Computer Engineering 4DN4
Laboratory 1
Network Scanning and Packet Sniffing

Tcpdump, Windump, WireShark, Ncat/Telnet, and Nmap are free open source tools that are used for network analysis, troubleshooting and protocol/application development. In addition to network scanning, data can be captured and displayed "live" from active network interfaces as they interact with external devices. WireShark, in particular, includes a wide variety of features including display/capture filters, and has built-in decoding for most standard networking protocols. In this lab, we introduce these tools and use them to do some network scanning and protocol tracing. You must first install them on your own PC or laptop and then use them to do the experiments. The final experiments require that you have Python 3 installed.

1 Preparation

1.1 Tcpdump, Windump, Telnet/Ncat, Nmap and WireShark

1. It is important that you view the lectures that discuss tcpdump, windump, Ncat, Nmap and WireShark. Directions for installing and using tcpdump (or windump), Ncat and Nmap were discussed in class and are available in the class notes.
2. In Windows, an issue that may arise is with conflicting network capture library versions and the order in which the installations occur may be important. Please see the file `compeng4dn4_Lab_1_Windows_Installation_v01.pdf` for some advice.
3. See the Appendix (below) for a reprint of the WireShark installation and introduction notes.
4. There is plenty of online documentation for Tcpdump/Windump, Nmap and WireShark including cheat sheets and sites that show examples.

2 Experiments

In all of the experiments below, describe the protocol interactions (including packet details) that have occurred. With these descriptions, include text output of the packet capture to illustrate your descriptions. These can be obtained from WireShark as follows.

After a packet capture has been obtained, use the

`File > Export Packet Dissections > as Plain Text file`

to save a text file containing your packet capture. If you find that this File option is "greyed out",

use the `File > Save` function to save the capture in pcap format first. Then you should be able to export a text file.

TCP: Use WireShark to capture a packet trace when downloading an image file from the Internet.

(Note: When repeating this experiment with the same image file, your web browser may cache the file locally rather than downloading it repeatedly. To prevent this, it is best to use your browser in *private browsing mode* (also called *privacy mode* or *incognito mode*). This should prevent local caching. An alternative is to go into your browser settings menu and clear its cache after each download. In Firefox, for example, it is found in `Preferences > Advanced > Web Content`.
You have to click on the "Clear Now" button.) Another option is to use a different file that has not been downloaded previously.

Proceed as follows:

1. Enter a Display Filter of
   `ip.addr == 99.236.34.223` or a Capture Filter of
   `host 99.236.34.223`
   i.e., the IP address of compeng4dn4.mooo.com. Then open a web browser and go to
   `http://compeng4dn4.mooo.com:50008/photos/`
   There is a collection of image files in the displayed directory.
2. Start the WireShark packet capture, then click on the filename of an image from inside your browser. Stop the packet capture when the image is finished downloading.

The WireShark capture will include the various TCP packets associated with the image download. Show and discuss the connection setup packets and some of the TCP sequence number interactions that have occurred. In your report, include an image of the downloaded picture that you used.

TCP: Do a packet capture using Tcpdump or Windump while using Ncat (or Nc) to connect to the Python echo server at port 50007 on `compeng4dn4.mooo.com`. Have the server echo your group member's names and MAC ID numbers[1]. Show and discuss the packet trace that you obtained including TCP connection setup and shutdown interactions.

DNS: Use WireShark to capture a DNS query. This probably happened in the previous experiment, but you can do it as follows.

1. Use the `nslookup` utility to do the DNS query. It is available on all OS platforms, i.e., start the packet capture, then type
   `nslookup <server>`
   in a terminal or command prompt (i.e., PowerShell) window, where `<server>` is the desired machine, e.g., `compeng4dn4.mooo.com`.

[1] This information will be logged by the echo server.

Note that the local DNS software may cache recent queries, and if the current request is already cached, there may be no network activity. To clear the cache, on Windows you can type `ipconfig /flushdns`. On Linux, there probably isn't any caching, depending on the Linux distribution and version. The latest Ubuntu distributions, for example, use `dnsmasq` but disable caching by default. On MacOS Yosemite and later, you can clear the cache using `sudo killall -HUP mDNSResponder`.

Collect data from the capture and discuss the results.

Traceroute: Use WireShark to capture a traceroute session. This can be easily done by starting WireShark then invoking traceroute in a (MacOS/Linux) shell window, i.e.,
`traceroute <hostname>`
or a Windows PowerShell window, i.e.,
`tracert <hostname>`.
Include data from part of the capture that illustrates how traceroute works. Discuss what has happened.

Nmap: Use Nmap to do some network scans as described below. You may want to use the nmap `-Pn` flag that will simplify the packet trace by preventing host ping discovery.

1. Do a standard TCP connection scan (i.e., -sT) of the server `compeng4dn4.mooo.com` over the port range: 50000-50009. Describe and explain what you find.
2. Do a TCP SYN scan (i.e., -sS) of the same server (`compeng4dn4.mooo.com`) over the port range: 50000-50009. Describe and explain what you find. Note that you may have to open a command window with admin/root permissions to use -sS.
3. Choose an Open port from the scans above and do a Tcpdump or Windump capture while scanning using both -sS and -sT scans for the two ports. Describe the packet interactions that are occurring.
4. Scan your own laptop or desktop using a standard nmap TCP connection scan. Explain what you found and discuss the purpose of any services that you found (You will probably have to google search their names.)
5. Scan TCP port 8000 on your laptop using a standard nmap TCP connection scan. Discuss what result you obtain.
6. On your laptop, launch the HTTP server that comes as a module with Python 3, i.e., open a terminal shell, and type:
   `python -m http.server`
   This will create a web server that is listening on port 8000 and serving up the contents of the directory where python was invoked. On Windows, a window may pop up asking you to authorize the server. Scan TCP port 8000, as before and discuss what result you obtain.

   Open a web browser and access this web service, i.e., if your IP address is 192.168.1.150, for example, then browse to:
   `http://192.168.1.150:8000`
   Describe what you see.

Warning: Python is serving up the contents of the directory where the server was launched, i.e., it can be read by anyone who can connect to that machine. Make sure that you exit the HTTP server when you are done!

3 Writeup

Submit a report for the lab on Avenue To Learn. Join a Lab 1 group on A2L (maximum of 3 people) and submit a single writeup for your group. Include in your writeup a brief description of everything that you did including an interpretation of the results that you obtained.

4 Appendix: WireShark Installation and Operation

1. To install WireShark, go to
   `https://WWW.wireShark.org/download.html`.

   Choose the stable release for your OS. There are versions for MacOS and Windows. For Linux, install it using your distribution's package manager.

2. Run the installer and confirm the installation. WireShark will install the packet capture (pcap) library.

3. Once the installer has finished, try running WireShark to make sure that the install finished properly. You should see a startup screen, something like that shown in Figure 1, depending on the version and your OS.

4. The WireShark startup screen contains links to their website and user guide:

   `https://www.wireSHARK.org`
   `https://www.wireshark.org/docs/wsug_html_chunked/`

5. Try some preliminary packet captures. Make sure that the proper network interface is selected from the start screen. Alternately, you can select it by pulling down the `Capture` menu. When a capture starts You should see some packets being captured but you may have to initiate some network activity, e.g., using a web browser. Hit the stop capture (Red) button or use `Capture > Stop`. You can then inspect some of the captured packets.

There are various sections of the WireShark capture screen. Below the menu section there are three panes:

1. Captured packet listing. This gives a list of the all packets that have been captured. You can scroll through then and use your mouse to select packets.

2. Packet details. This shows the protocol headers and their content for the selected packet.

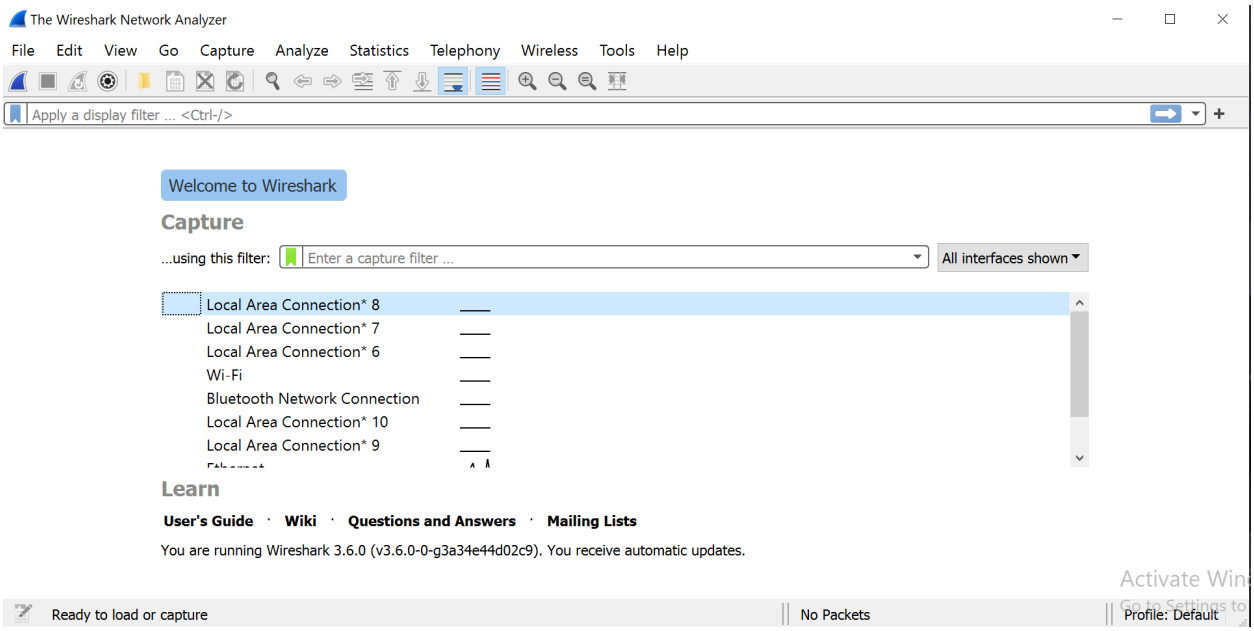

Figure 1: WireShark startup screen with interface selection.

3. Packet content. This a canonical hex+ASCII display of the selected packet content.

Display Filter: Below the menu bar on the left, there is a filter entry box that can be used to limit what is displayed in the packet capture panes. This is useful when there is a lot of network activity that would otherwise clutter up the listing. An example of using the display filter is as follows. First, figure out the IP address associated with the host that you are planning to interact with, i.e., `<address>` in dotted quad format. In some of the experiments below, the host is `compeng4dn4.mooo.com`, whose IP address is `99.236.34.223`. Then type `ip.addr == 99.236.34.223` into the Filter box. This will limit the packets displayed to those with that source or destination address.

There are many WireShark display filter examples online. There is also a cheat sheet at

`http://packetlife.net/media/library/13/Wireshark_Display_Filters.pdf`

Capture Filter: An alternative is to use a capture filter. To enter a capture filter, pull down the Capture > Options menu (or click on the circular capture options icon). Enter the filter in the text area. You can also save capture filters by using the Capture > Capture Filters menu. Just click on the "Create a new filter" button and enter a name and the filter. See the lecture notes for some examples of capture filters, e.g., to capture packet interactions with the host `compeng4dn4.mooo.com`, just enter `host compeng4dn4.mooo.com` as the capture filter.

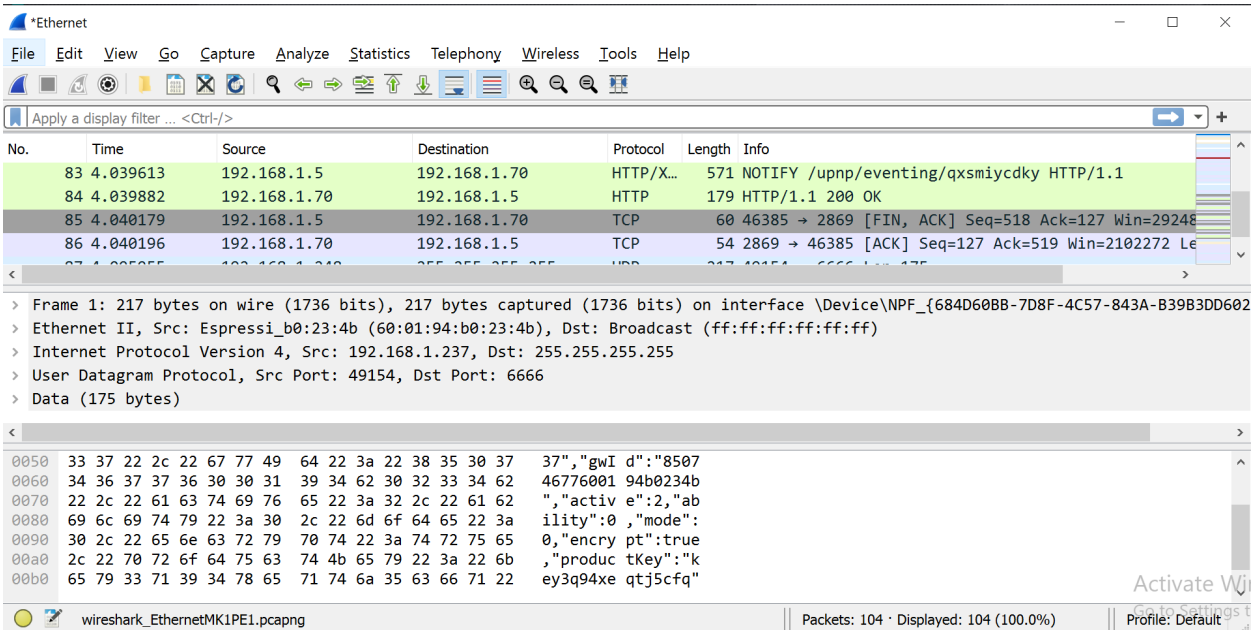

Figure 2: WireShark Capture Screen