CIS600 Internet of Things Security and Privacy
Fall 2025
Week 2: In-class Exercise
Total: 100 points

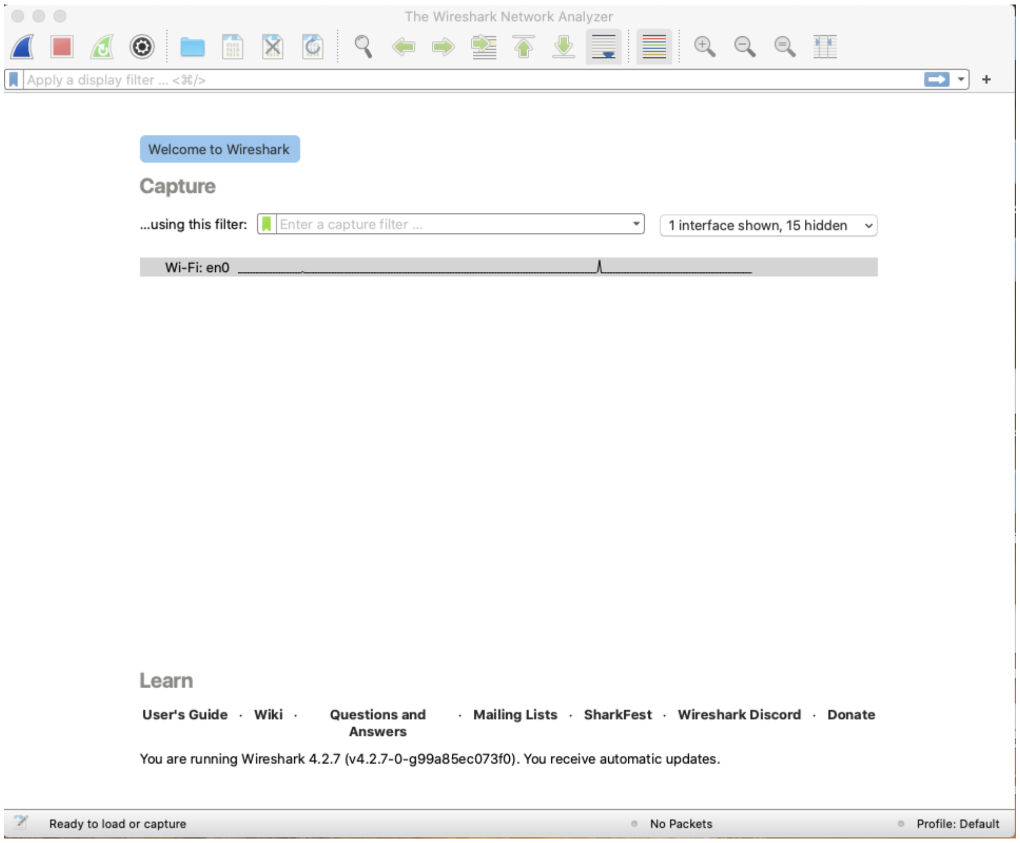
Packet Sniffing and Wireshark

Submission: Please submit your work as a PDF file via Blackboard. Please provide "Screenshots" of Wireshark if needed.

Due: 1:55pm Thursday, September 4 2025

Introduction

The first part of the lab introduces packet sniffer, Wireshark. Wireshark is a free open-source network protocol analyzer. It is used for network troubleshooting and communication protocol analysis. Wireshark captures network packets in real time and display them in human-readable format. It provides many advanced features including live capture and offline analysis, three-pane packet browser, coloring rules for analysis. This document uses Wireshark for the experiments, and it covers Wireshark installation, packet capturing, and protocol analysis.


The Wireshark Network Analyzer
Apply a display filter ... <⌘/>
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
1 interface shown, 15 hidden
Wi-Fi: en0
Learn
User's Guide · Wiki · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Donate
Ready to load or capture
No Packets
Profile: Default

You are running Wireshark 4.2.7 (v4.2.7-0-g99a85ec073f0). You receive automatic updates.

Figure 1: Wireshark Application

Background

TCP/IP Network Stack

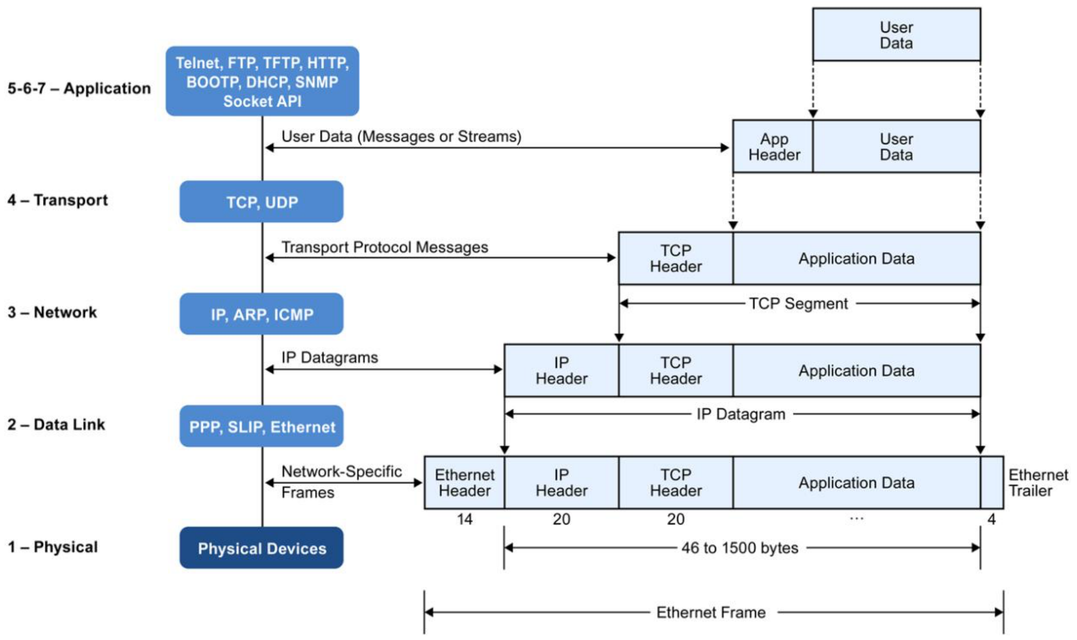


Figure 2: Encapsulation of Data in the TCP/IP Network Stack

TCP/IP network stack is the most commonly used network model for Internet services. Because its most important protocols, the Transmission Control Protocol (TCP) and the Internet Protocol (IP) were the first networking protocols defined in this standard, it is named as TCP/IP. However, it contains multiple layers including application layer, transport layer, network layer, and data link layer.

- *Application Layer*: The application layer includes the protocols used by most applications for providing user services. Examples of application layer protocols are Hypertext Transfer Protocol (HTTP), Secure Shell (SSH), File Transfer Protocol (FTP), and Simple Mail Transfer Protocol (SMTP).

- *Transport Layer*: The transport layer establishes process-to-process connectivity, and it provides end-to-end services that are independent of underlying user data. To implement the process-to-process communication, the protocol introduces a concept of port. The examples of transport layer protocols are Transport Control Protocol (TCP) and User Datagram Protocol (UDP). The TCP provides flow-control, connection establishment, and reliable transmission of data, while the UDP is a connectionless transmission model.

- *Internet Layer*: The Internet layer is responsible for sending packets to across networks. It has two functions: 1) Host identification by using IP addressing system (IPv4 and IPv6); and 2) packets routing from source to destination. The examples of Internet layer protocols are Internet Protocol (IP), Internet Control Message Protocol (ICMP), and Address Resolution Protocol (ARP).

- *Link Layer*: The link layer defines the networking methods within the scope of the local network link. It is used to move the packets between two hosts on the same link. An common example of link layer protocols is Ethernet.

Packet Sniffer

Packet sniffer is a basic tool for observing network packet exchanges in a computer. As the name suggests, a packet sniffer captures ("sniffs") packets being sent/received from/by your computer; it will also typically store and/or display the contents of the various protocol fields in these captured packets. A packet sniffer itself is passive. It observes messages being sent and received by applications and protocols running on your computer, but never sends packets itself.

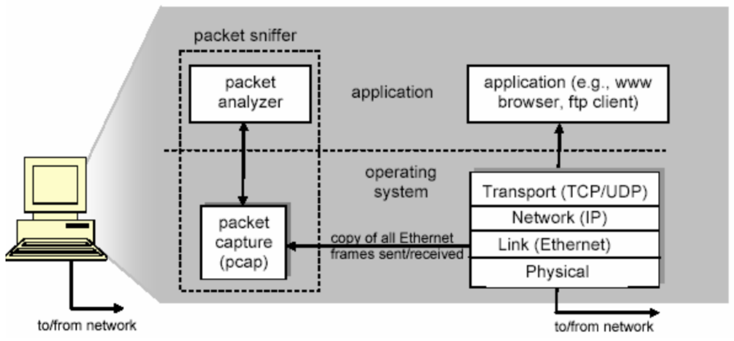


Figure 3 shows the structure of a packet sniffer.

At the right of **Figure** 3 are the protocols (in this case, Internet protocols) and applications (such as a web browser or ftp client) that normally run on your computer. The packet sniffer, shown within the dashed rectangle in **Figure** 3 is an addition to the usual software in your computer, and consists of two parts. The packet capture library receives a copy of every link-layer frame that is sent from or received by your computer. Messages exchanged by higher layer protocols such as HTTP, FTP, TCP, UDP, DNS, or IP all are eventually encapsulated in link-layer frames that are transmitted over physical media such as an Ethernet cable. In Figure 1, the assumed physical media is an Ethernet, and so all upper-layer protocols are eventually encapsulated within an Ethernet frame. Capturing all link-layer frames thus gives you access to all messages sent/received from/by all protocols and applications executing in your computer.

The second component of a packet sniffer is the packet analyzer, which displays the contents of all fields within a protocol message. In order to do so, the packet analyzer must “understand” the structure of all messages exchanged by protocols. For example, suppose we are interested in displaying the various fields in messages exchanged by the HTTP protocol in **Figure** 3. The packet analyzer understands the format of Ethernet frames, and so can identify the IP datagram within an Ethernet frame. It also understands the IP datagram format, so that it can extract the TCP segment within the IP datagram. Finally, it understands the TCP segment structure, so it can extract the HTTP message contained in the TCP segment. Finally, it understands the HTTP protocol and so, for example, knows that the first bytes of an HTTP message will contain the string “GET,” “POST,” or “HEAD”.

We will be using the Wireshark packet sniffer [http://www.wireshark.org/] for these labs, allowing us to display the contents of messages being sent/received from/by protocols at different levels of the protocol stack. (Technically speaking, Wireshark is a packet analyzer that uses a packet capture library in your computer). Wireshark is a free network protocol analyzer that runs on Windows, Linux/Unix, and Mac computers.

Installing Wireshark

To install Wireshark on Windows or macOS, you can follow these steps:

- Go to the Wireshark download page. (https://www.wireshark.org/)
- Click the release package for your operating system.
- The download will start automatically.
- Install the package.

If you're running another operating system, such as Linux, you might want to install from source.
The Wireshark installer includes WinPcap, so you don't need to download and install two separate packages. You can choose where to install the program and there are several optional components
You can find more information here: https://www.wireshark.org/docs/

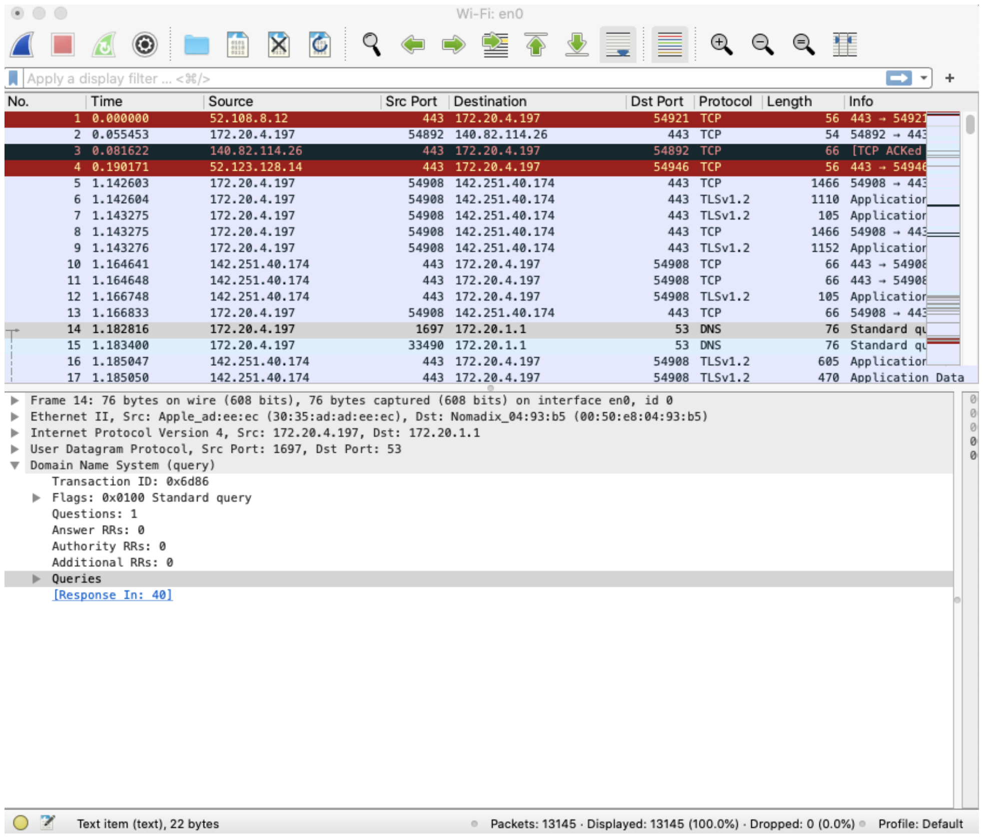
Capturing Packets

After downloading and installing Wireshark, you can launch it and click the name of an interface under Interface List to start capturing packets on that interface. For example, if you want to capture traffic on the wireless network, click your wireless interface.

Test Run

Do the following steps:

1. Start up the Wireshark program (select an interface and press start to capture packets).
2. Start up your favorite browser
3. In your browser, go to Syracuse University homepage by typing www.syracuse.edu.
4. After your browser has displayed the http://www.syracuse.edu page, stop Wireshark packet capture by selecting stop in the Wireshark capture window. This will cause the Wireshark capture window to disappear and the main Wireshark window to display all packets captured since you began packet capture see image below:
5. Color Coding: You'll probably see packets highlighted in green, blue, and black. Wireshark uses colors to help you identify the types of traffic at a glance. By default, green is TCP traffic, dark blue is DNS traffic, light blue is UDP traffic, and



No.	Time	Source	Src Port	Destination	Dst Port	Protocol	Length	Info
1	0.000000	52.108.8.12	443	172.20.4.197	54921	TCP	56	443 → 54921
2	0.055453	172.20.4.197	54892	140.82.114.26	443	TCP	54	54892 → 443
3	0.081622	140.82.114.26	443	172.20.4.197	54892	TCP	66	[TCP ACKed
4	0.190171	52.123.128.14	443	172.20.4.197	54946	TCP	56	443 → 54946
5	1.142603	172.20.4.197	54908	142.251.40.174	443	TCP	1466	54908 → 443
6	1.142604	172.20.4.197	54908	142.251.40.174	443	TLSv1.2	1110	Applicatior
7	1.143275	172.20.4.197	54908	142.251.40.174	443	TLSv1.2	105	Applicatior
8	1.143275	172.20.4.197	54908	142.251.40.174	443	TCP	1466	54908 → 443
9	1.143276	172.20.4.197	54908	142.251.40.174	443	TLSv1.2	1152	Applicatior
10	1.164641	142.251.40.174	443	172.20.4.197	54908	TCP	66	443 → 54908
11	1.164648	142.251.40.174	443	172.20.4.197	54908	TCP	66	443 → 54908
12	1.166748	142.251.40.174	443	172.20.4.197	54908	TLSv1.2	105	Applicatior
13	1.166833	172.20.4.197	54908	142.251.40.174	443	TCP	66	54908 → 443
14	1.182816	172.20.4.197	1697	172.20.1.1	53	DNS	76	Standard qu
15	1.183400	172.20.4.197	33490	172.20.1.1	53	DNS	76	Standard qu
16	1.185047	142.251.40.174	443	172.20.4.197	54908	TLSv1.2	605	Applicatior
17	1.185050	142.251.40.174	443	172.20.4.197	54908	TLSv1.2	470	Application Data

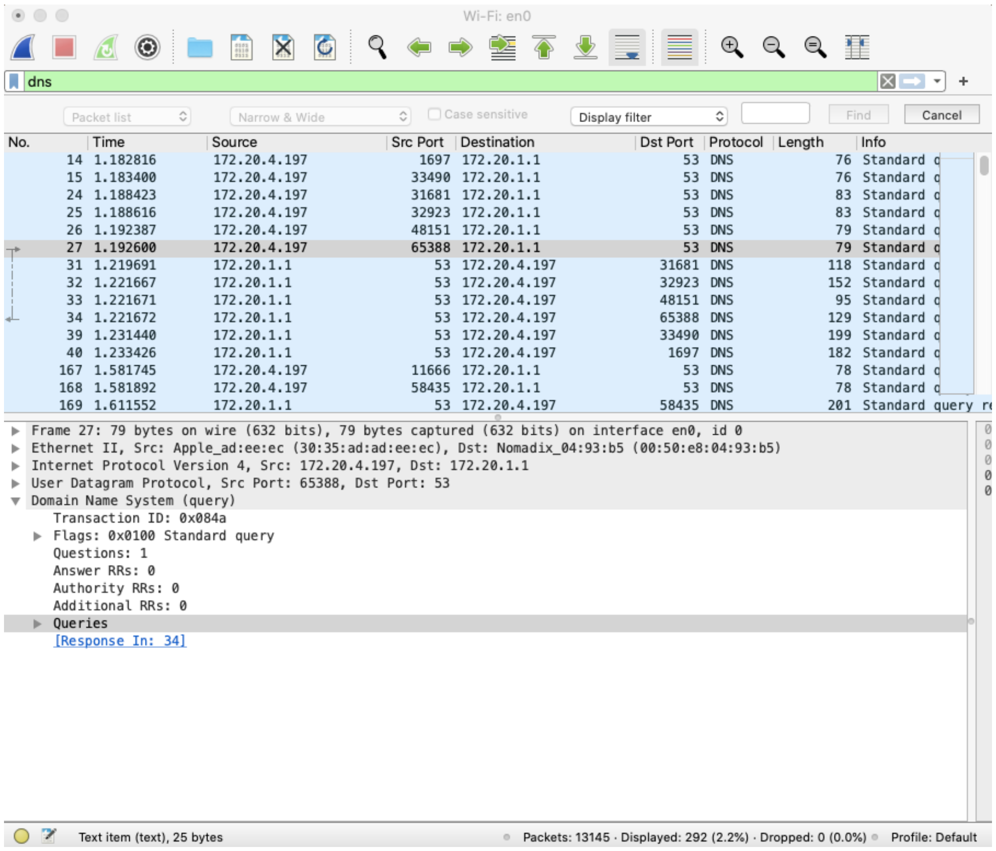
```
▶ Frame 14: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface en0, id 0
▶ Ethernet II, Src: Apple_ad:ee:ec (30:35:ad:ad:ee:ec), Dst: Nomadix_04:93:b5 (00:50:e8:04:93:b5)
▶ Internet Protocol Version 4, Src: 172.20.4.197, Dst: 172.20.1.1
▶ User Datagram Protocol, Src Port: 1697, Dst Port: 53
▼ Domain Name System (query)
    Transaction ID: 0x6d86
  ▶ Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ▶ Queries
    [Response In: 40]
```

Text item (text), 22 bytes   Packets: 13145 · Displayed: 13145 (100.0%) · Dropped: 0 (0.0%)   Profile: Default black identifies TCP packets with problems — for example, they could have been delivered out-of-order.

6. You now have live packet data that contains all protocol messages exchanged between your computer and other network entities! However, as you will notice the HTTP messages are not clearly shown because there are many other packets included in the packet capture. Even though the only action you took was to open your browser, there are many other programs in your computer that communicate via the network in the background. To filter the connections to the ones we want to focus on, we have to use the filtering functionality of Wireshark by typing "http" in the filtering field as shown below:

7. To further filter packets in Wireshark, we need to use a more precise filter. By setting the http.host==Syracuse.com, we are restricting the view to packets that have as an http host the www.syracuse.edu website. Notice that we need two equal signs to perform the match "==" not just one.

8. Now, we can try another protocol. Let's use Domain Name System (DNS) protocol as an example here.

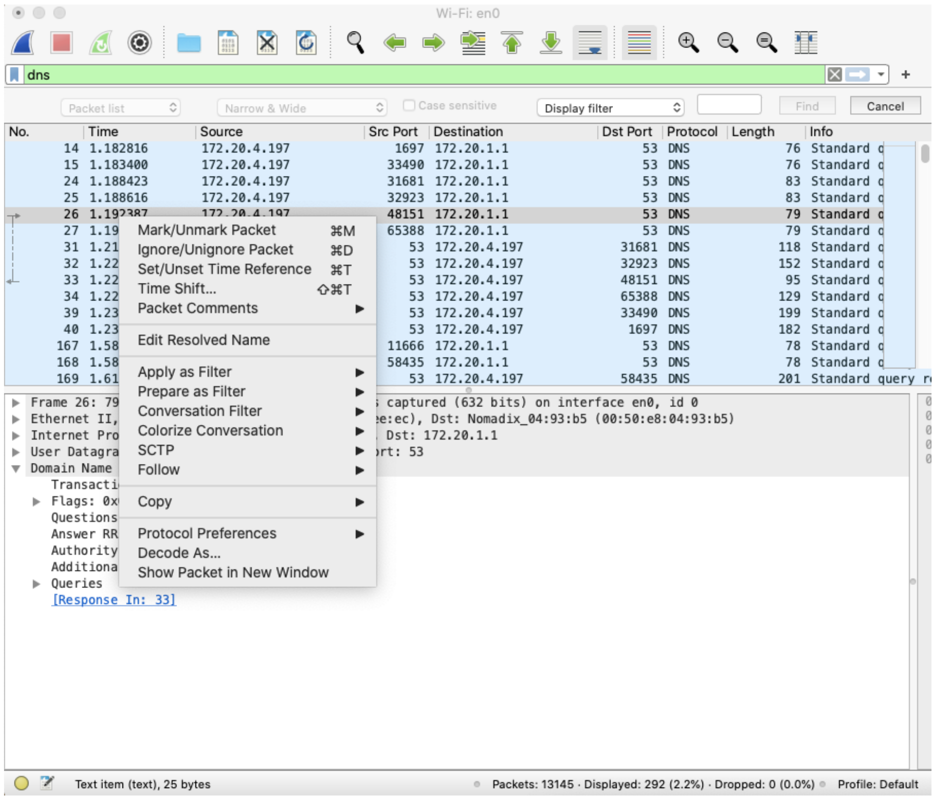


```
No.   Time      Source        Src Port Destination  Dst Port Protocol Length Info
 14 1.182816  172.20.4.197     1697  172.20.1.1       53  DNS       76 Standard q
 15 1.183400  172.20.4.197    33490  172.20.1.1       53  DNS       76 Standard q
 24 1.188423  172.20.4.197    31681  172.20.1.1       53  DNS       83 Standard q
 25 1.188616  172.20.4.197    32923  172.20.1.1       53  DNS       83 Standard q
 26 1.192387  172.20.4.197    48151  172.20.1.1       53  DNS       79 Standard q
 27 1.192600  172.20.4.197    65388  172.20.1.1       53  DNS       79 Standard q
 31 1.219691  172.20.1.1         53  172.20.4.197  31681  DNS      118 Standard q
 32 1.221667  172.20.1.1         53  172.20.4.197  32923  DNS      152 Standard q
 33 1.221671  172.20.1.1         53  172.20.4.197  48151  DNS       95 Standard q
 34 1.221672  172.20.1.1         53  172.20.4.197  65388  DNS      129 Standard q
 39 1.231440  172.20.1.1         53  172.20.4.197  33490  DNS      199 Standard q
 40 1.233426  172.20.1.1         53  172.20.4.197   1697  DNS      182 Standard q
167 1.581745  172.20.4.197    11666  172.20.1.1       53  DNS       78 Standard q
168 1.581892  172.20.4.197    58435  172.20.1.1       53  DNS       78 Standard q
169 1.611552  172.20.1.1         53  172.20.4.197  58435  DNS      201 Standard query re
```

```
▶ Frame 27: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface en0, id 0
▶ Ethernet II, Src: Apple_ad:ee:ec (30:35:ad:ad:ee:ec), Dst: Nomadix_04:93:b5 (00:50:e8:04:93:b5)
▶ Internet Protocol Version 4, Src: 172.20.4.197, Dst: 172.20.1.1
▶ User Datagram Protocol, Src Port: 65388, Dst Port: 53
▼ Domain Name System (query)
    Transaction ID: 0x084a
  ▶ Flags: 0x0100 Standard query
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ▶ Queries
    [Response In: 34]
```

```
Text item (text), 25 bytes        Packets: 13145 · Displayed: 292 (2.2%) · Dropped: 0 (0.0%)    Profile: Default
```

9. Let's try now to find out what are those packets contain by following one of the conversations (also called network flows), select one of the packets and press the right mouse button (if you are on a Mac use the command button and click), you should see something similar to the screen below:



```
No.    Time      Source          Src Port Destination    Dst Port Protocol Length Info
   14 1.182816  172.20.4.197        1697 172.20.1.1           53 DNS         76 Standard q
   15 1.183400  172.20.4.197       33490 172.20.1.1           53 DNS         76 Standard q
   24 1.188423  172.20.4.197       31681 172.20.1.1           53 DNS         83 Standard q
   25 1.188616  172.20.4.197       32923 172.20.1.1           53 DNS         83 Standard q
   26 1.192387  172.20.4.197       48151 172.20.1.1           53 DNS         79 Standard q
   27 1.19                         65388 172.20.1.1           53 DNS         79 Standard q
   31 1.21                            53 172.20.4.197      31681 DNS        118 Standard q
   32 1.22                            53 172.20.4.197      32923 DNS        152 Standard q
   33 1.22                            53 172.20.4.197      48151 DNS         95 Standard q
   34 1.22                            53 172.20.4.197      65388 DNS        129 Standard q
   39 1.23                            53 172.20.4.197      33490 DNS        199 Standard q
   40 1.23                            53 172.20.4.197       1697 DNS        182 Standard q
  167 1.58                         11666 172.20.1.1           53 DNS         78 Standard q
  168 1.58                         58435 172.20.1.1           53 DNS         78 Standard q
  169 1.61                            53 172.20.4.197      58435 DNS        201 Standard query r
```

```
▶ Frame 26: 79                  s captured (632 bits) on interface en0, id 0
▶ Ethernet II,                  ee:ec), Dst: Nomadix_04:93:b5 (00:50:e8:04:93:b5)
▶ Internet Pro                  . Dst: 172.20.1.1
▶ User Datagra                  ort: 53
▼ Domain Name
    Transacti
  ▶ Flags: 0x
    Questions
    Answer RR
    Authority
    Additiona
  ▶ Queries
    [Response In: 33]
```

Text item (text), 25 bytes   Packets: 13145 · Displayed: 292 (2.2%) · Dropped: 0 (0.0%)   Profile: Default

What is packet analysis and how to capture network traffic?

Packet capture is the process of intercepting data packets that travel over a network. Once captured, the packets can be stored for analysis.

Packets contain metadata such as the source and destination addresses, the protocol, the payload, and other metadata. Packet captures can provide a complete roadmap of the traffic a network experiences. They can also help identify traffic bottlenecks.

In this exercise we will analyze some previously captured traffic and explain the contents of the data in detail.

Exercise 1:
Open the sample_set1.pcap File in Wireshark (Check the Blackboard for sample_set1.pcap file)

To determine what type of data has been captured in this file we can to go the *Statistics* section and select *Protocol Hierarchy*

Q1: How many packets are there?
Q2: What networking protocol is used?
Q3: What is the source IP address?
Q4: What is the destination IP address?
Q5: What port number is the source using to communicate with the destination (or what port number is the destination listening on)?
Q6: Do you notice the "three-way handshake"?

Exercise 2: Reconstructing a conversation

- Click on a packet (it will be highlighted in blue)
- Right-click on packet
- Go to "Follow"
- Follow one of the following streams depending on protocol (UDP Stream is most common)

Exercise 3: Finding transmission protocols

- To see all protocols in a Wireshark capture, you can go to Statistics > Protocol Hierarchy. This will show a list of all protocols, along with the number of packets and bytes for each protocol.
- You can also examine capture file properties by going to Statistics > Capture File Properties. This will show general information about the PCAP file, including the first and last packets, timestamps, and the total number of packets.
- To filter to a particular stream, you can select a packet in the packet list and then select the menu item Analyze → Follow → TCP Stream.
- You can also find packets by selecting Edit → Find Packet in the main menu.

Open the sample_set2.pcap File in Wireshark (Check the Blackboard for sample_set2.pcap file)

Q1: What insecure protocol was used to transmit pictures on network?
Q2: How many pictures were transmitted?

Q3: Extract one of the pictures that was transmitted. HINT: show and save the picture as “Raw” format.

Exercise 4: Try to extract the Username:Password Pairs

Go to Tools → Credentials

Open the sample_set3.pcap File in Wireshark (Check the Blackboard for sample_set3.pcap file)

Q1: What protocol was used to transmit the username:password pair (credentials)?
Q2: What is one username:password pair in this PCAP set? (HINT: use Edit > Find Packet)
Q3: Is the username:password pair valid? Why / why not?