Lab 1: Wireshark

Introduction

The purpose of today's lab is to solidify your background in the 'nuts & bolts' of Internet technologies. It will also give some empirical experience in 'peeling back layers' of the Internet.

This should hopefully be a fun bit of poking around with what your computer is actually doing all the time — for better or worse, I always find something new every time I look at the firehose of packets coming in and out of my machine.

Goals

- Set up Wireshark on your system
- Understand how to use Wireshark to inspect communication
- Explore communication on your computer

Equipment

- Computer

Partners

- This lab should be done individually

Submission

- Write your answers up and submit a PDF to Gradescope.

  Remember: I'm not looking for a formal lab report. Just your answers in any format that makes sense. The goal is to prove that you did the lab and spent some time thinking about it.

1. Install Wireshark

Wireshark is available here: https://www.wireshark.org/

Wireshark works on Windows, MacOS, and Linux. You can install it right on your host or inside a virtual machine if you prefer. Virtual machines will take a little extra care to make sure they can access network interfaces on your computer.

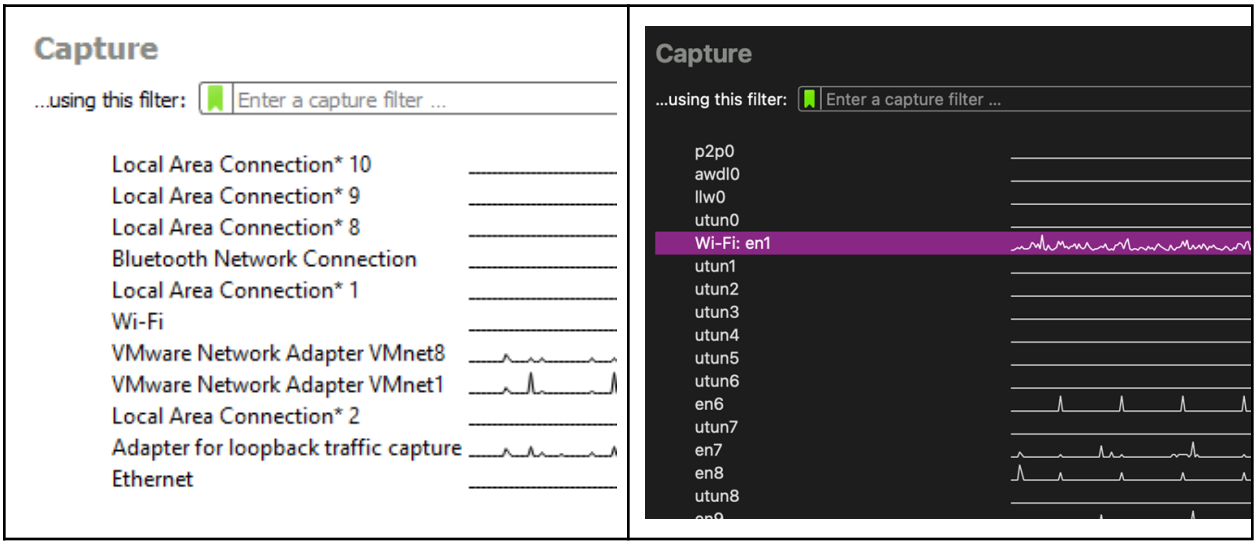
Sometimes, you can run into some permission headaches getting wireshark access to your network traffic. The modern installers are pretty good at getting all the permissions it needs, but if you have issues, Google is the best place to go.

Note: It is not a good idea to run Wireshark as root/administrator — it'll get all the packets, sure, but that's really opening yourself up for trouble. See more details here: https://superuser.com/questions/139206/concern-over-running-wireshark-as-root

TASK: None. Continue to the next section.

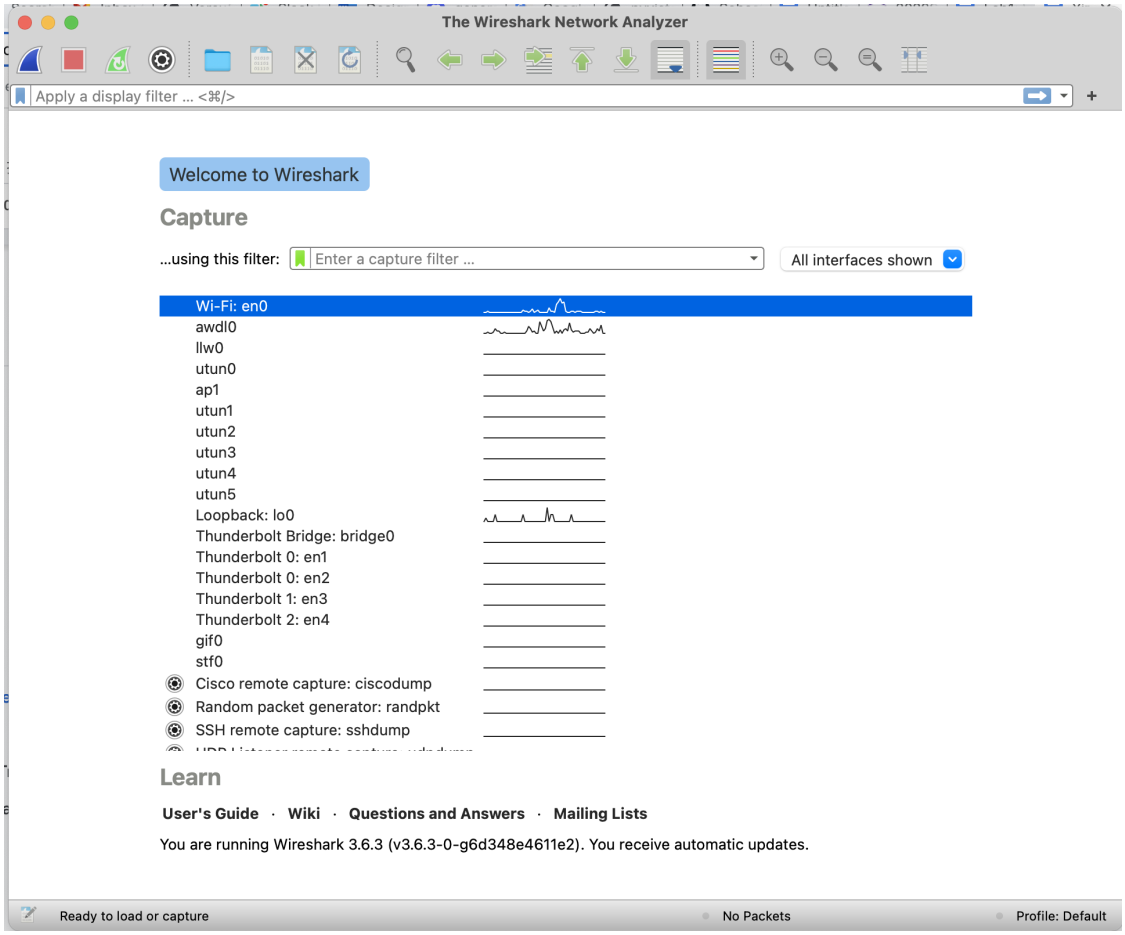
2. Understanding Interfaces

When you start Wireshark, it provides a list of interfaces that can be used to capture packets. Depending on your OS, you might get rather cryptic names



TASK: Explain in English what physical or digital thing each of the interfaces on your machine corresponds to (e.g., “en1 is my WiFi card”). Group interfaces as appropriate.

- Include a screenshot of the interfaces that Wireshark lists.



- If you're not sure about what an interface is, look around on Google for a bit. If you're still not sure, answer "Don't know". Don't spend too long stuck on any one interface.

3. Wireshark Practice

Especially if you've never used Wireshark before, Jim Kurose (he wrote the Networks textbook) has some excellent labs that can help you understand and practice with it. I strongly recommend you walk through these. It'll only take like 20 minutes to do so and they will teach you a lot about how Wireshark works.

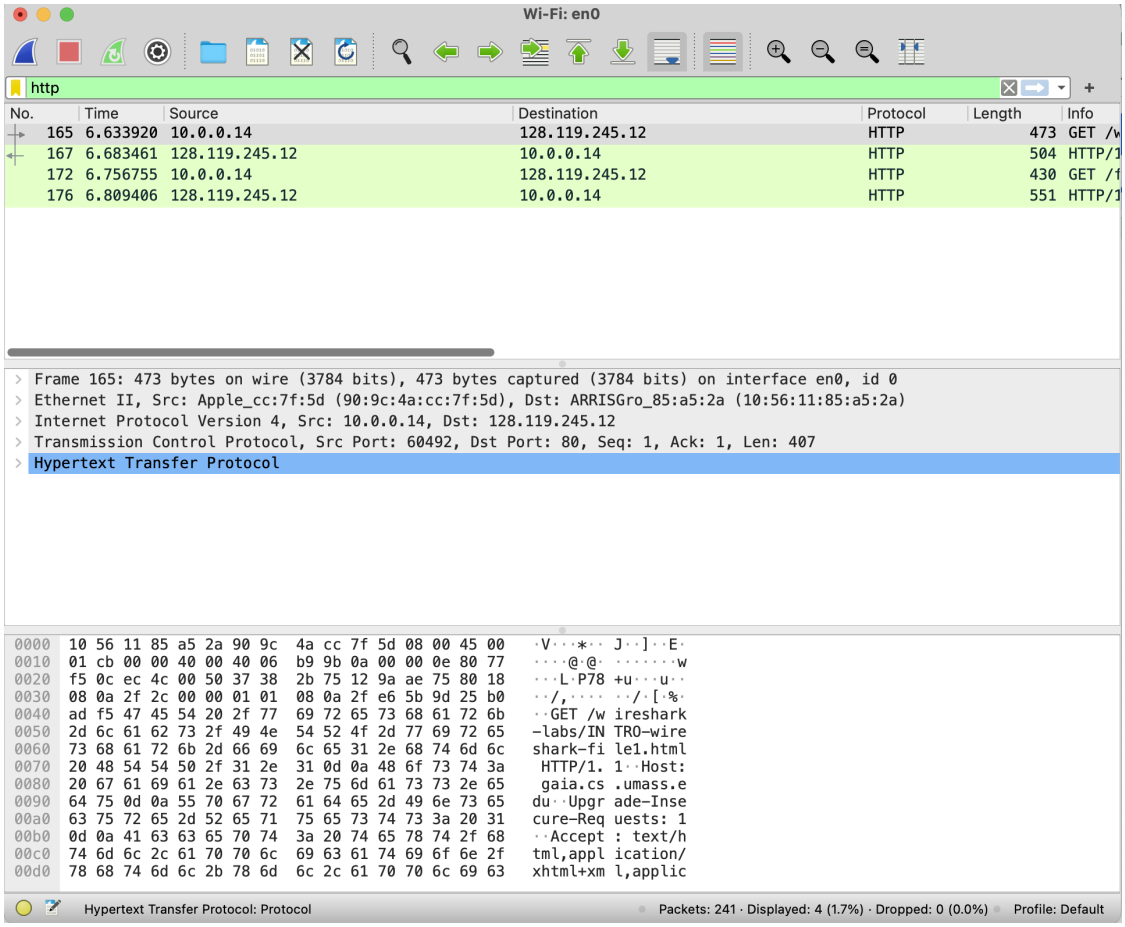
- Getting Started Lab:
  http://www-net.cs.umass.edu/wireshark-labs/Wireshark_Intro_v8.0.pdf

- DNS Query Lab:
  http://www-net.cs.umass.edu/wireshark-labs/Wireshark_DNS_v8.0.pdf

There are various other labs also available:
https://gaia.cs.umass.edu/kurose_ross/wireshark.php

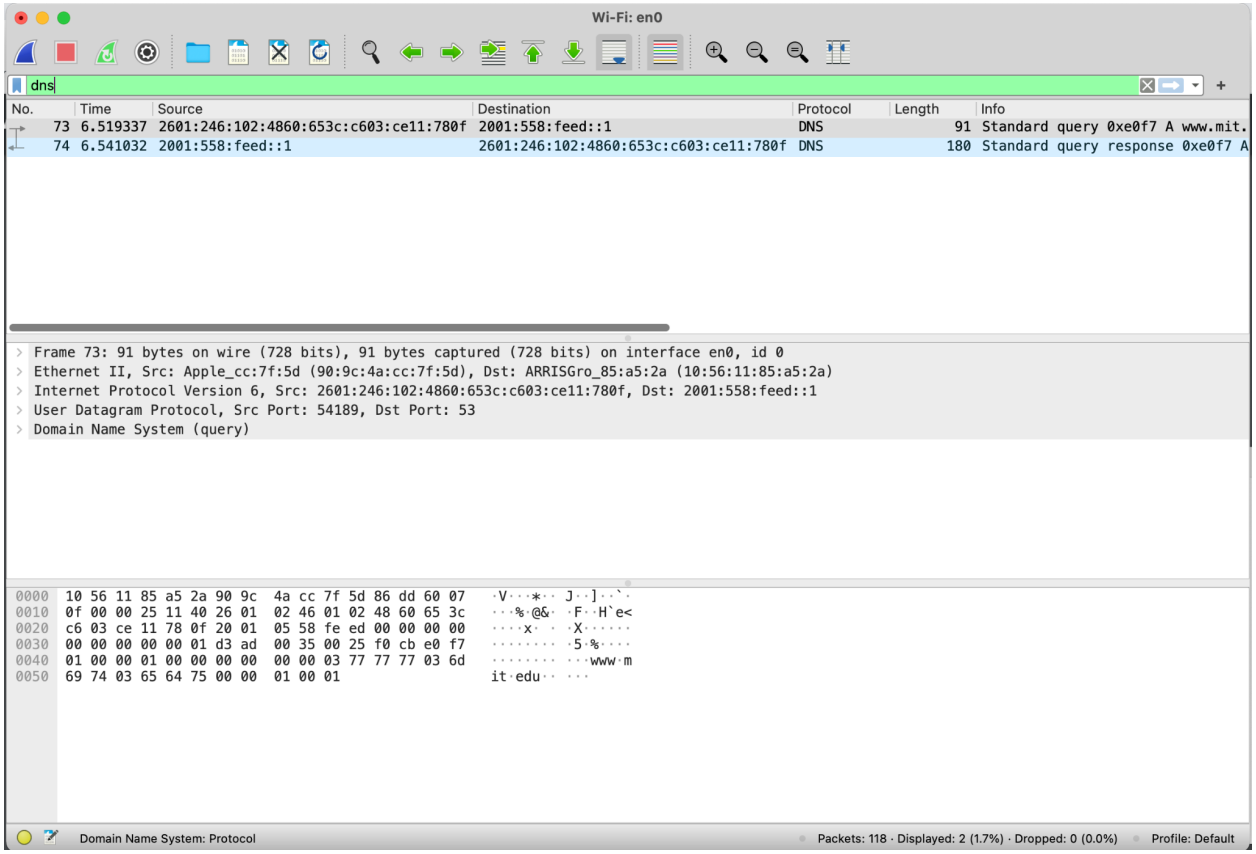
TASK: Demonstrate capturing an HTTP request/response in Wireshark.

- A screenshot makes a lot of sense here.



TASK: Demonstrate capturing a DNS request/response in Wireshark.

- A screenshot makes a lot of sense here.

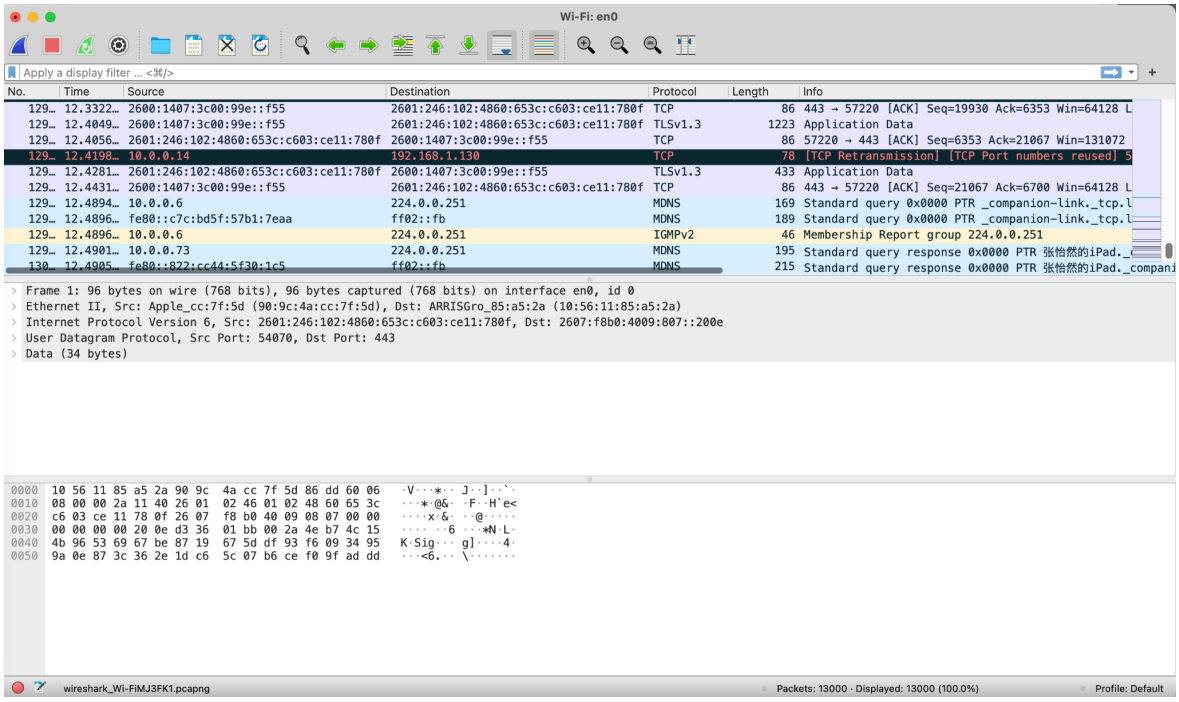


4. Investigate Intentional Traffic

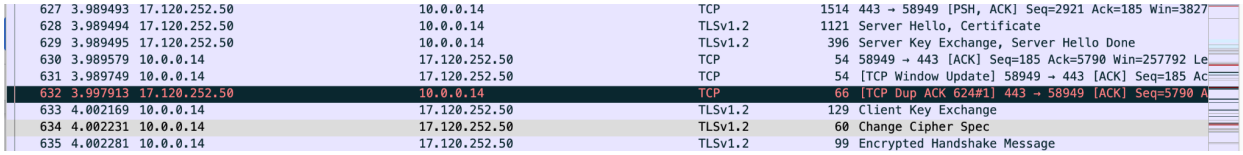
Use some application that you know will communicate over the internet and use Wireshark to find that communication. Pick something other than just visiting a website. You could play a Youtube video, play new songs on Spotify, start a video game, use Zoom, etc. Closing other non-essential applications on your computer can be helpful here to narrow down the possibilities.

TASK: Document your investigation.

- What does an example packet look like?
  Here I tried to open my apple music and play a song.



- What method did you use to find traffic from that application?
  I first closed out all the websites and applications running on the Internet. After several times of observation, I found that in a DNS info query on init.itunes.apple.com. Thus I know it begins to track after this message occurs.
- Did you notice anything interesting about the traffic?
  Everytime anything happens to the client or server, there is always “Change Cipher Spec” and “Encrypted Handshake Message” after it.

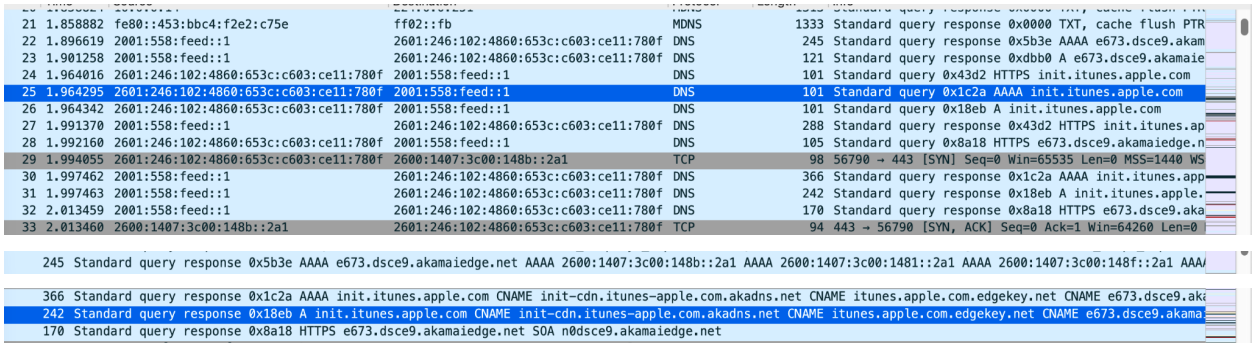


5. Investigate Unknown Traffic

Pick some traffic that looks interesting to you and investigate what communication is occurring. Maybe an interesting Protocol or to a Source/Destination you don't immediately recognize. Can you determine what the purpose of the communication was?

TASK: Document your investigation.

- What does your unknown packet look like?



21	1.858882	fe80::453:bbc4:f2e2:c75e	ff02::fb	MDNS	1333	Standard query response 0x0000 TXT, cache flush PTR
22	1.896619	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	245	Standard query response 0x5b3e AAAA e673.dsce9.akam
23	1.901258	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	121	Standard query response 0xdbb0 A e673.dsce9.akamaie
24	1.964016	2601:246:102:4860:653c:c603:ce11:780f	2001:558:feed::1	DNS	101	Standard query 0x43d2 HTTPS init.itunes.apple.com
25	1.964295	2601:246:102:4860:653c:c603:ce11:780f	2001:558:feed::1	DNS	101	Standard query 0x1c2a AAAA init.itunes.apple.com
26	1.964342	2601:246:102:4860:653c:c603:ce11:780f	2001:558:feed::1	DNS	101	Standard query 0x18eb A init.itunes.apple.com
27	1.991370	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	288	Standard query response 0x43d2 HTTPS init.itunes.ap
28	1.992160	2601:246:102:4860:653c:c603:ce11:780f	2001:558:feed::1	DNS	105	Standard query 0x8a18 HTTPS e673.dsce9.akamaiedge.n
29	1.994055	2601:246:102:4860:653c:c603:ce11:780f	2600:1407:3c00:148b::2a1	TCP	98	56790 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=1440 WS
30	1.997462	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	366	Standard query response 0x1c2a AAAA init.itunes.app
31	1.997463	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	242	Standard query response 0x18eb A init.itunes.apple.
32	2.013459	2001:558:feed::1	2601:246:102:4860:653c:c603:ce11:780f	DNS	170	Standard query response 0x8a18 HTTPS e673.dsce9.aka
33	2.013460	2600:1407:3c00:148b::2a1	2601:246:102:4860:653c:c603:ce11:780f	TCP	94	443 → 56790 [SYN, ACK] Seq=0 Ack=1 Win=64260 Len=0

```
245 Standard query response 0x5b3e AAAA e673.dsce9.akamaiedge.net AAAA 2600:1407:3c00:148b::2a1 AAAA 2600:1407:3c00:1481::2a1 AAAA 2600:1407:3c00:148f::2a1 AAA
```

```
366 Standard query response 0x1c2a AAAA init.itunes.apple.com CNAME init-cdn.itunes-apple.com.akadns.net CNAME itunes.apple.com.edgekey.net CNAME e673.dsce9.aka
242 Standard query response 0x18eb A init.itunes.apple.com CNAME init-cdn.itunes-apple.com.akadns.net CNAME itunes.apple.com.edgekey.net CNAME e673.dsce9.akama
170 Standard query response 0x8a18 HTTPS e673.dsce9.akamaiedge.net SOA n0dsce9.akamaiedge.net
```

- How did you determine what the traffic corresponded to?
  Here is what the traffic will be like when I try to open my apple music. I had several DNS queries and responses, TCP infos to enable application programs to "talk" to the information in the network, and TLSV to process the client, server, and application data.
- Did you notice anything interesting about the traffic?
  The second figure in the previous question corresponds to the column 22 of the first figure. In the DNS protocols, we can see how standard query and standard query response are like and how they were mapped to the specific website. Also, columns 25 26 28 are all queries that are "answered" by column 30-32. The detailed info is shown in the third figure above. Then after column 32, the TCP just directly maps its sources to 2600:1406:2c00:148b::2a1 as it first referred to column 22's info.