Wireshark – Packet & Traffic Analysis

1 OVERVIEW

The learning objective of this lab is for students to get familiar with the concepts of packet and traffic analysis. There are many tools for packet sniffing, network traffic analysis, and HTTP debugging. However, Wireshark is the most popular, complete tool for the job. Understanding of this tool is often considered a required skill for many security and network auditing jobs.

1.1 LAB SETUP

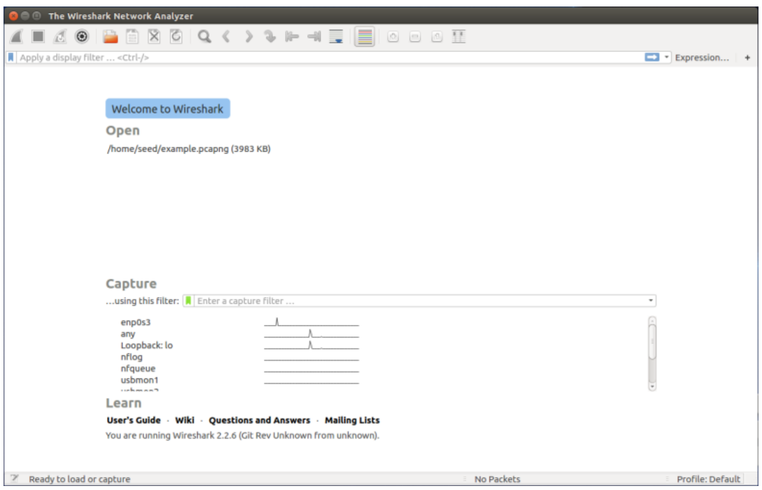
Wireshark has already been installed on lab VMs. Wireshark also works on Windows and macOS. Administrator or root privileges are required to run Wireshark at full functionality.

2 LAB TASKS

2.1 TASK 1: CAPTURING A TRACE

2.1.1 Start Wireshark

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. Wireshark is already installed on Lab VM, start Wireshark from Dash menu on the left. You should see following window.



2.1.2 Wireshark Live Capture

Wireshark can capture traffic from many different network media types - and despite its name - including wireless LAN as well. Which media types are supported, depends on many things like the operating system you are using. An overview of the supported media types can be found at https://wiki.wireshark.org/CaptureSetup/NetworkMedia.

You can start a live capture of your ethernet device by clicking on a device **enp0s3** in the Capture list. This captures all traffic going through your ethernet device. This capture can cause performance issues as it captures and tries to present all information passing through your network device. To narrow down your search you can apply filters to your capture. For example, filter *tcp port http* will capture only TCP traffic through port 80.

2.1.2.1 Saving Captured Traces

You can save captured packets simply by using the File › Save As… menu item. You can choose which packets to save and which file format to be used. Not all information will be saved in a capture file. For example, most file formats don't record the number of dropped packets.

2.1.3 Inspecting Captured Traces

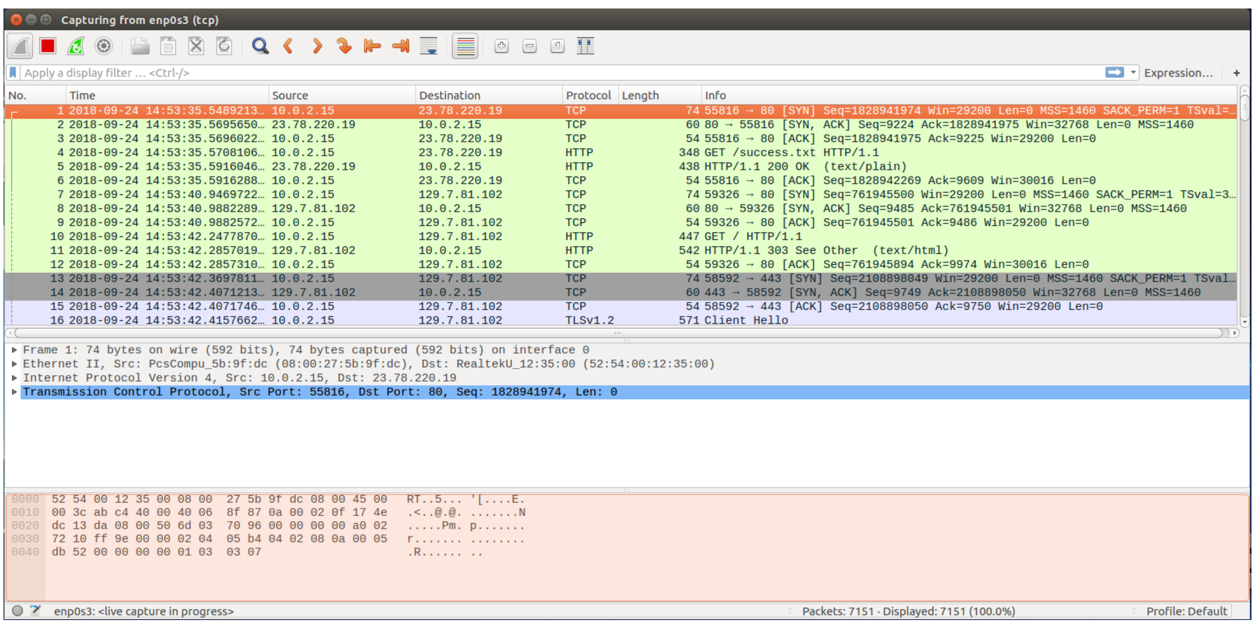
Wireshark also allows you to inspect captured packet traces from files. Wireshark is able to handle files generated from different software such as tcpdump, nmap, Microsoft Network Monitor, Cisco Networks NetXray, etc.

2.2 TASK 2: TCP BASICS AND BEHAVIOR

Start a TCP trace for your ethernet device (enp0s3)

Open a web browser and go to www.uhcl.edu

You should see a trace like following in your Wireshark window.



The green packets are HTTP, TCP segments are colored as light purple.

- Can you explain how encrypted connection is established?
- Do you see public key infrastructure at work?

Identify the TCP segments that are used to initiate the TCP connection between the client computer and www.uhcl.edu.

- How many segments are used?
- What is in the TCP header that identifies each segment as a handshaking segment?

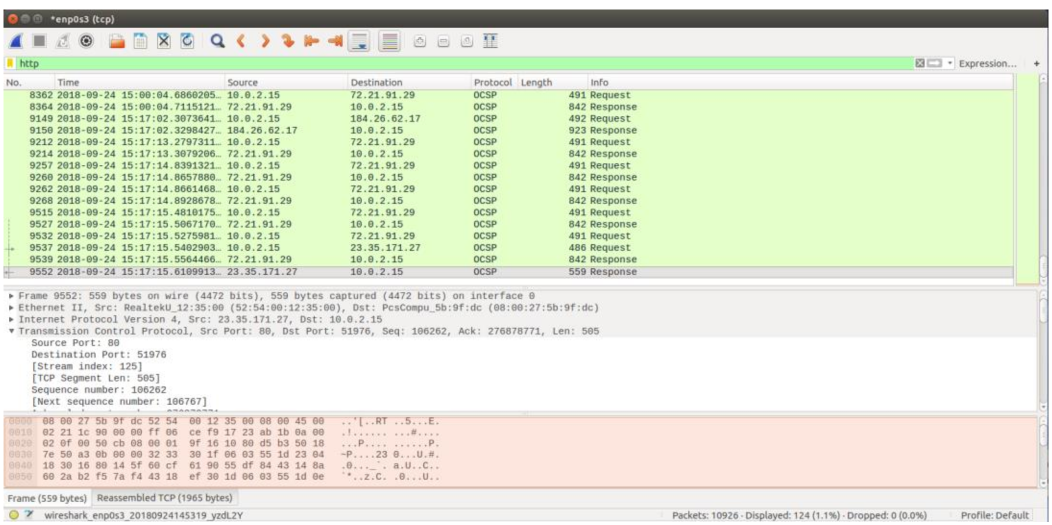
How are sequence and acknowledgment numbers are determined?

- Can you identify series of at least six TCP segments using these numbers?
- What is the length of each of these six TCP segments? The length of the TCP segment is only the number of data bytes carried inside the segment (excluding the headers).
- What is specified by the value of the Acknowledgement field in any received ACK-segment? How does www.uhcl.edu; for example, determine this value?
- How much data (number of bytes) does the receiver typically acknowledge in one ACK?

2.3 TASK 3: HTTP BASICS AND BEHAVIOR

Go ahead and browse the university website to generate more HTTP traffic, you can also log in your webmail account (make sure you don't save any passwords on VM). If you leave capture open for long enough, you'll notice that it captures a lot of unnecessary TCP traffic as well.

For this exercise, you'll need to focus on HTTP traffic only. To make this simpler, apply an **http** filter to your capture. Once you are done browsing the web, you can stop the live capture.



Find an HTTP request from a client (source) to server (destination) and try to answer following questions.

- Inspect the HTTP header by expanding the down arrow beside "Hypertext Transfer Protocol" In the middle pane.

Observe that the HTTP header follows the TCP and IP headers, as HTTP is an application protocol that is transported using TCP/IP. To view it, select the packet, find the HTTP block in the middle panel, and expand it (by using the "v" expander or icon).

Explore the headers that are sent along with the request. First, you will see the GET method at the start of the request, including details such as the path. Then you will see a series of headers in the form of tagged parameters. There may be many headers, and the choice of headers and their values vary from browser to browser.

See if you have any of these common headers

- Host. A mandatory header, it identifies the name (and port) of the server.
- User-Agent. The kind of browser and its capabilities.
- Accept, Accept-Encoding, Accept-Charset, Accept-Language. Descriptions of the formats that will be accepted in the response, e.g., text/html, including its encoding, e.g., gzip, and language.
- Cookie. The name and value of cookies the browser holds for the website.
- Cache-Control. Information about how the response can be cached.

Now select an HTTP packet sent from server (source) to the client (destination).

The Info for this packet should indicate “200 OK” in the case of a normal, successful transfer. You will see that the response is similar to the request, with a series of headers that follow the “200 OK” status code. However, different headers will be used, and the headers will be followed by the requested content.

Examine the common headers such as:

- Server. The kind of server and its capabilities
- Date, Last-Modified. The time of the response and the time the content last changed
- Cache-Control, Expires, Etag. Information about how the response can be cached

2.3.1 Inspecting Unencrypted Transmission

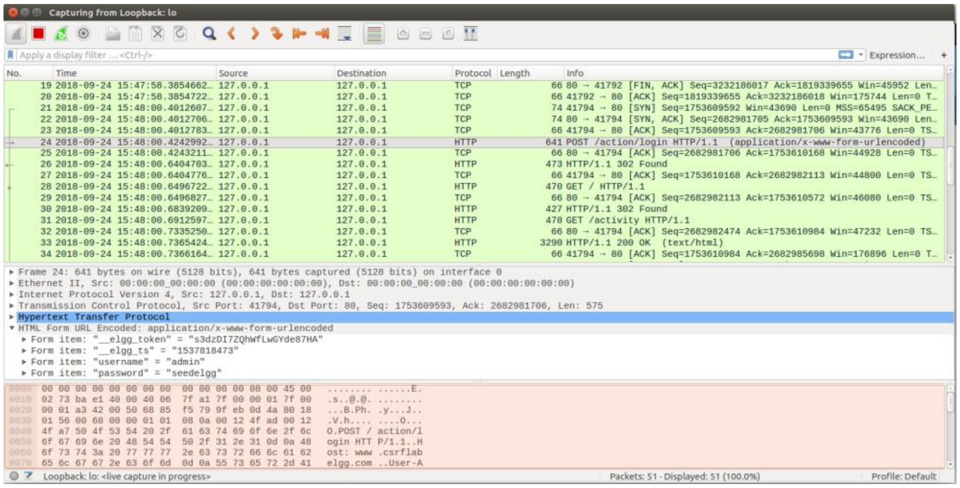
For this exercise, we will need a web application that does not use an encrypted HTTP connection.

In your Lab VM, open the browser and go to http://www.csrflabelgg.com

Make sure that your Wireshark is running and actively capturing traffic from loopback and not ep0s3. (Since the web application is hosted locally on the VM)

On the CSRF Lab Site, log in as admin. With username “admin” and password “seedelgg”

Find a HTTP POST request to page /action/login



Inspect HTTP request headers and contents

- Do you see the username and password in plain text?
- What are the implications of sending unencrypted HTTP requests?

Can you replicate the results for the UHCL webmail or your favorite email service?

- If not, why?
- How can you encrypt your web traffic?