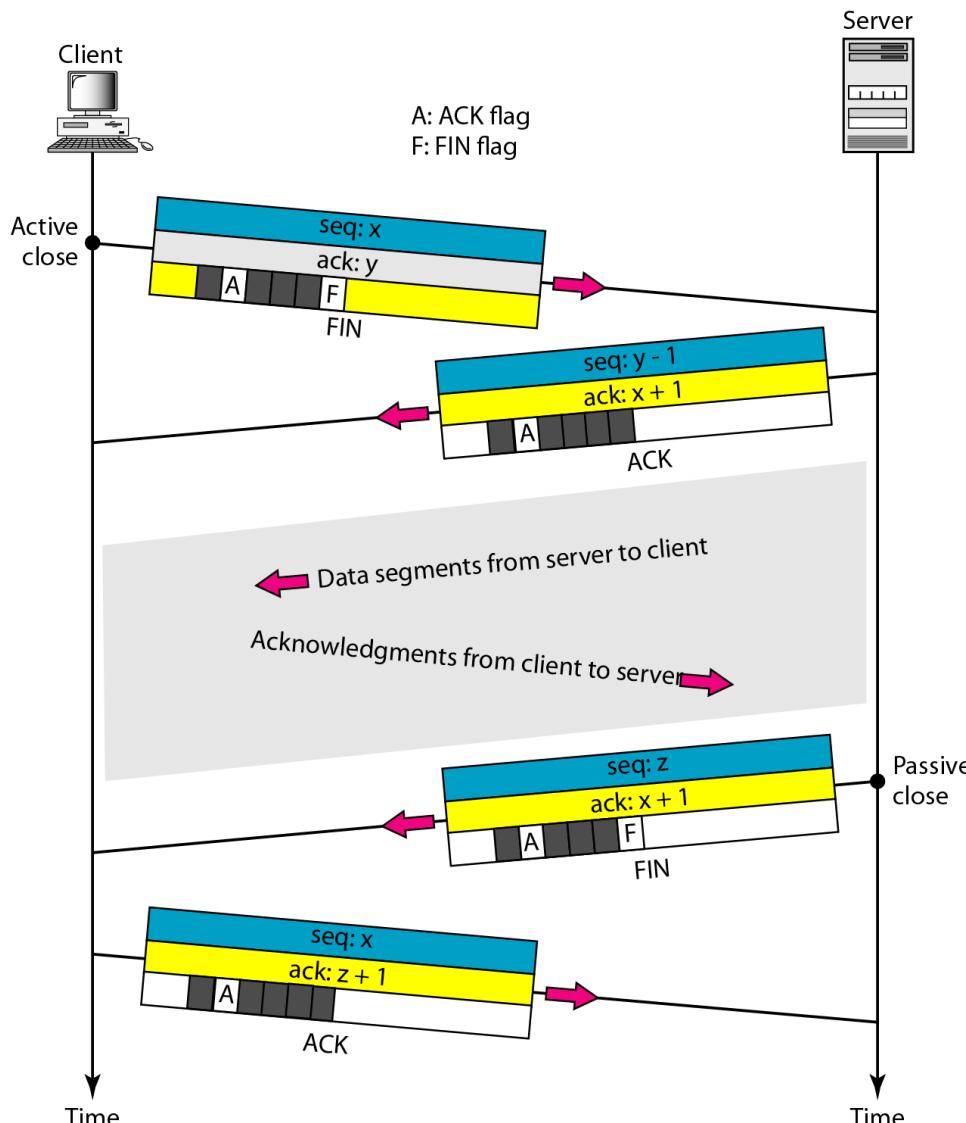
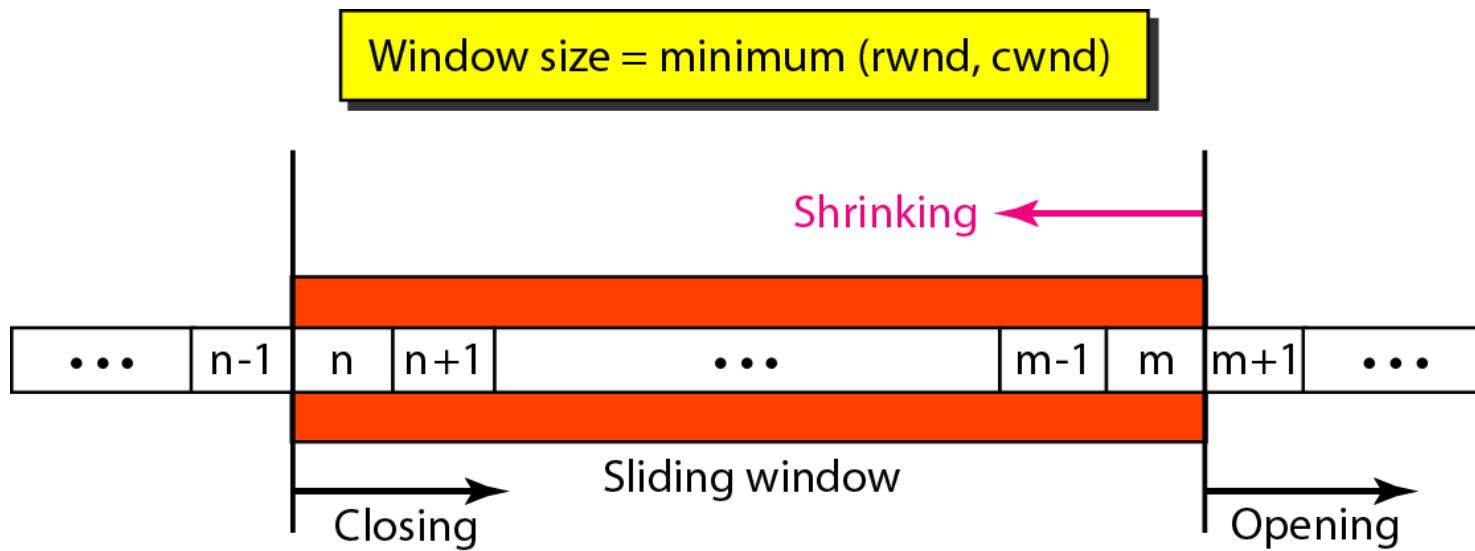
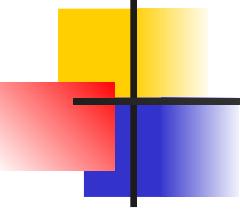


Figure 23.22 Sliding window

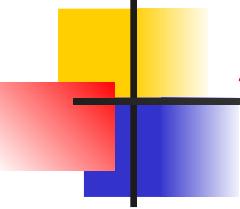




Note

A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data.

TCP sliding windows are byte-oriented.

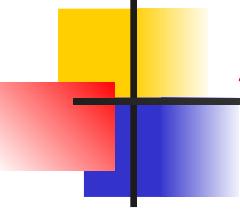


Example 23.4

What is the value of the receiver window (rwnd) for host A if the receiver, host B, has a buffer size of 5000 bytes and 1000 bytes of received and unprocessed data?

Solution

The value of rwnd = 5000 – 1000 = 4000. Host B can receive only 4000 bytes of data before overflowing its buffer. Host B advertises this value in its next segment to A.

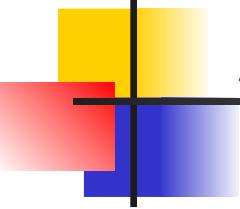


Example 23.5

What is the size of the window for host A if the value of rwnd is 3000 bytes and the value of cwnd is 3500 bytes?

Solution

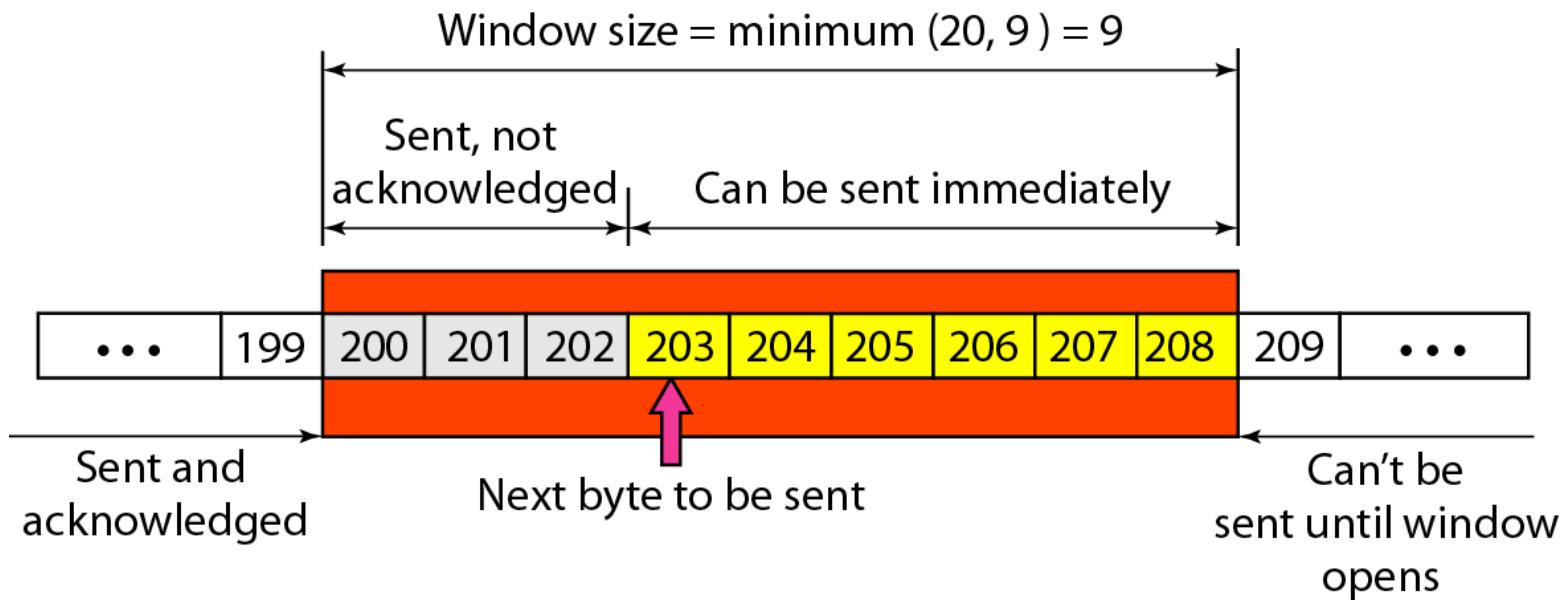
The size of the window is the smaller of rwnd and cwnd, which is 3000 bytes.



Example 23.6

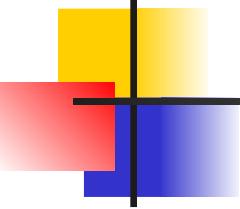
Figure 23.23 shows an unrealistic example of a sliding window. The sender has sent bytes up to 202. We assume that cwnd is 20 (in reality this value is thousands of bytes). The receiver has sent an acknowledgment number of 200 with an rwnd of 9 bytes (in reality this value is thousands of bytes). The size of the sender window is the minimum of rwnd and cwnd, or 9 bytes. Bytes 200 to 202 are sent, but not acknowledged. Bytes 203 to 208 can be sent without worrying about acknowledgment. Bytes 209 and above cannot be sent.

Figure 23.23 Example 23.6



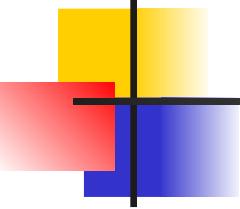
Some points about TCP sliding windows:

- The size of the window is the lesser of rwnd and cwnd.**
- The source does not have to send a full window's worth of data.**
- The window can be opened or closed by the receiver, but should not be shrunk.**
- The destination can send an acknowledgment at any time as long as it does not result in a shrinking window.**
- The receiver can temporarily shut down the window; the sender, however, can always send a segment of 1 byte after the window is shut down.**



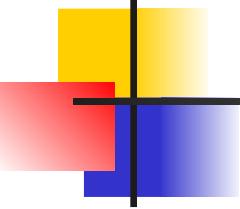
Note

ACK segments do not consume sequence numbers and are not acknowledged.



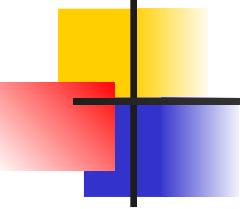
Note

In modern implementations, a retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived.



Note

**No retransmission timer is set for an
ACK segment.**



Note

Data may arrive out of order and be temporarily stored by the receiving TCP, but TCP guarantees that no out-of-order segment is delivered to the process.

Figure 23.24 Normal operation

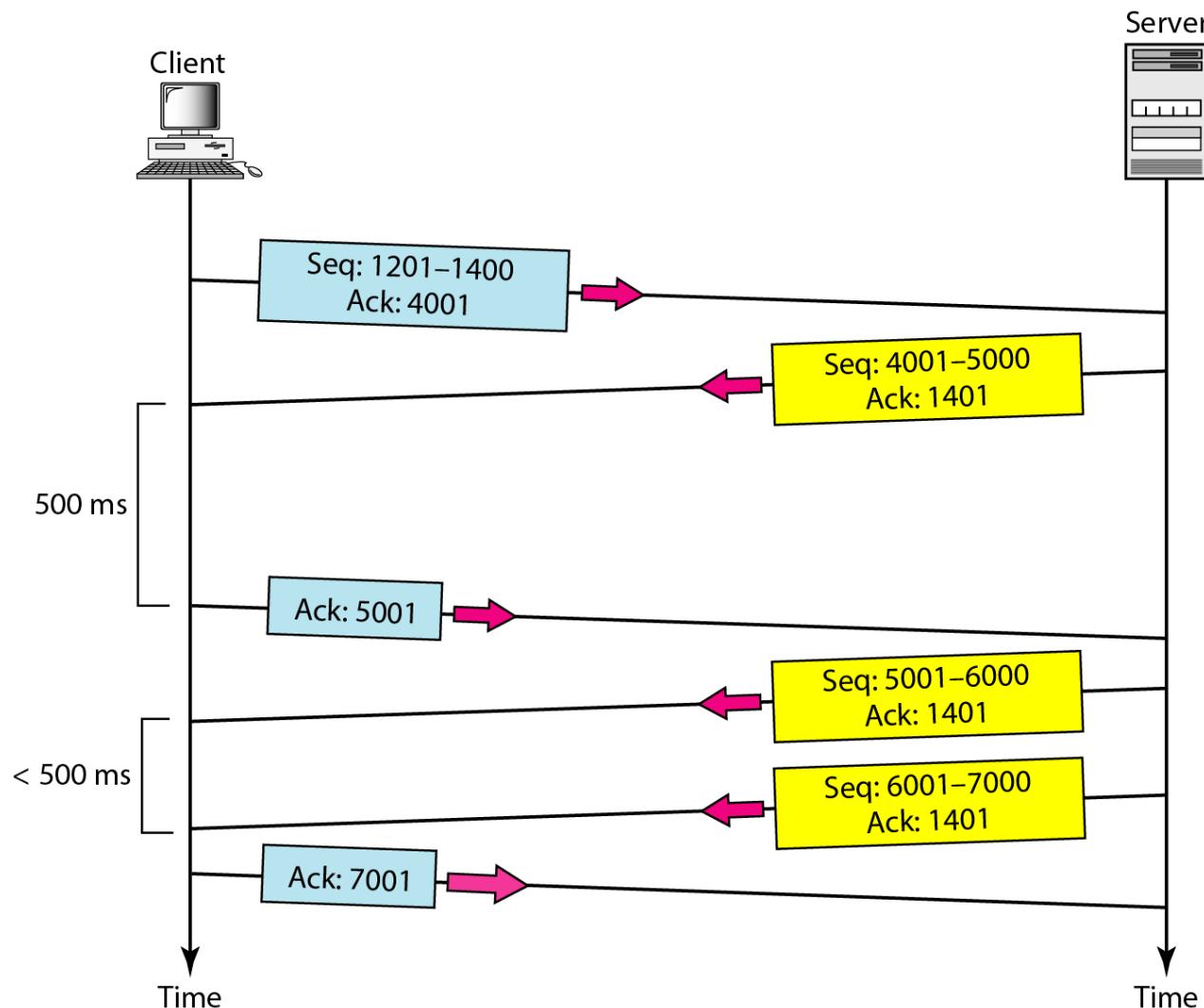
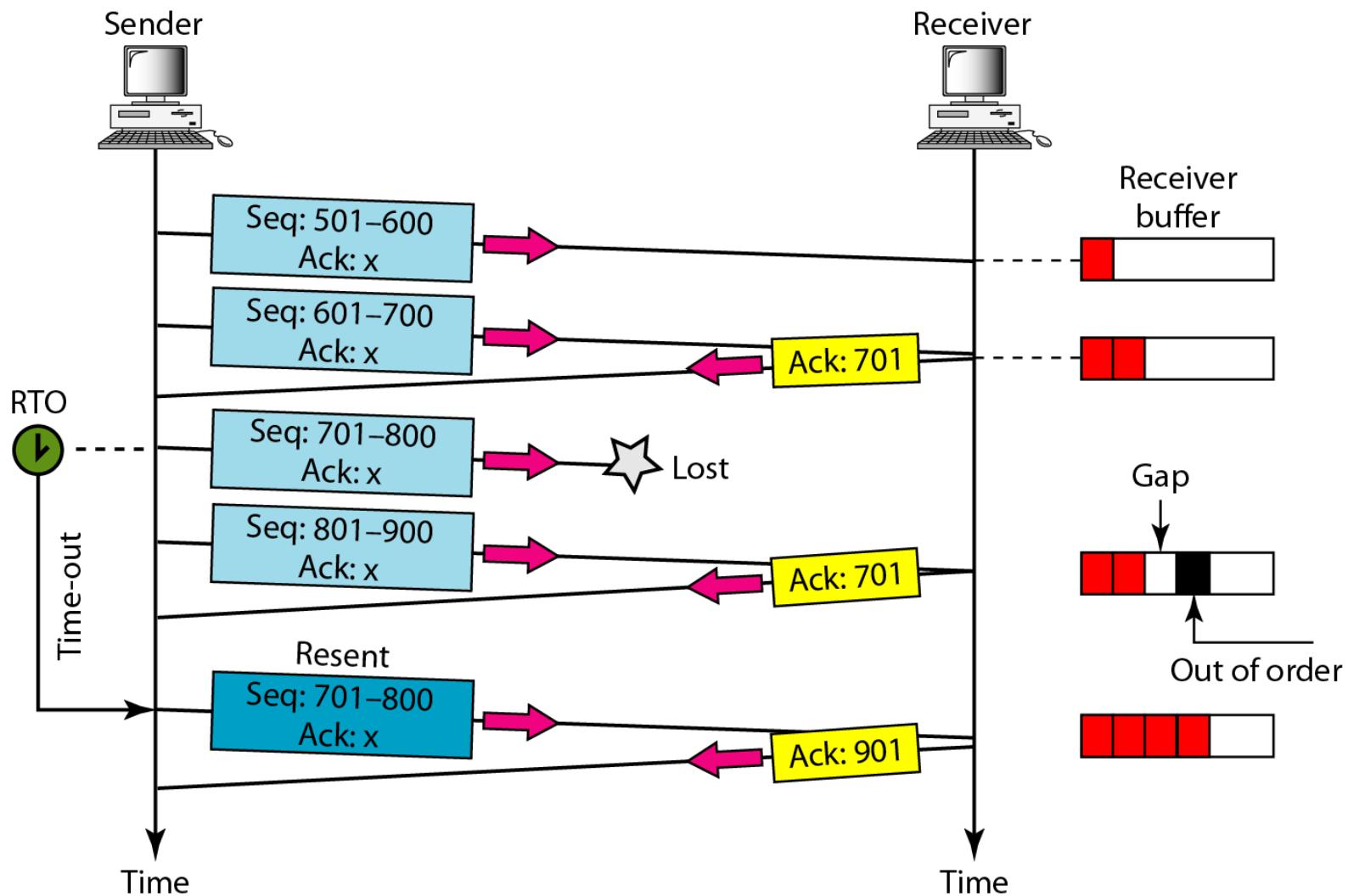
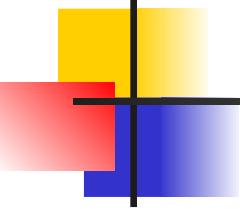


Figure 23.25 Lost segment

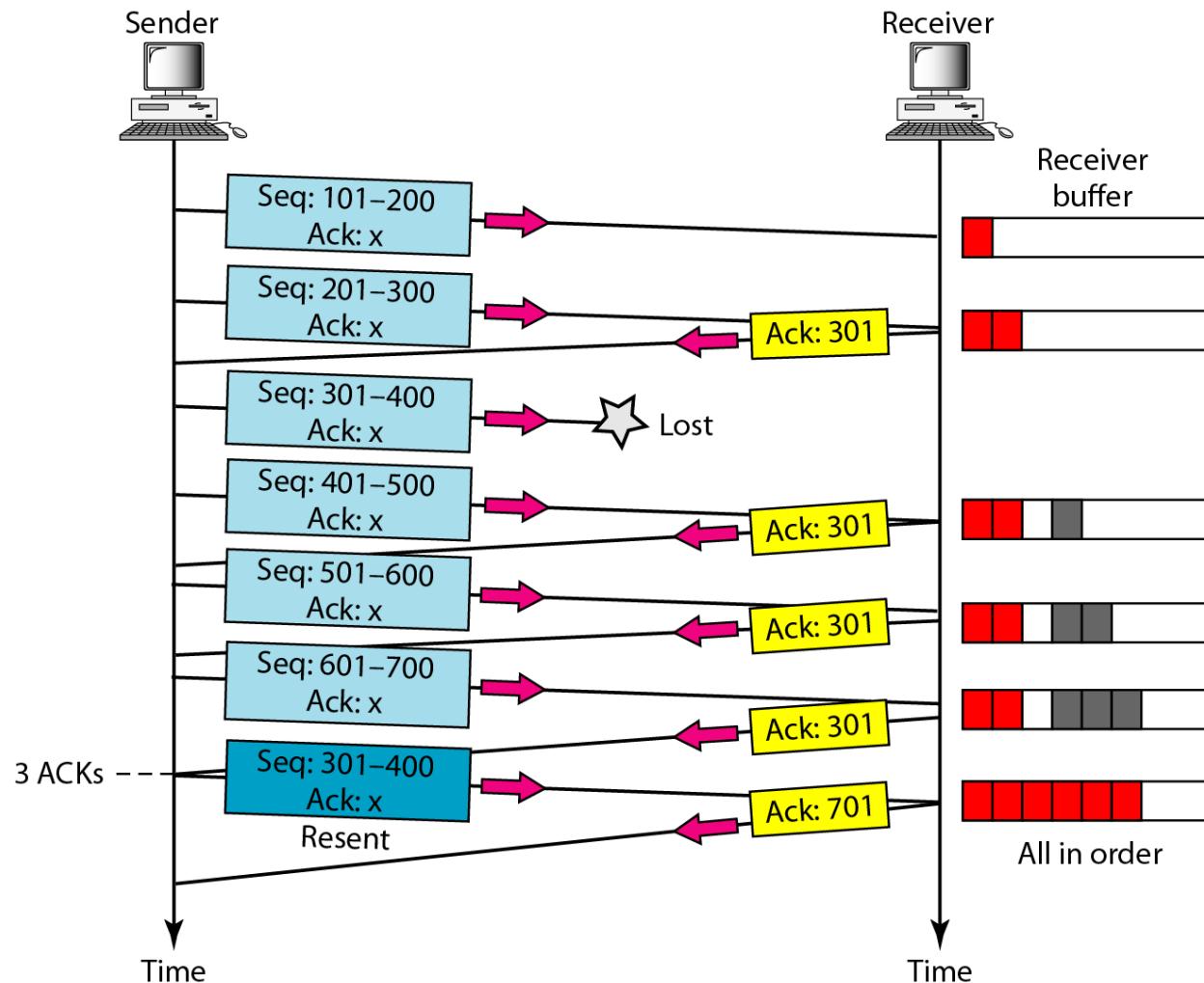




Note

The receiver TCP delivers only ordered data to the process.

Figure 23.26 Fast retransmission



23-4 SCTP

Stream Control Transmission Protocol (SCTP) is a new reliable, message-oriented transport layer protocol. SCTP, however, is mostly designed for Internet applications that have recently been introduced. These new applications need a more sophisticated service than TCP can provide.

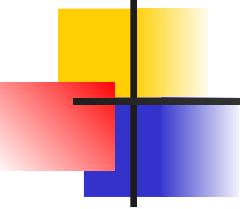
Topics discussed in this section:

SCTP Services and Features

Packet Format

An SCTP Association

Flow Control and Error Control



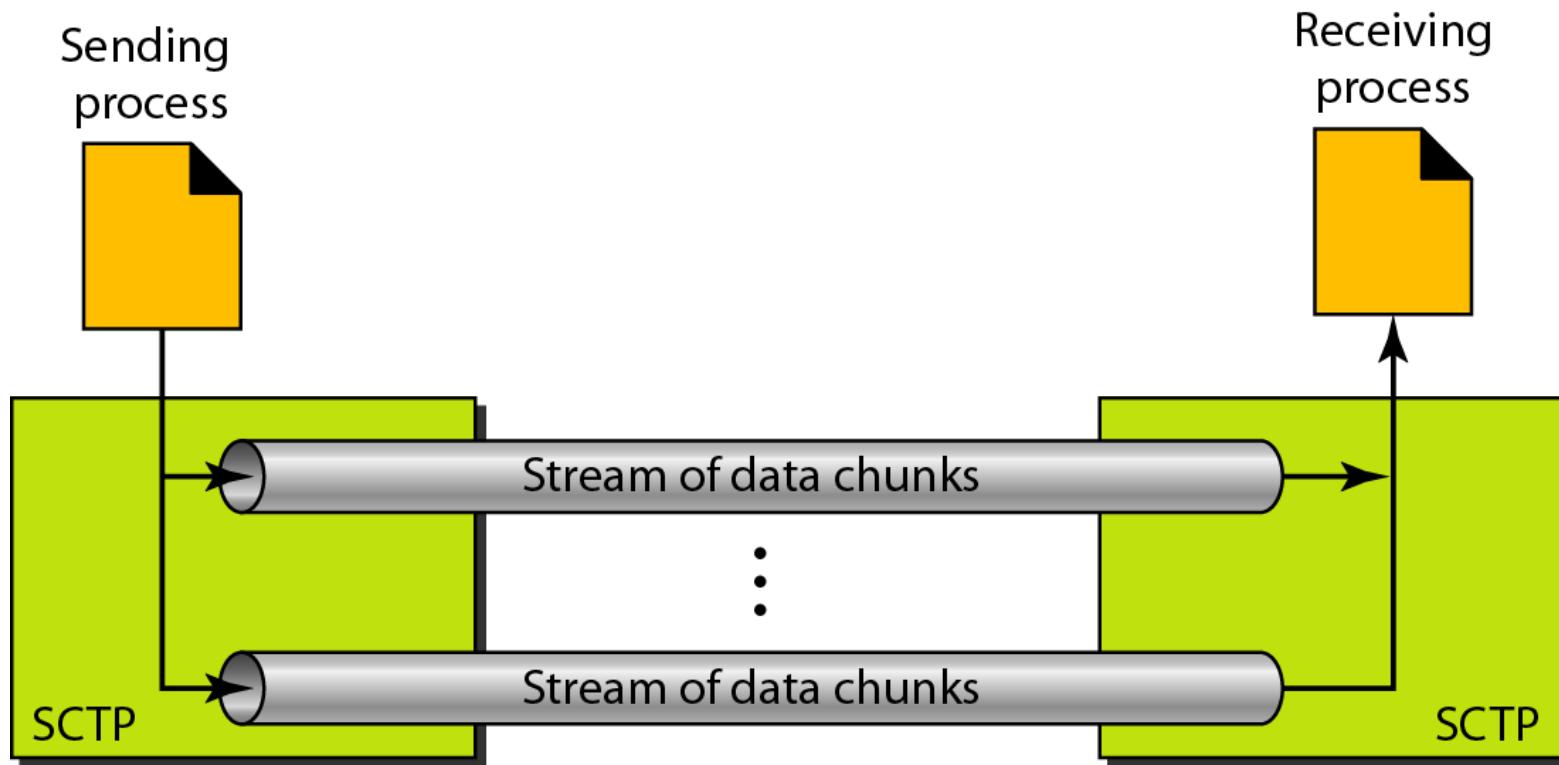
Note

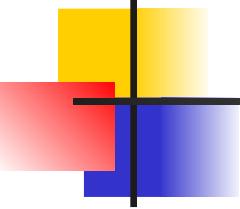
SCTP is a message-oriented, reliable protocol that combines the best features of UDP and TCP.

Table 23.4 Some SCTP applications

<i>Protocol</i>	<i>Port Number</i>	<i>Description</i>
IUA	9990	ISDN over IP
M2UA	2904	SS7 telephony signaling
M3UA	2905	SS7 telephony signaling
H.248	2945	Media gateway control
H.323	1718, 1719, 1720, 11720	IP telephony
SIP	5060	IP telephony

Figure 23.27 *Multiple-stream concept*

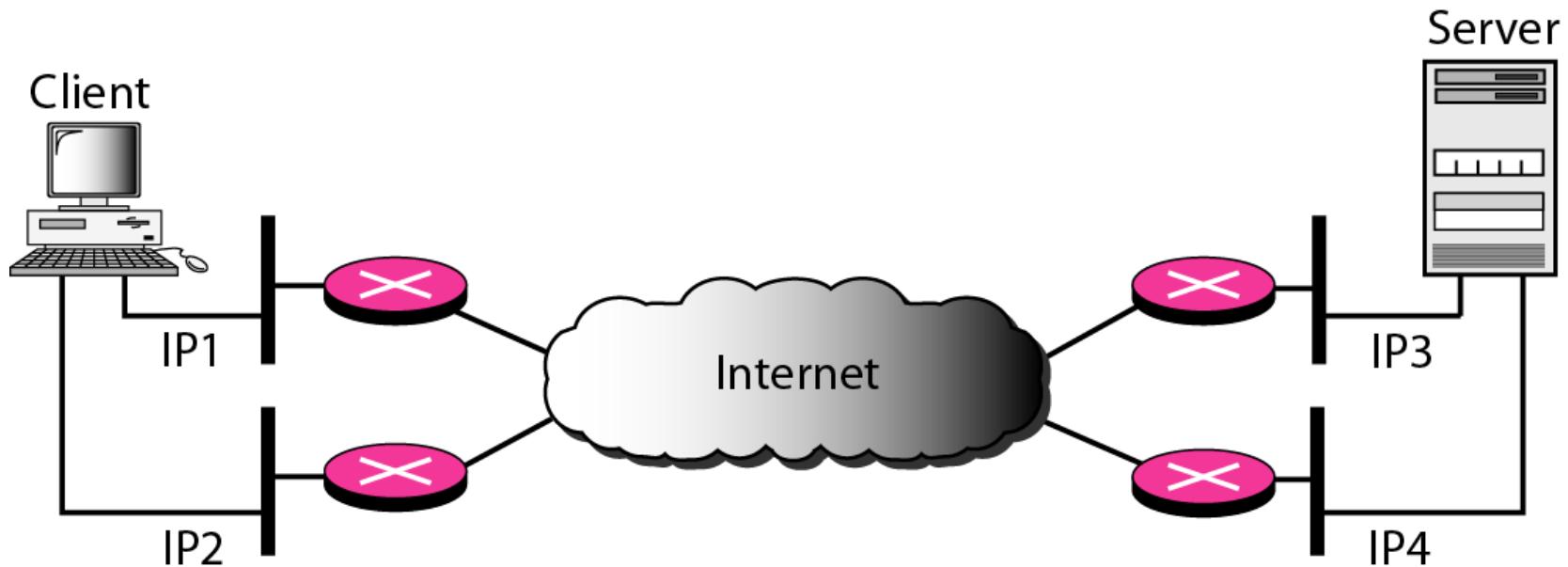


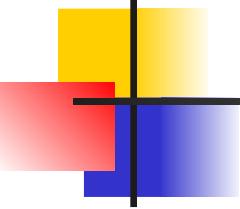


Note

An association in SCTP can involve multiple streams.

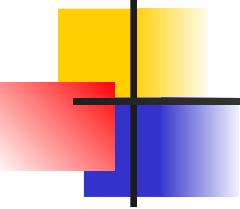
Figure 23.28 *Multihoming concept*





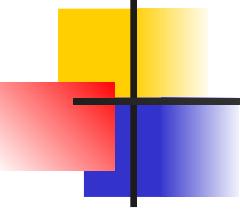
Note

SCTP association allows multiple IP addresses for each end.



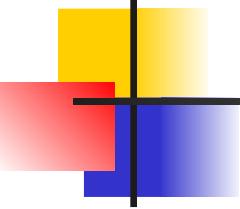
Note

In SCTP, a data chunk is numbered using a TSN.



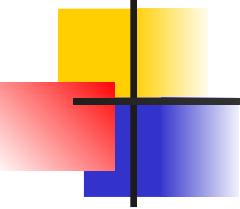
Note

To distinguish between different streams, SCTP uses an SI.



Note

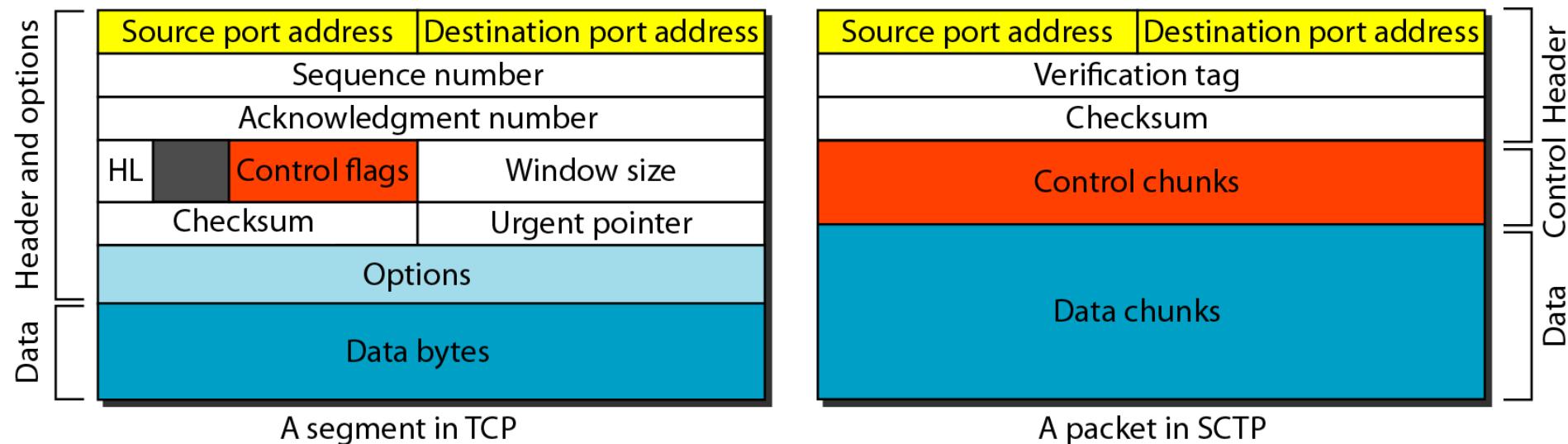
To distinguish between different data chunks belonging to the same stream, SCTP uses SSNs.

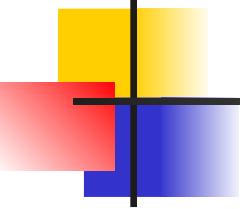


Note

TCP has segments; SCTP has packets.

Figure 23.29 Comparison between a TCP segment and an SCTP packet

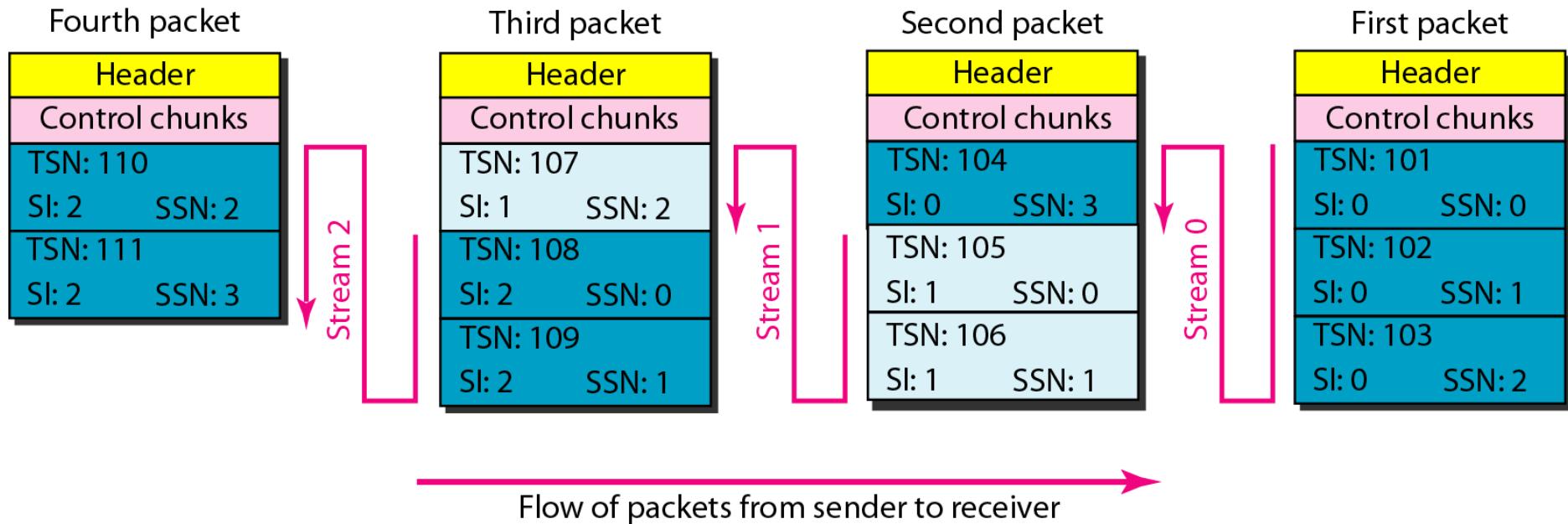


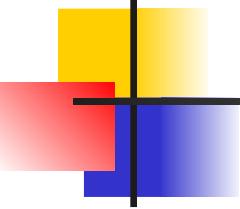


Note

In SCTP, control information and data information are carried in separate chunks.

Figure 23.30 Packet, data chunks, and streams

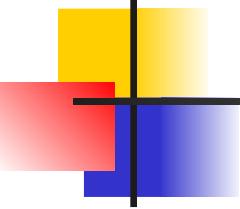




Note

Data chunks are identified by three items: TSN, SI, and SSN.

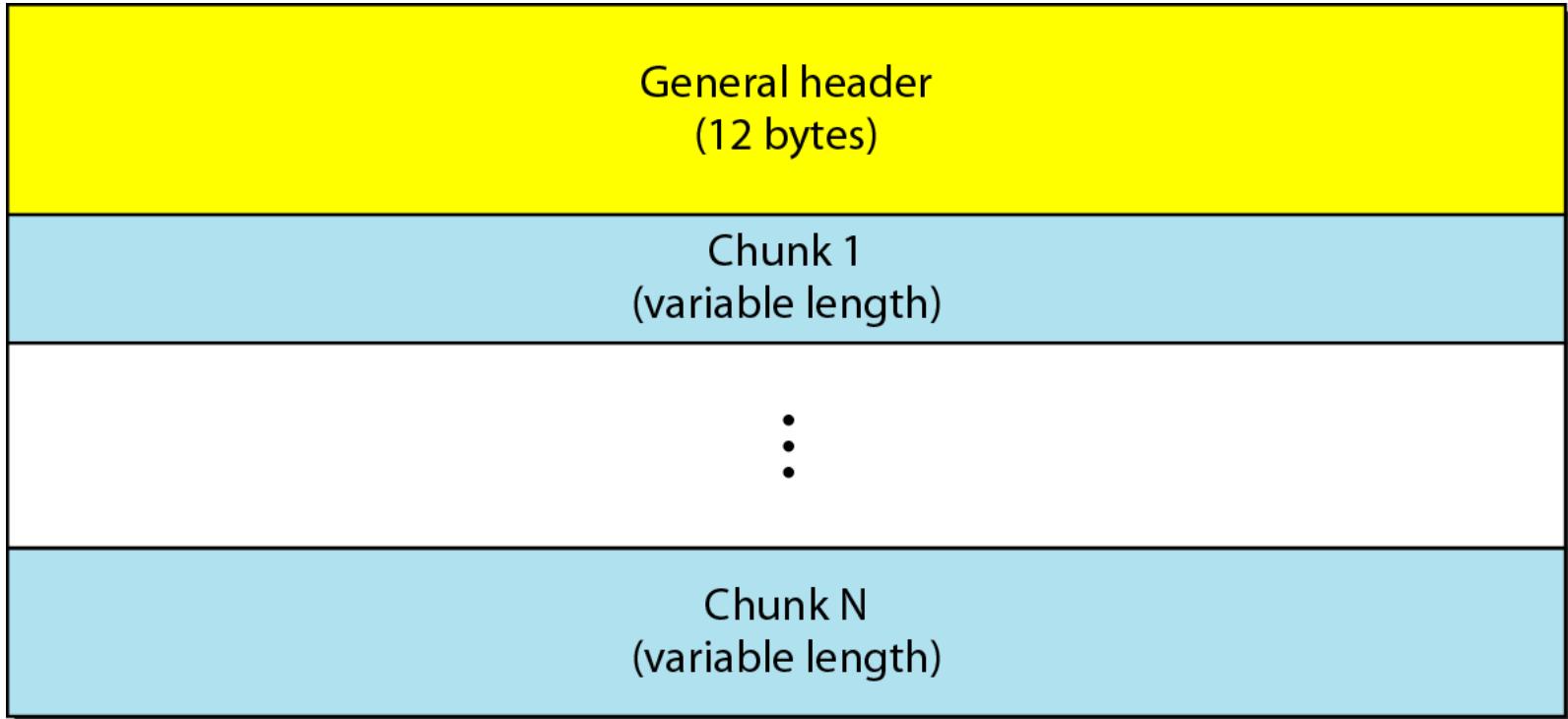
TSN is a cumulative number identifying the association; SI defines the stream; SSN defines the chunk in a stream.

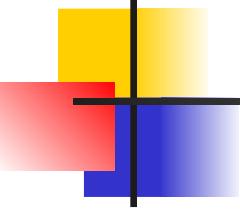


Note

In SCTP, acknowledgment numbers are used to acknowledge only data chunks; control chunks are acknowledged by other control chunks if necessary.

Figure 23.31 *SCTP packet format*





Note

In an SCTP packet, control chunks come before data chunks.

Figure 23.32 General header

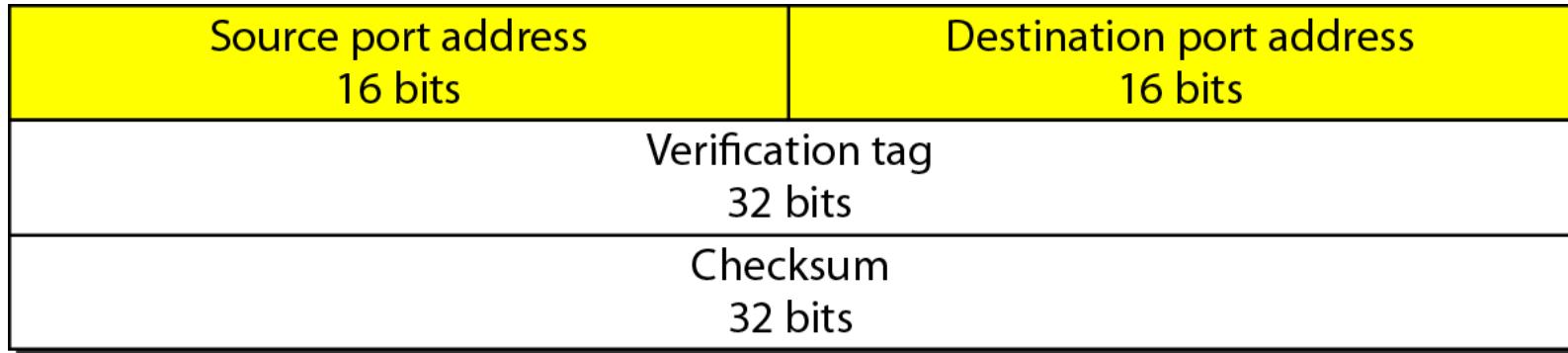
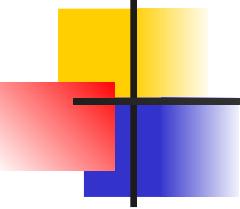


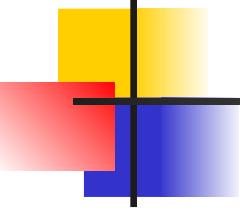
Table 23.5 *Chunks*

Type	Chunk	Description
0	DATA	User data
1	INIT	Sets up an association
2	INIT ACK	Acknowledges INIT chunk
3	SACK	Selective acknowledgment
4	HEARTBEAT	Probes the peer for liveness
5	HEARTBEAT ACK	Acknowledges HEARTBEAT chunk
6	ABORT	Aborts an association
7	SHUTDOWN	Terminates an association
8	SHUTDOWN ACK	Acknowledges SHUTDOWN chunk
9	ERROR	Reports errors without shutting down
10	COOKIE ECHO	Third packet in association establishment
11	COOKIE ACK	Acknowledges COOKIE ECHO chunk
14	SHUTDOWN COMPLETE	Third packet in association termination
192	FORWARD TSN	For adjusting cumulative TSN



Note

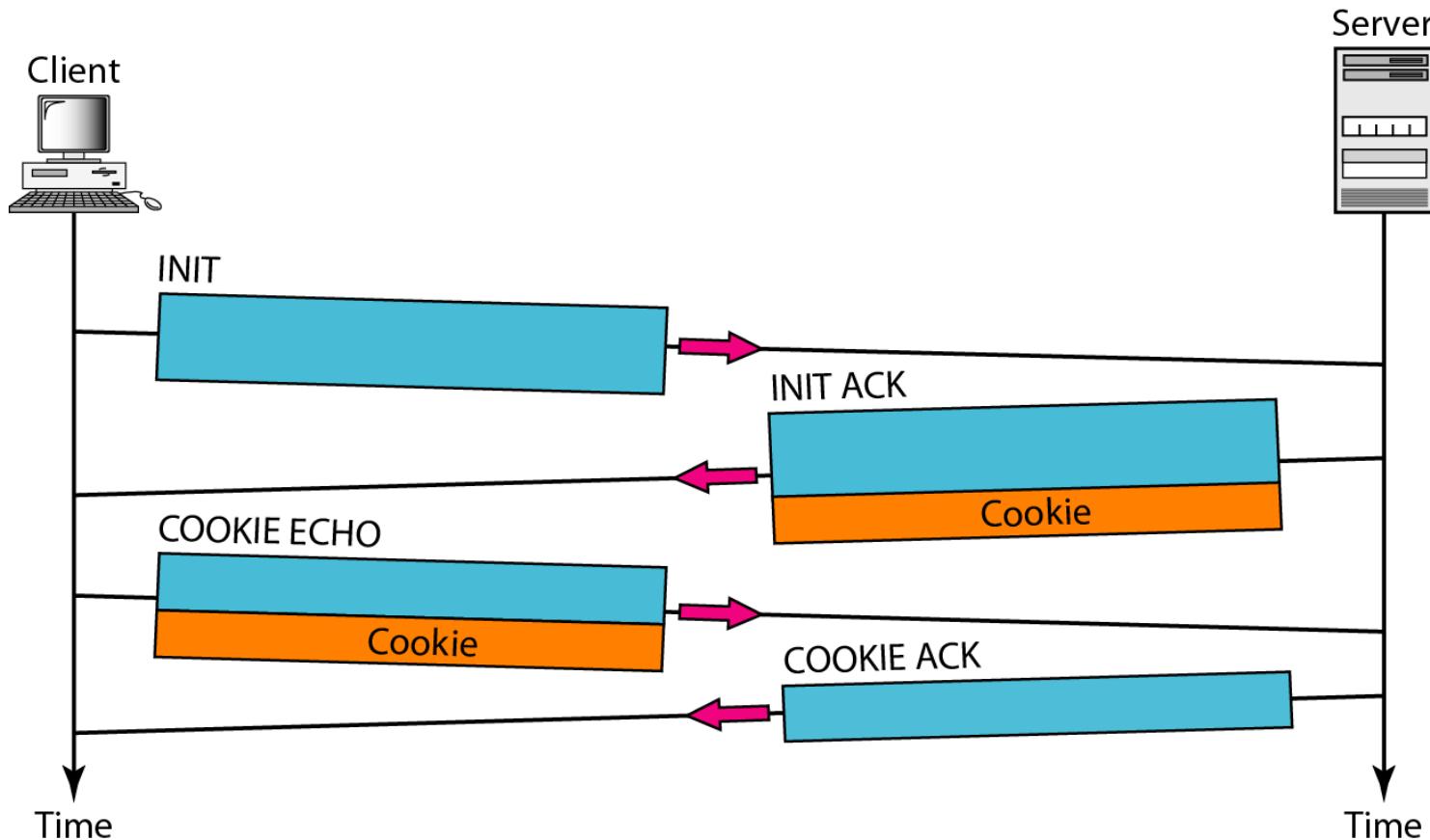
A connection in SCTP is called an association.

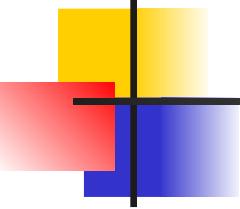


Note

**No other chunk is allowed in a packet carrying an INIT or INIT ACK chunk.
A COOKIE ECHO or a COOKIE ACK chunk can carry data chunks.**

Figure 23.33 Four-way handshaking

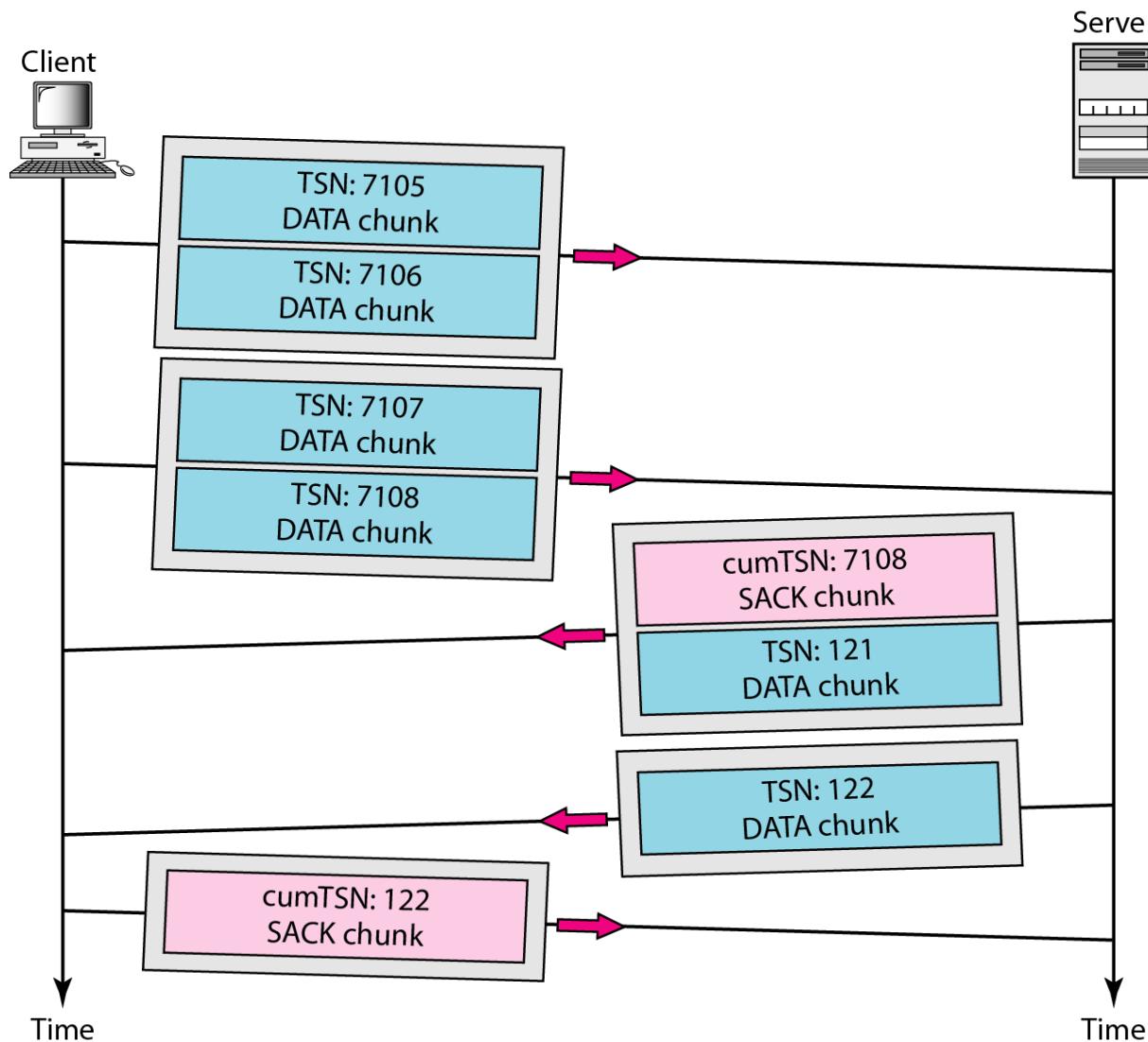


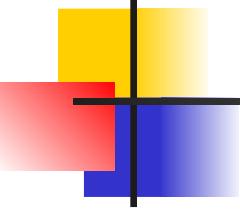


Note

**In SCTP, only DATA chunks
consume TSNs;
DATA chunks are the only chunks
that are acknowledged.**

Figure 23.34 Simple data transfer





Note

The acknowledgment in SCTP defines the cumulative TSN, the TSN of the last data chunk received in order.

Figure 23.35 Association termination

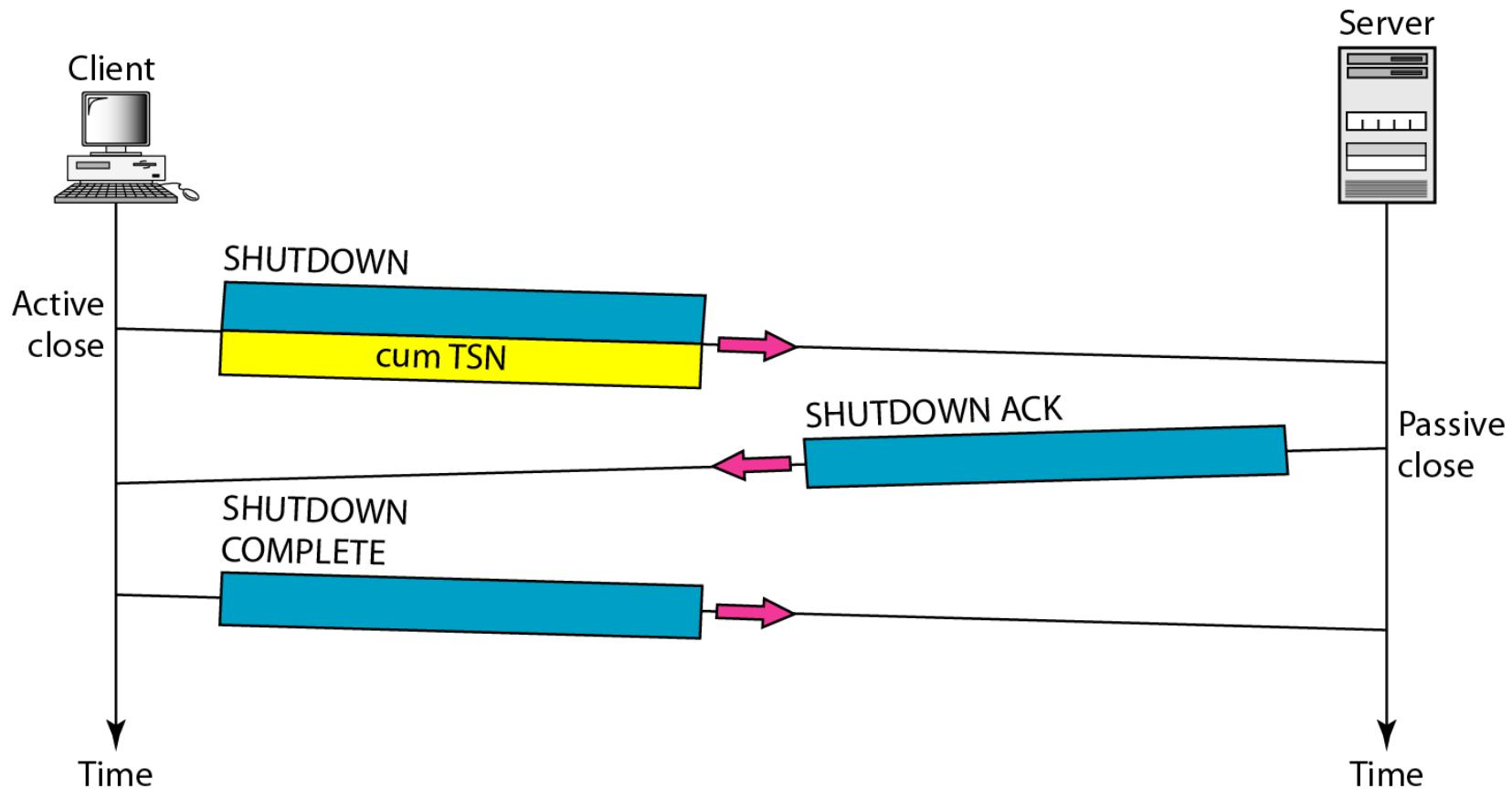


Figure 23.36 Flow control, receiver site

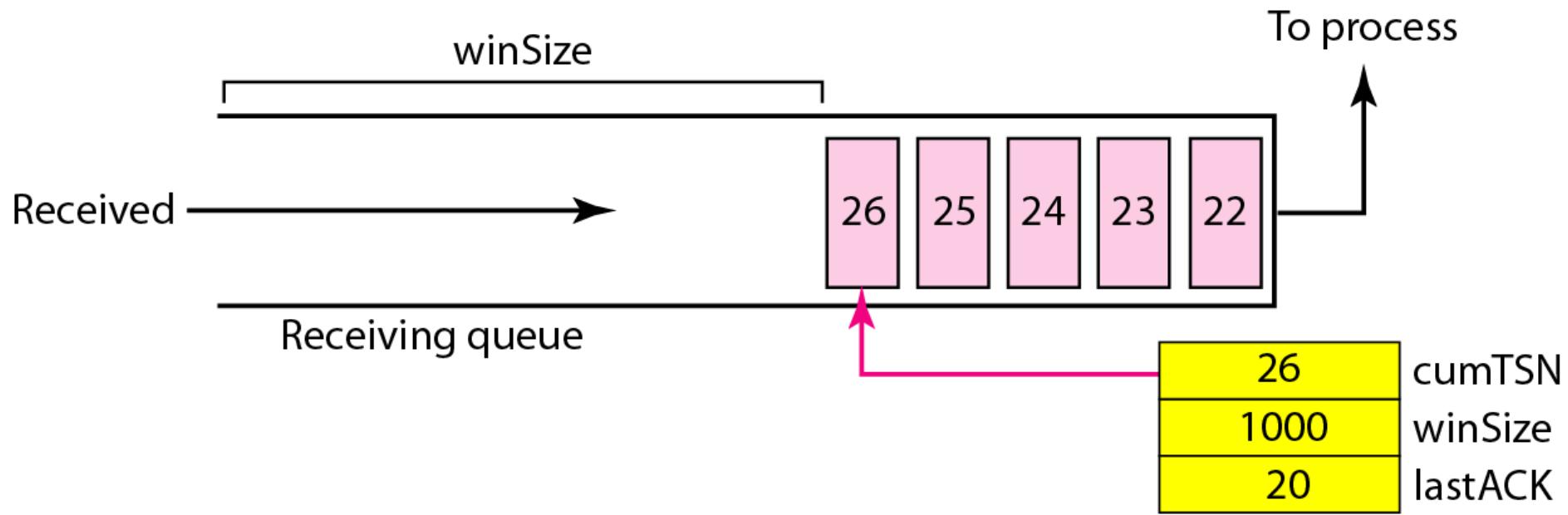


Figure 23.37 Flow control, sender site

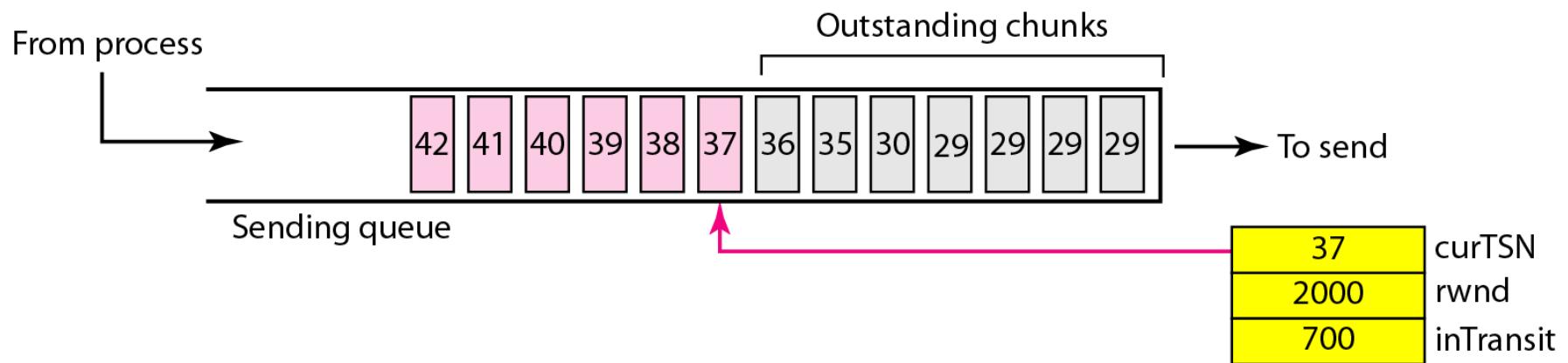


Figure 23.38 Flow control scenario

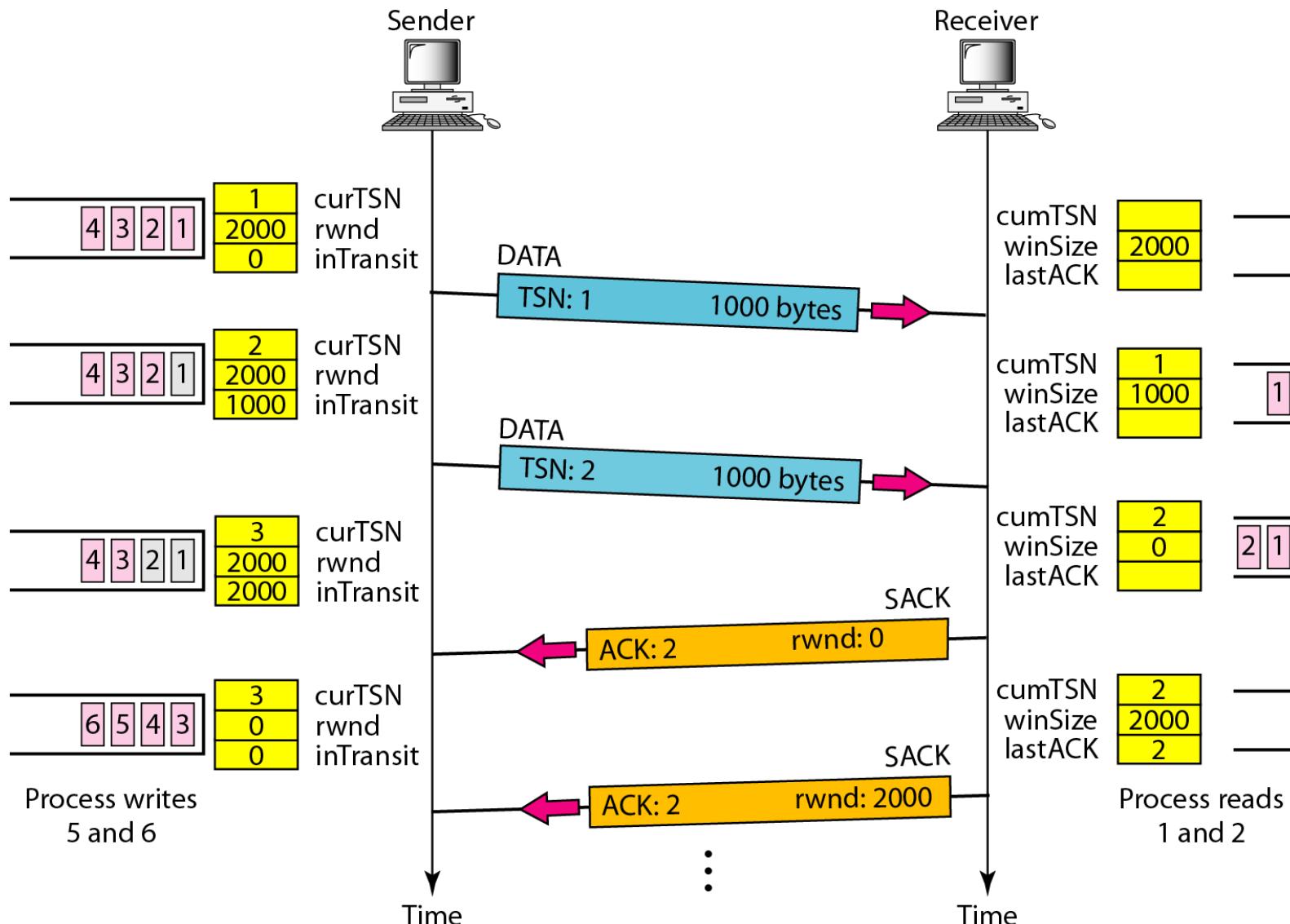


Figure 23.39 Error control, receiver site

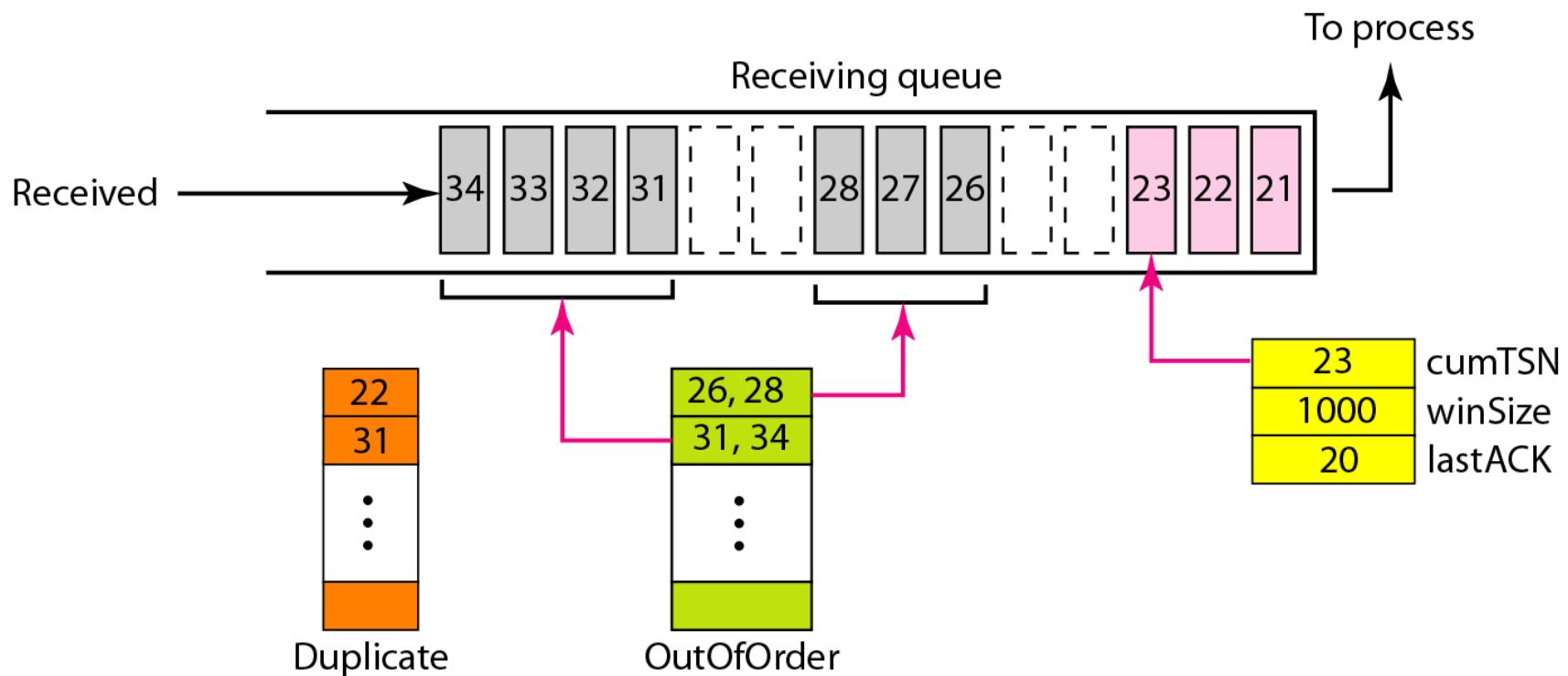
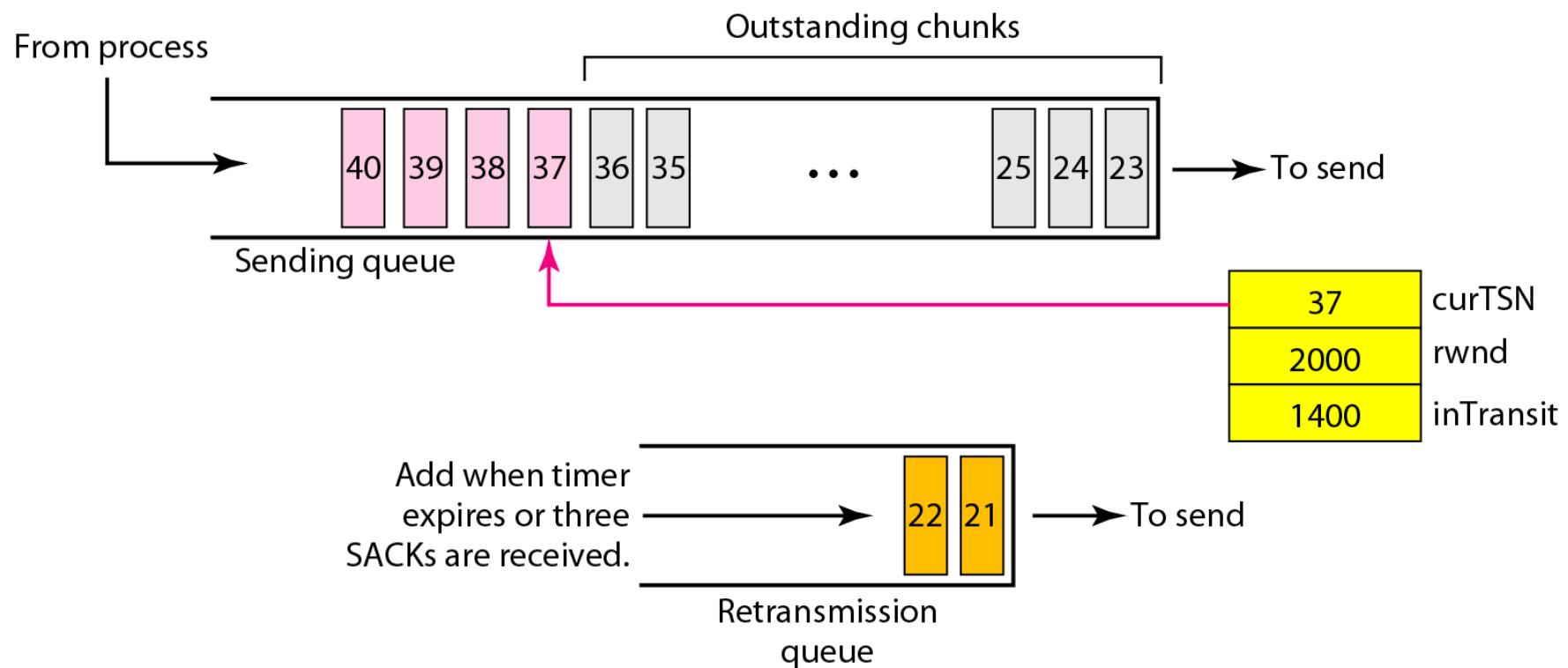


Figure 23.40 Error control, sender site





Data Communications and Networking

Fourth Edition

Forouzan

Chapter 24

Congestion Control and Quality of Service

24-1 DATA TRAFFIC

*The main focus of congestion control and quality of service is **data traffic**. In congestion control we try to avoid traffic congestion. In quality of service, we try to create an appropriate environment for the traffic. So, before talking about congestion control and quality of service, we discuss the data traffic itself.*

Topics discussed in this section:

Traffic Descriptor
Traffic Profiles

Figure 24.1 *Traffic descriptors*

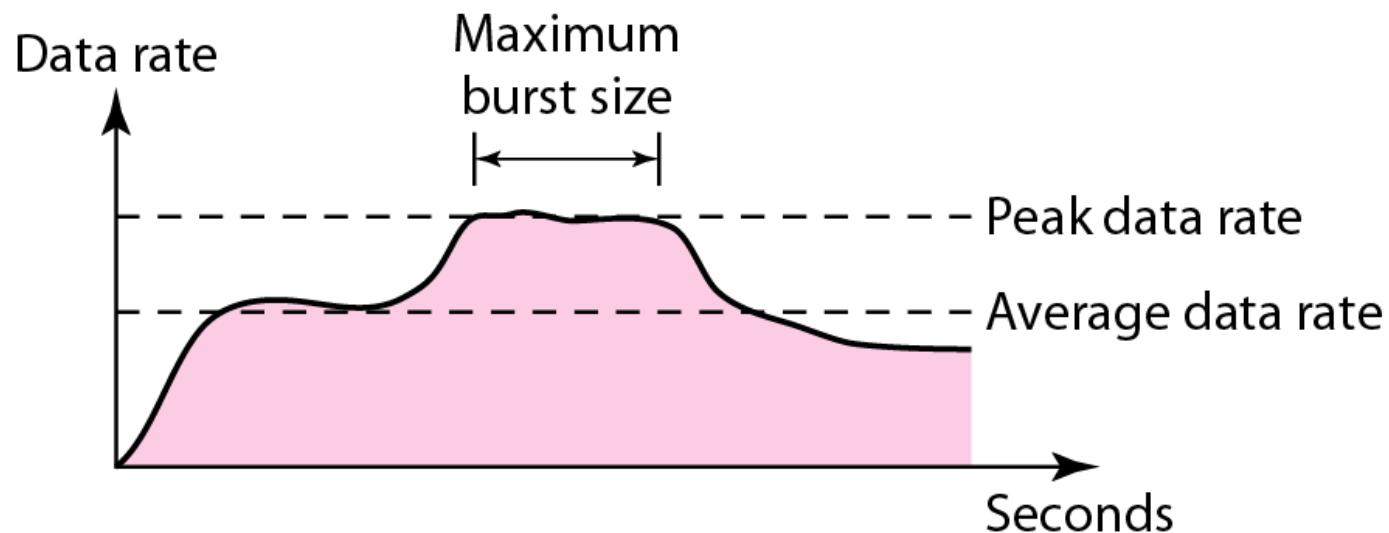
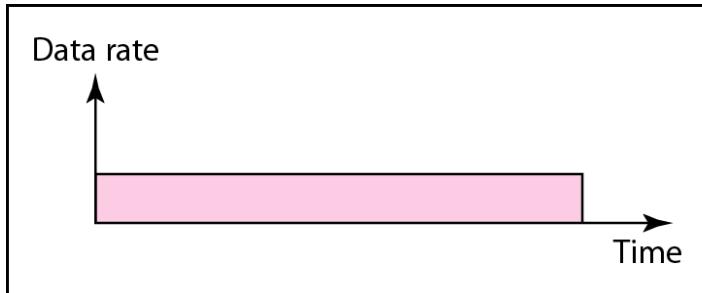
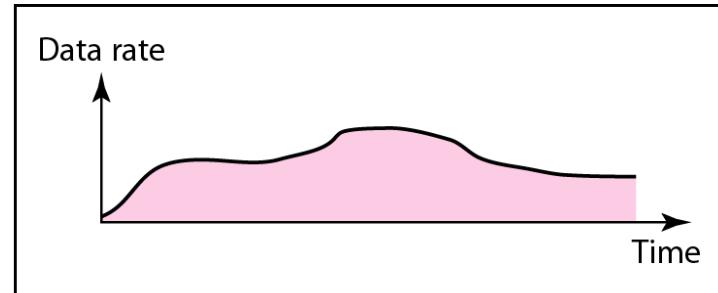


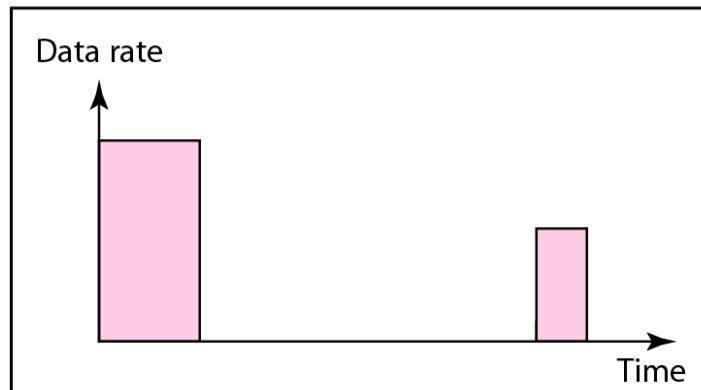
Figure 24.2 *Three traffic profiles*



a. Constant bit rate



b. Variable bit rate



c. Bursty

24-2 CONGESTION

Congestion in a network may occur if the load on the network—the number of packets sent to the network—is greater than the capacity of the network—the number of packets a network can handle. Congestion control refers to the mechanisms and techniques to control the congestion and keep the load below the capacity.

Topics discussed in this section:

Network Performance

Figure 24.3 *Queues in a router*

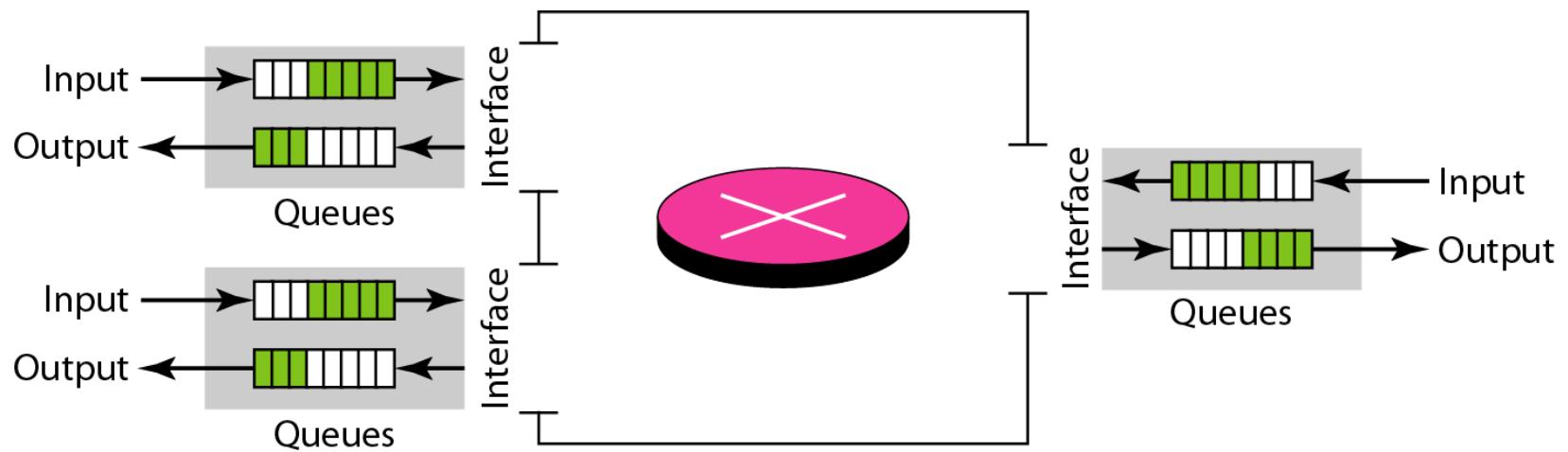
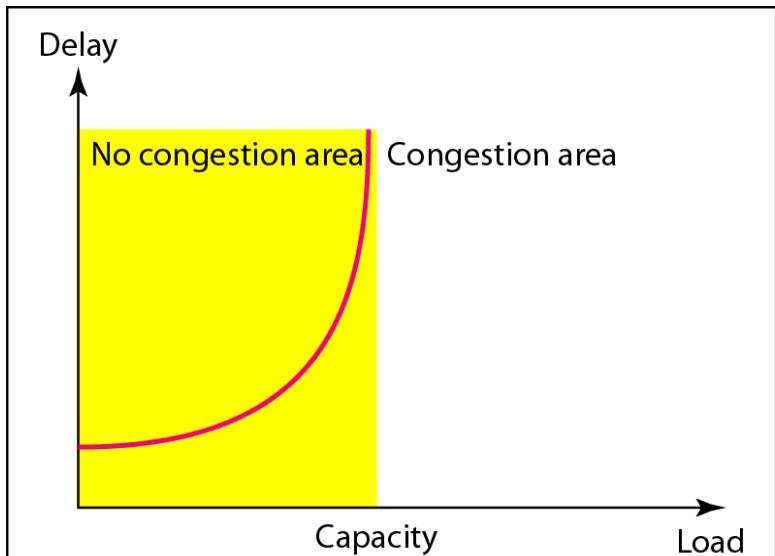
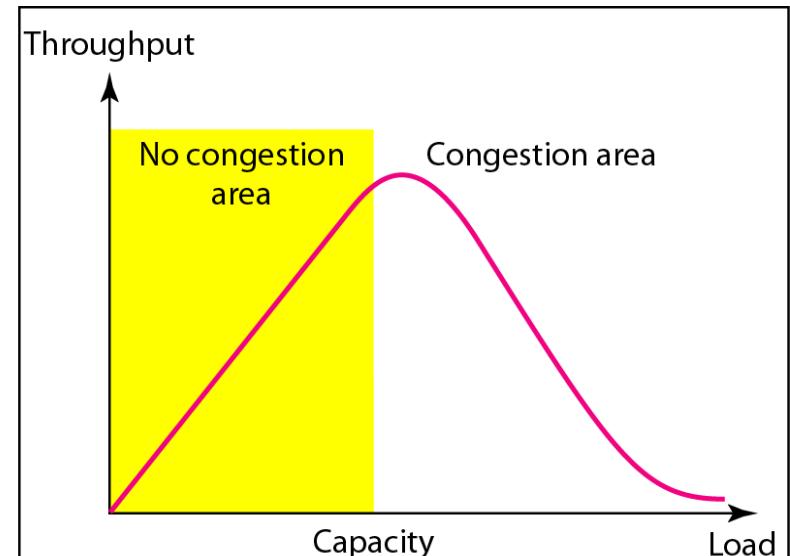


Figure *Packet delay and throughput as functions of load*



a. Delay as a function of load



b. Throughput as a function of load

24-3 CONGESTION CONTROL

Congestion control refers to techniques and mechanisms that can either prevent congestion, before it happens, or remove congestion, after it has happened. In general, we can divide congestion control mechanisms into two broad categories: open-loop congestion control (prevention) and closed-loop congestion control (removal).

Topics discussed in this section:

Open-Loop Congestion Control

Closed-Loop Congestion Control

Figure 24.5 *Congestion control categories*

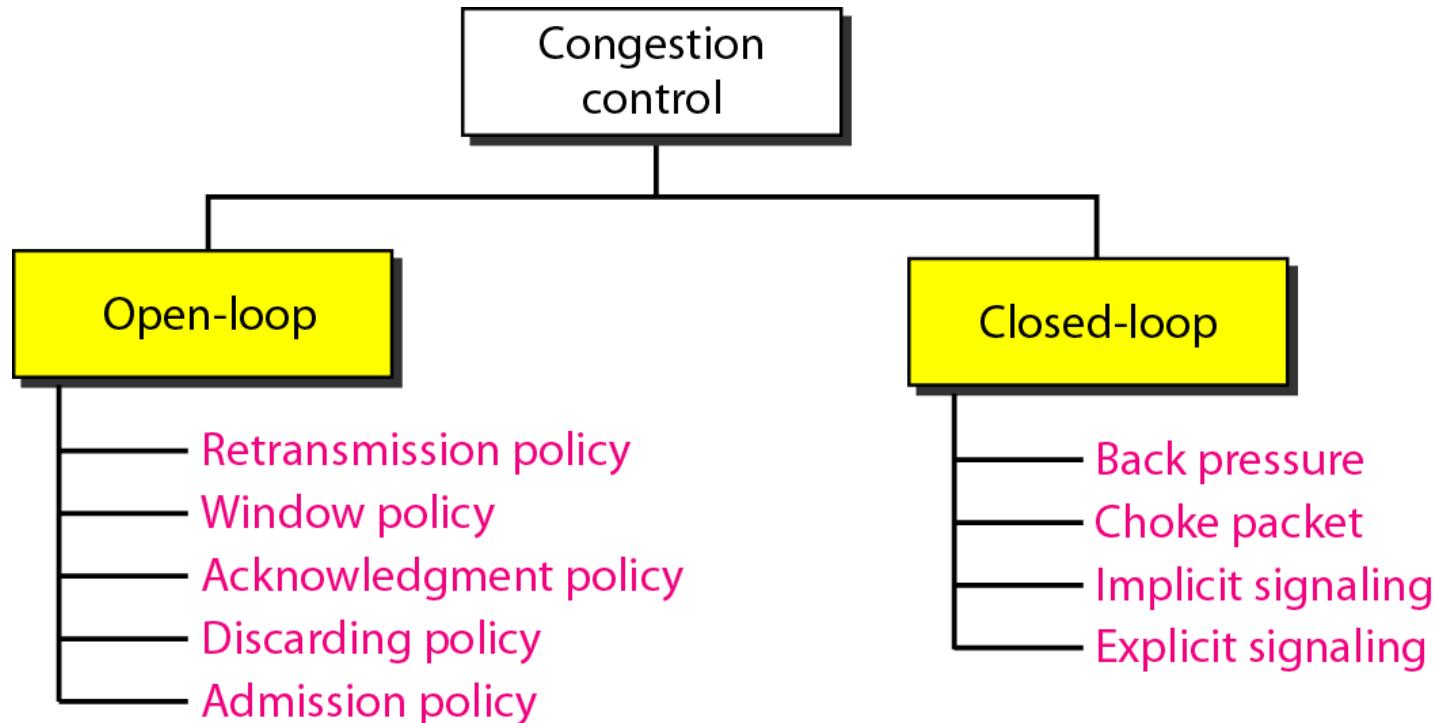


Figure 24.6 *Backpressure method for alleviating congestion*

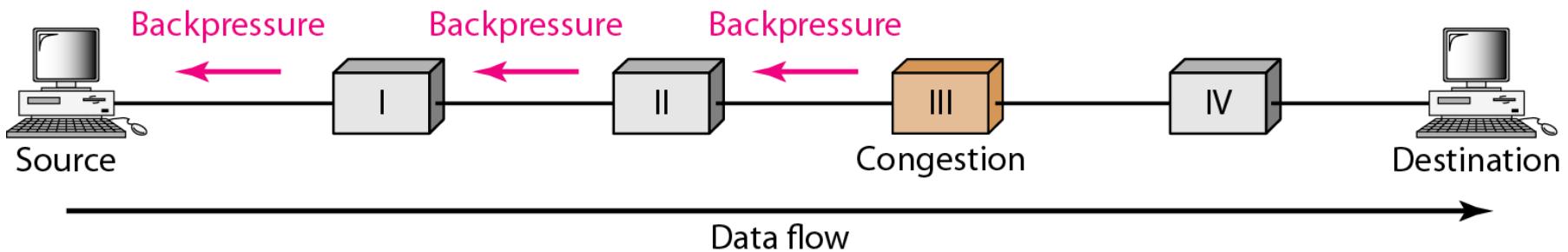
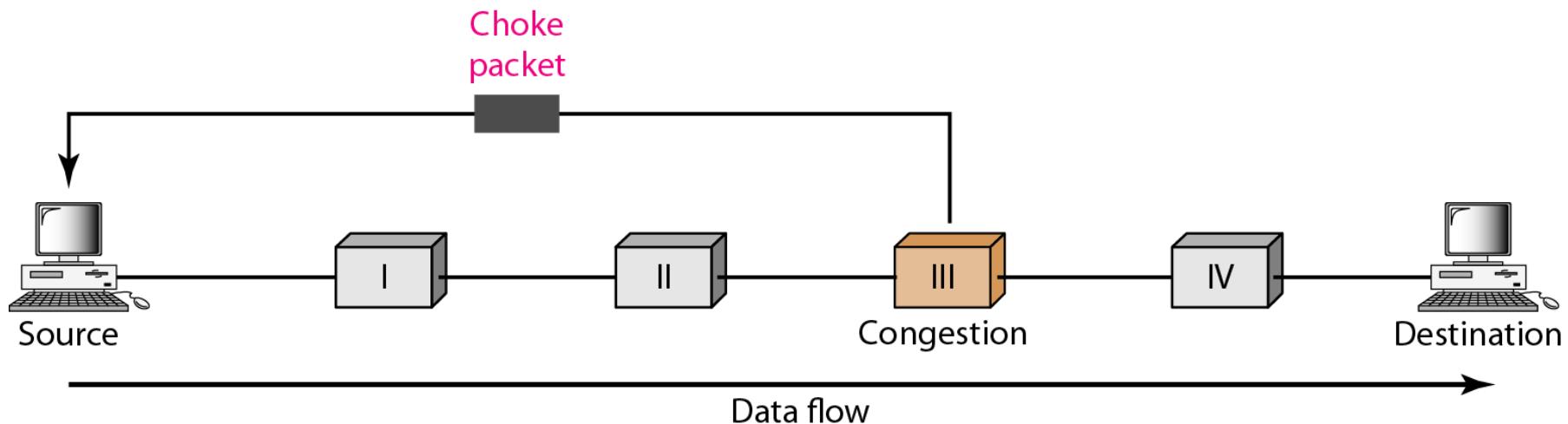


Figure 24.7 Choke packet



24-4 TWO EXAMPLES

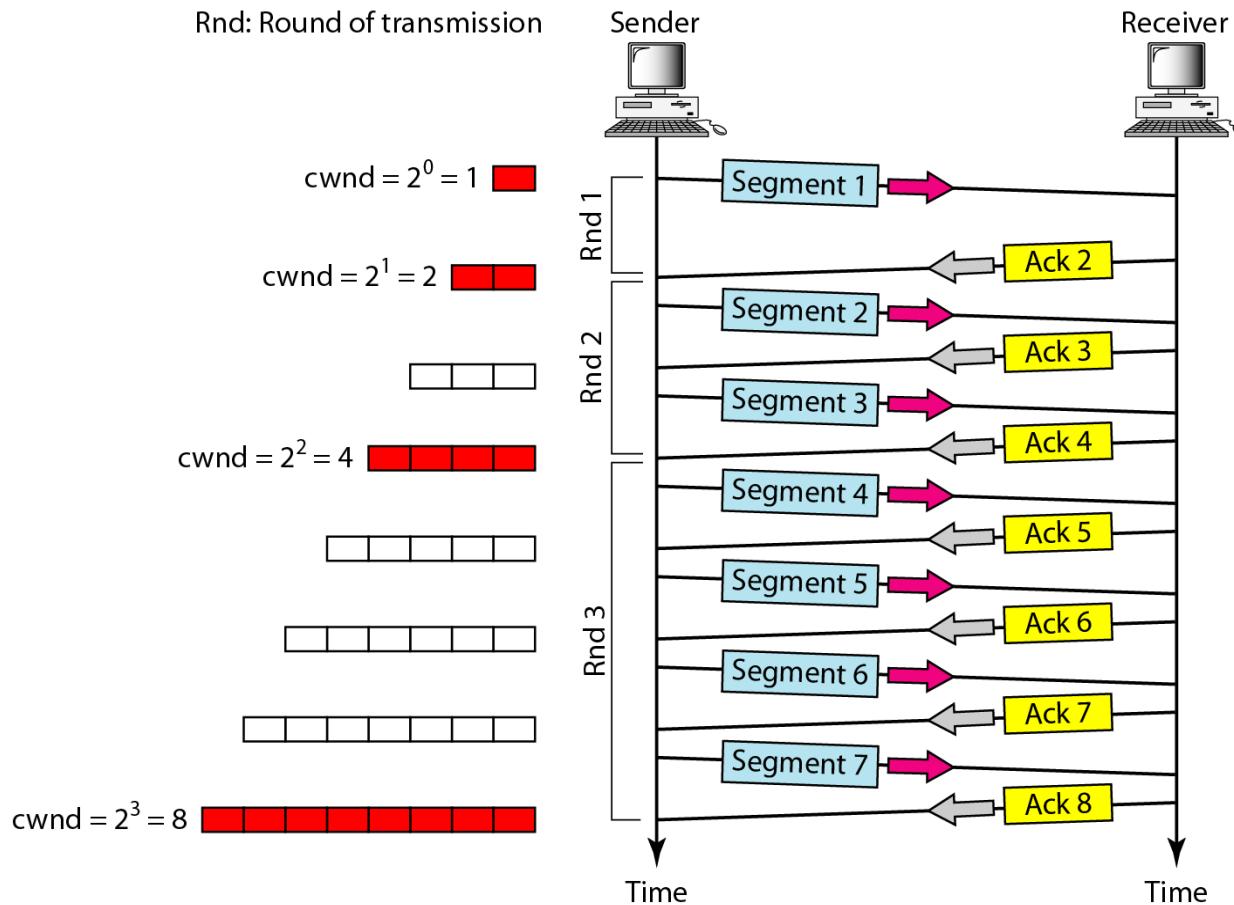
To better understand the concept of congestion control, let us give two examples: one in TCP and the other in Frame Relay.

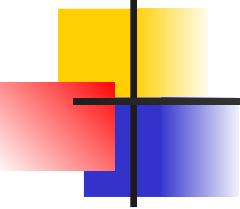
Topics discussed in this section:

Congestion Control in TCP

Congestion Control in Frame Relay

Figure 24.8 Slow start, exponential increase

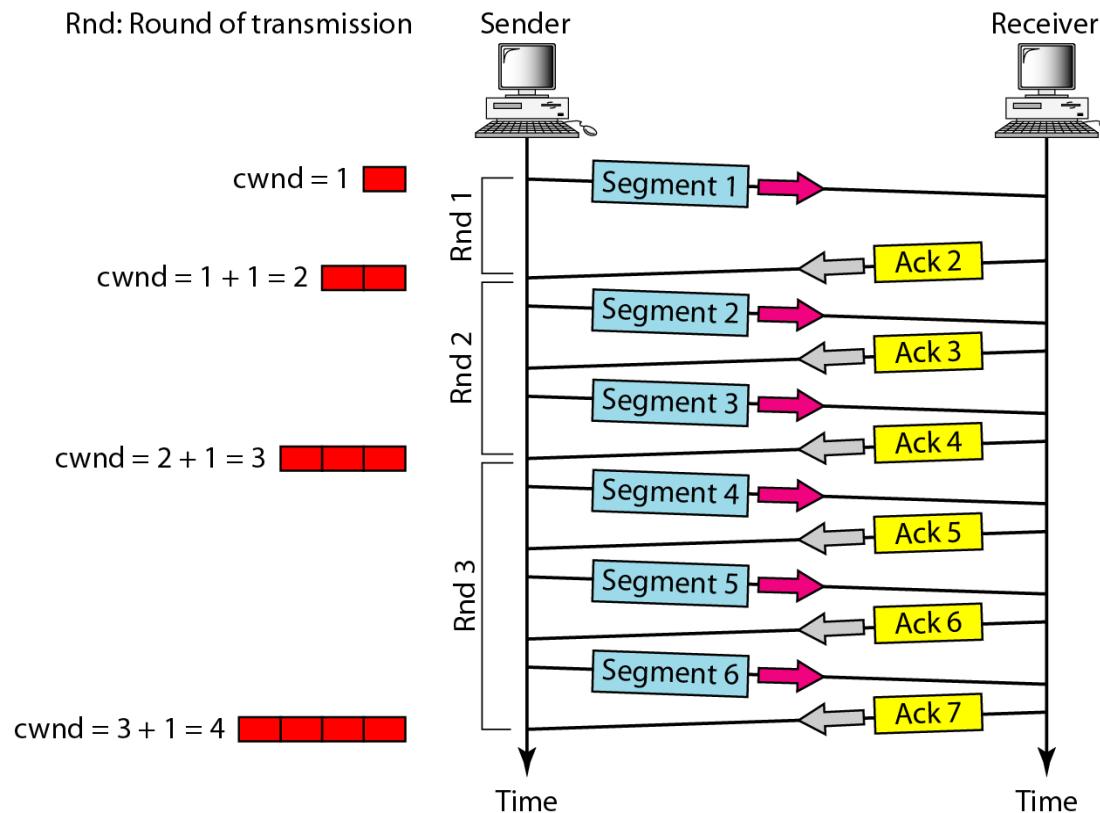


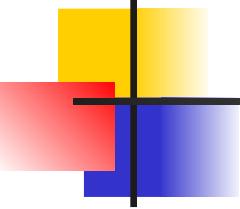


Note

In the slow-start algorithm, the size of the congestion window increases exponentially until it reaches a threshold.

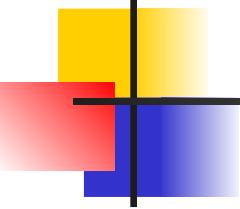
Figure 24.9 Congestion avoidance, additive increase





Note

**In the congestion avoidance algorithm,
the size of the congestion window
increases additively until
congestion is detected.**



Note

An implementation reacts to congestion detection in one of the following ways:

- If detection is by time-out, a new slow start phase starts.**
 - If detection is by three ACKs, a new congestion avoidance phase starts.**
-

Figure 24.10 TCP congestion policy summary

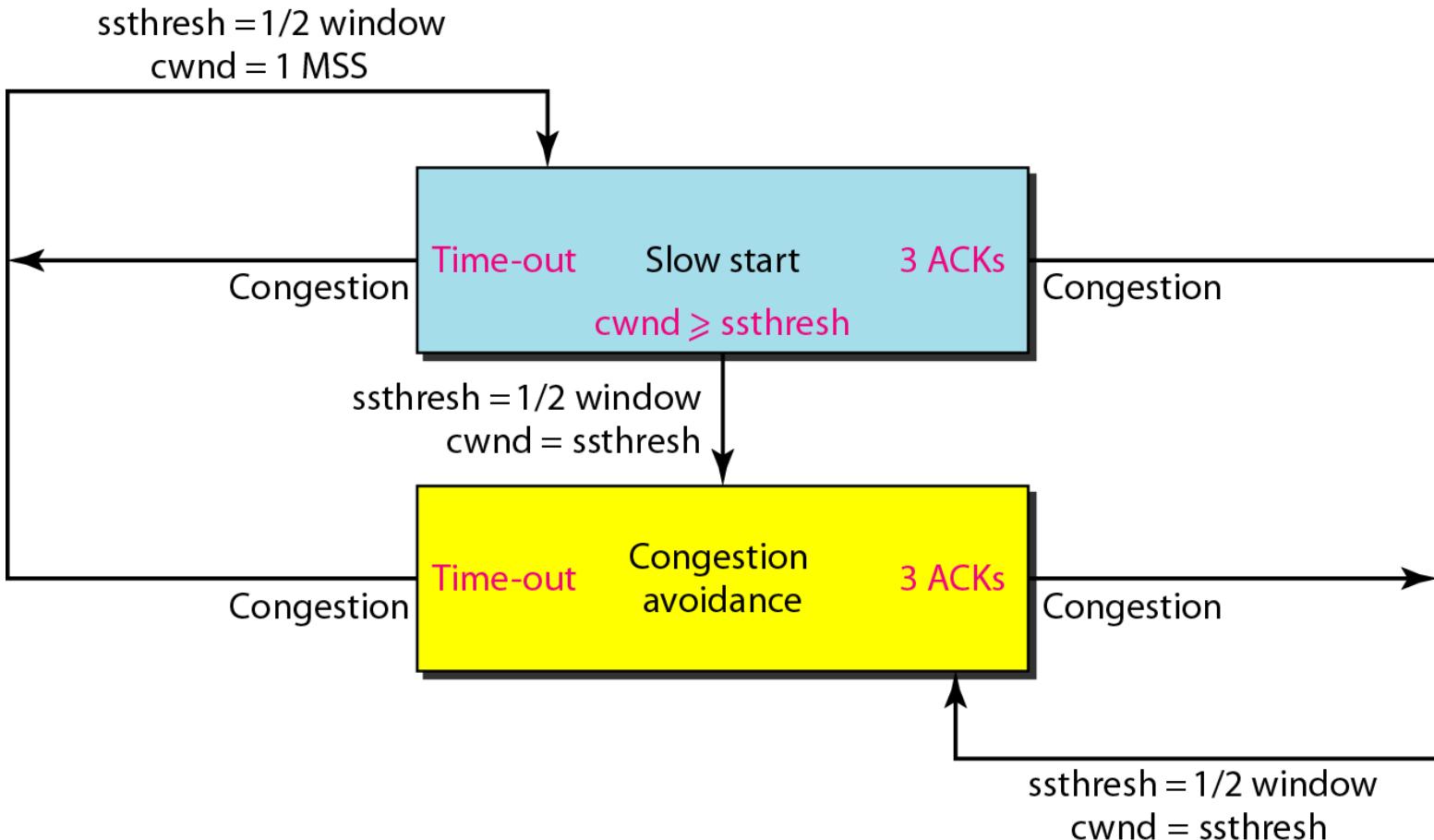


Figure 24.11 Congestion example

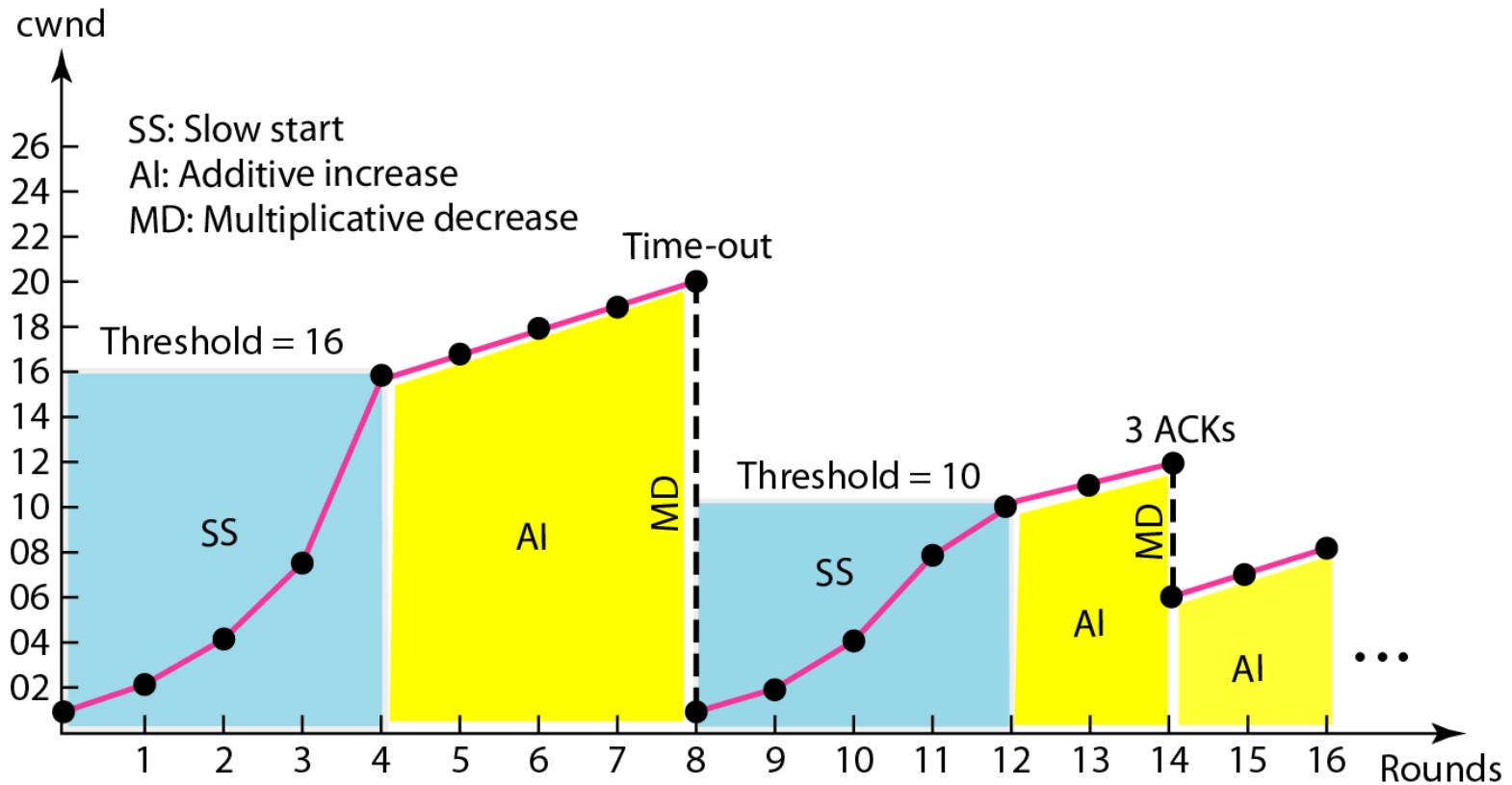


Figure 24.12 BECN

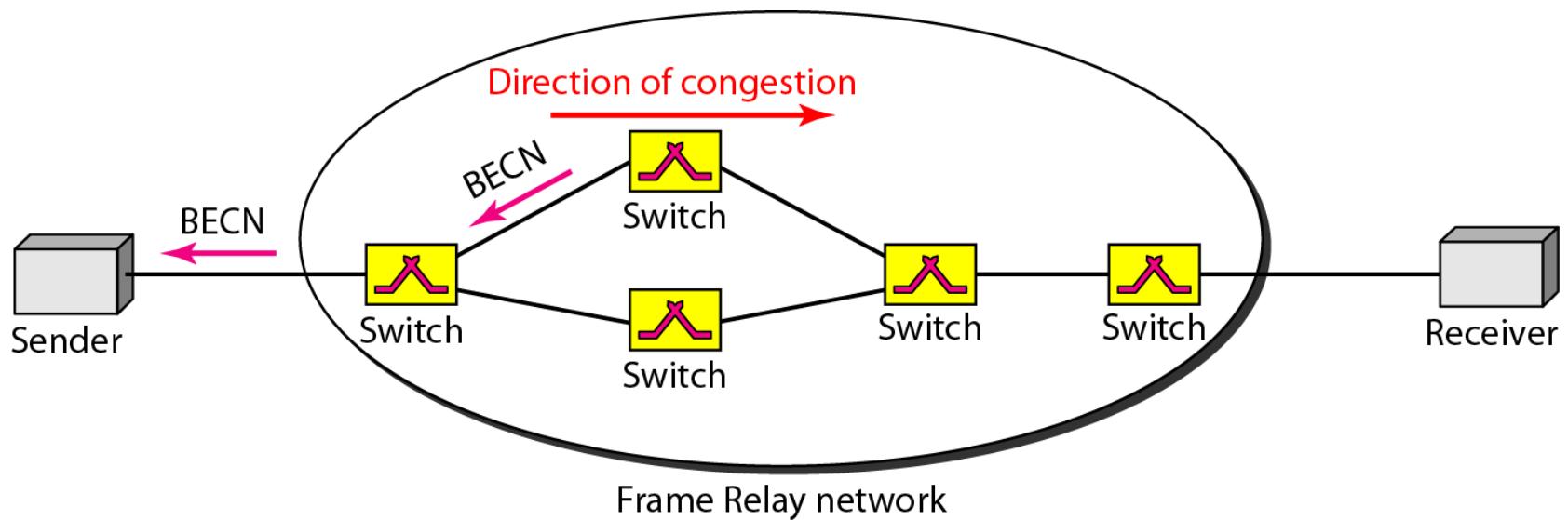


Figure 24.13 FECN

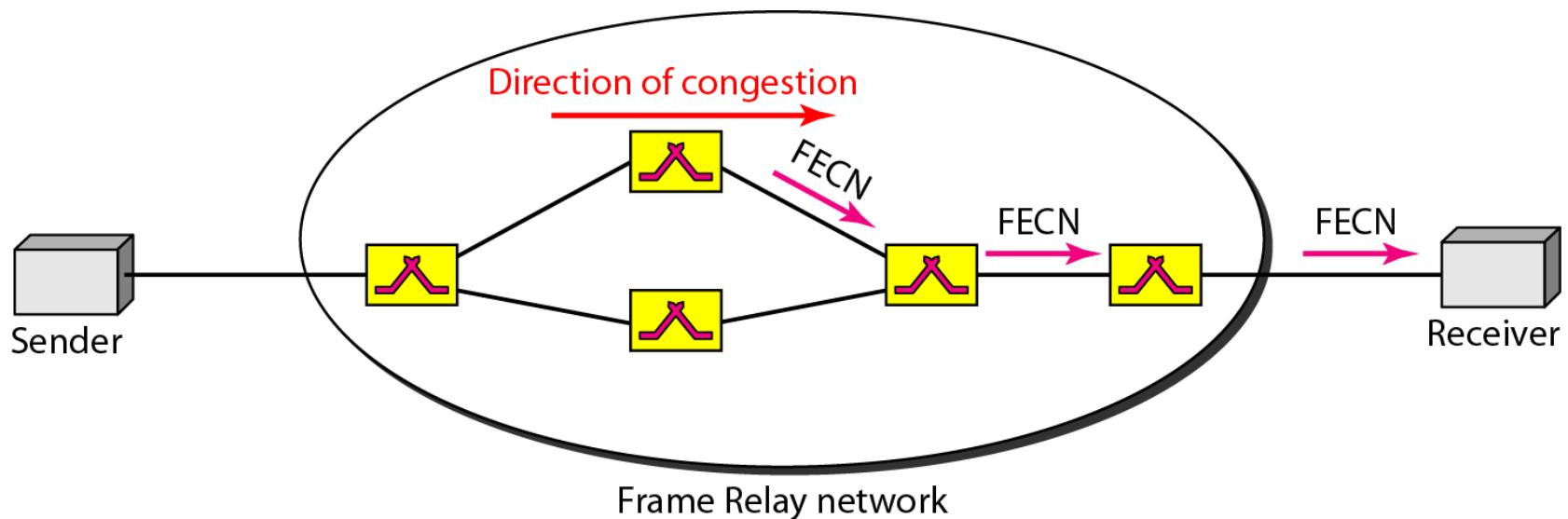
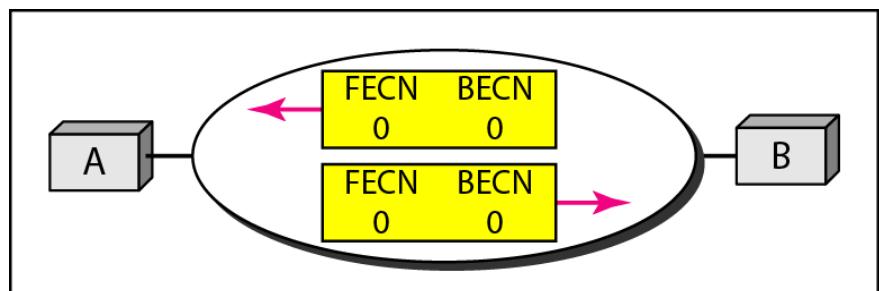
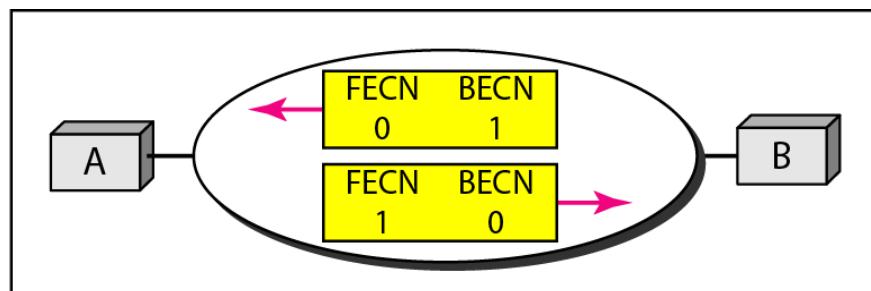


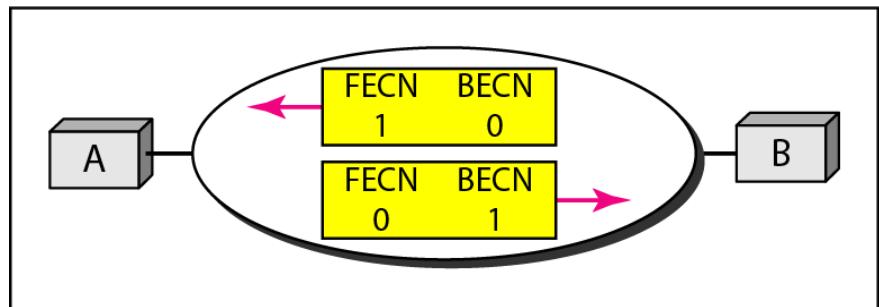
Figure 24.14 Four cases of congestion



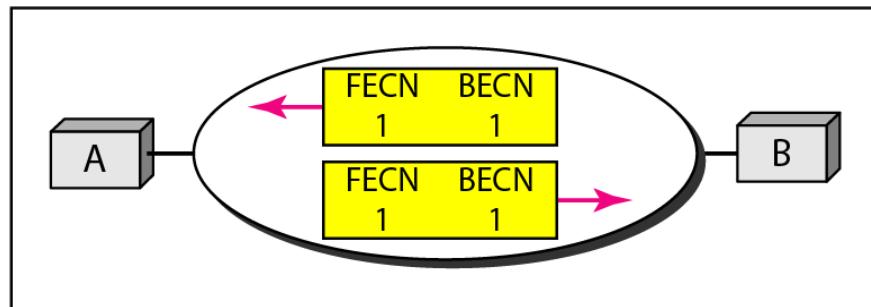
a. No congestion



b. Congestion in the direction A-B



c. Congestion in the direction B-A



d. Congestion in both directions

24-5 QUALITY OF SERVICE

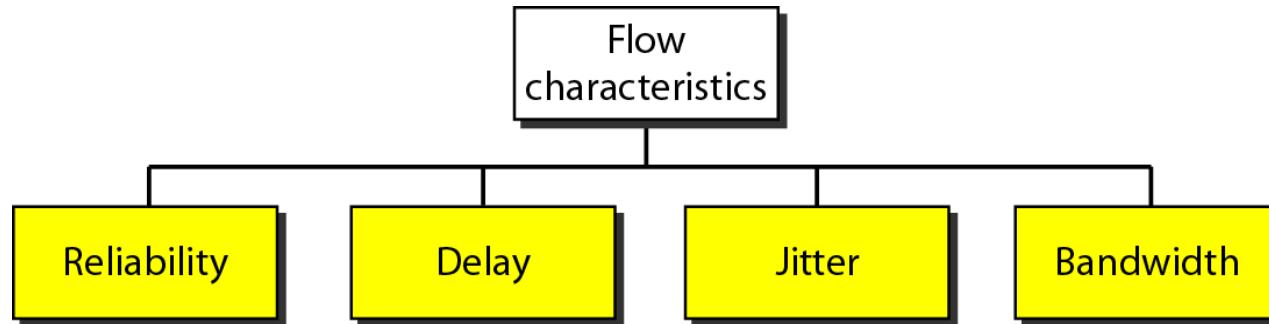
Quality of service (QoS) is an internetworking issue that has been discussed more than defined. We can informally define quality of service as something a flow seeks to attain.

Topics discussed in this section:

Flow Characteristics

Flow Classes

Figure 24.15 *Flow characteristics*



24-6 TECHNIQUES TO IMPROVE QoS

In Section 24.5 we tried to define QoS in terms of its characteristics. In this section, we discuss some techniques that can be used to improve the quality of service. We briefly discuss four common methods: scheduling, traffic shaping, admission control, and resource reservation.

Topics discussed in this section:

Scheduling

Traffic Shaping

Resource Reservation

Admission Control

Figure 24.16 FIFO queue

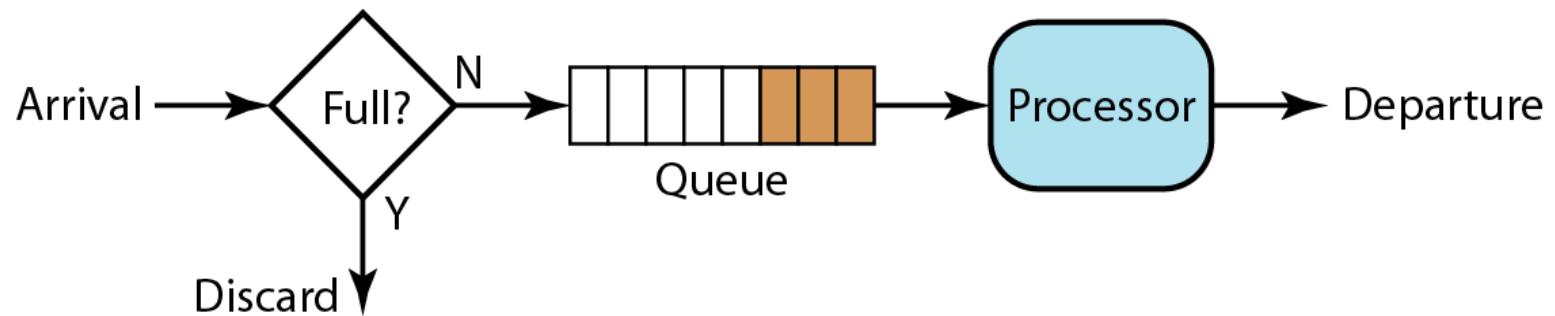


Figure 24.17 Priority queuing

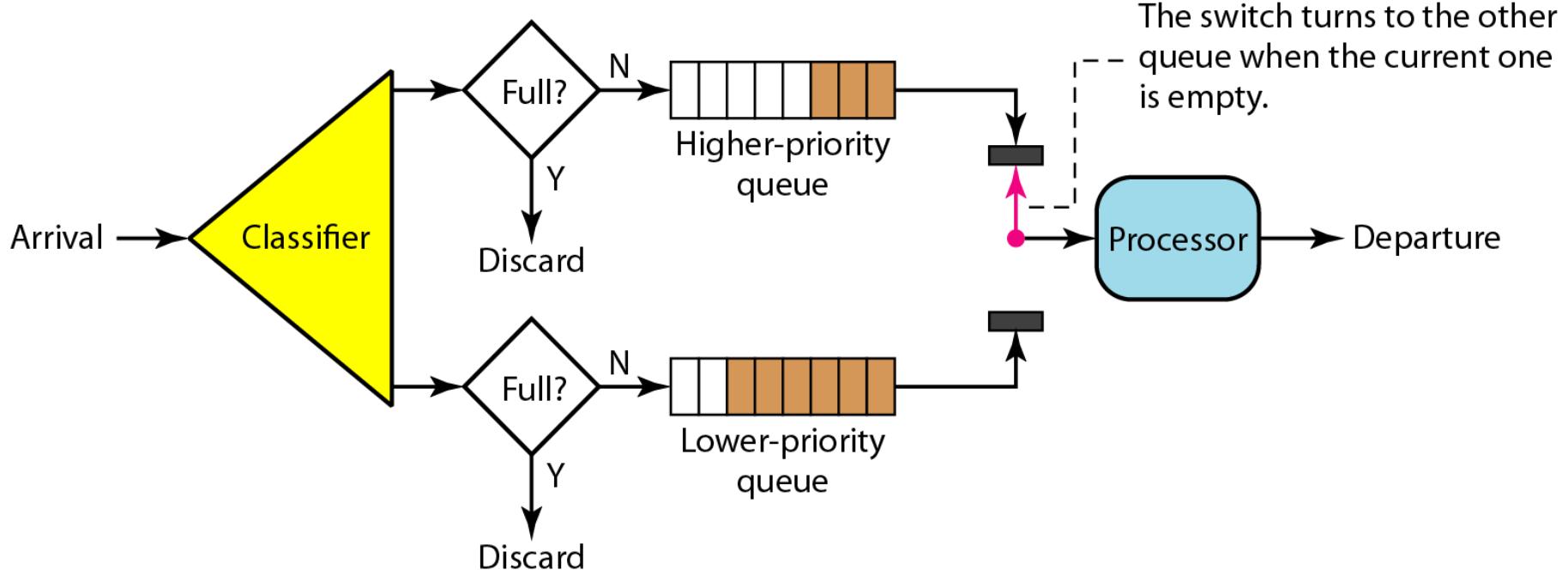


Figure 24.18 Weighted fair queuing

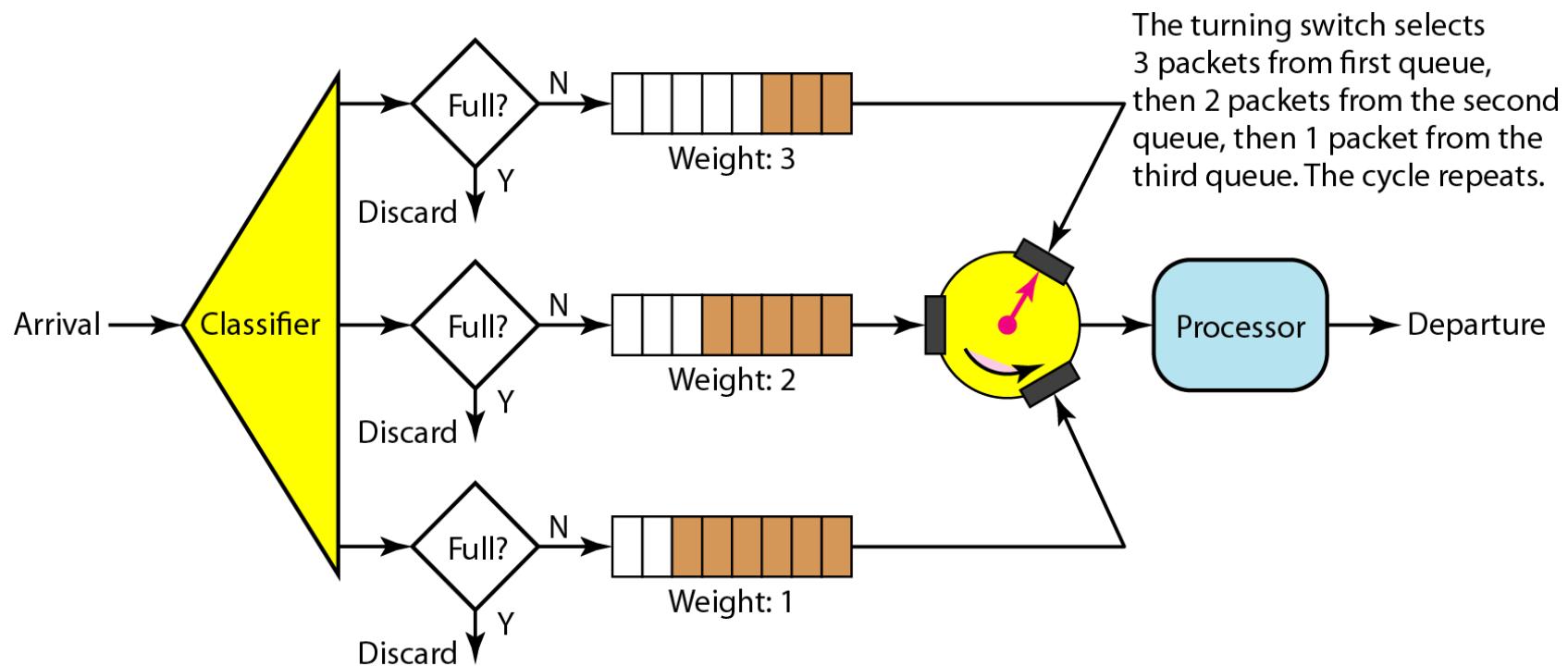


Figure 24.19 Leaky bucket

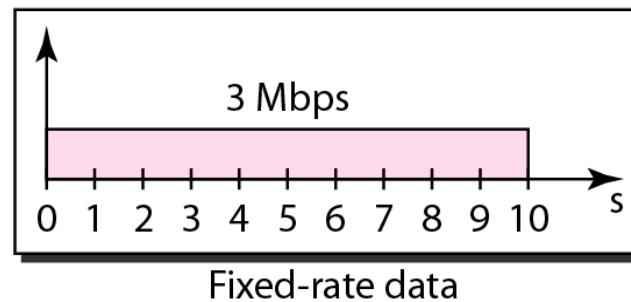
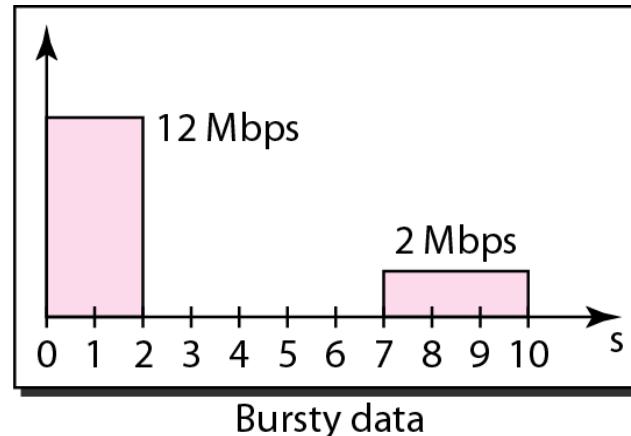
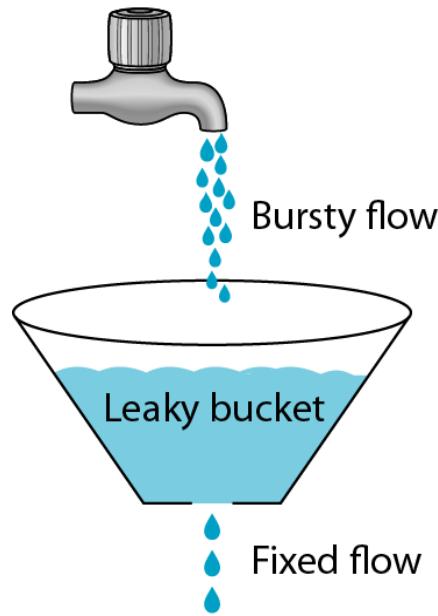
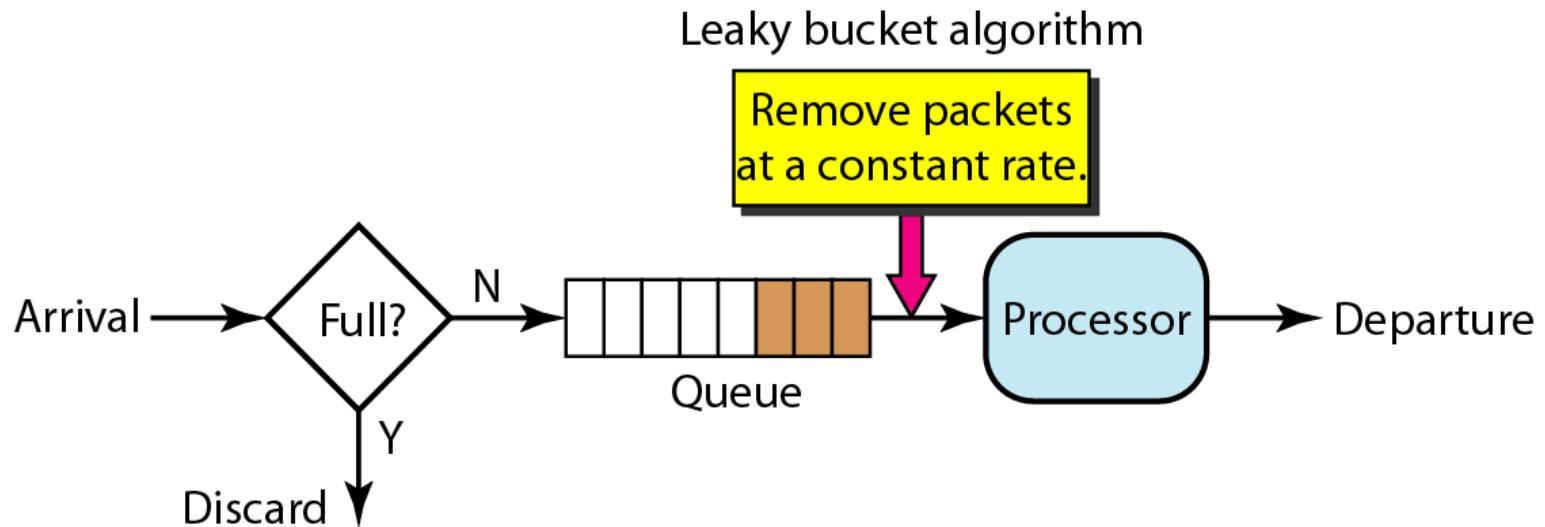
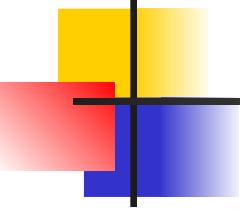


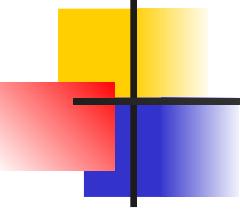
Figure 24.20 Leaky bucket implementation





Note

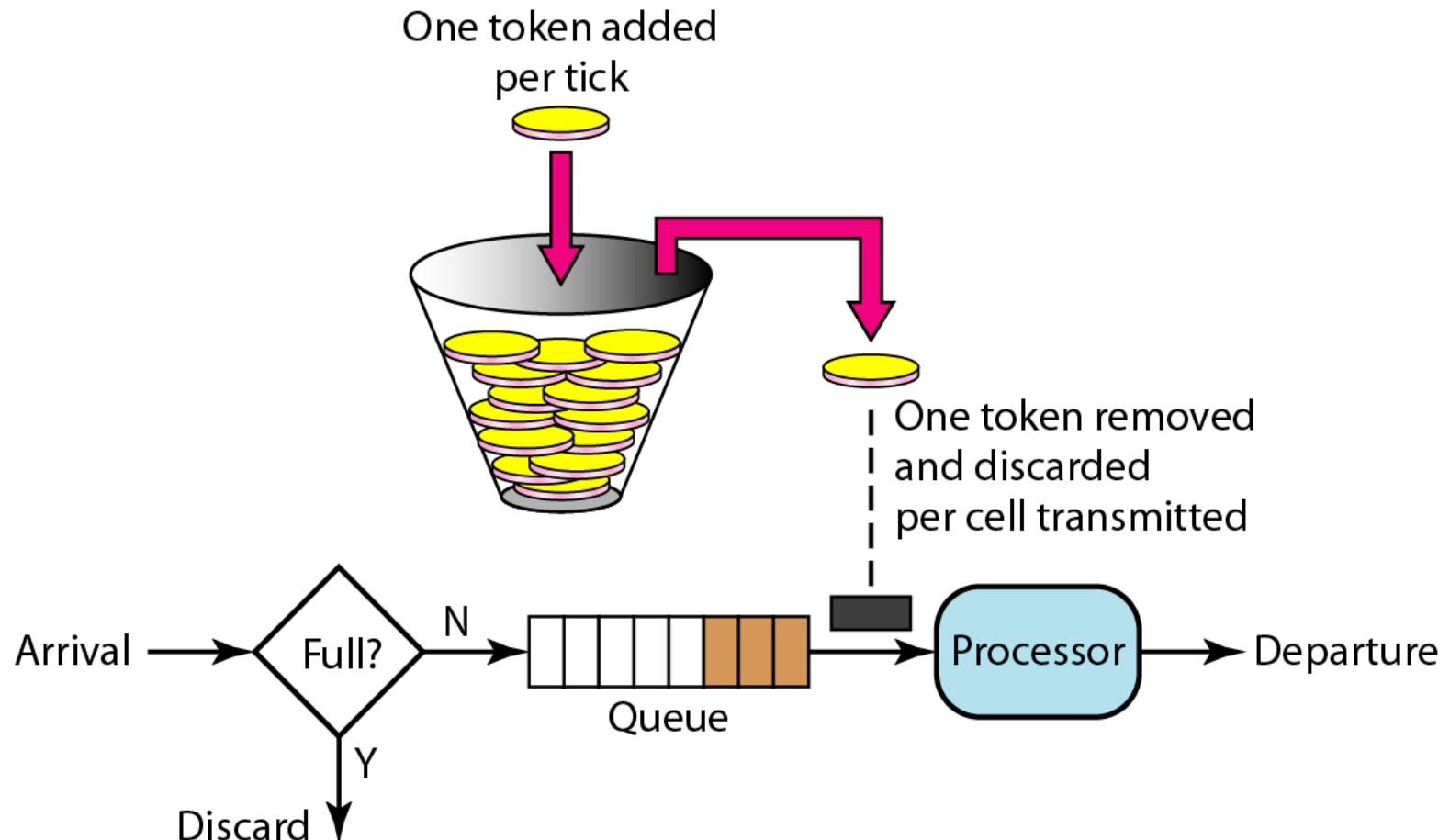
A leaky bucket algorithm shapes bursty traffic into fixed-rate traffic by averaging the data rate. It may drop the packets if the bucket is full.



Note

The token bucket allows bursty traffic at a regulated maximum rate.

Figure 24.21 Token bucket



24-7 INTEGRATED SERVICES

Two models have been designed to provide quality of service in the Internet: Integrated Services and Differentiated Services. We discuss the first model here.

Topics discussed in this section:

Signaling

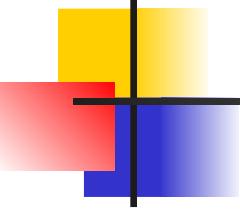
Flow Specification

Admission

Service Classes

RSVP

Problems with Integrated Services



Note

Integrated Services is a flow-based QoS model designed for IP.

Figure 24.22 Path messages

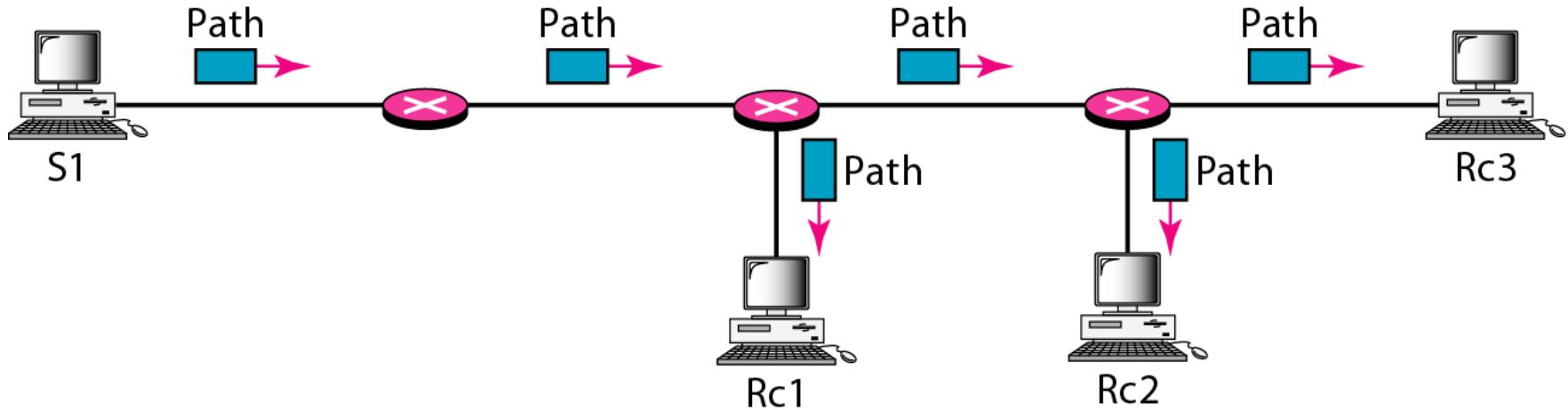


Figure 24.23 Resv messages

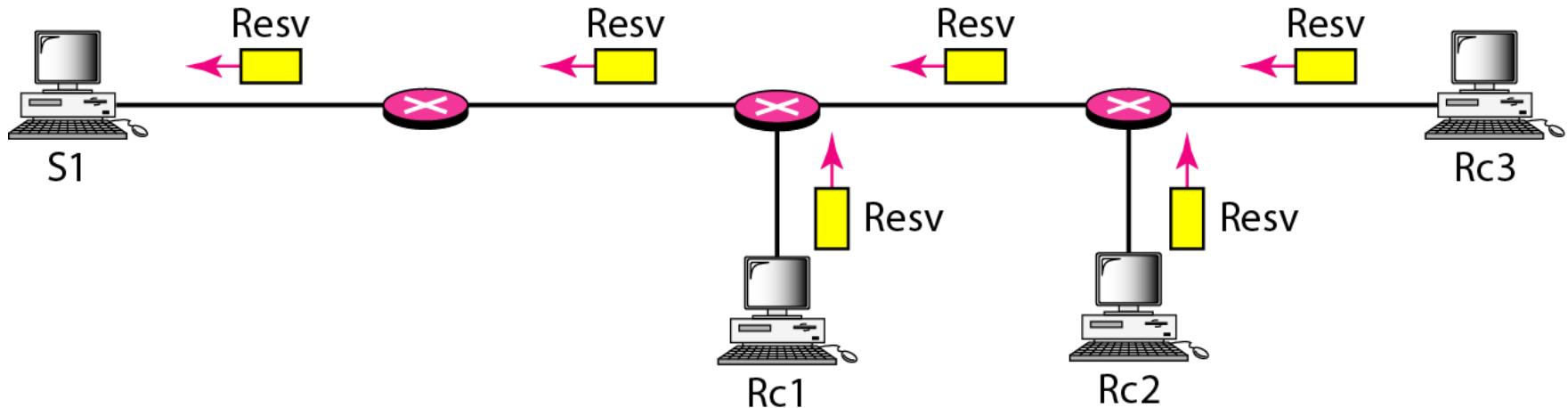


Figure 24.24 Reservation merging

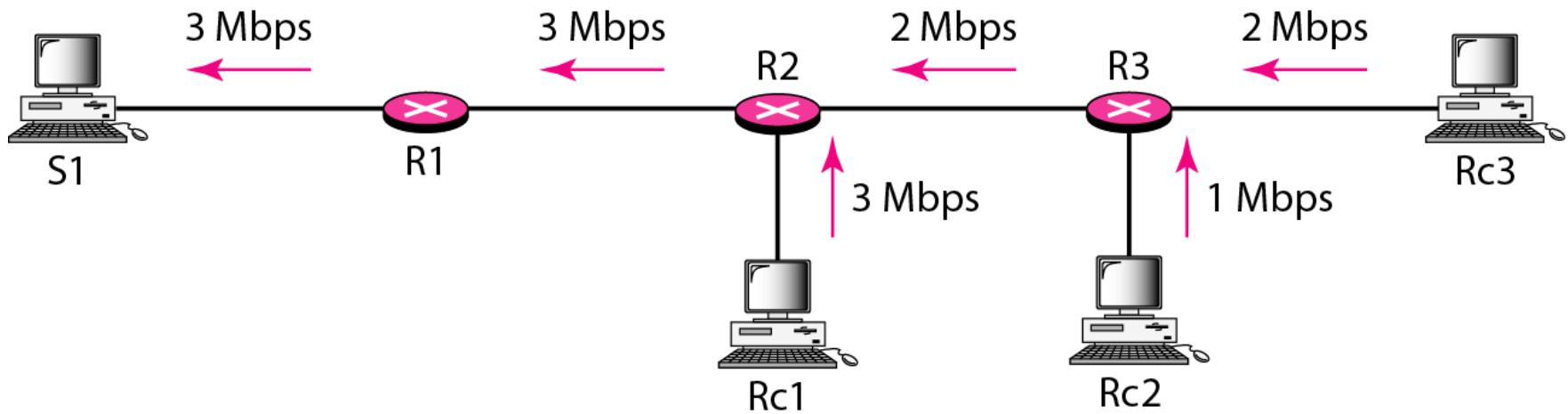
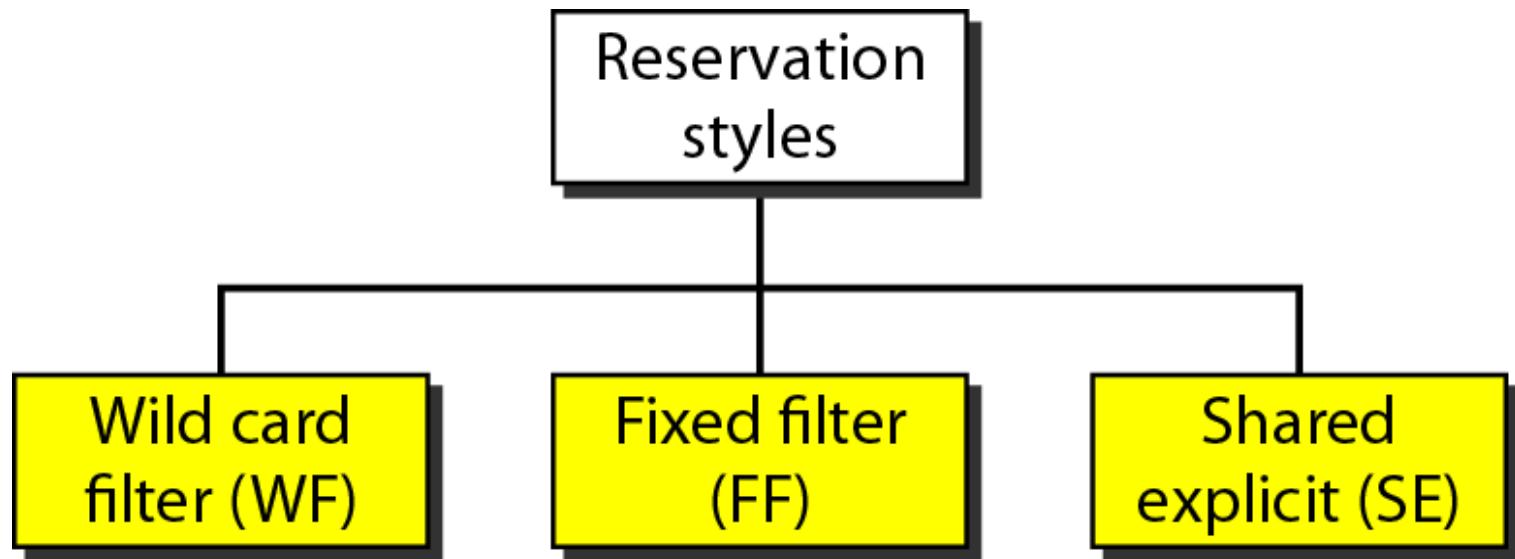


Figure 24.25 *Reservation styles*

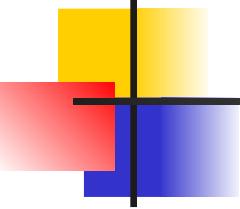


24-8 DIFFERENTIATED SERVICES

Differentiated Services (DS or Diffserv) was introduced by the IETF (Internet Engineering Task Force) to handle the shortcomings of Integrated Services.

Topics discussed in this section:

DS Field



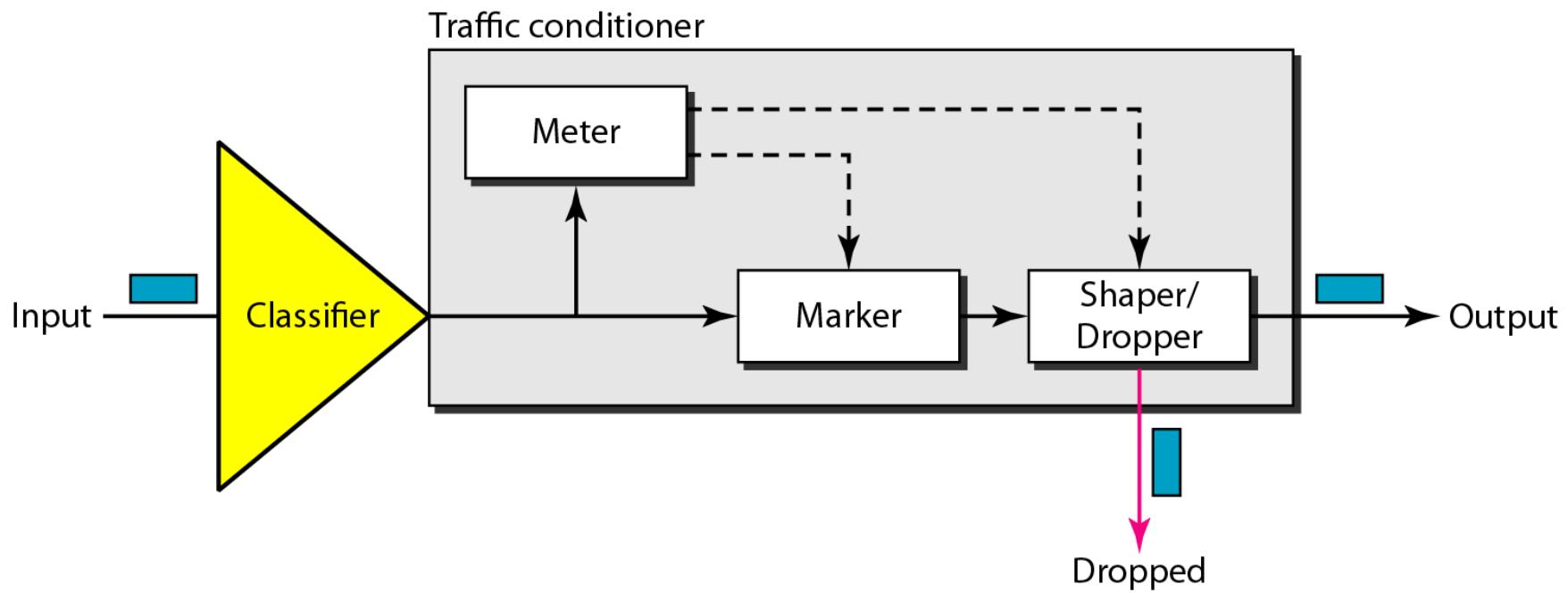
Note

Differentiated Services is a class-based QoS model designed for IP.

Figure 24.26 *DS field*



Figure 24.27 *Traffic conditioner*



24-9 QoS IN SWITCHED NETWORKS

Let us now discuss QoS as used in two switched networks: Frame Relay and ATM. These two networks are virtual-circuit networks that need a signaling protocol such as RSVP.

Topics discussed in this section:

QoS in Frame Relay

QoS in ATM

Figure 24.28 *Relationship between traffic control attributes*

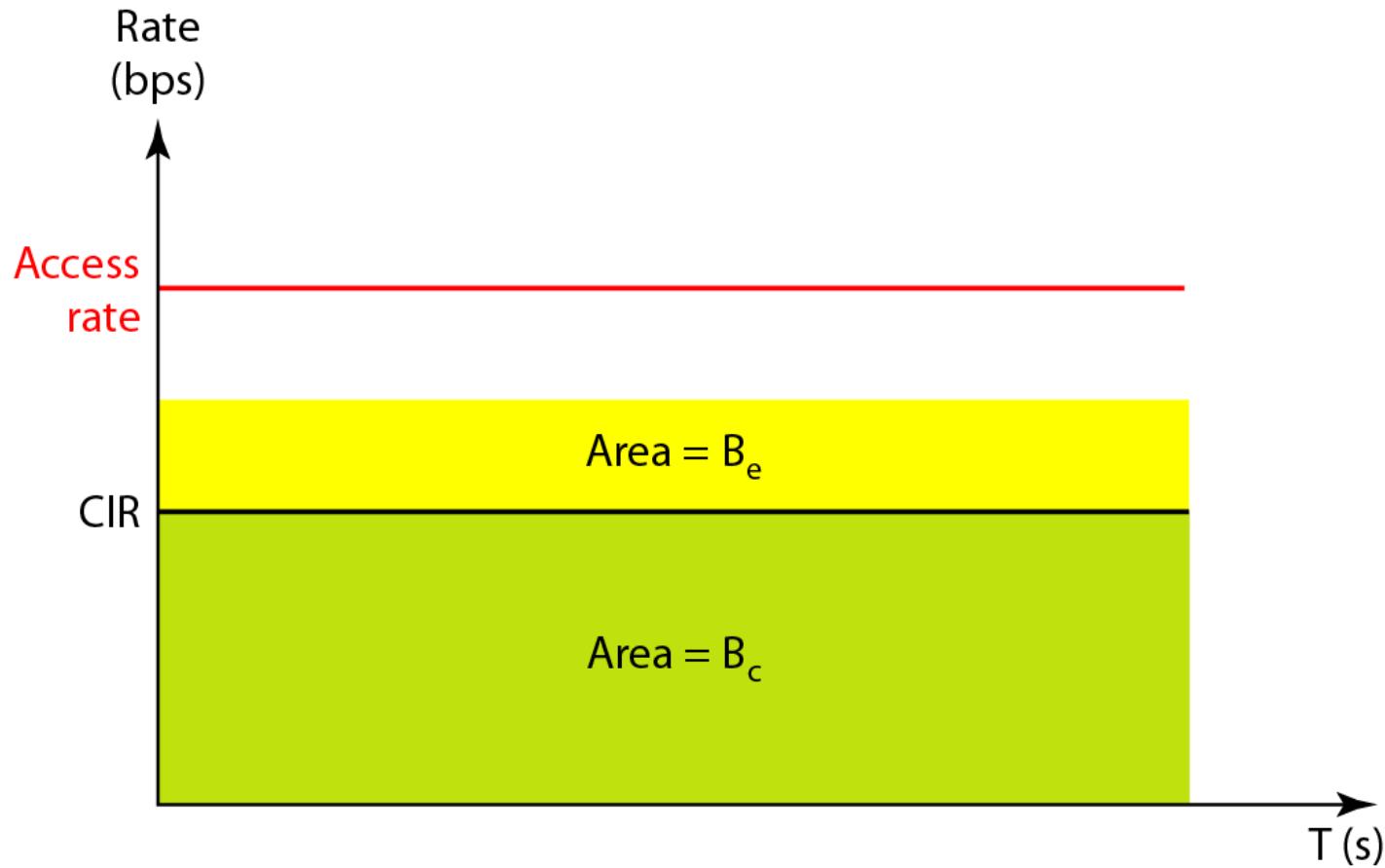


Figure 24.29 User rate in relation to B_c and $B_c + B_e$

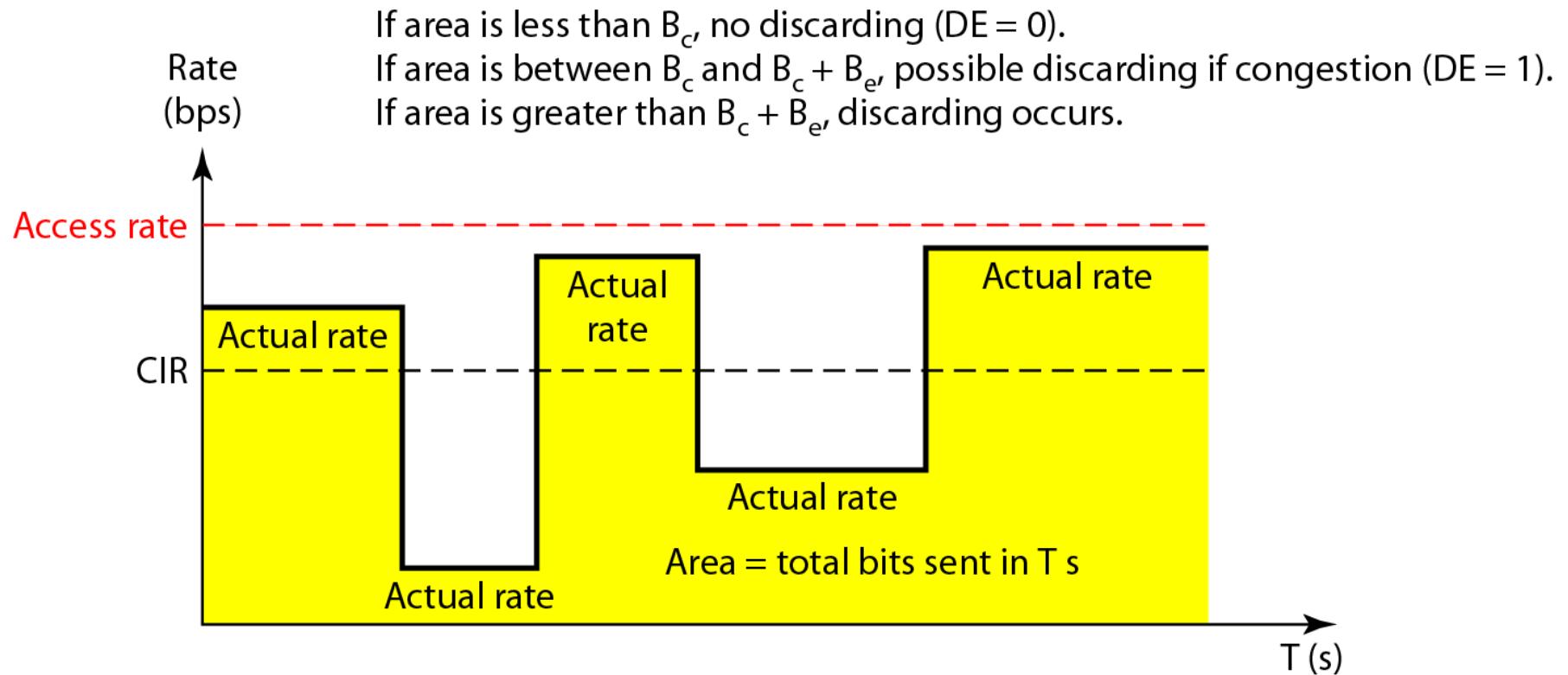


Figure 24.30 *Service classes*

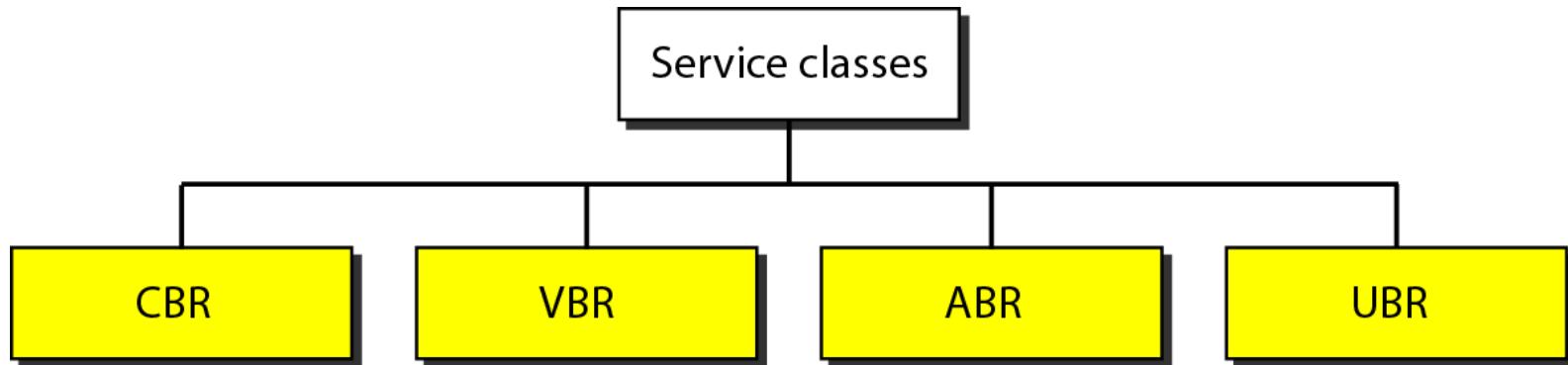


Figure 24.31 *Relationship of service classes to the total capacity of the network*

