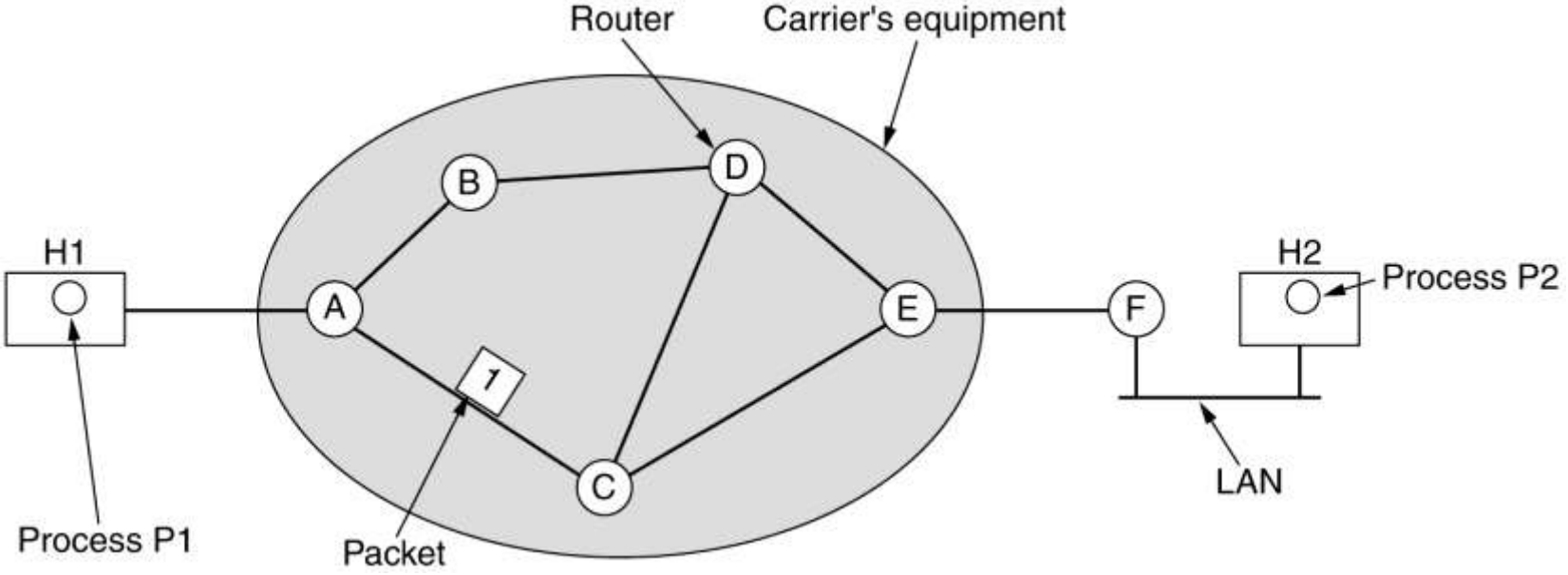
Network Layer Design Issues

- Store-and-Forward Packet Switching
- Services Provided to the Transport Layer
- Implementation of Connectionless Service
- Implementation of Connection-Oriented Service
- Comparison of Virtual-Circuit and Datagram Subnets

Store-and-Forward Packet Switching

The environment of the network layer protocols.



Services provided to Transport layer

The services need to be carefully designed with the following goals in mind:

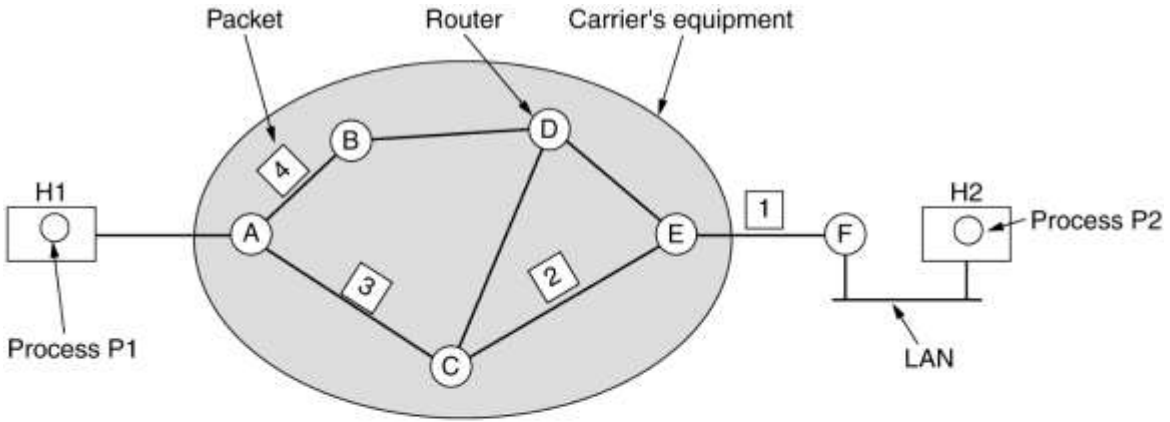
1. The services should be independent of the router technology.
2. The transport layer should be shielded from the number, type, and topology of the routers present.
3. The network addresses made available to the transport layer should use a uniform numbering plan, even across LANs and WANs.

Packet Switching

- Datagram Approach: Connectionless Service
- Virtual Circuit Approach: Connection Oriented Service

Implementation of Connectionless Service

Routing within a diagram subnet.



A's table

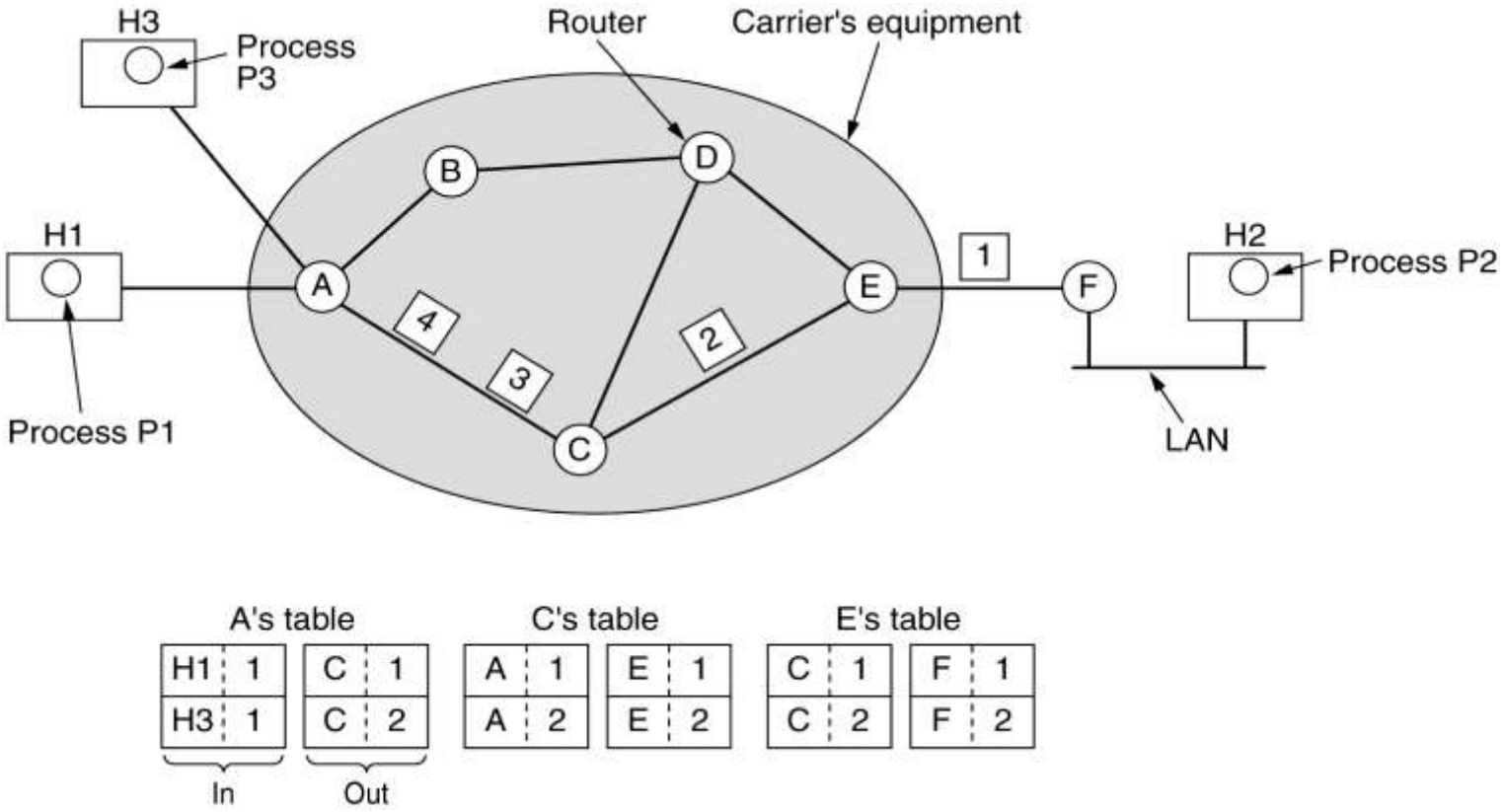
Dest.	initially	later
A	–	–
B	B	B
C	C	C
D	B	B
E	C	B
F	C	B

Dest.	C's table
A	A
B	A
C	–
D	D
E	E
F	E

Dest.	E's table
A	C
B	D
C	C
D	D
E	–
F	F

Implementation of Connection-Oriented Service

Routing within a virtual-circuit subnet.



Comparison of Virtual-Circuit and Datagram Subnets

Issue	Datagram subnet	Virtual-circuit subnet
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

Network Layer Services

- Packetizing: encapsulating the payload(data received from upper layer) at source and decapsulating at the destination.
- Routing: To find the best path from source to destination using routing protocols.
- Forwarding: Action applied by each router when packet arrives at one of its interface using routing or forwarding table.
- Routing and Forwarding are related to each other.

NETWORK-LAYER PERFORMANCE

The performance of a network can be measured in terms of:

- *Delay*
- *Throughput*
- *Packet loss*

Delay

The delays in a network can be divided into four types:

- Transmission delay: time it takes to push the packet's bits onto the link
- Propagation delay: time for a signal to propagate through the media
- Processing delay: time it takes a router to process the packet header
- Queuing delay: time the packet spends in routing queues

- *Transmission Delay*

  Delaytr = (Packet length) / (Transmission rate).

- *Propagation Delay*

  Delaypg = (Distance) / (Propagation speed).

- *Processing Delay*

  Delaypr = Time required to process a packet in a router or a destination host

- *Queuing Delay*

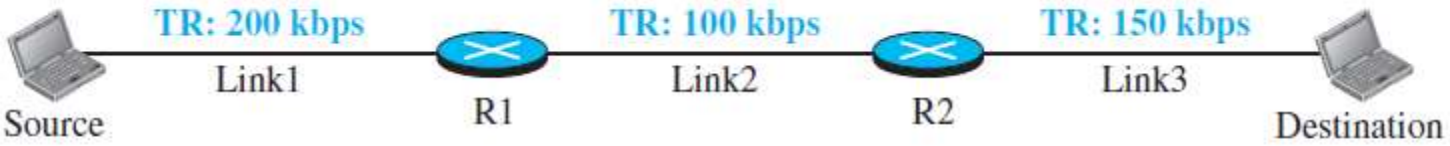
Delayqu = The time a packet waits in input and output queues in a router

- *Total Delay*

Total delay = (*n* + *1) (Delaytr* + *Delaypg* + *Delaypr)* + *(n) (Delayqu)*

Throughput

- **Throughput = minimum {TR1, TR2, . . . TRn}.**



a. A path through three links

TR: Transmission rate

Packet Loss

- When a router receives a packet while processing another packet, the received packet needs to be stored in the input buffer waiting for its turn.

- A router, however, has an input buffer with a limited size. A time may come when the buffer is full and the next packet needs to be dropped.

- The effect of packet loss on the Internet network layer is that the packet needs to be resent, which in turn may create overflow and cause more packet loss.

Note

An IPv4 address is 32 bits long.

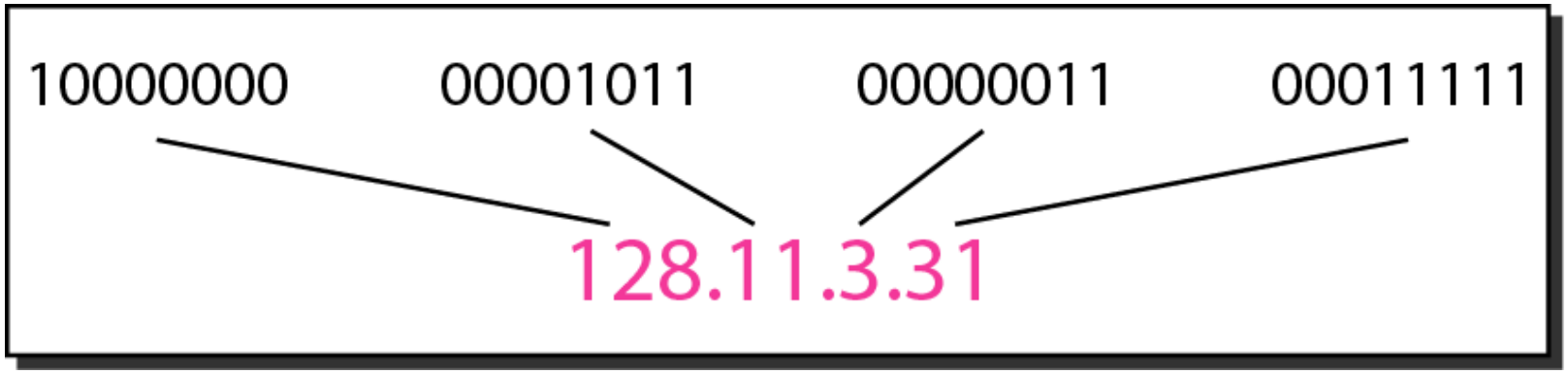
Note

The IPv4 addresses are unique
and universal.

Note

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

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



Example 19.1

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

a. 10000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

Solution

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

a. 129.11.11.239

b. 193.131.27.255

Example 19.2

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

a. 111.56.45.78

b. 221.34.7.82

Solution

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

a. 01101111 00111000 00101101 01001110

b. 11011101 00100010 00000111 01010010

Example 19.3

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

a. 111.56.045.78

b. 221.34.7.8.20

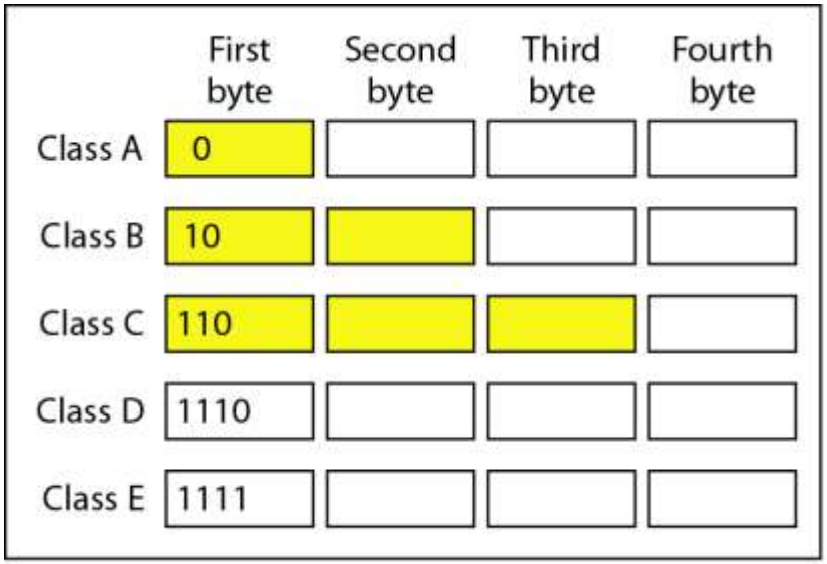
c. 75.45.301.14

d. 11100010.23.14.67

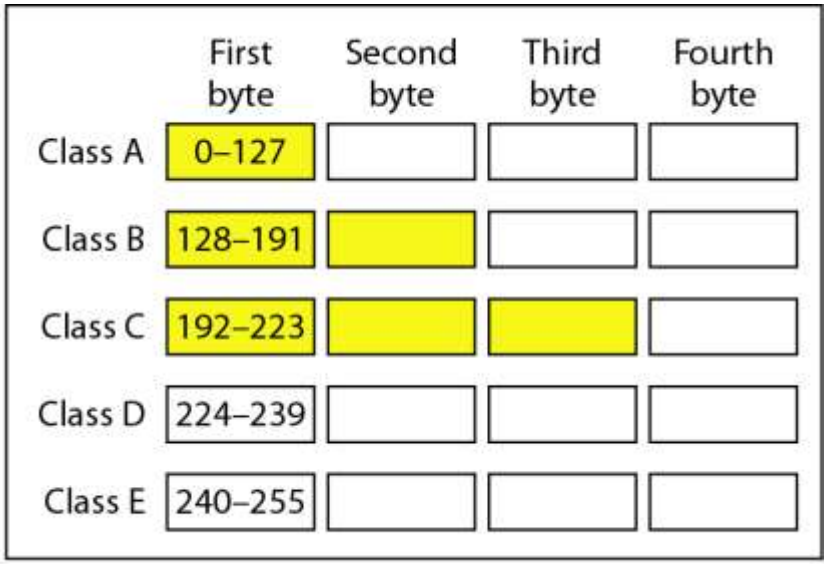
Note

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

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



a. Binary notation



b. Dotted-decimal notation

Example 19.4

Find the class of each address.

a. 00000001 00001011 00001011 11101111

b. 11000001 10000011 00011011 11111111

c. 14.23.120.8

d. 252.5.15.111

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

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

Note

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

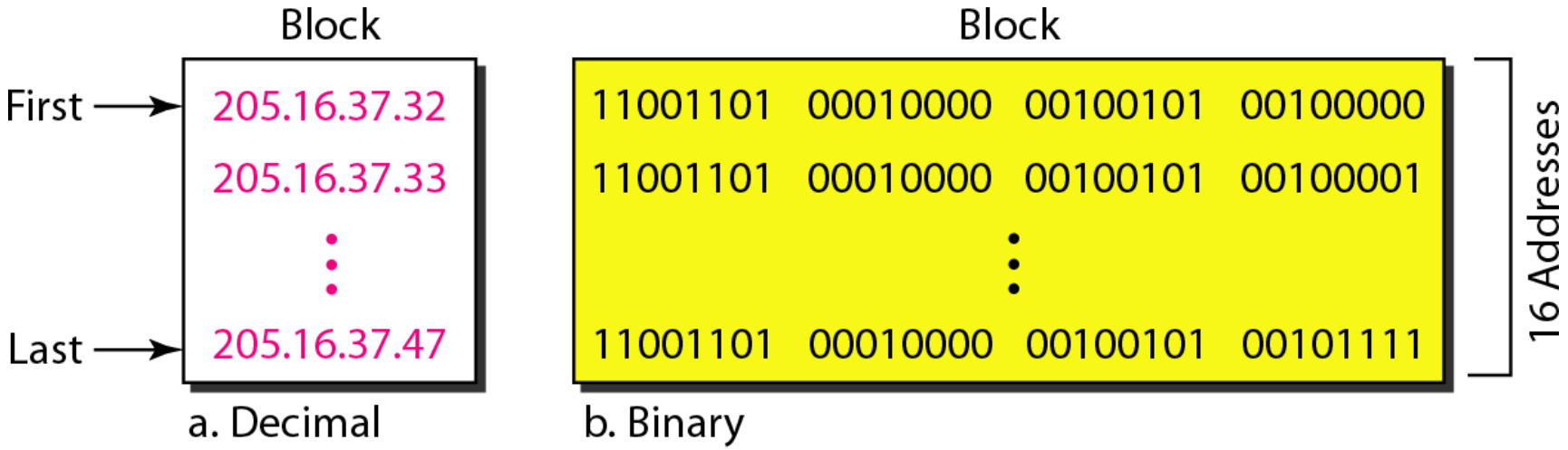
Table 19.2 *Default masks for classful addressing*

Class	*Binary*	*Dotted-Decimal*	*CIDR*
A	**11111111** 00000000 00000000 00000000	**255**.0.0.0	/8
B	**11111111 11111111** 00000000 00000000	**255.255**.0.0	/16
C	**11111111 11111111 11111111** 00000000	**255.255.255**.0	/24

Note

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

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



Note

In IPv4 addressing, a block of addresses can be defined as x.y.z.t /*n* in which x.y.z.t defines one of the addresses and the /*n* defines the mask.

Note

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

Note

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

Example 19.6

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

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

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

11001101 00010000 00100101 0010000

or

205.16.37.32.

This is actually the block shown in Figure 19.3.

Note

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

Example 19.7

Find the last address for the block in Example 19.6.

Solution

The binary representation of the given address is

11001101 00010000 00100101 00100111

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

11001101 00010000 00100101 00101111

or

205.16.37.47

This is actually the block shown in Figure 19.3.

Note

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

Find the number of addresses in Example 19.6.

Solution

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

Example 19.9

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

11111111 11111111 11111111 11110000

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

Find

a. The first address

b. The last address

c. The number of addresses.

Solution

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

Address:	11001101	00010000	00100101	00100111
Mask:	**11111111**	**11111111**	**11111111**	**11110000**
First address:	11001101	00010000	00100101	00100000

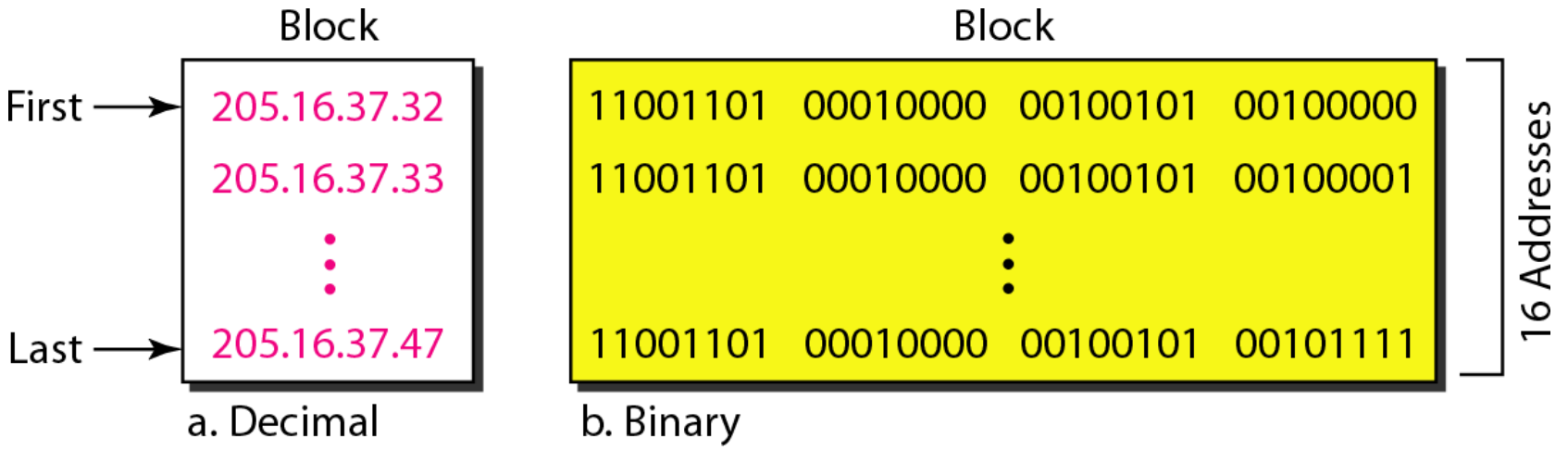
b. The last address can be found by ORing the given addresses with the complement of the mask. ORing here is done bit by bit. The result of ORing 2 bits is 0 if both bits are 0s; the result is 1 otherwise. The complement of a number is found by changing each 1 to 0 and each 0 to 1.

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

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

Mask complement:	**00000000 00000000 00000000 00001111**
Number of addresses:	15 + 1 = 16

Figure 19.4 *A network configuration for the block 205.16.37.32/28*



Note

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

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

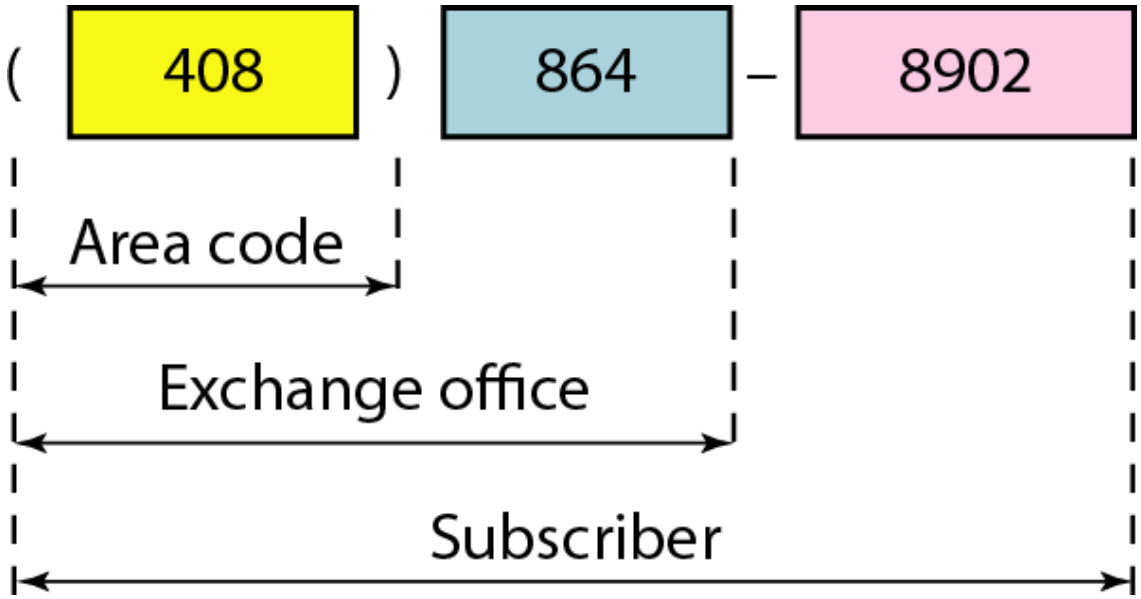
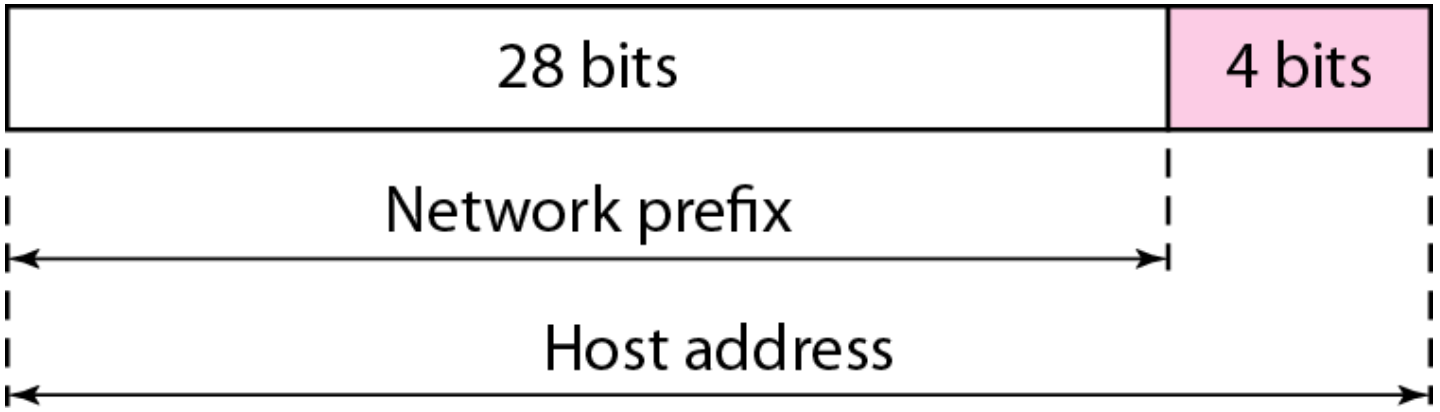


Figure 19.6 *A frame in a character-oriented protocol*



Note

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

Figure 19.7 *Configuration and addresses in a subnetted network*

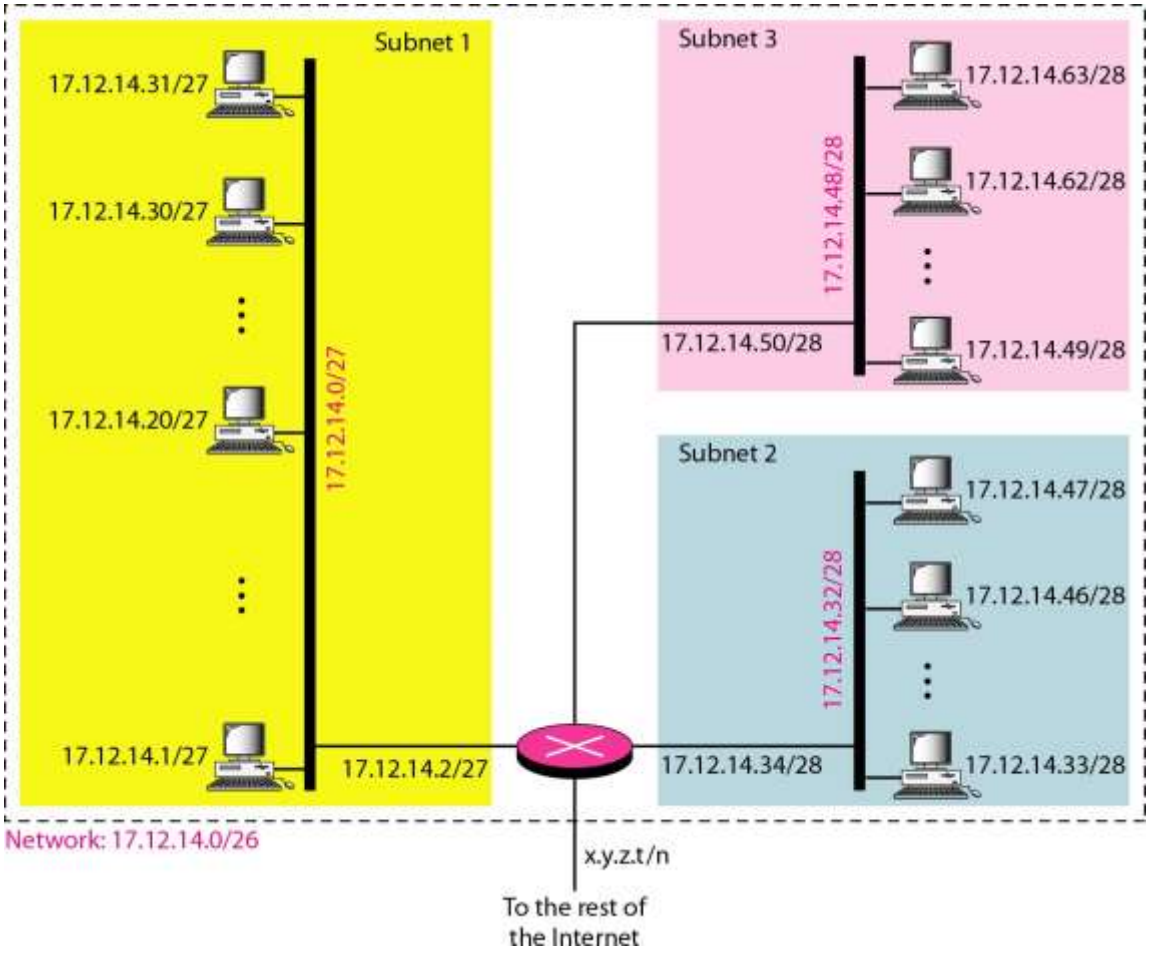
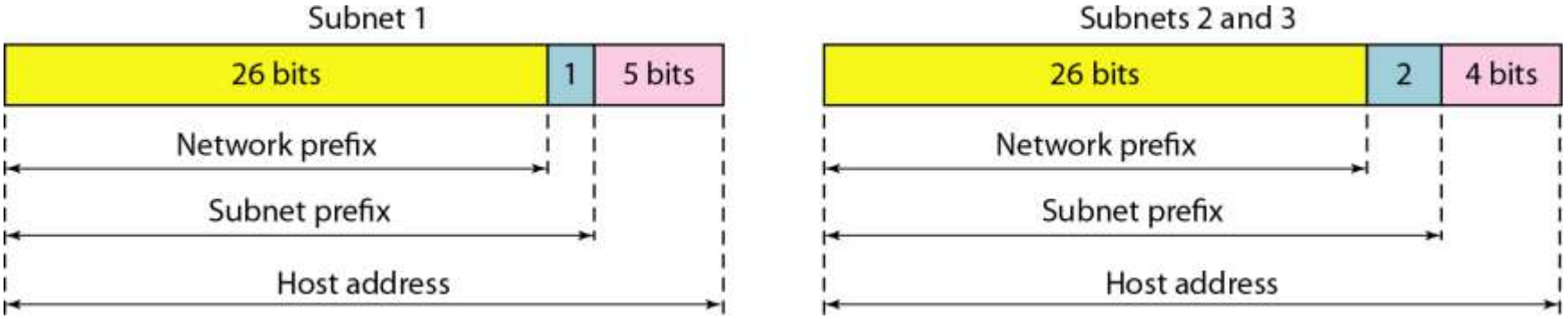


Figure 19.8 *Three-level hierarchy in an IPv4 address*



Example 19.10

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

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

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

Solution

Figure 19.9 shows the situation.

Group 1

For this group, each customer needs 256 addresses. This means that 8 (log2 256) bits are needed to define each host. The prefix length is then 32 − 8 = 24. The addresses are

1st Customer:	190.100.0.0/24	190.100.0.255/24
2nd Customer:	190.100.1.0/24	190.100.1.255/24
. . .		
64th Customer:	190.100.63.0/24	190.100.63.255/24

Total = 64 × 256 = 16,384

Group 2

For this group, each customer needs 128 addresses. This means that 7 (log2 128) bits are needed to define each host. The prefix length is then 32 − 7 = 25. The addresses are

1st Customer:	190.100.64.0/25	190.100.64.127/25
2nd Customer:	190.100.64.128/25	190.100.64.255/25
. . .		
128th Customer:	190.100.127.128/25	190.100.127.255/25

Total = 128 × 128 = 16,384

Group 3

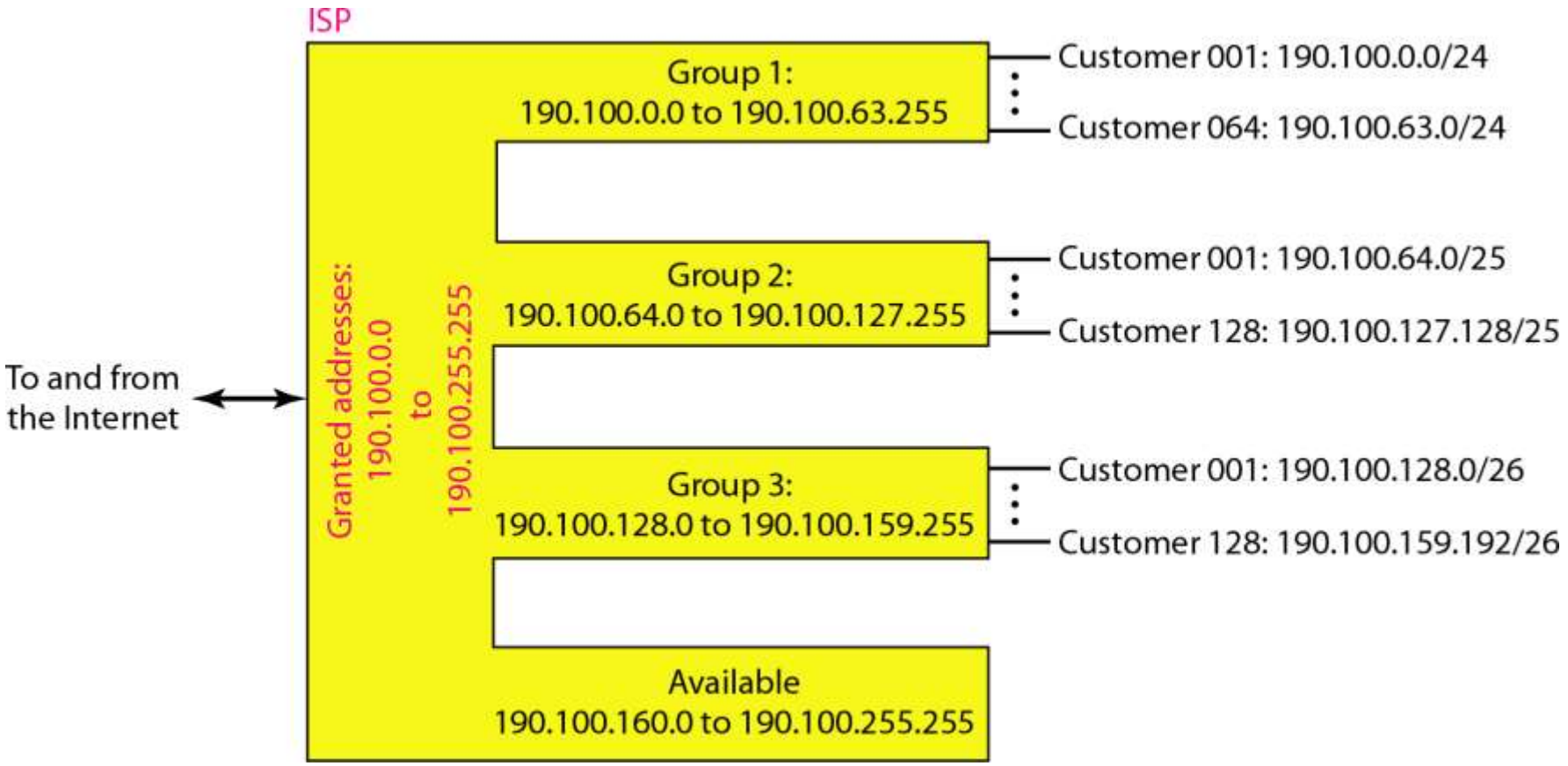
For this group, each customer needs 64 addresses. This means that 6 ($log_2 64$) bits are needed to each host. The prefix length is then 32 − 6 = 26. The addresses are

1st Customer:	190.100.128.0/26	190.100.128.63/26
2nd Customer:	190.100.128.64/26	190.100.128.127/26
. . .		
128th Customer:	190.100.159.192/26	190.100.159.255/26

Total = 128 × 64 = 8192

Number of granted addresses to the ISP: 65,536
Number of allocated addresses by the ISP: 40,960
Number of available addresses: 24,576

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



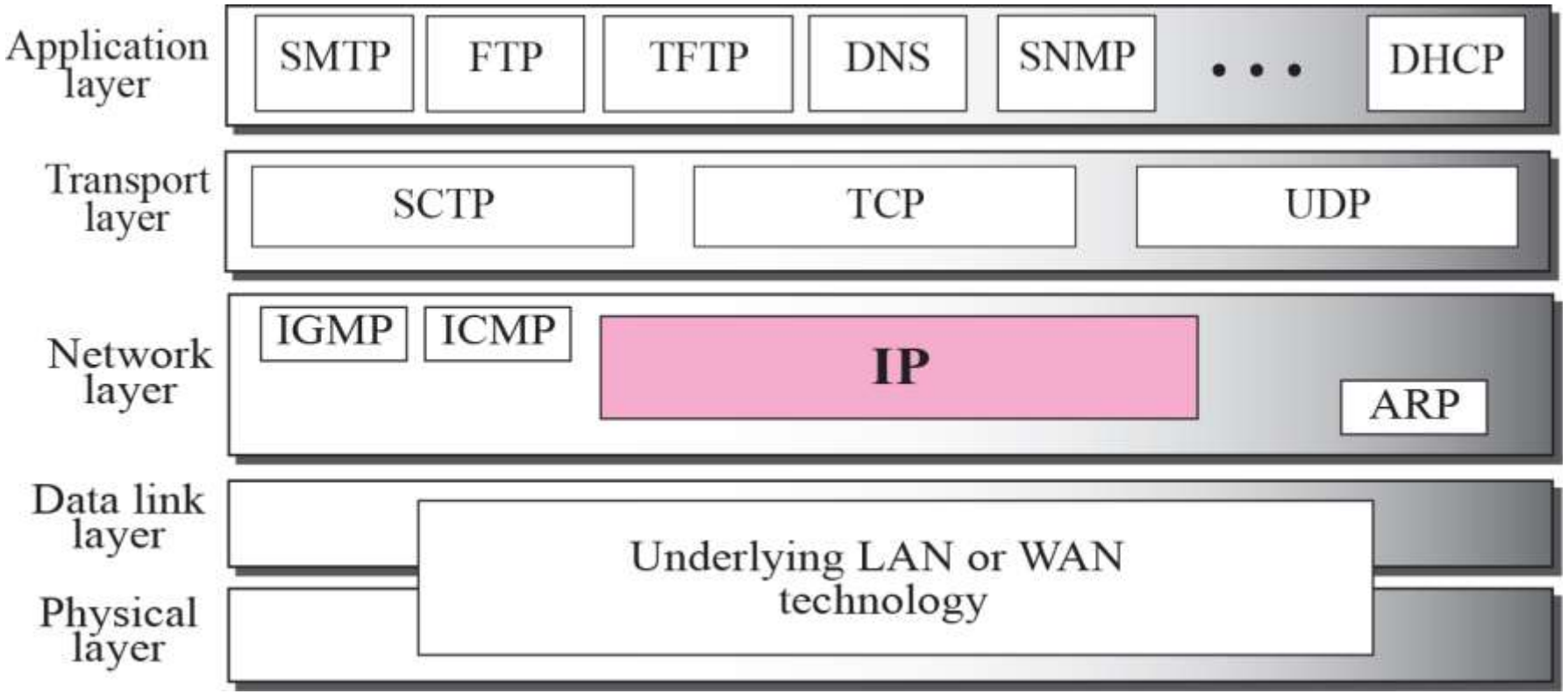
7-1 INTRODUCTION

The Internet Protocol (IP) is the transmission mechanism used by the TCP/IP protocols at the network layer.

Topics Discussed in the Section

- ✓ **Relationship of IP to the rest of the TCP/IP Suite**

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



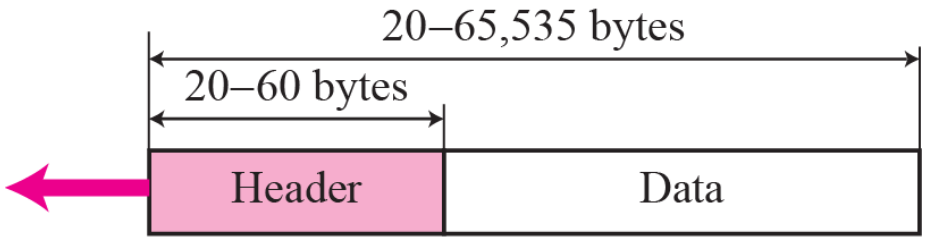
7-2 DATAGRAMS

Packets in the network (internet) layer are called *datagrams*. 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.

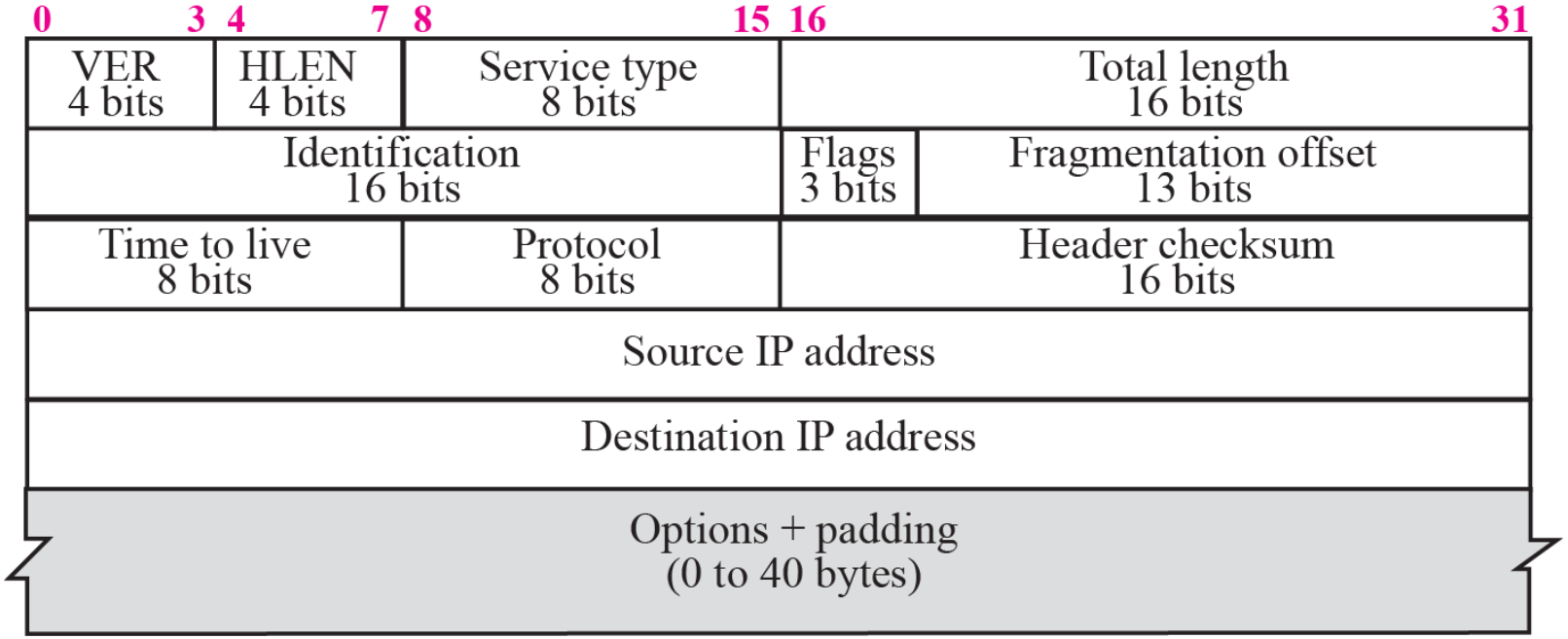
Topics Discussed in the Section

- ✓ **Format of the datagram packet**
- ✓ **Some examples**

Figure 7.2 *IP datagram*



a. IP datagram



b. Header format

Figure 20.6 *Service type or differentiated services*

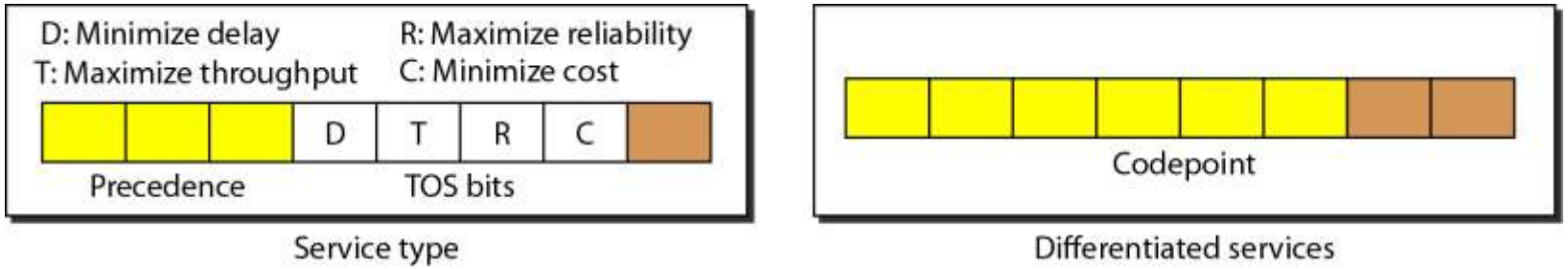


Table 20.1 *Types of service*

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

Note

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

Figure 7.4 *Encapsulation of a small datagram in an Ethernet frame*

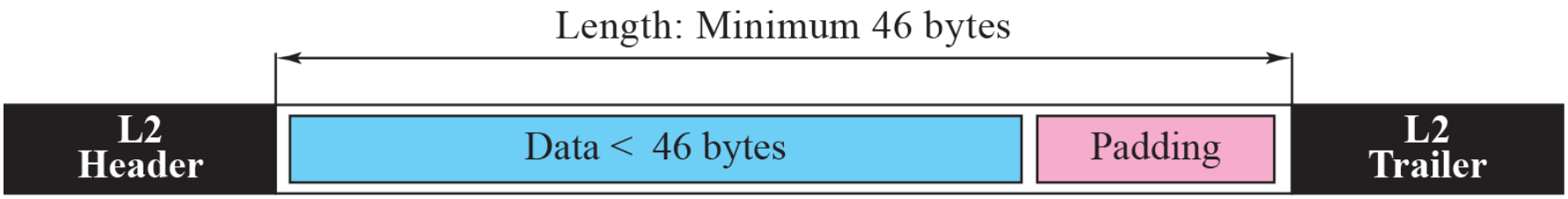


Figure 7.5 *Multiplexing*

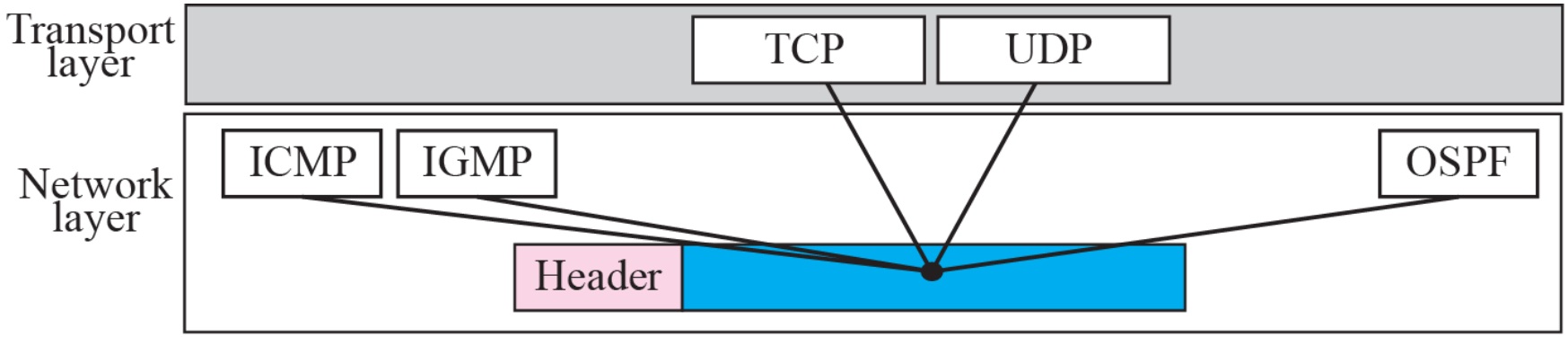


Table 7.2 *Protocols*

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

Example 7.1

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

```
01000010
```

The receiver discards the packet. Why?

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?

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 5_{16} 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 (see Table 7.2)

7-3 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.

Topics Discussed in the Section

- ✓ **Maximum Transfer Unit (MTU)**
- ✓ **Fields Related to Fragmentation**

Figure 7.6 *MTU*

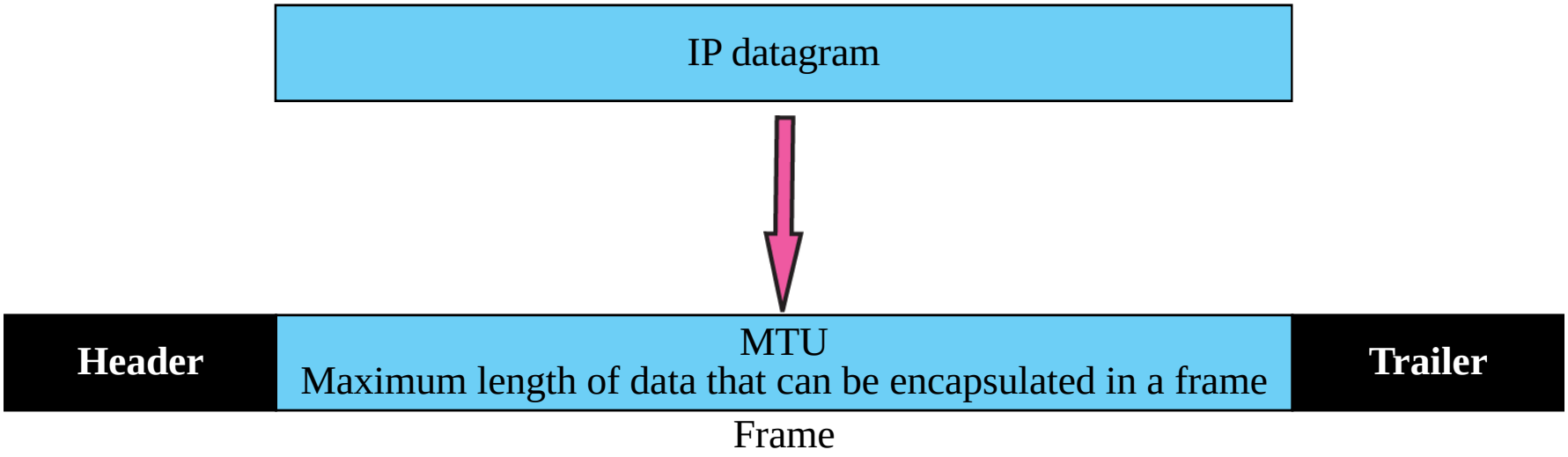


Table 20.5 *MTUs for some networks*

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

Note

Only data in a datagram is fragmented.

Figure 7.7 *Flags field*

D: Do not fragment
M: More fragments

Figure 7.8 *Fragmentation example*

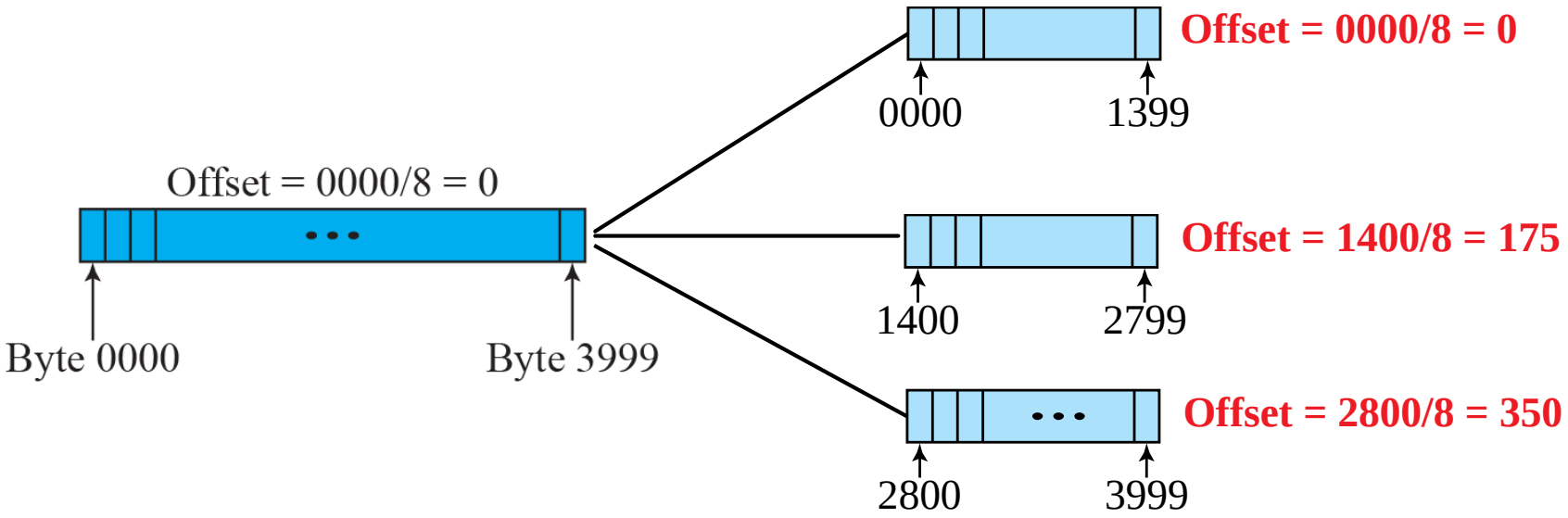
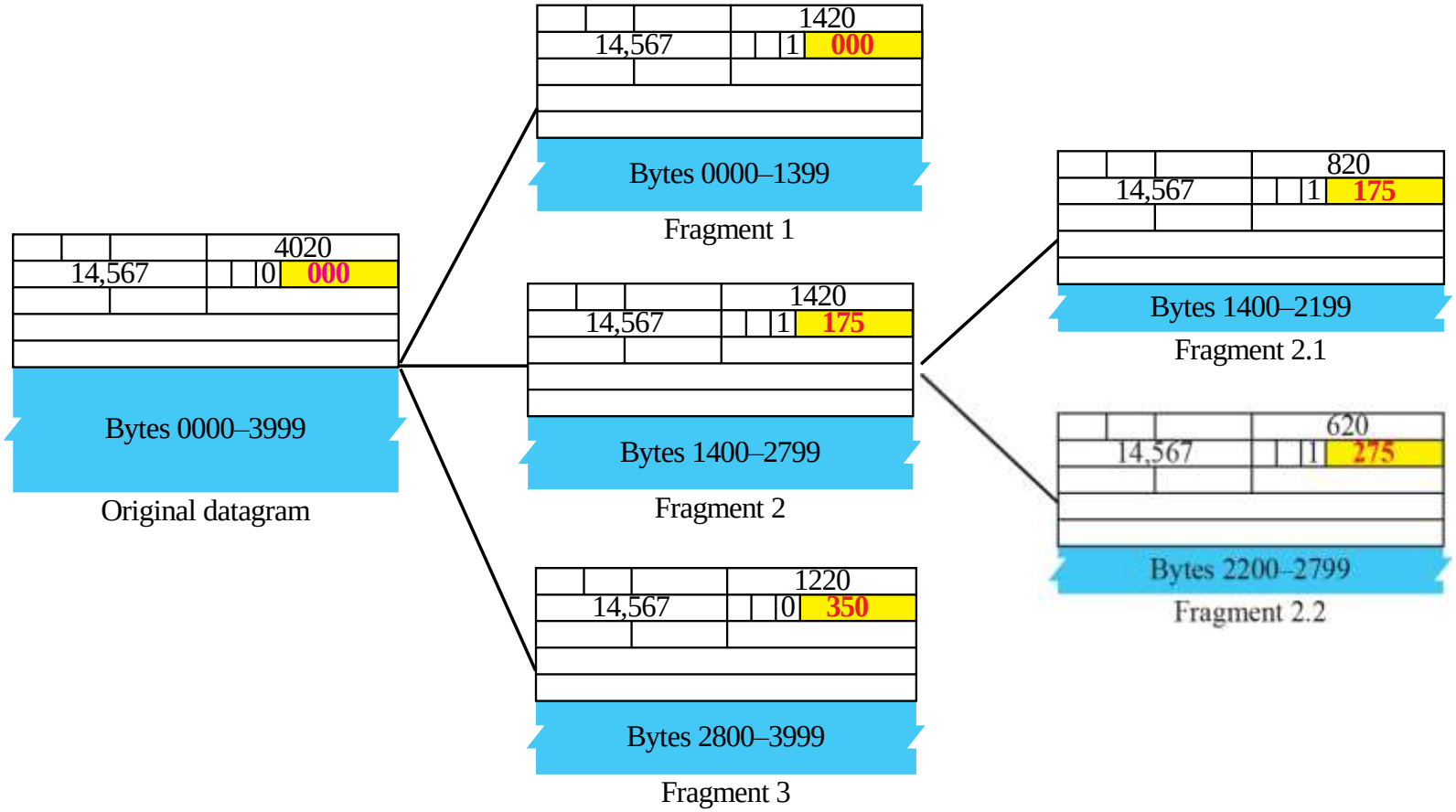


Figure 7.9 *Detailed fragmentation example*



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). See also the next example.

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.

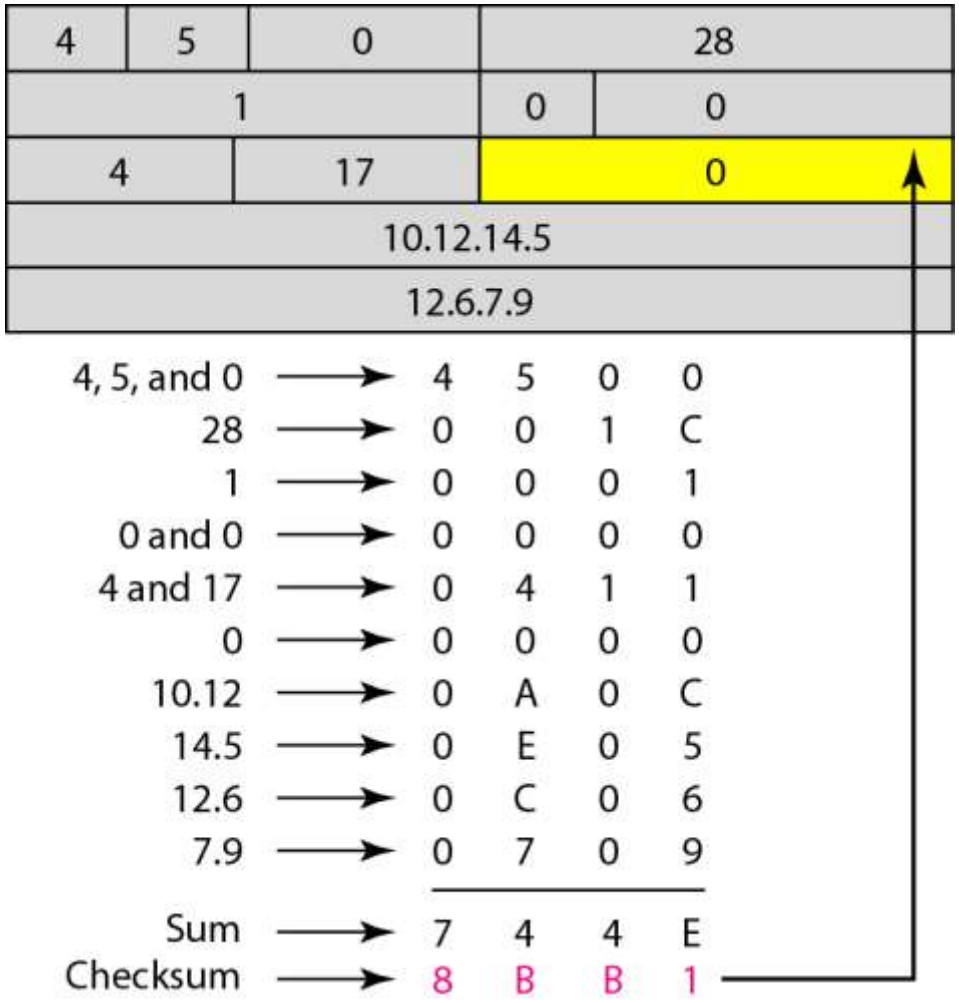
Example 7.9

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.

Figure 20.13 *Example of checksum calculation in IPv4*

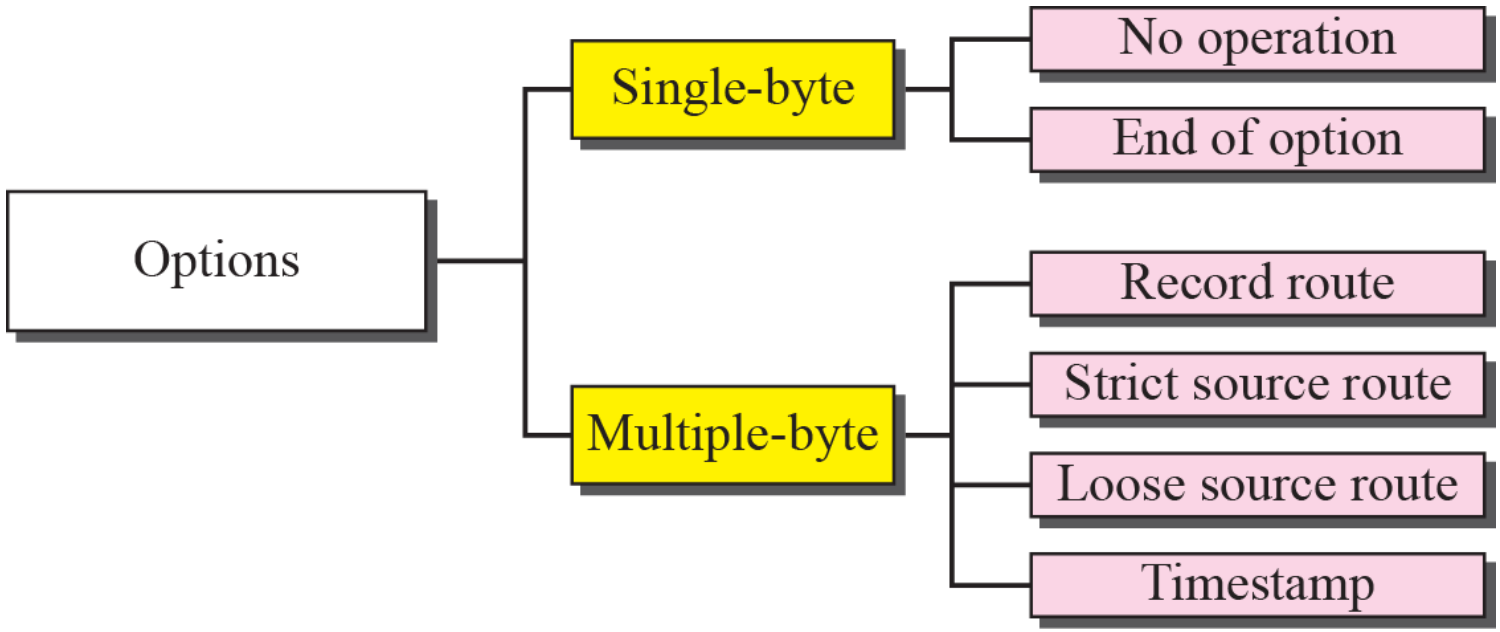


7-4 OPTIONS

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.

Figure 7.11 *Categories of options*



changes implemented in the protocol in addition to changing address size and format.

- ❑ Better header format. IPv6 uses a new header format in which options are separated from the base header and inserted, when needed, between the base header and the data. This simplifies and speeds up the routing process because most of the options do not need to be checked by routers.
- ❑ New options. IPv6 has new options to allow for additional functionalities.
- ❑ Allowance for extension. IPv6 is designed to allow the extension of the protocol if required by new technologies or applications.

- ❑ Support for resource allocation. In IPv6, the type-of-service field has been removed, but two new fields, traffic class and flow label, have been added to enable the source to request special handling of the packet. This mechanism can be used to support traffic such as real-time audio and video.

- ❑Support for more security. The encryption and authentication options in IPv6 provide confidentiality and integrity of the packet

27-2 PACKET FORMAT

The IPv6 packet is shown in Figure 27.1. Each packet is composed of a mandatory base header followed by the payload. The payload consists of two parts: optional extension headers and data from an upper layer. The base header occupies 40 bytes, whereas the extension headers and data from the upper layer contain up to 65,535 bytes of information.

Figure 20.15 ***IPv6 datagram header and payload***

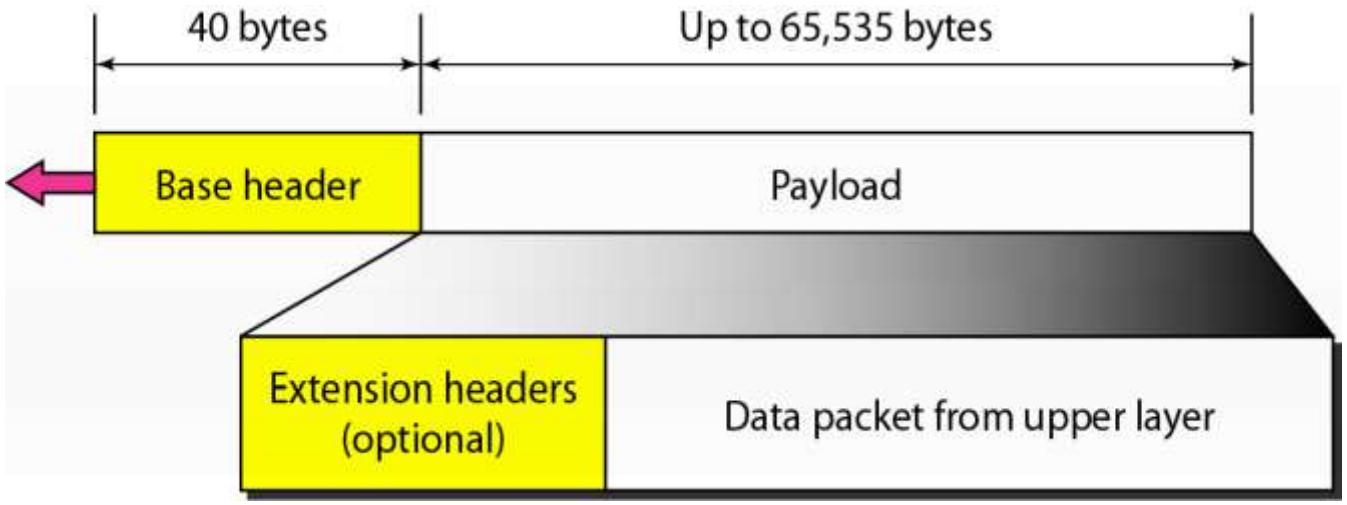


Figure 27.1 *IPv6 datagram*

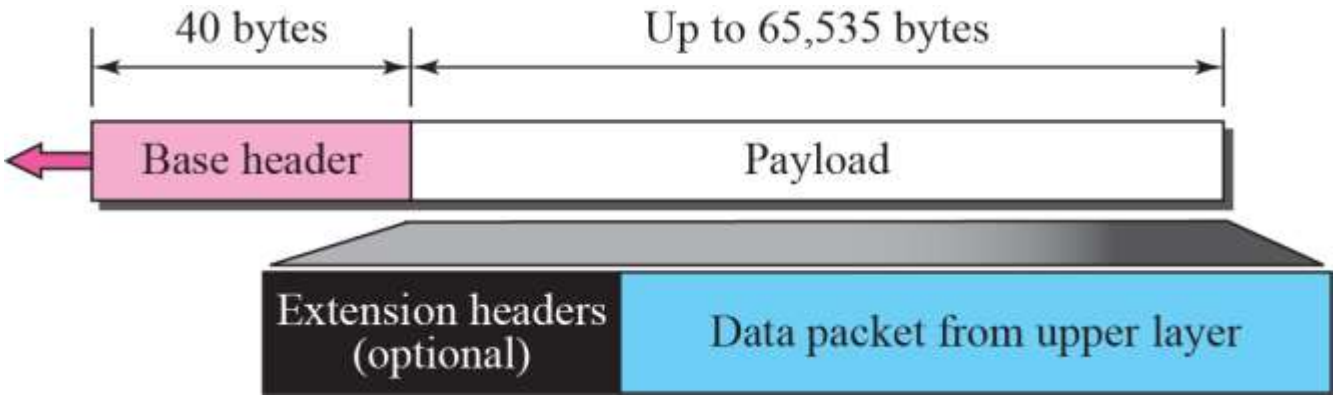
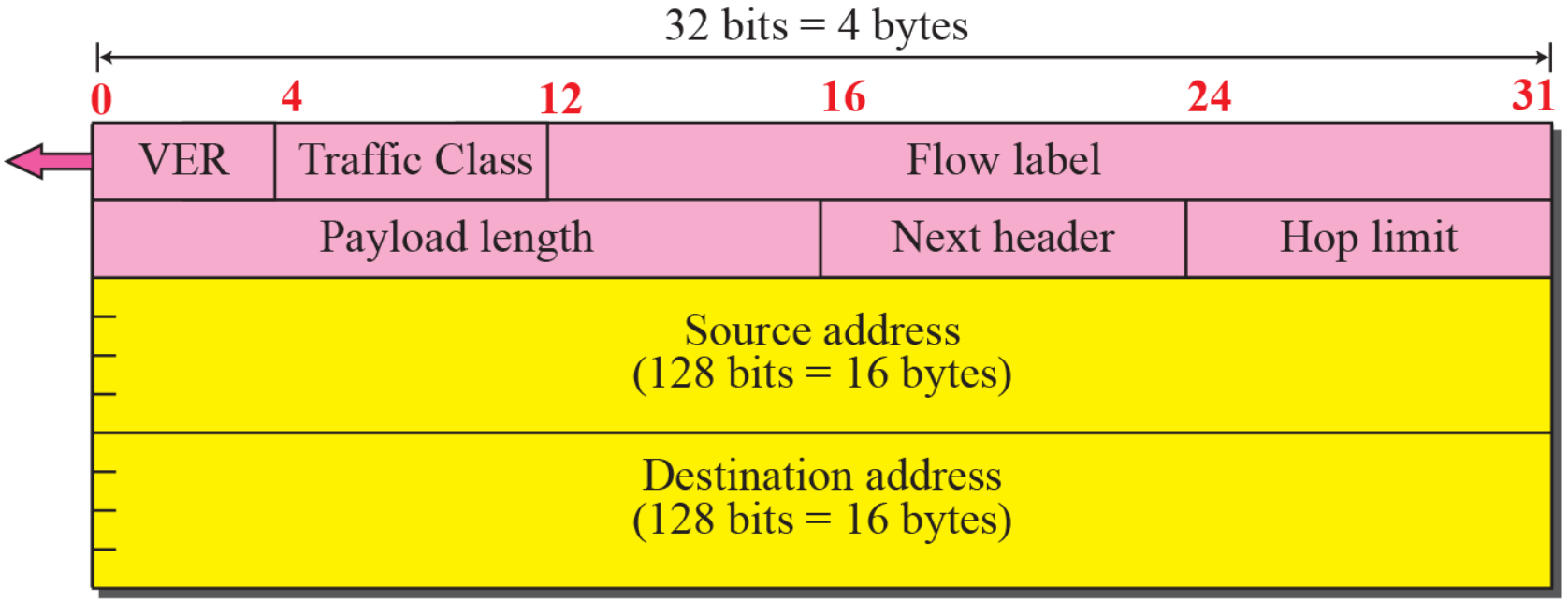


Figure 27.2 *Format of the base header*



- Version. The 4-bit version field defines the version number of the IP. For IPv6, the value is 6.
- ❑ Traffic class. The 8-bit traffic class field is used to distinguish different payloads with different delivery requirements. It replaces the type-of-service field in IPv4.
- ❑ Flow label. The flow label is a 20-bit field that is designed to provide special han□dling for a particular flow of data. We will discuss this field later.
- ❑ Payload length. The 2-byte payload length field defines the length of the IP datagram excluding the header. Note that IPv4 defines two fields related to the length: header length and total length. In IPv6, the length of the base header is fixed (40 bytes); only the length of the payload needs to be defined..

- ❑ Next header. The next header is an 8-bit field defining the type of the first extension header (if present) or the type of the data that follows the base header in the datagram. This field is similar to the protocol field in IPv4, but we talk more about it when we discuss the payload

- Hop limit. The 8-bit hop limit field serves the same purpose as the TTL field in IPv4.
- ❑ Source and destination addresses. The source address field is a 16-byte (128-bit) Internet address that identifies the original source of the datagram. The destination address field is a 16-byte (128-bit) Internet address that identifies the destination of the datagram.
- ❑ Payload. Compared to IPv4, the payload field in IPv6 has a different format

Table 27.1 *Next Header Codes*

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

Figure 27.3 *Extension header format*

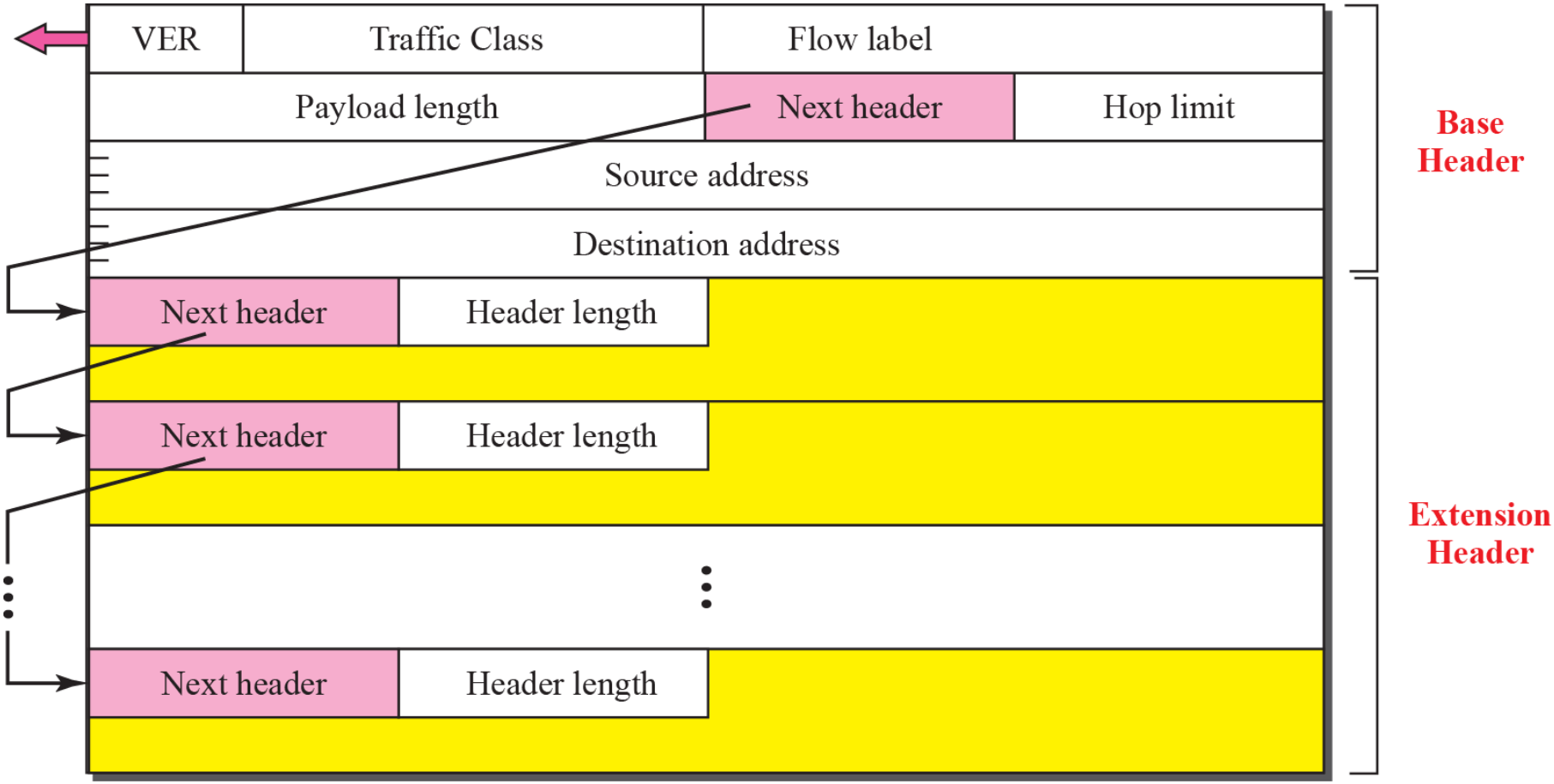


Table 20.9 *Comparison between IPv4 and IPv6 packet*

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

Figure 20.17 ***Extension header types***

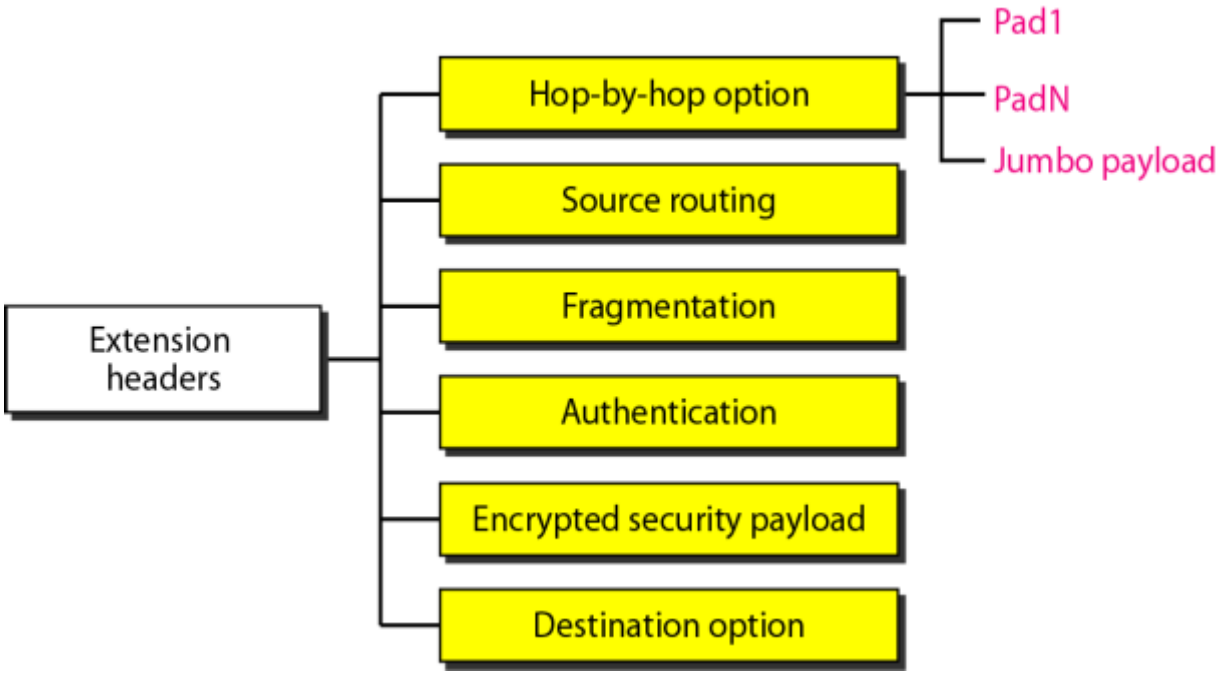


Table 20.10 *Comparison between IPv4 options and IPv6 extension headers*

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

Transition from IPv4 to IPv6

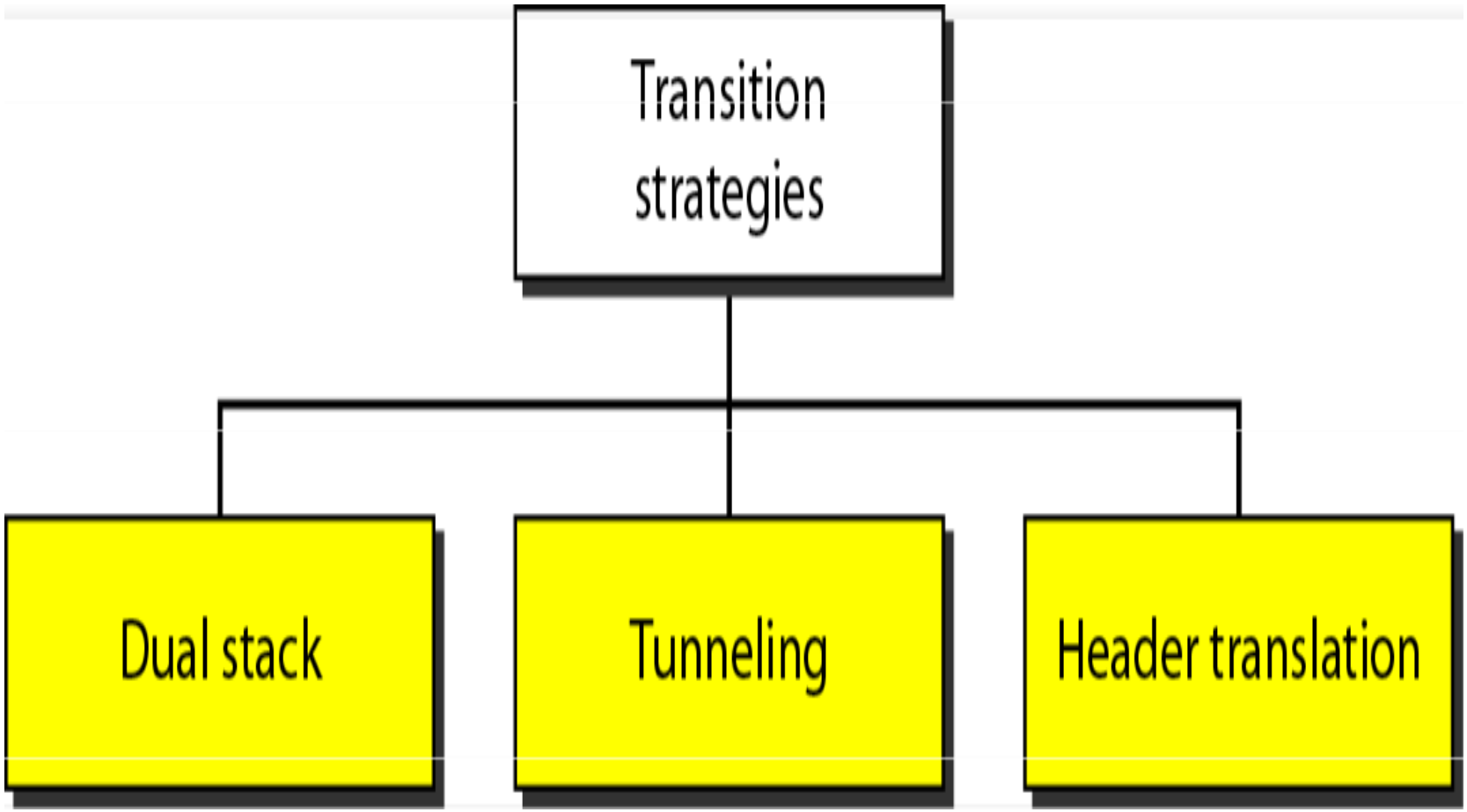


Figure 20.19 *Dual stack*

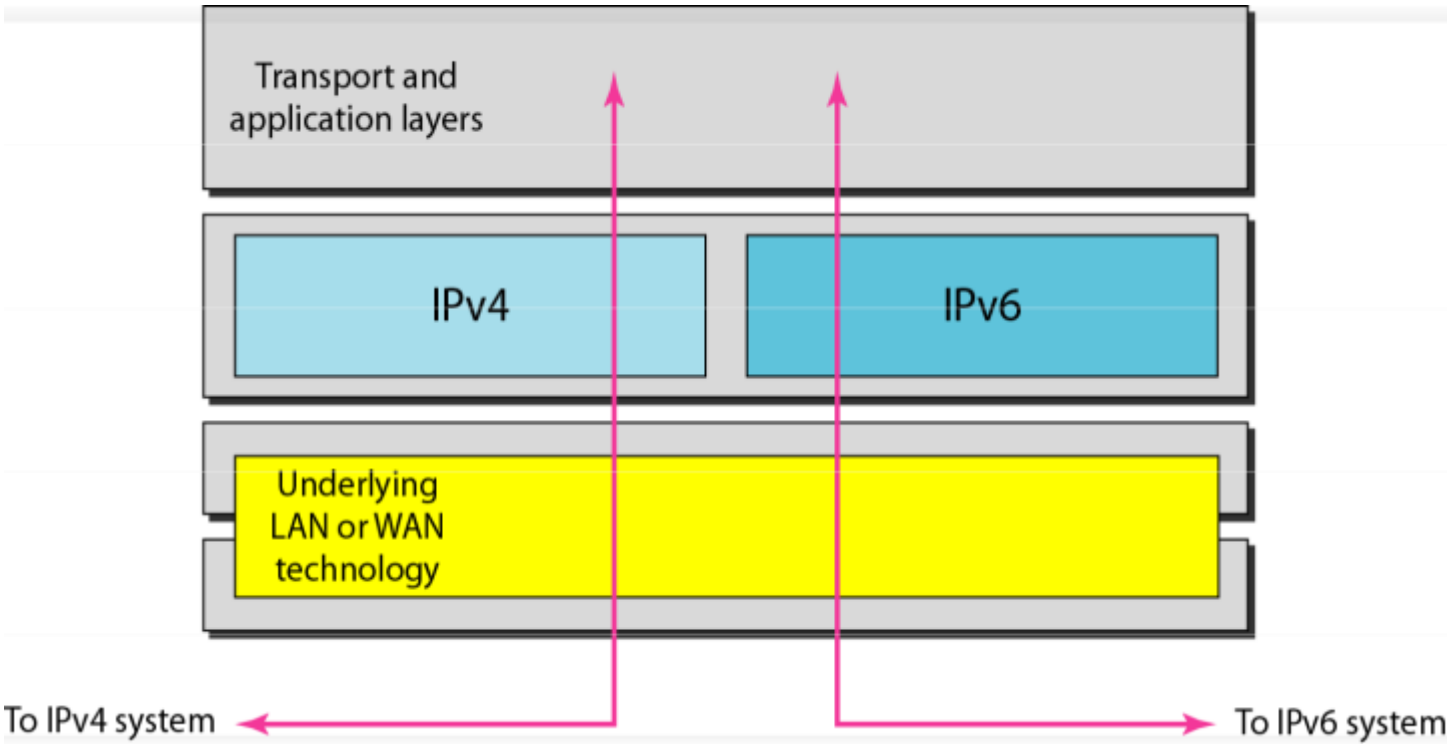


Figure 20.20 *Tunneling strategy*

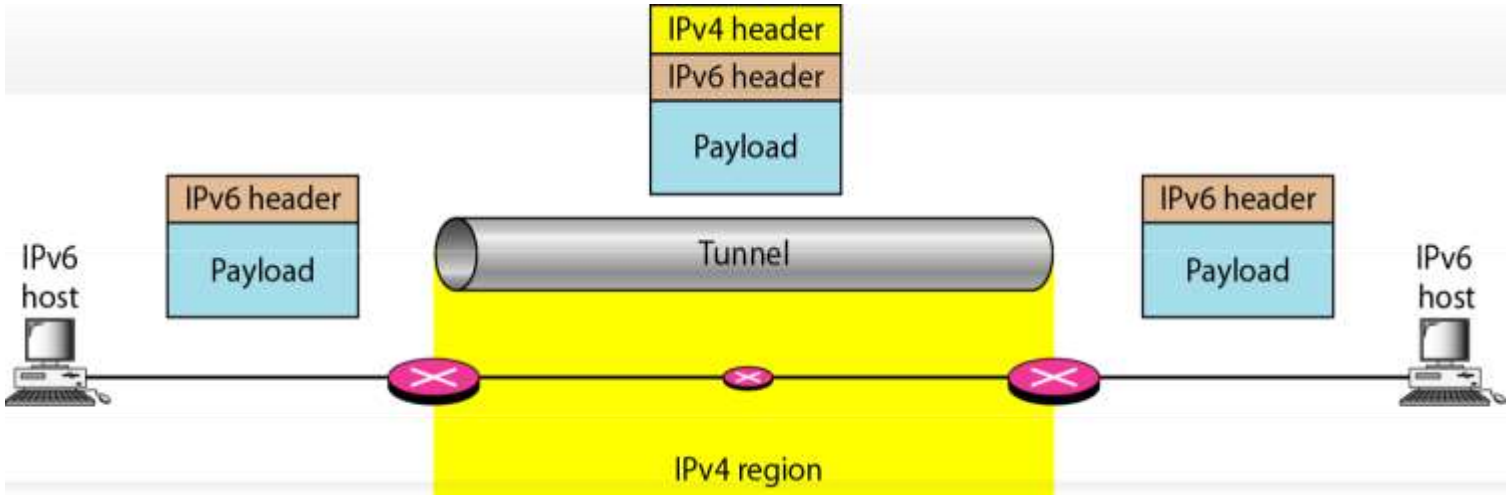
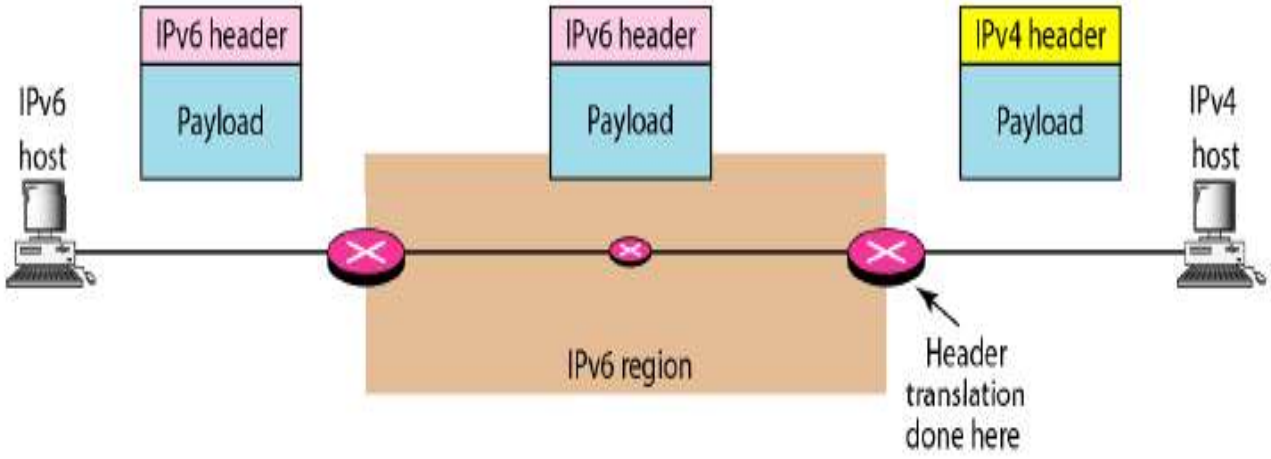


Figure 20.21 *Header translation strategy*



19-2 IPv6 ADDRESSES

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

Topics discussed in this section:

Structure
Address Space

Note

An IPv6 address is 128 bits long.

Figure 19.14 *IPv6 address in binary and hexadecimal colon notation*

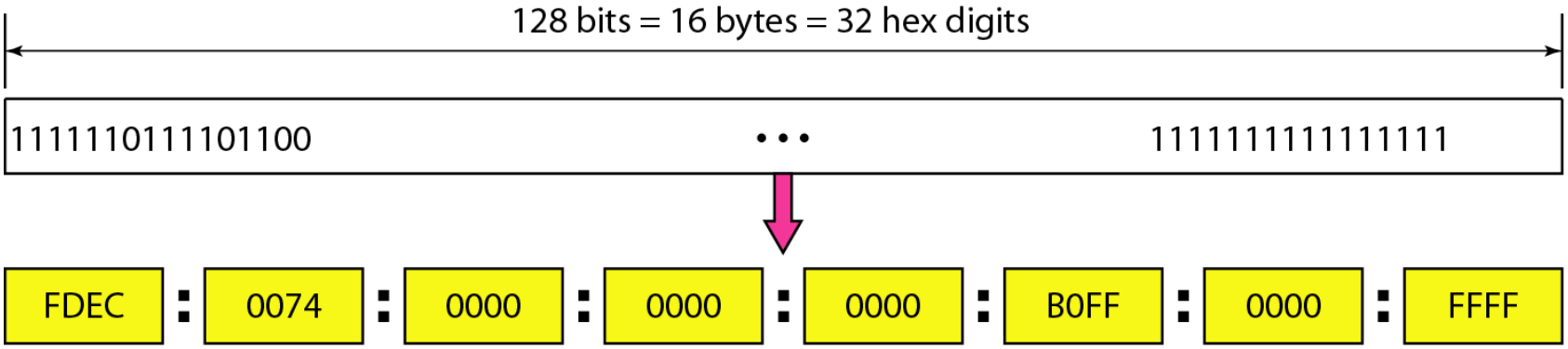
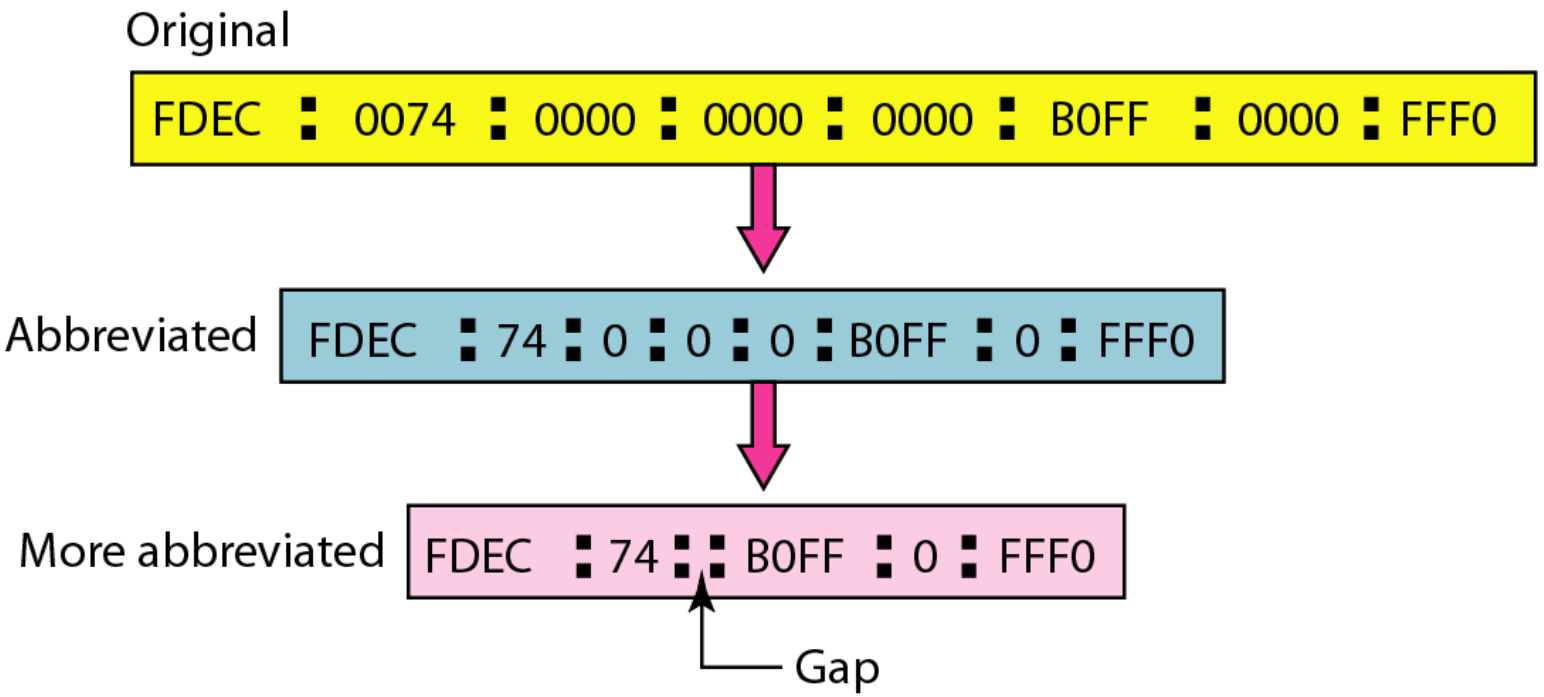


Figure 19.15 *Abbreviated IPv6 addresses*



Example 19.11

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

Solution

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

```
xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx:xxxx
  0:   15:                   :    1:   12:1213
```

This means that the original address is.

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

Subnetting Problems

- What is the Network ID, Broadcast Address, First Usable IP, or Last Usable IP on the subnetwork that the node 192.168.1.15/26 belongs to?

Step 1. Convert the shorthand subnet mask to decimal.

/26 = 255.255.255. + Two additional subnet bits.

Go to your cheat sheet, start at the bottom (128) and count up two, starting with 128. You should get to 192.

Thus, our decimal subnet mask is 255.255.255.192

Step 2. Determine the block size.

The block size is listed in the block size column parallel to your decimal mask.

The block size is 64.

Step 3. What is my Network ID?

Since we are working in the fourth octet and the block size is 64, the first network is 192.168.1.0.

Step 4. What is the next Network ID?

Again, we look at our block size of 64 to determine the next network is 192.168.1.64.

Network ID (First IP in the subnet): 192.168.1.0

Broadcast address (last IP in the subnet): 192.168.1.63

First Usable IP (the address after the network ID): 192.168.1.1

Last Usable IP (the address before the broadcast address): 192.168.1.62

- How many subnets and hosts per subnet can you get from the network 192.168.1.0 255.255.255.224?

Step 1: Determine the classful mask.

192 = Class C

Class C default mask = 255.255.255.0

Step 2: Determine how many additional subnet bits exist beyond the classful boundary.

Since this is a class C address, we are only going to look at the fourth octet. Since the decimal mask is 224, we know there were three subnet bits added (counting up from the bottom of our cheat sheet).

Step 3: Determine how many host bits remain.

We are still only working in the fourth octet. Since three bits went to the subnet instep 2, we have five host bits (0's) remaining.

Step 4: Find the exponents of both subnet bits and host bits using the cheat sheet:

Subnet Bits = 2^3 = 8

Host Bits = 2^5-2 = 30

The answer is 8 subnets and 30 hosts per subnet

- You have been asked to create a subnet mask for the 172.16.0.0 network. Your organization requires 900 subnets, with at least 50 hosts per subnet. What subnet mask should you use?

Step 1: Determine how many subnet bits (1's), you have to add to the classful boundary to cover the number of required subnets.

The IP address given was a class B address, making the first 16 subnet bits static.

- Using the cheat sheet, find the exponent of 2 that is equal to or greater than the number of subnets we require (900). We can quickly see that 10 additional subnet bits will give us 1,024 subnets. Make note of the corresponding subnet mask. In this case, 255.255.255.192. The third octet is eight 1's, and the four is two 1's. We can count up from the bottom on our cheat sheet to get to 192.

Step 2: Confirm the number of remaining 0's will cover our required hosts. In this case, there are 6 remaining 0's - 2^6-2=62, which is more than enough for our host requirements.

Our subnet mask is 255.255.255.192. Giving us 1024 subnets and 62 hosts per subnet.

- You work for a large communications corporation named GlobeComm which has been assigned a Class A network address. Currently, the company has 1,000 subnets in offices around the world. You want to add 100 new subnets over the next three years, and you want to allow for the largest possible number of host addresses per subnet.

  Which subnet mask would you choose?

- What is the network ID portion of the IP address 191.154.25.66 if the default subnet mask is used? What if /24 is used

- An organization is granted a block of addresses with the beginning address 14.24.74.0/24. The organization needs to have 3 subblocks of addresses to use in its three subnets: one subblock of 10 addresses, one subblock of 60 addresses, and one subblock of 120 addresses. Design the subblocks

Solution

There are 32 – 24 = 256 addresses in this block. The first address is 14.24.74.0/24; the last address is 14.24.74.255/24. To satisfy the third requirement, we assign addresses to subblocks, starting with the largest and ending with the smallest one.

a. The number of addresses in the largest subblock, which requires 120 addresses, is not a power of 2. We allocate 128 addresses. The subnet mask for this subnet can be found as *n1 = 32 − log2128 = 25. The first address in this block is 14.24.74.0/25; the last address is* 14.24.74.127/25.

b. The number of addresses in the second largest subblock, which requires 60 addresses, is not a power of 2 either. We allocate 64 addresses. The subnet mask for this subnet can be found as *n2 = 32 − log264 = 26. The first address in this block is 14.24.74.128/26; the last address* is 14.24.74.191/26.

c. The number of addresses in the smallest subblock, which requires 10 addresses, is not a power of 2 either. We allocate 16 addresses. The subnet mask for this subnet can be found as *n3 = 32 − log216 = 28. The first address in this block is 14.24.74.192/28; the last address is* 14.24.74.207/28.

An organization is granted the block 130.56.0.0/**16. The administrator wants** to create 1024 subnets.

- **a. Find the number of addresses in each subnet.**
- **b. Find the subnet prefix.**
- **c. Find the first and the last address in the first subnet.**
- **d. Find the first and the last address in the last subnet**

- An ISP is granted the block 80.70.56.0**/21. The ISP needs to allocate** addresses for two organizations each with 500 addresses, two organizations each with 250 addresses, and three organizations each with 50 addresses.
- **a. Find the number and range of addresses in the ISP block.**
- **b. Find the range of addresses for each organization and the range of unallocated** addresses