LAB MANUAL
(DHCP AND NAT)

Section 1: Investigating DHCP Behavior

Objective:

In this lab, we will explore the behavior of the **Dynamic Host Configuration Protocol (DHCP)**. DHCP is used in various network environments (corporate, university, home) to dynamically assign IP addresses and configure other network settings. The goal is to capture and analyze the four key DHCP messages: **Discover**, **Offer**, **Request**, and **ACK**. In this lab, we'll take a quick look at the Dynamic Host Configuration Protocol, DHCP. Recall that DHCP is used extensively in corporate, university and home-network wired and wireless LANs to dynamically assign IP addresses to hosts, as well as to configure other network configuration information.

we'll be studying the DHCP Discover, Offer, Request and ACK messages shown in Figure

As we've done in earlier Wireshark labs, you'll perform a few actions on your computer that will cause DHCP to spring into action, and then use Wireshark to collect and then the packet trace containing DHCP protocol messages.

Gathering a Packet Trace

The first two steps in the DHCP protocol in Figure 1 (using the Discover and Offer messages) are optional (in the sense that they need not always be used when, for example, a new IP address is needed, or an existing DHCP address is to be renewed); the Request and ACK messages are not. In order to collect a trace that will contain all four DHCP message types, we'll need to take a few command line actions on a Mac, Linux or PC.

On a Linux machine:

1. Open a terminal window and enter the following commands to release the current IP address:
    a. **sudo ip addr flush en0**
    b. **sudo dhclient -r**

Here, en0 represents the network interface you will capture packets from. You can find your interface name in Wireshark by selecting **Capture -> Options**.
This command will remove the existing IP address of the interface, and release any existing DHCP address leases.

2. Start Wireshark, capturing packets on the de-configured interface.
3. In the terminal window/shell, enter the following command:
    a. **sudo dhclient en0**

where, as with above, en0 is the interface on which you are currently capturing packets. This will cause the DHCP protocol to request and receive an IP address and other information from the DHCP server.

4. After waiting for a few seconds, stop Wireshark capture.

On a windows:

1. In a command-line window enter the following command:
    a. **ipconfig /release**
        - This command will cause your PC to give up its IP address.
2. Start Wireshark and capture packets.
3. In the command-line window enter the following command:
    a. **ipconfig /renew**

    This will cause the DHCP protocol to request and receive an IP address and other information from a DHCP server.
4. After waiting for a few seconds, stop Wireshark capture.

On a Mac:

1. Open a terminal window and de-configure the network interface with the following command:
    a. **sudo ipconfig set en0 none**
        i. Again, en0 is the network interface name, which can be found in Wireshark under Capture -> Options.
2. Start Wireshark, capturing packets on the de-configured interface.
3. Reconfigure the interface and request a new IP address using:

a. **Sudo ipconfig set en0 dhcp**

4. After a few seconds, stop the Wireshark capture.

Verifying the Capture

Once you've stopped the Wireshark capture, check if you've successfully captured the DHCP packets. To do this, enter dhcp into the display filter field in Wireshark to view only DHCP-related traffic. You should see the following four messages:

- **DHCP Discover** (Sent by the client to request an IP address)
- **DHCP Offer** (Response from the server offering an IP address)
- **DHCP Request** (Client requesting the offered IP address)
- **DHCP ACK** (Server acknowledging the client's request)

Figure 1 shows an example of what the Wireshark display might look like after filtering for DHCP messages.

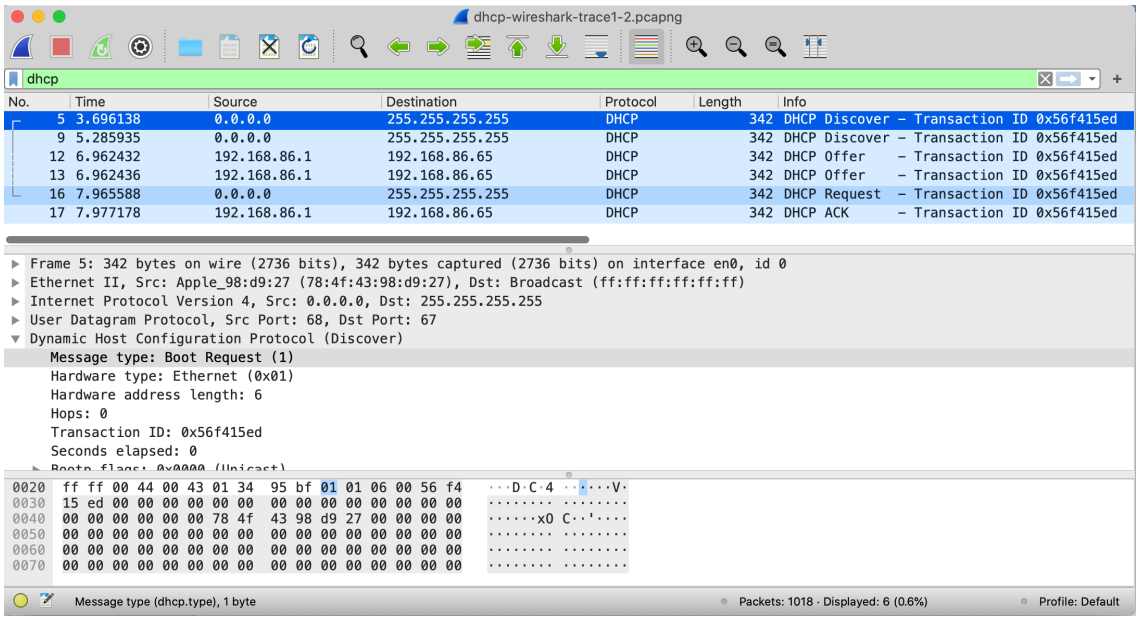

Figure 1: Wireshark display, showing the capture of DHCP Discover, Offer, Request and ACK messages

Using a Pre-Captured DHCP Trace File

If you are unable to capture the four DHCP messages on a live network, or your instructor has assigned you to use a pre-captured trace, you can download and analyze the provided DHCP trace file named **dhcp-wireshark-trace1-1.pcapng.** This trace file contains all the DHCP messages captured during a live session and can be analyzed in Wireshark using the same filtering technique mentioned above.
steps above on one of the author's computers. You may well find it valuable to download this trace even if you've captured your own trace and use it, as well as your own trace.

Section 2: Investigating NAT Behavior

Objective:

In this lab, we will explore the behavior of a NAT (Network Address Translation) router. This lab differs from previous Wireshark labs in that we will capture packets from two different points—on the LAN side and the WAN side of the NAT router—to observe how NAT translates IP addresses and port numbers.

NAT Measurement Scenario

In this lab, we'll capture packets containing a simple HTTP GET request message from a client inside a home network to a remote server, and the corresponding HTTP response from that server. Within the home network, the home network router provides a NAT
Figure 1 shows our Wireshark trace-collection scenario. We'll capture packets in *two* locations, and thus this lab has *two* trace files:

Trace Files:

1. **LAN Side Capture**: Packets received at the local area network (LAN) side of the NAT router. All devices in the LAN use addresses in the 192.168.10/24 range. The capture file is named nat-inside-wireshark-trace1-1.pcapng.
2. **WAN (Internet) Side Capture**: Packets sent from the NAT router to the Internet-facing side. This file is named nat-outside-wireshark-trace1-1.pcapng.

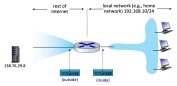

Figure 1: NAT packet capture scenario

Steps to Perform:

1. LAN Side Analysis:

- Open nat-inside-wireshark-trace1-1.pcapng in Wireshark.
- Locate the HTTP GET request addressed to the web server at IP address 138.76.29.8.
- Examine the following details:
  - Client's IP address (private IP) and the source port of the HTTP GET request.
  - The destination IP address (server) and the destination port in the TCP segment.
  - The timestamp of the HTTP GET request.
- Now, look for the HTTP response from the server (200 OK). Identify:
  - The source IP and port (the server's information).
  - The destination IP and port (the client's details).

2. WAN Side (Internet) Analysis:

- Open nat-outside-wireshark-trace1-1.pcapng in Wireshark.
- Find the HTTP GET request corresponding to the same request seen in the LAN-side capture. Pay attention to:
  - The time when the HTTP GET message appears in this trace.
  - The changes in the IP addresses and port numbers as a result of NAT translation.
- Next, locate the HTTP response (200 OK) from the web server. Compare the source and destination IP addresses and port numbers in this response with those seen on the LAN side.

3. Compare and Analyze:

- Compare the LAN-side and WAN-side captures for both the HTTP GET and HTTP 200 OK messages.
- Identify the differences in:
  - IP addresses (private vs public).
  - Port numbers (before and after NAT translation).
- Understand how NAT modifies these fields to facilitate communication between devices inside the LAN and external servers on the Internet.

References:

https://www.wireshark.org/download.html
https://gaia.cs.umass.edu/kurose_ross/index.php