IS8055556: Data and Computer Communications	**Recitation 12**
Semester 2 5785	**24 June 2025**
Lecturer: Michael J. May	**Tel Hai College**

Wireshark and UDP

In this recitation we'll use a few tools to analyze the behavior of UDP between computers in the network. We'll also explore how to write a UDP echo client.

1 UDP Echo Client

This week we'll introduce the basics of network communication using a UDP echo client and server. The code is written in Python. Both programs send packets back and forth between two computers.

We're going to use the echo client and server for future recitations and assignments, so it's worthwhile to spend time figuring out how they work.

2 About WireShark

WireShark is a popular free protocol analyzer and packet sniffer. WireShark is capable of recording packets that enter and exit the computer to create a *packet trace* of what data has entered and exited the computer. When running on Windows, WireShark only sees packets that have already passed the basic filtering done by the network card and Windows, so we can't see low level packets such as Ethernet frames not meant for the computer, link layer management frames, or network layer packets not destined for or sent by the recording computer. We can, however, see all network layer and higher packets sent by the recording computer and received by the recording computer. Two screen shots of WireShark are shown below. In Figure 1, the WireShark welcome screen is shown. In Figure 2, a sample trace is shown with filtering based on a single protocol type (DNS).

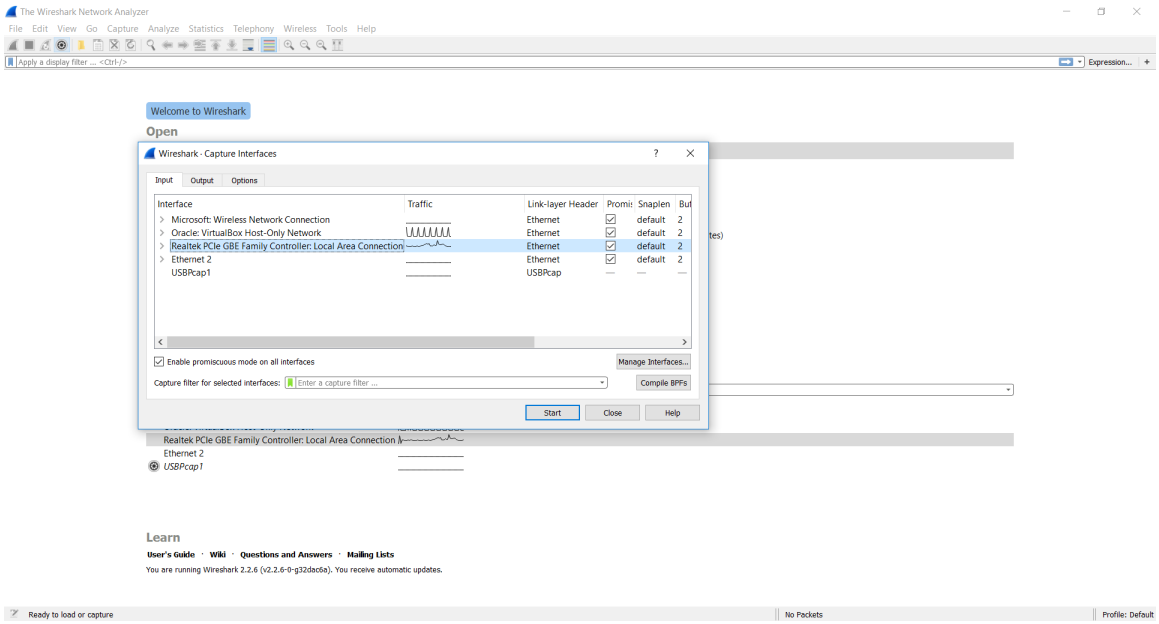

Figure 1: WireShark welcome page and capture interface

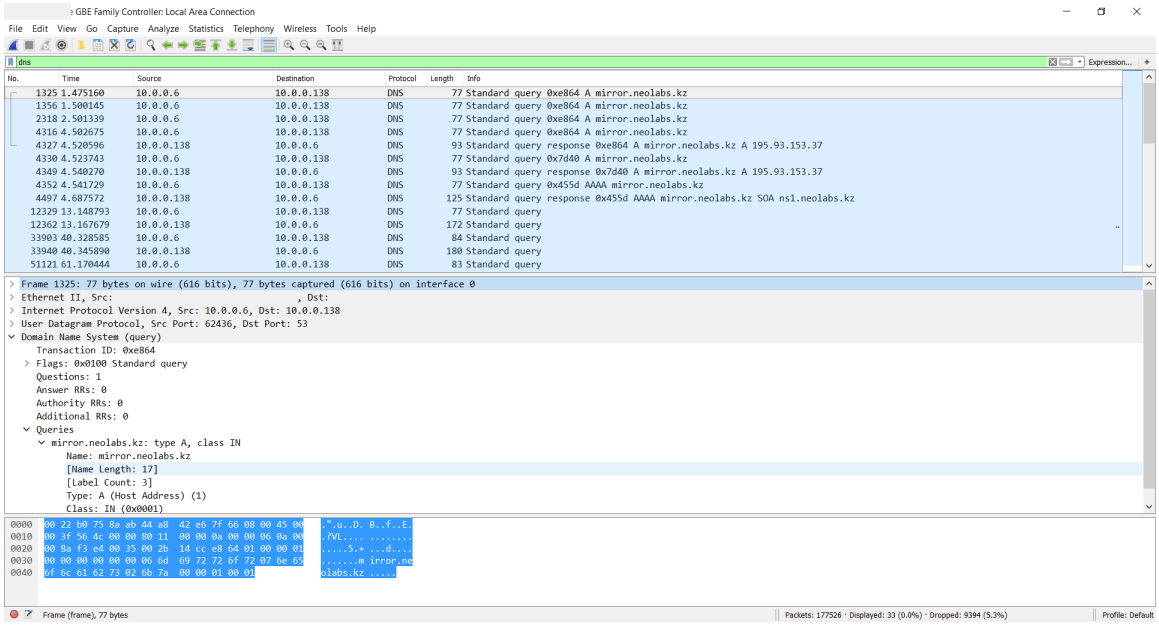

Figure 2: WireShark filtering a trace

In this recitation, we will use WireShark to see how Internet Protocol (layer 3) packets are sent by two computers communicating using TCP and UDP. WireShark will show us the packet trace from a *capture* on one of the computer's network interfaces. Most computers will have more than one capture interface available such as wired Ethernet (LAN), Wireless Network Connections (Wi-Fi), and Bluetooth. By clicking on the upper left hand icon ("List the available capture interfaces...") we see a list of interfaces that WireShark can listen to on a computer. A sample list of interfaces is shown in Figure 1. Note that you may see the fields "Packets" and "Packets/s" automatically update with the number of packets currently going in and out on one or more interfaces. Seeing the packet numbers go up can give you a hint about which interfaces are active and are interesting to listen on.

2.1 Capturing

You can select one or more interfaces using the checkboxes on the left hand side of the above window and start the packet capture by clicking on the "Start" button. Once the packet capture has started, you can stop the capture by clicking the "Stop the running live capture" button (the red square).

2.2 Filtering

You can filter packets as they are captured or after the capture has finished by entering a protocol name, protocol field value, or other conditional filter in the "Filter" text box. You can get help writing filters using the "Expression" button or by using the autocomplete feature of the Filter text box. For example, in Figure 2, a filter for "dns" is active so it only shows packets that conform to the DNS protocol.

Most protocols have intuitive names in the filter rules such as udp, tcp, dns, arp, and eth (Ethernet). Some names are not as intuitive, such as DHCP which is called "bootp" for historical reasons.

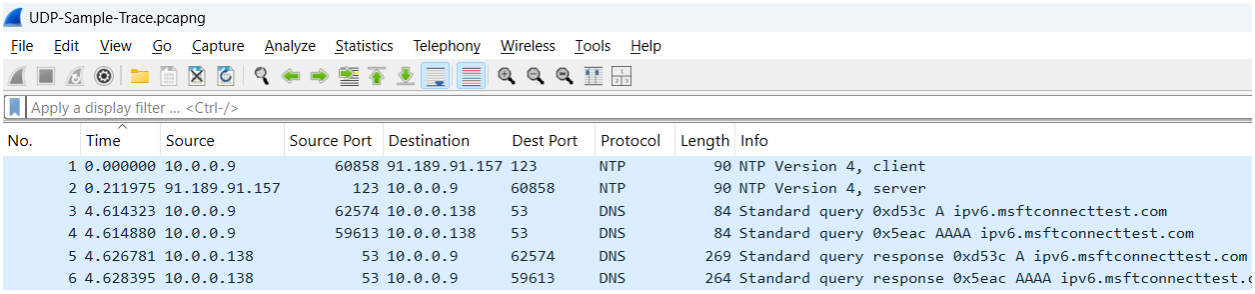

Figure 3: UDP Trace Contents

2.3 Examining a Packet

As shown in Figure 2, you can select a packet from a trace and examine its internal fields and values. In the packet selected in the figure, the DNS layer is opened and the DNS query fields are expanded. One query (for `mirror.neolabs.kz`) is opened. You can see the string in the raw byte window on the bottom along with other byte fields. The right part of the bottom pane attempts to map the hexadecimal values into ASCII characters. Since many are not printable characters, there are no characters shown for them.

We will use WireShark to examine the packets sent between two computers which talk in UDP and TCP. The traces will let us see the fields of all packets sent as well as higher level actions such as TCP's three way handshake and the interaction between packet data and acknowledgement (ACK) messages.

3 UDP and WireShark

Let's start with a simple UDP trace and analyze its contents. The trace is called UDP-Sample-Trace.pcapng and is found on Moodle.

First, open the file using WireShark. You should see 6 packets inside. The trace is shown in Figure 3. The trace labels each packet using two different methods:

1. Packet number - a sequential numbering of the packets in the trace
2. Time - the time in seconds the packet was sent/received relative to the start of the trace.

To make things a bit clearer, the screen shot above has 2 extra columns that you won't see in most traces by default - source port and destination port.

3.1 What to do: A Tour of the UDP Trace

Let's look through the packets step by step.

1. Select the packet from time 0.00000. What is the IP address of the computer that sent the packet? What's the IP address of the computer who is supposed to receive it?
2. What is the source port for the packet? What is the destination port for the packet?
3. What protocol is being sent here? How do you know? (Hint: Look at `https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers`)
4. Packet 0.00000 was a request. Find the packet that is the response for it. What time was is sent? How do you know it's the response?
5. Find the packet sent at time 4.614323. What are the IP address and port of the sender? What are the IP address and port of the destination?

6. What protocol is being sent here? How do you know? (Hint: Look at `https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers`)
7. The packet at 4.614323 is a request packet. Find the response packet. How do you know it's the response?

4 TCP and Wireshark

Looking at a trace of a download of a large file from `mirror.neolabs.kz` (195.93.153.37), we can use the filter to see packets which come from the remote server (see Figure 4). The first two packets show the SYN/ACK packet part of the three way handshake and the second one shows a regular ACK from the server.

If we click on the Analyze menu and select *Analyze* → *Conversation Filter* → *TCP* we can see just the packets in the conversation detailed. The result is shown in Figure 4. The conversation in the figure shows the full three way handshake in TCP (SYN → SYN/ACK → ACK) followed by the HTTP request which lead to the download (requesting `/linuxmint/iso/stable/18.1/linuxmint-18.1-cinnamon-64bit.iso`). Some of the following download can be seen in the segments below, including two ACKs from the client (10.0.0.6).

Selecting *Statistics* → *TCP Stream Graphs* → *Throughput* shows us a graph of the behavior of the TCP conversation (in this case it's Stream 6 - you can switch between streams using the bottom right hand side *Stream* selector). The result can be seen in Figure 5. Notice the slight ups and downs in the conversation over time due to occasional drops and out of order packets. Since the conversation took place more or less unimpeded, there isn't much to bother the download. If multiple conversations had been going on at the same time, we'd see more variance.

4.1 What to do: A TCP trace using Wireshark

Once you have Wireshark running, we will use it to track a TCP download.

1. Start a Wireshark packet capture on the wired or wireless interface that your computer is using.
2. Open your favorite browser and point it to a non-encrypted URL. Some examples you can try are:
   - `http://www.faqs.org/images/library.jpg`
   - `http://www.people.com.cn/img/2020peopleindex/img/copy_icon1.png`
3. Once the download is finished, stop the packet capture.
4. Find the download session in the packet trace. You can use the packet filter to find TCP and HTTP conversations. You may want to use the IP address of the remote web site when searching since there are likely to be multiple TCP sessions going on at once from your computer. The IP addresses for the above two web sites are:
   - `www.faqs.org` → 199.231.164.68
   - `www.people.com.cn` → 138.113.247.70
   - For any other website (*e.g.* www.example.com), you can use the `nslookup` tool from the command line. To find out www.example.com's IP address, enter the following command: `nslookup www.example.com`. You'll get an answer that should include one or more lines that are called "Non-authoritative answer". The address shown there is the IP address.
5. Use Wireshark to show only the TCP conversation for the download. How many packets were sent? Were any of them fragmented? Were any of them dropped?

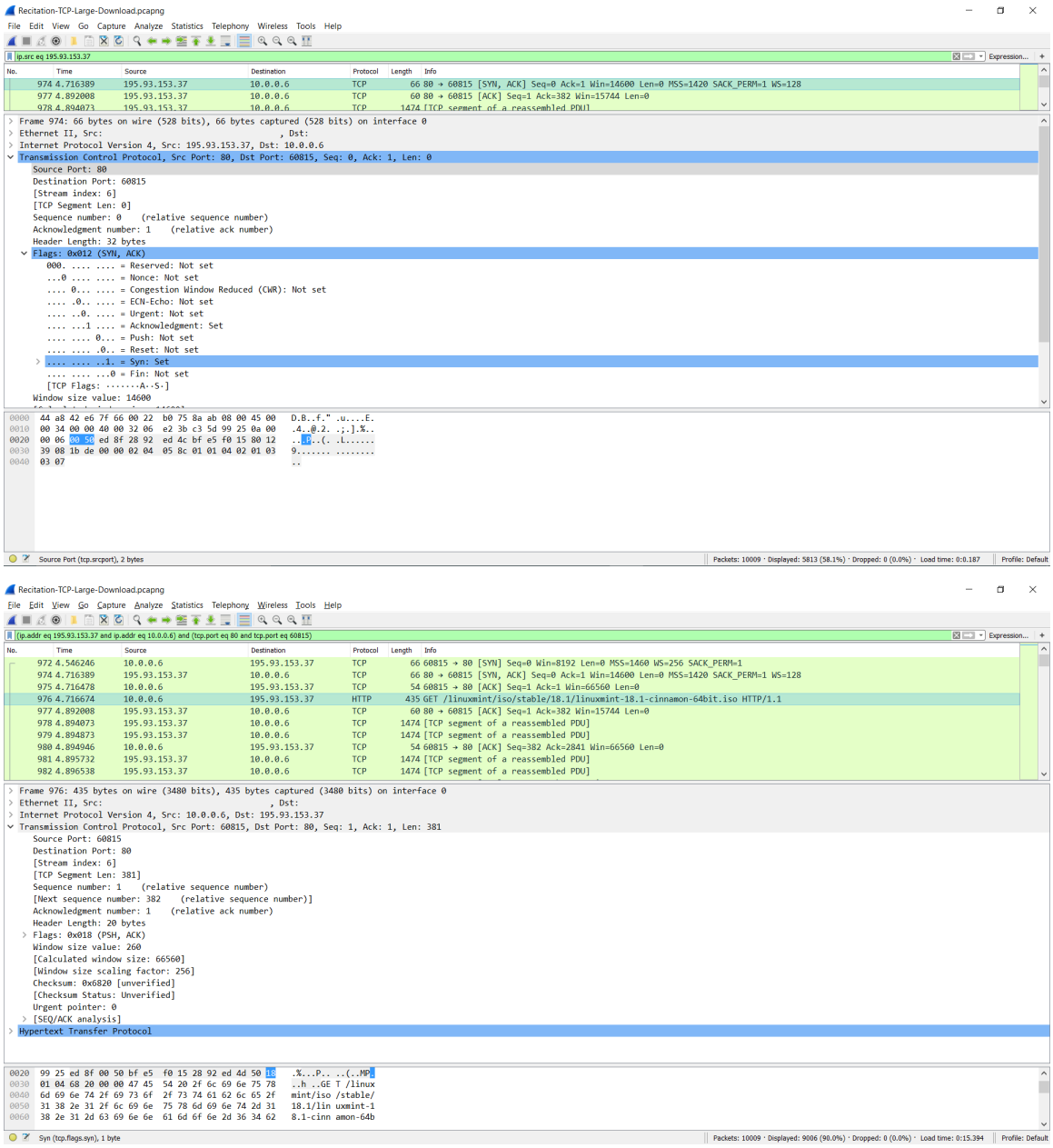

Figure 4: WireShark showing only packets from a single source (top) and showing a TCP conversation for an HTTP request (bottom)

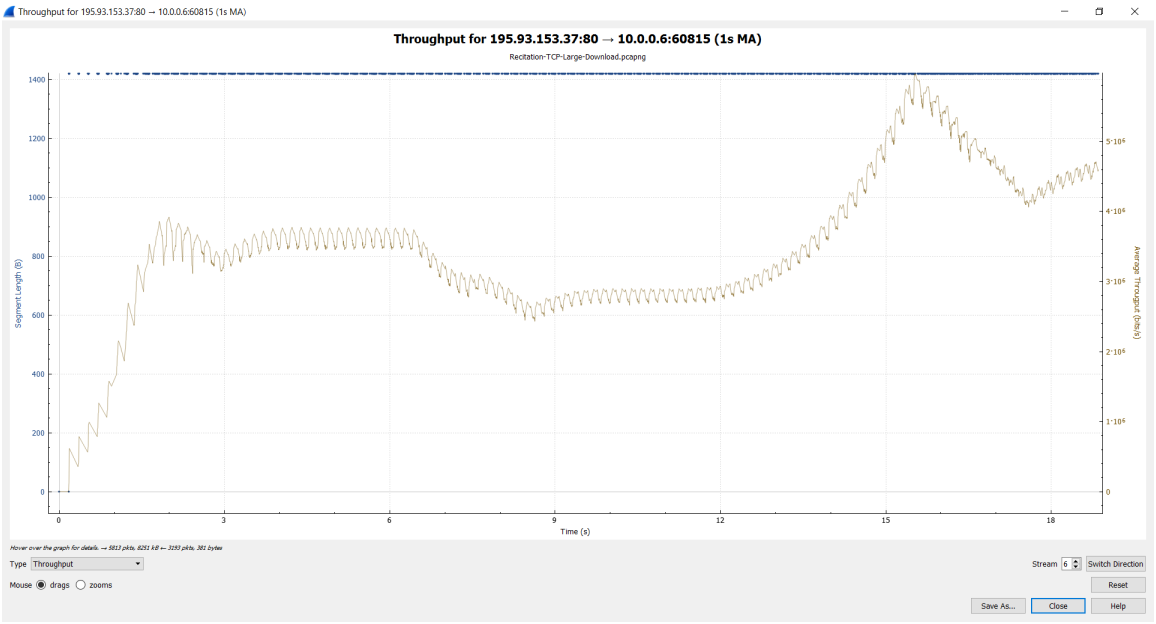

Figure 5: WireShark showing a TCP throughput graph for the download shown above

6. Use the TCP Stream Graphs feature shown above to display the throughput behavior for the conversation.

The previous steps are performed on a local website in the college. Repeat the steps above on a similar size file from a remote web site. For example, try the URLs: `http://www.zolsefer.co.il/image/users/171616/ftp/my_files/%D7%9E%D7%A1%D7%98%D7%A8%D7%A7%D7%90%D7%A8%D7%93.jpg?id=7938266` (Israel) or `http://www.milliondollarhomepage.com/index_files/logo-tm.gif` (USA). Do the TCP streams to far away locations behave any differently than the ones that are within Israel?