Lab 1 Manual

# Introduction

The purpose of this 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.

**Warning:** In general, be careful when sniffing traffic. It can be illegal to monitor communications you were not supposed to have access to.

**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 for each task and submit a PDF to [Gradescope](https://www.gradescope.com/).
  + We’re not looking for the most formal lab report. Just your answers in a reasonable format that makes sense. The primary goal is to prove that you did the lab and spent some time thinking about it.
* We’ve provided template [Pre-Lab Worksheet](https://docs.google.com/document/u/0/d/131d3044xhuMnxCHAGDCydKZRNxqBiAGtrZgFsgOrQfs/edit) and [Post-Lab Worksheet](https://docs.google.com/document/u/0/d/1kdMUrhTRlDCOyKIFYha9V_h8SQTCMmFJUeYiBa5rB1Y/edit) documents with just the questions pulled out for your submissions.

# Pre-Lab

## A.) 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.

## B.) 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 (or table) 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, look at some different ones. Don’t spend too long stuck on any one interface.

# 

# Lab

## C.) 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.
* **Note:** some students have found HTTP packets were disabled by default. They had to go to Analyze->Enabled Protocols and then enable HTTP.
* **Note:** some students have found that HTTP packets don’t show up at all (even after enabling HTTP packets). If you’re using HTTPS in your browser, Wireshark doesn’t label HTTPS traffic with the HTTP protocol. Instead it will show up under TCP or TLS protocol (you can filter with “tcp” or “ssl”). To see HTTP packets you can send a request using plain, insecure HTTP. You can go to something like <http://neverssl.com/>. Just make sure that the URL has “http://” at the start. Alternatively, you can send a regular HTTP request using the “curl” terminal command. If that doesn’t work and you are using the Chrome browser, you might want to disable QUIC packets (in Analyze->Enabled Protocols).

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

* A screenshot makes a lot of sense here.
* **Note:** Firefox sends DNS traffic over HTTPS requests ([DoH](https://en.wikipedia.org/wiki/DNS_over_HTTPS)) by default, so you’ll have to use a different browser (or a terminal) to make the request if you want to see DNS traffic.

## D.) Inspect Ping Traffic

1. Open a terminal window (this works in Command Prompt on Windows too) and run:

ping 4.2.2.2

1. Open wireshark and start collecting traffic on your default interface.
2. Add a filter:  
     
   ip.addr==4.2.2.2
3. You should see something like this:

# 

1. Explore a bit in Wireshark and see what you can learn about the packets you’re observing.

**TASK:** Understand the Ping traffic.

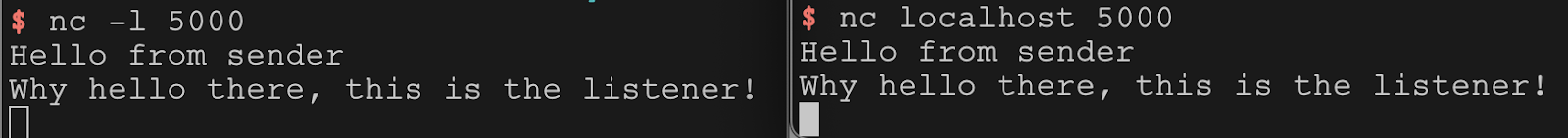
* What does ICMP stand for?
* For one of your *ping* packets, start from the PHY and list each of the layers that were used to send the packet, and which technology was used. A screenshot from wireshark would be useful here as evidence. (You might notice one of your layers being labeled as “Ethernet II” when you’re not using Ethernet at all! You can read more [here](https://superuser.com/questions/1242454/why-do-i-see-ethernet-ii-protocol-in-wireshark-in-wireless-connection).)

# 

## E.) Investigate Intentional Traffic

### Insecure Chat

For this section, we will use the [netcat (nc) utility](https://en.wikipedia.org/wiki/Netcat). If using MacOS or Linux, this is probably already built-in to your machine and can be run from a terminal window. If using Windows, you can get the ncat tool by installing the [nmap package](https://nmap.org/download.html#windows). To run ncat you will need to run Command Prompt as administrator.

1. ~~Ensure you are connected to eduroam or UCSD-Protected.~~ **Join “tock\_tutorial” WiFi network and use the “Discovery Board” on the whiteboard.**
2. Set up a “listen” (nc -l [PORT]) or (ncat -l [PORT]) endpoint. This tells your machine to listen for incoming connections on a specific port. Ports are 16 bits, and the lower numbers are typically reserved. Choose your preferred port number.  
     
   **NOTE**: If you are on Windows, you will need to disable the windows firewall for public networks to receive incoming connections.
3. List your IP address and port on our class [discovery board](https://docs.google.com/document/d/17S4bPpzTBLXqS7YJTxuWoU0V9ry0EVMf0aTu1jFOOkU/edit?tab=t.0#heading=h.t9t8qnayib32).
4. Open a new terminal, choose another IP address from the board, and try to connect (nc [IP] [PORT]). You may want to instruct netcat to use a timeout so that it will fail if the listener already has a connection. On my machine that looks like (nc [IP] [PORT] -w 500) or (ncat [IP] [PORT] -w 500ms). You may need to view the options in case the flag is different on your machine.
5. You should be able to chat! Introduce yourself, and find your virtual connection in real life. With whom did you connect?  
    *What is "localhost"? Why can't you use that?*
6. Start a wireshark capture.
7. After sending a few messages back and forth, stop and save your wireshark capture.
8. Add a filter so that you are only looking at traffic from your chat session. (*Hint: what unique number did you control when you set this chat up?*)
9. Which machines can you see your chat traffic in Wireshark? Why? Can you see other chat traffic?
10. You might want to remove your IP address from the discovery board when you are finished with this section.

## F.) Discover WiFi Networks Around You

Computers discover WiFi routers using a variety of *probe packets*. These probe packets contain information such as the name of the WiFi network (SSID), signal strength information, security capabilities, etc.

1. You will need a trace of raw WiFi packets. If you happen to have a Mac, you should be able to find a tool called Wireless Diagnostics. Once you open the tool go to Window -> Sniffer. Here you should see an option to set the Channel and Channel Width. Choose Channel as 5, leave width unchanged and press Start. The tool will run a wireless capture until you press Stop. The capture output pcap file can be found in /var/tmp/. This pcap file can be directly opened in Wireshark and analyzed.  
     
   On a Mac, you can open the pcap with File -> Open in wireshark, then choosing “Computer”, and the right clicking in the file window to select “Show hidden files”. The the pcap is located at /var/tmp/\*\*\*\*\*.pcap.  
     
   If you don’t have a Mac, you can use the first pcap file that we recorded, available in the Canvas Files tab.
2. We are running a test network in lab. *Without connecting to it*, can you find it?
3. If you are capturing, let this run for about thirty seconds, then stop and save the capture.
4. What are the types of probe packets you can see from our test network?
5. Filter for only packets from our test network. What filter did you use?
6. What is the MAC address of the router for the test network?

Once you are finished, submit your report via Gradescope.