Computer Network
Unit–IV

Network Layer

By:– Dr. D.R.Patil

• Outline

- Network Layer Design Issues, Communication Primitives: Unicast, Multicast, Broadcast.
- IPv4 Addressing (Classful and Classless), Subnetting, Supernetting Design Problems, IPv4 Protocol, Network Address Translation (NAT), Routing Algorithms: Shortest Path (Dijkastra's), Link State Routing, Distance Vector
- Routing Protocols, ARP, RARP, ICMP, IGMP Congestion Control Algorithms:
- Open Loop Congestion Control, Closed Loop Congestion Control, QoS Parameters, Token & Leaky Bucket Algorithms.

- **Network Layer Design Issues**
- The issues include the service provided to the transport layer and the internal design of the network.
- **Services Provided to the Transport Layer**
- The network layer provides services to the transport layer at the network layer/transport layer interface.
- An important question is precisely what kind of services the network layer provides to the transport layer.
- The services need to be carefully designed with the following goals in mind:

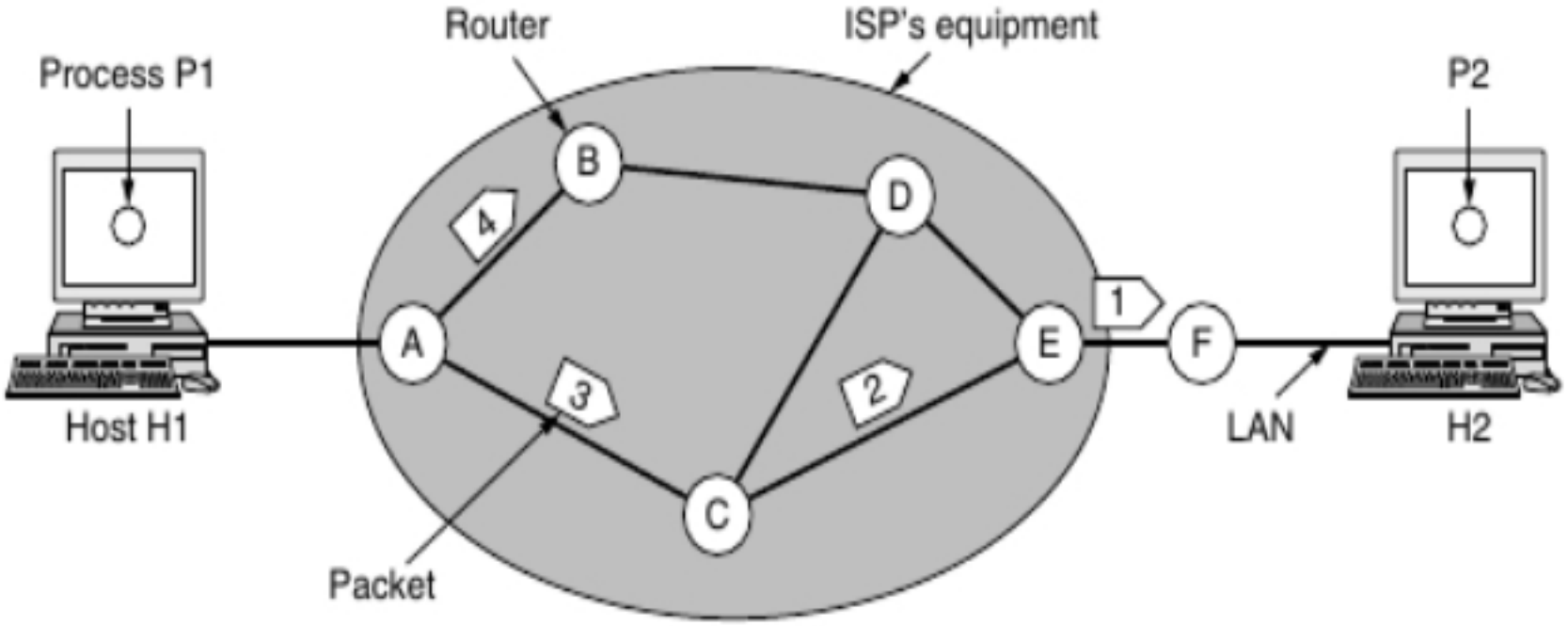
- 1. The services should be independent of the router technology.
- 2. The transport layer should be shielded from the number, type, and topology of the routers present.
- 3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

- Given these goals, the designers of the network layer have a lot of freedom in writing detailed specifications of the services to be offered to the transport layer.
- This freedom often degenerates into a raging battle between two warring factions.
- The discussion centers on whether the network layer should provide **connection–oriented service or connectionless service.**

- **Implementation of Connectionless Service**
- Having looked at the two classes of service the network layer can provide to its users, it is time to see how this layer works inside.
- Two different organizations are possible, depending on the type of service offered.
- If connectionless service is offered, packets are injected into the network individually and routed independently of each other.
- No advance setup is needed.
- In this context, the packets are frequently **called datagrams (in analogy with telegrams) and the network is called a datagram network.**

- **Implementation of Connectionless Service**
- If connection–oriented service is used, a path from the source router all the way to the destination router must be established before any data packets can be sent.
- **This connection is called a VC (virtual circuit), in analogy with the physical circuits set up by the telephone system, and the network is called a virtual–circuit network.**
- In this section, we will examine datagram networks; in the next one, we will examine virtual–circuit networks.

- **Implementation of Connectionless Service**
- If connection–oriented service is used, a path from the source router all the way to the destination router must be established before any data packets can be sent.
- **This connection is called a VC (virtual circuit), in analogy with the physical circuits set up by the telephone system, and the network is called a virtual–circuit network.**
- In this section, we will examine datagram networks; in the next one, we will examine virtual–circuit networks.



A's table (initially)

Dest.	Line
A	–
B	B
C	C
D	B
E	C
F	C

A's table (later)

A	–
B	B
C	C
D	B
E	B
F	B

C's table

A	A
B	A
C	–
D	E
E	E
F	E

E's table

A	C
B	D
C	C
D	D
E	–
F	F

Figure 5-2. Routing within a datagram network.

- **Implementation of Connection–Oriented Service**
- For connection–oriented service, we need a virtual–circuit network.
- Let us see how that works.
- The idea behind virtual circuits is to avoid having to choose a new route for every packet sent, as in Fig. 5–2.
- Instead, when a connection is established, a route from the source machine to the destination machine is chosen as part of the connection setup and stored in tables inside the routers.
- That route is used for all traffic flowing over the connection, exactly the same way that the telephone system works.

- When the connection is released, the virtual circuit is also terminated.
- With connection–oriented service, each packet carries an identifier telling which virtual circuit it belongs to.

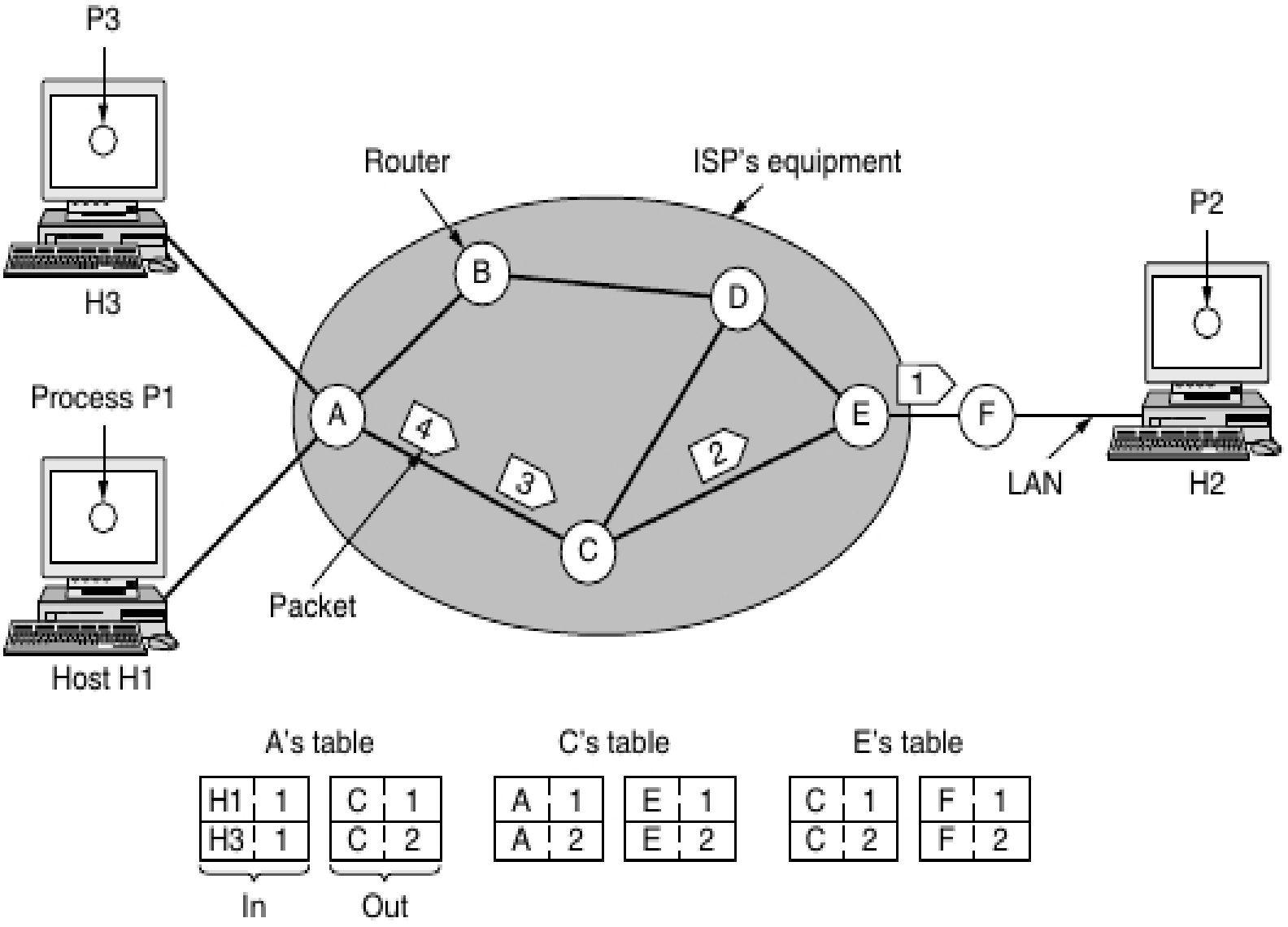


Figure 5-3. Routing within a virtual-circuit network.

- **Comparison of Virtual–Circuit and Datagram Networks**

Issue	Datagram network	Virtual-circuit network
Circuit setup	Not needed	Required
Addressing	Each packet contains the full source and destination address	Each packet contains a short VC number
State information	Routers do not hold state information about connections	Each VC requires router table space per connection
Routing	Each packet is routed independently	Route chosen when VC is set up; all packets follow it
Effect of router failures	None, except for packets lost during the crash	All VCs that passed through the failed router are terminated
Quality of service	Difficult	Easy if enough resources can be allocated in advance for each VC
Congestion control	Difficult	Easy if enough resources can be allocated in advance for each VC

Figure 5-4. Comparison of datagram and virtual-circuit networks.

- **Objectives**
  - The network layer is responsible for the source-to-destination delivery of a packet, possibly across multiple networks (links).
  - Whereas the data link layer oversees the delivery of the packet between two systems on the same network (links), the network layer ensures that each packet gets from its point of origin to its final destination.
  - The network layer adds a header that includes the logical addresses of the sender and receiver to the packet corning from the upper layer.
  - If a packet travels through the Internet, we need this addressing system to help distinguish the source and destination.

- **Network Layer: Logical Addressing**
  - The network layer is host-to-host (computer-to-computer); a computer somewhere in the world needs to communicate with another computer somewhere else in the world.
  - Usually, computers communicate through the Internet.
  - The packet transmitted by the sending computer may pass through several LANs or WANs before reaching the destination computer.
  - For this level of communication, **we need a global addressing scheme; we called this logical addressing** .
  - Today, we use the term IP address to mean a logical address in the network layer of the TCP/IP protocol suite.
  - The Internet addresses are 32 bits in length; this gives us a maximum of 2^{32} addresses.
  - These addresses are referred to as IPv4 (IP version 4) addresses or simply IP addresses if there is no confusion.

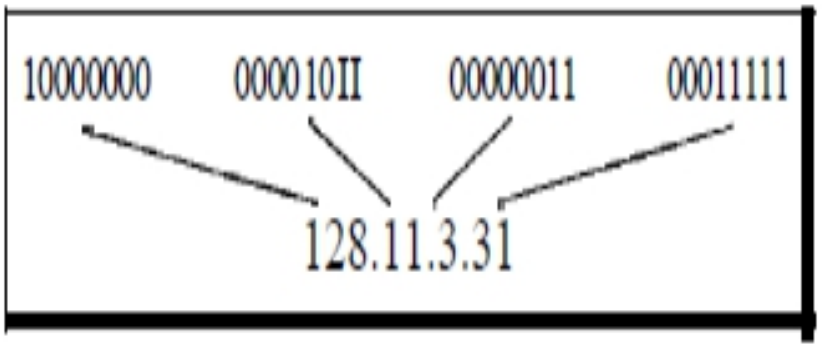
- **IPv4 ADDRESSES**
  - An IPv4 address is a 32-bit address that uniquely and universally defines the connection of a device (for example, a computer or a router) to the Internet.
  - IPv4 addresses are unique.
  - They are unique in the sense that each address defines one, and only one, connection to the Internet.
  - Two devices on the Internet can never have the same address at the same time.
  - The IPv4 addresses are universal in the sense that the addressing system must be accepted by any host that wants to be connected to the Internet.

- **Address Space**
  - A protocol such as IPv4 that defines addresses has an address space.
  - **An address space is the total number of addresses used by the protocol.**
  - If a protocol uses N bits to define an address, the address space is 2^N because each bit can have two different values (0 or 1) and N bits can have 2^N values.
  - IPv4 uses 32-bit addresses, which means that the address space is 2^{32} or 4,294,967,296 (more than 4 billion).
  - This means that, theoretically, if there were no restrictions, more than 4 billion devices could be connected to the Internet.
  - **The address space of IPv4 is 2^{32} or 4,294,967,296**

- **Notations**
  - There are two prevalent notations to show an IPv4 address: binary notation and dotteddecimal notation.
  - **Binary Notation**
  - In binary notation, the IPv4 address is displayed as 32 bits.
  - Each octet is often referred to as a byte.
  - So it is common to hear an IPv4 address referred to as a 32-bit address or a 4-byte address.
  - The following is an example of an IPv4 address in binary notation:
  - 01110101 10010101 00011101 00000010
  - **Dotted-Decimal Notation**
  - To make the IPv4 address more compact and easier to read, Internet addresses are usually written in decimal form with a decimal point (dot) separating the bytes.
  - The following is the dotted~decimal notation of the above address:
  - 117.149.29.2

- Figure 19.1 shows an IPv4 address in both binary and dotted-decimal notation.
- Note that because each byte (octet) is 8 bits, each number in dotted-decimal notation is a value ranging from 0 to 255.

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

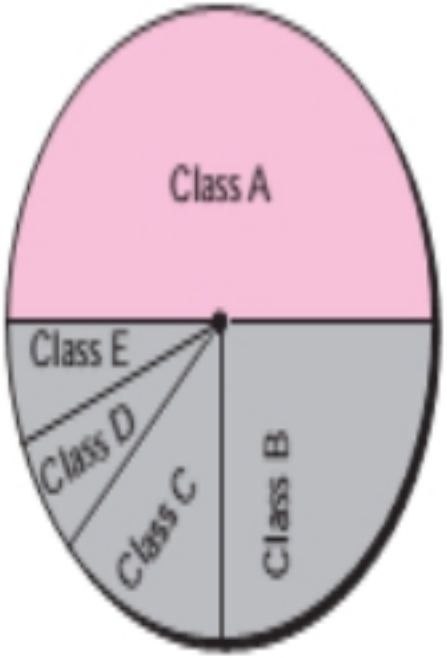


- Example 19.1
  - Change the following IPv4 addresses from binary notation to dotted-decimal notation.
  - a. 10000001 00001011 00001011 11101111
  - b. 11000001 10000011 00011011 11111111
  - Solution
  - We replace each group of 8 bits with its equivalent decimal number and add dots for separation.
  - 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 in an IPv4 address.
  - c. Each number needs to be less than or equal to 255 (301 is outside this range).
  - d. A mixture of binary notation and dotted-decimal notation is not allowed.

- **CLASSFUL ADDRESSING**
  - IP addresses, when started a few decades ago, used the concept of classes.
  - This architecture is called classful addressing.
  - In the mid-1990s, a new architecture, called classless addressing,was introduced that supersedes the original architecture.
  - In this section, we introduce classful addressing because it paves the way for understanding classless addressing and justifies the rationale for moving to the new architecture.
  - Classless addressing is discussed in the next section.
  - **Classes**
  - **In classful addressing, the IP address space is divided into five classes: A, B, C, D,and E.**
  - Each class occupies some part of the whole address space. Figure 5.5 shows the class occupation of the address space

Figure 5.5 *Occupation of the address space*



Class A: 2^{31} = 2,147,483,648 addresses, 50%

Class B: 2^{30} = 1,073,741,824 addresses, 25%

Class C: 2^{29} = 536,870,912 addresses, 12.5%

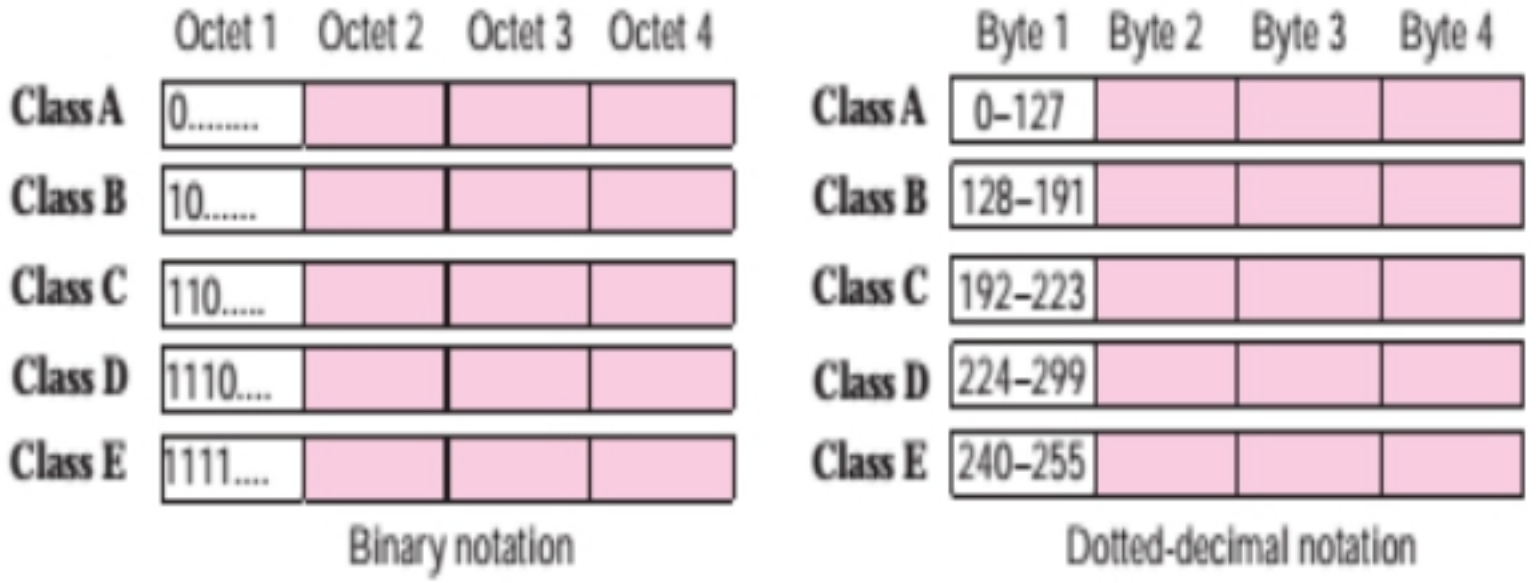
Class D: 2^{28} = 268,435,456 addresses, 6.25%

Class E: 2^{28} = 268,435,456 addresses, 6.25%

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

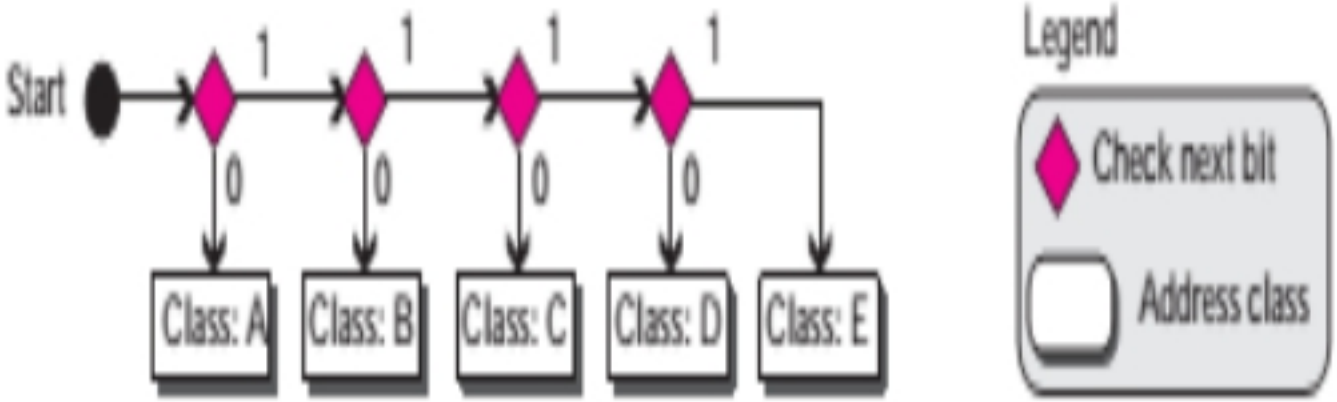
- **Recognizing Classes**
  - We can find the class of an address when the address is given either in binary or dotteddecimal notation.
  - In the binary notation, the first few bits can immediately tell us the class of the address; in the dotted-decimal notation, the value of the first byte can give the class of an address (Figure 5.6).

Figure 5.6 *Finding the class of an address*



- Note that some special addresses fall in class A or E.
- Computers often store IPv4 addresses in binary notation.
- In this case, it is very convenient to write an algorithm to use a continuous checking process for finding the address as shown in Figure 5.7.

Figure 5.7 *Finding the address class using continuous checking*

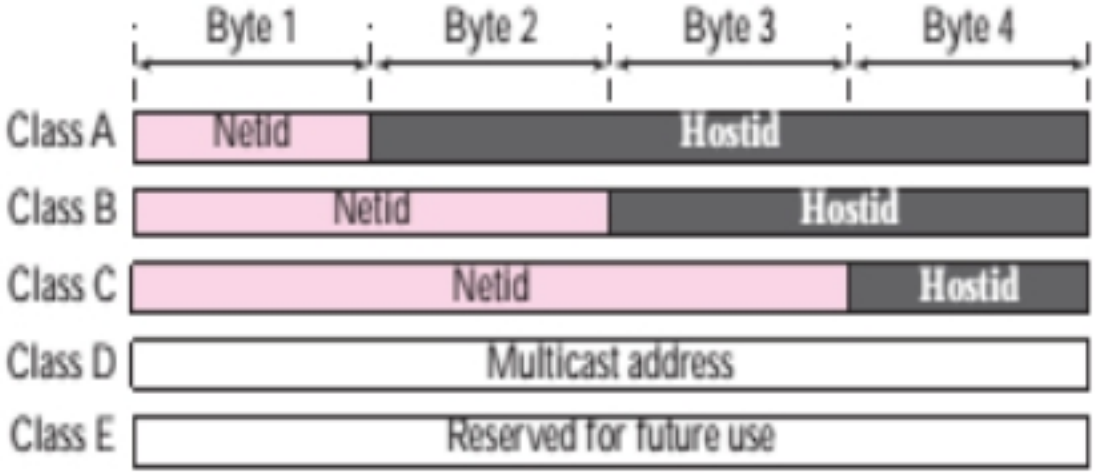


- **Example 5.10**
  - Find the class of each address:
  - a. 00000001 00001011 00001011 11101111
  - b. 11000001 10000011 00011011 11111111
  - c. 10100111 11011011 10001011 01101111
  - d.11110011 10011011 11111011 00001111
  - Solution
  - See the procedure in Figure 5.7.
  - a. The first bit is 0. 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 bit is 1; the second bit is 0. This is a class B address.
  - d.The first 4 bits are 1s. This is a class E address.

- **Example 5.11**
  - Find the class of each address:
  - a. 227.12.14.87
  - b. 193.14.56.22
  - c. 14.23.120.8
  - d.252.5.15.111
  - Solution
  - a. The first byte is 227 (between 224 and 239); the class is D.
  - b. The first byte is 193 (between 192 and 223); the class is C.
  - c. The first byte is 14 (between 0 and 127); the class is A.
  - d.The first byte is 252 (between 240 and 255); the class is E.

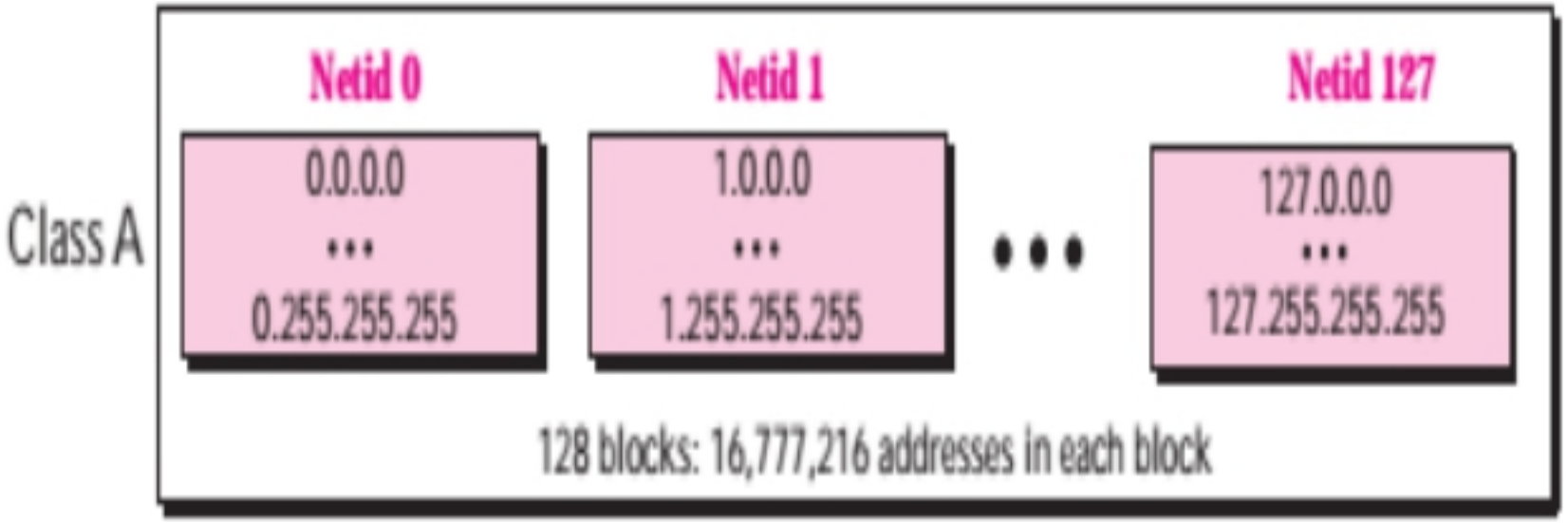
- **Netid and Hostid**
  - In classful addressing, an IP address in classes A, B, and C is divided into netidand hostid.
  - These parts are of varying lengths, depending on the class of the address.
  - Figure 5.8 shows the netid and hostid bytes.
  - Note that classes D and E are not divided into netid and hostid.

Figure 5.8 *Netid and hostid*



- **Classes and Blocks**
  - One problem with classful addressing is that each class is divided into a fixed number of blocks with each block having a fixed size. Let us look at each class.
  - **Class A**
  - Since only 1 byte in class A defines the netid and the leftmost bit should be 0, the next 7 bits can be changed to find the number of blocks in this class.
  - Therefore, class A is divided into $2^7 = 128$ blocks that can be assigned to 128 organizations (the number is less because some blocks were reserved as special blocks).
  - However, each block in this class contains 16,777,216 addresses, which means the organization should be a really large one to use all these addresses.
  - Many addresses are wasted in this class.
  - Figure 5.9 shows the block in class A.

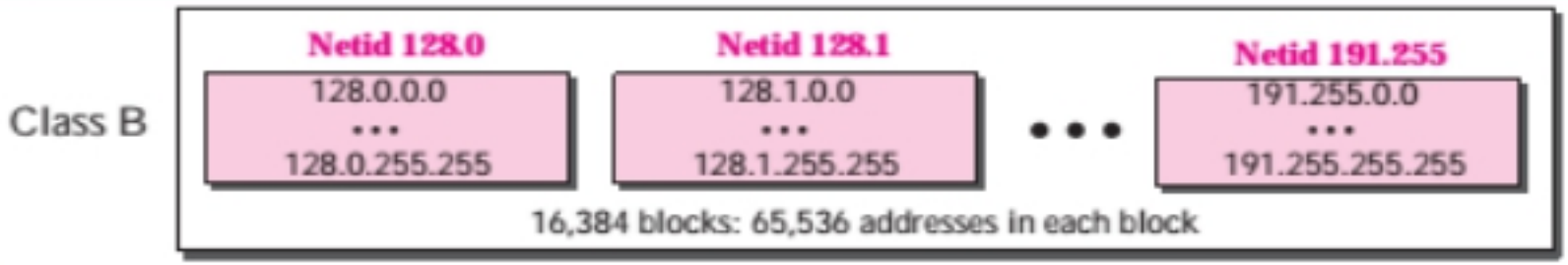
Figure 5.9 *Blocks in class A*



Millions of class A addresses are wasted.

- **Class B**
  - Since 2 bytes in class B define the class and the two leftmost bit should be 10 (fixed), the next 14 bits can be changed to find the number of blocks in this class.
  - Therefore, class B is divided into 2^{14}= 16,384 blocks that can be assigned to 16,384 organizations (the number is less because some blocks were reserved as special blocks).
  - However, each block in this class contains 65,536 addresses.
  - Not so many organizations can use so many addresses.
  - Many addresses are wasted in this class. Figure 5.10 shows the blocks in class B.

Figure 5.10 *Blocks in class B*



Many class B addresses are wasted.

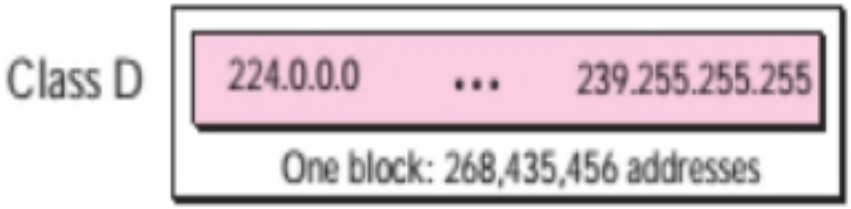
- **Class C**
  - Since 3 bytes in class C define the class and the three leftmost bits should be 110 (fixed), the next 21 bits can be changed to find the number of blocks in this class.
  - Therefore, class C is divided into 2^{21} =2,097,152 blocks, in which each block contains 256 addresses, that can be assigned to 2,097,152 organizations (the number is less because some blocks were reserved as special blocks).
  - Each block contains 256 addresses.
  - However, not so many organizations were so small as to be satisfied with a class C block. Figure 5.11 shows the blocks in class C.

Figure 5.11 *Blocks in class C*

Not so many organizations are so small to have a class C block.

- **Class D**
  - There is just one block of class D addresses.
  - It is designed for multicasting.
  - When a group is assigned an address in this class, every host that is a member of this group will have a multicast address in addition to its normal (unicast) address. Figure 5.12 shows the block.

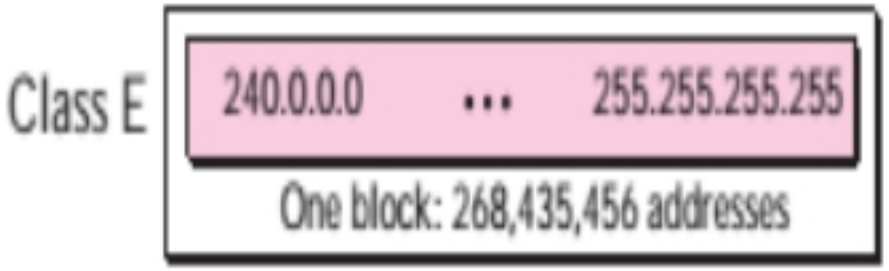
Figure 5.12 *The single block in Class D*



Class D addresses are made of one block, used for multicasting.

- **Class E**
  - There is just one block of class E addresses.
  - It was designed for use as reserved addresses, as shown in Figure 5.13

Figure 5.13 *The single block in Class E*



The only block of class E addresses was reserved for future purposes.

- **Mask**
  - Although the length of the netid and hostid (in bits) is predetermined in classful addressing,
  - we can also use a mask (also called the default mask), a 32-bit number made of contiguous 1s followed by contiguous 0s.
  - The masks for classes A, B, and C are shown in Table 19.2.
  - The concept does not apply to classes D and E.

Table 19.2 *Default masks for classful addressing*

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	11111111 00000000 00000000 00000000	255.0.0.0	18
B	11111111 11111111 00000000 00000000	255.255.0.0	116
C	11111111 11111111 11111111 00000000	255.255.255.0	124

- The mask can help us to find the netid and the hostid.
- For example, the mask for a class A address has eight 1s, which means the first 8 bits of any address in class A define the netid; the next 24 bits define the hostid.
- The last column of Table 19.2 shows the mask in the form \n where n can be 8, 16, or 24 in classful addressing.
- **This notation is also called slash notation or Classless Interdomain Routing (CIDR) notation.**
- The notation is used in classless addressing, which we will discuss later.
- We introduce it here because it can also be applied to classful addressing.
- We will show later that classful addressing is a special case of classless addressing.

- **Subnetting**
  - During the era of classful addressing, subnetting was introduced.
  - If an organization was granted a large block in class A or B, it could divide the addresses into several contiguous groups and assign each group to smaller networks (called subnets) or, in rare cases, share part of the addresses with neighbors.
  - **Subnetting increases the number of 1s in the mask.**

- **Supernetting**
  - The time came when most of the class A and class B addresses were depleted; however, there was still a huge demand for midsize blocks.
  - The size of a class C block with a maximum number of 256 addresses did not satisfy the needs of most organizations.
  - Even a midsize organization needed more addresses.
  - One solution was supernetting.
  - In supernetting, an organization can combine several class C blocks to create a larger range of addresses.
  - In other words, several networks are combined to create a supernetwork or a supenet.

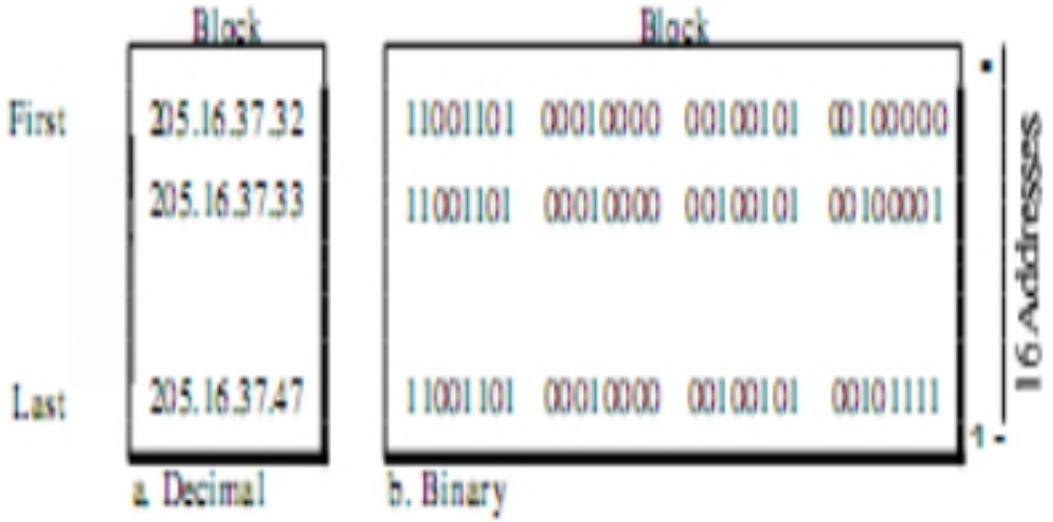
- **Address Depletion**
  - The flaws in classful addressing scheme combined with the fast growth of the Internet led to the near depletion of the available addresses.
  - Yet the number of devices on the Internet is much less than the 2^{32} address space.
  - We have run out of class A and B addresses, and a class C block is too small for most midsize organizations.
  - One solution that has alleviated the problem is the idea of classless addressing.
  - Classful addressing, which is almost obsolete, is replaced with classless addressing.

- **Classless Addressing**
  - To overcome address depletion and give more organizations access to the Internet, classless addressing was designed and implemented.
  - In this scheme, there are no classes, but the addresses are still granted in blocks.
- **Address Blocks**
  - In classless addressing, when an entity, small or large, needs to be connected to the Internet, it is granted a block (range) of addresses.
  - The size of the block (the number of addresses) varies based on the nature and size of the entity.
  - For example, a household may be given only two addresses; a large organization may be given thousands of addresses.
  - An ISP, as the Internet service provider, may be given thousands or hundreds of thousands based on the number of customers it may serve.

- **Restriction:**
  - To simplify the handling of addresses, the Internet authorities impose three restrictions on classless address blocks:
  - 1. The addresses in a block must be contiguous, one after another.
  - 2. The number of addresses in a block must be a power of 2 (1, 2, 4, 8, ...).
  - 3. The first address must be evenly divisible by the number of addresses.

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

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



- 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.

- **Mask**
  - A better way to define a block of addresses is to select any address in the block and the mask.
  - As we discussed before, a mask is a 32-bit number in which the n leftmost bits are 1s and the 32 - n rightmost bits are Os.
  - However, in classless addressing the mask for a block can take any value from 0 to 32.
  - It is very convenient to give just the value of n preceded by a slash (CIDR notation).

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

x.y.z.tln

in which x.y.z.t defines one of the addresses and the *In* defines the mask.

- The address and the /n notation completely define the whole block (the first address, the last address, and the number of addresses).

- **First Address**
  - The first address in the block can be found by setting the 32 - n right-most bits in the binary notation of the address to Os.
  - The first address in the block can be found by setting the rightmost 32 - n bits to Os.
- **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 00100 I 11.
  - If we set 32 - 28 rightmost bits to 0, we get 11001101 000100000100101 0010000 or 205.16.37.32.
  - This is actually the block shown in Figure 19.3.
- **Last Address**
  - The last address in the block can be found by setting the 32 - n right- most bits in the binary notation of the address 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 00010000001001010010011 1.
  - If we set 32 - 28 rightmost bits to 1, we get 11001101 00010000 00100101001 0 1111 or 205.16.37.47.
  - This is actually the block shown in Figure 19.3.
- **Number of Addresses**
  - The number of addresses in the block is the difference between the last and first address.
  - It can easily be found using the formula 2^{32-n}.
  - The number of addresses in the block can be found by using the formula 2^{32-n}.
  - 2^{32-28}
  - $2^4=16$

- **Numerical Problems**
- 1. Find the netid and the hostid of the following IP addresses:
  - a. 114.34.2.8
  - b. 132.56.8.6
  - c. 208.34.54.12
  - d. 251.34.98.5
- 2. Find the range of addresses in the following blocks:
  - a. 123.56.77.32/29
  - b. 200.17.21.128/27
  - c. 17.34.16.0/23
  - d. 180.34.64.64/30
- 3. In a block of addresses, we know the IP address of one host is 25.34.12.56/16. What is the first address (network address) and the last address in this block?
- 4. In a block of addresses, we know the IP address of one host is 182.44.82.16/26. What is the first address (network address) and the last address in this block?

- 5. A block of addresses is granted to a small organization.
  - We know that one of the addresses is 40.16.37.39. What is the first address, last address and number of addresses in the block?

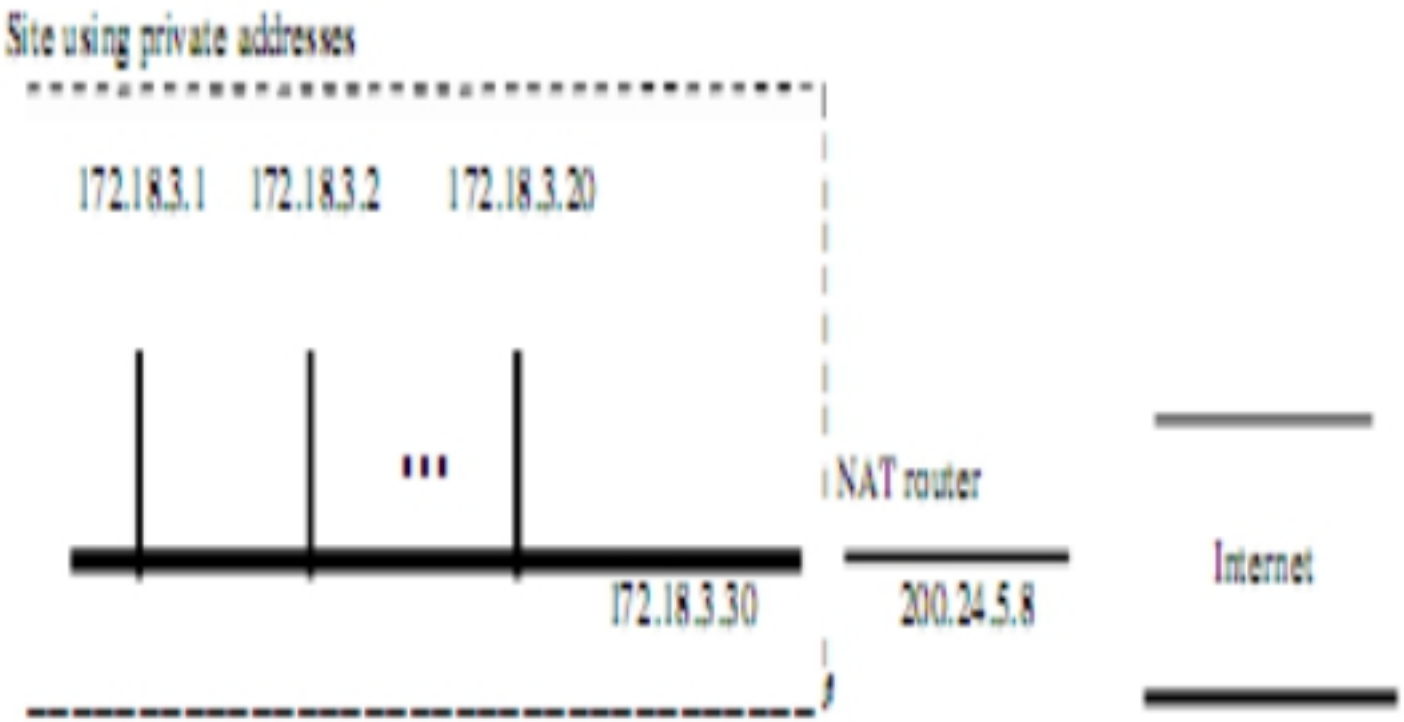
- **Network Address Translation (NAT)**
  - The number of home users and small businesses that want to use the Internet is ever increasing.
  - In the beginning, a user was connected to the Internet with a dial-up line, which means that she was connected for a specific period of time.
  - An ISP with a block of addresses could dynamically assign an address to this user.
  - With the shortage of addresses, this is a serious problem.
  - A quick solution to this problem is called network address translation (NAT).
  - NAT enables a user to have a large set of addresses internally and one address, or a small set of addresses, externally.
  - The traffic inside can use the large set; the traffic outside, the small set.
  - To separate the addresses used inside the home or business and the ones used for the Internet, the Internet authorities have reserved three sets of addresses as private addresses, shown in Table 19.3.

Table 19.3 *Addresses for private networks*

Range			*Total*
10.0.0.0	to	10.255.255.255	2^{24}
172.16.0.0	to	172.31.255.255	2^{20}
192.168.0.0	to	192.168.255.255	2^{16}

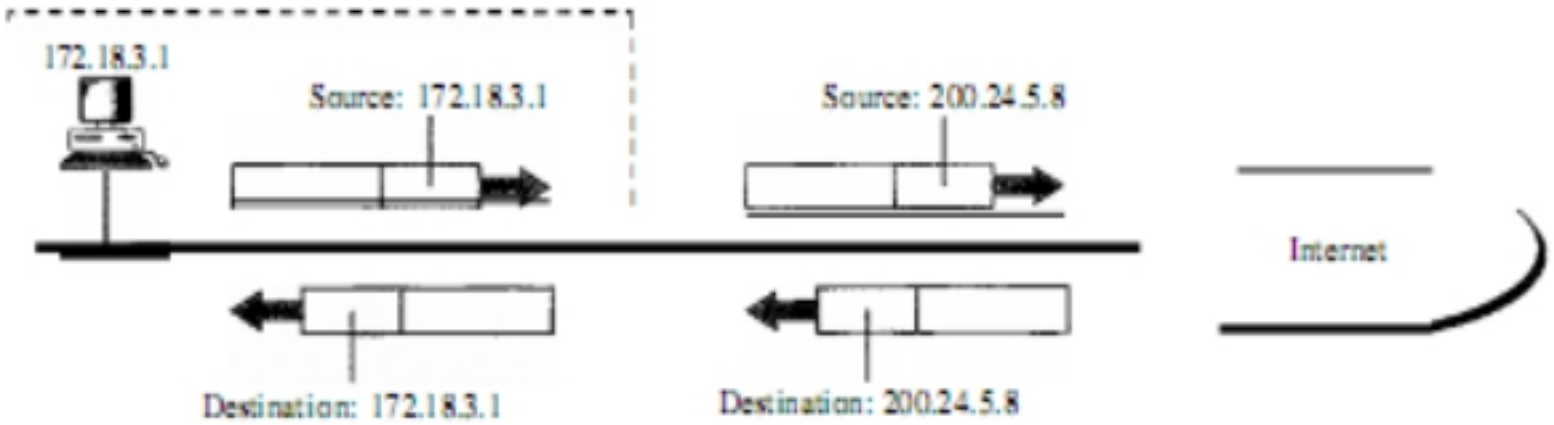
- Any organization can use an address out of this set without permission from the Internet authorities.
- Everyone knows that these reserved addresses are for private net-works.
- They are unique inside the organization, but they are not unique globally.
- No router will forward a packet that has one of these addresses as the destination address.
- The site must have only one single connection to the global Internet through a router that runs the NAT software.
- Figure 19.10 shows a simple implementation of NAT.
- As Figure 19.10 shows, the private network uses private addresses.
- The router that connects the network to the global address uses one private address and one global address.
- The private network is transparent to the rest of the Internet; the rest of the Internet sees only the NAT router with the address 200.24.5.8.

Figure 19.10 *A NAT implementation*



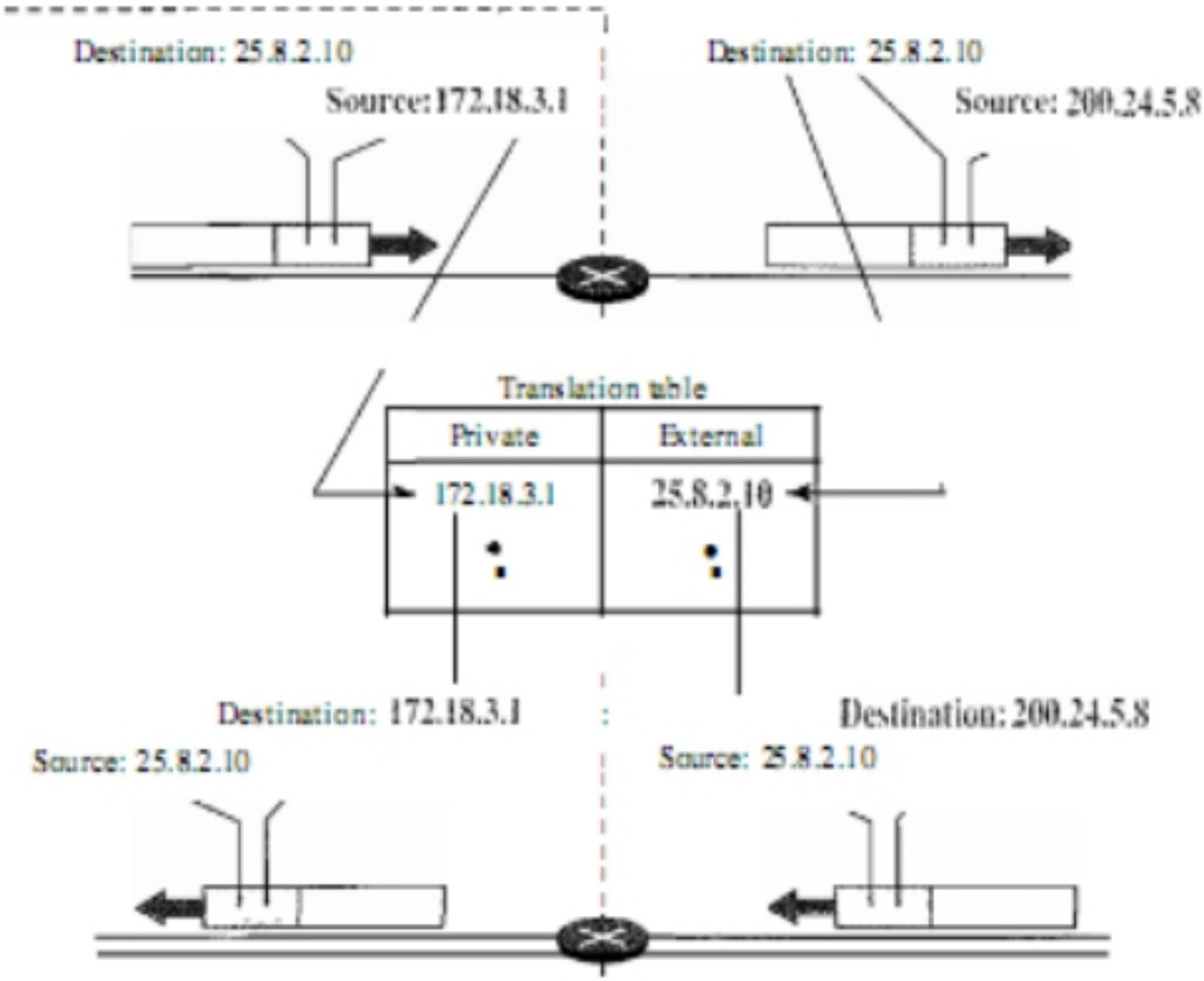
- **Address Translation**
  - All the outgoing packets go through the NAT router, which replaces the source address in the packet with the global NAT address.
  - All incoming packets also pass through the NAT router, which replaces the destination address in the packet (the NAT router global address) with the appropriate private address.
  - Figure 19.11 shows an example of address translation.

Figure 19.11 *Addresses in a NAT*



- **Translation Table**
  - The reader may have noticed that translating the source addresses for outgoing packets is straightforward.
  - But how does the NAT router know the destination address for a packet coming from the Internet?
  - There may be tens or hundreds of private IP addresses, each belonging to one specific host.
  - The problem is solved if the NAT router has a translation table.
- **Using One IP Address**
  - In its simplest form, a translation table has only two columns: the private' address and the external address (destination address of the packet).
  - When the router translates the source address of the outgoing packet, it also makes note of the destination address-where the packet is going.
  - When the response comes back from the destination, the router uses the source address of the packet (as the external address) to find the private address of the packet.
  - Figure 19.12 shows the idea.

Figure 19.12 *NAT address translation*



- In this strategy, communication must always be initiated by the private network.
- The NAT mechanism described requires that the private network start the communication.
- As we will see, NAT is used mostly by ISPs which assign one single address to a customer.
- The customer, however, may be a member of a private network that has many private addresses.
- In this case, communication with the Internet is always initiated from the customer site, using a client program such as HTTP, TELNET, or FTP to access the corresponding server program.
- For example, when e-mail that originates from a non- customer site is received by the ISP e-mail server, the e-mail is stored in the mailbox of the.customer until retrieved.
- A private network cannot run a server program for clients outside of its network if it is using NAT technology.

- **Using a Pool of IP Addresses**
  - Since the NAT router has only one global address, only one private network host can access the same external host.
  - To remove this restriction, the NAT router uses a pool of global addresses.
  - For example, instead of using only one global address (200.24.5.8), the NAT router can use four addresses (200.24.5.8, 200.24.5.9,200.24.5.10, and 200.24.5.11).
  - In this case, four private network hosts can communicate with the same external host at the same time because each pair of addresses defines a connection.
  - However, there are still some drawbacks.
  - In this example, no more than four connections can be made to the same destination.
  - Also, no private-network host can access two external server programs (e.g., HTTP and FfP) at the same time.

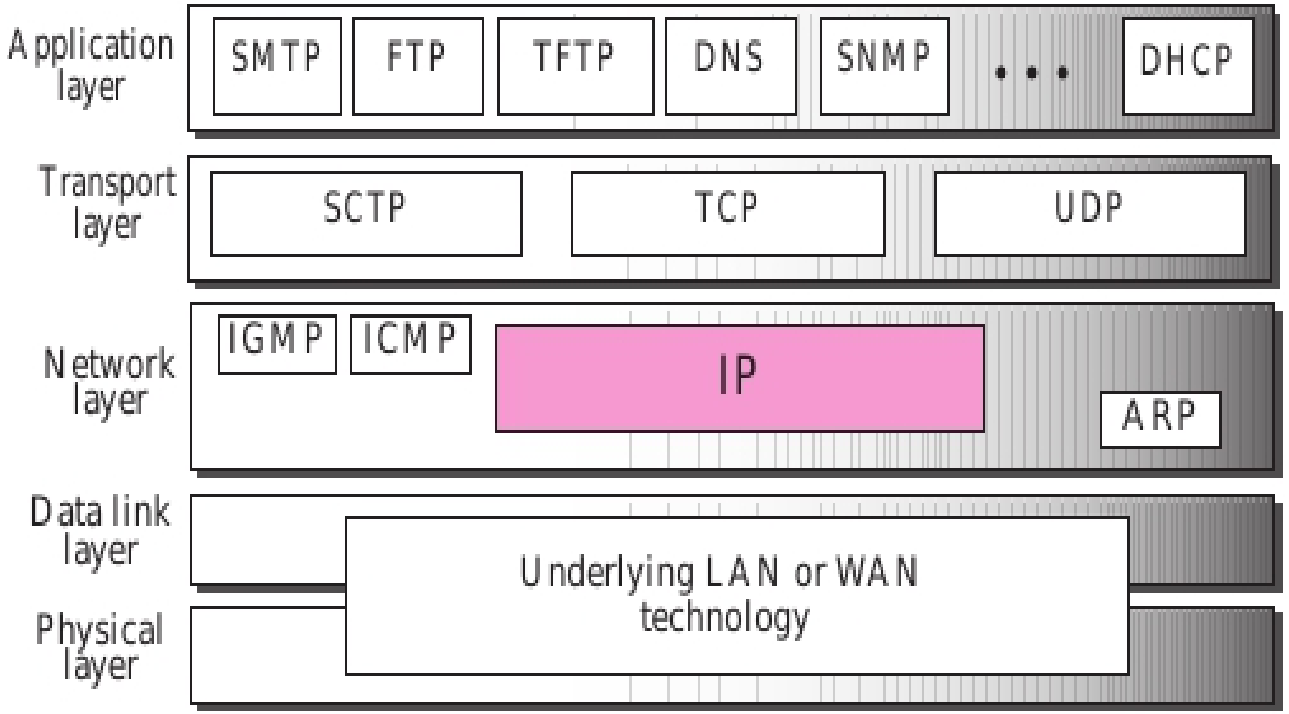
- **Using Both IP Addresses and Port Numbers**
  - To allow a many-to-many relationship between private-network hosts and external server programs, we need more information in the translation table.
  - For example, suppose two hosts with addresses 172.18.3.1 and 172.18.3.2 inside a private network need to access the HTTP server on external host 25.8.3.2.
  - If the translation table has five columns, instead of two, that include the source and destination port numbers of the transport layer protocol, the ambiguity is eliminated.

Table 19.4 *Five-column translation table*

Private Address	*Private Port*	*External Address*	*External Port*	*Transport Protocol*
172.18.3.1	1400	25.8.3.2	80	TCP
172.18.3.2	1401	25.8.3.2	80	TCP
. . .	. . .	. . .	. . .	. . .

- **Internet Protocol**
- The Internet Protocol (IP) is the transmission mechanism used by the TCP/IP protocols at the network layer. Figure 7.1 shows the position of IP in the suite.

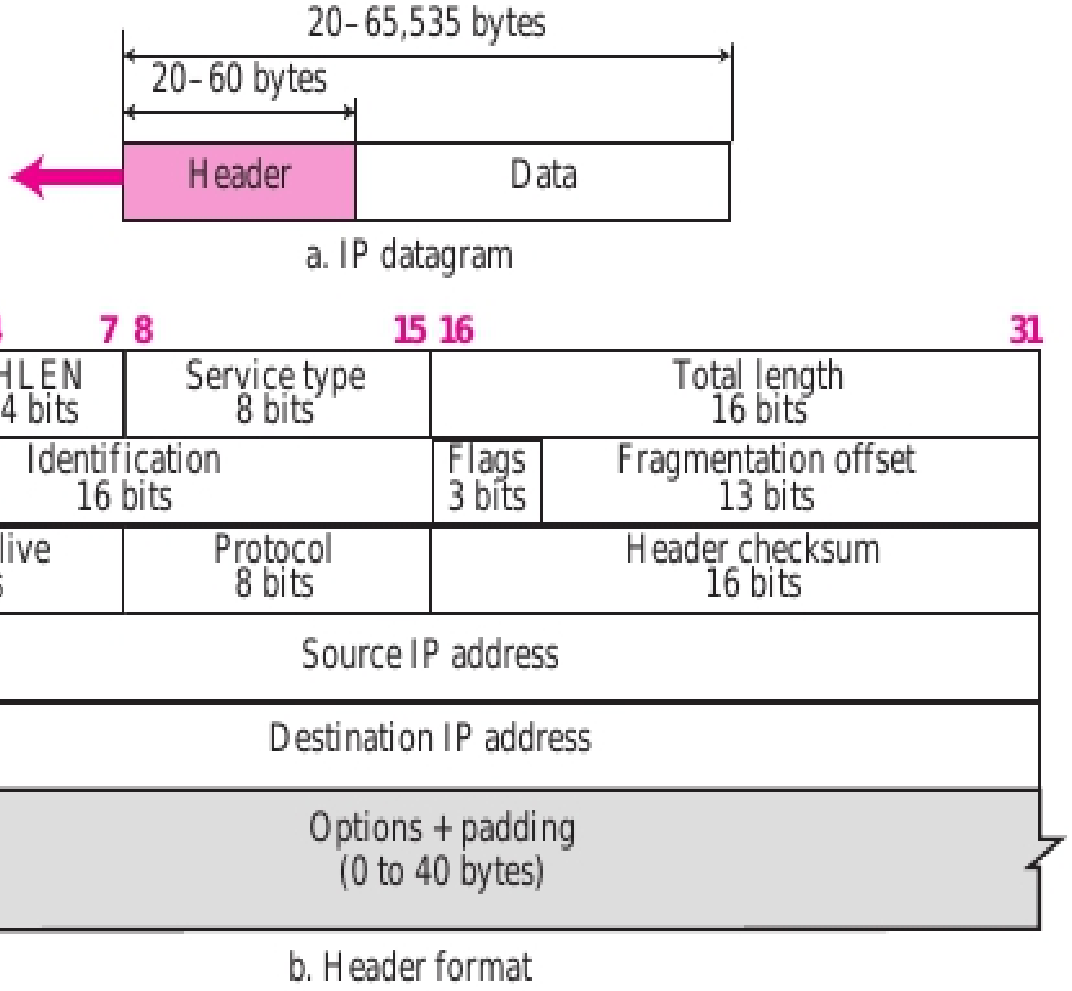
Figure 7.1 *Position of IP in TCP/IP protocol suite*



- IP is an unreliable and connectionless datagram protocol—a best-effort deliverybservice.
- The term best-effort means that IP packets can be corrupted, lost, arrive out of order, or delayed and may create congestion for the network.
- If reliability is important, IP must be paired with a reliable protocol such as TCP
- An example of a more commonly understood best-effort delivery service is the post office.
- The post office does its best to deliver the mail but does not always succeed.
- If an unregistered letter is lost, it is up to the sender or would-be recipient to discover the loss and rectify the problem.
- The post office itself does not keep track of every letter and cannot notify a sender of loss or damage.
- IP is also a connectionless protocol for a packet switching network that uses the datagram approach .
- This means that each datagram is handled independently, and each datagram can follow a different route to the destination.
- This implies that datagrams sent by the same source to the same destination could arrive out of order.
- Also, some could be lost or corrupted during transmission.
- Again, IP relies on a higher-level protocol to take care of all these problems.

- **DATAGRAMS**
  - Packets in the network (internet) layer are called datagrams.
  - Figure 7.2 shows the IP datagram format.
  - A datagram is a variable-length packet consisting of two parts: header and data.
  - The header is 20 to 60 bytes in length and contains information essential to routing and delivery.
  - It is customary in TCP/IP to show the header in 4-byte sections.
  - A brief description of each field is in order.

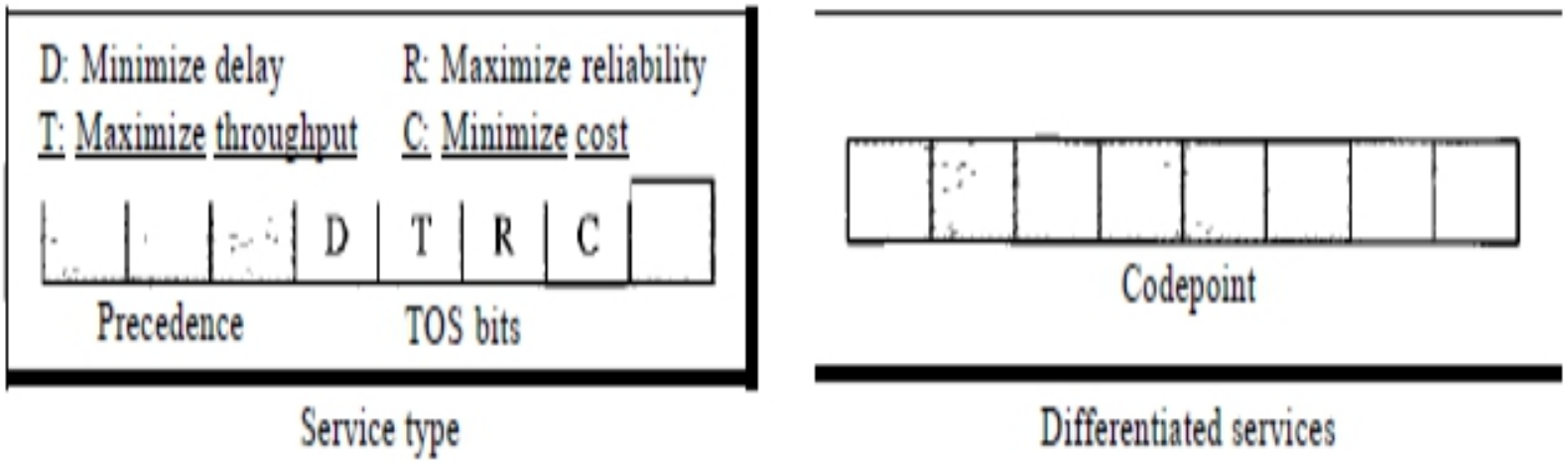
Figure 7.2 *IP datagram*



- **Version (VER).**
  - This 4-bit field defines the version of the IP protocol.
  - Currently the version is 4.
  - However, version 6 (or IPv6) may totally replace version 4 in the future.
  - This field tells the IP software running in the processing machine that the datagram has the format of version 4.
  - All fields must be interpreted as specified in the fourth version of the protocol.
  - If the machine is using some other version of IP, the datagram is discarded rather than interpreted incorrectly.
- **Header length (HLEN)**.
  - This 4-bit field defines the total length of the datagram header in 4-byte words.
  - This field is needed because the length of the header is variable (between 20 and 60 bytes).
  - When there are no options, the header length is 20 bytes, and the value of this field is 5 ($5 \times 4 = 20$).
  - When the option field is at its maximum size, the value of this field is 15 ($15 \times 4 = 60$).

- **Service Type**
  - IETF has changed the interpretation and name of this 8–bit field.
  - This field, previously called service type, is now called differentiated services.
  - We show both interpretations in Figure 20.6.

Figure 20.6 *Service type or differentiated services*



- Service Type
  - In this interpretation, the first 3 bits are called precedence bits.
  - The next 4 bits are called type of service (TOS) bits, and the last bit is not used.
  - **Precedence** is a 3–bit subfield ranging from 0 (000 in binary) to 7 (111 in binary).
  - The precedence defines the priority of the datagram in issues such as congestion.
  - If a router is congested and needs to discard some datagrams, those datagrams with lowest precedence are discarded first.
  - Some datagrams in the Internet are more important than others.
  - For example, a datagram used for network management is much more urgent and important than a datagram containing optional information for a group.
  - **TOS bits** is a 4–bit subfield with each bit having a special meaning.
  - Although a bit can be either 0 or 1, one and only one of the bits can have the value of 1 in each datagram.
  - The bit patterns and their interpretations are given in Table 20.1.
  - With only 1 bit set at a time, we can have five different types of services.

Table 20.1 *Types of service*

TOS Bits	*Description*
0000	Normal (default)
0001	Minimize cost
0010	Maximize reliability
0100	Maximize throughput
1000	Minimize delay

Table 20.2 *Default types of service*

Protocol	*TOS Bits*	*Description*
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

- **Differentiated Services**
  - In this interpretation, the first 6 bits make up the codepoint subfield, and the last 2 bits are not used.
  - The codepoint subfield can be used in two different ways.
    - **a.** When the 3 rightmost bits are Os, the 3 leftmost bits are interpreted the same as the precedence bits in the service type interpretation.
    - In other words, it is compatible with the old interpretation.
    - **b.** When the 3 rightmost bits are not all Os, the 6 bits define 64 services based on the priority assignment by the Internet or local authorities according to Table 20.3.
    - The first category contains 32 service types; the second and the third each contain 16.
    - The first category (numbers 0, 2,4, ... ,62) is assigned by the Internet authorities (IETF).
    - The second category (3, 7, 11, 15, , 63) can be used by local authorities (organizations).
    - The third category (1, 5, 9, ,61) is temporary and can be used for experimental purposes.

Table 20.3 *Values for codepoints*

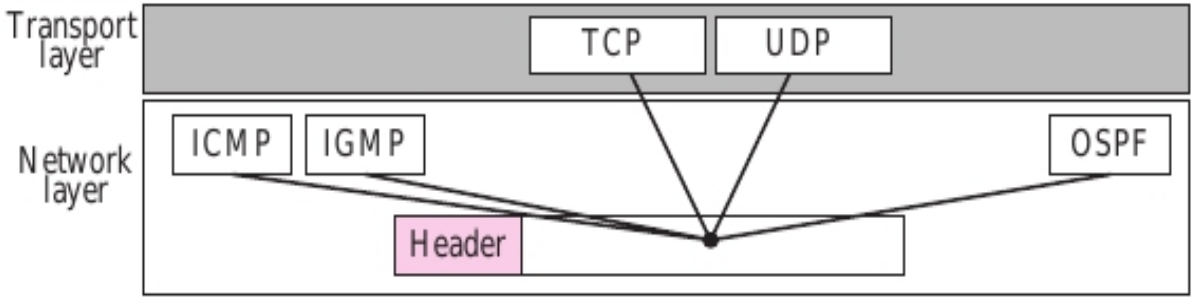
Category	*Codepoint*	*Assigning Authority*
1	XXXXXO	Internet
2	XXXXl1	Local
3	XXXXOI	Temporary or experimental

- **Total length**.
  - This is a 16-bit field that defines the total length (header plus data) of the IP datagram in bytes.
  - To find the length of the data coming from the upper layer, subtract the header length from the total length.
  - The header length can be found by multiplying the value in the HLEN field by four.
  - Length of data = total length – header length
  - Since the field length is 16 bits, the total length of the IP datagram is limited to 65,535 ($2^{16} - 1$) bytes, of which 20 to 60 bytes are the header and the rest is data from the upper layer.
- **Identification.** This field is used in fragmentation (discussed in the next section).
- **Flags.** This field is used in fragmentation (discussed in the next section).
- **Fragmentation offset**. This field is used in fragmentation (discussed in the next section).

- **Time to live.**
  - A datagram has a limited lifetime in its travel through an internet.
  - This field was originally designed to hold a timestamp, which was decremented by each visited router.
  - The datagram was discarded when the value became zero.
  - However, for this scheme, all the machines must have synchronized clocks and must know how long it takes for a datagram to go from one machine to another.
  - Today, this field is mostly used to control the maximum number of hops (routers) visited by the data-gram.
  - When a source host sends the datagram, it stores a number in this field.
  - This value is approximately two times the maximum number of routes between any two hosts.
  - Each router that processes the datagram decrements this number by one.
  - If this value, after being decremented, is zero, the router discards the datagram.

- **Protocol.**
  - This 8-bit field defines the higher-level protocol that uses the services of the IP layer.
  - An IP datagram can encapsulate data from several higher level protocols such as TCP, UDP, ICMP, and IGMP.
  - This field specifies the final destination protocol to which the IP datagram should be delivered.
  - In other words, since the IP protocol multiplexes and demultiplexes data from different higher-level protocols, the value of this field helps in the demultiplexing process when the datagram arrives at its final destination .

Figure 7.5 *Multiplexing*



Some of the value of this field for different higher-level protocols is shown in Table 7.2.

Table 7.2 *Protocols*

Value	*Protocol*	*Value*	*Protocol*
1	ICMP	17	UDP
2	IGMP	89	OSPF
6	TCP		

- **Checksum.**
  - The checksum concept and its calculation are discussed later
- **Source address.**
  - This 32-bit field defines the IP address of the source.
  - This field must remain unchanged during the time the IP datagram travels from the source host to the destination host.
- **Destination address.**
  - This 32-bit field defines the IP address of the destination.
  - This field must remain unchanged during the time the IP datagram travels from the source host to the destination host.

- **Example 7.1**
  - An IP 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 left-most bits (0100) show the version, which is correct.
  - The next 4 bits (0010) show the wrong header length ($2 \times 4 = 8$).
  - The minimum number of bytes in the header must be 20.
  - The packet has been corrupted in transmission.

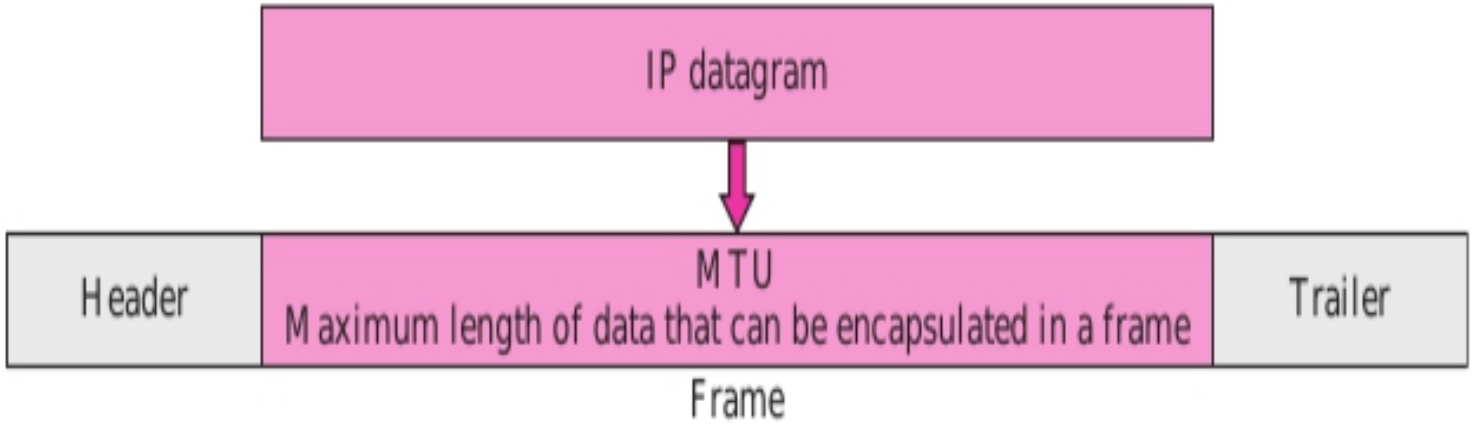
- **Example 7.2**
  - In an IP packet, the value of HLEN is 1000 in binary. How many bytes of options are being carried by this packet?
- **Solution**
  - The HLEN value is 8, which means the total number of bytes in the header is 8 × 4 or 32 bytes.
  - The first 20 bytes are the base header, the next 12 bytes are the options.
- **Example 7.3**
  - In an IP packet, the value of HLEN is 516 and the value of the total length field is 0028 16. 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 7.4**
  - An IP packet has arrived with the first few hexadecimal digits as shown below:
  - 45000028000100000102 . . .
  - 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 (16 hexadecimal digits).
  - 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.

- **FRAGMENTATION**
  - A datagram can travel through different networks.
  - Each router decapsulates the IP datagram from the frame it receives, processes it, and then encapsulates it in another frame.
  - The format and size of the received frame depend on the protocol used by the physical network through which the frame has just traveled.
  - The format and size of the sent frame depend on the protocol used by the physical network through which the frame is going to travel.
  - For example, if a router connects a LAN to a WAN, it receives a frame in the LAN format and sends a frame in the WAN format.

- **Maximum Transfer Unit (MTU)**
  - Each data link layer protocol has its own frame format in most protocols.
  - One of the fields defined in the format is the maximum size of the data field.
  - In other words, when a datagram is encapsulated in a frame, the total size of the datagram must be less than this maximum size, which is defined by the restrictions imposed by the hardware and software used in the network .

Figure 7.6 *MTU*



- The value of the MTU differs from one physical network protocol to another.
- For example, the value for the Ethernet LAN is 1500 bytes, for FDDI LAN is 4352 bytes, and for PPP is 296 bytes.
- In order to make the IP protocol independent of the physical network, the designers decided to make the maximum length of the IP datagram equal to 65,535 bytes.
- This makes transmission more efficient if we use a protocol with an MTU of this size.
- How- ever, for other physical networks, we must divide the datagram to make it possible to pass through these networks.
- **This is called fragmentation**.

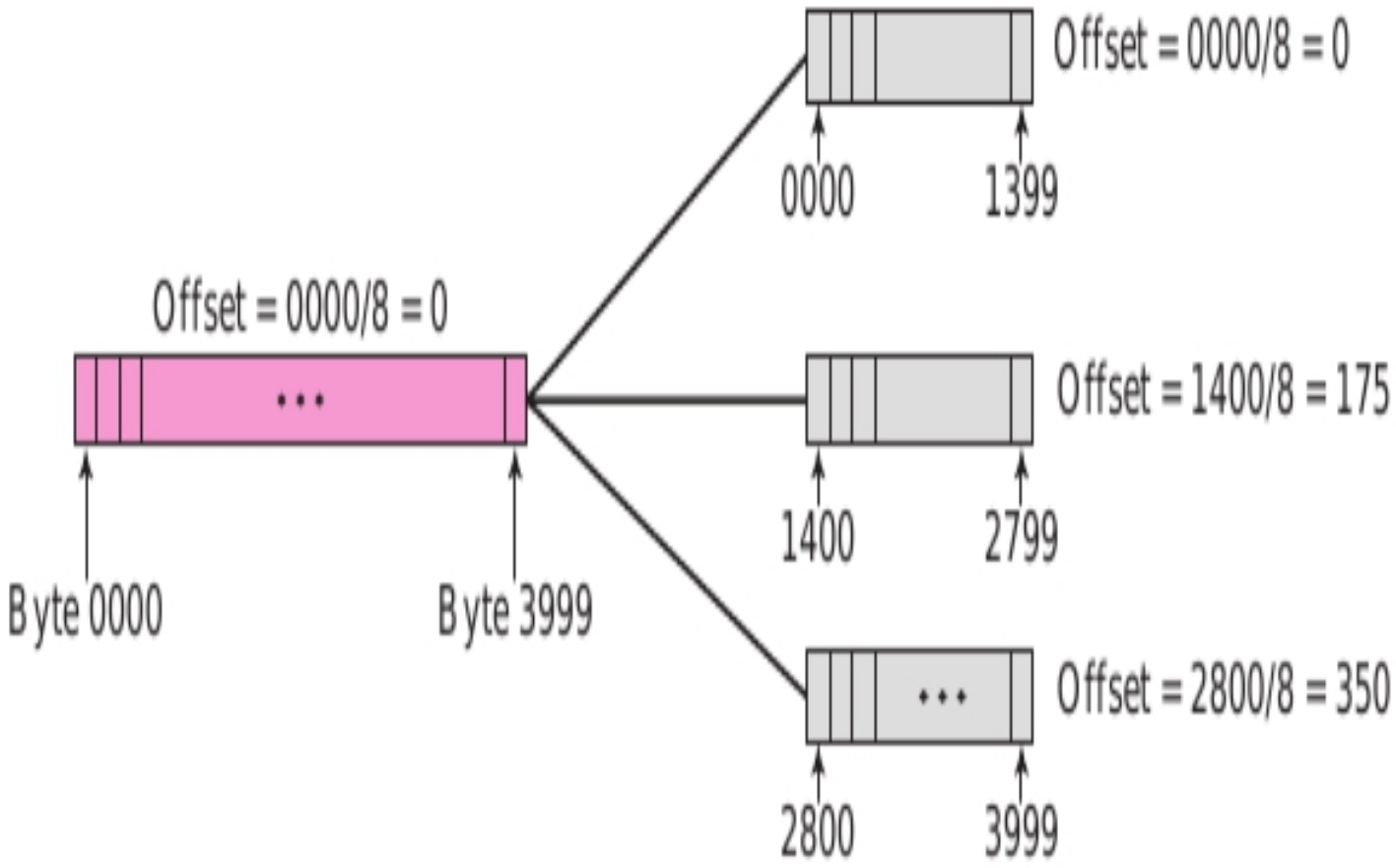
- **Fields Related to Fragmentation**
  - The fields that are related to fragmentation and reassembly of an IP datagram are the **identification, flags, and fragmentation offset fields.**
  - **Identification.**
  - This 16-bit field identifies a datagram originating from the source host.
  - The combination of the identification and source IP address must uniquely define a datagram as it leaves the source host.
  - To guarantee uniqueness, the IP protocol uses a counter to label the datagrams.
  - The counter is initialized to a positive number.
  - When the IP protocol sends a datagram, it copies the current value of the counter to the identification field and increments the counter by one.
  - As long as the counter is kept in the main memory, uniqueness is guaranteed.
  - When a datagram is fragmented, the value in the identification field is copied into all fragments.
  - In other words, all fragments have the same identification number, which is also the same as the original datagram.
  - The identification number helps the destination in reassembling the datagram.
  - It knows that all fragments having the same identification value should be assembled into one datagram.

- **Flags**.
- This is a three-bit field.
- The first bit is reserved (not used).
- The second bit is called the do not fragment bit.
- If its value is 1, the machine must not fragment the datagram.
- If it cannot pass the datagram through any available physical network, it discards the datagram and sends an ICMP error message to the source host.
- If its value is 0, the datagram can be fragmented if necessary.
- The third bit is called the more fragment bit.
- If its value is 1, it means the datagram is not the last fragment; there are more fragments after this one.
- If its value is 0, it means this is the last or only fragment (see Figure 7.7).

Figure 7.7 *Flags field*

- **Fragmentation offset.**
- This 13-bit field shows the relative position of this fragment with respect to the whole datagram.
- It is the offset of the data in the original datagram measured in units of 8 bytes.
- Figure 7.8 shows a datagram with a data size of 4000 bytes fragmented into three fragments.
- The bytes in the original datagram are numbered 0 to 3999.
- The first fragment carries bytes 0 to 1399.
- The offset for this datagram is 0/8 = 0.
- The second fragment carries bytes 1400 to 2799; the offset value for this fragment is 1400/8 = 175.
- Finally, the third fragment carries bytes 2800 to 3999.
- The offset value for this fragment is 2800/8 = 350.

Figure 7.8 *Fragmentation example*



- **Example 7.5**
  - 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 nonfragmented packet is considered the last fragment.
- **Example 7.6**
  - 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).

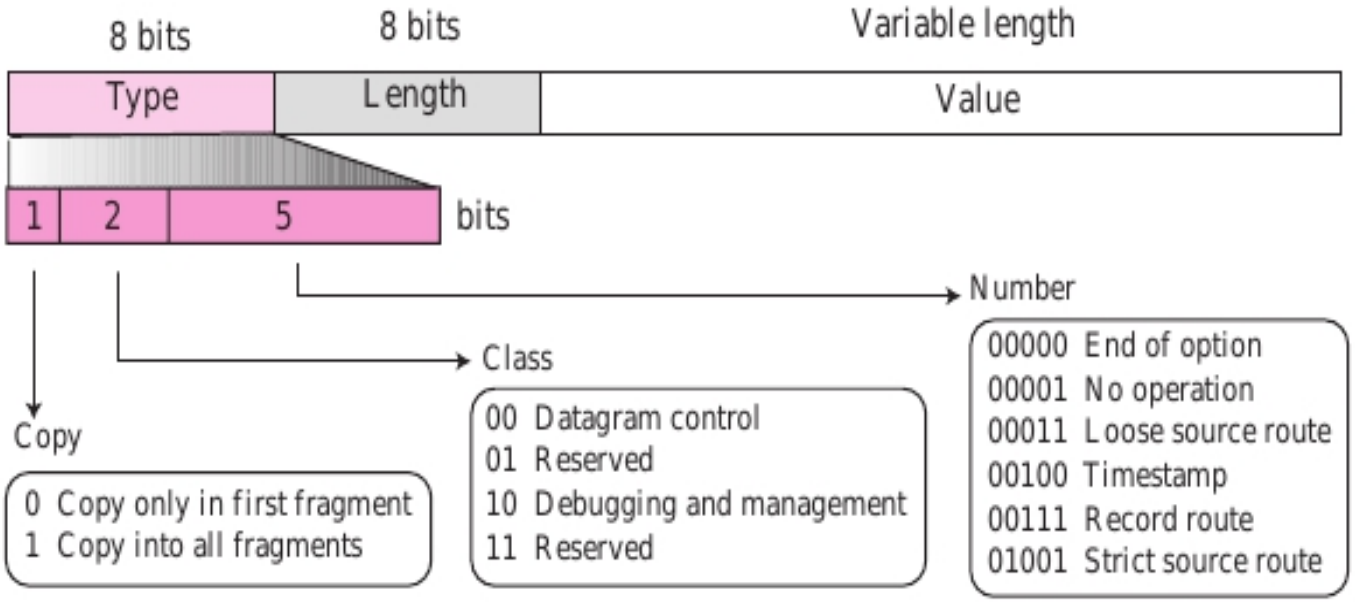
- **Example 7.7**
  - A packet has arrived with an M bit value of 1 and a fragmentation offset value of zero. Is this the first fragment, the last fragment, or a middle fragment?
  - **Solution**
  - Because the M bit is 1, it is either the first fragment or a middle one.
  - Because the offset value is 0, it is the first fragment.
- **Example 7.8**
  - 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 of the data.

- **Example 7.9**
  - 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 is the number 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.

- **OPTIONS**
  - The header of the IP datagram is made of two parts: a fixed part and a variable part.
  - The fixed part is 20 bytes long and was discussed in the previous section.
  - The variable part comprises the options, which can be a maximum of 40 bytes.
  - Options, as the name implies, are not required for a datagram.
  - They can be used for network testing and debugging.
  - Although options are not a required part of the IP header, option processing is required of the IP software.
  - This means that all implementations must be able to handle options if they are present in the header.

- **Format**
  - Figure 7.10 shows the format of an option.
  - It is composed of a 1-byte type field, a 1-byte length field, and a variable-sized value field.
  - The three fields are often referred to as type-length-value or TLV.

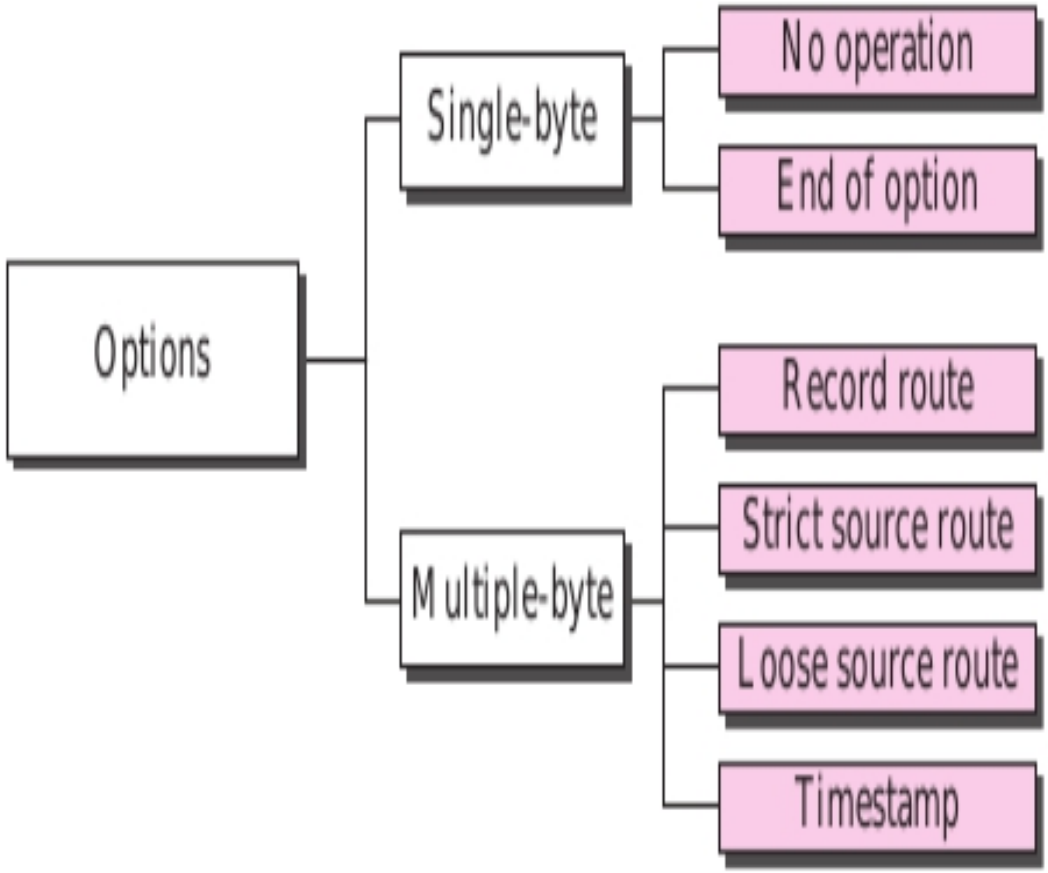
Figure 7.10 *Option format*



- **Type**
  - The type field is 8 bits long and contains three subfields: copy, class, and number.
  - **Copy.**
  - This 1-bit subfield controls the presence of the option in fragmentation.
  - When its value is 0, it means that the option must be copied only to the first frag- ment.
  - If its value is 1, it means the option must be copied to all fragments.
  - **Class**.
  - This 2-bit subfield defines the general purpose of the option.
  - When its value is 00, it means that the option is used for datagram control.
  - When its value is 10, it means that the option is used for debugging and management.
  - The other two possible values (01 and 11) have not yet been defined.
  - **Number.**
  - This 5-bit subfield defines the type of option.
  - Although 5 bits can define up to 32 different types, currently only 6 types are in use.

- **Length**
  - The length field defines the total length of the option including the type field and the length field itself.
  - This field is not present in all of the option types.
- **Value**
  - The value field contains the data that specific options require.
  - Like the length field, this field is also not present in all option types.
- **Option Types**
  - As mentioned previously, only six options are currently being used.
  - Two of these are 1-byte options, and they do not require the length or the data fields.
  - Four of them are multiple-byte options; they require the length and the data fields (see Figure 7.11).

Figure 7.11 *Categories of options*



- **No-Operation Option**
  - A no-operation option is a 1-byte option used as a filler between options.
  - For example, it can be used to align the next option on a 16-bit or 32-bit boundary.
- **End-of-Option Option**
  - An end-of-option option is also a 1-byte option used for padding at the end of the option field.
  - It, however, can only be used as the last option.
  - Only one end-of-option option can be used.
  - After this option, the receiver looks for the payload data.
  - This means that if more than 1 byte is needed to align the option field, some no-operation options must be used, followed by an end-of-option option .

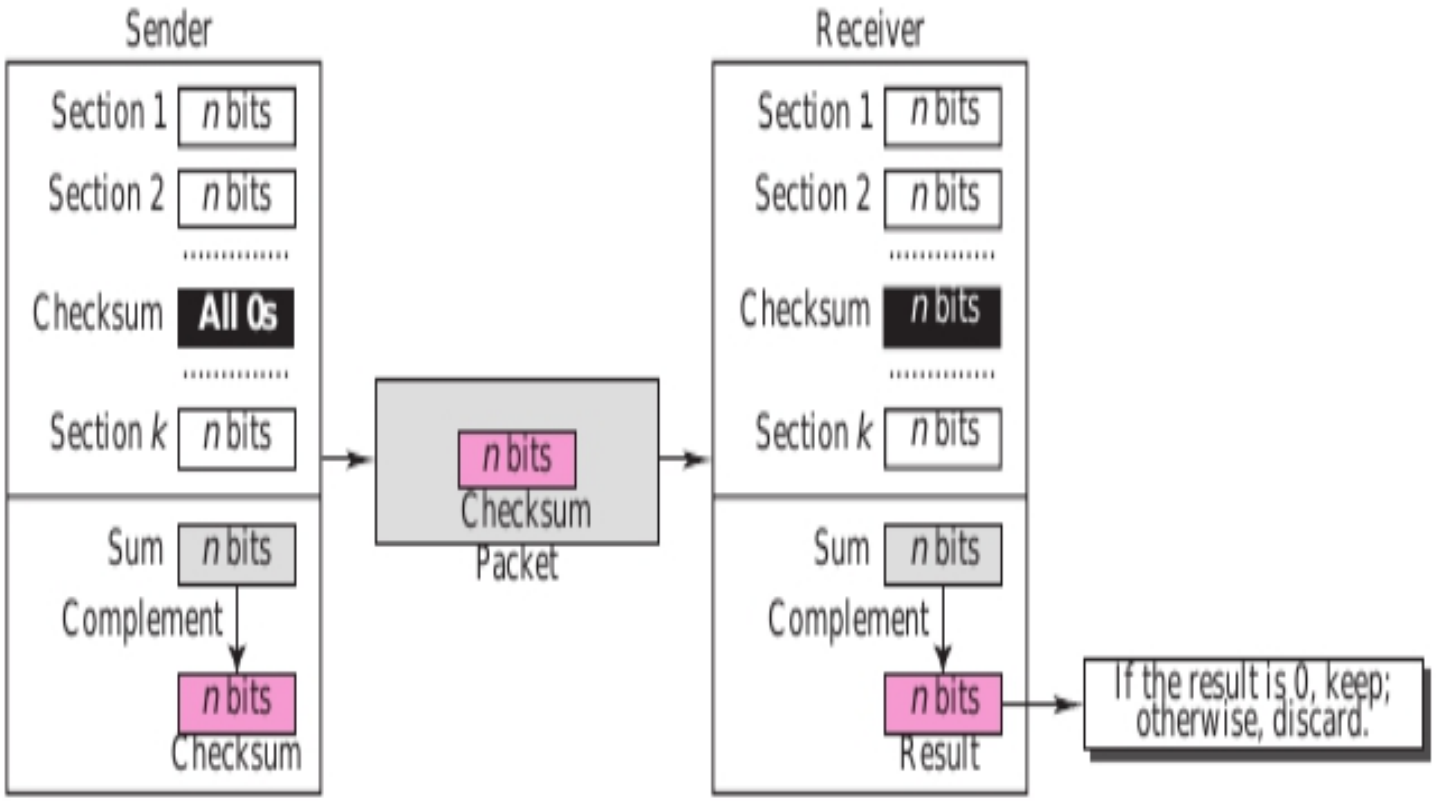
- **Record-Route Option**
  - A record-route option is used to record the Internet routers that handle the datagram.
  - It can list up to nine router IP addresses since the maximum size of the header is 60 bytes, which must include 20 bytes for the base header.
  - This implies that only 40 bytes are left over for the option part.
  - The source creates placeholder fields in the option to be filled by the visited routers.
- **Strict-Source-Route Option**
  - A strict-source-route option is used by the source to predetermine a route for the datagram as it travels through the Internet.
  - Dictation of a route by the source can be useful for several purposes.
  - The sender can choose a route with a specific type of service, such as minimum delay or maximum throughput.
  - Alternatively, it may choose a route that is safer or more reliable for the sender's purpose.
  - For example, a sender can choose a route so that its datagram does not travel through a competitor's network.

- **Loose-Source-Route Option**
  - A loose-source-route option is similar to the strict source route, but it is more relaxed.
  - Each router in the list must be visited, but the datagram can visit other routers as well.
- **Timestamp**
  - A timestamp option is used to record the time of datagram processing by a router.
  - The time is expressed in milliseconds from midnight, Universal Time.
  - Knowing the time a datagram is processed can help users and managers track the behavior of the routers in the Internet.
  - We can estimate the time it takes for a datagram to go from one router to another.
  - We say estimate because, although all routers may use Universal Time, their local clocks may not be synchronized.

- **CHECKSUM**
  - The error detection method used by most TCP/IP protocols is called the checksum.
  - The checksum protects against the corruption that may occur during the transmission of a packet.
  - It is redundant information added to the packet.
  - The checksum is calculated at the sender and the value obtained is sent with the packet.
  - The receiver repeats the same calculation on the whole packet including the checksum.
  - If the result is satisfactory (see below), the packet is accepted; otherwise, it is rejected.

- **Checksum Calculation at the Sender**
  - At the sender, the packet header is divided into n-bit sections (n is usually 16).
  - These sections are added together using one's complement arithmetic , resulting in a sum that is also n bits long.
  - The sum is then complemented (all 0s changed to 1s and all 1s to 0s) to produce the checksum.
- **To create the checksum the sender does the following:**
  - The packet is divided into k sections, each of n bits.
  - All sections are added together using one's complement arithmetic.
  - The final result is complemented to make the checksum.
- **Checksum Calculation at the Receiver**
  - The receiver divides the received packet into k sections and adds all sections.
  - It then complements the result.
  - If the final result is 0, the packet is accepted; otherwise, it is rejected.
  - Figure 7.22 shows graphically what happens at the sender and the receiver.

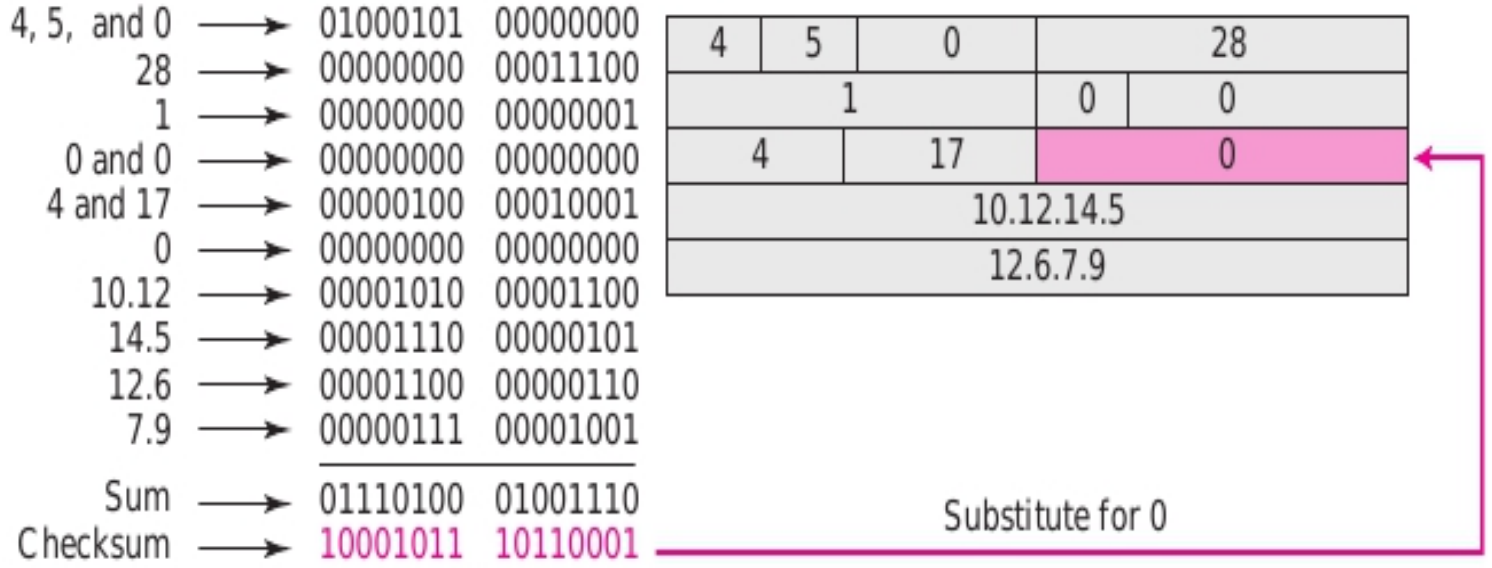
Figure 7.22 *Checksum concept*



- **Checksum in the IP Packet**
  - The implementation of the checksum in the IP packet follows the same principles discussed above.
  - First, the value of the checksum field is set to 0.
  - Then, the entire header is divided into 16-bit sections and added together.
  - The result (sum) is complemented and inserted into the checksum field.
  - **The checksum in the IP packet covers only the header, not the data.**
  - There are two good reasons for this.
  - **First,** all higher-level protocols that encapsulate data in the IP datagram have a checksum field that covers the whole packet.
  - Therefore, the checksum for the IP datagram does not have to check the encapsulated data.
  - **Second**, the header of the IP packet changes with each visited router, but the data do not.
  - So the checksum includes only the part that has changed.
  - If the data were included, each router would have to recalculate the checksum for the whole packet, which means an increase in processing time.

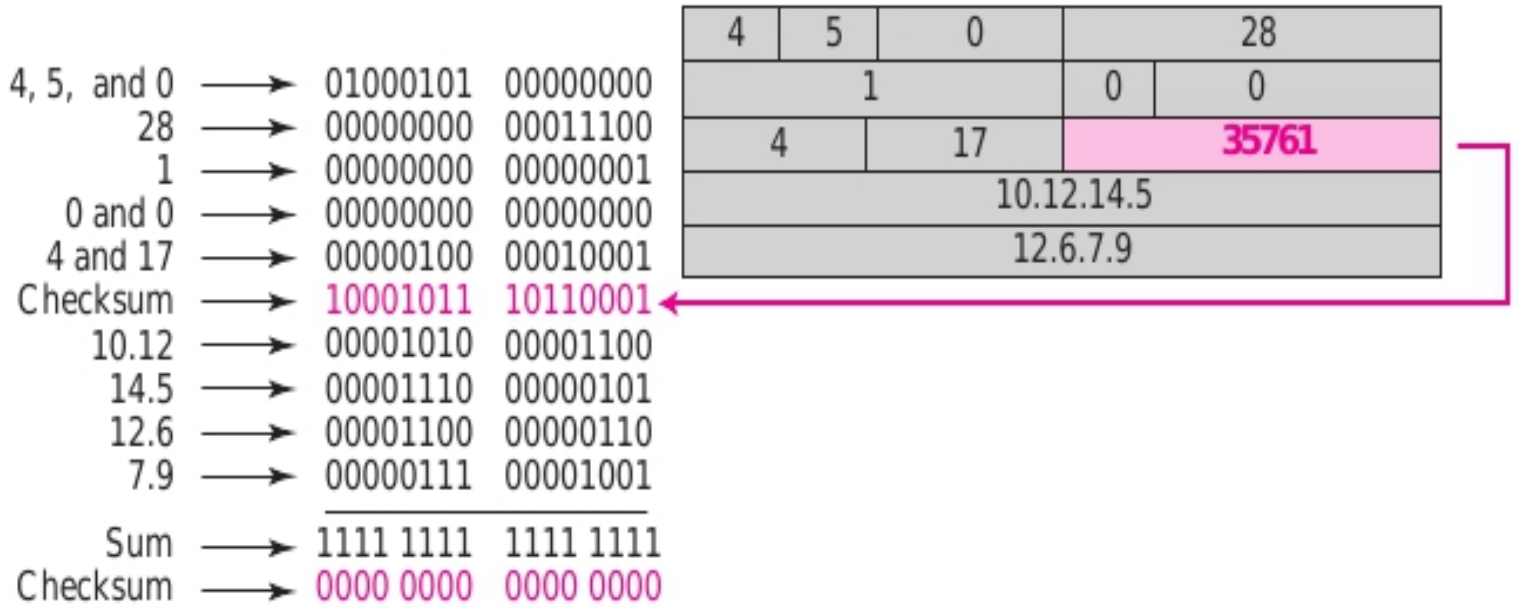
- **Example 7.17**
  - Figure 7.24 shows an example of a checksum calculation at the sender site for an IP header with- out options.
  - The header is divided into 16-bit sections.
  - All the sections are added and the sum is complemented.
  - The result is inserted in the checksum field.

Figure 7.24 *Example of checksum calculation at the sender*



- **Example 7.18**
  - Figure 7.25 shows the checking of checksum calculation at the receiver site (or intermediate router) assuming that no errors occurred in the header.
  - The header is divided into 16-bit sections.
  - All the sections are added and the sum is complemented.
  - Since the result is 16 0s, the packet is accepted.

Figure 7.25 *Example of checksum calculation at the receiver*



- **Q.19** An IPv4 datagram has arrived with the following information in the header (in hexadecimal):
- Ox45 00 00 54 00 03 58 50 20 06 00 00 7C 4E 03 02 B4 OE OF 02
- a. Is the packet corrupted?
- b. Are there any options?
- c. Is the packet fragmented?
- d. What is the size of the data?
- e. How many more routers can the packet travel to?
- f. What is the identification number of the packet?
- g. What is the type of service?
- **ANS :**
- Let us first find the value of header fields before answering the questions:
- VER = 0x4 = 4
- HLEN =0x5 = 5 →5*4 = 20
- Service =0x00 = 0
- Total Length = 0x0054 = 84
- Identification = 0x0003 = 3
- Flags and Fragmentation = 0x0000 →D = 0 M= 0 offset = 0
- Time to live = 0x20 = 32
- Protocol = 0x06 = 6
- Checksum = 0x5850
- Source Address: 0x7C4E0302 = 124.78.3.2
- Destination Address: 0xB40E0F02 = 180.14.15.2

- We can then answer the questions:
- a. If we calculate the checksum, we get 0x0000. The packet is not corrupted.
- b. Since the length of the header is 20 bytes, there are no options.
- c. Since M = 0 and offset = 0, the packet is not fragmented.
- d. The total length is 84. Data size is 64 bytes (84 −20).
- e. Since the value of time to live = 32, the packet may visit up to 32 more routers.
- f. The identification number of the packet is 3.
- g. The type of service is normal.

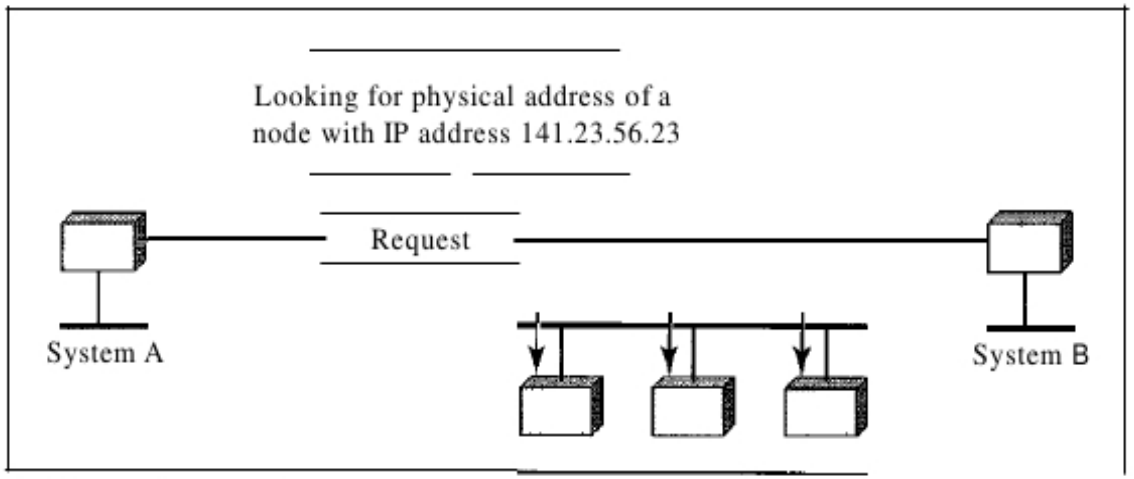
- **Address Mapping**
  - We need protocols to create a mapping between physical and logical addresses.
  - IP packets use logical (host-to-host) addresses.
  - These packets, however, need to be encapsulated in a frame, which needs physical addresses (node-to-node).
  - We will see that a protocol called ARP, the Address Resolution Protocol, is designed for this purpose.
  - We sometimes need reverse mapping-mapping a physical address to a logical address.
  - For example, when booting a diskless network or leasing an IP address to a host.
  - Three protocols are designed for this purpose: RARP, BOOTP, and DHCP.

- An internet is made of a combination of physical networks connected by internetworking devices such as routers.
- A packet starting from a source host may pass through several different physical networks before finally reaching the destination host.
- The hosts and routers are recognized at the network level by their logical (IP) addresses.
- However, packets pass through physical networks to reach these hosts and routers.
- At the physical level, the hosts and routers are recognized by their physical addresses.
- A physical address is a local address. Its jurisdiction is a local network.
- It must be unique locally, but is not necessarily unique universally.
- It is called a physical address because it is usually (but not always) implemented in hardware.
- An example of a physical address is the 48-bit MAC address in the Ethernet protocol, which is imprinted on the NIC installed in the host or router.
- The physical address and the logical address are two different identifiers.
- We need both because a physical network such as Ethernet can have two different protocols at the network layer such as IP and IPX (Novell) at the same time.

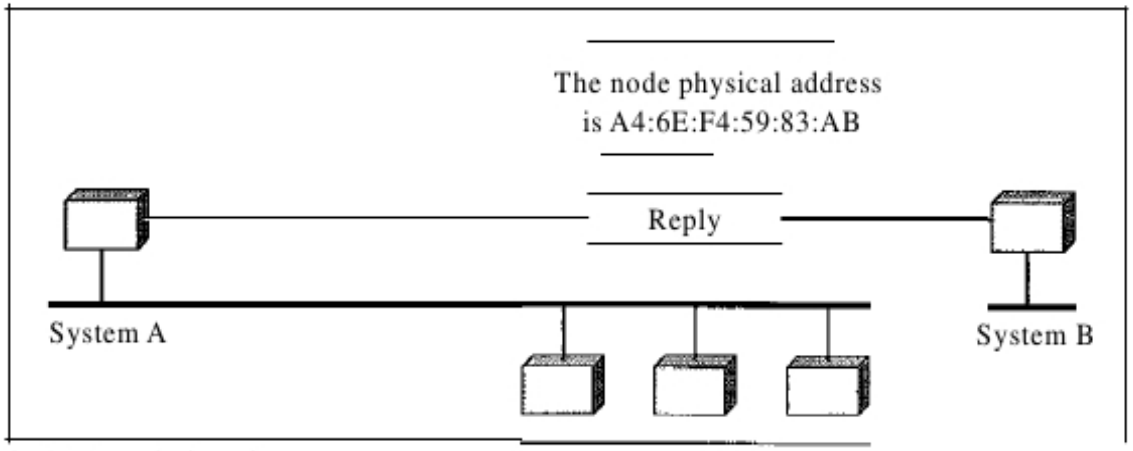
- This means that delivery of a packet to a host or a router requires two levels of addressing: logical and physical.
- We need to be able to map a logical address to its cor- responding physical address and vice versa.
- These can be done by using either static or dynamic mapping.
- **Static mapping** involves in the creation of a table that associates a logical address with a physical address.
- This table is stored in each machine on the network.
- Each machine that knows, for example, the IP address of another machine but not its physical address can look it up in the table.
- This has some limitations because physical addresses may change in the following ways:
  - 1. A machine could change its NIC, resulting in a new physical address.
  - 2. In some LANs, such as LocalTalk, the physical address changes every time the computer is turned on.
  - 3. A mobile computer can move from one physical network to another, resulting in a change in its physical address.

- **Mapping Logical to Physical Address: ARP**
  - Anytime a host or a router has an IP datagram to send to another host or router, it has the logical (IP) address of the receiver.
  - The logical (IP) address is obtained from the DNS if the sender is the host or it is found in a routing table if the sender is a router.
  - But the IP datagram must be encapsulated in a frame to be able to pass through the physical network.
  - This means that the sender needs the physical address of the receiver.
  - The host or the router sends an ARP query packet.
  - The packet includes the physical and IP addresses of the sender and the IP address of the receiver.
  - Because the sender does not know the physical address of the receiver, the query is broadcast over the network (see Figure 21.1).

Figure 21.1 *ARP operation*



a. ARP request is broadcast

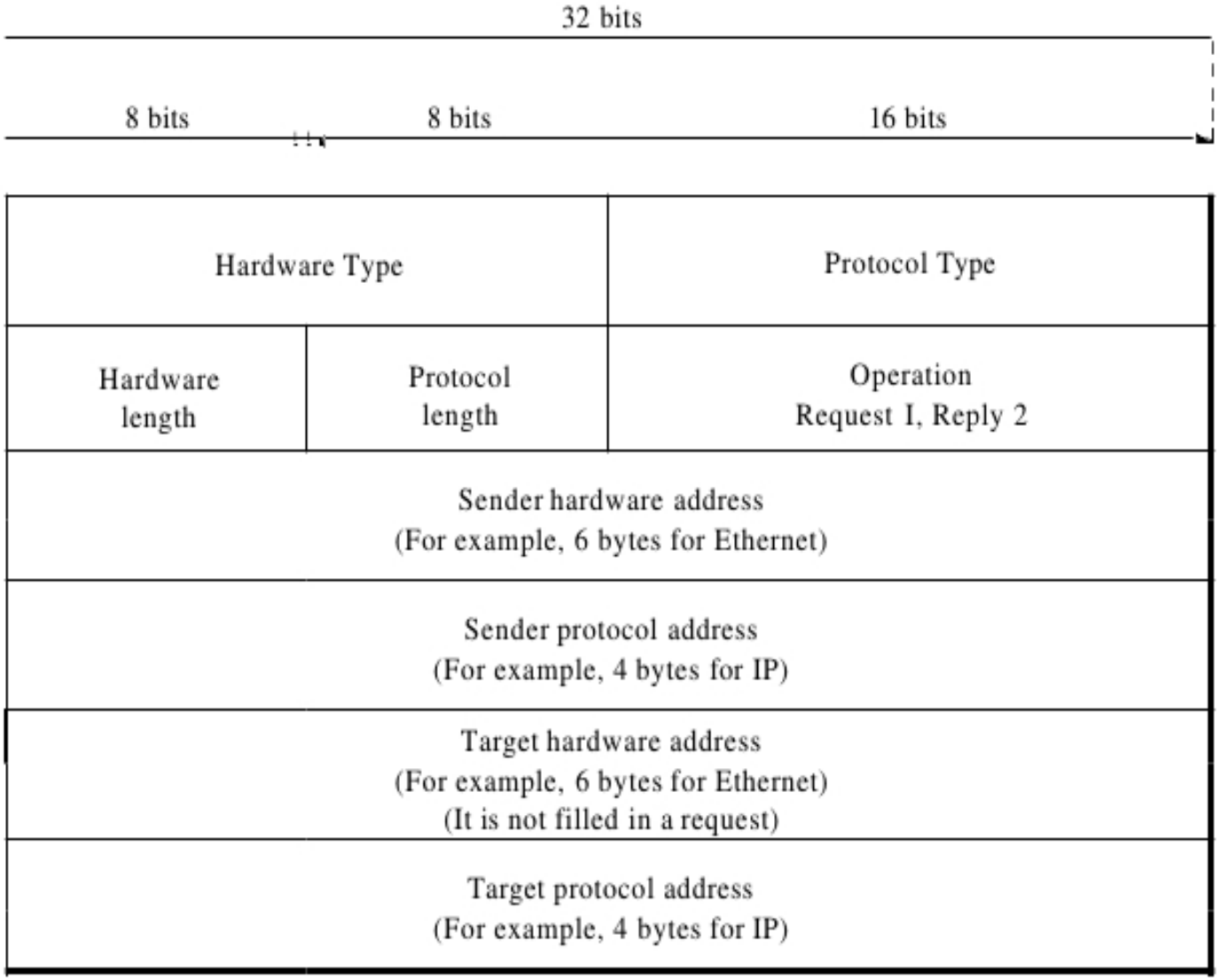


b. ARP reply is unicast

- Every host or router on the network receives and processes the ARP query packet, but only the intended recipient recognizes its IP address and sends back an ARP response packet.
- The response packet contains the recipient's IP and physical addresses.
- The packet is unicast directly to the inquirer by using the physical address received in the query packet.

- **Packet Format**
  - Figure 21.2 shows the format of an ARP packet.

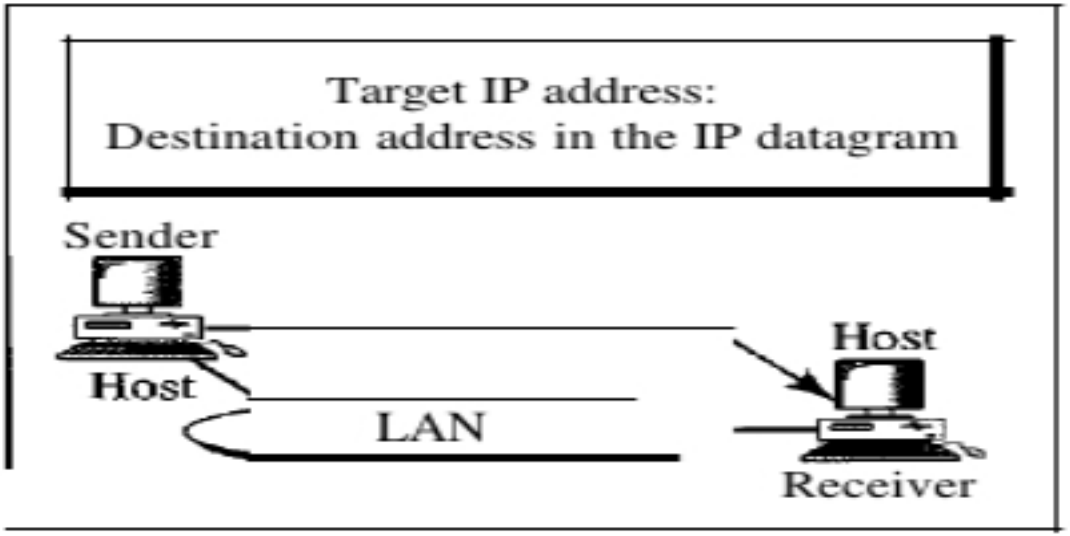
Figure 21.2 *ARP packet*



- The fields are as follows:
- **Hardware type**.
  - This is a 16-bit field defining the type of the network on which ARP is running.
  - Each LAN has been assigned an integer based on its type.
  - For example, Ethernet is given type 1.
  - ARP can be used on any physical network.
- **Protocol type.**
  - This is a 16-bit field defining the protocol.
  - For example, the value of this field for the IPv4 protocol is 0800 16 , ARP can be used with any higher-level protocol.
- **Hardware length.**
  - This is an 8-bit field defining the length of the physical address in bytes.
  - For example, for Ethernet the value is 6.
- **Protocol length**.
  - This is an 8-bit field defining the length of the logical address in bytes.
  - For example, for the IPv4 protocol the value is 4.
- **Operation**.
  - This is a 16-bit field defining the type of packet.
  - Two packet types are defined: ARP request (1) andARP reply (2).

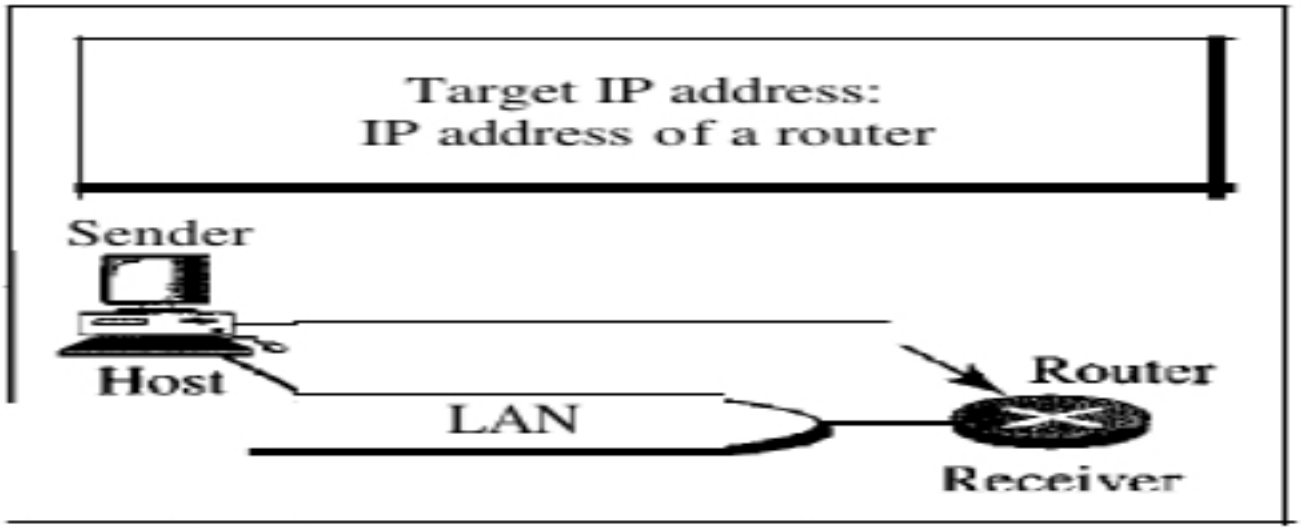
- **Sender hardware address**.
  - This is a variable-length field defining the physical address of the sender.
  - For example, for Ethernet this field is 6 bytes long.
- **Sender protocol address**.
  - This is a variable-length field defining the logical (for example, IP) address of the sender.
  - For the IP protocol, this field is 4 bytes long.
- **Target hardware address.**
  - This is a variable-length field defining the physical address of the target.
  - For example, for Ethernet this field is 6 bytes long.
  - For an ARP request message, this field is allOs because the sender does not know the physical address of the target.
- **Target protocol address**.
  - This is a variable-length field defining the logical (for example, IP) address of the target.
  - For the IPv4 protocol, this field is 4 bytes long.

- **Four Different Cases of ARP**
  - The following are four different cases in which the services of ARP can be used (see Figure 21.4).
  - **1.** The sender is a host and wants to send a packet to another host on the same network.
  - In this case, the logical address that must be mapped to a physical address is the destination IP address in the datagram header.



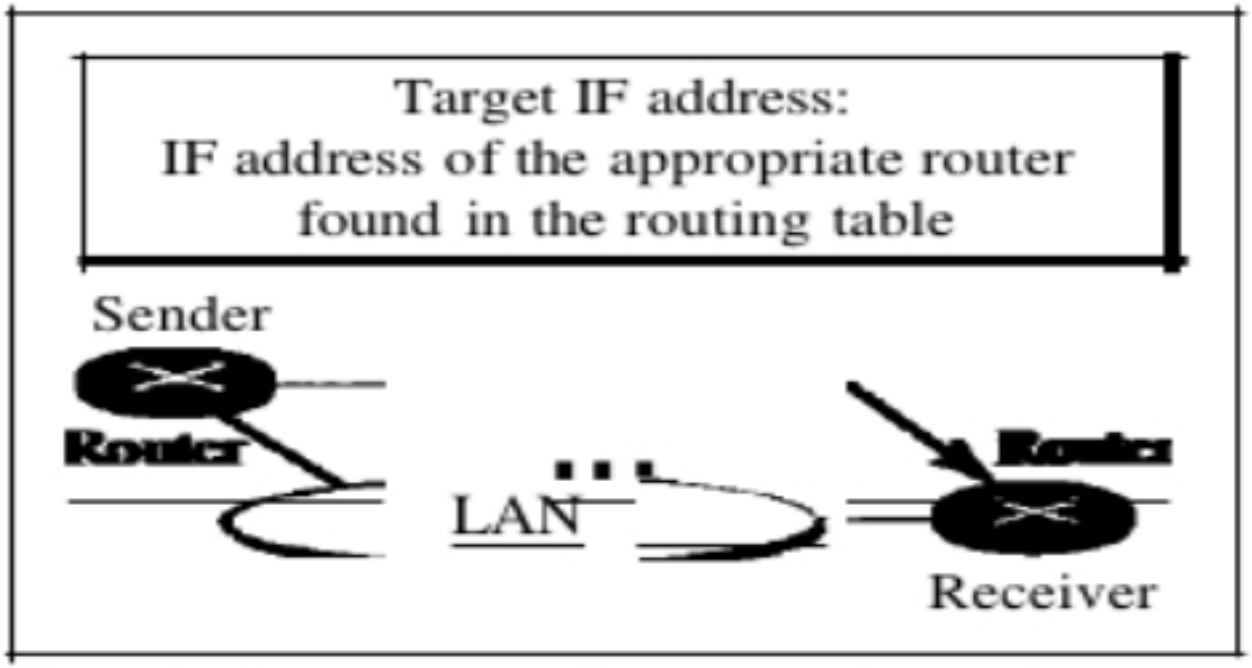
Case 1. A host has a packet to send to another host on the same network.

- **2.** The sender is a host and wants to send a packet to another host on another network.
- In this case, the host looks at its routing table and finds the IP address of the next hop (router) for this destination.
- If it does not have a routing table, it looks for the IP address of the default router.
- The IP address of the router becomes the logical address that must be mapped to a physical address.



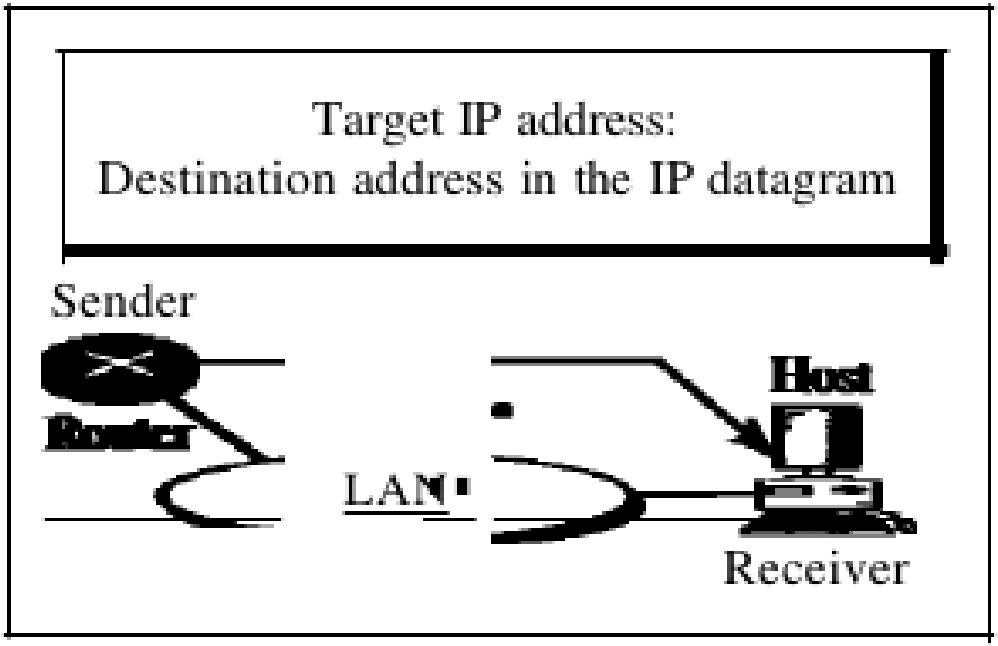
Case 2. A host wants to send a packet to another host on another network.
It must first be delivered to a router.

- **3.** The sender is a router that has received a datagram destined for a host on another network.
- It checks its routing table and finds the IP address of the next router.
- The IP address of the next router becomes the logical address that must be mapped to a physical address.



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.

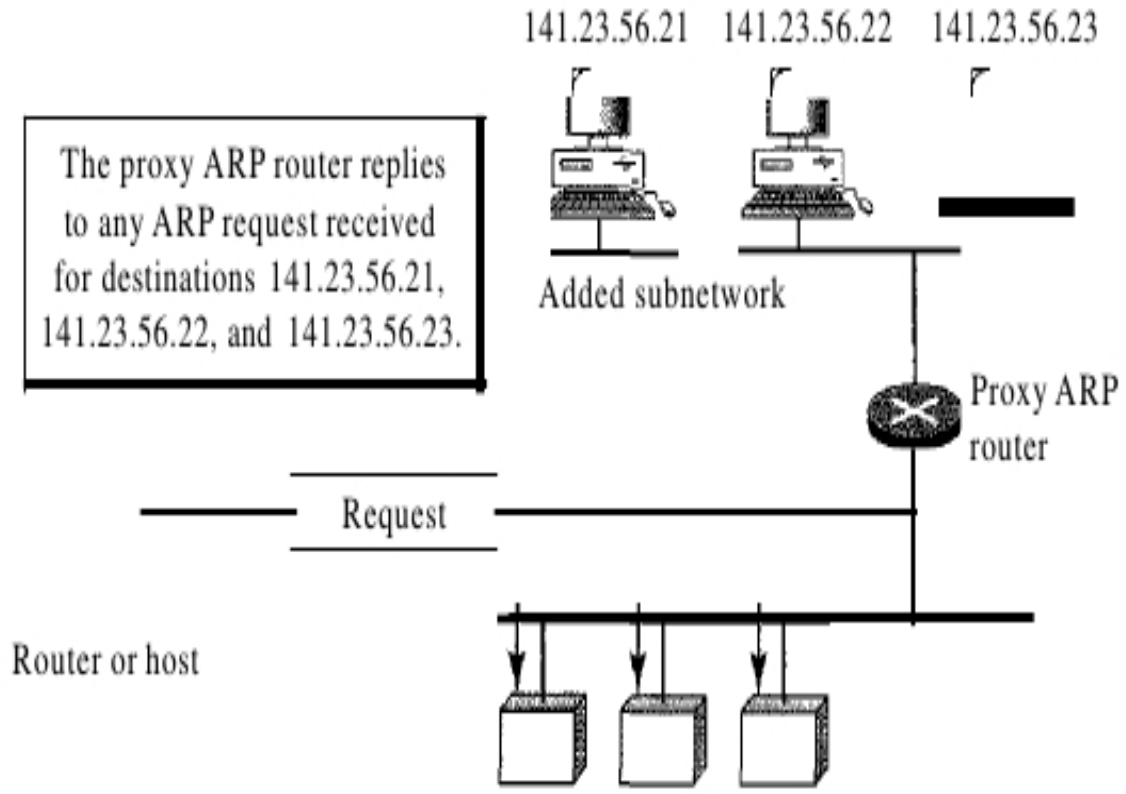
– **4.** The sender is a router that has received a datagram destined for a host on the same network.
– The destination IP address of the datagram becomes the logical address that must be mapped to a physical address.



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

- **Proxy ARP**
  - A technique called proxy ARP is used to create a subnetting effect.
  - A proxy ARP is an ARP that acts on behalf of a set of hosts.
  - Whenever a router running a proxy ARP receives an ARP request looking for the IP address of one of these hosts, the router sends an ARP reply announcing its own hardware (physical) address.
  - After the router receives the actual IP packet, it sends the packet to the appropriate host or router.
  - Let us give an example.
  - In Figure 21.6 the ARP installed on the right-hand host will answer only to an ARP request with a target IP address of 141.23.56.23.

Figure 21.6 *Proxy ARP*



- However, the administrator may need to create a subnet without changing the whole system to recognize subnetted addresses.
- One solution is to add a router running a proxy ARP.
- In this case, the router acts on behalf of all the hosts installed on the subnet.
- When it receives an ARP request with a target IP address that matches the address of one of its proteges (141.23.56.21, 141.23.56.22, or 141.23.56.23), it sends an ARP reply and announces its hardware address as the target hardware address.
- When the router receives the IP packet, it sends the packet to the appropriate host.

- **Mapping Physical to Logical Address: RARP, BOOTP, and DHCP**
  - There are occasions in which a host knows its physical address, but needs to know its logical address.
  - This may happen in two cases:
  - 1. A diskless station is just booted.
    - The station can find its physical address by checking its interface, but it does not know its IP address.
  - 2. An organization does not have enough IP addresses to assign to each station; it needs to assign IP addresses on demand.
    - The station can send its physical address and ask for a short time lease.

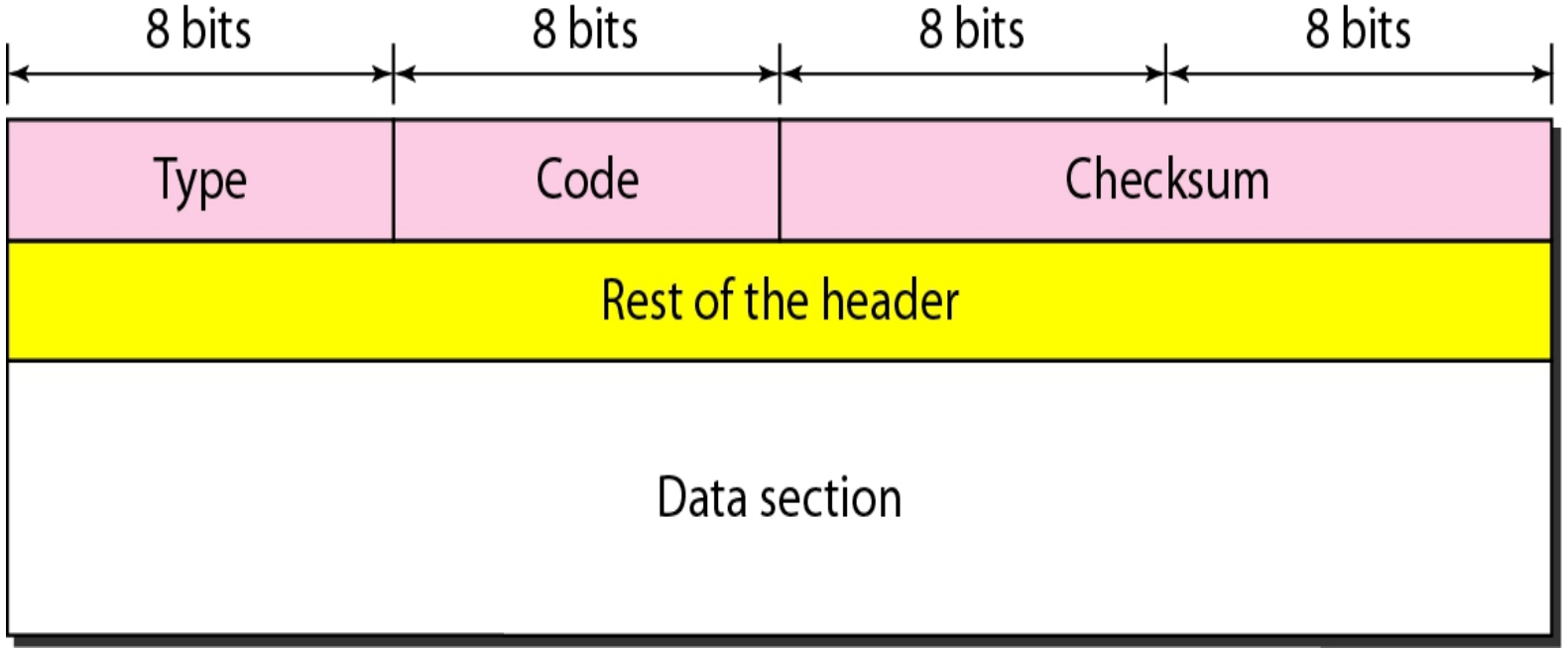
- **RARP**
  - Reverse Address Resolution Protocol (RARP) finds the logical address for a machine that knows only its physical address.
  - Each host or router is assigned one or more logical (IP) addresses, which are unique and independent of the physical (hardware) address of the machine.
  - To create an IP datagram, a host or a router needs to know its own IP address or addresses.
  - The IP address of a machine is usually read from its configuration file stored on a disk file.
  - However, a diskless machine is usually booted from ROM, which has minimum booting information.
  - The ROM is installed by the manufacturer.
  - It cannot include the IP address because the IP addresses on a network are assigned by the network administrator.
  - The machine can get its physical address (by reading its NIC, for example), which is unique locally.
  - It can then use the physical address to get the logical address by using the RARP protocol.
  - A RARP request is created and broadcast on the local network.
  - Another machine on the local network that knows all the IP addresses will respond with a RARP reply.
  - The requesting machine must be running a RARP client program; the responding machine must be running a RARP server program.

- **Problem**
  - There is a serious problem with RARP: Broadcasting is done at the data link layer.
  - The physical broadcast address, all is in the case of Ethernet, does not pass the boundaries of a network.
  - This means that if an administrator has several networks or several subnets, it needs to assign a RARP server for each network or subnet.
  - This is the reason that RARP is almost obsolete.
  - Two protocols, BOOTP and DHCp, are replacing RARP.

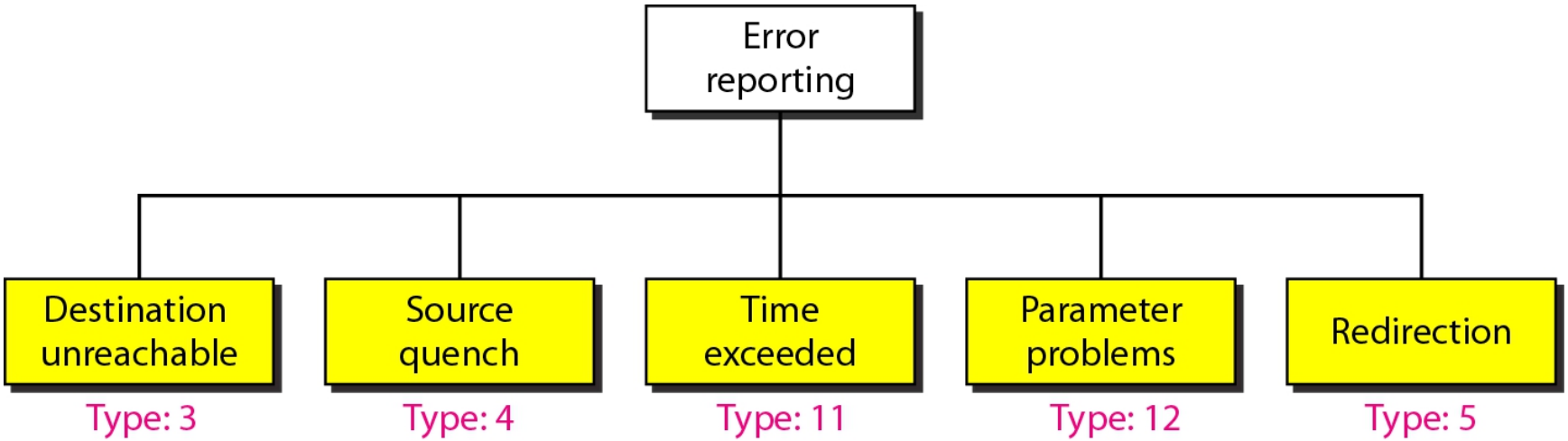
- **Error Reporting: ICMP:**
  - IP provides unreliable and connectionless datagram delivery.
  - It was designed this way to make efficient use of network resources.
  - The IP protocol is a best-effort delivery service that delivers a datagram from its original source to its final destination.
  - However, it has two deficiencies: lack of error control and lack of assistance mechanisms.
  - The IP protocol has no error-reporting or error-correcting mechanism.
  - What happens if something goes wrong? What happens if a router must discard a datagram because it cannot find a router to the final destination, or because the time-to-live field has a zero value? What happens if the final destination host must discard all fragments of a datagram because it has not received all fragments within a predetermined time limit?
  - These are examples of situations where an error has occurred and the IP protocol has no built-in mechanism to notify the original host.
  - The IP protocol also lacks a mechanism for host and management queries.
  - A host sometimes needs to determine if a router or another host is alive.
  - And sometimes a network administrator needs information from another host or router.

- **Types of Messages**
  - ICMP messages are divided into two broad categories: error-reporting messages and query messages.
  - The error-reporting messages report problems that a router or a host (destination) may encounter when it processes an IP packet.
  - The query messages, which occur in pairs, help a host or a network manager get specific information from a router or another host.
  - For example, nodes can discover their neighbors.
  - Also, hosts can discover and learn about routers on their network, and routers can help a node redirect its messages.
- **Message Format**
  - An ICMP message has an 8-byte header and a variable-size data section.
  - Although the general format of the header is different for each message type, the first 4 bytes are common to all.

- As Figure 21.8 shows, the first field, ICMP type, defines the type of the message.
- The code field specifies the reason for the particular message type.
- The last common field is the checksum field .
- The rest of the header is specific for each message type.
- The data section in error messages carries information for finding the original packet that had the error.
- In query messages, the data section carries extra information based on the type of the query.



- **Error Reporting**
  - One of the main responsibilities of ICMP is to report errors. Although technology has produced increasingly reliable transmission media, errors still exist and must be handled.
  - IP, is an unreliable protocol.
  - This means that error checking and error control are not a concern of IP.
  - ICMP was designed, in part, to compensate for this shortcoming.
  - However, ICMP does not correct errors-it simply reports them.
  - Error correction is left to the higher-level protocols.
  - Error messages are always sent to the original source because the only information available in the datagram about the route is the source and destination IP addresses.
  - ICMP uses the source IP address to send the error message to the source (originator) of the datagram.



•The following are 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 IeMP 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.

Note that all error messages contain a data section that includes the IP header of the original datagram plus the first 8 bytes of data in that datagram.

The original datagram header is added to give the original source, which receives the error message, information about the datagram itself.

The 8 bytes of data are included because, UDP and TCP protocols, the first 8 bytes provide information about the port numbers (UDP and TCP) and sequence number (TCP).

This information is needed so the source can inform the protocols (TCP or UDP) about the error.

ICMP forms an error packet, which is then encapsulated in an IP datagram.

Figure 9.2 *ICMP encapsulation*

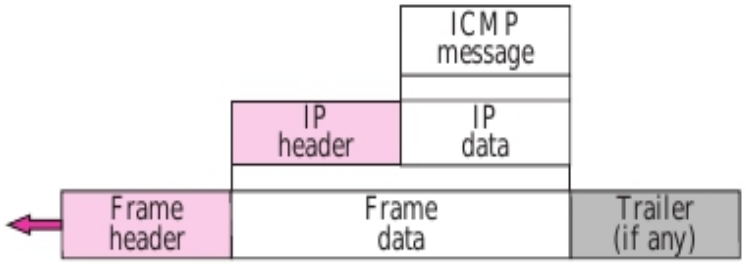
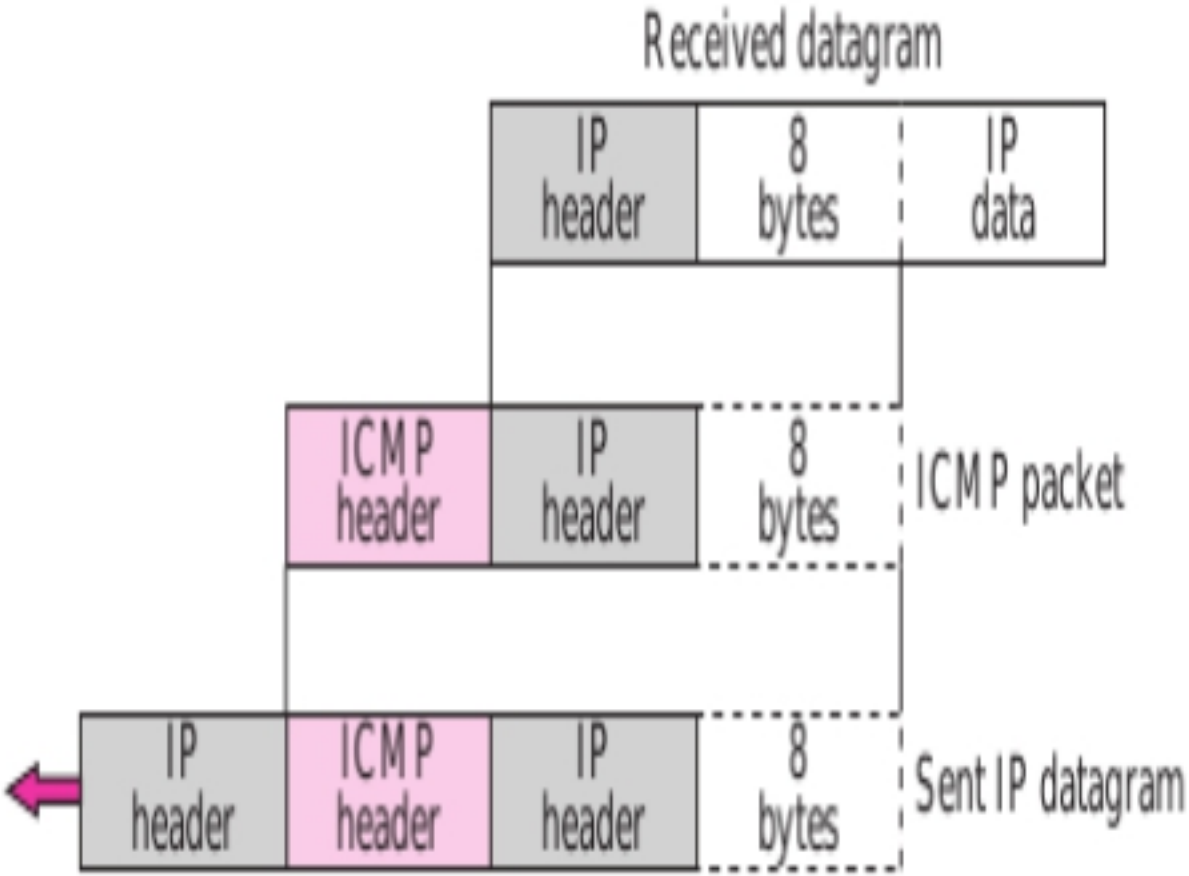


Figure 9.5 *Contents of data field for the error messages*



- **Destination Unreachable**
  - When a router cannot route a datagram or a host cannot deliver a datagram, the datagram is discarded and the router or the host sends a destination-unreachable message back to the source host that initiated the datagram.
  - Note that destination-unreachable messages can be created by either a router or the destination host.
- **Source Quench**
  - The IP protocol is a connectionless protocol.
  - There is no communication between the source host, which produces the datagram, the routers, which forward it, and the destination host, which processes it.
  - One of the ramifications of this absence of communication is the lack of flow control.
  - IP does not have a flow control mechanism embedded in the protocol.
  - The lack of flow control can create a major problem in the operation of IP: congestion.
  - The source host never knows if the routers or the destination host has been overwhelmed with datagrams.
  - The source host never knows if it is producing datagrams faster than can be forwarded by routers or processed by the destination host.

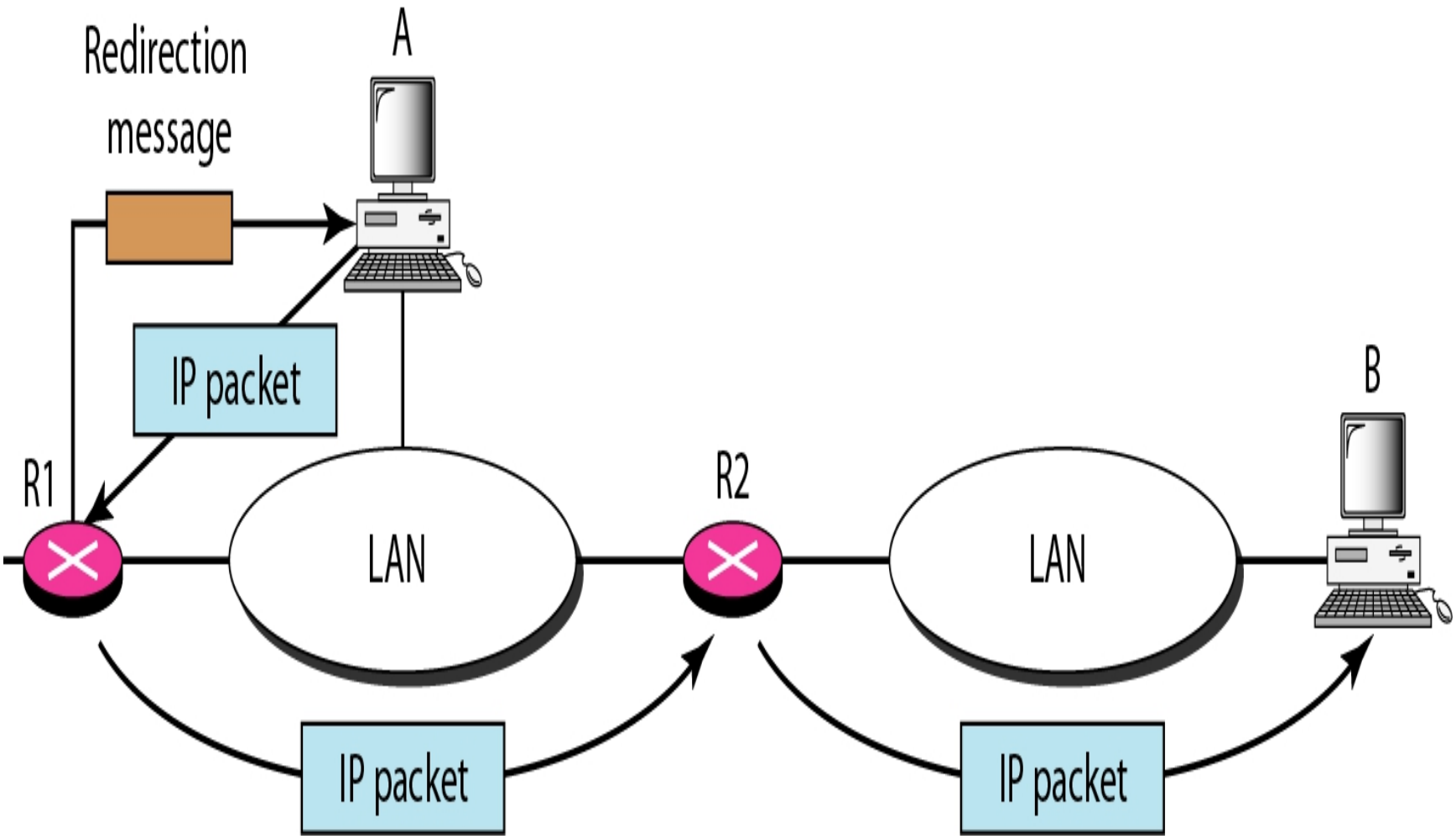
- The lack of flow control can create congestion in routers or the destination host.
- A router or a host has a limited-size queue (buffer) for incoming datagrams waiting to be forwarded (in the case of a router) or to be processed (in the case of a host).
- If the datagrams are received much faster than they can be forwarded or processed, the queue may overflow.
- In this case, the router or the host has no choice but to discard some of the datagrams.
- The source-quench message in ICMP was designed to add a kind of flow control to the IP.
- When a router or host discards a datagram due to congestion, it sends a source-quench message to the sender of the datagram.
- This message has two purposes.
- First, it informs the source that the datagram has been discarded.
- Second, it warns the source that there is congestion somewhere in the path and that the source should slow down (quench) the sending process.

- **Time Exceeded**
  - The time-exceeded message is generated in two cases: routers use routing tables to find the next hop (next router) that must receive the packet.
  - If there are errors in one or more routing tables, a packet can travel in a loop or a cycle, going from one router to the next or visiting a series of routers endlessly.
  - Each datagram contains a field called time to live that controls this situation.
  - When a datagram visits a router, the value of this field is decremented by 1.
  - When the time-to-live value reaches 0, after decrementing, the router discards the datagram.
  - However, when the datagram is discarded, a time-exceeded message must be sent by the router to the original source.
  - Second, a time-exceeded message is also generated when not all fragments that make up a message arrive at the destination host within a certain time limit.

- **Parameter Problem**
  - Any ambiguity in the header part of a datagram can Create serious problems as the datagram travels through the Internet.
  - If a router or the destination host discovers an ambiguous or missing value in any field of the datagram, it discards the datagram and sends a parameter-problem message back to the source.
- **Redirection**
  - When a router needs to send a packet destined for another network, it must know the IP address of the next appropriate router. The same is true if the sender is a host.
  - Both routers and hosts, then, must have a routing table to find the address of the router or the next router.
  - Routers take part in the routing update process, and are supposed to be updated constantly.
  - Routing is dynamic.

- This concept of redirection is shown in Figure 21.11.
- Host A wants to send a datagram to host B.
- Router R2 is obviously the most efficient routing choice, but host A did not choose router R2.
- The datagram goes to R1 instead.
- Router R1, after consulting its table, finds that the packet should have gone to R2.
- It sends the packet to R2 and, at the same time, sends a redirection message to host A.
- Host A's routing table can now be updated.

Figure 21.11 ***Redirection concept***



- **Query**

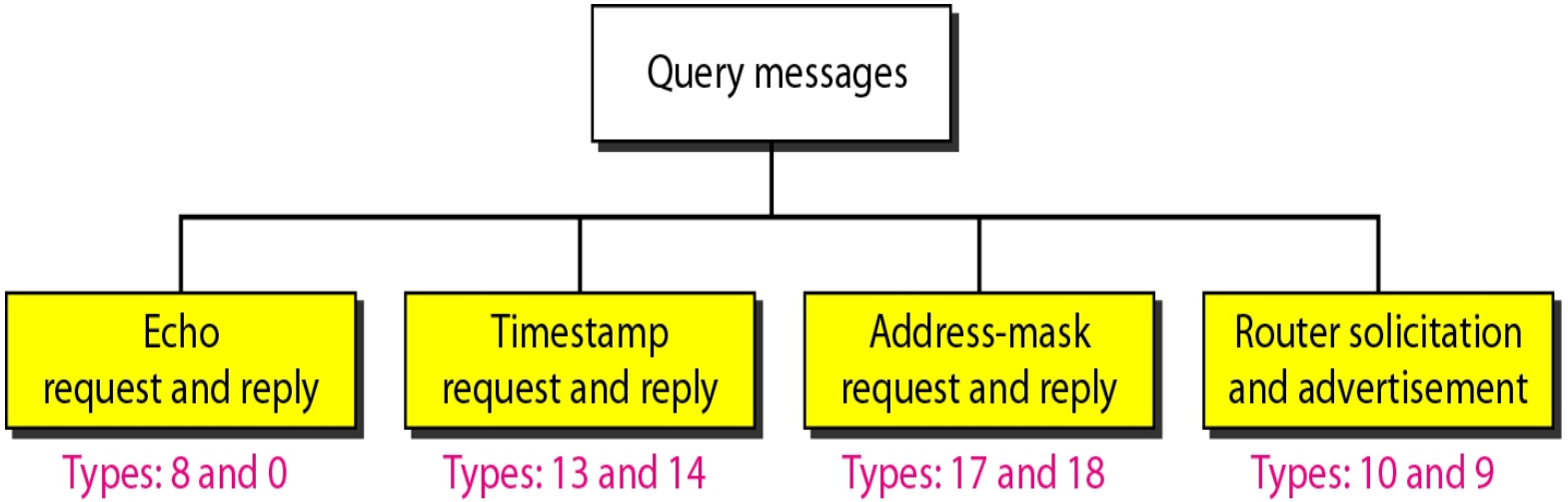
  In addition to error reporting, ICMP can diagnose some network problems.

  This is accomplished through the query messages, a group of four different pairs of messages, as shown in Figure.

  In this type of ICMP message, a node sends a message that is answered in a specific format by the destination node.

  A query message is encapsulated in an IP packet, which in tum is encapsulated in a data link layer frame.

  However, in this case, no bytes of the original IP are included in the message, as shown in Figure.



- **Echo Request and Reply**
  - The echo-request and echo-reply messages are designed for diagnostic purposes.
  - The combination of echo-request and echo-reply messages detennines whether two systems (hosts or routers) can communicate with each other.
  - The echo-request and echo-reply messages can be used to determine if there is communication at the IP level.
  - Also, it is proof that the intermediate routers are receiving, processing, and forwarding IP datagrams.
  - Today, most systems provide a version of the ping command that can create a series (instead of just one) of echo-request and echo-reply messages, providing statistical information.

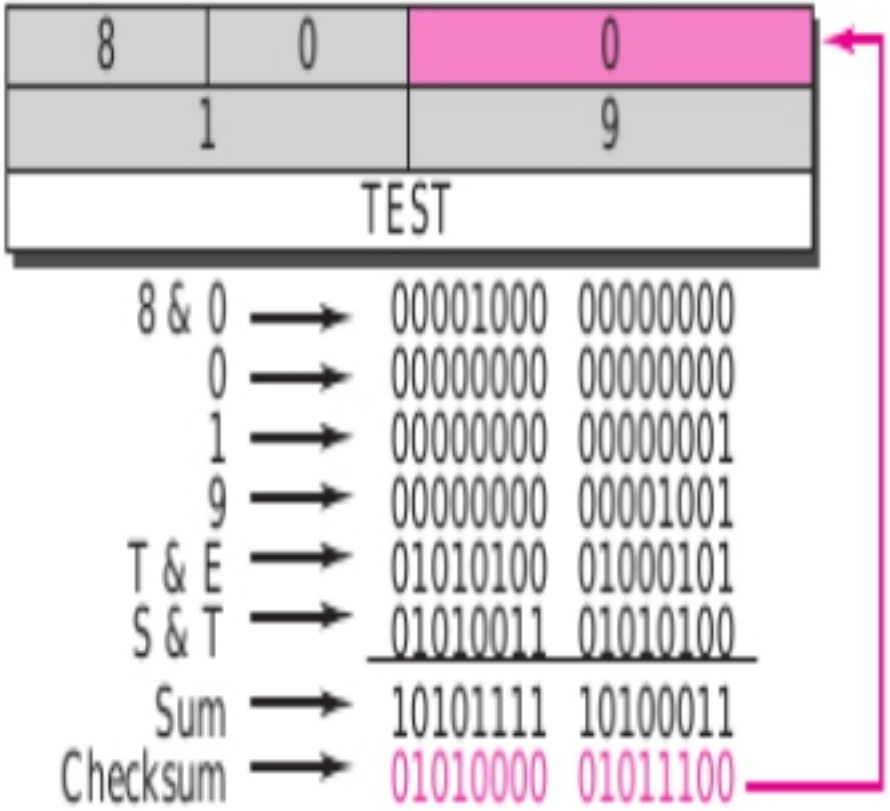
- **Timestamp Request and Reply**
  - Two machines (hosts or routers) can use the timestamp request and timestamp reply messages to determine the round-trip time needed for an IP datagram to travel between them.
  - It can also be used to synchronize the clocks in two machines.
- **Address-Mask Request and Reply**
  - A host may know its IP address, but it may not know the corresponding mask.
  - To obtain its mask, a host sends an address-mask-request message to a router on the LAN.
  - If the host knows the address of the router, it sends the request directly to the router.
  - If it does not know, it broadcasts the message.
  - The router receiving the address-mask-request message responds with an address-mask-reply message, providing the necessary mask for the host.

- **Router Solicitation and Advertisement**
  - A host can broadcast (or multicast) a router-solicitation message.
  - The router or routers that receive the solicitation message broadcast their routing information using the router-advertisement message.
  - A router can also periodically send router advertisement messages even if no host has solicited.
  - Note that when a router sends out an advertisement, it announces not only its own presence but also the presence of all routers on the network of which it is aware.

- **Checksum**
  - In ICMP the checksum is calculated over the entire message (header and data).
- **Checksum Calculation**
  - The sender follows these steps using one's complement arithmetic:
  - 1. The checksum field is set to zero.
  - 2. The sum of all the 16-bit words (header and data) is calculated.
  - 3. The sum is complemented to get the checksum.
  - 4. The checksum is stored in the checksum field.
- **Checksum Testing**
  - The receiver follows these steps using one's complement arithmetic:
  - 1. The sum of all words (header and data) is calculated.
  - 2. The sum is complemented.
  - 3. If the result obtained in step 2 is 16 0s, the message is accepted; otherwise, it is rejected.

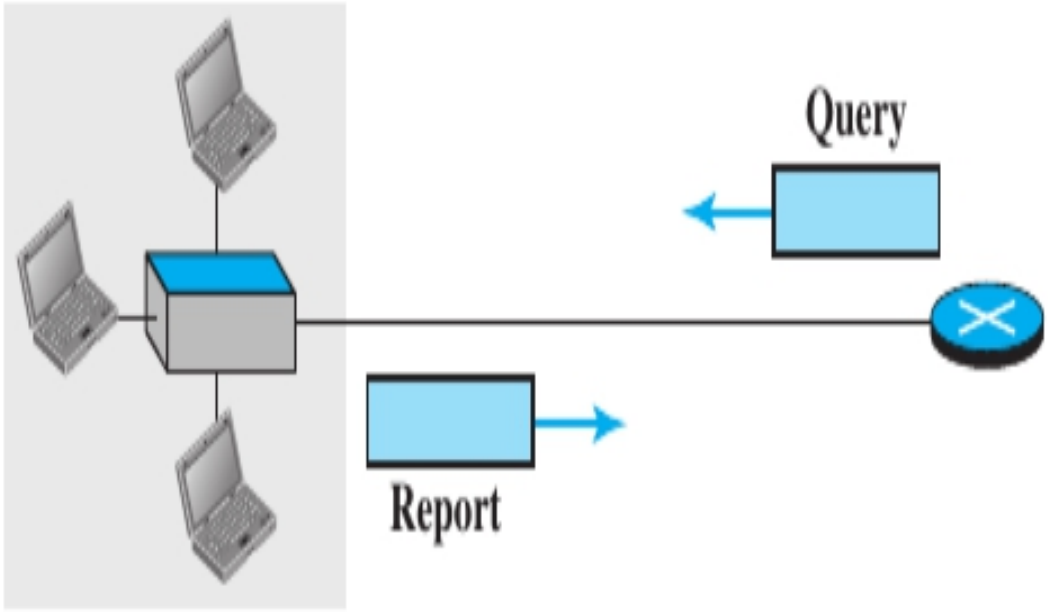
- **Example 9.1**
- Figure 9.14 shows an example of checksum calculation for a simple echo-request message.
- We randomly chose the identifier to be 1 and the sequence number to be 9.
- The message is divided into 16-bit (2-byte) words.
- The words are added together and the sum is complemented.
- Now the sender can put this value in the checksum field.

Figure 9.14 *Example of checksum calculation*



- **IGMP**
- The protocol that is used today for collecting information about group membership is the Internet Group Management Protocol (IGMP).
- IGMP is a protocol defined at the network layer; it is one of the auxiliary protocols, like ICMP, which is considered part of the IP.
- IGMP messages, like ICMP messages, are encapsulated in an IP datagram.
- **Messages**
- There are only two types of messages in IGMP version 3, query and report messages, as shown in Figure 21.16.
- A query message is periodically sent by a router to all hosts attached to it to ask them to report their interests about membership in groups.
- A report message is sent by a host as a response to a query message.

Figure 21.16 *IGMP operation*



- **Query Message**
- The query message is sent by a router to all hosts in each interface to collect information about their membership.
- There are three versions of query messages, as described
- below:
- **a.** A **general** query message is sent about membership in any group.
- It is encapsulated in a datagram with the destination address 224.0.0.1 (all hosts and routers).
- Note that all routers attached to the same network receive this message to inform them that this message is already sent and that they should refrain from resending it.

- **Query Message**
- **b.** A **group-specific query** message is sent from a router to ask about the membership related to a specific group.
- This is sent when a router does not receive a response about a specific group and wants to be sure that there is no active member of that group in the network.
- The group identifier (multicast address) is mentioned in the message.
- The message is encapsulated in a datagram with the destination address set to the corresponding multicast address.
- Although all hosts receive this message, those not interested drop it.

- **Query Message**
- **c. A source-and-group-specific query** message is sent from a router to ask about the membership related to a specific group when the message comes from a specific source or sources.
- Again the message is sent when the router does not hear about a specific group related to a specific host or hosts.
- The message is encapsulated in a datagram with the destination address set to the corresponding multicast address.
- Although all hosts receive this message, those not interested drop it.

- **Report Message**
- A report message is sent by a host as a response to a query message.
- The message contains a list of records in which each record gives the identifier of the corresponding group (multicast address) and the addresses of all sources that the host is interested in receiving messages from (inclusion).
- The record can also mention the source addresses from which the host does not desire to receive a group message (exclusion).
- The message is encapsulated in a datagram with the multicast address 224.0.0.22 (multicast address assigned to IGMPv3).

- **Report Message**
- In IGMPv3, if a host needs to join a group, it waits until it receives a query message and then sends a report message.
- If a host needs to leave a group, it does not respond to a query message.
- If no other host responds to the corre- sponding message, the group is purged from the router database.

- **Propagation of Membership Information**
- After a router has collected membership information from the hosts and other routers at its own level in the tree, it can propagate it to the router located in a higher level of the tree.
- Finally, the router at the tree root can get the membership information to build the multicast tree.
- The process, however, is more complex than what we can explain in one paragraph.
- Interested readers can check the book website for the complete description of this protocol.

- **Encapsulation**
- The IGMP message is encapsulated in an IP datagram with the value of the protocol field set to 2 and the TTL field set to 1.
- The destination IP address of the datagram, however, depends on the type of message, as shown in Table 21.1.

Table 21.1 *Destination IP Addresses*

Message Type	*IP Address*
General Query	224.0.0.1
Other Queries	Group address
Report	224.0.0.22

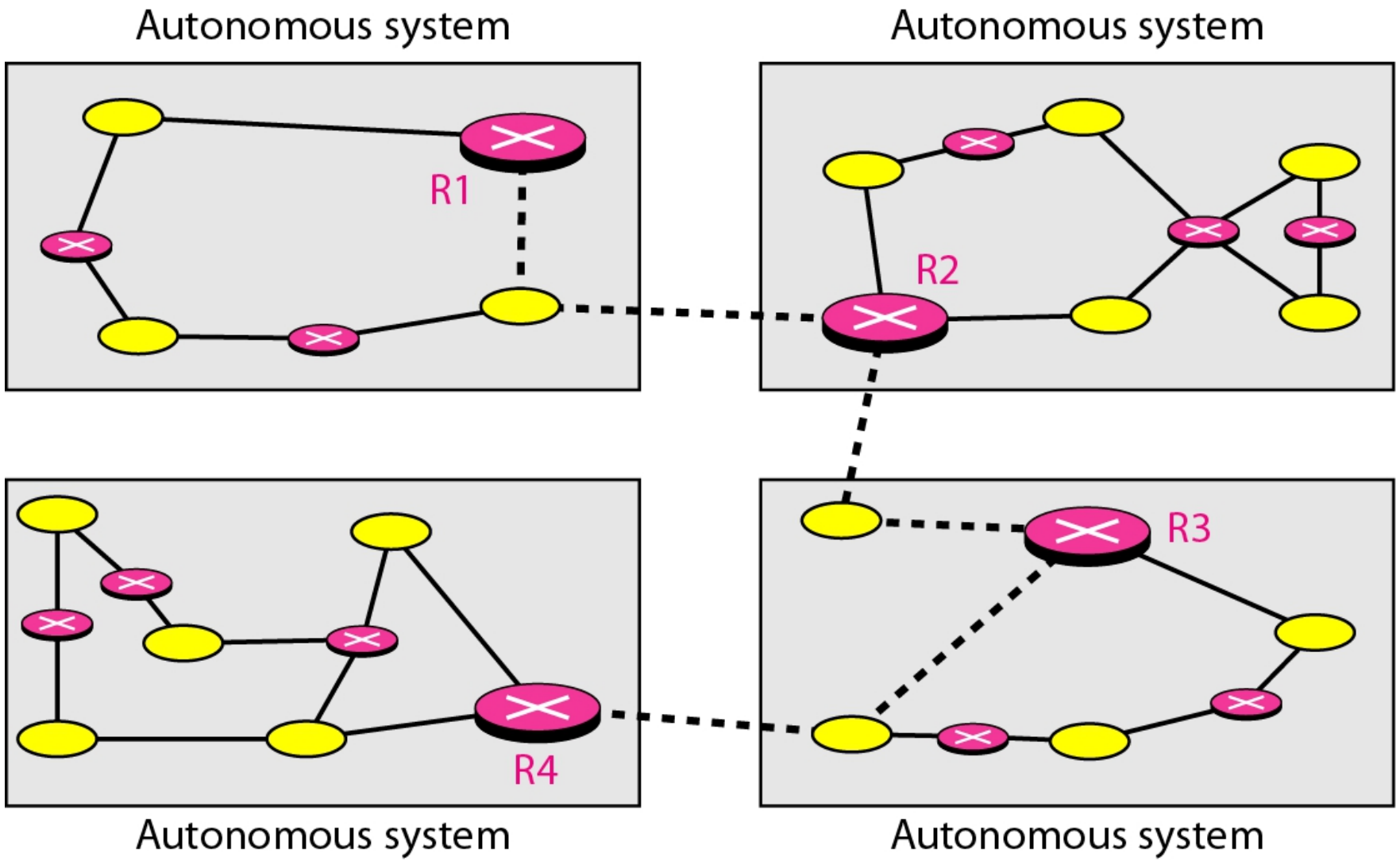
- **Unicast Routing Protocols:**
  - A routing table can be either static or dynamic.
  - A static table is one with manual entries.
  - A dynamic table, on the other hand, is one that is updated automatically when there is a change somewhere in the internet.
  - Today, an internet needs dynamic routing tables.
  - The tables need to be updated as soon as there is a change in the internet.
  - For instance, they need to be updated when a router is down, and they need to be updated whenever a better route has been found.
  - Routing protocols have been created in response to the demand for dynamic routing tables.
  - A routing protocol is a combination of rules and procedures that lets routers in the internet inform each other of changes.
  - It allows routers to share whatever they know about the internet or their neighborhood.
  - The sharing of information allows a router in San Francisco to know about the failure of a network in Texas.
  - The routing protocols also include procedures for combining information received from other routers.

- **Routing**
- **Optimization**
  - One approach is to assign a cost for passing through a network.
  - We call this cost a metric.
  - However, the metric assigned to each network depends on the type of protocol.
  - Some simple protocols, such as the Routing Information Protocol (RIP), treat all networks as equals.
  - The cost of passing through a network is the same; it is one hop count.
  - So if a packet passes through 10 networks to reach the destination, the total cost is 10 hop counts.
  - Other protocols, such as Open Shortest Path First (OSPF), allow the administrator to assign a cost for passing through a network based on the type of service required.
  - A route through a network can have different costs (metrics).
  - For example, if maximum through- put is the desired type of service, a satellite link has a lower metric than a fiber-optic line.

- On the other hand, if minimum delay is the desired type of service, a fiber-optic line has a lower metric than a satellite link.
- Routers use routing tables to help decide the best route.
- OSPF protocol allows each router to have several routing tables based on the required type of service.
- Other protocols define the metric in a totally different way.
- In the Border Gateway Protocol (BGP), the criterion is the policy, which can be set by the administrator.
- The policy defines what paths should be chosen.

- **Intra- and Interdomain Routing**
  - Today, an internet can be so large that one routing protocol cannot handle the task of updating the routing tables of all routers.
  - For this reason, an internet is divided into autonomous systems.
  - An autonomous system (AS) is a group of networks and routers under the authority of a single administration.
  - **Routing inside an autonomous system is referred to as intradomain routing.**
  - **Routing between autonomous systems is referred to as interdomain routing.**
  - Each autonomous system can choose one or more intradomain routing protocols to handle routing inside the autonomous system.
  - However, only one interdomain routing protocol handles routing between autonomous systems (see Figure 22.12).

Figure 22.12 *Autonomous systems*



Several intradomain and interdomain routing protocols are in use.

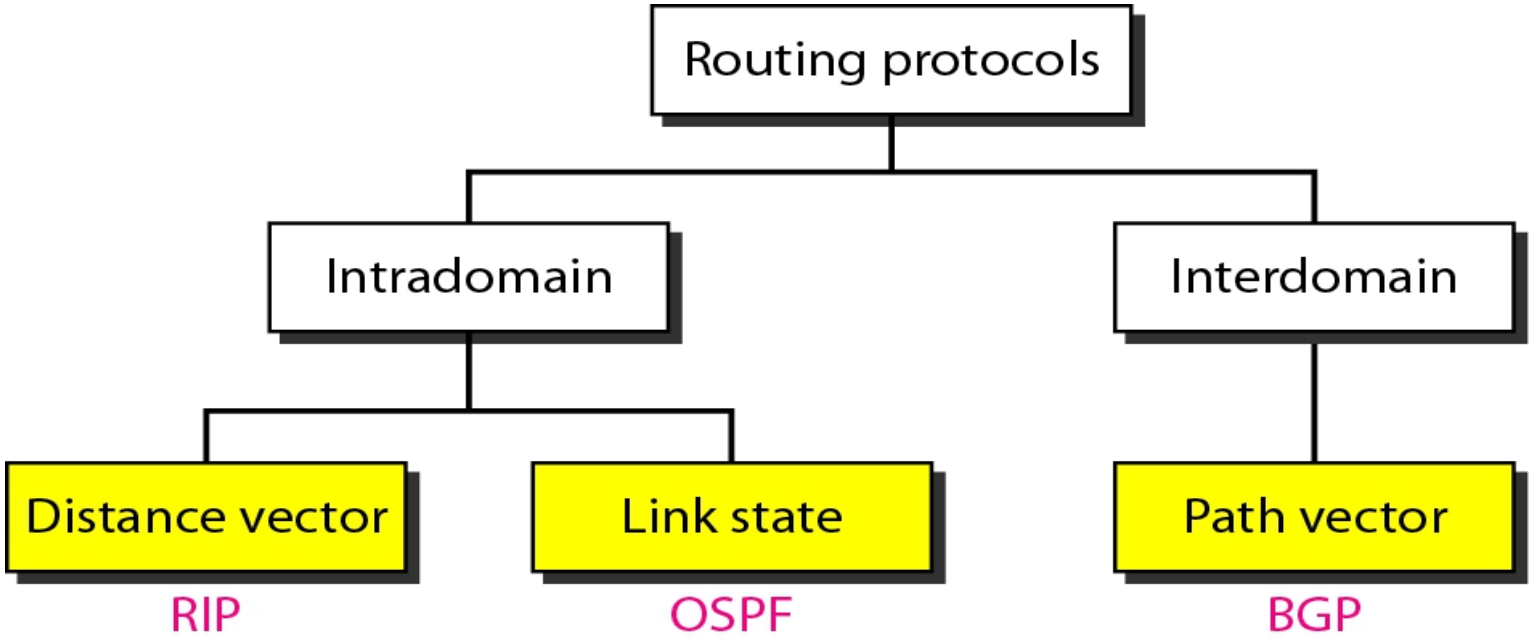
We discuss two intradomain routing protocols: distance vector and link state.

We also introduce one interdomain routing protocol: path vector

Routing Information Protocol (RIP) is an implementation of the distance vector protocol.

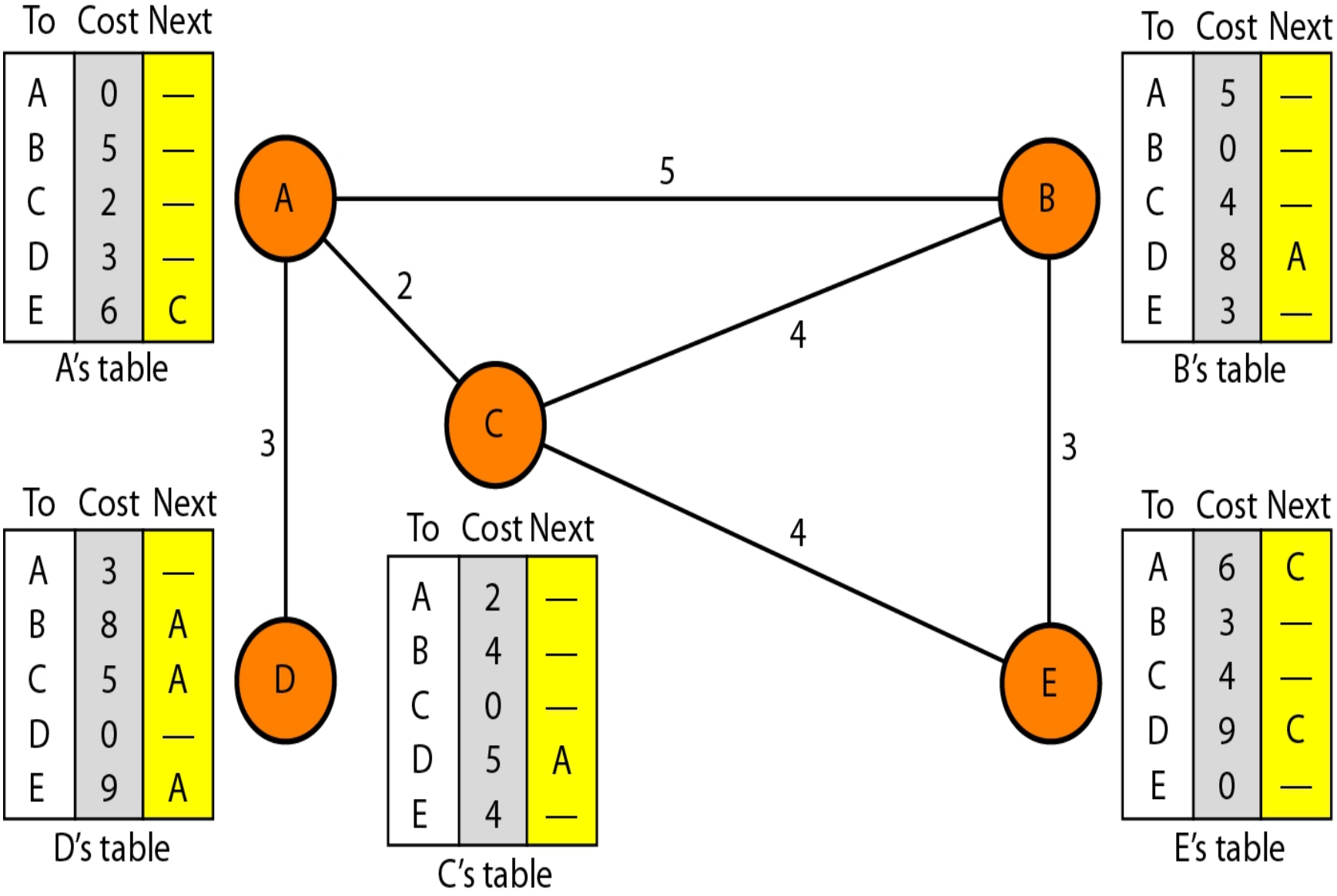
Open Shortest Path First (OSPF) is an implementation of the link state protocol.

Border Gateway Protocol (BGP) is an implementation of the path vector protocol.



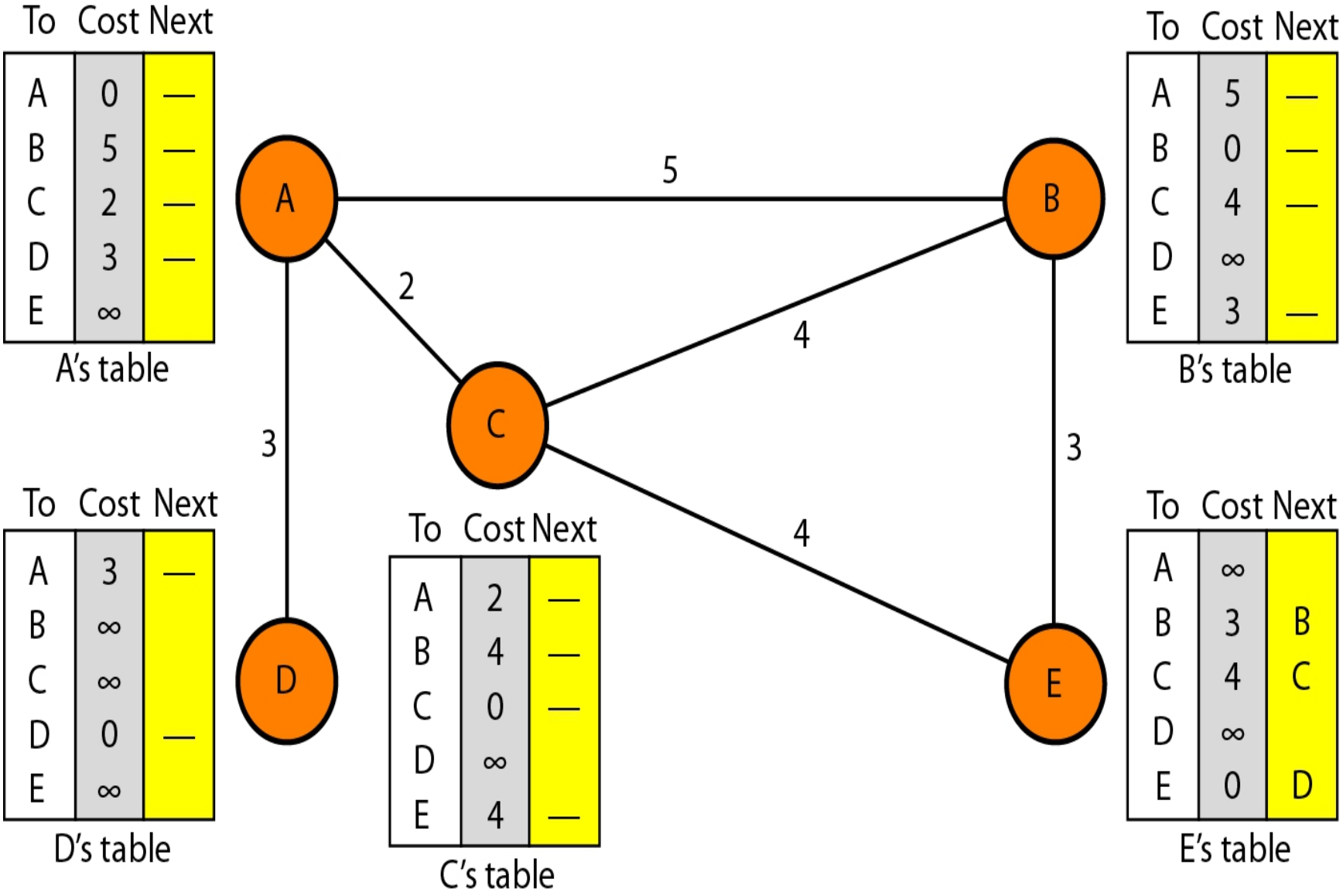
- **Distance Vector Routing**
  - In distance vector routing, the least-cost route between any two nodes is the route with minimum distance.
  - In this protocol, as the name implies, each node maintains a vector (table) of minimum distances to every node.
  - The table at each node also guides the packets to the desired node by showing the next stop in the route (next-hop routing).
  - We can think of nodes as the cities in an area and the lines as the roads connecting them.
  - A table can show a tourist the minimum distance between cities.
  - In Figure 22.14, we show a system of five nodes with their corresponding tables.
  - The table for node A shows how we can reach any node from this node.
  - For example, our least cost to reach node E is 6.
  - The route passes through C.

Figure 22.14 *Distance vector routing tables*



- **Initialization**
  - The tables in Figure 22.14 are stable; each node knows how to reach any other node and the cost.
  - At the beginning, however, this is not the case.
  - Each node can know only the distance between itself and its immediate neighbors, those directly connected to it.
  - So for the moment, we assume that each node can send a message to the immediate neighbors and find the distance between itself and these neighbors.
  - Figure 22.15 shows the initial tables for each node.
  - The distance for any entry that is not a neighbor is marked as infinite (unreachable).

Figure 22.15 *Initialization of tables in distance vector routing*



- **Sharing**
  - The whole idea of distance vector routing is the sharing of information between neighbors.
  - Although node A does not know about node E, node C does.
  - So if node C shares its routing table with A, node A can also know how to reach node E.
  - On the other hand, node C does not know how to reach node D, but node A does.
  - If node A shares its routing table with node C, node C also knows how to reach node D.
  - In other words, nodes A and C, as immediate neighbors, can improve their routing tables if they help each other.
- **Updating**
  - When a node receives a two-column table from a neighbor, it needs to update its routing table.
  - Updating takes three steps:

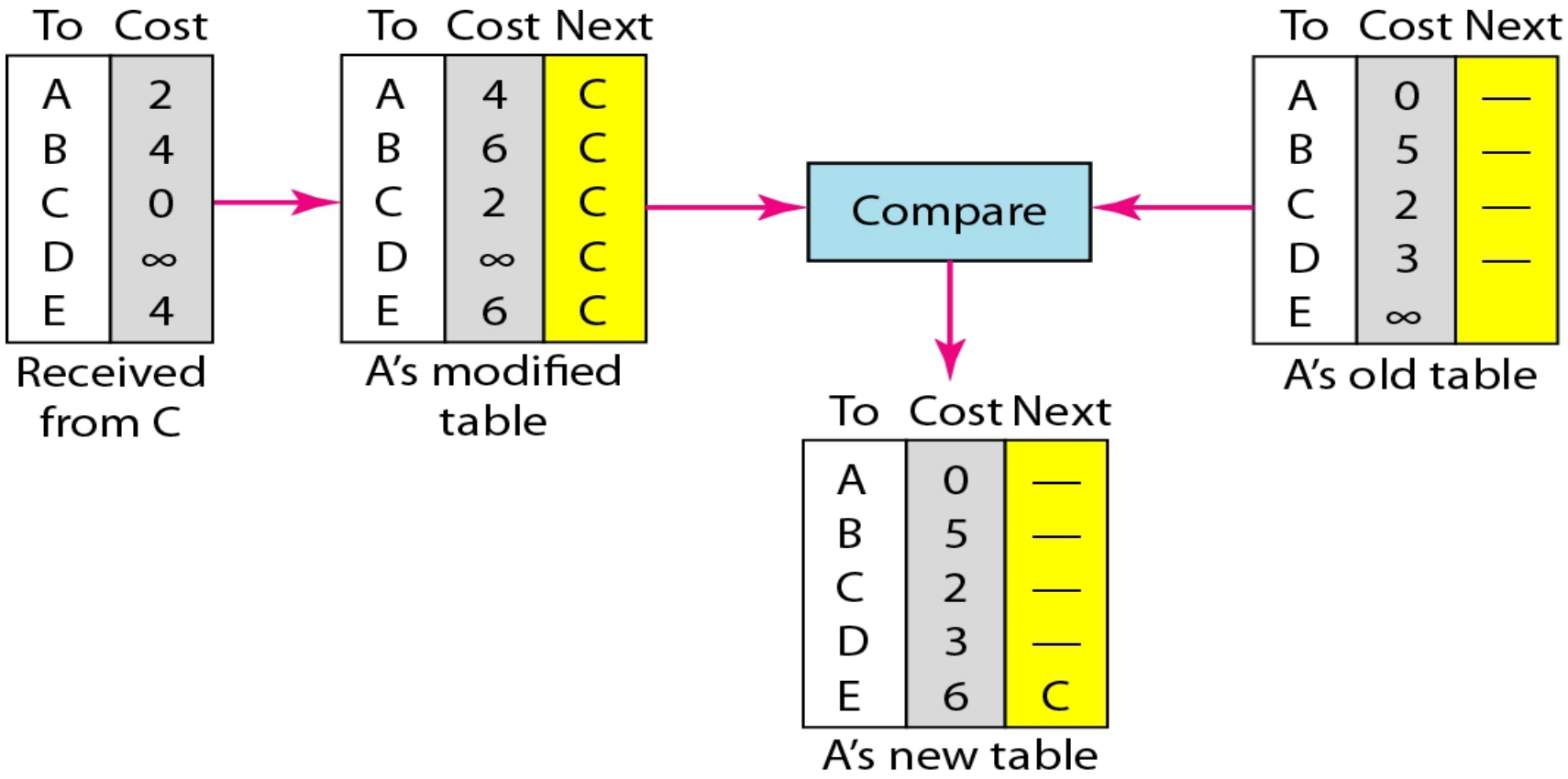
- **1.**The receiving node needs to add the cost between itself and the sending node to each value in the second column.
- The logic is clear. If node C claims that its distance to a destination is x mi, and the distance between A and C is y mi, then the distance between A and that destination, via C, is x + y mi.
- **2.** The receiving node needs to add the name of the sending node to each row as the third column if the receiving node uses information from any row.
- The sending node is the next node in the route.
- **3**. The receiving node needs to compare each row of its old table with the corresponding row of the modified version of the received table.
  - a. If the next-node entry is different, the receiving node chooses the row with the smaller cost.
  - If there is a tie, the old one is kept.
  - b. If the next-node entry is the same, the receiving node chooses the new row.
  - For example, suppose node C has previously advertised a route to node X with distance

3. Suppose that now there is no path between C and X; node C now advertises this route with a distance of infinity.

Node A must not ignore this value even though its old entry is smaller.

The old route does not exist any more.

The new route has a distance of infinity.



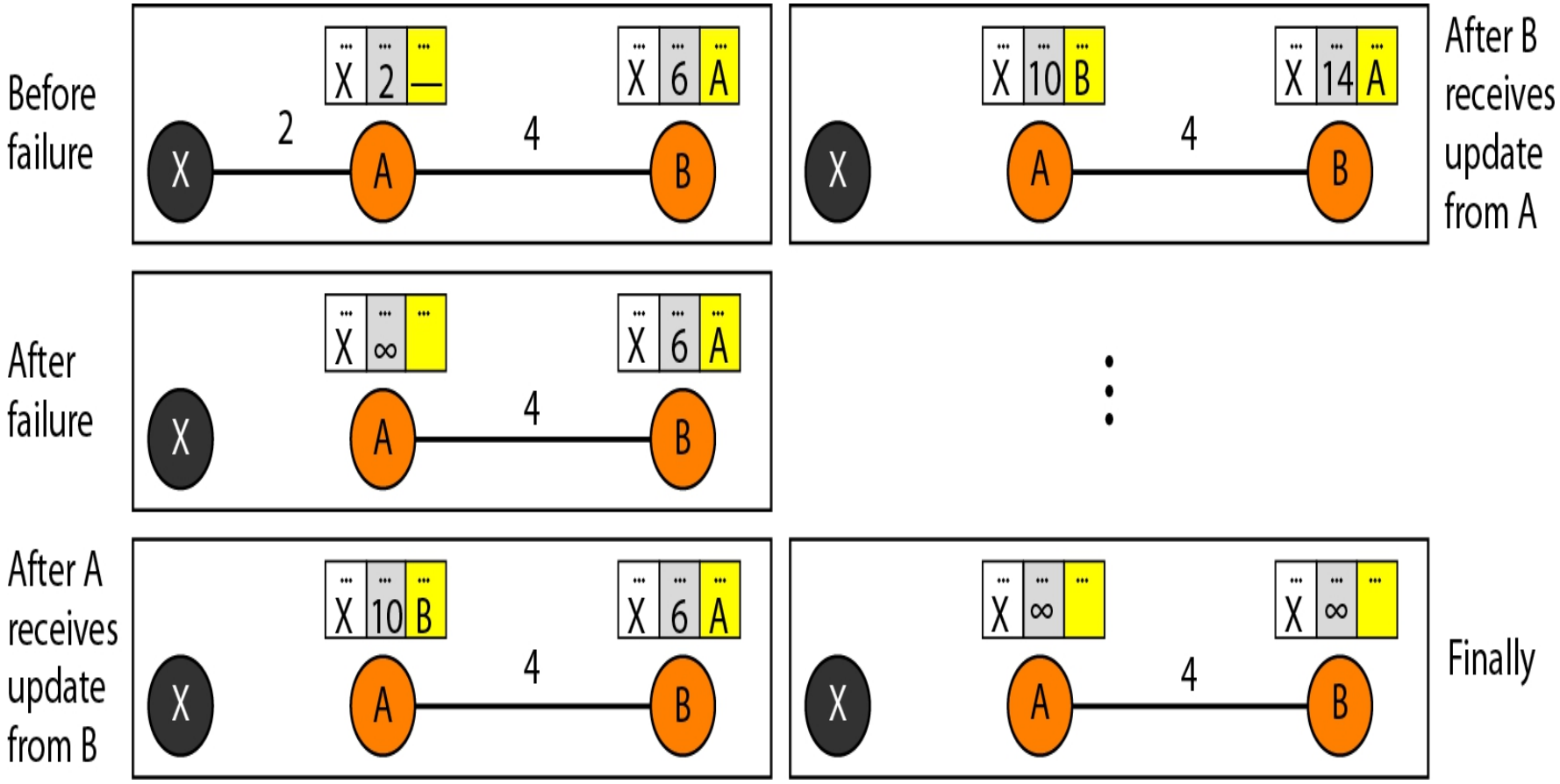
- There are several points we need to emphasize here.
- First, as we know from mathematics, when we add any number to infinity, the result is still infinity.
- Second, the modified table shows how to reach E from A via C.
- If A needs to reach itself via C, it needs to go to C and come back, a distance of 4.
- Third, the only benefit from this updating of node A is the last entry, how to reach E.
- Previously, node A did not know how to reach E (distance of infinity); now it knows that the cost is 6 via C.

- **When to Share**
  - The question now is, When does a node send its partial routing table (only two columns) to all its immediate neighbors?
  - The table is sent both periodically and when there is a change in the table.
  - **Periodic Update**
  - A node sends its routing table, normally every 30 s, in a periodic update.
  - The period depends on the protocol that is using distance vector routing.
  - **Triggered Update**
  - A node sends its two-column routing table to its neighbors any- time there is a change in its routing table.
  - This is called a triggered update.
  - The change can result from the following.
    - 1. A node receives a table from a neighbor, resulting in changes in its own table after updating.
    - 2. A node detects some failure in the neighboring links which results in a distance change to infinity.

- **Two-Node Loop Instability**

  A problem with distance vector routing is instability, which means that a network using this protocol can become unstable.

  To understand the problem, let us look at the scenario depicted in Figure 22.17.



- Figure 22.17 shows a system with three nodes.
- We have shown only the portions of the routing table needed for our discussion.
- At the beginning, both nodes A and B know how to reach node X.
- But suddenly, the link between A and X fails.
- Node A changes its table.
- If A can send its table to B immediately, everything is fine.
- However, the system becomes unstable if B sends its routing table to A before receiving A's routing table.
- Node A receives the update and, assuming that B has found a way to reach X, immediately updates its routing table.
- Based on the triggered update strategy, A sends its new update to B.

- Now B thinks that something has been changed around A and updates its routing table.
- The cost of reaching X increases gradually until it reaches infinity.
- At this moment, both A and B know that X cannot be reached.
- However, during this time the system is not stable.
- Node A thinks that the route to X is via B; node B thinks that the route to X is via A.
- If A receives a packet destined for X, it goes to B and then comes back to A.
- Similarly, if B receives a packet destined for X, it goes to A and comes back to B.
- Packets bounce between A and B, creating a two-node loop problem.
- A few solu- tions have been proposed for instability of this kind.

- **Defining Infinity**
  - The first obvious solution is to redefine infinity to a smaller number, such as 100.
  - For our previous scenario, the system will be stable in less than 20 updates.
  - As a matter of fact, most implementations of the distance vector protocol define the distance between each node to be I and define 16 as infinity.
  - However, this means that the distance vector routing cannot be used in large systems.
  - The size of the network, in each direction, can not exceed 15 hops.

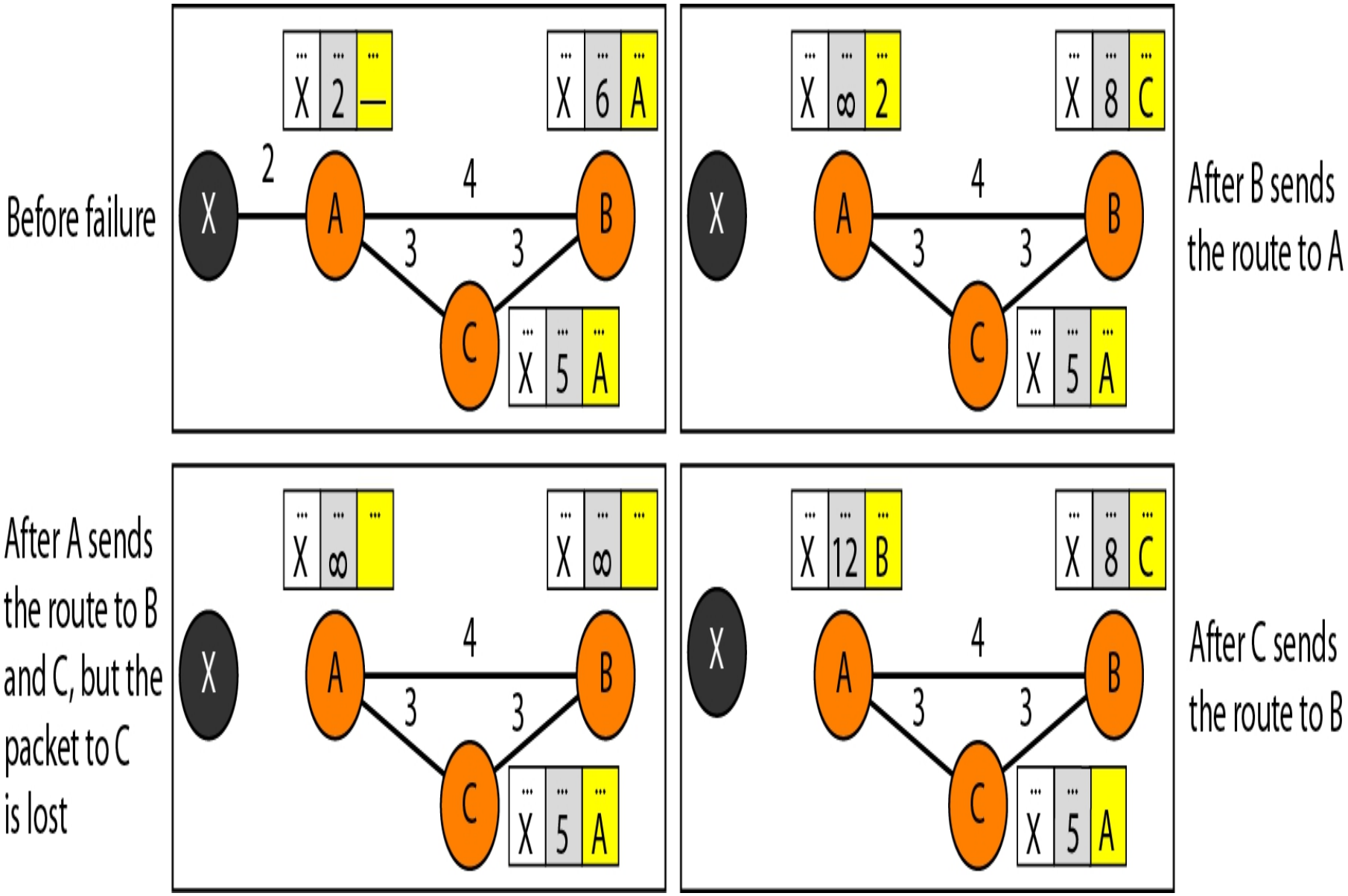
- **Split Horizon**
  - Another solution is called split horizon.
  - In this strategy, instead of flooding the table through each interface, each node sends only part of its table through each interface.
  - If, according to its table, node B thinks that the optimum route to reach X is via A, it does not need to advertise this piece of information to A; the information has corne from A (A already knows).
  - Taking information from node A, modifying it, and sending it back to node A creates the confusion.
  - In our scenario, node B eliminates the last line of its routing table before it sends it to A.
  - In this case, node A keeps the value of infinity as the distance to X.
  - Later when node A sends its routing table to B, node B also corrects its routing table.
  - The system becomes stable after the first update: both node A and B know that X is not reachable.

- **Split Horizon and Poison Reverse**
  - Using the split horizon strategy has one drawback.
  - Normally, the distance vector protocol uses a timer, and if there is no news about a route, the node deletes the route from its table.
  - When node B in the previous scenario eliminates the route to X from its advertisement to A, node A cannot guess that this is due to the split horizon strategy (the source of information was A) or because B has not received any news about X recently.
  - The split horizon strategy can be combined with the poison reverse strategy.
  - Node B can still advertise the value for X, but if the source of information is A, it can replace the distance with infinity as a warning:
  - "Do not use this value; what I know about this route comes from you."

- **Three-Node Instability**
  - The two-node instability can be avoided by using the split horizon strategy combined with poison reverse.
  - However, if the instability is between three nodes, stability cannot be guaranteed.
  - Figure shows the scenario.
  - Suppose, after finding that X is not reachable, node A sends a packet to B and C to inform them of the situation.
  - Node B immediately updates its table, but the packet to C is lost in the network and never reaches C.
  - Node C remains in the dark and still thinks that there is a route to X via A with a distance of 5.
  - After a while, node C sends to B its routing table, which includes the route to X.
  - **Node B is totally fooled here.**
  - It receives information on the route to X from C, and according to the algorithm, it updates its table, showing the route to X via C with a cost of 8.

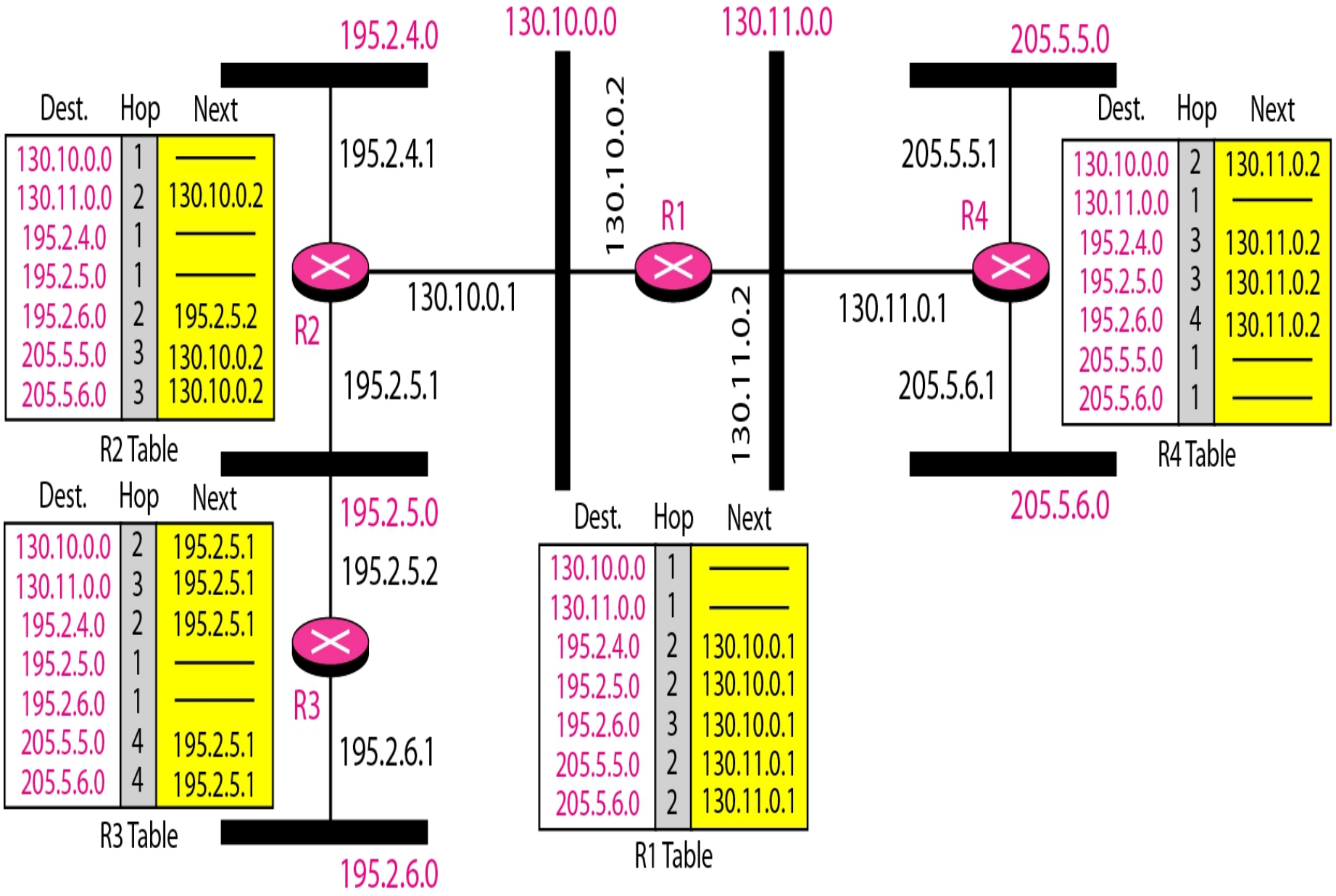
- This information has come from C, not from A, so after a while node B may advertise this route to A.
- **Now A is fooled** and updates its table to show that A can reach X via B with a cost of 12.
- Of course, the loop continues; now A advertises the route to X to C, with increased cost, but not to B.
- Node C then advertises the route to B with an increased cost.
- Node B does the same to A.
- And so on.
- The loop stops when the cost in each node reaches infinity.

Figure 22.18 *Three-node instability*



- **RIP**
  - The Routing Information Protocol (RIP) is an intradomain routing protocol used inside an autonomous system.
  - It is a very simple protocol based on distance vector routing.
  - RIP implements distance vector routing directly with some considerations:
    - 1. In an autonomous system, we are dealing with routers and networks (links).
    - The routers have routing tables; networks do not.
    - 2. The destination in a routing table is a network, which means the first column defines a network address.
    - 3. The metric used by RIP is very simple; the distance is defined as the number of links (networks) to reach the destination.
    - For this reason, the metric in RIP is called a hop count.
    - 4. Infinity is defined as 16, which means that any route in an autonomous system using RIP cannot have more than 15 hops.
    - 5. The next-node column defines the address of the router to which the packet is to be sent to reach its destination.

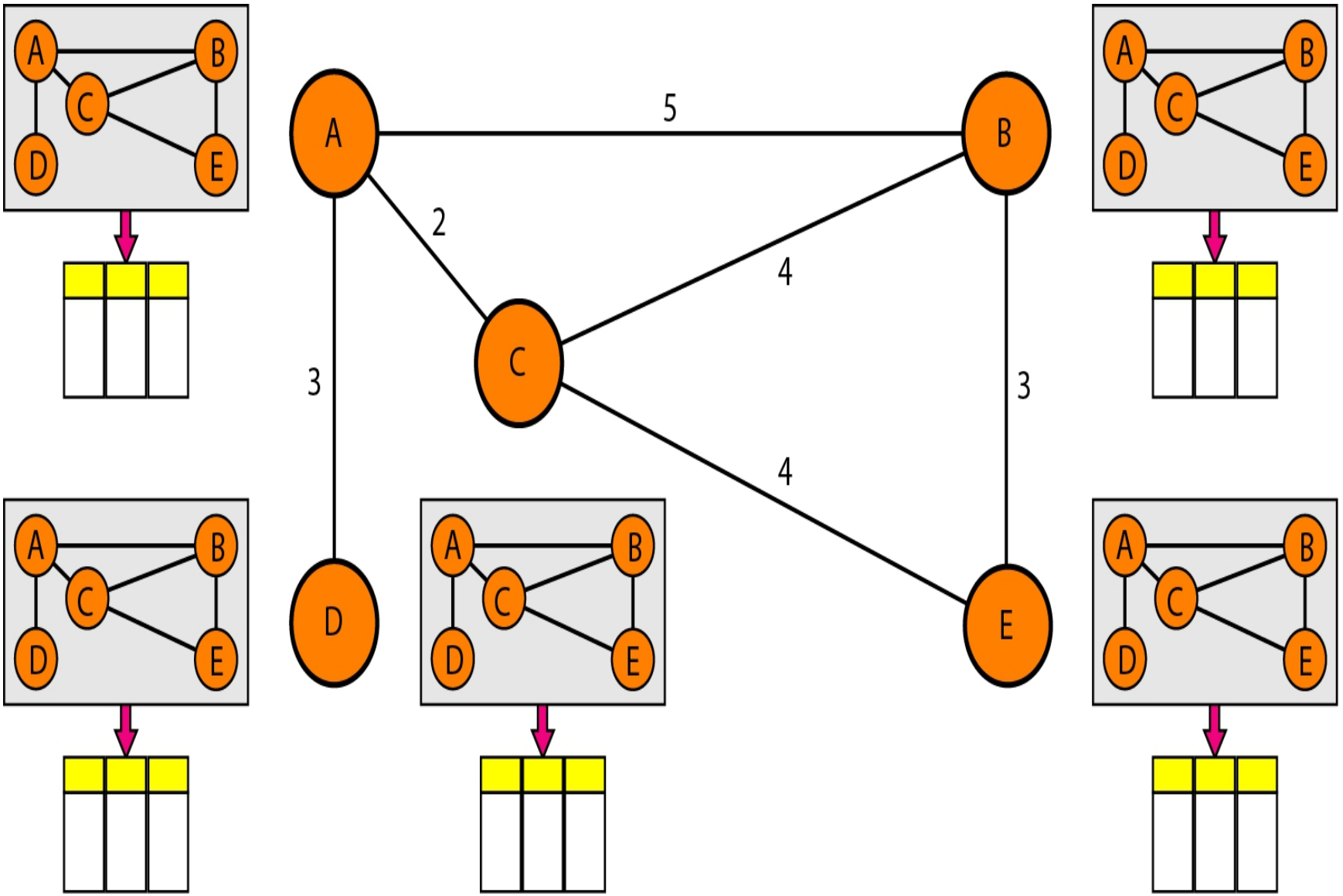
Figure 22.19 *Example of a domain using RIP*



- Figure shows an autonomous system with seven networks and four routers.
- The table of each router is also shown.
- Let us look at the routing table for Rl.
- The table has seven entries to show how to reach each network in the autonomous system.
- Router Rl is directly connected to networks 130.10.0.0 and 130.11.0.0, which means that there are no next-hop entries for these two networks.
- To send a packet to one of the three networks at the far left, router Rl needs to deliver the packet to R2.
- The next-node entry for these three networks is the interface of router R2 with IP address 130.10.0.1.
- To send a packet to the two networks at the far right, router Rl needs to send the packet to the interface of router R4 with IP address 130.11.0.1.
- The other tables can be explained similarly.

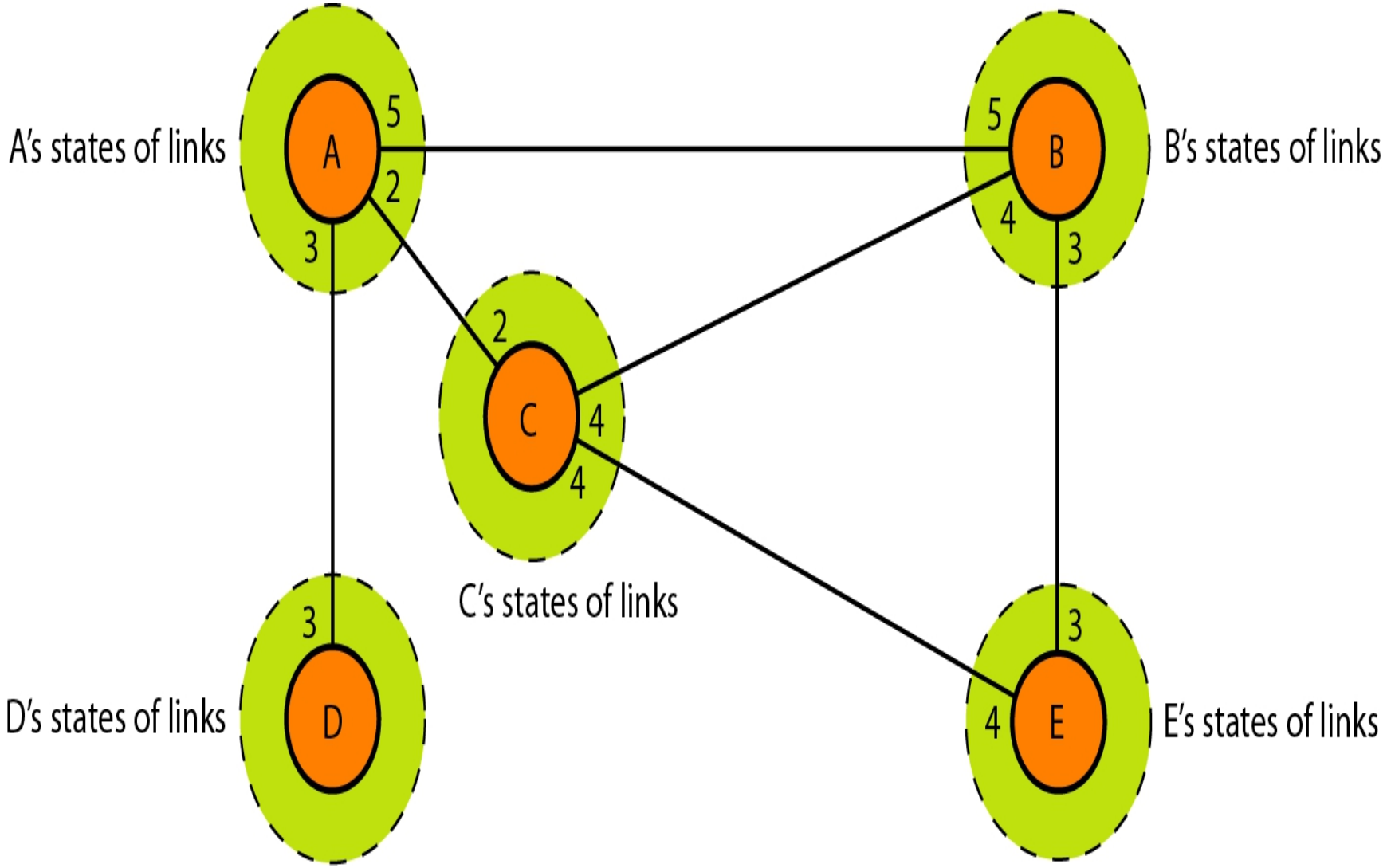
- **Link State Routing**
  - Link state routing has a different philosophy from that of distance vector routing.
  - In link state routing, if each node in the domain has the entire topology of the domain- the list of nodes and links, how they are connected including the type, cost (metric), and condition of the links (up or down)-the node can use Dijkstra's algorithm to build a routing table.
  - Figure shows the concept.

Figure 22.20 *Concept of link state routing*



- The topology must be dynamic, representing the latest state of each node and each link.
- If there are changes in any point in the network (a link is down, for example), the topology must be updated for each node.
- How can a common topology be dynamic and stored in each node? No node can know the topology at the beginning or after a change somewhere in the network.
- Link state routing is based on the assumption that, although the global knowledge about the topology is not clear, each node has partial knowledge: it knows the state (type, condi- tion, and cost) of its links.
- In other words, the whole topology can be compiled from the partial knowledge of each node.
- Figure shows the same domain as in above Figure , indicating the part of the knowledge belonging to each node.

Figure 22.21 *Link state knowledge*



- Node A knows that it is connected to node B with metric 5, to node C with metric 2, and to node D with metric 3.
- Node C knows that it is connected to node A with metric 2, to node B with metric 4, and to node E with metric 4.
- Node D knows that it is connected only to node A with metric 3.
- And so on.
- Although there is an overlap in the knowledge, the overlap guarantees the creation of a common topology-a picture of the whole domain for each node.

- **Building Routing Tables**
  - In link state routing, four sets of actions are required to ensure that each node has the routing table showing the least-cost node to every other node.
  - 1. Creation of the states of the links by each node, called the link state packet (LSP).
  - 2. Dissemination of LSPs to every other router, called flooding, in an efficient and reliable way.
  - 3. Formation of a shortest path tree for each node.
  - 4. Calculation of a routing table based on the shortest path tree.

- **Creation of Link State Packet (LSP)**
  - A link state packet can carry a large amount of information.
  - For the moment, however, we assume that it carries a minimum amount of data:
    - the node identity,
    - the list of links,
    - a sequence number, and
    - age.
  - The first two, node identity and the list of links, are needed to make the topology.
  - The third, sequence number, facilitates flooding and distinguishes new LSPs from old ones.
  - The fourth, age, prevents old LSPs from remaining in the domain for a long time.

- LSPs are generated on two occasions:
  - 1. When there is a change in the topology of the domain.
  - 2. On a periodic basis.
- As a matter of fact, there is no actual need for this type of LSP dissemination.
- It is done to ensure that old information is removed from the domain.
- The timer set for periodic dissemination is normally in the range of 60 min or 2 h based on the implementation.
- A longer period ensures that flooding does not create too much traffic on the network.

- **Flooding of LSPs**
  - After a node has prepared an LSP, it must be disseminated to all other nodes, not only to its neighbors.
  - The process is called flooding and based on the following:
    - 1. The creating node sends a copy of the LSP out of each interface.
    - 2. A node that receives an LSP compares it with the copy it may already have.
    - If the newly arrived LSP is older than the one it has (found by checking the sequence number), it discards the LSP.
    - If it is newer, the node does the following:
      - a. It discards the old LSP and keeps the new one.
      - b. It sends a copy of it out of each interface except the one from which the packet arrived.
      - This guarantees that flooding stops somewhere in the domain (where a node has only one interface).

- **Formation of Shortest Path Tree:**
  - Dijkstra Algorithm After receiving all LSPs, each node will have a copy of the whole topology.
  - However, the topology is not sufficient to find the shortest path to every other node; a shortest path tree is needed.
  - A tree is a graph of nodes and links; one node is called the root.
  - All other nodes can be reached from the root through only one single route.
  - A shortest path tree is a tree in which the path between the root and every other node is the shortest.
  - What we need for each node is a shortest path tree with that node as the root.
  - The Dijkstra algorithm creates a shortest path tree from a graph.
  - The algorithm divides the nodes into two sets: tentative and permanent.
  - It finds the neighbors of a current node, makes them tentative, examines them, and if they pass the criteria, makes them permanent.
  - We can informally define the algorithm by using the flowchart in Figure .

Figure 22.22 *Dijkstra algorithm*

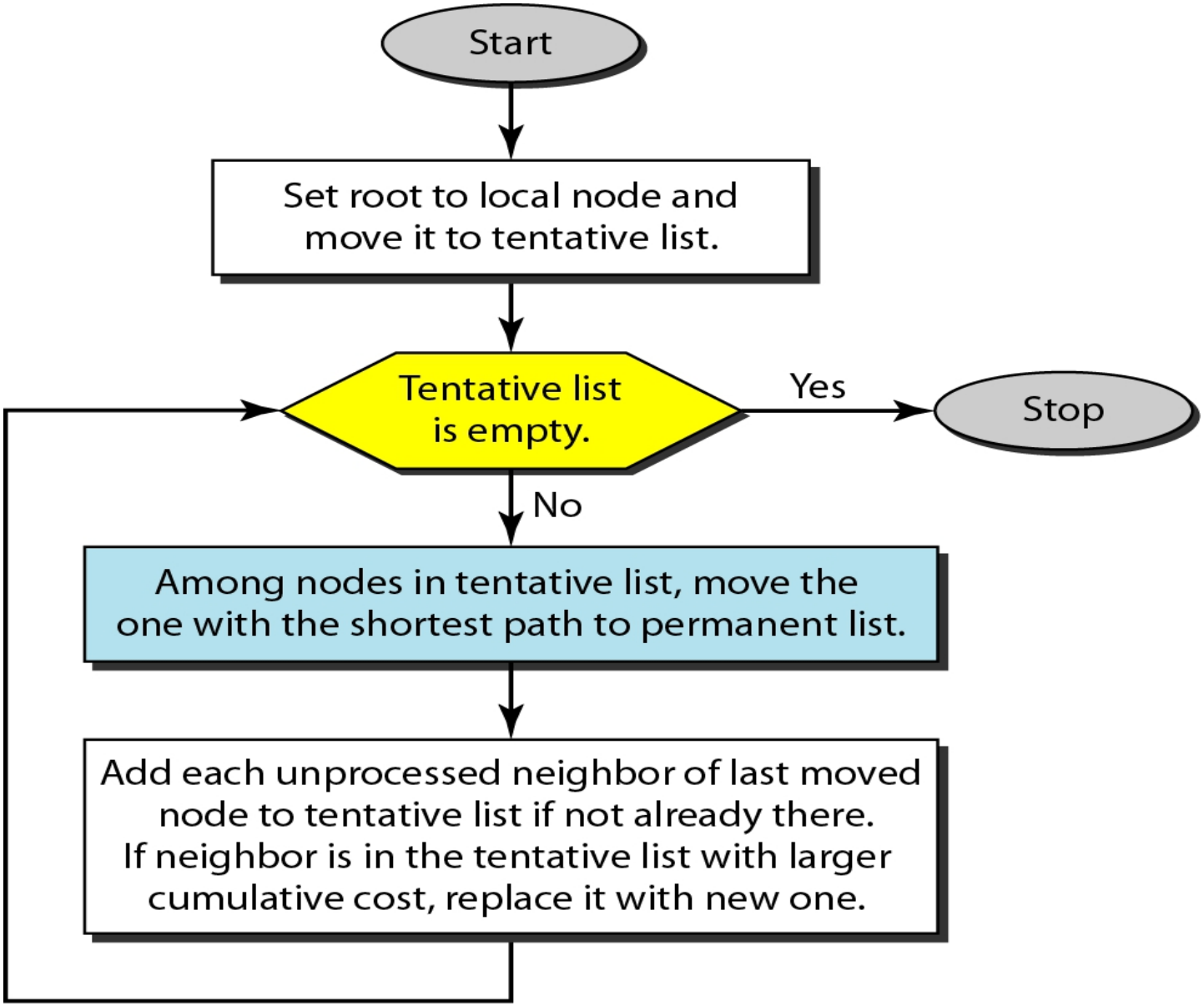
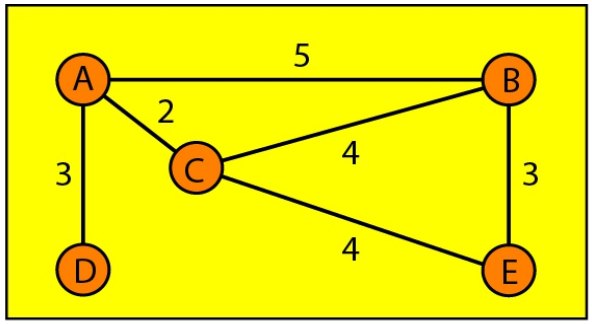
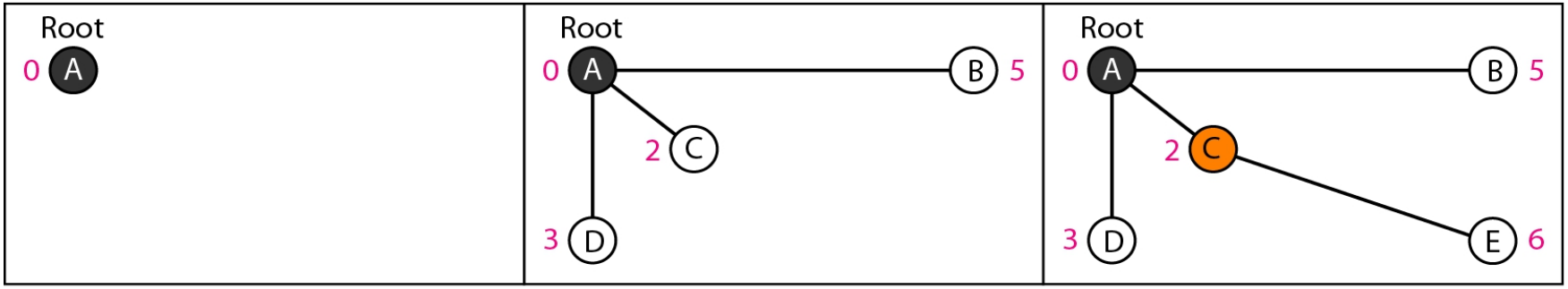


Figure 22.23 Example of formation of shortest path tree



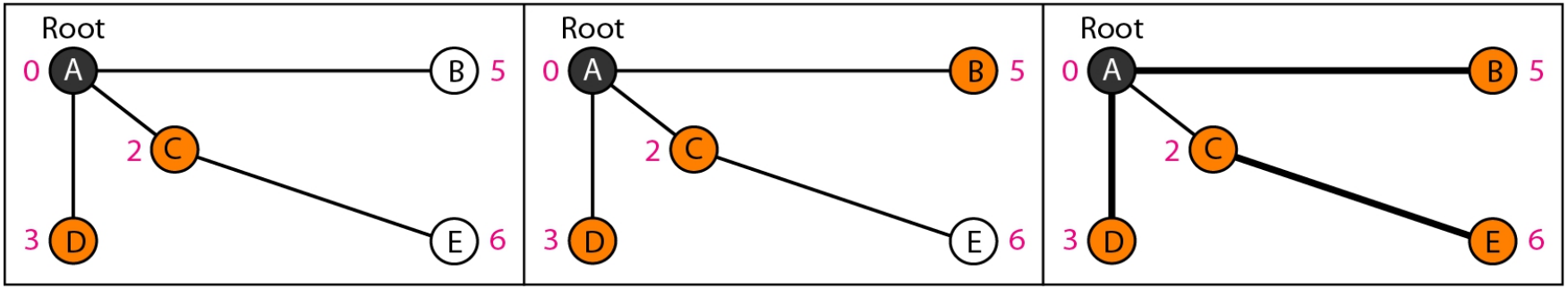
Topology



1. Set root to A and move A to tentative list.

2. Move A to permanent list and add B, C, and D to tentative list.

3. Move C to permanent and add E to tentative list.



4. Move D to permanent list.

5. Move B to permanent list.

6. Move E to permanent list (tentative list is empty).

1. We make node A the root of the tree and move it to the tentative list. Our two lists are

   Permanent list: empty Tentative list: A(O)

2. Node A has the shortest cumulative cost from all nodes in the tentative list. We move A to the pennanent list and add all neighbors of A to the tentative list. Our new lists are

   Permanent list: A(O) Tentative list: B(5), C(2), D(3)

3. Node C has the shortest cumulative cost from all nodes in the tentative list. We move C to the permanent list. Node C has three neighbors, but node A is already processed, which makes the unprocessed neighbors just B and E. However, B is already in the tentative list with a cumulative cost of 5. Node A could also reach node B through C with a cumulative cost of 6. Since 5 is less than 6, we keep node B with a cumulative cost of 5 in the tentative list and do not replace it. Our new lists are

   Permanent list: A(O), e(2) Tentative list: B(5), 0(3), E(6)

4. Node D has the shortest cumulative cost of all the nodes in the tentative list. We move D to the permanent list. Node D has no unprocessed neighbor to be added to the tentative list. Our new lists are

   Permanent list: A(O), C(2), 0(3) Tentative list: B(5), E(6)

5. Node B has the shortest cumulative cost of all the nodes in the tentative list. We move B to the permanent list. We need to add all unprocessed neighbors of B to the tentative list (this is just node E). However, E(6) is already in the list with a smaller cumulative cost. The cumulative cost to node E, as the neighbor of B, is 8. We keep node E(6) in the tentative list. Our new lists are

Permanent list: A(0), B(5), C(2), 0(3) Tentative list: E(6)

6. Node E has the shortest cumulative cost from all nodes in the tentative list. We move E to the permanent list. Node E has no neighbor. Now the tentative list is empty. We stop; our shortest path tree is ready. The final lists are

Permanent list: A(0), B(5), C(2), D(3), E(6) Tentative list: empty

- **Calculation of Routing**

  Table from Shortest Path Tree Each node uses the shortest path tree protocol to construct its routing table.

  The routing table shows the cost of reaching each node from the root.

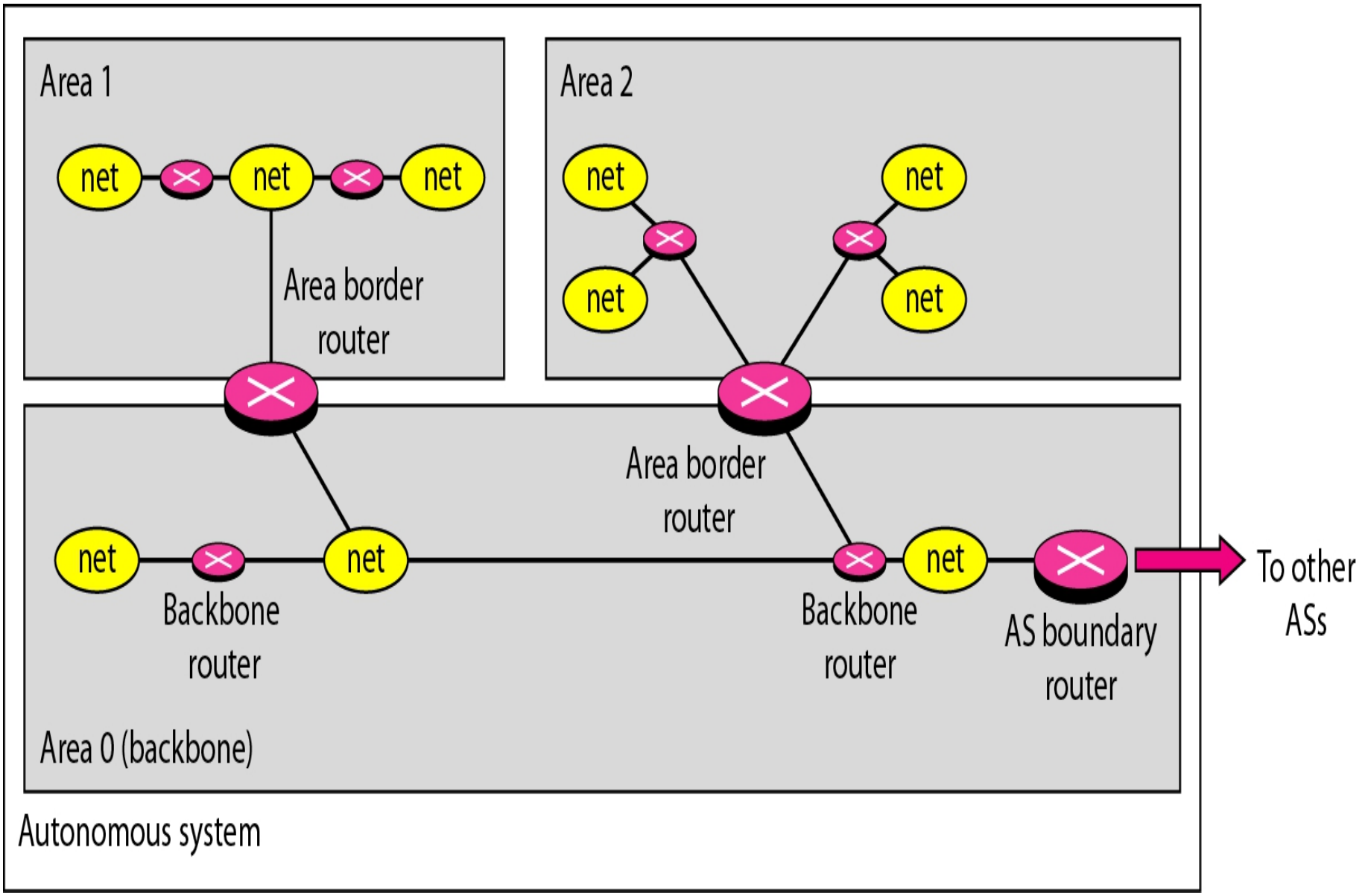
  Table 22.2 shows the routing table for node A.

Table 22.2 *Routing table for node A*

Node	*Cost*	*Next Router*
A	0	-
B	5	-
C	2	-
D	3	-
E	6	C

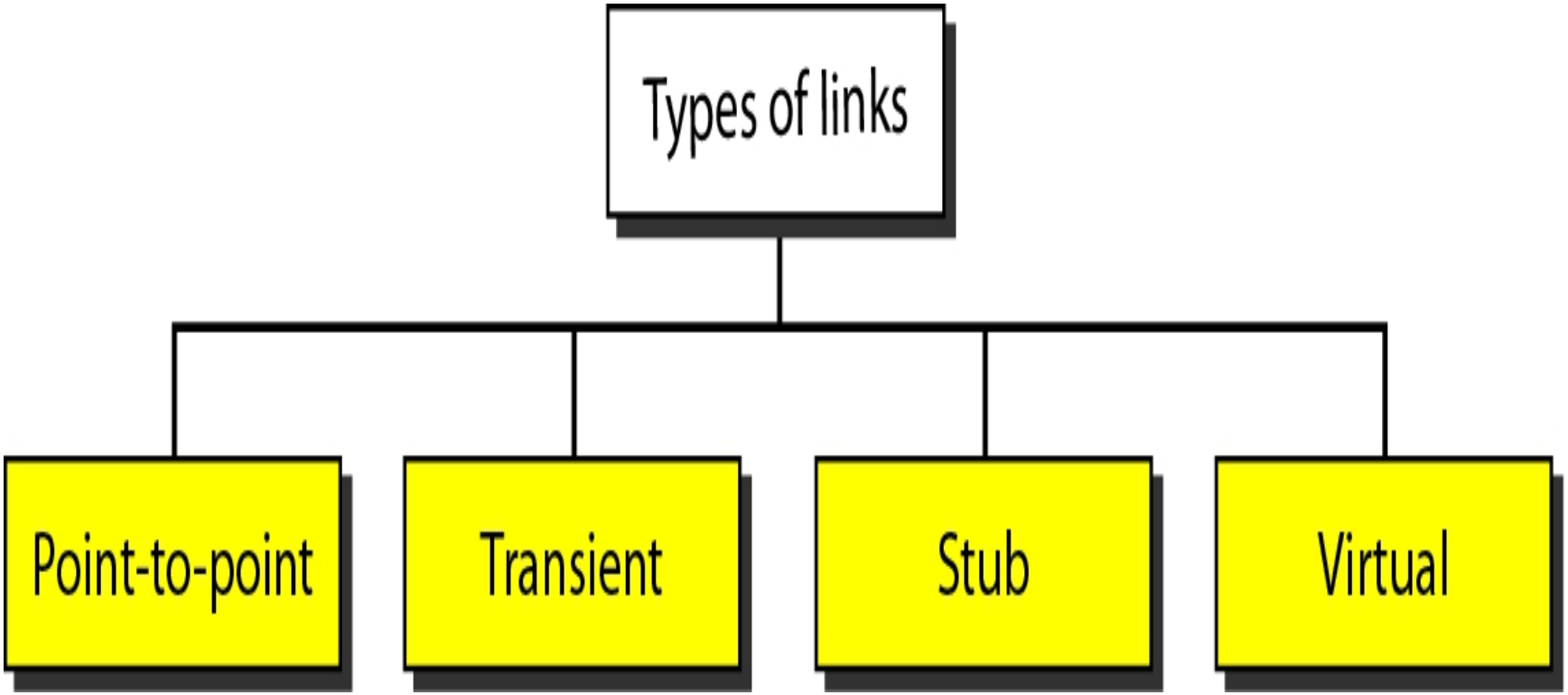
- **OSPF**
  - The Open Shortest Path First or OSPF protocol is an intradomain routing protocol based on link state routing.
  - Its domain is also an autonomous system.
- **Areas**
  - To handle routing efficiently and in a timely manner, OSPF divides an autonomous system into areas.
  - An area is a collection of networks, hosts, and routers all contained within an autonomous system.
  - An autonomous system can be divided into many different areas.
  - All networks inside an area must be connected.

Figure 22.24 ***Areas in an autonomous system***



- **Metric**
  - The OSPF protocol allows the administrator to assign a cost, called the metric, to each route.
  - The metric can be based on a type of service (minimum delay, maximum throughput, and so on).
  - As a matter of fact, a router can have multiple routing tables, each based on a different type of service.
- **Types of Links**
  - In OSPF terminology, a connection is called a link.
  - Four types of links have been defined:
    - point-to-point,
    - transient,
    - stub, and
    - virtual.

Figure 22.25 Types of links



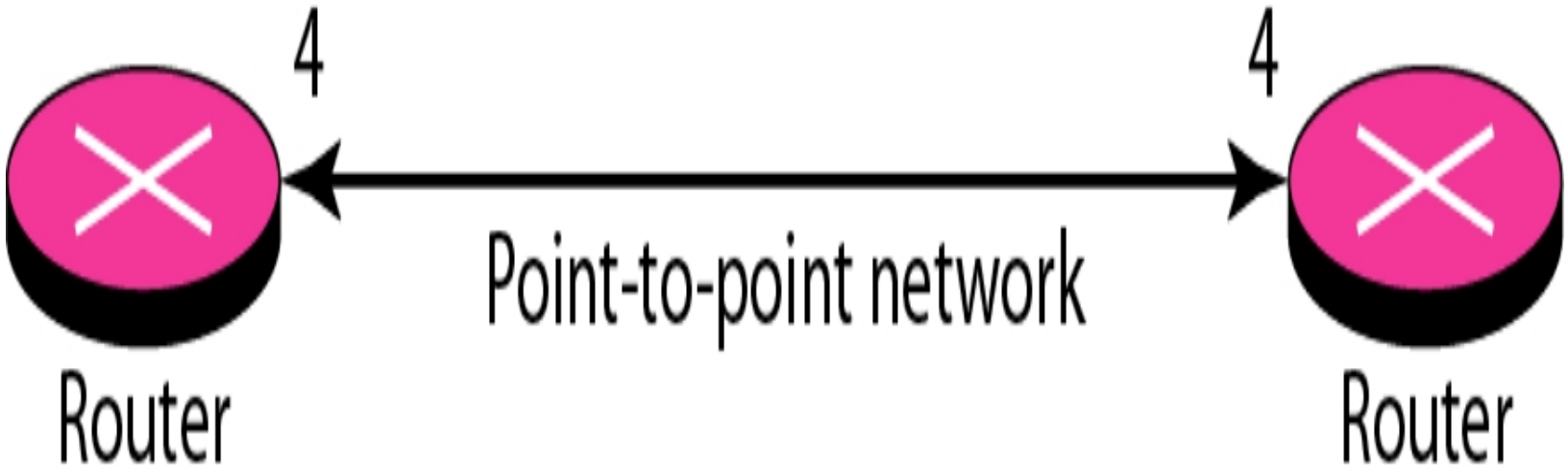
- **A point-to-point link**

  A point-to-point link connects two routers without any other host or router in between.

  In other words, the purpose of the link (network) is just to connect the two routers.

  An example of this type of link is two routers connected by a telephone line or a T line.

  The metrics, which are usually the same, are shown at the two ends, one for each direction.



- **A transient link**

  A transient link is a network with several routers attached to it.

  The data can enter through any of the routers and leave through any router.

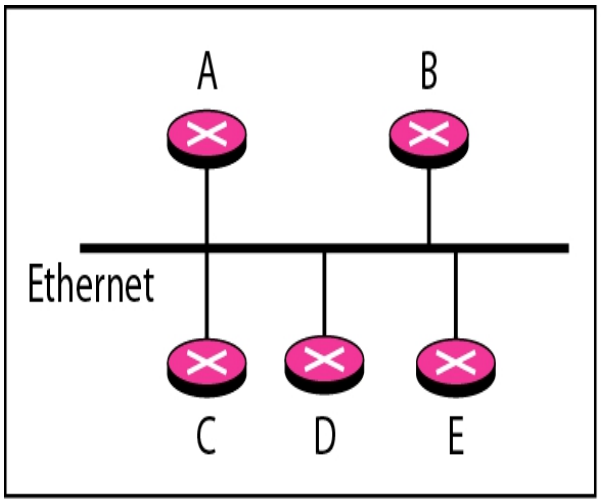
  All LANs and some WANs with two or more routers are of this type.

  In this case, each router has many neighbors.

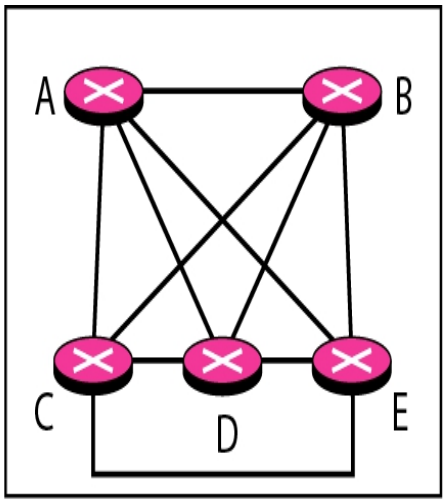
  This is neither efficient nor realistic.

  It is not efficient because each router needs to advertise the neighborhood to four other routers, for a total of 20 advertisements.

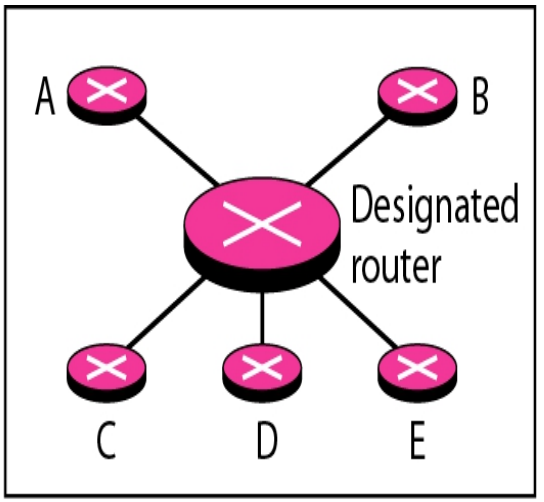
  It is not realistic because there is no single network (link) between each pair of routers; there is only one network that serves as a crossroad between all five routers.



a. Transient network



b. Unrealistic representation



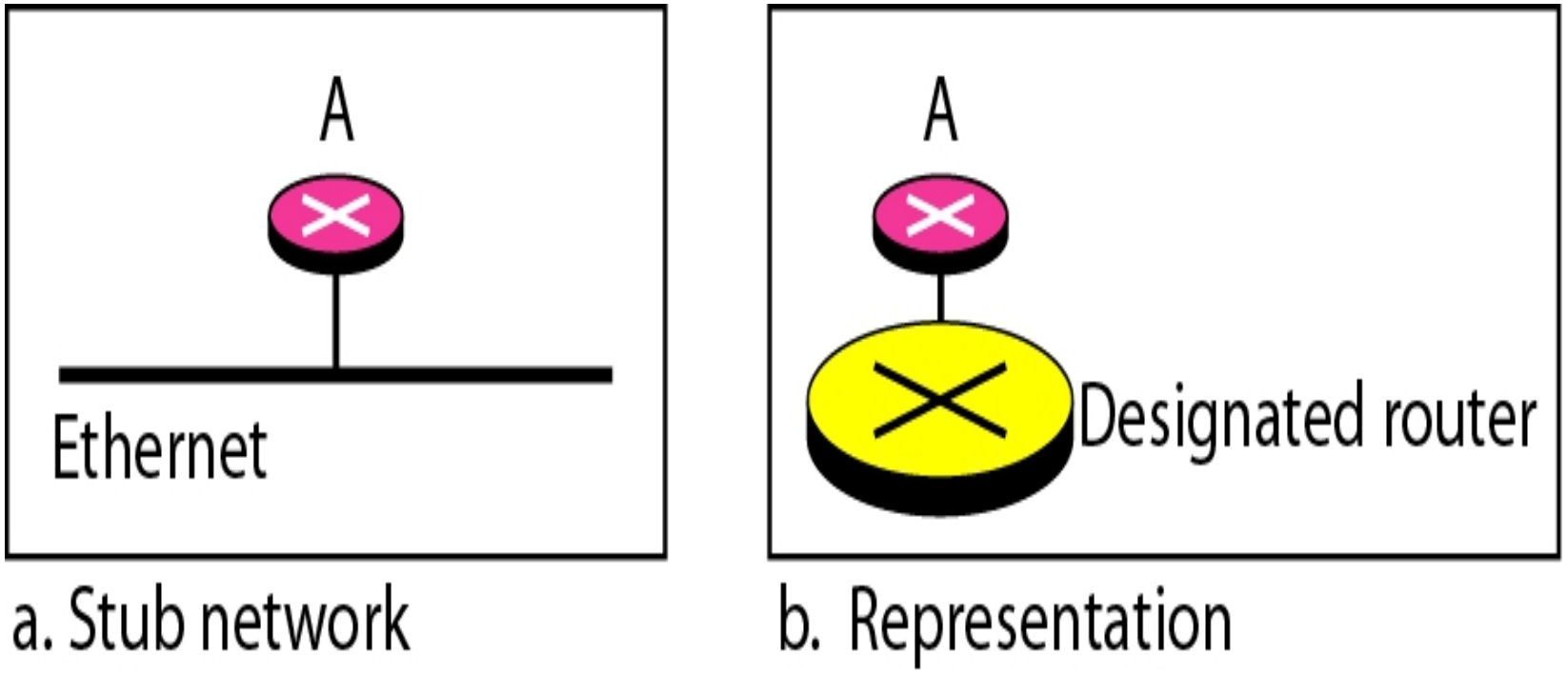
c. Realistic representation

- **A stub link**

  stub link is a network that is connected to only one router.

  The data packets enter the network through this single router and leave the network through this same router.

  This is a special case of the transient network.



a. Stub network

b. Representation

- **Virtual Link**
  - When the link between two routers is broken, the administration may create a virtual link between them, using a longer path that probably goes through several routers.

- **Congestion Control Algorithms:**
  - –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).
  - –**Open-Loop Congestion Control**
  - –In open-loop congestion control, policies are applied to prevent congestion before it happens.
  - –In these mechanisms, congestion control is handled by either the source or the destination.
  - –We give a brief list of policies that can prevent congestion.

- **Retransmission Policy**
- Retransmission is sometimes unavoidable.
- If the sender feels that a sent packet is lost or corrupted, the packet needs to be retransmitted.
- Retransmission in general may increase congestion in the network.
- However, a good retransmission policy can prevent congestion.
- The retransmission policy and the retransmission timers must be designed to optimize efficiency and at the same time prevent congestion.

- **Window Policy**
- The type of window at the sender may also affect congestion.
- The Selective Repeat window is better than the Go-Back-N window for congestion control.
- In the Go-Back-N window, when the timer for a packet times out, several packets may be resent, although some may have arrived safe and sound at the receiver.
- This duplication may make the congestion worse.
- The Selective Repeat window, on the other hand, tries to send the specific packets that have been lost or corrupted.

- **Acknowledgment Policy**
- The acknowledgment policy imposed by the receiver may also affect congestion.
- If the receiver does not acknowledge every packet it receives, it may slow down the sender and help prevent congestion.
- Several approaches are used in this case.
- A receiver may send an acknowledgment only if it has a packet to be sent or a special timer expires.
- A receiver may decide to acknowledge only N packets at a time.

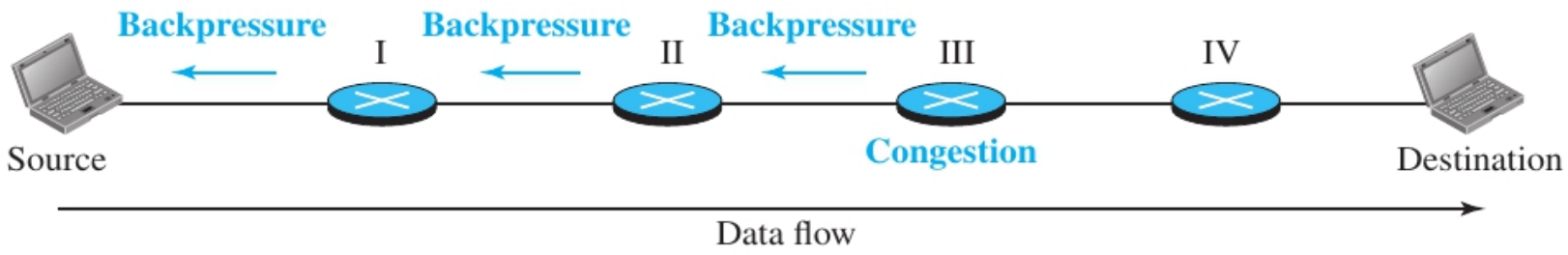
- **Discarding Policy**
- A good discarding policy by the routers may prevent congestion and at the same time may not harm the integrity of the transmission.
- For example, in audio transmission, if the policy is to discard less sensitive packets when congestion is likely to happen, the quality of sound is still preserved and congestion is prevented or alleviated.

- **Admission Policy**
- An admission policy, which is a quality-of-service mechanism can also prevent congestion in virtual-circuit networks.
- Switches in a flow first check the resource requirement of a flow before admitting it to the network.
- A router can deny establishing a virtual-circuit connection if there is congestion in the network or if there is a possibility of future congestion.

- **Closed-Loop Congestion Control**
- Closed-loop congestion control mechanisms try to alleviate congestion after it happens.
- Several mechanisms have been used by different protocols.
- We describe a few of them here.
- **Backpressure**
- The technique of backpressure refers to a congestion control mechanism in which a congested node stops receiving data from the immediate upstream node or nodes.
- This may cause the upstream node or nodes to become congested, and they, in turn, reject data from their upstream node or nodes, and so on.

- Backpressure is a node-to-node congestion control that starts with a node and propagates, in the opposite direction of data flow, to the source.
- The backpressure technique can be applied only to virtual circuit networks, in which each node knows the upstream node from which a flow of data is coming.
- Figure 18.14 shows the idea of backpressure.

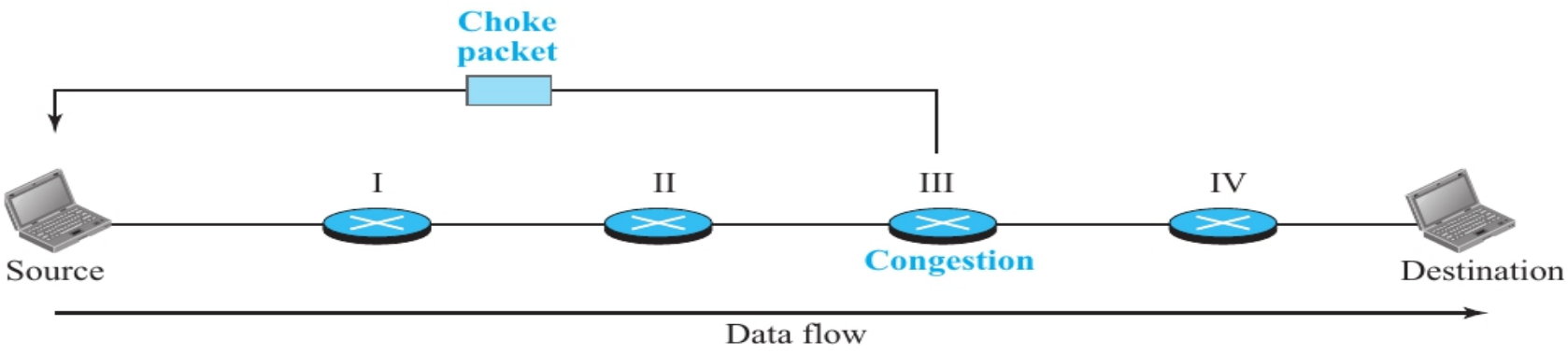
Figure 18.14 *Backpressure method for alleviating congestion*



- **Choke Packet**
- A choke packet is a packet sent by a node to the source to inform it of congestion.
- Note the difference between the backpressure and choke-packet methods.
- In backpressure, the warning is from one node to its upstream node, although the warning may eventually reach the source station.
- In the choke-packet method, the warning is from the router, which has encountered congestion, directly to the source station.
- The intermediate nodes through which the packet has traveled are not warned.

- **Choke Packet**
- When a router in the Internet is overwhelmed with IP datagrams, it may discard some of them, but it informs the source host, using a source quench ICMP message.
- The warning message goes directly to the source station; the intermediate routers do not take any action.
- Figure 18.15 shows the idea of a choke packet.

Figure 18.15 *Choke packet*

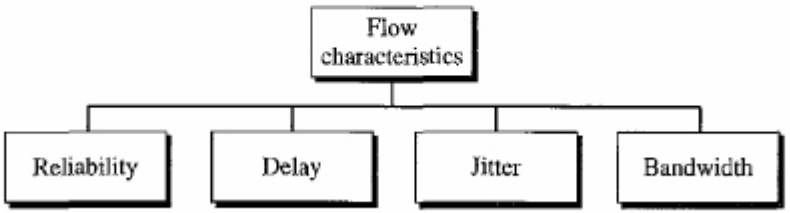


- **Implicit Signaling**
- In implicit signaling, there is no communication between the congested node or nodes and the source.
- The source guesses that there is congestion somewhere in the network from other symptoms.
- For example, when a source sends several packets and there is no acknowledgment for a while, one assumption is that the network is congested.
- The delay in receiving an acknowledgment is interpreted as congestion in the network; the source should slow down.

- **Explicit Signaling**
- The node that experiences congestion can explicitly send a signal to the source or destination.
- The explicit-signaling method, however, is different from the choke-packet method.
- In the choke-packet method, a separate packet is used for this purpose; in the explicit-signaling method, the signal is included in the packets that carry data.
- Explicit signaling can occur in either the forward or the backward direction.

- **QoS Parameters**
- 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.
- **Flow Characteristics**
- Traditionally, four types of characteristics are attributed to a flow: reliability, delay, jitter, and bandwidth, as shown in Figure 24.15.

Figure 24.15 Flow characteristics



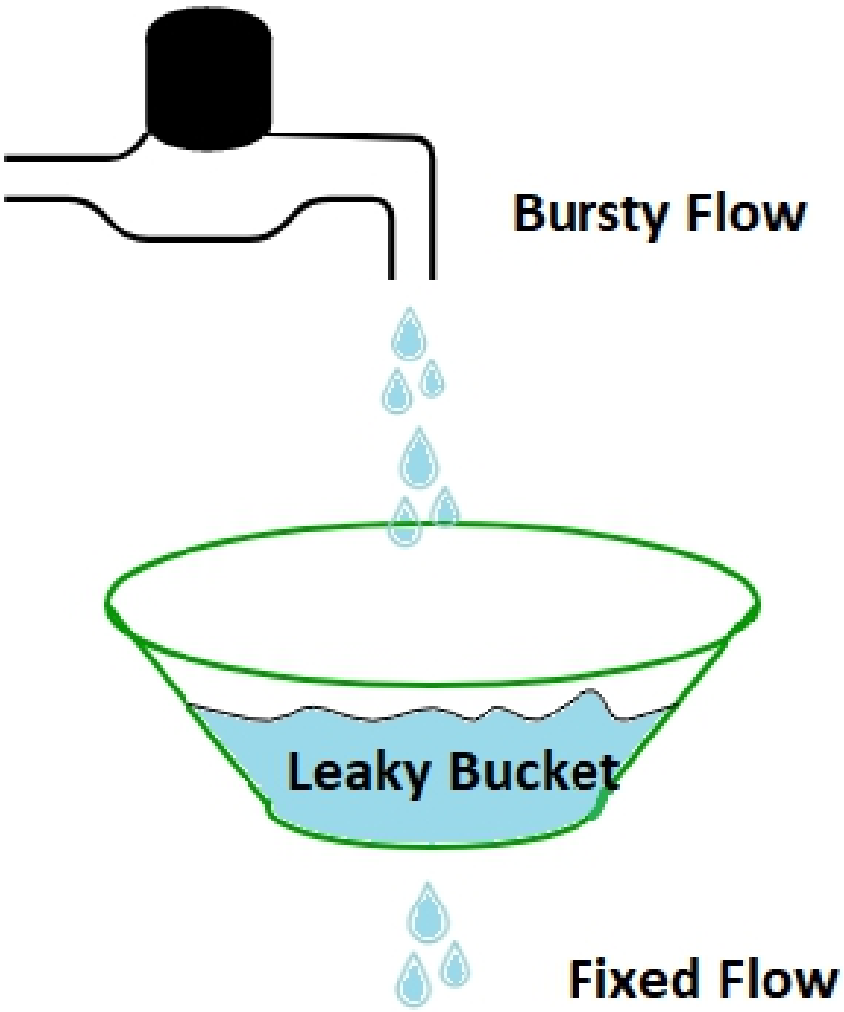
- **Reliability**
- Reliability is a characteristic that a flow needs.
- Lack of reliability means losing a packet or acknowledgment, which entails retransmission.
- However, the sensitivity of application programs to reliability is not the same.
- For example, it is more important that electronic mail, file transfer, and Internet access have reliable transmissions than telephony or audio conferencing.

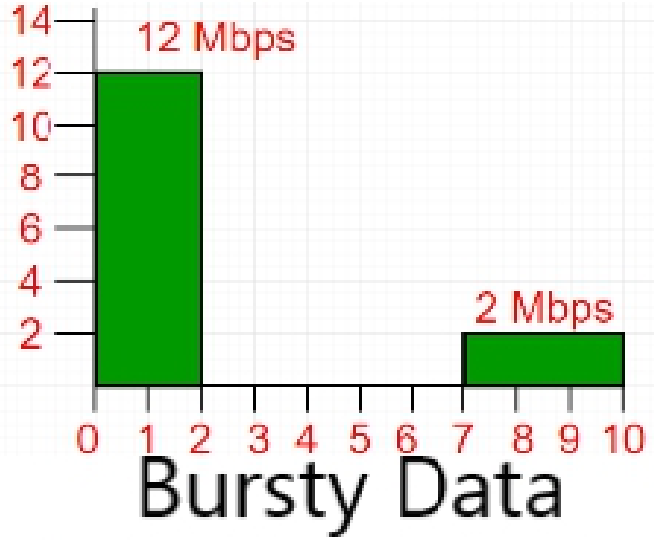
- **Delay**
- Source-to-destination delay is another flow characteristic.
- Again applications can tolerate delay in different degrees.
- In this case, telephony, audio conferencing, video conferencing, and remote log-in need minimum delay, while delay in file transfer or e-mail is less important.
- **Jitter**
- Jitter is the variation in delay for packets belonging to the same flow.
- For example, if four packets depart at times 0, 1, 2, 3 and arrive at 20, 21, 22, 23, all have the same delay, 20 units of time. On the other hand, if the above four packets arrive at 21, 23, 21, and 28, they will have different delays: 21,22, 19, and 24.

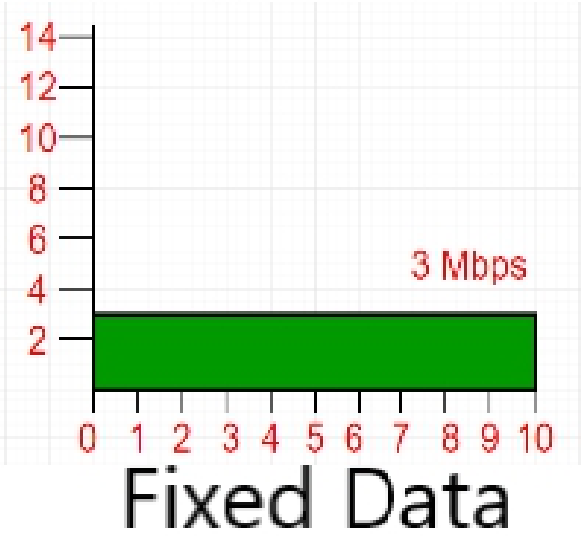
- **Bandwidth**
- Different applications need different bandwidths.
- In video conferencing we need to send millions of bits per second to refresh a color screen while the total number of bits in an e-mail may not reach even a million.

- **Leaky Bucket Algorithm**
- In the network layer, before the network can make Quality of service guarantees, it must know what traffic is being guaranteed.
- One of the main causes of congestion is that traffic is often bursty.
- To understand this concept first we have to know little about traffic shaping.
- Traffic Shaping is a mechanism to control the amount and the rate of the traffic sent to the network.
- Approach of congestion management is called Traffic shaping.
- Traffic shaping helps to regulate rate of data transmission and reduces congestion.

- Suppose we have a bucket in which we are pouring water in a random order but we have to get water in a fixed rate, for this we will make a hole at the bottom of the bucket.
- It will ensure that water coming out is in a some fixed rate, and also if bucket will full we will stop pouring in it.
- The input rate can vary, but the output rate remains constant.
- Similarly, in networking, a technique called leaky bucket can smooth out bursty traffic.
- Bursty chunks are stored in the bucket and sent out at an average rate.


Bursty Flow
Leaky Bucket
Fixed Flow


14
12
10
8
6
4
2
12 Mbps
2 Mbps
0 1 2 3 4 5 6 7 8 9 10
Bursty Data

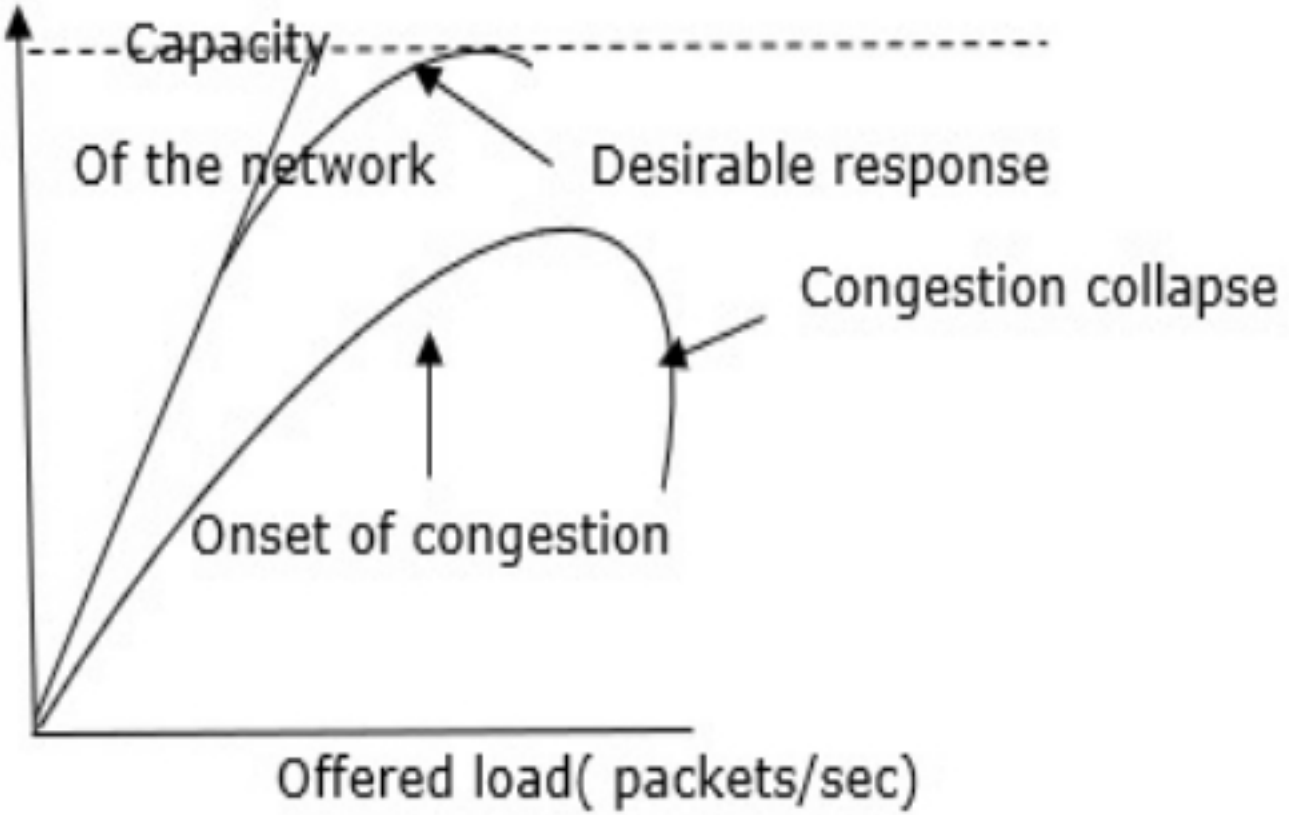

14
12
10
8
6
4
2
3 Mbps
0 1 2 3 4 5 6 7 8 9 10
Fixed Data

- In the figure, we assume that the network has committed a bandwidth of 3 Mbps for a host.
- The use of the leaky bucket shapes the input traffic to make it conform to this commitment.
- In Figure the host sends a burst of data at a rate of 12 Mbps for 2 s, for a total of 24 Mbits of data.
- The host is silent for 5 s and then sends data at a rate of 2 Mbps for 3 s, for a total of 6 Mbits of data.
- In all, the host has sent 30 Mbits of data in 10 s.
- The leaky bucket smooths the traffic by sending out data at a rate of 3 Mbps during the same 10 s.

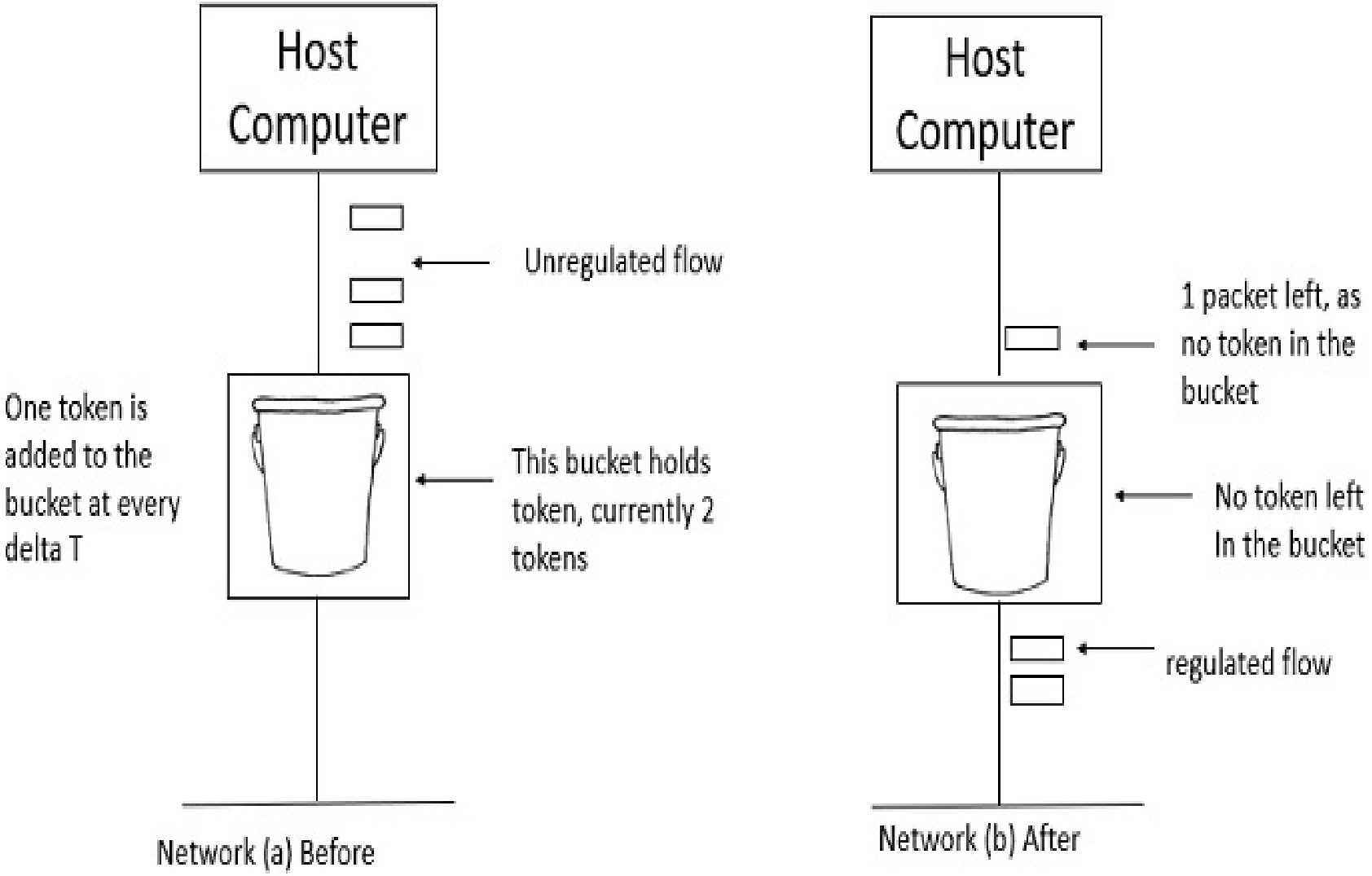
- **The following is an algorithm for variable-length packe**ts:
- Initialize a counter to n at the tick of the clock.
- If n is greater than the size of the packet, send the packet and decrement the counter by the packet size.
- Repeat this step until n is smaller than the packet size.
- Reset the counter and go to step 1.

- **Token Bucket Algorithm**
- Token bucket algorithm is one of the techniques for congestion control algorithms.
- When too many packets are present in the network it causes packet delay and loss of packet which degrades the performance of the system.
- This situation is called congestion.
- The network layer and transport layer share the responsibility for handling congestions.
- One of the most effective ways to control congestion is trying to reduce the load that transport layer is placing on the network.
- To maintain this network and transport layers have to work together.

•The Token Bucket Algorithm is diagrammatically represented as follows –



- **Token Bucket Algorithm**
- The leaky bucket algorithm enforces output patterns at the average rate, no matter how busy the traffic is.
- So, to deal with the more traffic, we need a flexible algorithm so that the data is not lost.
- One such approach is the token bucket algorithm.
- Let us understand this algorithm step wise as given below –
- Step 1 – In regular intervals tokens are thrown into the bucket f.
- Step 2 – The bucket has a maximum capacity f.
- Step 3 – If the packet is ready, then a token is removed from the bucket, and the packet is sent.
- Step 4 – Suppose, if there is no token in the bucket, the packet cannot be sent.

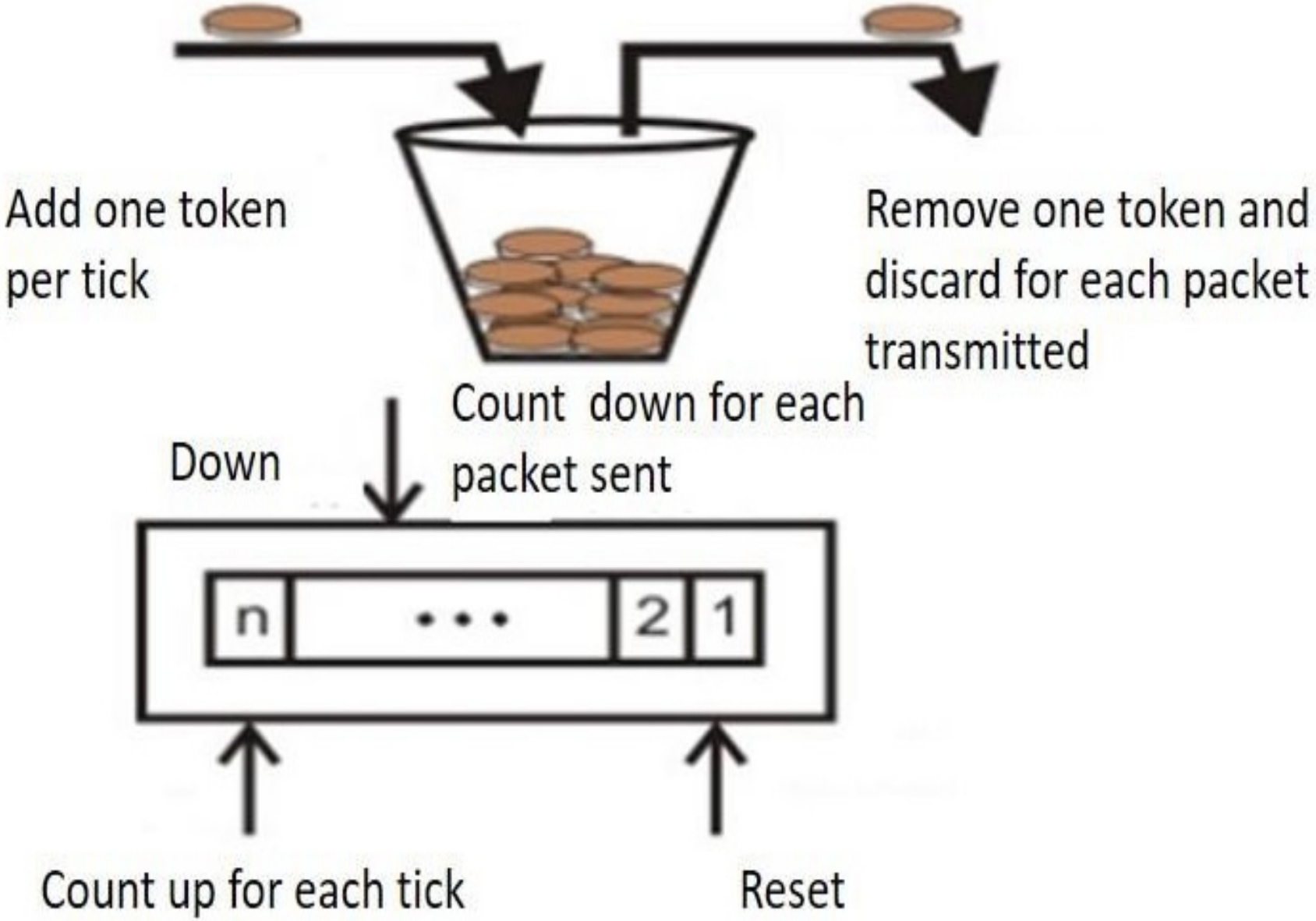

Host Computer
Unregulated flow
One token is added to the bucket at every delta T
This bucket holds token, currently 2 tokens
Network (a) Before
Host Computer
1 packet left, as no token in the bucket
No token left In the bucket
regulated flow
Network (b) After

•In figure (a) the bucket holds two tokens, and three packets are waiting to be sent out of the interface.

•In Figure (b) two packets have been sent out by consuming two tokens, and 1 packet is still left.

•When compared to Leaky bucket the token bucket algorithm is less restrictive that means it allows more traffic. The limit of busyness is restricted by the number of tokens available in the bucket at a particular instant of time.

•The implementation of the token bucket algorithm is easy – a variable is used to count the tokens. For every t seconds the counter is incremented and then it is decremented whenever a packet is sent. When the counter reaches zero, no further packet is sent out.


Add one token per tick
Remove one token and discard for each packet transmitted
Count down for each packet sent
Down
n
• • •
2
1
Count up for each tick
Reset