TLEN 5410 – Network Management

and Automation

Lab 0.3

Protocol and Network Analyzers (Wireshark, tcpdump)

University of Colorado Boulder

Interdisciplinary Telecom Program

Professor Levi Perigo, Ph.D.

Objectives

Learn the basic operations of Wireshark.

Lean how to capture and analyze ICMP traffic.

Demonstrate best practices for analyzer placement.

Differentiate Wireshark captures between switches and routers.

Wireshark Features: display filter buttons, profiles.

Display which NICs on analyzer are capturing traffic.

Display IPv4 and/or IPv6 addresses on NICs.

Learn how to perform continuous captures for HTTP requests.

Explain and display different ways to demonstrate top talkers on the network.

Create coloring rules.

Create graphs for visual representation.

Summary

In this lab, you’ll get acquainted with Wireshark, and make some simple packet captures and observations.

The basic tool for observing the messages exchanged between executing protocol entities is called a packet sniffer. As the name suggests, a packet sniffer captures (“sniffs”) messages 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 messages. 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. Similarly, received packets are never explicitly addressed to the packet sniffer. Instead, a packet sniffer receives a copy of packets that are sent/received from/by application and protocols executing on your machine. The packet capture library receives a copy of every link-layer frame that is sent from or received by 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. 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/]](http://www.wireshark.org/) for this lab, 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.

Objective 1 - Downloading Wireshark and Navigation Overview

1. Download and install Wireshark:

• Go to [http://www.wireshark.org/download.html a](http://www.wireshark.org/download.html)nd download and install.

• The Wireshark FAQ has many helpful hints and interesting tidbits of information, particularly if you have trouble installing or running Wireshark.

• Helpful Install video: <https://www.youtube.com/watch?v=flDzURAm8wQ>

2. Wireshark Navigation

Helpful navigation video: <https://www.youtube.com/watch?v=PYrCS21sPbA>

Objective 2.1 – ICMP

Before proceeding, please disable IPv6 in your system.

1. Open command prompt/terminal (depending on the operation system).

2. Start Wireshark and begin capture on the interface through via which Internet is accessible.

3. Ping any “hostname” (where the “hostname” is a URL, example: ping [www.google.com](http://www.google.com/))

4. When the ping finishes, stop the capture. [Press Ctrl + C in MAC to stop the ping.]

5. Filter the capture to only display the Ping traffic, and display only that output. [**5 points**]

6. Were the pings successful? Provide the filtered Wireshark screenshot, and explain how you know they were/were not successful. [**5 points**]

Yes, they were successful because I got request and response

7. What is the IP address of your host? (show within Wireshark) [**5 points**]

10.0.2.15

8. What is the IP address of the destination host? (show within Wireshark) [**5 points**]

8.8.8.8

9. Examine one of the ping request packets sent by your host. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number, and identifier fields? [**10 points**]

Type: 8

Code: 0

Checksum bytes, sequence number, and identifier fields: 4 bytes

10. Examine the corresponding ping reply packet. What are the ICMP type and code numbers? What other fields does this ICMP packet have? How many bytes are the checksum, sequence number and identifier fields? [**10 points**]

11. Start a New Wireshark Capture. Ping a hostname or IP that gives you a “Request Timed Out” message. (e.g. You can try [www.wellsfargo.com or](http://www.wellsfargo.com/) any another website/IP of your choice.). Filter the ICMP traffic. Find the Type and Code of the packet in the above scenario. Paste the relevant screenshots. [**5 points**]

12. Do you see both ICMP Echo Request and Echo Reply messages? [**5 points**]

No, I don’t see the reply message. The type is 3 (Destination unreachable)

Objective 2.2 – ICMP and Traceroute

1. Open command prompt / terminal (depending on the operation system)

2. Start Wireshark and begin capture.

3. Traceroute to a “hostname” (where the “hostname” is a URL)

a. Example: tracert [www.google.com](http://www.google.com/)

4. When trace completes, stop capture.

5. Provide a screenshot of the trace. Was it successful? How do you know? [**5 points**]

No, it wasn’t successful because I still got type 3 (Destination unreachable)

6. Filter the Wireshark to only show the relevant data. Examine the ICMP traffic in Wireshark. Is the protocol number the same on a Ping as it is with a trace? What is that protocol number? What is different in the capture from the trace when compared to the capture of the Ping in Objective 1? [**10 points**]

Traceroute protocol: UDP(17)

I don't see the traceroute protocol number in ping

7. Within the Wireshark trace, is there a link whose delay was significantly longer than the others? How do you know this from within Wireshark? What was the name of the router at that hop? What do you think is the reason of the delay and does this delay equate to a corresponding delay in web traffic? [**10 points**]

Objective 3 - Wireshark NICs and IPv4/IPv6 addresses

1. Provide a screenshot of the NICs that Wireshark has to choose from on the PC. [**5 points**]

2. Which interface is currently capturing traffic? How do you know? (Provide a

screenshot) [**5 points**]

3. While Wireshark is running:

a. Type <http://gaia.cs.umass.edu/wireshark-labs/INTRO-wireshark-file1.html> in your browser.

b. In your Wireshark Capture display only the HTTP (Web) traffic. (Paste screenshot). [**5 points**]

c. Can you see the text displayed on the browser in your Wireshark packets as well? Why/why not? Paste relevant screenshots. [**5 points**]

1. While Wireshark is running:
   1. Type <https://sites.google.com/a/colorado.edu/rahil-gandotra/projects/netmanlab1> in your browser.
   2. Mention the steps to be followed to display this specific traffic. [**5 points**]

* Start the wireshark capturing by clicking the button
* Type the <https://sites.google.com/a/colorado.edu/rahil-gandotra/projects/netmanlab1> in the url of browser,
* Once it’s done loading and displaying the content of html page, stop the wireshark, and apply the filter string (HTTP) to the display.
* Sometimes it doesn’t capture some packets, but if you scroll the webpage, then you can capture.
  1. Can you see the text displayed on the browser in your Wireshark packets as well? Why/ why not? [**5 points**]

No, I don’t think that I’m able to capture the packets because it’s HTTPS. Wireshark captures all traffic on a network interface. The thing with HTTPS is that it is application layer encryption. Wireshark is not able to decrypt the content of HTTPS. This is because HTTPS encrypts point to point between applications.

The idea here is that HTTPS traffic that travels over the Internet is confidential, a random router or person who happens to capture your packages cannot decrypt the HTTPS without the decryption key. Thus, Wireshark cannot decrypt HTTPS traffic without the decryption key

Continuous Captures, Filtering, and Analysis

Objective 4.1

1. Initiate a Wireshark capture that uses multiple files, where it creates a new file every 5 minutes, for a total of 15 minutes. Provide a screenshot of the Capture Options you selected. Remember where you save this file, as we will use it in the future. Also, try to use the wireless NIC if possible, as there will be more traffic in the capture to analyze. [**10 points**]

2. Browse ten different websites, during this 15-minute continual capture time window.

3. What are 3 reasons why you would want to create multiple files? [**10 points**]

4. How do you view the three files captured within Wireshark, and move between them, after they have been completed and saved? [**5 points**]

5. How do you see which websites you browsed during the capture? [**5 points**]

Objective 4.2

1. Create two Display filter buttons. One for traffic sourced from your machine’s IP

address and one that only displays HTTP GET requests. (Hint: HTTP contains)

2. Provide a screenshot of the buttons you created, and the corresponding filtered capture. [**10 points**]

Objective 4.3

1. Create a coloring rule for HTTP traffic. [**5 points**]

2. Provide a screenshot of your capture from above, showing where you changed the color of HTTP GET requests to Red background with White lettering. (Hint: Did you remember to move your color rule to the top?) [**5 points**]

Top Talkers, Profiles, and Graphs

Objective 5.1

1. For the following objectives, use any of the saved .pcap files from Objective 4.1.

2. Determine Top Talkers on the network. [**10 points**]

3. What are two ways you can determine what device/IP address is transmitting the most traffic on the network? Provide a screenshot of one of those ways. (Hint: Protocol Hierarchy; Conversations; Endpoints) [**10 points**]

Objective 5.2

1. For the following objectives, use any of the saved .pcap files from Objective 4.1.

2. Create a graph that displays the top 4 protocols from the capture.

3. Provide screenshot of the graph. [**10 points**]

Objective 6 – Parsing .pcap using Python

1. Start a new capture in Wireshark using the capture filter of ‘icmp’. Open the command prompt/terminal and execute these commands -

ping -4 google.com

ping wellsfargo.com

[Use -c 4 option if pinging from MAC.]

1. Stop the capture and save the file as .pcap. Paste a screenshot of the saved capture file.
2. Write a script using Python that parses the saved .pcap file and prints out only the source and destination IPs of each packet of the file sequentially. You can use the Python library pcapfile [<https://pythonhosted.org/pypcapfile/installing.html>] or dpkt [<https://dpkt.readthedocs.io/en/latest/>] for this purpose. Paste screenshots of the working of your script, and submit your script along with the lab report.

[**20 points**]

Tcpdump

Objectives

Learn the basic operations of packet analyzers using tcpdump.

Learn how to capture, store and analyze packet captures using tcpdump.

Objective 1 – Basic operations of Tcpdump

1. Start the course VM and ensure that Internet is accessible.
2. Install tcpdump if not already installed.
3. Execute the command: sudo tcpdump -D and paste the screenshot. What does the output indicate? [**10 points**]
4. On another terminal initiate a ping to google.com.
5. Start capturing packets using the command: sudo tcpdump -i <interface-name>. Paste the screenshot. [**5 points**]
6. How would you filter for ICMP traffic only? Paste the screenshot. [**2.5 points**]
7. How would you display all IP addresses instead of domain names? Paste the screenshot. [**2.5 points**]
8. Stop the packet captures and the ping.
9. Initiate a new ping using the command: ping -s 1000 google.com.
10. Start a new capture (using same filters as above). What is the difference in the capture now? [**5 points**]
11. Start a new capture and filter to display only TCP traffic. On another terminal execute the command: curl <http://google.com>. Paste the screenshot. [**5 points**]
12. How would you filter to display only HTTP traffic? Execute the curl again and paste the screenshot of the capture. [**5 points**]

Report Questions

1. Now that you have used both Wireshark and tcpdump, which packet analyzing tool would you prefer and why? [**5 points**]
2. Describe situations in which one would be preferred over the other. [**5 points**]

Bonus Objective

Run tcpdump on the VM and save the output to a .pcap file. Browse to the sites mail.google.com, reddit.com, youtube.com, learn.colorado.edu, netflix.com, time.gov and cnn.com. Write a script in Python that uses regular expressions to find all the unique A records in the .pcap file and prints them. Paste screenshots of the saved .pcap file and the working of your code, and submit your script along with the lab report. [[What is an A record?](https://support.dnsimple.com/articles/a-record/)]