CHAPTER 19

Lab Assignments for Chapter 19

Although the Internet uses several protocols at the network layer, we have included only two lab assignments for this chapter. We have also included two lab report sheets, which means that each lab should be reported in a separate sheet. It is assumed that you have done the lab assignment for Chapter 1, which told you how to install the Wireshark software and how to use it.

The first assignment, Lab19-1, is about the IP protocol. The second, Lab19-2, is about the ICMP protocol. We have included no lab assignments for routing protocols, such as RIP, OSPF, or BGP, because these protocols handle communication between routers. To be able to capture packets transmitted by these protocols, we need to have access to the routers between two hosts, which is not the intension of these lab assignments.

19.1 LAB19-1: IP

The Internet Protocol (IP), the main protocol in the Internet layer of the TCP/IP protocol suite, is used to route datagrams from source to destination based on the logical (IP) addresses. In this lab we capture and study packets belonging to the current IP version (IPv4). We do not deal with IPv6 because part of the Internet do not support IPv6 yet.

Like UDP, IP is a connectionless protocol. This means that we cannot find any frame with the source or sink at the network layer. An IP header can be captured only when an upper-layer protocol uses IP.

19.1.1 Assignment

- Start your web browser and clear the browser's cache memory, but do not access any site yet.
- Open Wireshark and start capturing.
- Go back to your web browser and retrieve any file from a site. Wireshark starts capturing packets.
- After enough packets have been captured, stop Wireshark and save the captured file.

- In the packet list pane, select any packet. In the packet detail pane, select the **Internet Protocol**. The hexdump of the IP header will be highlighted in the packet byte lane.

Questions

From the captured information, answer the following questions in your lab-report sheet.

1. Using the hexdump and consulting Figure 19.2 in the textbook, determine
   a. IP version.
   b. header length and number of bytes in the header.
   c. service type.
   d. total length.
   e. identification.
   f. set flags.
   g. fragmentation offset.
   h. TTL value.
   i. upper-layer protocol.
   j. checksum.
   k. source IP address.
   l. destination IP address.
2. Are answers to question 1 verified by information in packet detail pane?
3. If the checksum field in the packet detail pane is marked correct, can we conclude that the IP payload is not corrupted? Explain.
4. Is the datagram fragmented? Explain.
5. Does source or destination address belong to one of the special addresses? If yes which one?
6. How many bytes of data are in IP payload?

19.1.2 Documents to Turn in

1. A copy of the Lab19-1 report sheet that contains answered questions.
2. A printout of the supporting captured information.

19.2 LAB19-2: ICMP

The Internet Control Message Protocol (ICMP) is an auxiliary protocol at the network layer. It is designed for two purposes. It reports errors (unexpected conditions) about IP, UDP, and TCP. It can also be used to check the liveliness of the hosts or routers or finds the route followed by an IP datagram. Although we can see two versions of this protocol, ICMPv4 and ICMPv6, in use today, we discuss only ICMPv4 which is more common.

ICMPv4 uses two categories of packets, error-reporting and query. Figure 19.8 in the textbook gives the general format for each category. We also recommend to check the extra materials for Chapter 19 at the book website if you need to find the exact format for each type of packet.

In this assignment, our goal is to capture and analyze ICMPv4 packets (both error-reporting and query types). We can easily create ICMPv4 query messages using a program called **ping** as shown in Example 19.11 on Page 578 of the textbook. Creating error-reporting packets is more tricky. These packets are created automatically when there is an error in the path. We cannot wait for an error to occur in the path, but we can artificially force a condition to make IP create an error-reporting ICMP packet. This can be done using a program called **traceroute** (in Unix-like environment) or a **tracert** (in Windows environment) as shown in Figure 19.10 on Page 580 of the textbook. We use the *ping* and *tracert* in this document to capture ICMP packets.

19.2.3 Using The *ping* Utility

In this section of the assignment, we use the *ping* utility to capture ICMPv4 query packets: *echo request* and *echo reply* (See the extra materials for Chapter 19 at the book website for more information). The *ping* utility is, in fact, a client-server program. The client program, which needs to be invoked at the command prompt, triggers an *echo request* packets; the server program, which is running at the background all the time, waits for a signal from an *echo request* message, and triggers an *echo reply* message. The ping program, however, does not insert a message in the ICMPv4 packets, it just simply triggers its creation and provides the values for *identification* and the *sequence number* fields.

We have divided this section into two parts: Part I and Part II. In Part I, we want to see the messages exchanged at the ping level. In part II, we want to capture frames carrying ICMPv4 packet, which are encapsulated in an IPv4 packet.

Part I: Analyzing Ping

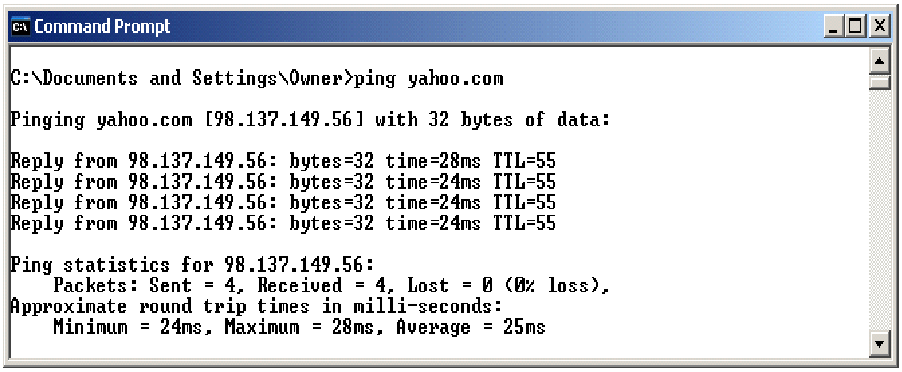
- Open the Wireshark and start packet capturing. Although, we are not using the these frames Part I, we will use them in Part II of the assignment section.
- Open Command Prompt and type ***ping*** *hostname*. The *hostname* can be the domain name or the IP address of a site you know (be sure that there is no firewall to filter out the packets). An example of the result of the ping command in the Command Prompt window looks as shown in Figure 19.1.
- Return to the Wireshark window, stop packet capturing, and save the captured file.

Questions

Using the result of the ping command in the Command Prompt window, answer the following questions in your lab report.

1. What is the destination IP address?
2. How many ping messages are sends?
3. How many bytes of data are in each ping message?
4. What is the round-trip time for each packet?

Figure 19.1 *Example of using ping*



```
C:\Documents and Settings\Owner>ping yahoo.com

Pinging yahoo.com [98.137.149.56] with 32 bytes of data:

Reply from 98.137.149.56: bytes=32 time=28ms TTL=55
Reply from 98.137.149.56: bytes=32 time=24ms TTL=55
Reply from 98.137.149.56: bytes=32 time=24ms TTL=55
Reply from 98.137.149.56: bytes=32 time=24ms TTL=55

Ping statistics for 98.137.149.56:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 24ms, Maximum = 28ms, Average = 25ms
```

5. What are the minimum, average, and maximum round trip times?

Part II: Wireshark Capture of Ping Commands

- Open the file you captured in Part I.
- In the Filter field of the Wireshark window type icmp (lower case) and click **Apply**.

Questions

Using the result of the information in the captured file, answer the following questions in your lab-report sheet.

1. What is the destination IP address of Echo request ICMP messages? Does the result agree with the information in Part I of the lab?
2. How many Echo request ICMP packet are in the packet list pane? How many Echo reply ICMP packets are in the packet list pane? Does the result agree with the information in Part I of the lab?
3. How many bytes of data are carried by each ICMP packet? Does the result agree with the information in Part I of the lab?
4. Evaluate the difference between the time the first Echo message was sent and the time the first reply message was received. Does the result agree with the information in Part I of the lab?
5. Comparing Echo request ICMP messages with Echo reply ICMP message, determine
   a. What fields are the same? Explain the reason.
   b. What fields are different? Explain the reason.
6. Comparing all Echo request ICMP messages, determine
   a. What fields are the same? Explain the reason.

b. What fields are different? Explain the reason.

19.2.4 Using *traceroute* or *tracert* Utility

To capture some ICMPv4 error-reporting packets, we can use either the *traceroute* utility (in Unix-like environment) or the *tracert* utility (in the Windows environment). The *traceroute* utility is a client-server at the application layer that uses the services of UDP; the *tracert* utility is a client-server program that uses the services of IP. In this assignment, we use *tracert,* but the assignment can be easily changed to use *traceroute* if Unix-like environment is available.

The *tracert* utility in Windows is different from the *traceroute* utility in Unix. In this case, instead of sending UDP with an unavailable port number, *tracert* sends (n +1) ICMP echo messages encapsulated in IP datagrams with TTL values 1, 2, 3, and so on. When a router receives one of these packets, it decrements the value of the TTL field and sends a time-exceeded message when the TTL value becomes zero. When undropped echo message reaches the destination host, an echo response message is sent back to the source host.

We have divided this section into two parts: Part III and Part IV. In Part III, we want to see the messages exchanged at the *tracert* level. In part IV, we want to capture frames carrying ICMPv4 packets, which are encapsulated in an IPv4 packet.

Part III: Analyzing tracert

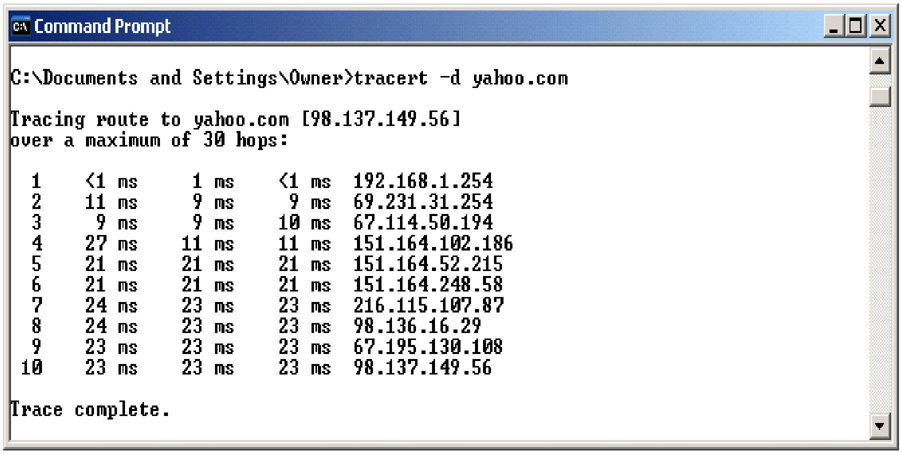
- Open the Wireshark and start packet capturing. Although, we are not using the captured frames in Part I, we will use them in Part IV.
- Open Command Prompt and type ***tracert –d*** *hostname*. The *hostname* can be the domain name or the IP address of a site you know. The ***–d*** option prevents tracert from resolving the IP addresses to their names and slowing down the process of displaying information. The result of the *tracert* command in the Command Prompt window looks as shown in Figure 19.2.
- Return to the Wireshark window, stop packet capturing, and save the captured file.

Questions

Using the result of the *traceroute* command, answer the following questions in your lab-report sheet.

1. How many probe packets are sent from the source to the destination for each TTL value?
2. What is the first IP address in the list? This is the IP address of the default router, the host on the local subnet that provides the physical connection to remote networks.
3. How many routers are between the source and the destination?
4. What is the IP address of the destination?

Figure 19.2 *The result of using tracert*



```
C:\Documents and Settings\Owner>tracert -d yahoo.com

Tracing route to yahoo.com [98.137.149.56]
over a maximum of 30 hops:

  1    <1 ms     1 ms    <1 ms  192.168.1.254
  2    11 ms     9 ms     9 ms  69.231.31.254
  3     9 ms     9 ms    10 ms  67.114.50.194
  4    27 ms    11 ms    11 ms  151.164.102.186
  5    21 ms    21 ms    21 ms  151.164.52.215
  6    21 ms    21 ms    21 ms  151.164.248.58
  7    24 ms    23 ms    23 ms  216.115.107.87
  8    24 ms    23 ms    23 ms  98.136.16.29
  9    23 ms    23 ms    23 ms  67.195.130.108
 10    23 ms    23 ms    23 ms  98.137.149.56

Trace complete.
```

Part IV: Wireshark Capture of tracert

Open the file you captured in Part III, type icmp (lower case) in the filter field and click **Apply**.

Questions

Using the result of the capture, answer the following questions in your lab report.

1. How many ICMP packets are in the packet list pane?
2. How the number of the ICMP packets is related to the number of network visited in Part III of the lab?
3. Except for the last few ICMP packets, every echo request ICMP packet in the packet list pane is followed by a time-to-live exceeded ICMP packet. In the packet detail pane, open an echo request ICMP packet and the time-to-live exceeded ICMP packet that follows it. Describe the content of the two packets. How the content of the time-to-live exceeded ICMP packet is related to the content of the corresponding echo request ICMP packet
4. The last few echo-request ICMP packets are followed by the echo-reply ICMP packets. What are the TTL values of these packets. Using the TTL values, find the number of routers visited between the source host and destination host.
5. What is the source IP address of the echo reply messages? Which entity does this address define?

Documents to Turn in

1. A copy of the Lab19-2 report sheet that contains answered questions.
2. A printout of supporting captured information.