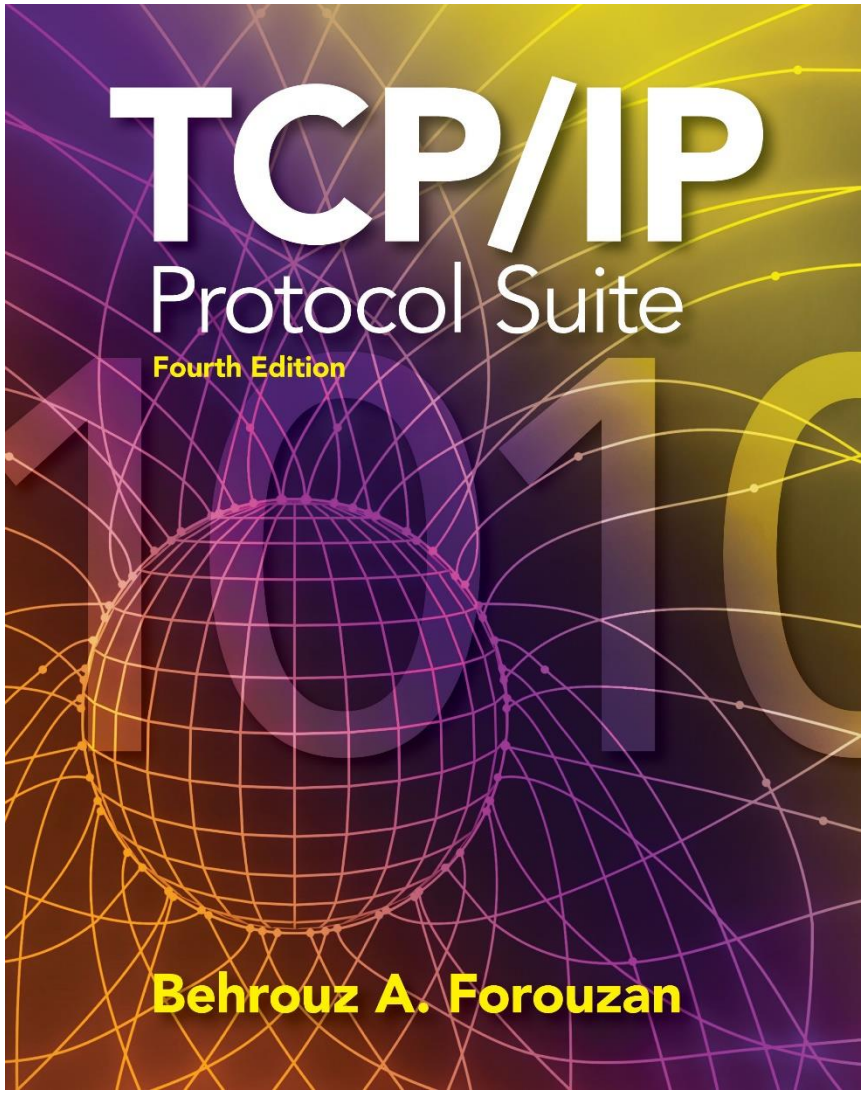
Chapter 9

Internet Control Message Protocol Version 4 (ICMPv4)



OBJECTIVES:

- To discuss the rationale for the existence of ICMP.
- To show how ICMP messages are divided into two categories: error reporting and query messages.
- To discuss the purpose and format of error-reporting messages.
- To discuss the purpose and format of query messages.
- To show how the checksum is calculated for an ICMP message.
- To show how debugging tools using the ICMP protocol.
- To show how a simple software package that implements ICMP is organized.

Chapter Outline

9-1 INTRODUCTION

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? These are examples of situations where an error has occurred and the IP protocol has no built-in mechanism to notify the original host.

Topics Discussed in the Section

- ✓ **The position of ICMP in the TCP/IP suite**
- ✓ **Encapsulation of ICMP Packets**

Figure 9.1 *Position of ICMP in the network layer*

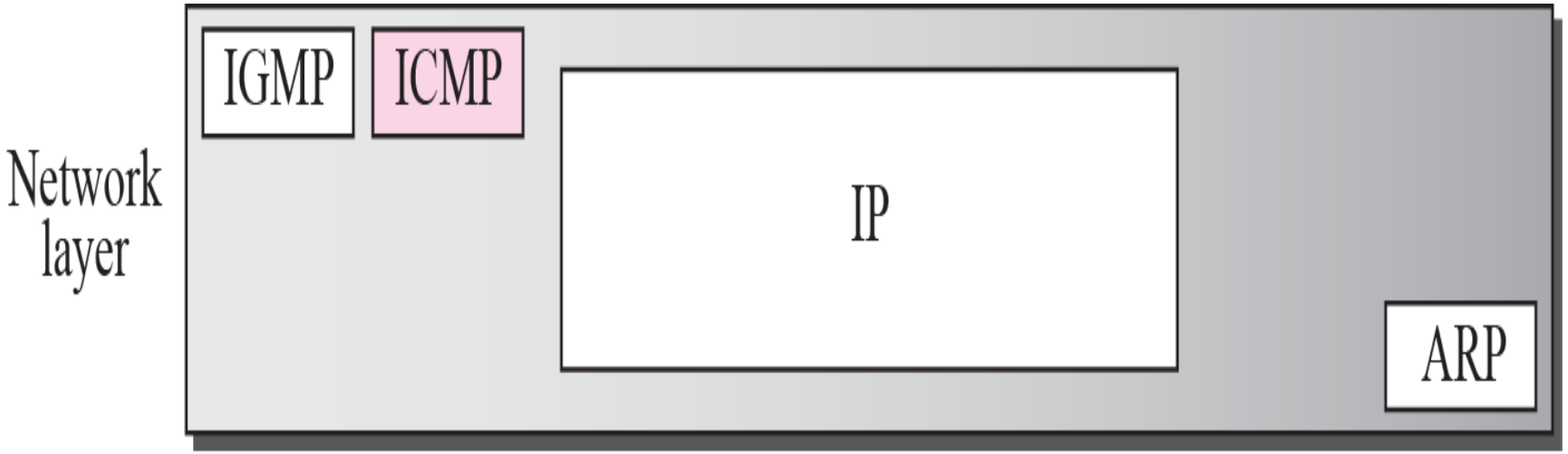
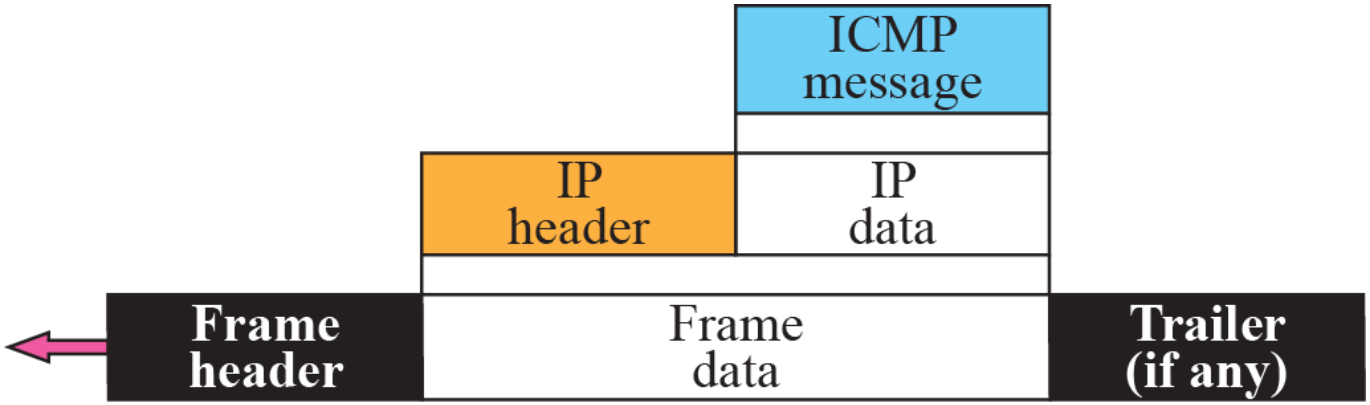


Figure 9.2 *ICMP encapsulation*



9-2 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. Also, hosts can discover and learn about routers on their network and routers can help a node redirect its messages.

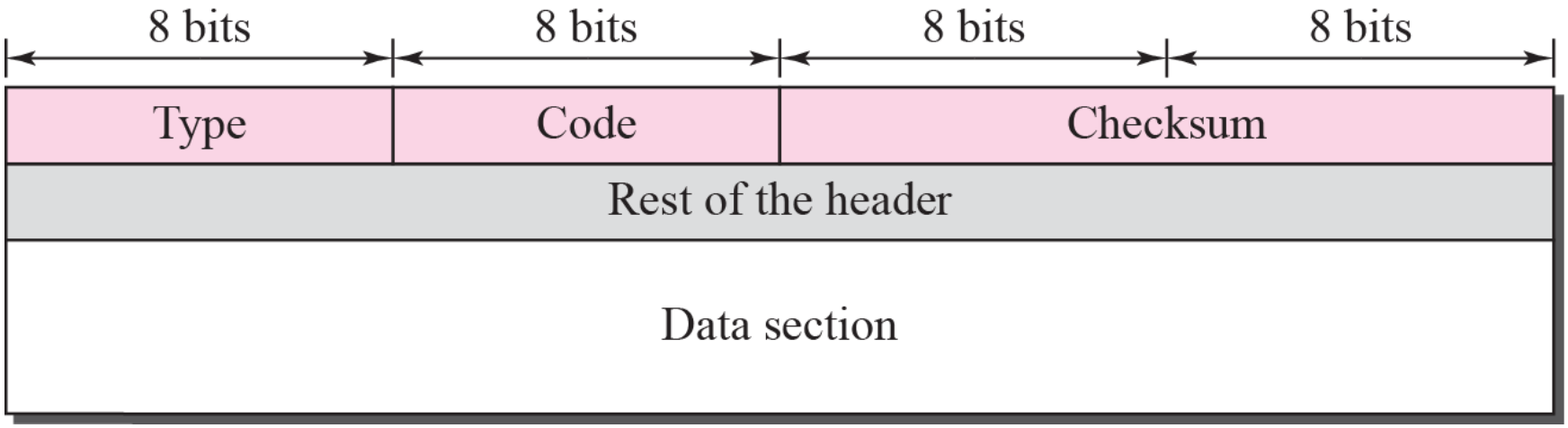
Topics Discussed in the Section

- ✓ **Message Format**
- ✓ **Error Reporting Messages**
- ✓ **Query Messages**
- ✓ **Checksum**

Table 9.1 *ICMP messages*

Category	*Type*	*Message*
Error-reporting messages	3	Destination unreachable
	4	Source quench
	11	Time exceeded
	12	Parameter problem
	5	Redirection
Query messages	8 or 0	Echo request or reply
	13 or 14	Timestamp request or reply

Figure 9.3 *General format of ICMP messages*

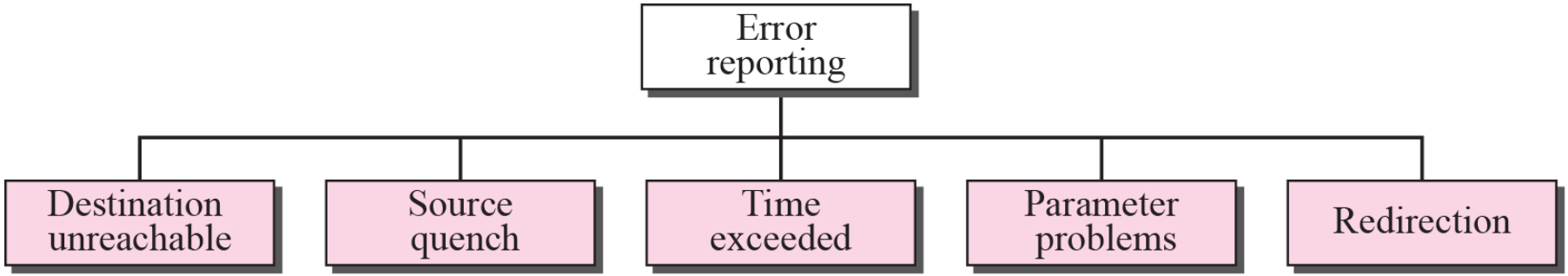


The first field, ICMP type, defines the type of the message. The code field specifies the reason for the particular message type. The last common field is the checksum field. The rest of the header is specific for each message type. The data section in error messages carries information for finding the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

Note

ICMP always reports error messages to the original source.

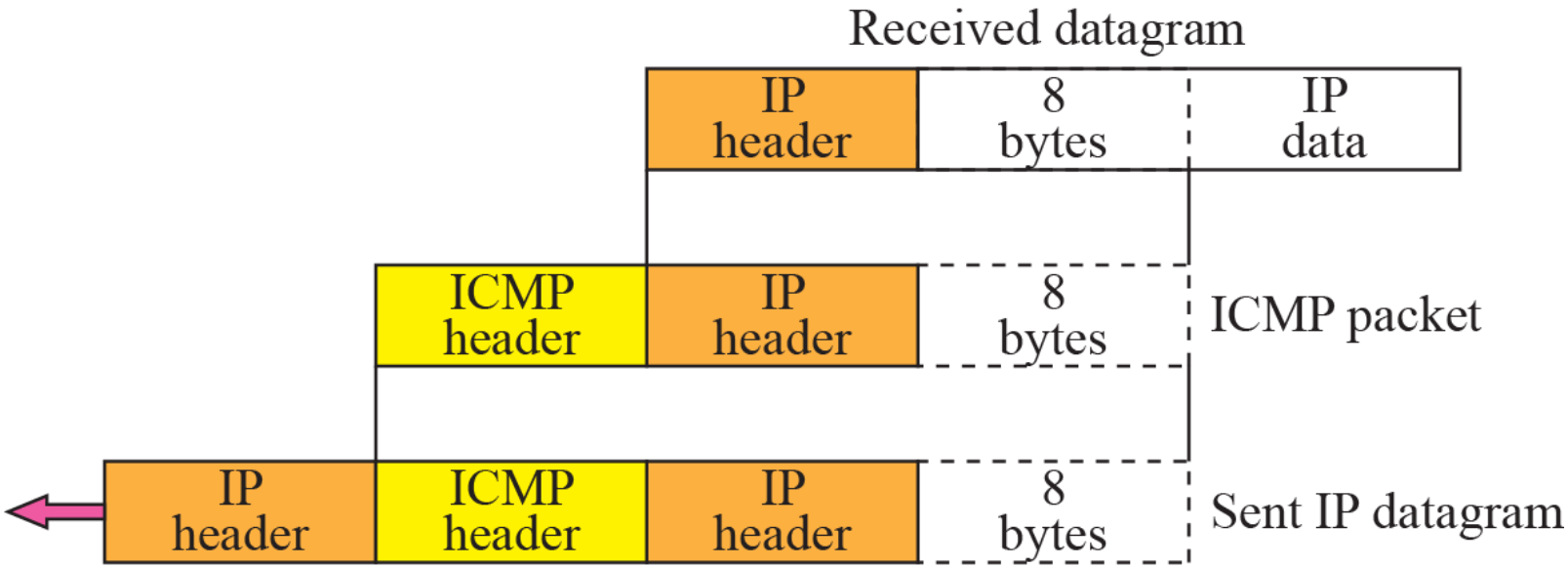
✓Figure 9.4 Error Reporting Messages



The following are important points about ICMP error messages:

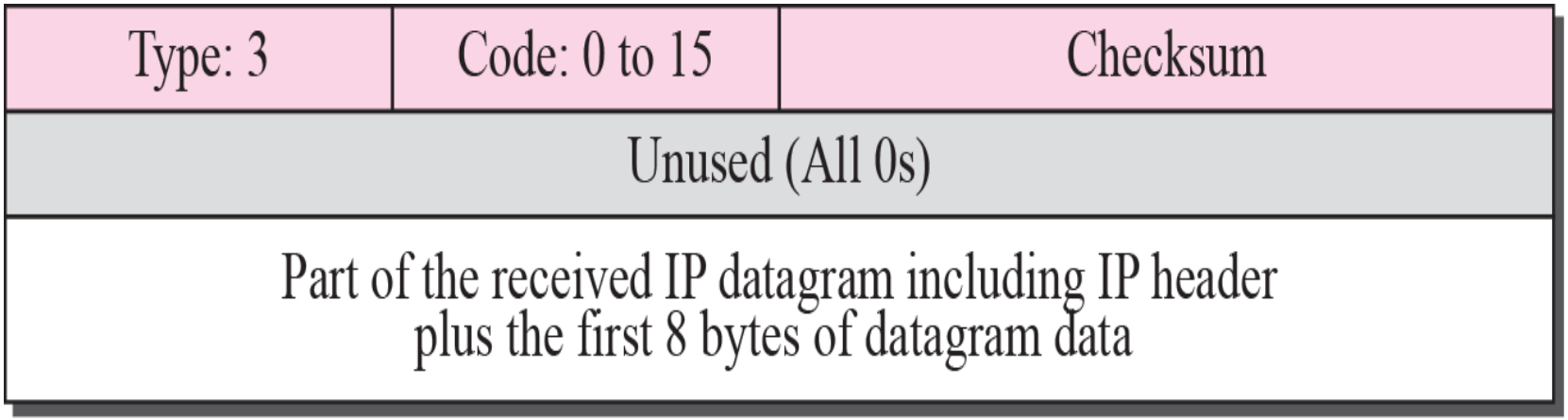
- No ICMP error message will be generated in response to a datagram carrying an ICMP error message.
- No ICMP error message will be generated for a fragmented datagram that is not the first fragment.
- No ICMP error message will be generated for a datagram having a multicast address.
- No ICMP error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

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



Error Reporting Messages

Figure 9.6 ***Destination-unreachable format***



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.

Error Reporting Messages

Destination-unreachable format:

The code field for this type specifies the reason for discarding the datagram:

- ❑ Code 0. The network is unreachable, possibly due to hardware failure.
- ❑ Code 1. The host is unreachable. This can also be due to hardware failure.
- ❑ Code 2. The protocol is unreachable. An IP datagram can carry data belonging to higher-level protocols such as UDP, TCP, and OSPF. If the destination host receives a datagram that must be delivered, for example, to the TCP protocol, but the TCP protocol is not running at the moment, a code 2 message is sent.
- ❑ Code 3. The port is unreachable. The application program (process) that the datagram is destined for is not running at the moment.

.
.
.
.
.
.
.
.
.
.

Code 15: ……

Error Reporting Messages

Destination-unreachable format

Note

Destination-unreachable messages with codes 2 or 3 can be created only by the destination host.

Other destination-unreachable messages can be created only by routers.

Source-quench format

Note

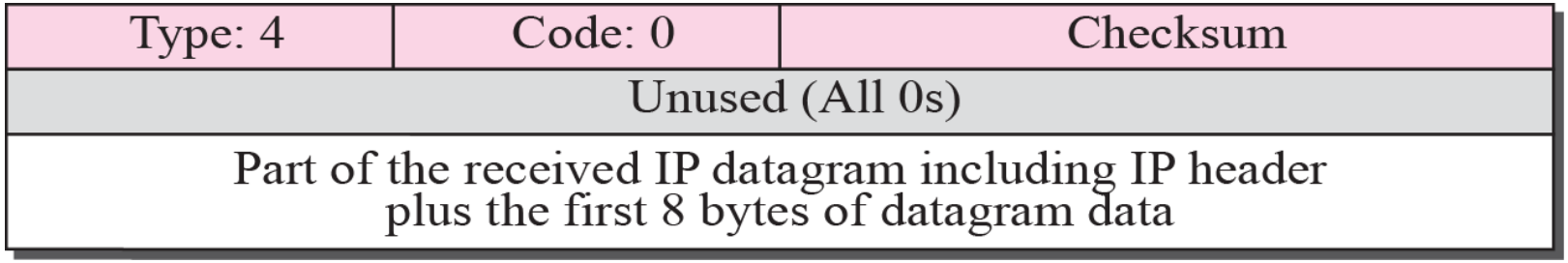
A router cannot detect all problems that prevent the delivery of a packet.

Source-quench format

Note

There is no flow-control or congestion-control mechanism in the IP protocol.

Figure 9.7 *Source-quench format*



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

Source-quench format

Note

A source-quench message informs the source that a datagram has been discarded due to congestion in a router or the destination host.

The source must slow down the sending of datagrams until the congestion is relieved.

Source-quench format

Note

One source-quench message is sent for each datagram that is discarded due to congestion.

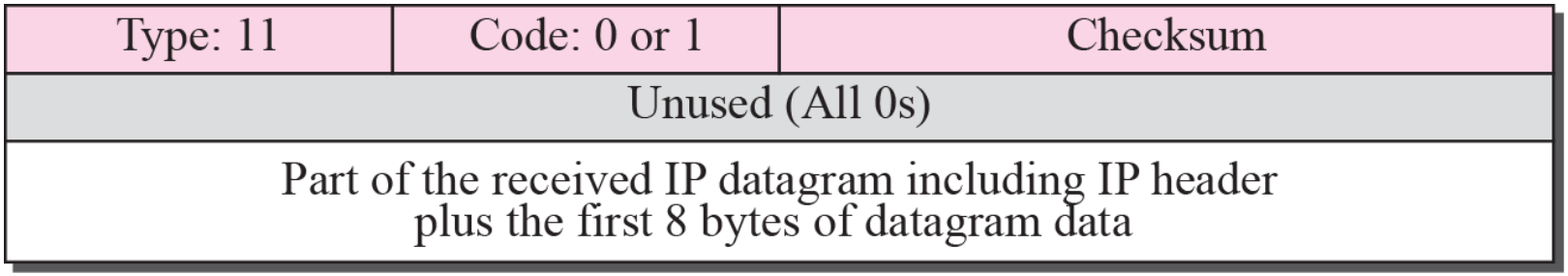
Note

Time-exceeded message format

Whenever a router decrements a datagram with a time-to-live value to zero, it discards the datagram and sends a time-exceeded message to the original source.

When the final destination does not receive all of the fragments in a set time, it discards the received fragments and sends a time-exceeded message to the original source.

Figure 9.8 *Time-exceeded message format*



In a time-exceeded message,

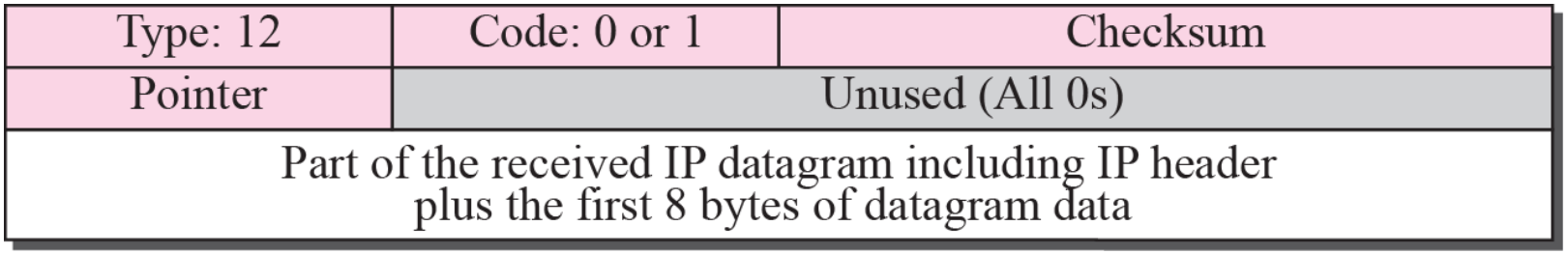
- ***Code 0 is used only by routers to show that the value of the time-to-live field is zero.***
- ***Code 1 is used only by the destination host to show that not all of the fragments have arrived within a set time.***

Parameter-problem message format

Note

A parameter-problem message can be created by a router or the destination host.

Figure 9.9 *Parameter-problem message format*



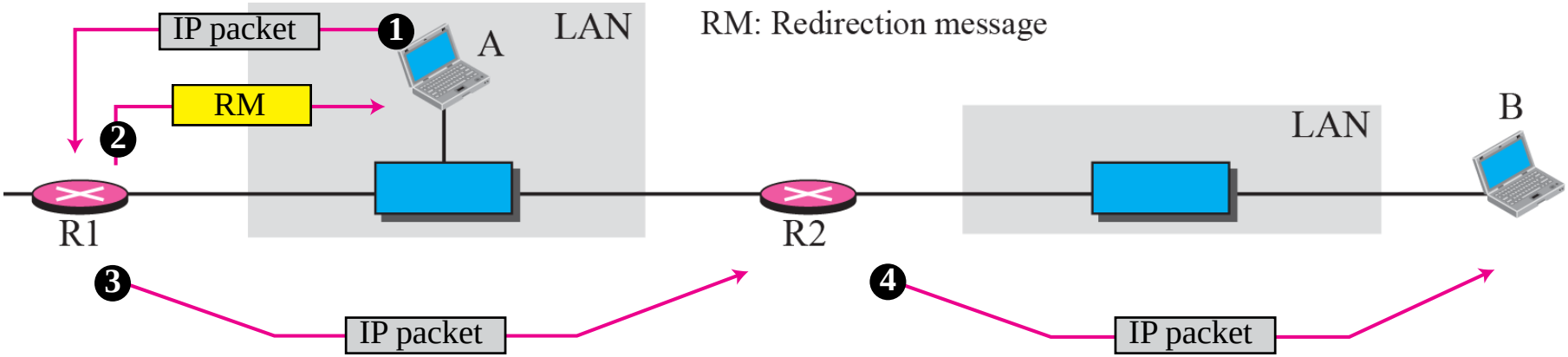
Any ambiguity in the header part of a datagram can create serious problems as the datagram travels through the Internet. If a router or the destination host discovers an ambiguous or missing value in any field of the datagram, it discards the datagram and sends a parameter-problem message back to the source.

The code field in this case specifies the reason for discarding the datagram:

❑ Code 0. There is an error or ambiguity in one of the header fields. In this case, the value in the pointer field points to the byte with the problem. For example, if the value is zero, then the first byte is not a valid field.

❑ Code 1. The required part of an option is missing. In this case, the pointer is not used.

Figure 9.10 ***Redirection concept***



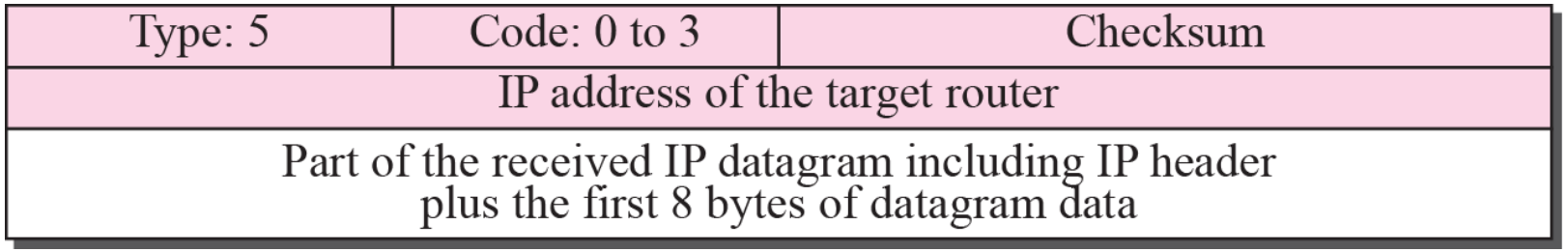
When a router needs to send a packet destined for another network, it must know the IP address of the next appropriate router. The same is true if the sender is a host. Both routers and hosts then must have a routing table to find the address of the router or the next router. Routers take part in the routing update process as we will see in Chapter 11 and are supposed to be updated constantly. Routing is dynamic.

Note

A host usually starts with a small routing table that is gradually augmented and updated.

One of the tools to accomplish this is the redirection message.

Figure 9.11 *Redirection message format*



Note

A redirection message is sent from a router to a host on the same local network.

✓ Query Messages

In addition to error reporting, ICMP can also diagnose some network problems. This is accomplished through the query messages. A group of five different pairs of messages

Echo Request and Reply

The echo-request and echo-reply messages can be used to determine if there is communication at the IP level.

The echo-request and echo-reply messages can also be used by a host to see if another host is reachable. At the user level, this is done by invoking the packet Internet groper (ping) command. Today, most systems provide a version of the ping command

Note

An echo-request message can be sent by a host or router.

An echo-reply message is sent by the host or router that receives an echo-request message.

Note

Echo-request and echo-reply messages can be used by network managers to check the operation of the IP protocol.

Note

Echo-request and echo-reply messages can test the reachability of a host.

This is usually done by invoking the ping command.

Figure 9.12 *Echo-request and echo-reply message*

Type 8: Echo request
Type 0: Echo reply

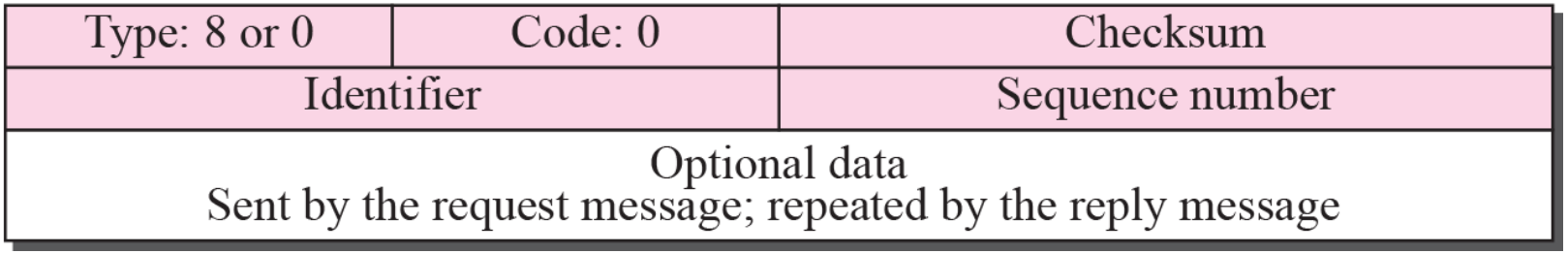


Figure 9.13 *Timestamp-request and timestamp-reply message format*

Type 13: request
Type 14: reply

Type: 13 or 14	Code: 0	Checksum
Identifier		Sequence number
Original timestamp		
Receive timestamp		
Transmit timestamp		

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

Note

Timestamp-request and timestamp-reply messages can be used to calculate the round-trip time between a source and a destination machine even if their clocks are not synchronized.

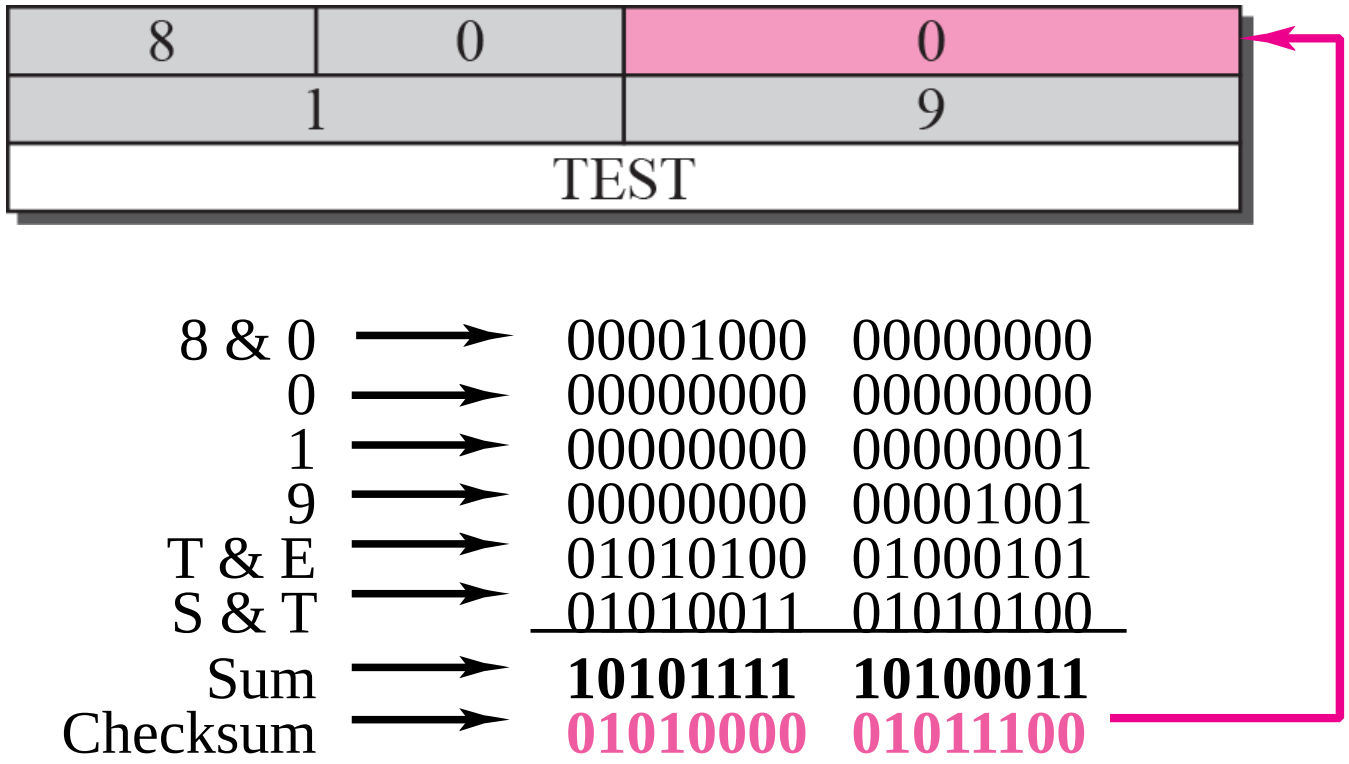
Note

The timestamp-request and timestamp-reply messages can be used to synchronize two clocks in two machines if the exact one-way time duration is known.

Example 9.1

Figure 9.14 shows an example of checksum calculation for a simple echo-request message (see Figure 9.12). 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*



9-3 DEBUGGING TOOLS

There are several tools that can be used in the Internet for debugging. We can find if a host or router is alive and running. We can trace the route of a packet. We introduce two tools that use ICMP for debugging: ping and traceroute. We will introduce more tools in future chapters after we have discussed the corresponding protocols.

Topics Discussed in the Section

- ✓ **Ping**
- ✓ **Traceroute**

Example 9.2

We can use the ping program to find if a host is alive and responding.

Using ping program to test the server fhda.edu. The result is shown below:

```
$ ping fhda.edu
PING fhda.edu (153.18.8.1)   56 (84)  bytes of data.
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=0    ttl=62    time=1.91 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=1    ttl=62    time=2.04 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=2    ttl=62    time=1.90 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=3    ttl=62    time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=4    ttl=62    time=1.93 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=5    ttl=62    time=2.00 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=6    ttl=62    time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=7    ttl=62    time=1.94 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=8    ttl=62    time=1.97 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=9    ttl=62    time=1.89 ms
64 bytes from tiptoe.fhda.edu (153.18.8.1): icmp_seq=10   ttl=62    time=1.98 ms

--- fhda.edu ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10103 ms
rtt min/avg/max = 1.899/1.955/2.041 ms
```

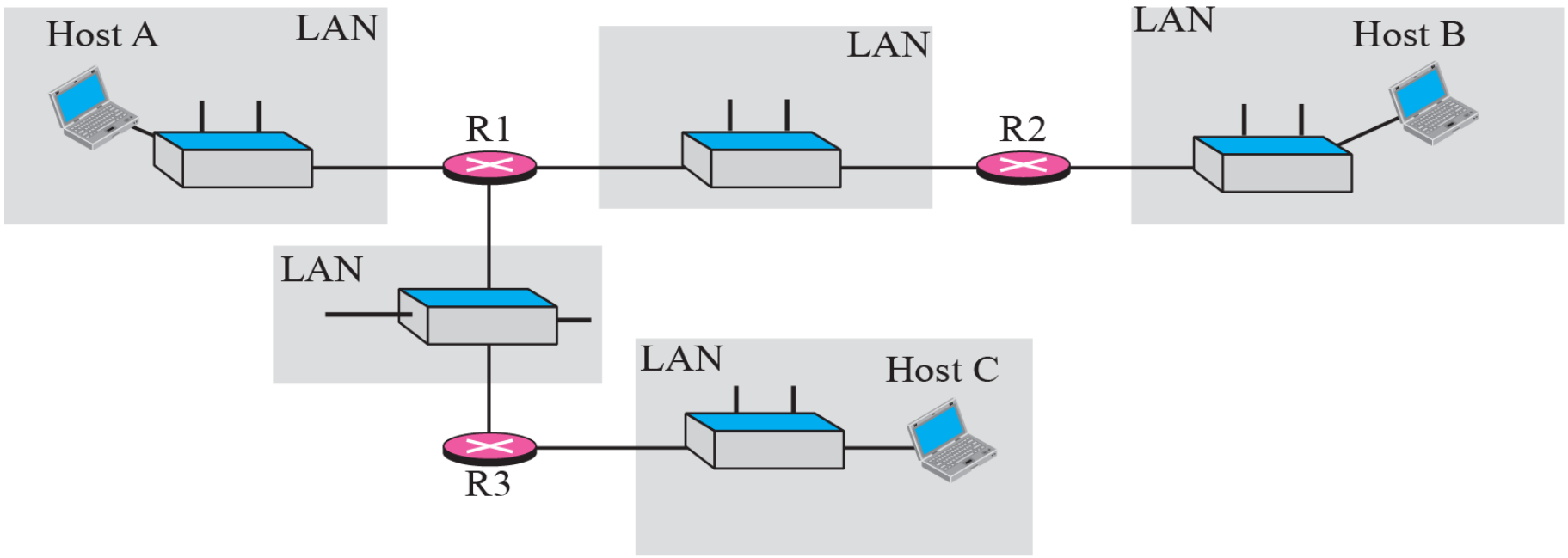
Example 9.3

For the second example, we want to know if the adelphia.net mail server is alive and running. The result is shown below: Note that in this case, we sent 14 packets, but only 13 have been returned. We may have interrupted the program before the last packet, with sequence number 13, was returned.

```
$ ping mail.adelphia.net
PING mail.adelphia.net (68.168.78.100) 56(84) bytes of data.
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=0    ttl=48    time=85.4 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=1    ttl=48    time=84.6 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=2    ttl=48    time=84.9 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=3    ttl=48    time=84.3 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=4    ttl=48    time=84.5 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=5    ttl=48    time=84.7 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=6    ttl=48    time=84.6 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=7    ttl=48    time=84.7 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=8    ttl=48    time=84.4 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=9    ttl=48    time=84.2 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=10   ttl=48    time=84.9 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=11   ttl=48    time=84.6 ms
64 bytes from mail.adelphia.net (68.168.78.100): icmp_seq=12   ttl=48    time=84.5 ms

--- mail.adelphia.net ping statistics ---
14 packets transmitted, 13 received, 7% packet loss, time 13129 ms
rtt min/avg/max/mdev = 84.207/84.694/85.469
```

Figure 9.15 *The traceroute program operation*



Example 9.4

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

```
$ traceroute fhda.edu
traceroute to fhda.edu      (153.18.8.1), 30 hops max, 38 byte packets
1 Dcore.fhda.edu        (153.18.31.25)      0.995 ms      0.899 ms      0.878 ms
2 Dbackup.fhda.edu      (153.18.251.4)      1.039 ms      1.064 ms      1.083 ms
3  tiptoe.fhda.edu      (153.18.8.1)        1.797 ms      1.642 ms      1.757 ms
```

Example 9.5

In this example, we trace a longer route, the route to xerox.com. The following is a partial listing.

```
$ traceroute xerox.com
traceroute to xerox.com (13.1.64.93), 30 hops max, 38 byte packets
1  Dcore.fhda.edu      (153.18.31.254)     0.622 ms    0.891 ms    0.875 ms
2  Ddmz.fhda.edu       (153.18.251.40)     2.132 ms    2.266 ms    2.094 ms
3  Cinic.fhda.edu      (153.18.253.126)    2.110 ms    2.145 ms    1.763 ms
4  cenic.net           (137.164.32.140)    3.069 ms    2.875 ms    2.930 ms
5  cenic.net           (137.164.22.31)     4.205 ms    4.870 ms    4.197 ms
6  cenic.net           (137.164.22.167)    4.250 ms    4.159 ms    4.078 ms
7  cogentco.com        (38.112.6.225)      5.062 ms    4.825 ms    5.020 ms
8  cogentco.com        (66.28.4.69)        6.070 ms    6.207 ms    5.653 ms
9  cogentco.com        (66.28.4.94)        6.070 ms    5.928 ms    5.499 ms
```

Example 9.6

An interesting point is that a host can send a traceroute packet to itself. This can be done by specifying the host as the destination. The packet goes to the loopback address as we expect.

```
$ traceroute voyager.deanza.edu
traceroute to voyager.deanza.edu   (127.0.0.1), 30 hops max, 38 byte packets
1  voyager          (127.0.0.1)          0.178 ms          0.086 ms          0.055 ms
```

Example 9.7

Finally, we use the traceroute program to find the route between fhda.edu and mhhe.com (McGraw-Hill server). We notice that we cannot find the whole route. When traceroute does not receive a response within 5 seconds, it prints an asterisk to signify a problem (not the case in this example), and then tries the next hop.

```
$ traceroute mhhe.com
traceroute to mhhe.com (198.45.24.104), 30 hops max, 38 byte packets
```

1	Dcore.fhda.edu	(153.18.31.254)	1.025 ms	0.892 ms	0.880 ms
2	Ddmz.fhda.edu	(153.18.251.40)	2.141 ms	2.159 ms	2.103 ms
3	Cinic.fhda.edu	(153.18.253.126)	2.159 ms	2.050 ms	1.992 ms
4	cenic.net	(137.164.32.140)	3.220 ms	2.929 ms	2.943 ms
5	cenic.net	(137.164.22.59)	3.217 ms	2.998 ms	2.755 ms
6	SanJose1.net	(209.247.159.109)	10.653 ms	10.639 ms	10.618 ms
7	SanJose2.net	(64.159.2.1)	10.804 ms	10.798 ms	10.634 ms
8	Denver1.Level3.net	(64.159.1.114)	43.404 ms	43.367 ms	43.414 ms
9	Denver2.Level3.net	(4.68.112.162)	43.533 ms	43.290 ms	43.347 ms
10	unknown	(64.156.40.134)	55.509 ms	55.462 ms	55.647 ms
11	mcleodusa1.net	(64.198.100.2)	60.961 ms	55.681 ms	55.461 ms
12	mcleodusa2.net	(64.198.101.202)	55.692 ms	55.617 ms	55.505 ms
13	mcleodusa3.net	(64.198.101.142)	56.059 ms	55.623 ms	56.333 ms
14	mcleodusa4.net	(209.253.101.178)	297.199 ms	192.790 ms	250.594 ms
15	eppg.com	(198.45.24.246)	71.213 ms	70.536 ms	70.663 ms
16	…	…	…	…	…

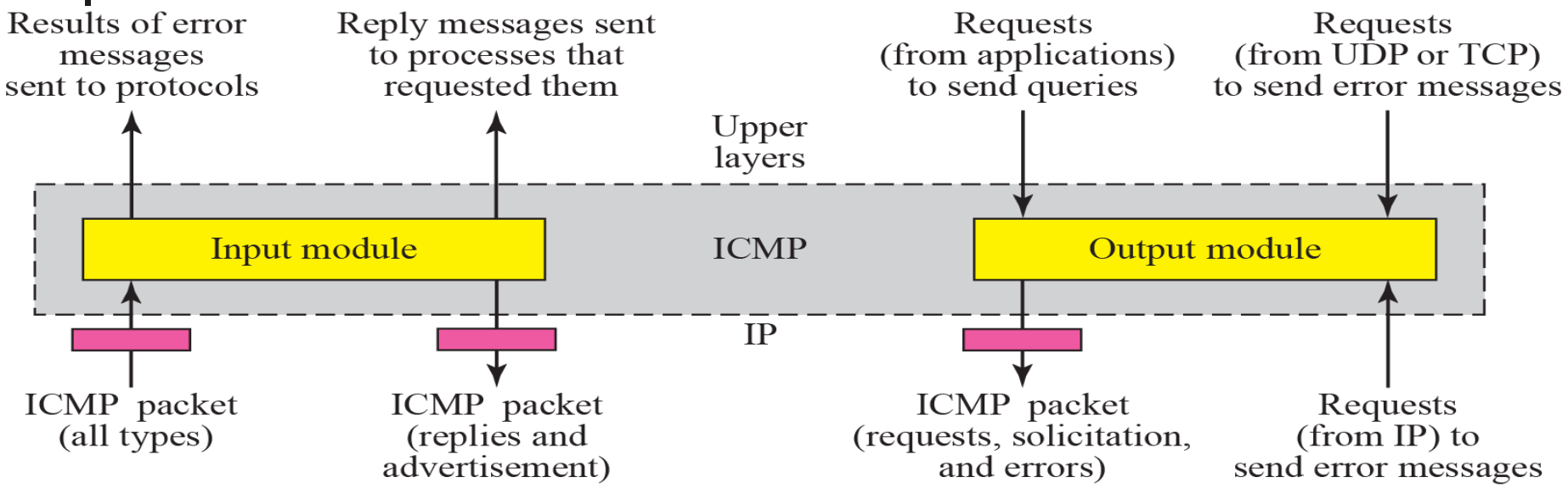
9-4 ICMP PACKAGE

To give an idea of how ICMP can handle the sending and receiving of ICMP messages, we present our version of an ICMP package made of two modules: an input module and an output module.

Topics Discussed in the Section

- ✓ **Input Module**
- ✓ **Output Module**

Figure 9.16 *ICMP package*



The input module handles all received ICMP messages. It is invoked when an ICMP packet is delivered to it from the IP layer. If the received packet is a request, the module creates a reply and sends it out. If the received packet is a redirection message, the module uses the information to update the routing table. If the received packet is an error message, the module informs the protocol about the situation that caused the error.

The output module is responsible for creating request, solicitation, or error messages requested by a higher level or the IP protocol. The module receives a demand from IP, UDP, or TCP to send one of the ICMP error messages. If the demand is from IP, the output module must first check that the request is allowed