Data Communications and Networking Fourth Edition

Forouzan

Chapter 20

Network Layer: Internet Protocol

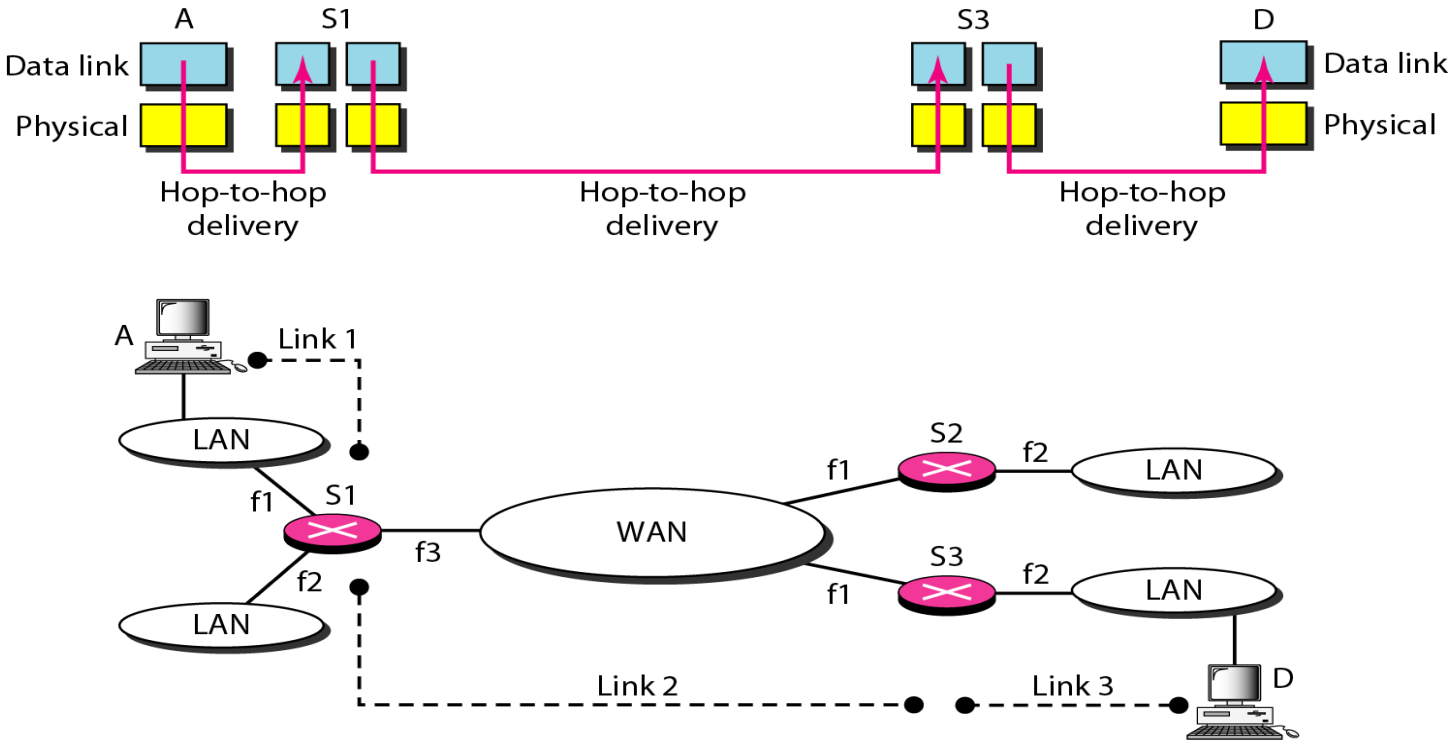
20-1 INTERNETWORKING

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

Topics discussed in this section:

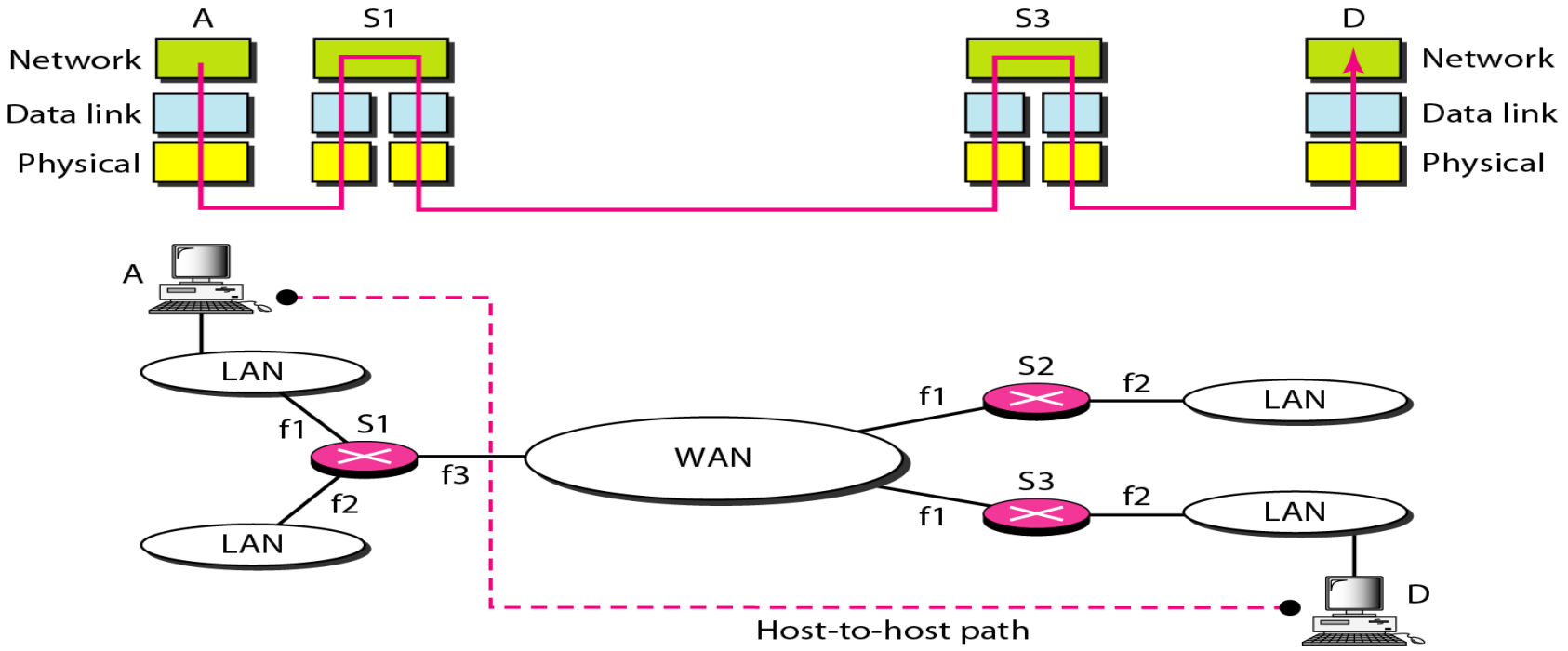
Need for Network Layer
Internet as a Datagram Network
Internet as a Connectionless Network

Figure 20.1 *Links between two hosts*



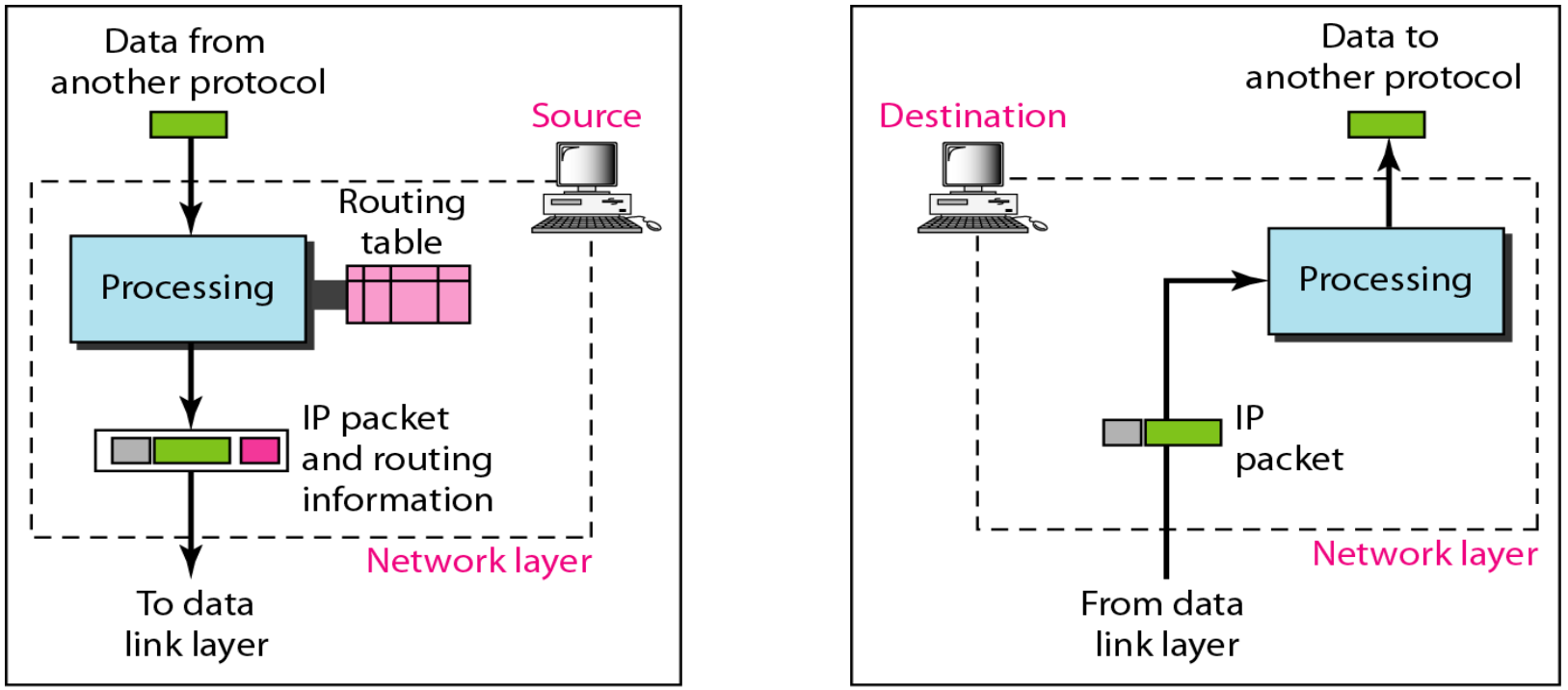
- ✓ Physical and data link layers of a network are jointly responsible for data delivery on the network from one node to the next.
- ✓ In each link, two physical and two data link layers are involved.
- ✓ If S1 is a layer-2 devices, then it has no information about the address/host D on a different network.

Figure 20.2 *Network layer in an internetwork*



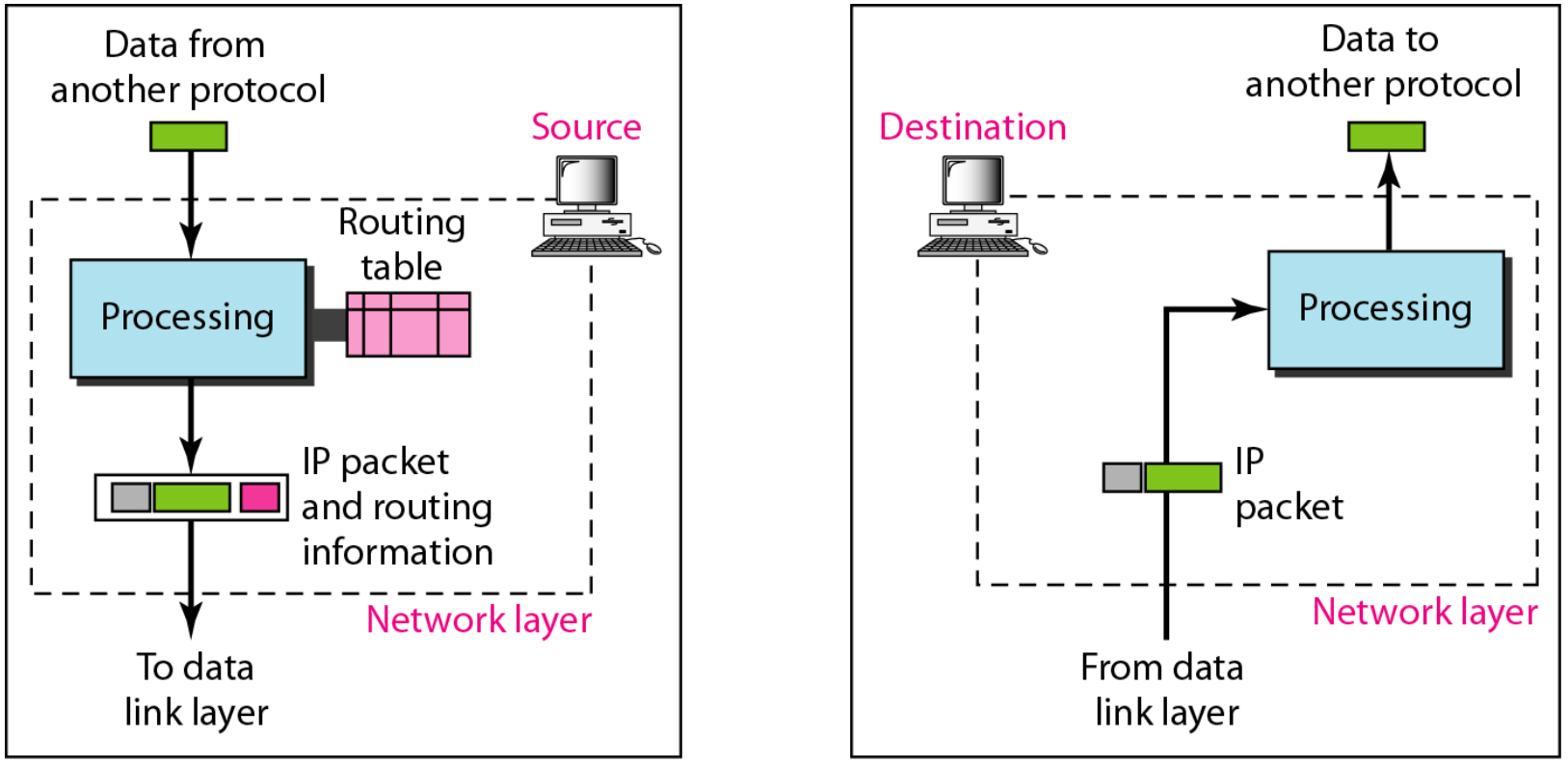
- ✓ The **network layer is responsible for host-to-host delivery and for routing the packets** through the routers or switches

Figure 20.3 *Network layer at the **source***



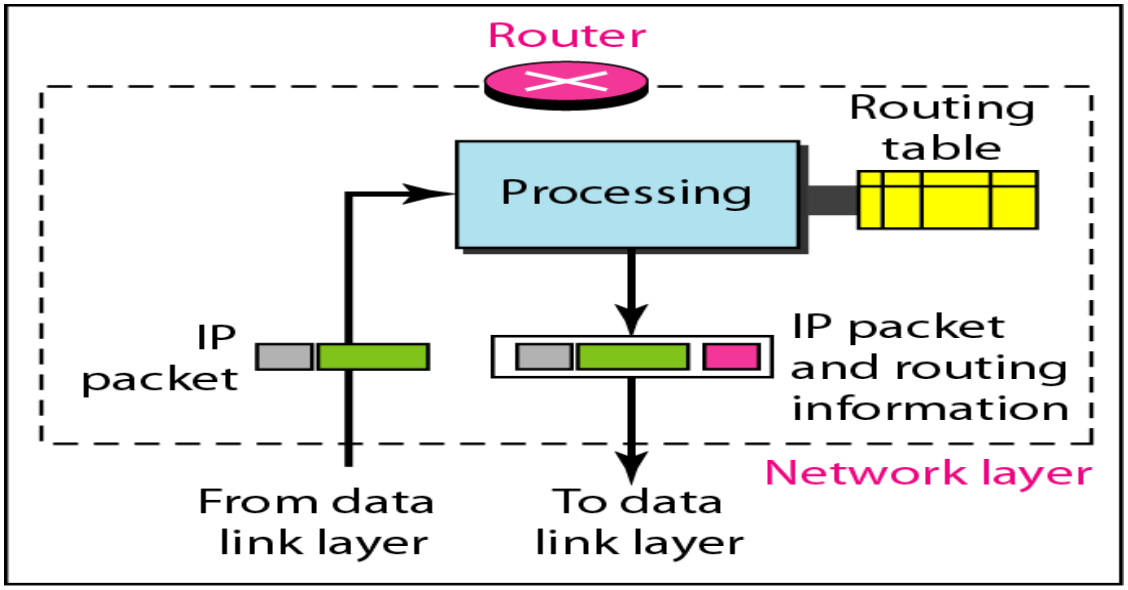
- ✓ The network layer at the source is **responsible for creating a packet** from the data coming from another protocol (such as a **transport layer protocol** or a routing protocol)
- ✓ The **header of the packet contains**, information, such as the **logical addresses** of the source and destination

Figure 20.3 *Network layer at the **source***



- ✓ The network layer is **responsible for checking routing table to find routing information (such as outgoing interface of the packet or the physical address of the next node)**
- ✓ If the **packet is too large**, the packet is **fragmented**

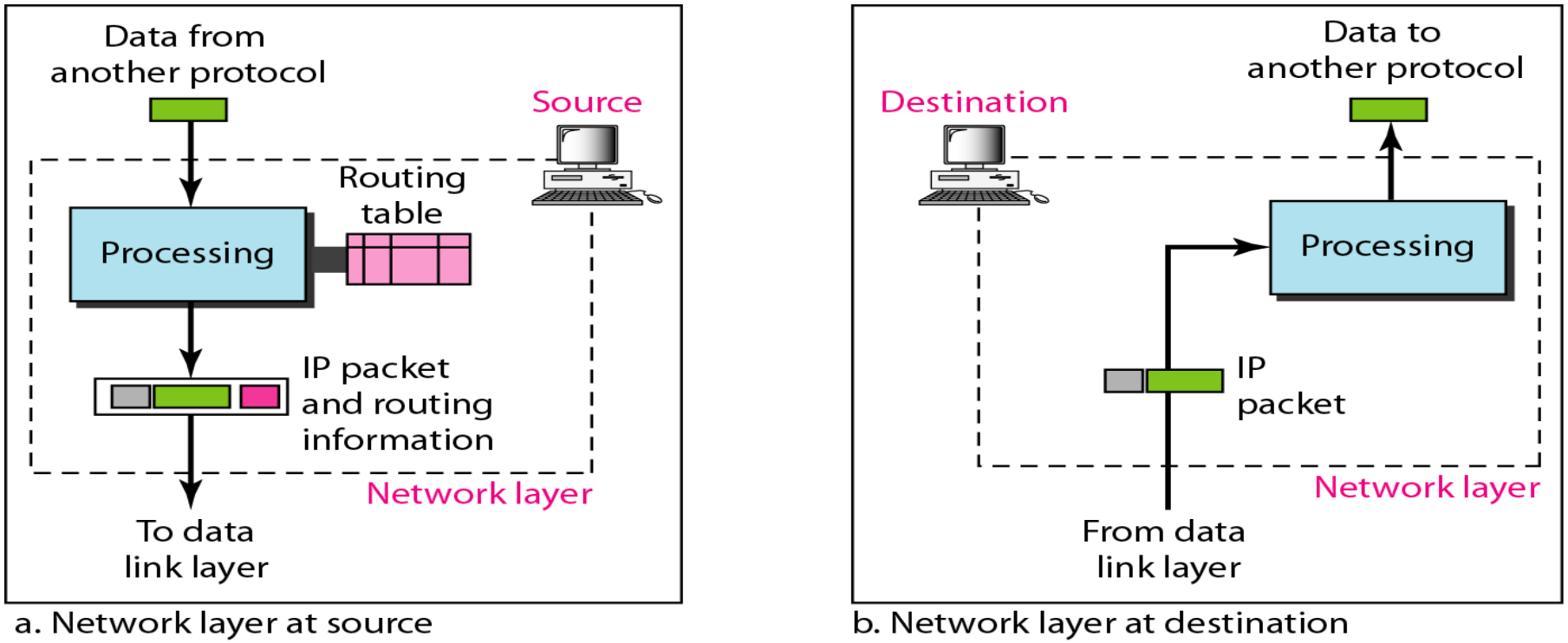
Figure 20.3 *Network layer at the router (continued)*



c. Network layer at a router

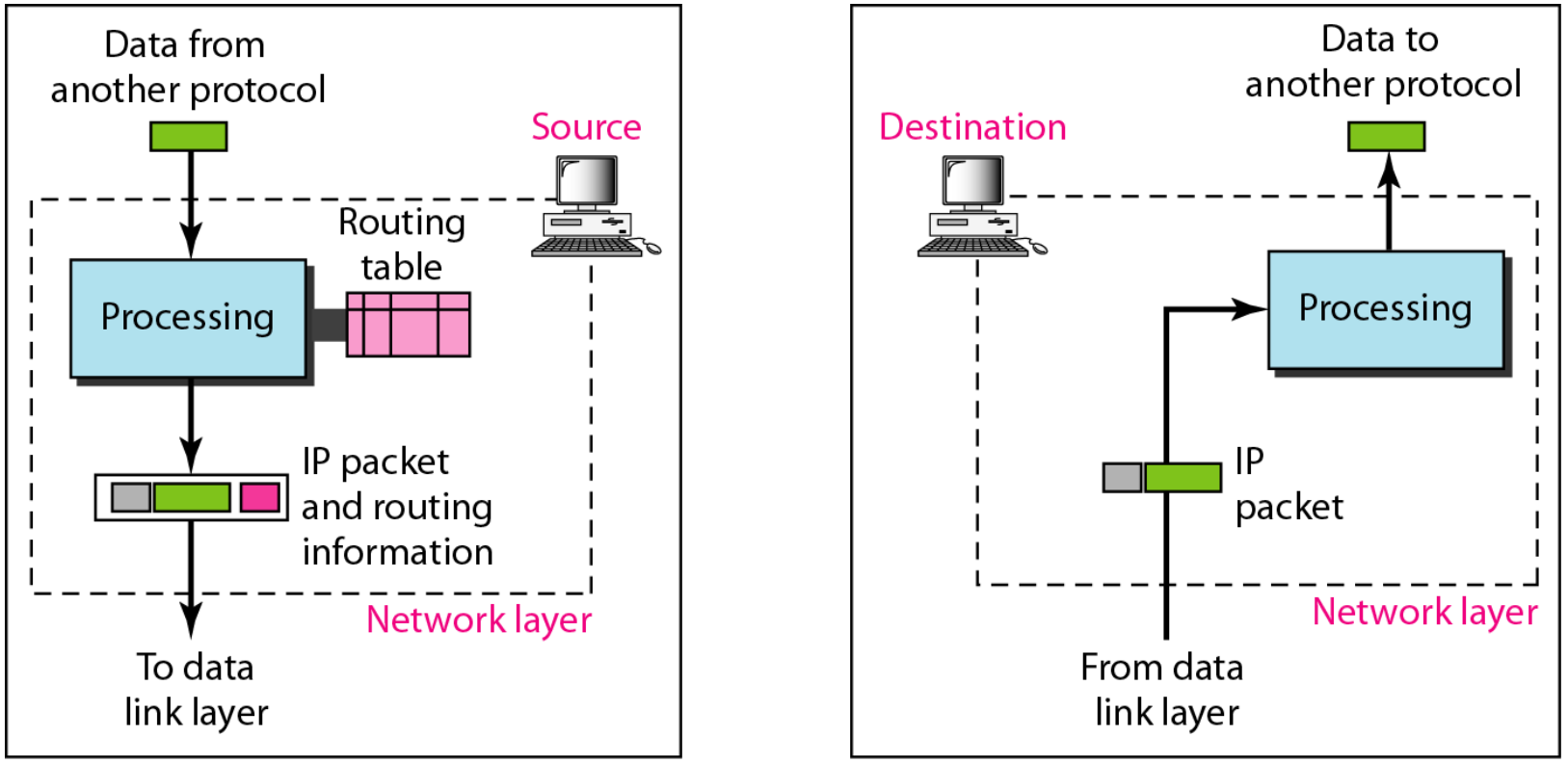
- ✓ When a **packet arrives**, the **router** or L3-switch **consults its routing table** and **finds the interface** from which the **packet must be sent**.
- ✓ The packet, after some **changes** in the **header**, with the routing information is **passed to the data link layer again (for next hop)**.

Figure 20.3 *Network layer at the **destination***



- ✓ The network layer at the **destination** is **responsible for address verification**
- ✓ It makes sure that the destination address on the packet is the same as the address of the host.

Figure 20.3 *Network layer at the **destination***



a. Network layer at source

b. Network layer at destination

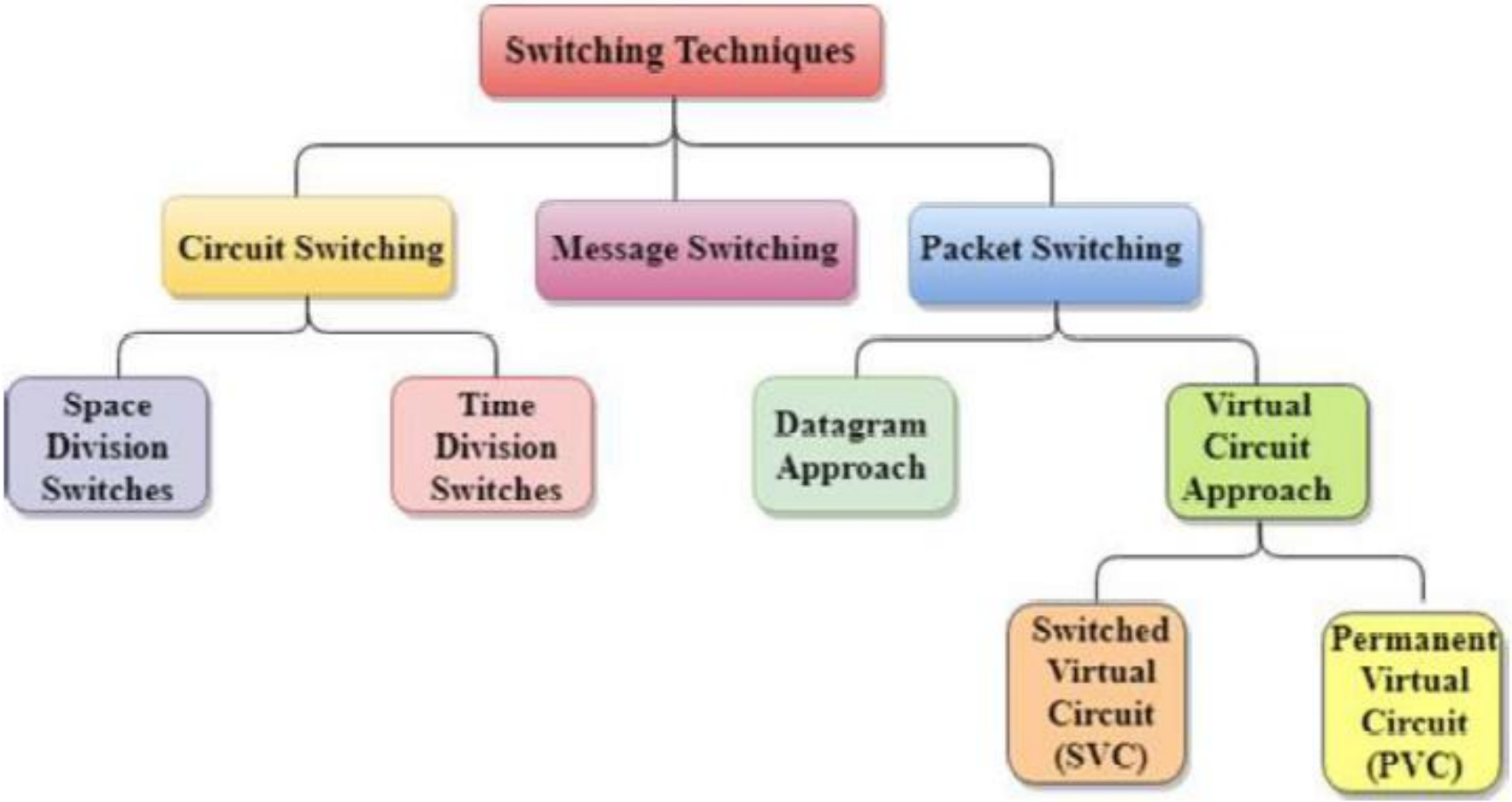
- ✓ **If the packet is a fragment**, the **network layer waits until all fragments have arrived**, and then **reassembles them** and **delivers the reassembled packet** to the **transport layer**

Note

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

Recall

Classification of Switching Techniques



Internet as a Connectionless Network

- ✓ Packet delivery can be done either by using:
  - ✓ a **connection-oriented** network service or
  - ✓ a **connectionless** network service.

Internet as a Connectionless Network

- **Connection-oriented service:**
  - The source first **makes a connection** with the destination before sending a packet.
  - When the connection is **established**, a **sequence of packets** from the **same source** to the **same destination** can be **sent** one after another.
  - Packets are sent on the same path in sequential order.
  - When all packets of a message have been delivered, the **connection is terminated.**
  - Example of such a **virtual-circuit approach** to packet switching are in Frame Relay and ATM.

Internet as a Connectionless Network

- **Connectionless service:**
  - The network layer protocol treats **each packet independent**, having no relationship to any other packet.
  - The packets in a message **may or may not travel the same path to their destination**.
  - This type of service is used in the **datagram approach to packet switching**.
  - **The Internet** has chosen this type of service at the network layer.
  - **Reason for this decision: the Internet** is made of many heterogeneous networks, thus, impossible to create a connection from source to destination without knowing the nature of the networks in advance.

Note

Communication at the network layer in the Internet is connectionless.

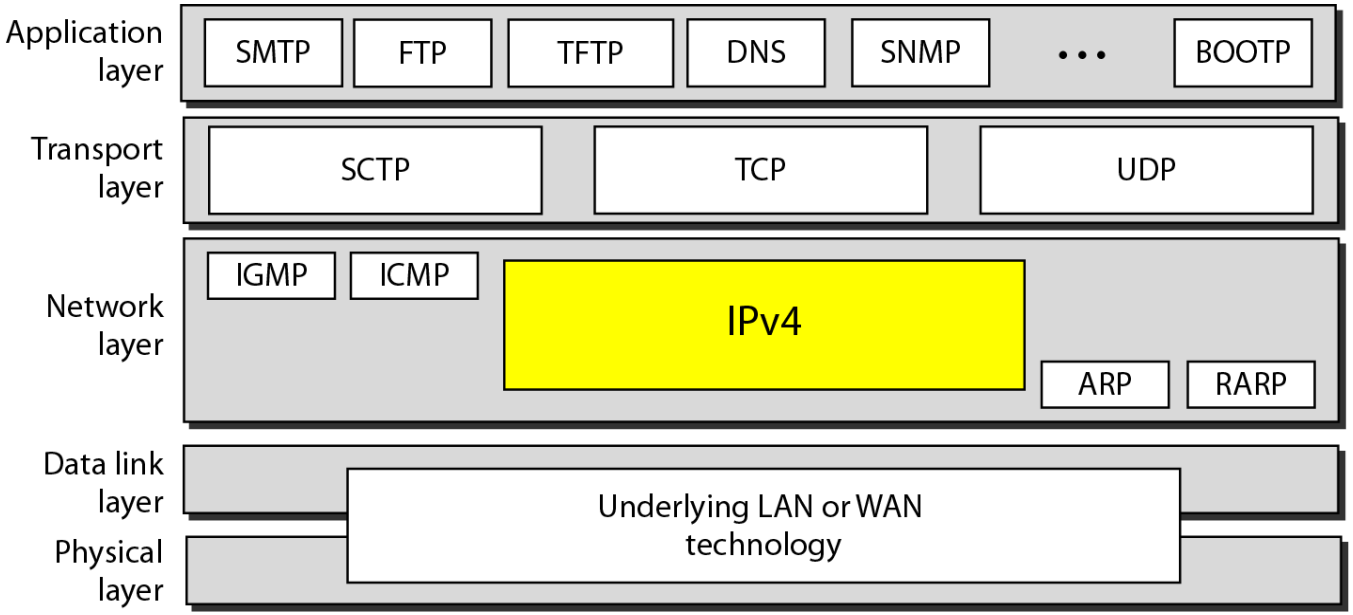
20-2 IPv4

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

Topics discussed in this section:

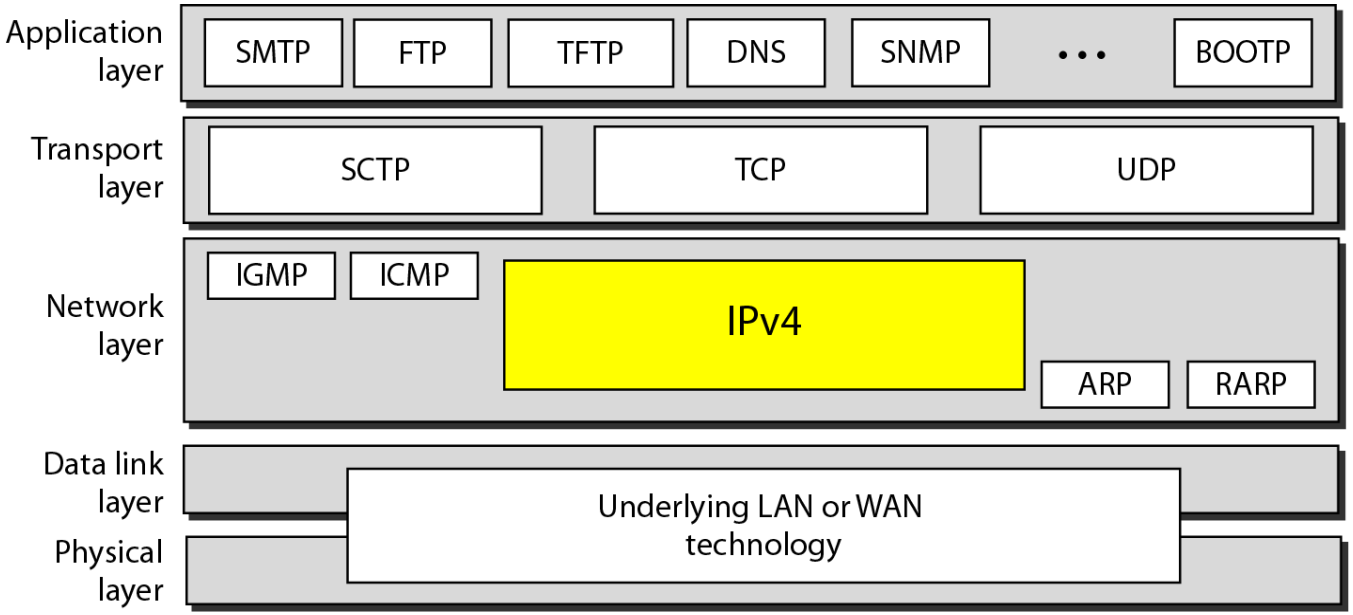
Datagram
Fragmentation
Checksum
Options

Figure 20.4 *Position of IPv4 in TCP/IP protocol suite*



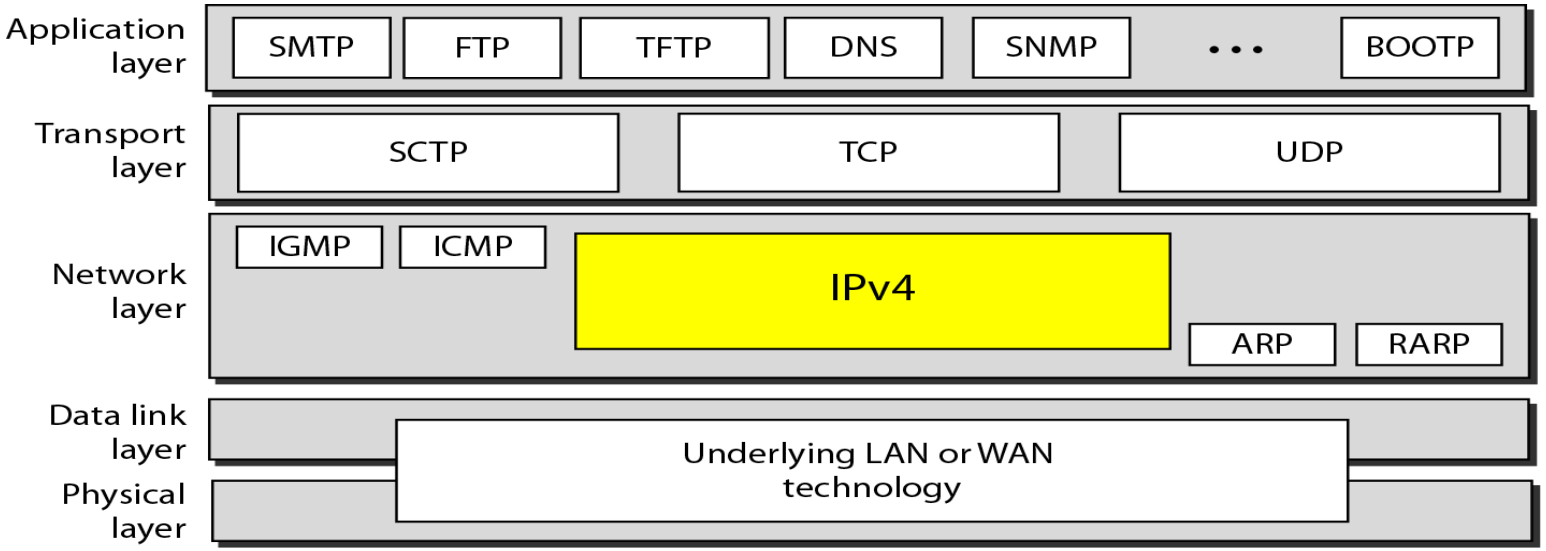
- ✓ IPv4 is an **unreliable and connectionless datagram protocol**.
- ✓ It provides *best-effort* service without error control or flow control, except for error detection on the header.
- ✓ IPv4 assumes that lower layers may be unreliable and tries its best to **deliver data, but with no guarantees of success**.

Figure 20.4 *Position of IPv4 in TCP/IP protocol suite*



- ✓ If reliability is important, IPv4 must be paired with a reliable protocol such as TCP
- ✓ Example: post office letter delivery system

Figure 20.4 *Position of IPv4 in TCP/IP protocol suite*

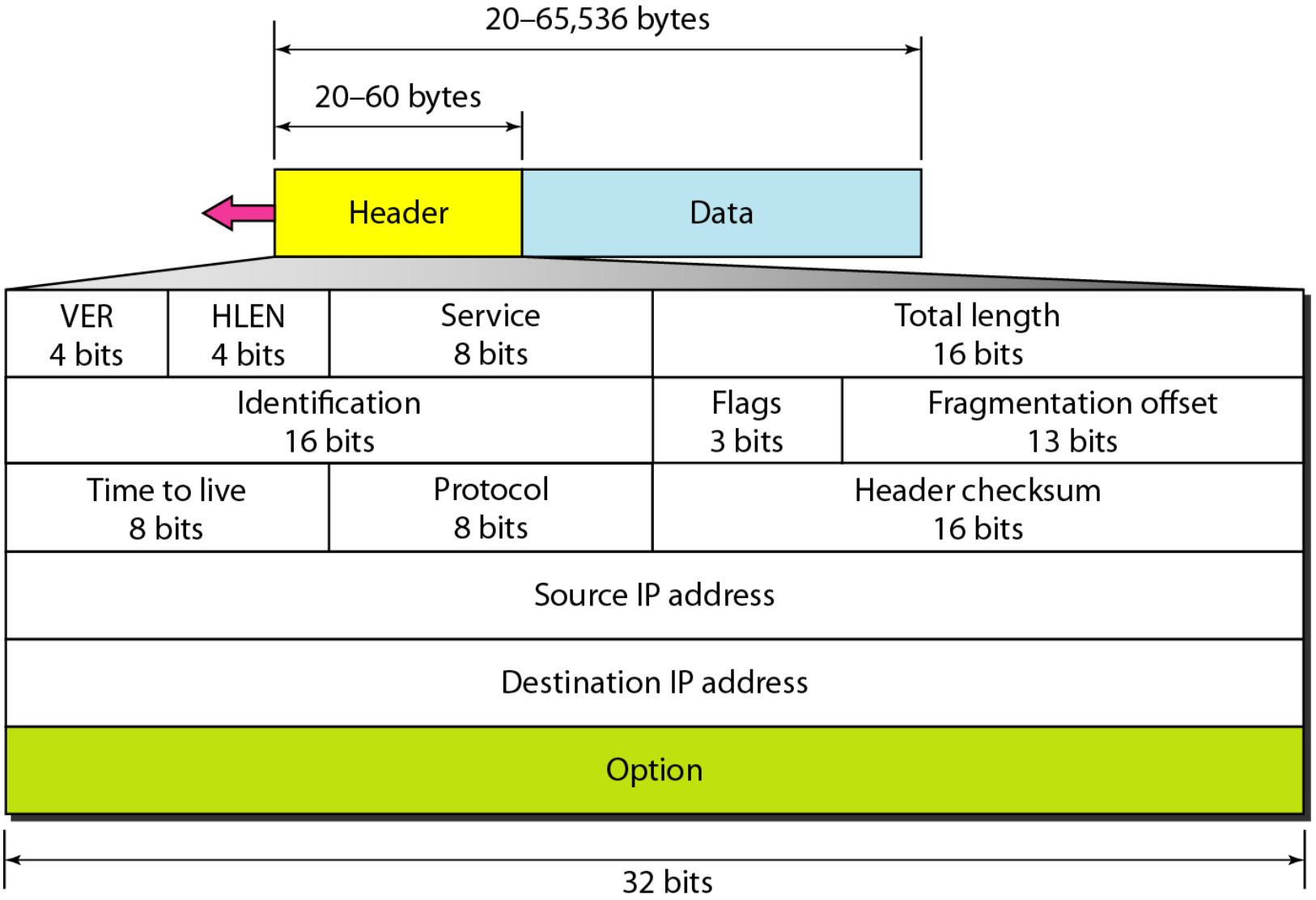


- ✓ **IPv4 is connectionless**, meaning each **data packet (datagram)** is handled separately and can take different routes.
- ✓ Packets may arrive out of order, be lost, or corrupted.
- ✓ IPv4 **depends on higher-level protocols** to fix these issues.

Datagram

- ✓ Packets in the **IPv4 layer are called datagrams**.

Figure 20.5 IPv4 Datagram Format



Datagram Format

- ✓ 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 (32-bit) sections**.
- ✓ A brief description of each field:
- **Ver: 4-bit** field defines the version of the IPv4 protocol. However, version 6 may totally replace version 4 in the future.
- **Header length (HLEN)**. **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 x 4 = 20). When the option field is at its maximum size, the value of this field is 15 (15 x 4 = 60).

Datagram Format

- **Services:** Internet Engineering Task Force (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.

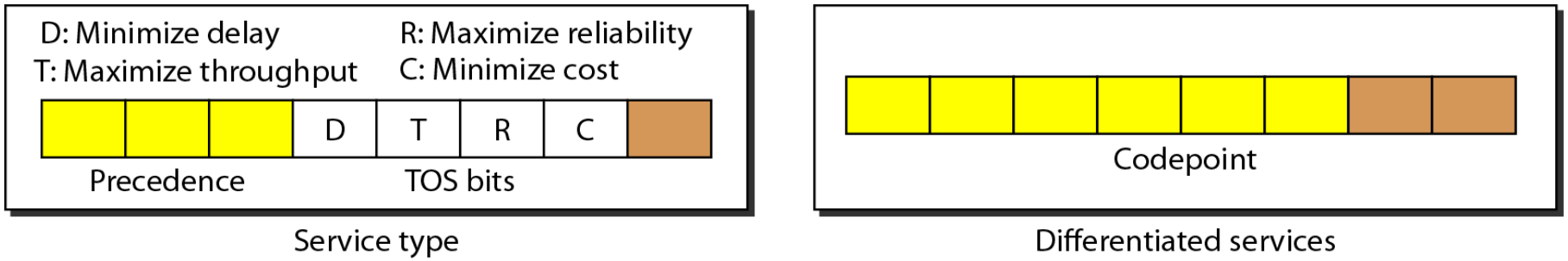
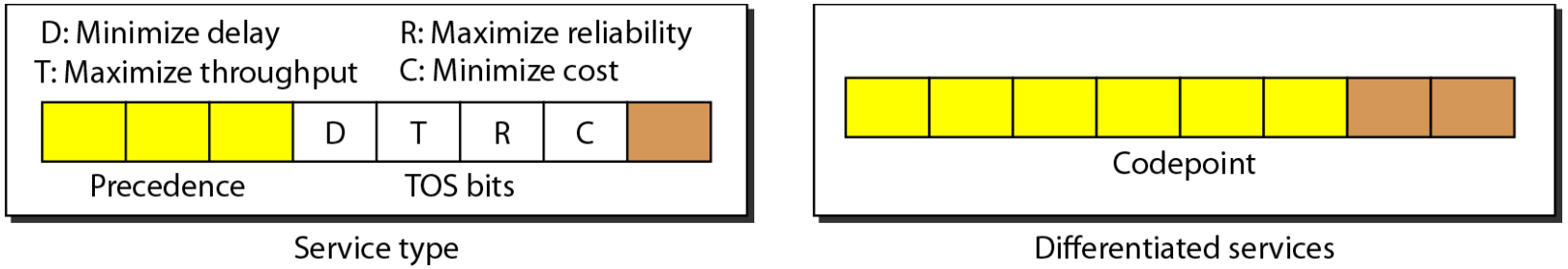


Figure 20.6 ***Service type*** ***or differentiated services***

Figure 20.6 *Service type* *or differentiated services*



Service Type (in this interpretation):

- ✓ **Precedence bits (**first **3 bits): defines priority from** 0 (lowest) to 7 (highest). **Routers** discard lower-precedence datagrams first during **congestion**.
- ✓ **Type of Service (TOS)** bits (next **4 bits):**
  - ✓ **Specify quality preferences like minimizing delay, maximizing throughput, or ensuring high reliability..**
  - ✓ **Only one bit can be set to 1 in each datagram, allowing for five distinct types of services.**
- ✓ **Last bit is unused**.

Table 20.1 *Types of service*

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

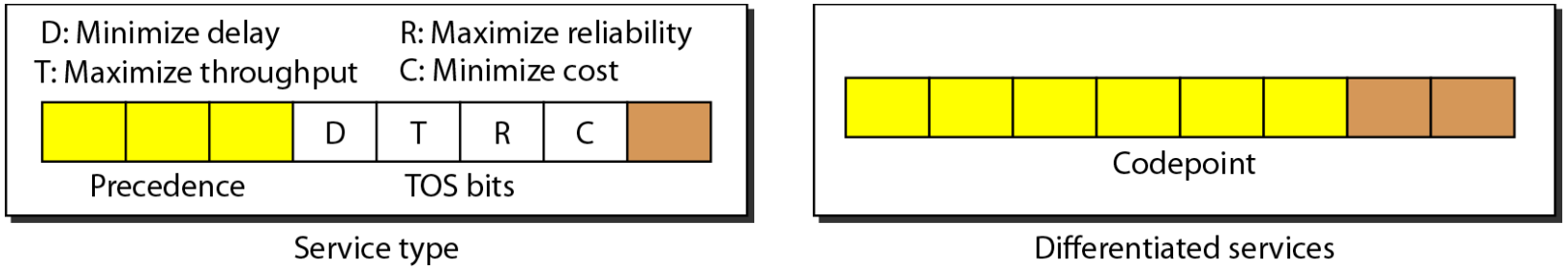
Five distinct types of services

Application programs can request a specific type of service. The defaults for some applications are shown in Table 20.2.

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

Table 20.2 ***Default types of service***

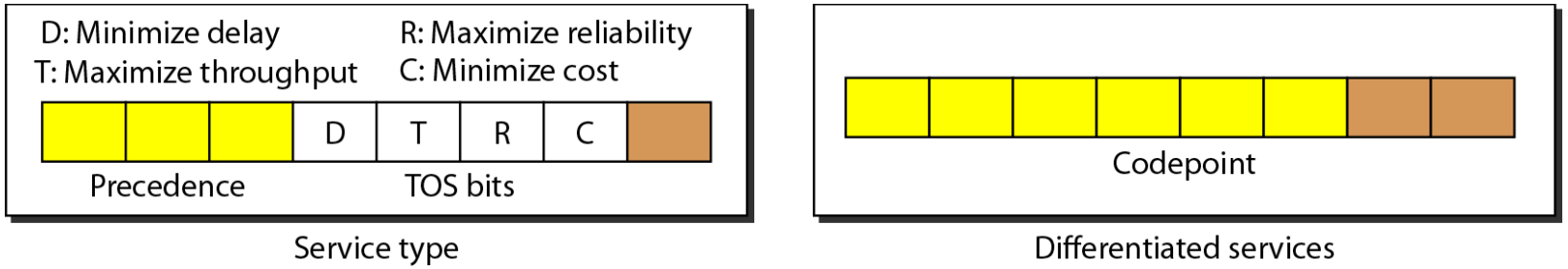
Figure 20.6 *Service type* *or differentiated services*



Service Type (in this interpretation):

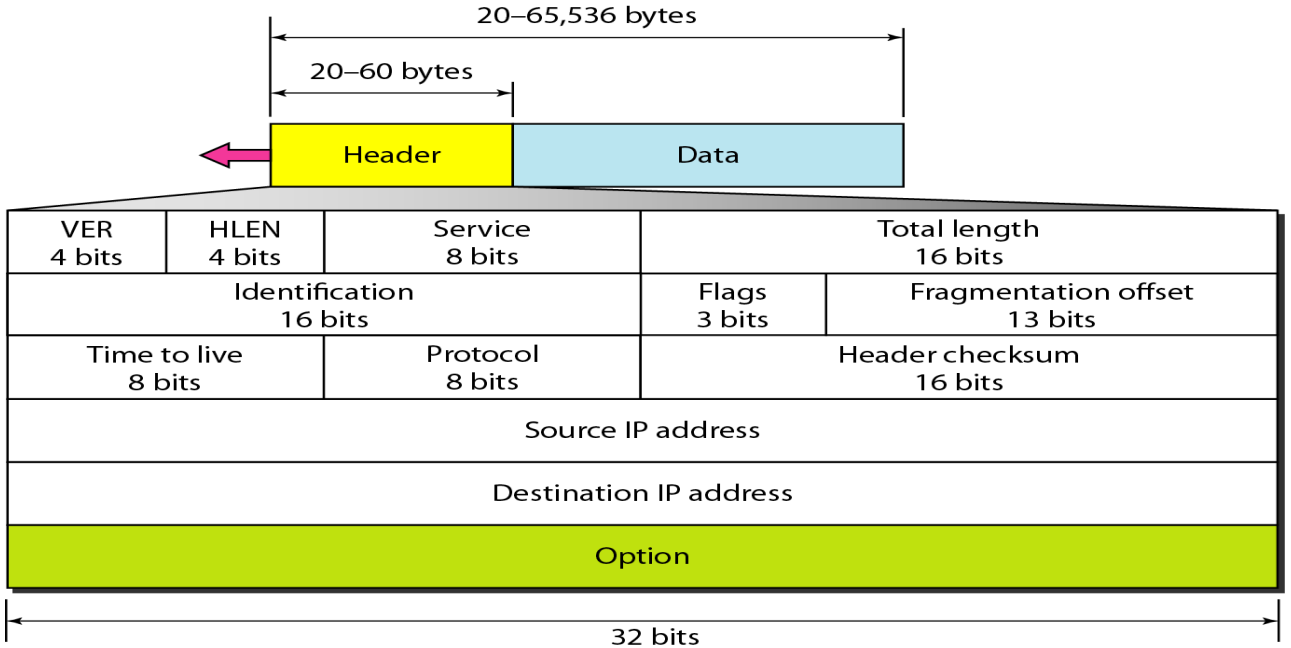
- ✓ **Interactive and urgent** activities require **minimum delay** (immediate attention).
- ✓ **Bulk data transmission** requires **maximum throughput**.
- ✓ **Network management** activities needs **maximum reliability**.
- ✓ **Background tasks** need **minimum cost**.

Figure 20.6 *Service type or differentiated services*



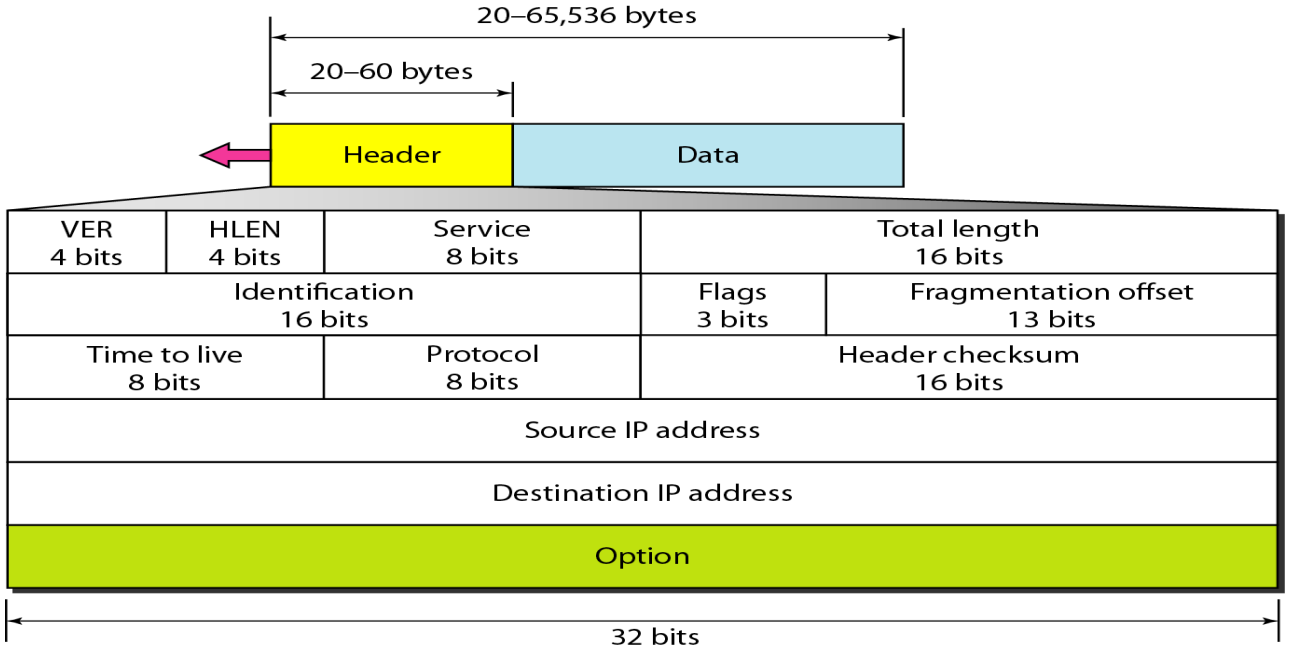
Differentiated Services (in this interpretation):

- ✓ **Codepoint subfield: First 6 bits;** Last 2 bits are unused.
- ✓ **Two interpretations:**
  1. If the last 3 bits are 0s, the first 3 bits function like the old precedence bits.
  2. If the last 3 bits are non-zero, all **6 bits define 64 service levels**:
     - ✓ Category 1 (XXXXX0): 32 services (**assigned by IETF**).
     - ✓ Category 2 (XXXX11): 16 services (used by **local organizations**).
     - ✓ Category 3 (XXXX01): 16 services (**for experiments**).

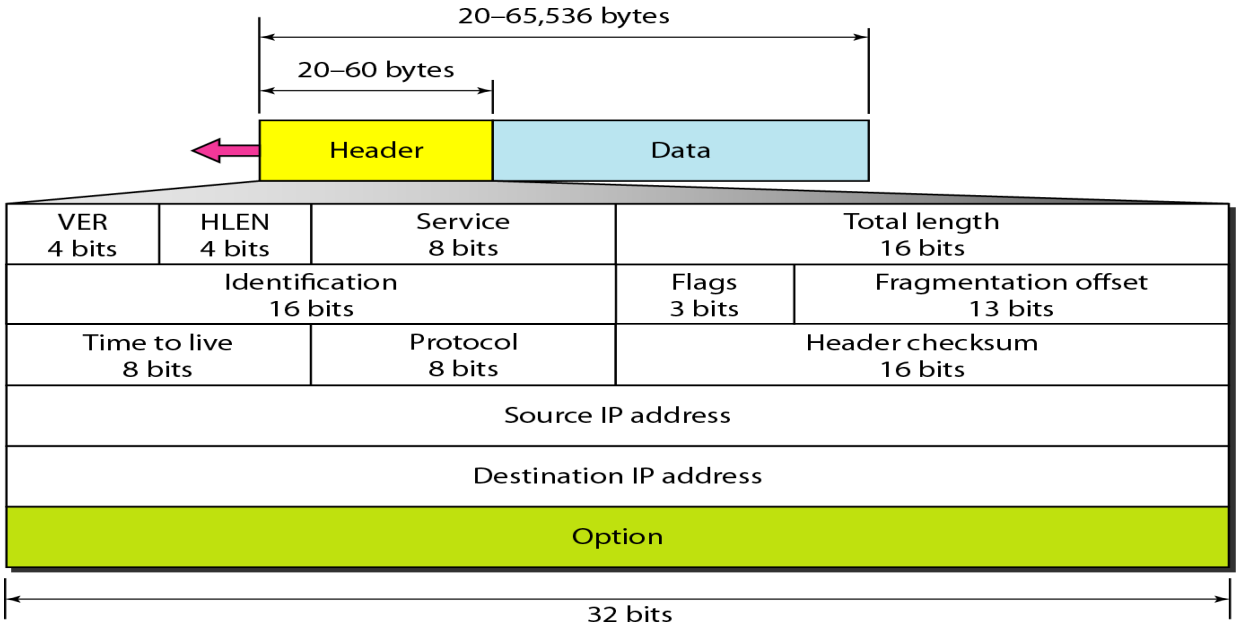


- ✓ **Total length Field**: 16-bit field representing the total size of the IPv4 datagram (header + data) in bytes.
- ✓ **Data Length**:

  Length of data = total length - header length
- ✓ **Header Length**: Calculated by multiplying the HLEN value by 4 bytes(32bits).
- ✓ **Max Datagram Size**: **65,535 bytes**, with the **header** ranging from **20 to 60 bytes**.
- ✓ **Future Outlook**: Larger datagram sizes might be supported as network technologies evolve and provide higher bandwidth.



- ✓ **Identification, Flags, and Fragmentation offset:** are fields used in fragmentation.
- ✓ **Time to live:** Limits the datagram's lifetime. Originally meant as a timestamp, it now indicates the **maximum number of hops** a datagram can take. Each router **decrements the value by 1**. When it **reaches zero**, the **datagram is discarded**. TTL also helps limit datagrams to local networks if needed. For example, setting the value to 1 restricts travel to one hop before being discarded.



- ✓ **Protocol**: This 8-bit field identifies the higher-level protocol (e.g., TCP, UDP, ICMP) using IPv4 services. It helps the receiving layer know which protocol the data belongs to.
- ✓ **Checksum**: Used for error checking of the IPv4 header.
- ✓ **Source Address**: A 32-bit field containing the IPv4 address of the **sender**, **unchanged** throughout transmission.
- ✓ **Destination Address**: A 32-bit field holding the **recipient's** IPv4 address, also unchanged during transmission.

Table 20.3 *Protocol values*

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

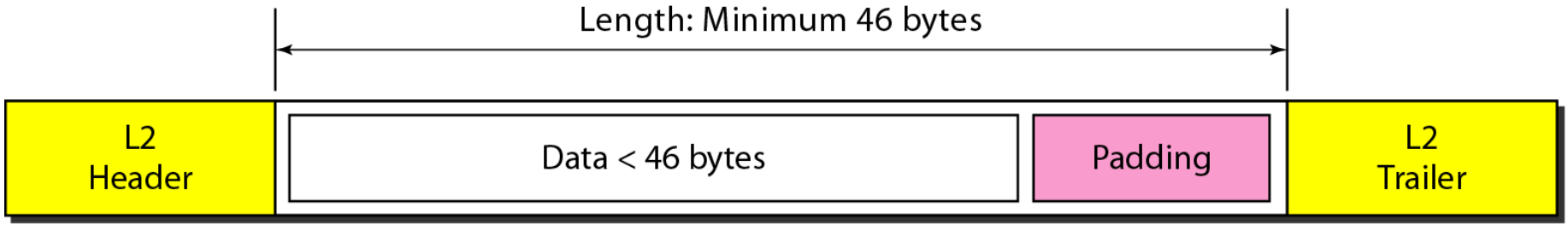
Note

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

Note

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

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



- ✓ In some cases, physical networks cannot handle large IPv4 datagrams (up to 65,535 bytes), so they must be fragmented to pass through these networks.
- ✓ Although most networks do not need the "total length" field, **some do**.
- ✓ For example, Ethernet frames have size restrictions (46-1500 bytes). If the datagram is smaller than 46 bytes, padding is added to meet this size.
- ✓ When the datagram is decapsulated, the receiver uses the total length field to distinguish the real data from the padding added for Ethernet size requirements.

Figure 20.8 *Protocol field and encapsulated data*

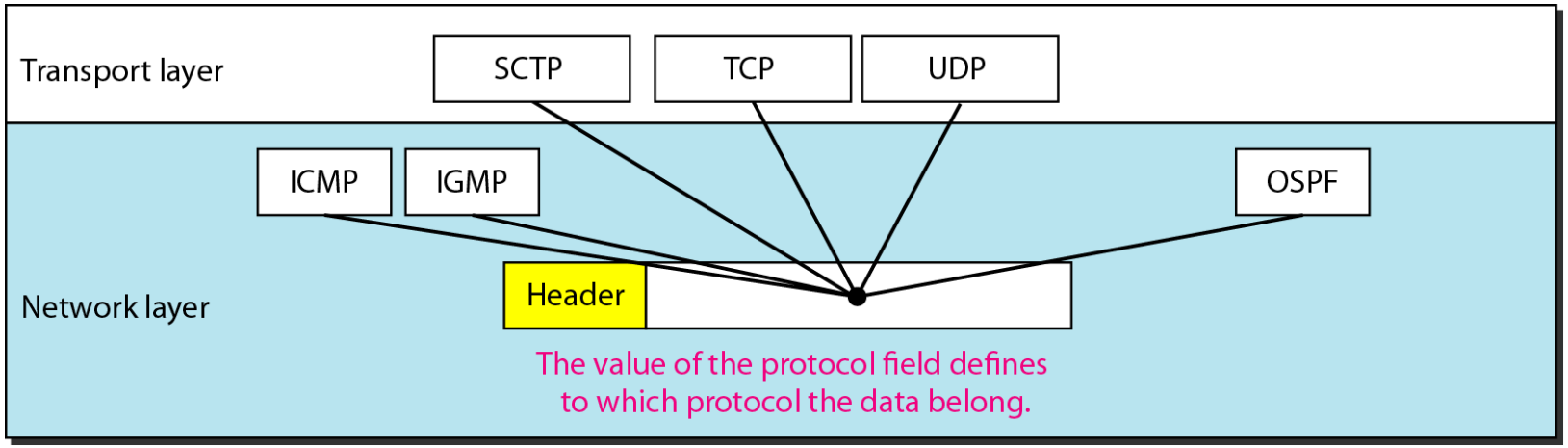
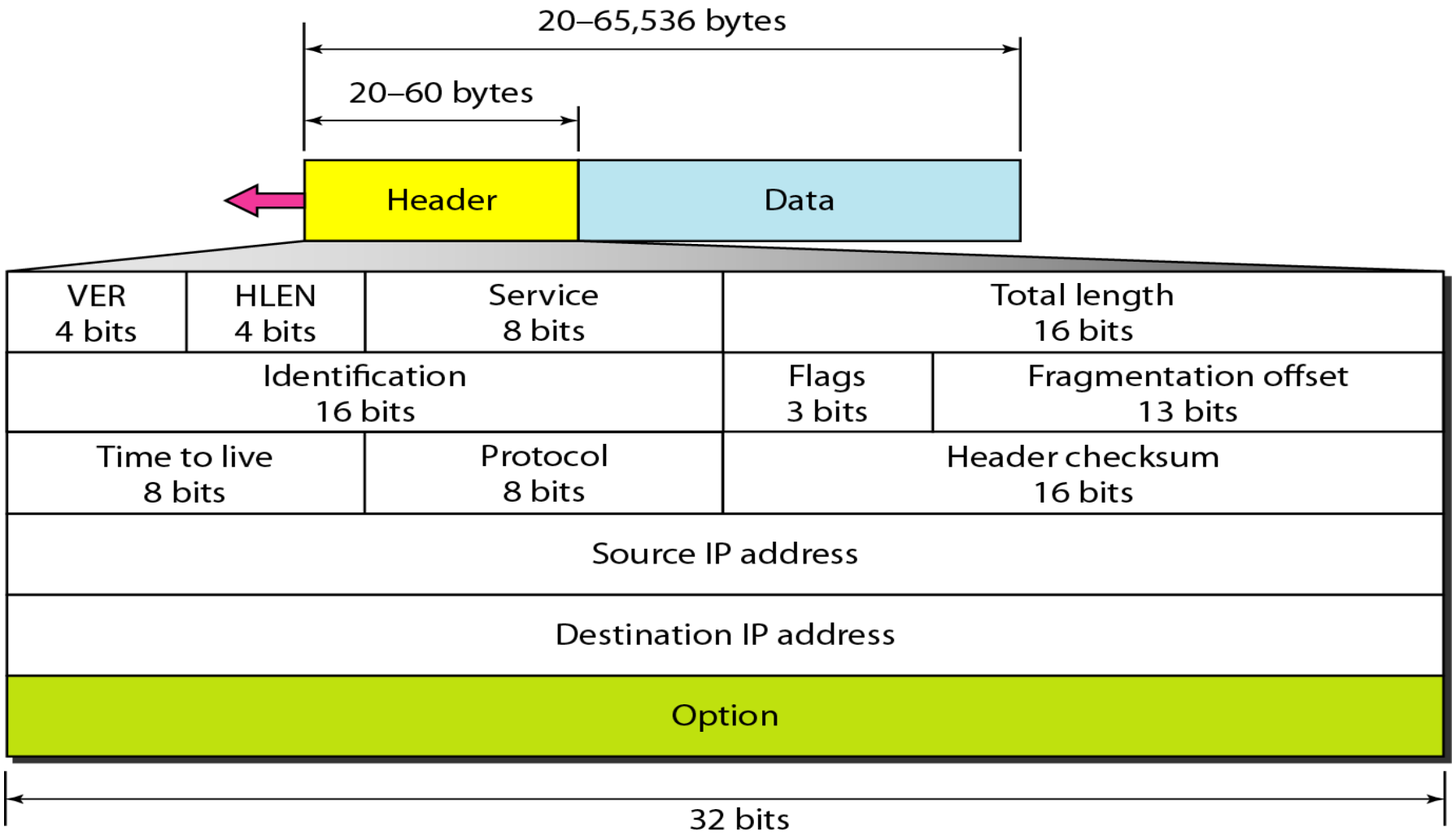


Table 20.4 ***Protocol values***

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

Example


20–65,536 bytes
20–60 bytes
Header
Data
VER
4 bits
HLEN
4 bits
Service
8 bits
Total length
16 bits
Identification
16 bits
Flags
3 bits
Fragmentation offset
13 bits
Time to live
8 bits
Protocol
8 bits
Header checksum
16 bits
Source IP address
Destination IP address
Option
32 bits

Example 20.1

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

01000010

The receiver discards the packet. Why?

Solution

There is an error in this packet. The 4 leftmost bits (0100) show the version, which is correct.

The next 4 bits (0010) show an invalid header length (2 × 4 = 8).

The minimum number of bytes in the header must be 20. The packet has been corrupted in transmission.

Example 20.2

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

Solution

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

Example 20.3

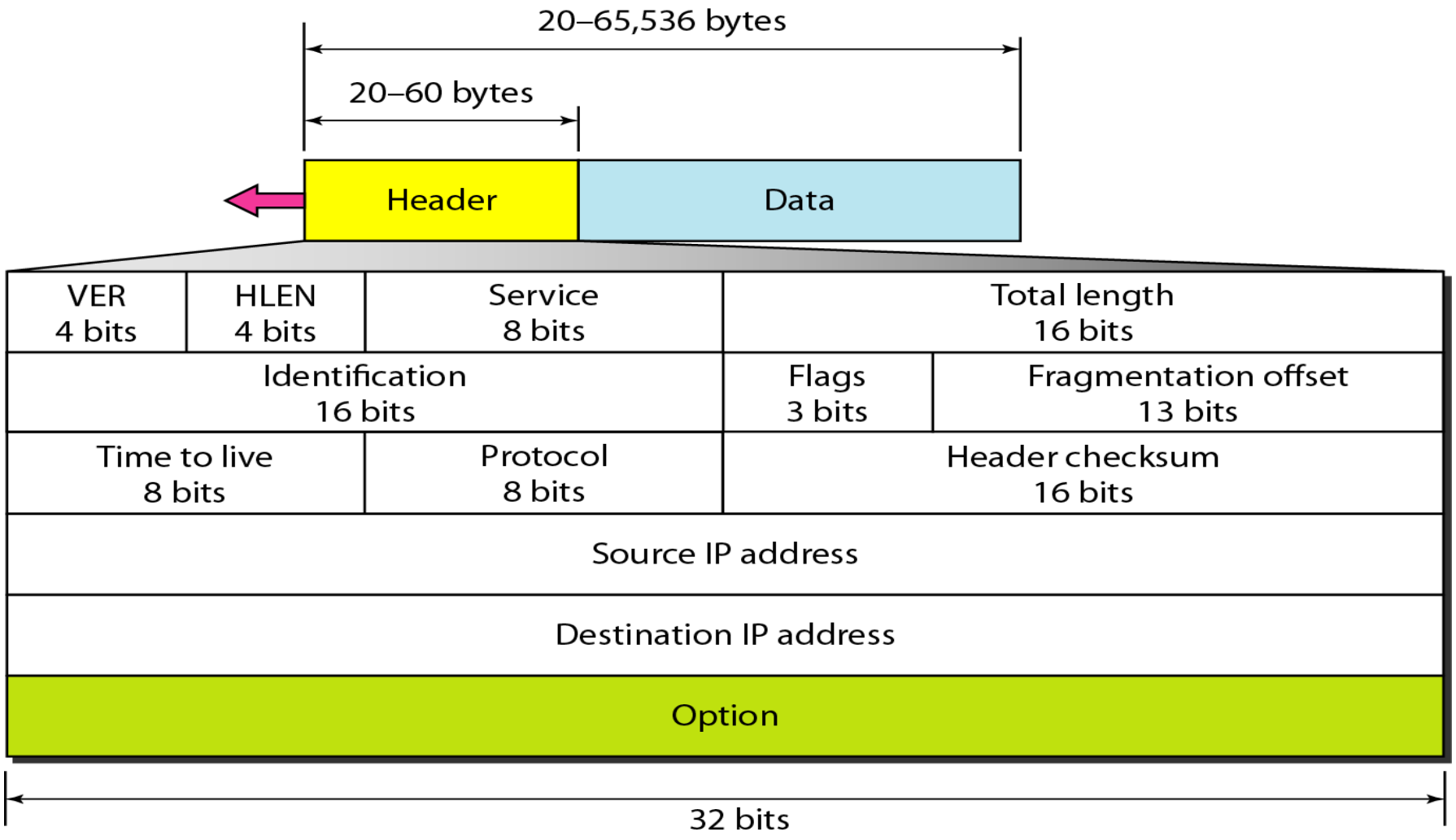
In an IPv4 packet, the value of HLEN is 5, and the value of the total length field is 0x0028. How many bytes of data are being carried by this packet?

Solution

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

Hexadecimal (0x0028) = *0*16^3 + 0*16^2 + 2*16^1 + 8*16^0*
Decimal = *0 + 0 + 2 * 16 + 8 * 1 => 32 + 8 => 40*

IPv4 Datagram Format



Example 20.4

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

0x45000028000100000102 . . .

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

Solution

To find the time-to-live field, we skip 8 bytes (32+32 bits). 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.

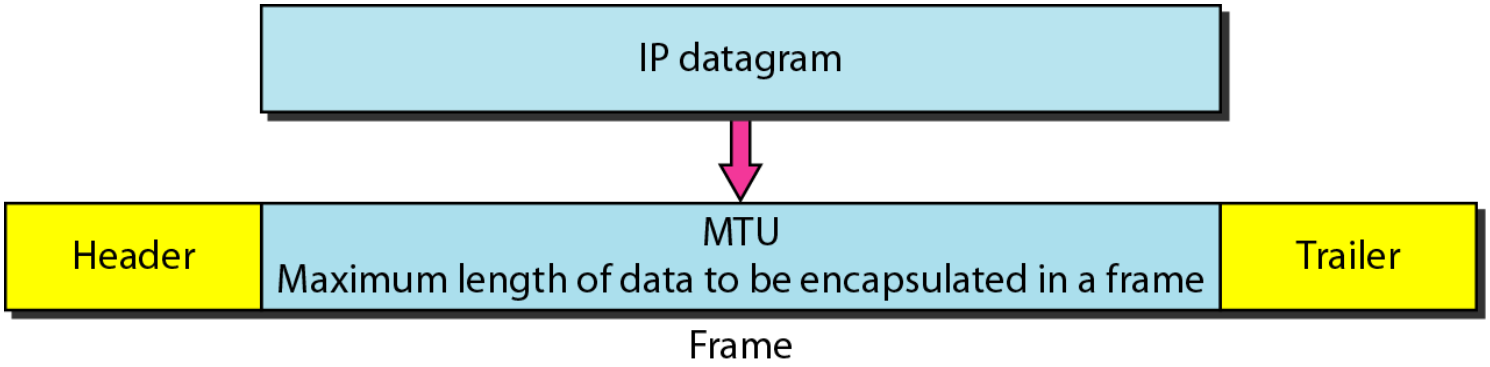
Each Hexadecimal character represent 4 bits. Therefore, 2 hexa-characters represent 1 byte.

0x45	00	00	28	00	01	00	00	01	02	. . .

Fragmentation

- ✓A datagram travels through various networks, each router re-encapsulating it.
- ✓Routers decapsulate the datagram from one frame format and encapsulate it in another.
- ✓Frame size and format depend on the protocols of the networks involved.
- ✓For instance, a router linking a LAN and WAN will receive a LAN-format frame and send it in WAN format.

Figure 20.9 *Maximum transfer unit (MTU)*



- ✓ Each data link layer protocol has a **unique frame format**.
- ✓ The format includes a **maximum data field size**, defining the largest datagram it can encapsulate.
- ✓ This **maximum size** (**MTU**) is **determined by network hardware and software limits**.
- ✓ The **MTU value varies by network protocol**, as shown in protocol-specific tables.

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

Fragmentation

- ✓ **IPv4 Datagram Size: Maximum length is 65,535 bytes.**
- ✓ Designed to work across various network types.
- ✓ IPv4 designed to support MTU (Maximum Transmission Unit) of 65,535 bytes.
- ✓ However, for other physical networks, we must divide the datagram to make it possible to pass through these networks. This is called **fragmentation**.
- ✓ The **source (network-layer)** usually **does not fragment** the IPv4 packet.
- ✓ The **transport layer will** instead **segment the data into** a size **that can be accommodated by IPv4 and the data link layer** in use.

Fragmentation

When a datagram is **fragmented**:

- ✓ Each fragment has its own header, with most fields repeated but some modified.
- ✓ A fragment can be fragmented again (multiple fragmentation) if it encounters a smaller MTU.
- ✓ Fragmentation Responsibility: Can be done by the source host or routers along the path, but usually at the **source**.

Fragmentation

Reassembly Process:

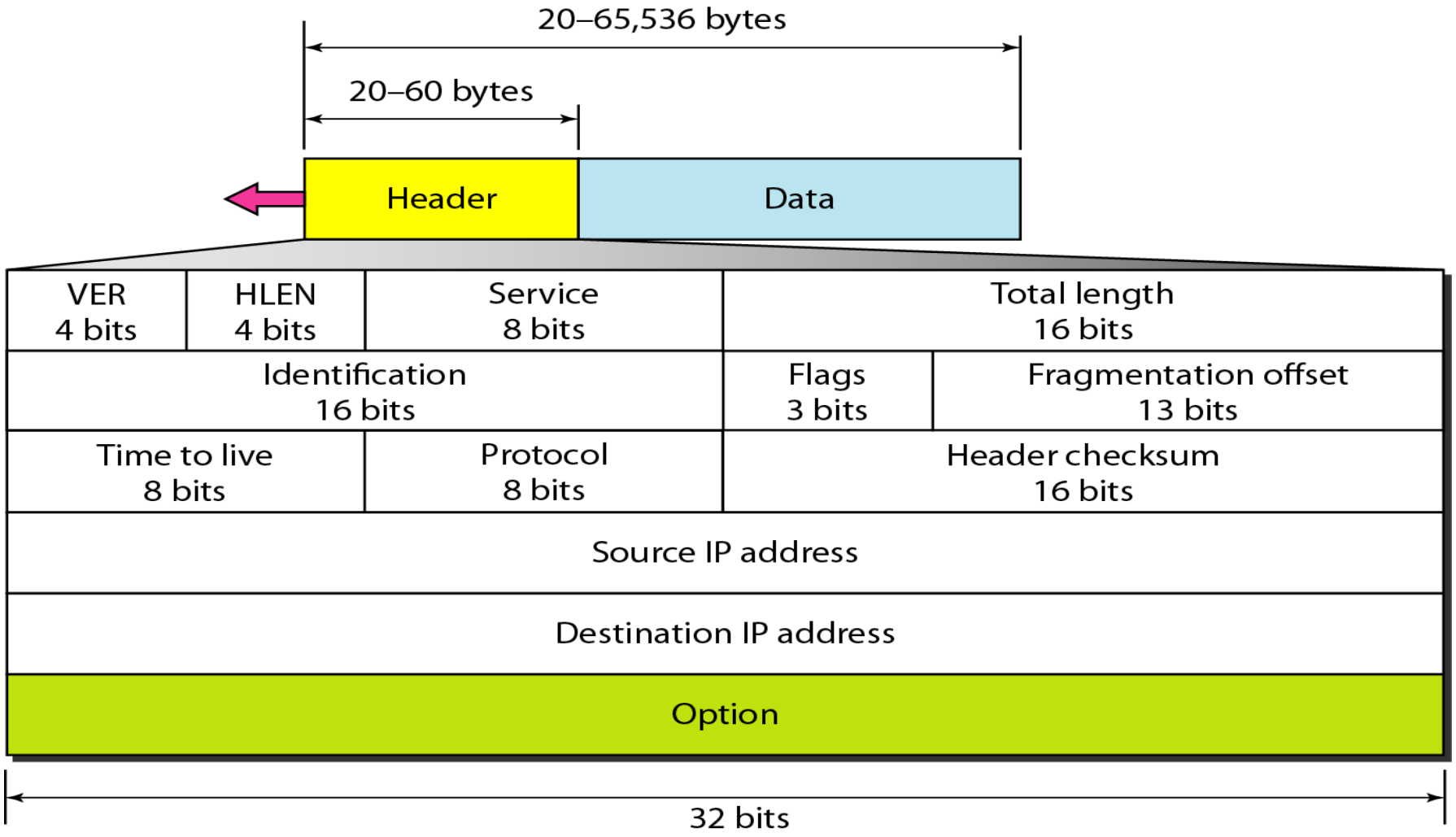
- ✓ Only the destination host reassembles the fragments since each travels independently.
- ✓ Fragments may take different routes but must all reach the destination for successful reassembly.
- ✓ Header Modifications: When fragmented, three fields in the header must change: **flags, fragmentation offset**, and **total length**.
- ✓ **Checksum value must always be recalculated**.

Fragmentation

Fields Related to Fragmentation and Reassembly of an IPv4 datagram are:

- ✓ Identification:
- ✓ Flags:
- ✓ Fragment offset:

IPv4 Datagram Format



Fragmentation

- Identification:
  - **16-Bit Identification Field:** Uniquely identifies a datagram from the source host.
  - **Combination for Uniqueness:** The **identification field** and the **source IPv4 address** **together must uniquely define** each datagram.
  - **To guarantee uniqueness use Counter:**
    - A counter is used to label datagrams, initialized to a positive number.
    - When sending a datagram, the **current counter value is copied to the identification field** and then **incremented by 1**.
    - **As long** as the **counter remains in memory**, each datagram's **identification is unique**.

Fragmentation

- Identification:
  - **Fragmentation Process:**
    - When a datagram is fragmented, the identification value is copied to all fragments.
    - **All fragments share the same identification number** as the original datagram.
  - **Reassembly Aid:**
    - **The identification number assists the destination host in reassembling** the datagram, ensuring fragments with the same identification value are combined into one datagram.

Fragmentation

Flags is a 3-Bit Field:

- **First Bit:** Reserved for future use.
- **Second Bit:** "Do Not Fragment" (DF):
  - ✓ Value 1: Datagram **must not be fragmented**. If fragmentation is needed, the datagram is discarded, and an ICMP error is sent to the source.
  - ✓ Value 0: Datagram can be fragmented if necessary.
- **Third Bit:** "**More Fragments**" (MF):
  - ✓ Value 1: Indicates more fragments will follow.
  - ✓ Value 0: Indicates this is the last or only fragment of the datagram.

Figure 20.10 ***Flags used in fragmentation***

Fragmentation

Fragmentation Offset (13 bits):

- ✓ Indicates the position of a fragment within the original datagram.
- ✓ **Measured in 8-Byte Units**: Offset value shows data position relative to the whole datagram, divided by 8.

Example:

- ✓ Original datagram size: 4000 bytes (numbered 0 to 3999).
- ✓ First Fragment: Carries bytes **0** to 1399, offset = 0/8 = 0.
- ✓ Second Fragment: Carries bytes **1400** to 2799, offset = 1400/8 = 175.
- ✓ Third Fragment: Carries bytes **2800** to 3999, offset = 2800/8 = 350.
- ✓ Offset Limitation: Offset field (13 bits) supports values up to 8191.

- ✓ Identification Field: Same across all fragments.
- ✓ Flags Field: "More Fragment" bit is set for all fragments except the last one.

Figure 20.11 *Fragmentation example*

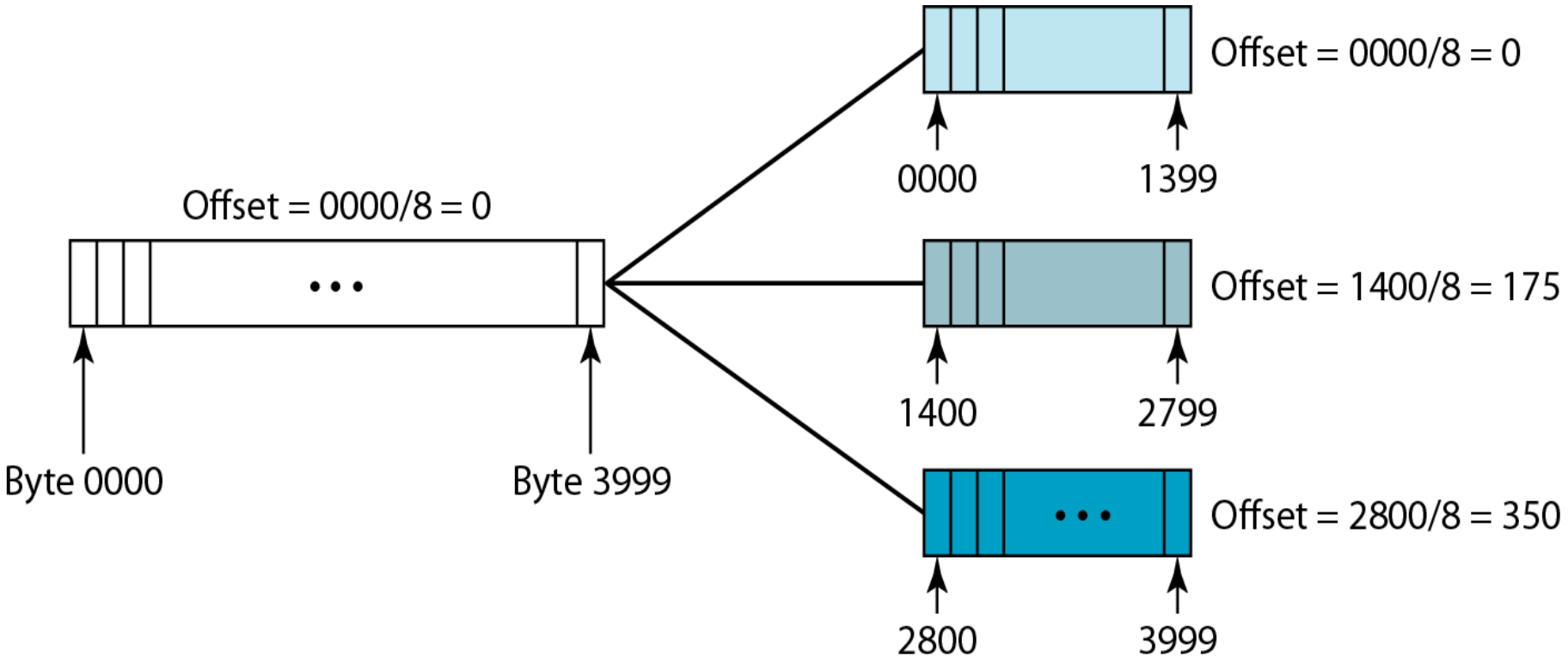
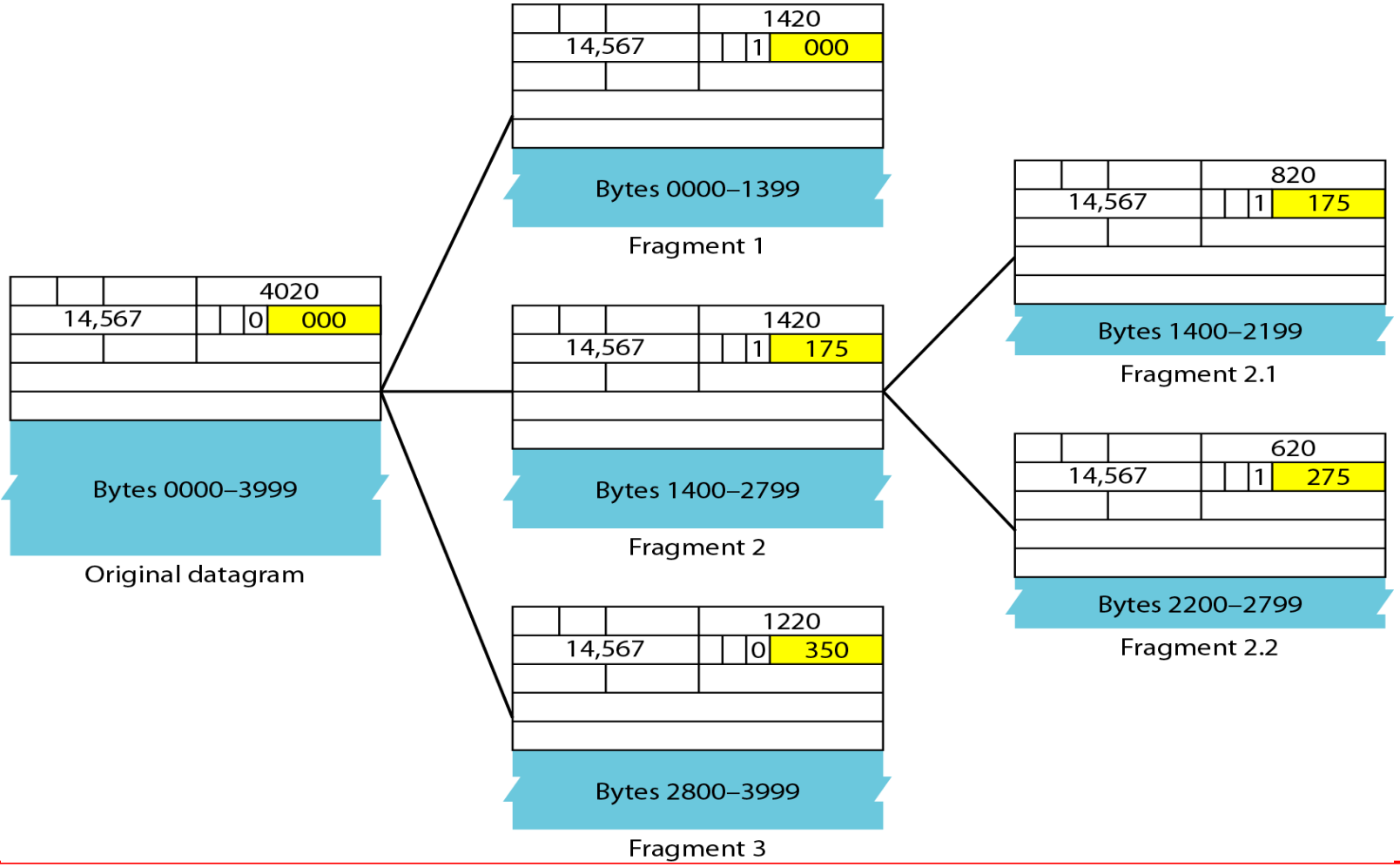


Figure 20.12 *Detailed fragmentation example*



Notice the identification field is the same in all fragments. Notice the flags with the ***more*** bit set for all fragments except the last. Also, the value of the offset field for each fragment is shown.

Example 20.5

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

Solution

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

Example 20.6

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

Solution

If the M bit is 1, it means that there is at least one more fragment.

This fragment can be the first one or a middle one, but not the last one.

We don't know if it is the first one or a middle one; we need more information (the value of the fragmentation offset).

Example 20.7

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

Solution

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

Example 20.8

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

Solution

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

Example 20.9

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

Solution

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

Example 20.10

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

Figure 20.13 *Example of checksum calculation in IPv4*

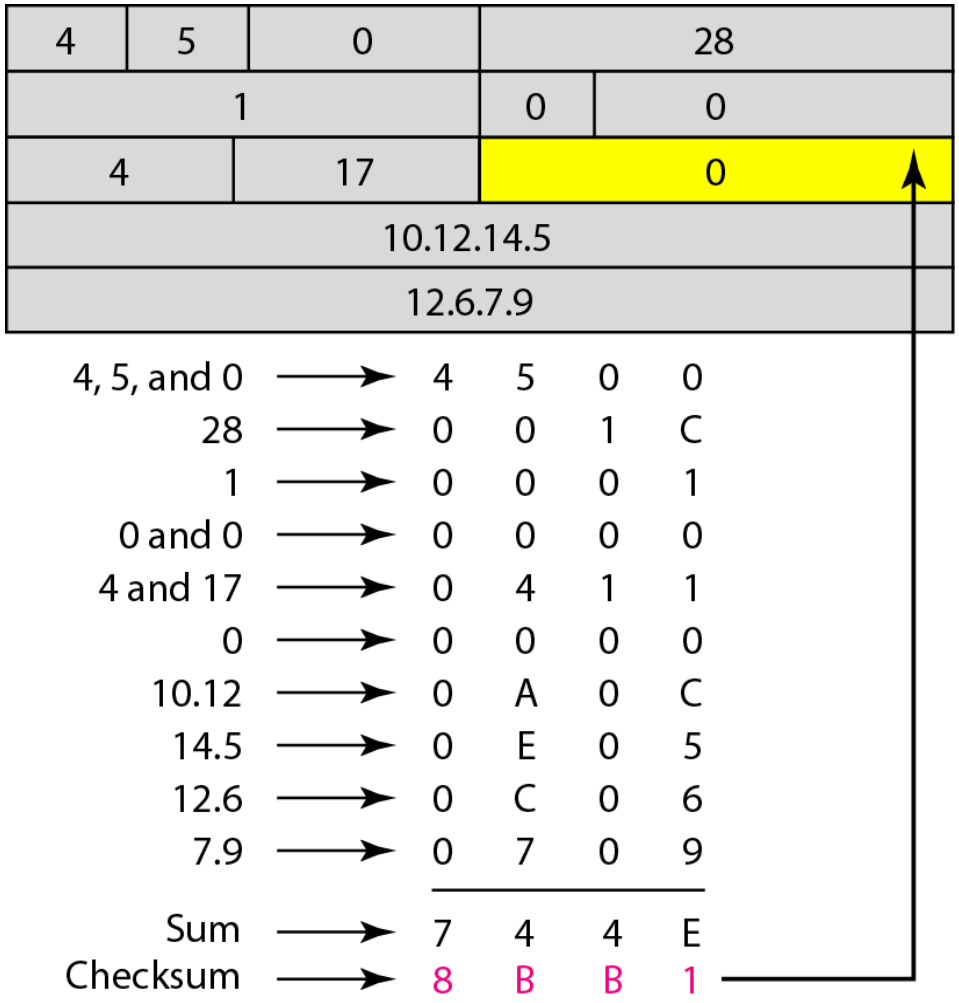
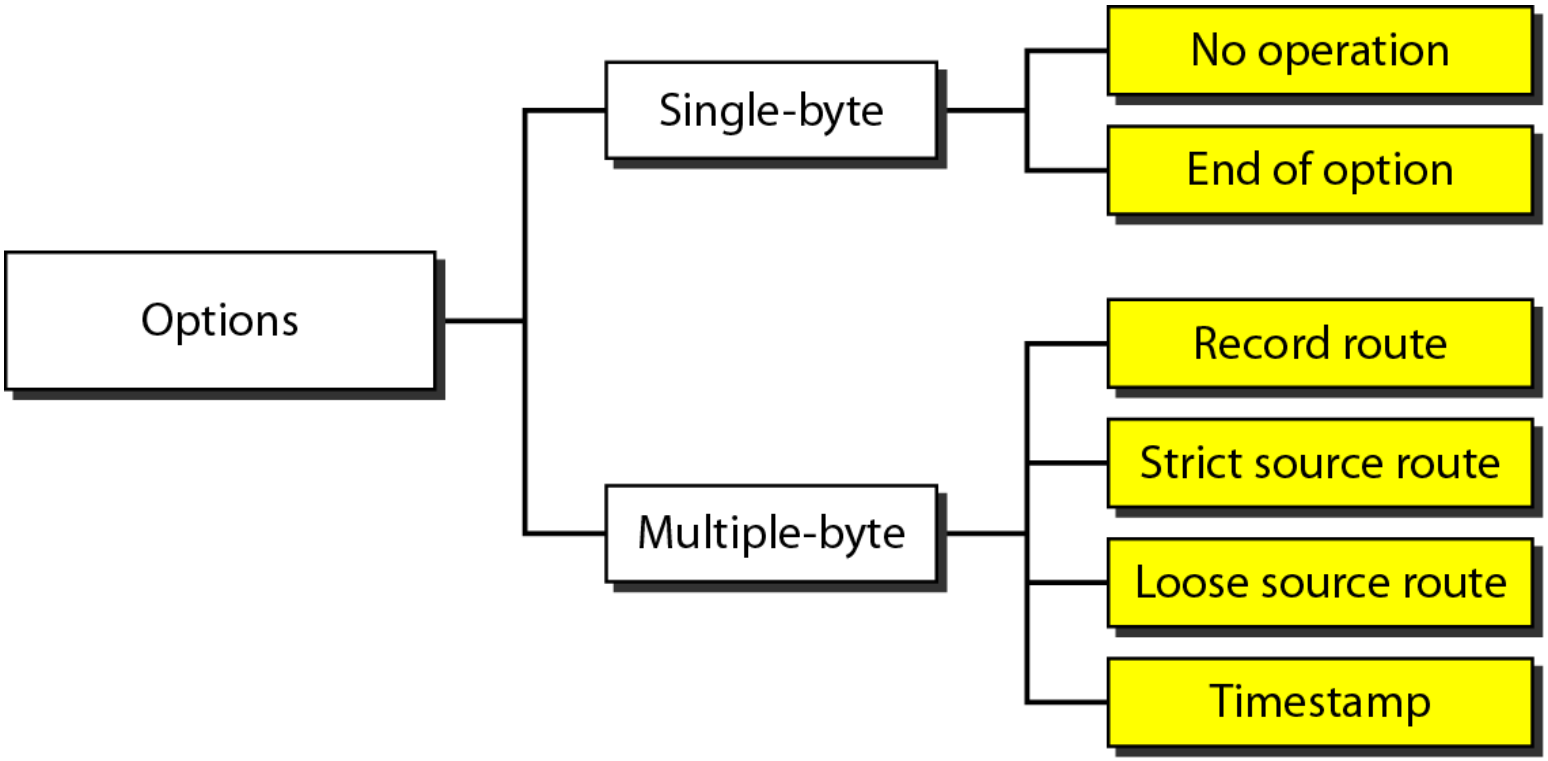


Figure 20.14 *Taxonomy of options in IPv4*



20-3 IPv6

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

Topics discussed in this section:

Advantages
Packet Format
Extension Headers

Figure 20.15 *IPv6 datagram header and payload*

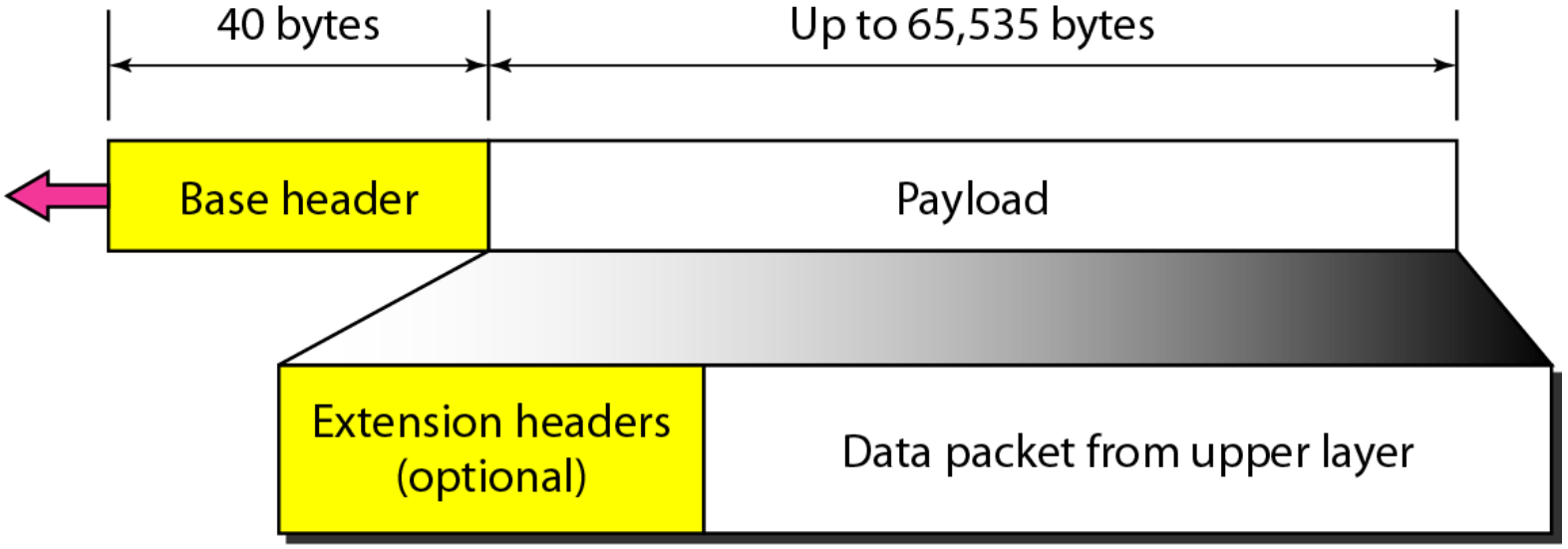


Figure 20.16 *Format of an IPv6 datagram*

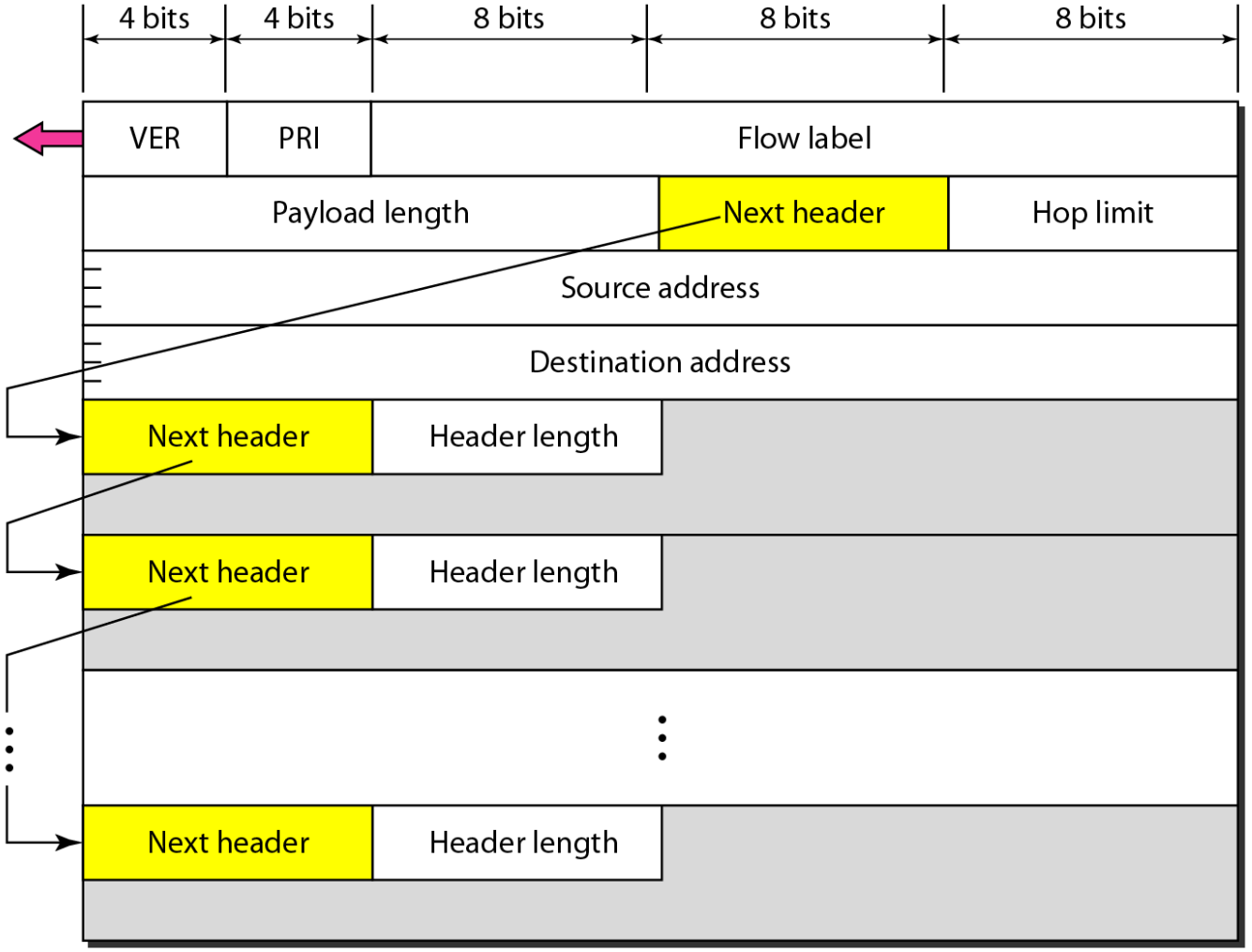


Table 20.6 ***Next header codes for IPv6***

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

Table 20.7 ***Priorities for congestion-controlled traffic***

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

Table 20.8 ***Priorities for noncongestion-controlled traffic***

Priority	*Meaning*
8	Data with greatest redundancy
. . .	. . .
15	Data with least redundancy

Table 20.9 *Comparison between IPv4 and IPv6 packet headers*

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*

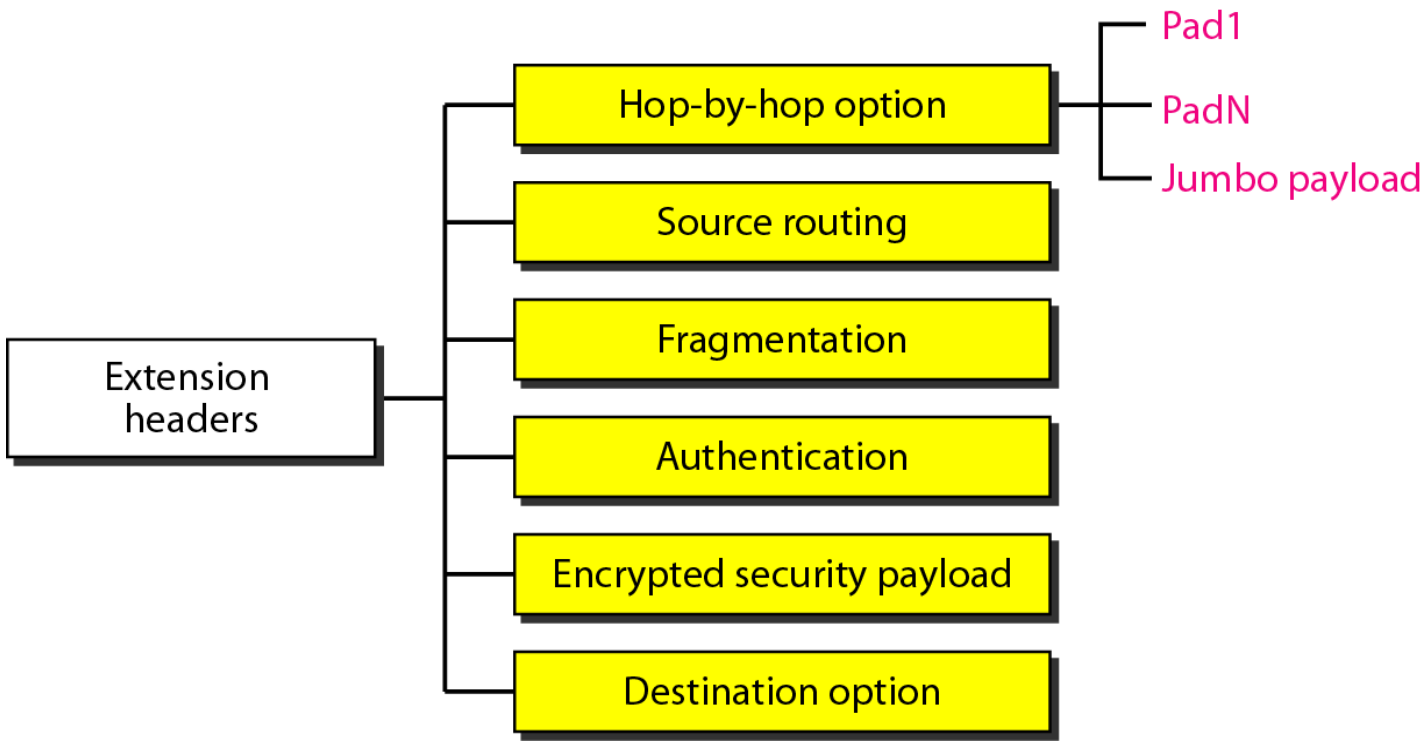


Table 20.10 *Comparison between IPv4 options and IPv6 extension headers*

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.

20-4 TRANSITION FROM IPv4 TO IPv6

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

Topics discussed in this section:

Dual Stack
Tunneling
Header Translation

Figure 20.18 *Three transition strategies*

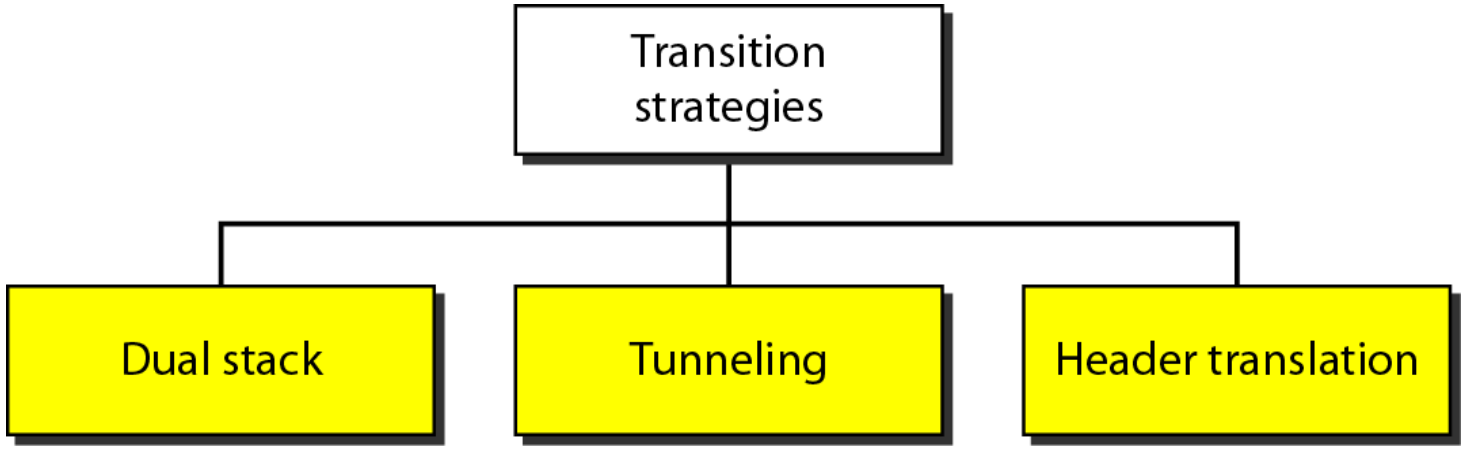


Figure 20.19 *Dual stack*

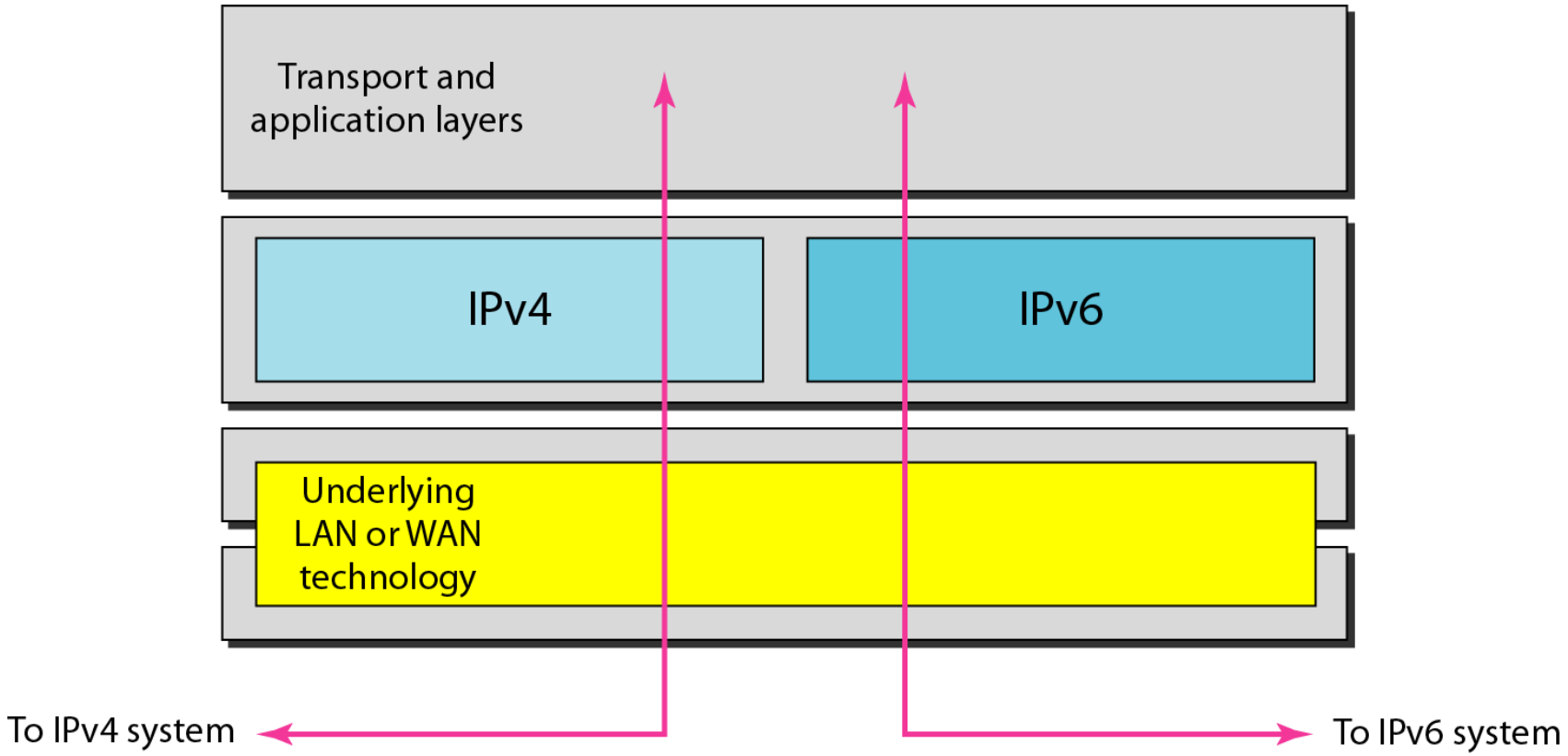


Figure 20.20 *Tunneling strategy*

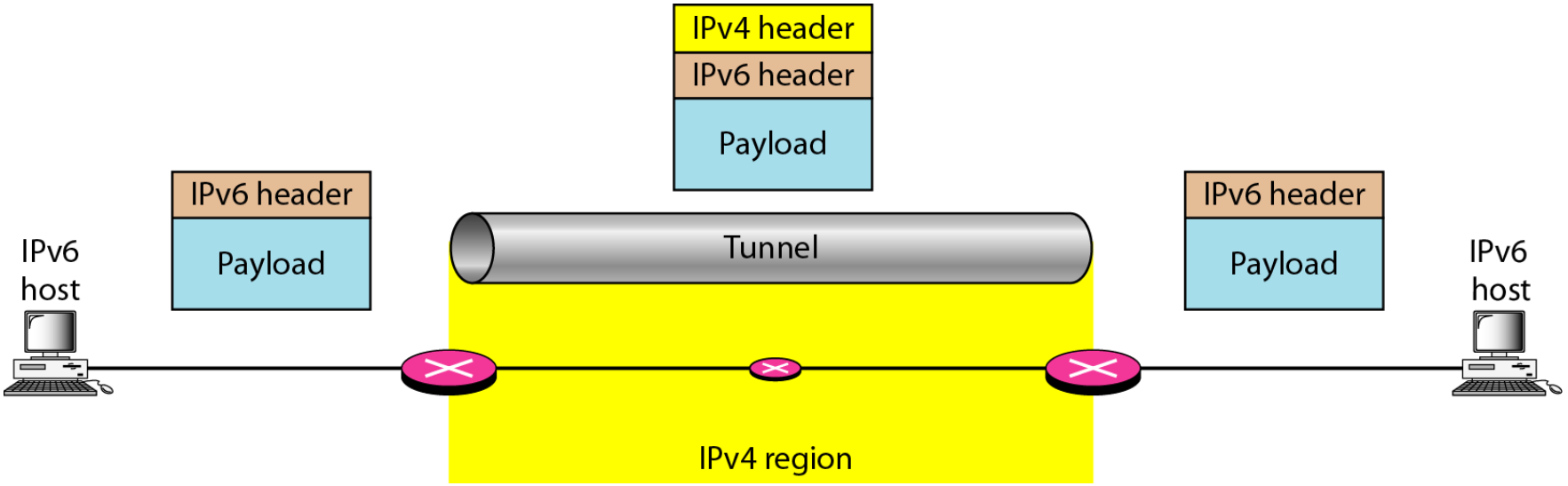


Figure 20.21 *Header translation strategy*

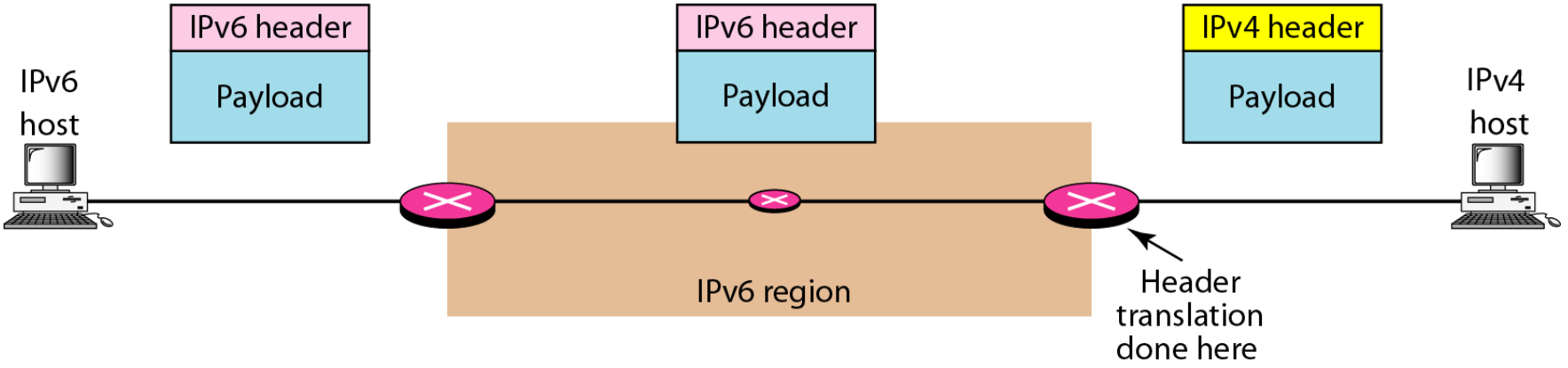


Table 20.11 *Header translation*

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