CS 2690 Packet Sniffer Lab
Using Wireshark v3.0.7 and Npcap v0.9983

Lab Exercise 2: Dissecting ARP, DHCP and ICMP

Approximate time required: 2 hours

Introduction

I this lab, we'll reinforce our understanding of the ARP, DHCP and ICMP protocols discussed in lectures 4 and 5. Each section of this lab can be completed as a separate activity. Use the following slides for reference.

Lecture	Slide Title
4	The ARP Broadcast Process
4	DHCP Message Exchange for New Client Boot Up
5	Essential ICMP Concepts

You can use a WiFi or Ethernet network. It is possible to complete this exercise using Linux or macOS, however it will be slightly more complicated with those operating systems due to variations in DHCP implementations. A virtual machine won't work well for this lab. The final **Inspecting ICMP** section will be easier to do if you are at home or on some other relatively "quiet" network". (Locating ICMP packets can sometimes be a challenge on the busy UVU WiFi network.) If you don't have access to a Windows computer, I recommend that you use one of the Windows computers in the Network Research Lab (CS 516). See Lab 1 for CS 516 access information.

The ipconfig and ifconfig Utilities

The `ipconfig` utility in Windows and macOS (`ifconfig` in Linux) provides a way of viewing and modifying some of the parameters related to the IP layer. If you are using your own computer, log in with administrator privileges. Open a command line (terminal) and try one of the following to view available options:

Windows:	`ipconfig /?`
macOS:	`ipconfig`
Linux:	`ifconfig –h`

(For macOS and Linux, check the man pages for additional information.)

Next, try one of the following:

In Windows, use `ipconfig /all`

In macOS, use `ipconfig getpacket en?` where `?` is your network interface (normally `en0` for Ethernet or WiFi, possibly `en1`, `en2`, `en3` or `en4` if both adapters are enabled).

In Linux, use `ifconfig eth?`, where `?` is your network interface (often `eth0` for Ethernet or `eth1` for WiFi).

Expand the command line window containing the output from the previous command so that as much of the output as possible is visible (drag the bottom border), *make a screen shot, and include this with your lab report.*

Inspect the output from this command and answer the following questions. Your computer is likely to have multiple network adapters (Wi-Fi, Ethernet, Bluetooth, Npcap loopback), so be sure to answer these questions for the adapter that is currently active.

1. What is the IPv4 address of your computer? What is the class of that address?

   In macOS, your computer's IP address will be labeled "yiaddr".

   In Linux, your computer's IP address will be labeled "inet addr".

   (An IPv4 address will be shown in dotted decimal format (e.g, `100.150.200.250`. An IPv6 address will be delimited with colons and represented in hexadecimal (e.g., `2001:0db8:85a3:0000:0000:8a2e:0370:7334`.)

2. What is the subnet mask for the network that you are connected to? List it in both dotted decimal and "slash" notation. Is this the default mask for your IP address? If not, how many additional subnet bits have been added? What is the subnet number, i.e., the network number for the entire IPv4 subnet? (Don't include the host bits.)

3. Do you have a default gateway/router? If so, what is its IP address?

   In macOS, the IP address of your default router can be found in the `ipconfig getpacket en?` output labeled "router (ip_mult)".

   In Linux, the IP address of your default gateway can be found by typing the `route` command and reading the row labeled "default".

4. Is DHCP (not DHCPv6) enabled? If so, what is the DHCP server's IP address? The DHCP server process will often be running on the default router/gateway. Is that the case on your network? What is the length of your DHCP lease?

   In macOS, DHCP server information is grouped under the "server_identifier" label in the `ipconfig getpacket en?` output. Lease time is specified in seconds, and the '`0x`' notation indicates a hexadecimal value.

   In Linux, you will probably need to do something like this (depending on the Linux distribution): `cat /var/lib/dhcp/dhclient.leases`, and then find the appropriate interface.

Analyzing ARP

Locate another machine *on your local network* that will respond to a ping, and write down its IP address. (If you don't have another computer on your local network, you may be able to ping your router, which is often `192.168.1.1.` If you ping your router, you will need to do the things described on the next page quickly, so read through it first to see what is involved.)

> If you can't get a computer to respond to a ping with an ICMP **Echo Reply**, try using the `-4` option (for example, `ping -4 192.168.1.1`) to override sending an IPv6 ping, and check the firewalls on both computers to make sure that ping/ICMP messages aren't being blocked. Some operating systems block ICMP by default. (See **Allowing Inbound ICMP with Windows Firewall** on the last page.)

From the command line window, type `arp -a` (`arp` in Linux) to display your current ARP table. You may see a separate ARP table for each active interface. Addresses displayed in the middle column are the MAC addresses that correspond to the IPv4 addresses to the left.

> In Windows, a hyphen separates each byte, instead of the usual colon. Individual *dynamic* entries will time out a few minutes after communication with the device in that row ends. *Static* entries like IPv4 Limited Broadcast (`255.255.255.255`) are essentially permanent.

Start Wireshark, select the desired network interface (**Capture | Options**), set promiscuous mode off at the lower left, and begin capturing packets.

Type one of the following to clear any existing entries in the ARP cache:

Windows 10	`arp -d *` or `netsh interface ip delete arpcache`
(Administrator privileges required for both)	
Windows 7	`arp -d *`
Linux	`sudo ip -s -s neigh flush all`
macOS	`sudo arp -da`

If you are running Windows with User Account Control settings enabled, you may receive the following error message:

```
The ARP entry deletion failed: The requested operation
requires elevation.
```

This is an indication that administrative privileges are required to use the `-d` option on your computer. In Windows 10, search on "command", right click on the **Command Prompt** button, and click **Run as administrator**. In Windows 7, click the **Start** button at the lower left of your screen. Then click **All Programs | Accessories**. Right click **Command Prompt**, and click **Run as administrator**.

Ping the IP address of the machine that you located earlier. Stop the ping after a few ping replies have been received, and stop the Wireshark packet capture.

Create a display filter (using lower case characters) that displays only ARP traffic. With this filter enabled, you should see only ARP messages in the "Protocol" column of the *Packet List* (top) pane. Select the first ARP message in the *Packet List* pane that was transmitted by your computer *to the target computer's IP address*, and which says the following in the right-hand *Info* column:

```
who has [target IP address]?  Tell [your IP address]
```

Note that other ARP requests (to your router, for example) may have been captured before and after the one sent to the target computer.

Because ARP messages are not IP packets, Wireshark will display MAC addresses in the "Source" and "Destination" columns of the *Packet List* pane, in place of the usual IPv4 addresses. In most cases, Wireshark will replace the three left-hand bytes of the MAC addresses (the OUI codes) with an abbreviation of the network adapter manufacturer's name (e.g., "Cisco" or "IntelCor").

> You may need to sort through ARP request messages sent by other computers in the Packet List pane to find one transmitted by your computer. If your computer runs any excessively "chatty" services, use Task Manager to terminate that program for the duration of this exercise. Otherwise, it may repopulate your ARP cache more rapidly than you can clear it.

Answer the following questions:

5. What was the MAC address of the interface on your computer used for the Wireshark capture? (Give the 6-byte hex value.)

6. What is the two-byte hexadecimal Ethernet Type code for ARP?

7. Is this message transmitted as a MAC layer unicast or broadcast? How can you tell?

8. Is the ARP message carried within an IP packet? Can it be forwarded by a router?

9. What is this ARP message's type/Opcode (request or reply)?

10. Explain the function of this ARP message.

Locate the response to this ARP message. Be careful – this may *not* be the next ARP message in the Wireshark capture file.

> If you can't find the matching response, the computer that you sent the `ping` to is not operational, not running TCP/IP, or not responding to `ping` requests. Or you may have terminated the Wireshark capture prematurely. Clear the ARP cache again, and try pinging a different machine *on your own network* until you receive an ARP response.

11. What is the Opcode of this ARP message (request or reply)?

12. Is this message transmitted as a MAC layer unicast, multicast or broadcast?

13. What is the MAC address of the computer that sent this message? (This computer will be the target of the first ARP message.)

Save or discard your current Wireshark capture file. (You won't need it any more for this lab.)

Dissecting DHCP

In this section, you will modify the TCP/IP configuration of your computer. *Follow the directions carefully so that you are able to restore your normal IP configuration afterwards. Don't try this on an employer's computer without permission.*

We are going to trigger the DHCP message exchange illustrated in the Lecture 4 slide titled "DHCP Message Exchange for New Client Boot Up". These messages are generated when a computer that is configured to use DHCP starts up. This happens at Windows startup, for example, if you have set *TCP/IPv4 Properties* to "Obtain an IP address automatically". You can't run Wireshark while your computer is booting, of course, so we will use an alternate method to trigger DHCP after booting, and while Wireshark is running.

If you determined, in Question 4, that DHCP is enabled, then you will need to disable it temporarily using the process for your operating system described below. If DHCP is not enabled, skip to the **If DHCP Is Not Enabled** section below.

Disabling DHCP in Windows

In Windows, open **Control Panel**, click **Network and Sharing Center | Network and Internet** or **Network and Sharing Center**. Then click **Change adapter settings** at the upper left, right click on your active network adapter, and click **Properties**. Select **Internet Protocol Version 4 (TCP/IPv4)** and click **Properties**.

Click the *Use the following IP address* radio button. Enter a phony IP address like `10.0.0.100`. In the *Subnet mask* field, verify the corresponding default mask, then click **OK** and **Close**. Give your computer a moment to process the configuration changes. This will disable DHCP, causing your computer to "forget" the boot parameters previously provided by the DHCP server on your network, and disconnect you from your network. Skip to the **Continue Here** section below.

Disabling DHCP in macOS

Go to **System Preferences | Network**, and select the appropriate interface. **Click the Advanced** button and select **TCP/IP** at the top. Change **Configure IPv4: Using DHCP** to **Manually.** Enter a phony IPv4 host address like `10.0.0.1`. (Any class A/B/C host address will work.) In the *Subnet mask:* field, enter the corresponding default mask. Use an address for the **Router:** that is on the same network (such as `10.0.0.254)`, click **OK**, and then **Apply**. This will disable DHCP, causing your computer to "forget" the boot parameters previously provided by the DHCP server on your network, and disconnect you from your network. Skip to the **Continue Here** section below.

Disabling DHCP in Linux

Most Linux distributions have a GUI Network Connection Manager tool. In Ubuntu 12, for example, the ↑↓ icon at the upper right on the Unity desktop provides an **Edit Connections…** option. Select the *Wired* tab, then **Wired connection 1** and **Edit…**, then *IPv4 Settings*. That will provide access to the necessary configuration options. Change *Method:* to **Manual** and enter a phony IP address like 10.0.0.1. In the *Netmask* field, enter the corresponding default mask (e.g. 255.0.0.0), and save your configuration. This will disable DHCP, causing your computer to "forget" the boot parameters previously provided by the DHCP server on your network, and disconnect you from your network. Skip to the **Continue Here** section below.

If DHCP Is Not Enabled

If DHCP is *not* enabled on your system (possible, but unlikely), you are using a static IP address and other parameters assigned by your ISP or system administrator.

In Windows 7, 8 or 10, open **Control Panel**, click **Network and Sharing Center | Network and Internet** or **Network and Sharing Center**. Then click **Change adapter settings** at the upper left, right click on your active network adapter and click **Properties**. Select **Internet Protocol Version 4 (TCP/IPv4)** and click **Properties**. *Write down all of the addresses names, address values, button settings and masks listed under the* ”General” *tab (or save a screen shot), because you will need to manually reset them later on.* Click **OK** and **Close**.

For Linux and macOS, see **Disabling macOS DHCP** or **Disabling Linux DHCP** above for information on where to find the required parameters. *Write down all of the addresses names, address values, button settings and masks listed (or save a screen shot), because you will need to manually restore them later on.*

Continue Here

Now (for all operating systems), start Wireshark, select the appropriate interface and begin capturing packets. Use Windows Control Panel or the macOS or Linux equivalent to return to your network connection (as before), and (re)enable DHCP.

> To do this in Windows, select **Internet Protocol Version 4 (TCP/IPv4)** and **Properties**, click the *Obtain IP address automatically* button to (re)enable DHCP, click **OK** and **Close**.
>
> To do this in macOS, you’ll need to go back in to **System Preferences | Network**, and select the appropriate interface. **Click the Advanced** button and select **TCP/IP** at the top. Change **Configure IPv4: Manually** to **Using DHCP.** Click **OK**, and then **Apply**.

Let DHCP and Wireshark run for *at least 30 seconds*, and then stop the Wireshark capture.

Create a Wireshark display filter that shows only DHCP traffic. (Your machine will probably also be running DHCPv6 for IPv6, which we will ignore here.)

Confirm that you see only DHCP messages in the *Packet List* pane. If not, repeat the process from the beginning of **Dissecting DHCP** above.

If you were originally using a *static* IP address in Windows, restore that configuration.

> In Windows, go back into **Control Panel** and return to the IPv4 properties configuration pane for your network connection. Click the "Use the following IP address:" button, and *reset the IP address and any other information in the window back to its original value(s). Open a web browser and connect to a website on the Internet to confirm that the restored configuration is working correctly.*
>
> To do this in macOS, go back in to **System Preferences | Network**, and select the appropriate interface. **Click the Advanced** button and select **TCP/IP** at the top. Change **Configure IPv4:** to **Manually**. *Reset the IP address and any other information in the window back to its original value(s). Click **OK**, and then **Apply**. Open a web browser and connect to a website on the Internet to confirm that the restored configuration is working correctly.*

Compare the series of DHCP transactions displayed in Wireshark with the Lecture 4 slide titled “DHCP Message Exchange for New Client Boot Up”. If you *were* running DHCP initially, they should match. You may see multiple **DHCP Discover** and **DHCP Offer** messages.

If DHCP was *not* enabled initially, there is a small possibility that there may not be a DHCP server on your local network. In that case, you will observe one or more **DHCP Discover** messages being transmitted by your client, but no **DHCP Offer** messages being returned by a DHCP server.

In either case, select a **DHCP Discover** message in the *Packet List* pane and answer the following questions:

14. What Transport protocol is used to transmit DHCP messages?

15. Is the **DHCP Discover** message transmitted as an IP-level unicast, multicast, or broadcast message? How do you know?

Note that DHCP was based on a predecessor protocol called Bootstrap Protocol (BOOTP), and some diagnostic tools may identify a DHCP message as BOOTP. DHCP and BOOTP use nearly identical message formats, and the same UDP port numbers (67 & 68).

Drill down into the Dynamic Host Configuration Protocol (Discover) header .

16. What is the Message type?

17. What is the *DHCP* Message type? (Look for "Option 53" further down.)

*Highlight the first **DHCP Discover** message in the Packet List pane, make a screen shot that shows all of the DHCP messages in the Packet List pane (as well as the lower panes showing the contents of the highlighted packet), and paste it into your lab report.*

Inspecting ICMP

(If possible, do this section on a relatively quiet network.) The `ping` command sends an ICMP **Echo Request** and receives an **Echo Response**. So you have already generated one set of ICMP messages in Lab 1. Here we'll generate an ICMP **Time-to-live (TTL) Exceeded** message. **TTL Exceeded** is more typical of the main function of ICMP, which is reporting IP-level errors.

TTL Exceeded is returned by a router that decrements the TTL value in the IP header from 1 to 0. When that occurs, it is assumed that the packet has been misrouted and can't be delivered to its intended destination. The misrouted packet is discarded by that router, and the ICMP **TTL Exceeded** message returned by that router to the source host carries (at least) the beginning of the undelivered/expired packet as the ICMP data.

Prepare the following command in a command line or terminal window, but don't execute it yet…

```
Windows:            tracert wcg.net
macOS & Linux:      traceroute wcg.net
```

Start a Wireshark capture, execute the `tracert` or `traceroute` command, capture about 30 seconds worth of traffic, and then stop the capture. The `tracert/traceroute` command will transmit some special `ping` messages that should force routers in the path to `wcg.net` to respond with a series of ICMP **TTL Exceeded** messages.

Create a Wireshark display filter that shows only packets carrying ICMP messages. Activate your display filter and select the first ICMP **Time-to-live exceeded** message received in the (upper) *Packet List* pane. Be careful – other types of ICMP messages may have been captured as well.

> If you are using your own computer and you don't find any **Time-to-live exceeded** messages in Wireshark, it is most likely because your firewall is configured to block inbound ICMP messages. See **Allowing Inbound ICMP with Windows Firewall** on the last page for instructions on correcting this in Windows.

With the **Time-to-live exceeded message** selected, go to the *Packet Details* (middle) pane and expand the *Internet Protocol Version 4* header (just below the *Ethernet II* header). Note the following:

- The IP header's *Protocol* field is set to 1, specifying that the IP payload contains an ICMP header instead of the usual TCP (6) or UDP (17) Transport layer header.
- The IP source address of this inbound message is not `wcg.net`, the final destination (`64.200.241.26`). This suggests that some router in the path to `wcg.net` returned the **TTL Exceeded** message.

Expand the Internet Control Message Protocol header below the IP header, and compare what you see to Figure 1 below.

Ethernet Header 14 bytes	IPv4 Header 20 bytes	ICMP Header (TTL Exceeded) 8 bytes	ICMP Payload (Data) (IP header with TTL exceeded + message content)

Figure 1. Structure of ICMP TTL Exceeded Error Message

The ICMP header contains a Type 11, indicating a **TTL Exceeded** message.

Let's verify that the content of this ICMP message corresponds to a `ping` message previously transmitted by the `tracert/traceroute` command. Expand the IP header contained *within* the ICMP payload (see Figure 1) and write down the IP packet **Identification** number (shown in both hex and decimal). Verify that the **TTL** in this inner IP header is set to 1. (It won't be set to 0, since we are viewing the IP header that arrived at the router *immediately before its TTL expired*.) Finally, verify that the **Source IP Address** is your computer (i.e., this is the `ping` that you originally sent, being echoed back to you).

Return to the *Packet List* pane above and select the **Echo (ping) request** that your browser transmitted to trigger the **TTL Exceeded** error. This will typically be the `ping` that immediately precedes your **TTL Exceeded** message. Expand the IP header of that **Echo (ping) request** packet and verify that the packet **Identification** number in the IP header is the same as the number being returned in the ICMP **TTL Exceeded** message.

> There may be some delay between the transmission of the ping and the arrival of the ICMP **TTL Exceeded** message, so other traffic may be intermingled. If the **Identification** numbers don't match, work backwards in time, checking the next-most-recent **Echo (ping) request** until you find a match. You shouldn't have to look very far.

Select the ICMP **Time-to-live exceeded packet** again in the *Packet List* pane. *Make a screen shot that shows both the **Echo (ping) request** and the ICMP **Time-to-live exceeded** packets in the Packet List pane, and include that with your report.*

Summary of Deliverables

- Command line screen shot of `ipconfig /all`
- Answers to `ipconfig` questions 1 - 4
- Answers to ARP questions 5 - 13
- Answers to DHCP questions 14 - 17
- Wireshark screen shot of the DHCP **Discover** message
- Wireshark screen shot of **Echo (ping) request** and matching **TTL Time-to-live exceeded** messages

Submit the lab report to Canvas in word processed format. If you are working in the lab, log off and shut down your workstation. *If you modified your firewall to allow inbound ICMP, you may wish to disable or delete that rule.*

Allowing Inbound ICMP with Windows Firewall (only if needed)

Some versions of the Windows Firewall block inbound ICMP messages by default. To change this, start **Control Panel | Windows Firewall**. Choose **Advanced Settings | Inbound Rules | New Rule**.

On the *Rule Type* page, select **Custom** and click **Next**. On the *Program* page, click **Protocol and Ports**. Select **ICMPv4** from the *Protocol Type*: menu, and click **Next**. Take the defaults on the *Scope* page and click **Next**. Click **Allow the connection** on the *Action* page and click **Next**. Take the defaults on the *Profile* page (or uncheck **Public** and **Domain** if you are using a laptop), and click **Next**., Name your rule “Allow all inbound ICMP” and click **Finish**. You will need to reboot Windows to get the firewall changes to take effect.

For Windows 10, see…
https://docs.microsoft.com/en-us/windows/security/threat-protection/windows-firewall/create-an-inbound-icmp-rule

For Windows 7, see…
https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2008-R2-and-2008/cc972926(v=ws.10)

(You will need to do something similar if you are using a different “personal” firewall on your computer.)