Practical work #1

Michel FACERIAS

January 26, 2022

Abstract

This PW contains 3 sections :

- Discovering Wireshark ;
- UDP session analysis ;
- TCP session analysis.

Along this document, we are going to use IPv4 address format to identify hosts. IPv4 address (ie IP address) is a string made with 4 numbers delimited by 3 dots, like *1.2.3.4*.

At this stage, you don't need to know anything else about IP addresses.

Be careful !
Installing or using a packet capture application may violate an organization's security policy, which may result in serious legal and financial consequences. It is therefore recommended to obtain the required authorizations before using it.

Contents

1 Discovering Wireshark **4**
1.1 Global approach . . . 4
1.1.1 Question : What is Wireshark ? . . . 4
1.1.2 Question : What are Wireshark main features ? . . . 4
1.1.3 Question : What Wireshark is not ? . . . 4
1.2 Starting Wireshark . . . 4
1.2.1 To do : Run your first capture . . . 4
1.2.2 Questions : What do the colored lines in the upper part of the main window represent ? . . . 5
1.2.3 Question : If you left-click one of these lines, what is the content of the centered part of the main window ? . . . 5
1.2.4 Question : What are the information in the lower part of the main window ? . . . 5
1.2.5 Question : What are this 4 buttons used for ? . . . 5
1.2.6 To do : Start an advanced capture . . . 5
1.2.7 Question : how many packets have been transmitted by your host ? . . . 6
1.2.8 Question : how many packets have been received by your host ? . . . 6
1.2.9 Question : What are the four lines you can see in the Wireshark capture ? . . . 6
1.3 Filtering while capturing . . . 6
1.3.1 To do : Read reference information . . . 6
1.3.2 Question : how to capture packets from the 1.2.3.4 host ? . . . 6
1.3.3 Question : how to capture packets to the 6.7.8.9 host ? . . . 6
1.3.4 Question : how to capture packets from the 1.2.3.4 host and to the 6.7.8.9 host ? 6
1.3.5 Question : how to capture packets from the 1.2.3.4 host and to the 6.7.8.9 host and returned as an answer ? . . . 6
1.3.6 To do : look at port/service mapping . . . 6
1.3.7 Question : what is the http service port number ? . . . 6
1.3.8 Question : what is the ssh service port number ? . . . 6
1.3.9 Question : how to capture packets relative to an http connection ? . . . 6
1.3.10 Question : how to capture packets relative to an ssh connection between 1.2.3.4 and 6.7.8.9 ? . . . 6

2 Transport layers analysis **6**
2.1 Understanding NetCat . . . 6
2.1.1 To do : Read the NetCat manual . . . 7
2.1.2 Question : What's the command line to connect as a client to 1.2.3.4 host on port 80 in TCP mode ? . . . 7
2.1.3 Question : What's the command line to connect as a client to 1.2.3.4 host on port 53 in UDP mode ? . . . 7
2.1.4 Question : What's the command line to build a server host on port 25 in TCP mode ? . . . 7
2.1.5 Question : What's the command line to build a server host on port 53 in UDP mode ? . . . 7
2.2 UDP transport . . . 7
2.2.1 Question : What is the *UDP echo service* port number ? . . . 7
2.2.2 Question : What is the command line to connect to the *echo server* using UDP? 7
2.2.3 Question : What is the Wireshark capture filter to only observe such connection ? 7
2.2.4 Todo : Implement the UDP client and analyse frames . . . 7
2.2.5 Question : What happens when you input `Hello from client` and why ? . . . 7
2.2.6 Question : How many frame can you see before the frame carrying `Hello from client` sent by the client? . . . 7
2.2.7 Question : How many frame can you see after the frame carrying `Hello from client` sent back by the server? . . . 7

2.2.8 Question : How many frame in total can you see in this capture ? 7
2.2.9 Question : Describe the frames can you see in this capture ? 7
2.2.10 Question : What is the server port number and why ? 7
2.2.11 Question : What is the client port number and why ? 7
2.3 TCP transport 7
2.3.1 Question : What is the *TCP echo service* port number ? 8
2.3.2 Question : What is the command line to connect to the *echo server* using TCP? 8
2.3.3 Question : What is the Wireshark capture filter to only observe such connection ? 8
2.3.4 Todo : Implement the TCP client and analyse frames 8
2.3.5 Question : What happens when you input `Hello from client` and why ? . . . 8
2.3.6 Question : How many frames can you see before the frame carrying `Hello from client` sent by the client ? 8
2.3.7 Question : What happens just after the `Hello from client` frame sent by the client? 8
2.3.8 Question : What happens just after the `Hello from client` frame sent back by the server? 8
2.3.9 Question : How many frame can you see after the frame carrying `Hello from client` sent back by the server? 8
2.3.10 Question : How many frame in total can you see in this capture ? 8
2.3.11 Question : Describe the frames can you see in this capture ? 8
2.3.12 To do : Enumerate the sequence number of all the client frames 8
2.3.13 Question : How can you describe this sequence ? 9
2.3.14 Question : How can you prove there were no lost frames ? 9
2.3.15 Question : What happens after each client send ? 9
2.3.16 Question : What happens after each server send ? 9
2.3.17 Question : What is the server port number and why ? 9
2.3.18 Question : What is the client port number and why ? 9

3 Conclusion of this practical work **9**
3.0.1 Question : How can we know which service we want to join on a server ? 9
3.0.2 Question : How can we know which port a client is going to use to join on a server ? 9
3.1 About UDP 9
3.1.1 Question : Is UDP a secure way to transport data ? 9
3.1.2 Question : What should UDP use for, and why ? 9
3.2 About TCP 9
3.2.1 Question : Is TCP a secure way to transport data ? 9
3.2.2 Question : What should TCP use for, and why ? 9

Be careful !

Installing or using a packet capture application may violate an organization's security policy, which may result in serious legal and financial consequences. It is therefore recommended to obtain the required authorizations before using it.

1 Discovering Wireshark

Official website : http://www.wireshark.org/

1.1 Global approach

To help you, you can read Wireshark documentation.

1.1.1 Question : What is Wireshark ?

1.1.2 Question : What are Wireshark main features ?

1.1.3 Question : What Wireshark is not ?

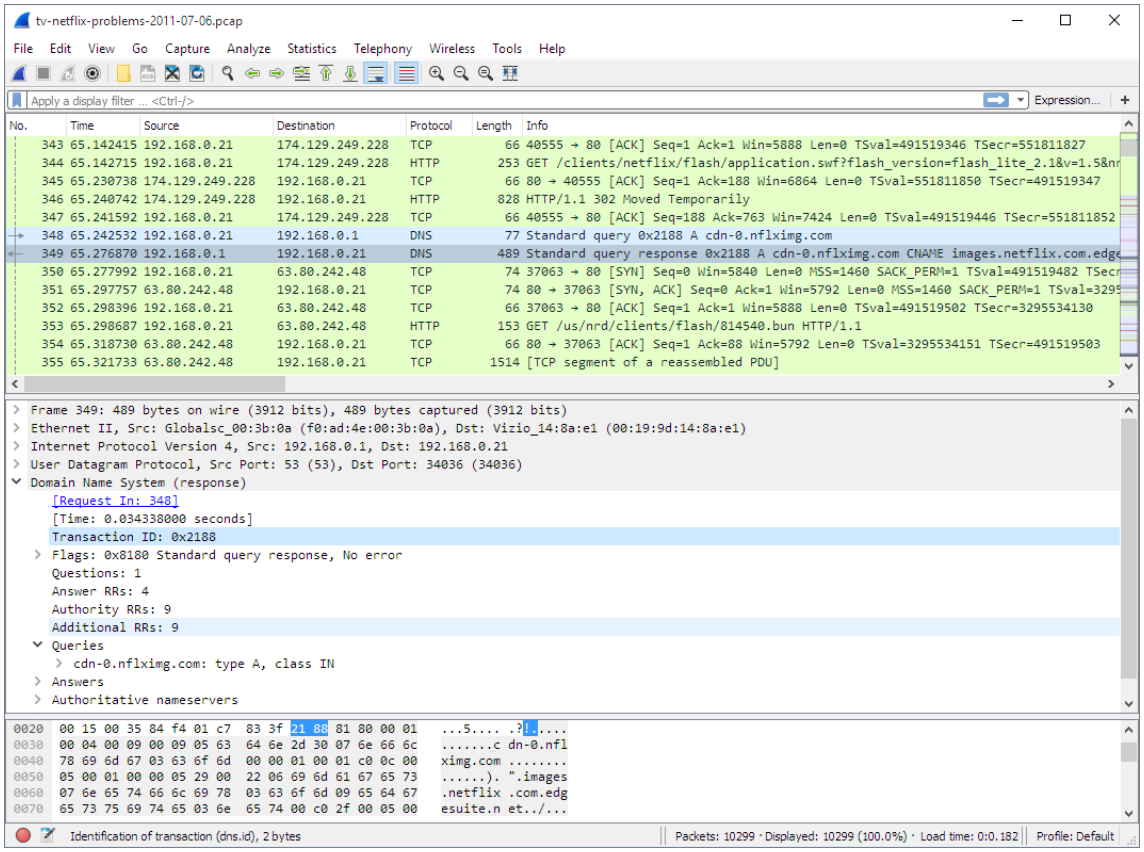
1.2 Starting Wireshark

1.2.1 To do : Run your first capture

On Debian GNU linux, search in *Application/Internet/Wireshark* menu. Left-click on the menu, Wireshark opens.

Left-click on the blue shark fin at the left of the toolbar. Wait a few seconds. Left-click on the red square.

This is an example of the main window, accessible after a capture has ran :



1.2.2 Questions : What do the colored lines in the upper part of the main window represent ?

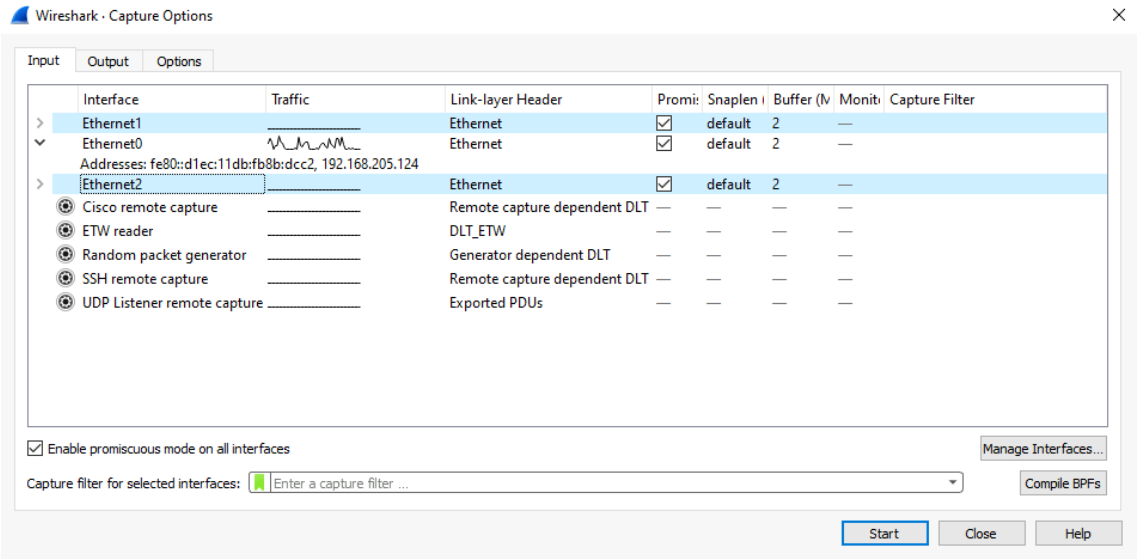
1.2.3 Question : If you left-click one of these lines, what is the content of the centered part of the main window ?

1.2.4 Question : What are the information in the lower part of the main window ?

1.2.5 Question : What are this 4 buttons used for ?

1.2.6 To do : Start an advanced capture

Left-click on the fourth toolbar's button. Using this dialog box, you can tell Wireshark what you want exactly to capture.

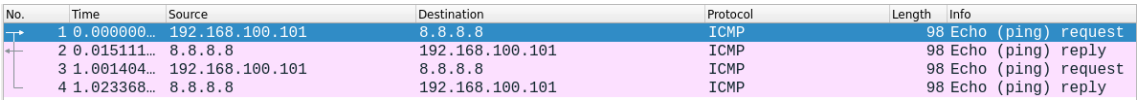


First, you need to chose the network interface on which you are going to record data. Don't choose *any* interface if it exists. If your computer has more than one network card, the small *traffic* graph tells which is carrying data.

In a second step, type a capture filter rule. The background of this field will remain red until you enter a valid filter. As an example, use `icmp` as filter and click on start button.

Open a console from *Application/System/Terminal*. In the console input `ping -c2 8.8.8.8` and validate your command. Then stop the capture by clicking the red square on Wireshark window.

You should obtain something like bellow.



Look at the console the number packets transmitted and received by your host.

1.2.7 Question : how many packets have been transmitted by your host ?

1.2.8 Question : how many packets have been received by your host ?

1.2.9 Question : What are the four lines you can see in the Wireshark capture ?

1.3 Filtering while capturing

In the previous section, we learned how to use Wireshark to capture network flows and how to decode these flows. Now, we need to learn about how to write some advanced filters.

1.3.1 To do : Read reference information

Read carefully this reference information are at :
https://www.wireshark.org/docs/wsug_html_chunked/ChCapCaptureFilterSection.html.

1.3.2 Question : how to capture packets from the 1.2.3.4 host ?

1.3.3 Question : how to capture packets to the 6.7.8.9 host ?

1.3.4 Question : how to capture packets from the 1.2.3.4 host and to the 6.7.8.9 host ?

1.3.5 Question : how to capture packets from the 1.2.3.4 host and to the 6.7.8.9 host and returned as an answer ?

1.3.6 To do : look at port/service mapping

Open a console and input `less /etc/services`. Use the mouse whell to browse the file. Quit typing `q` or close console window.

1.3.7 Question : what is the http service port number ?

1.3.8 Question : what is the ssh service port number ?

1.3.9 Question : how to capture packets relative to an http connection ?

1.3.10 Question : how to capture packets relative to an ssh connection between 1.2.3.4 and 6.7.8.9 ?

2 Transport layers analysis

2.1 Understanding NetCat

Netcat is a simple Unix utility which reads and writes data across network connections, using TCP or UDP protocol.

It is designed to be a reliable "back-end" tool that can be used directly or easily driven by other programs and scripts. At the same time, it is a feature-rich network debugging and exploration tool, since it can create almost any kind of connection you would need and it has several interesting built-in capabilities.

NetCat, or "nc" as the actual program is named, should have been supplied long ago as another one of those cryptic but standard Unix tools.

In the simplest usage, it works as a TCP client. It creates a TCP connection to the given port on the given target host server. Your standard input is then sent to the server, and anything that comes back through the connection is sent to your standard output.

This continues indefinitely, until the network connection shuts down, from server side, or by stopping the client with *CTRL+C*.

NetCat can act as a mono client TCP server to. Launched before any client, it waits for an incoming connection. When a connection is established, it works exactly as in client mode, sending all inputs to the network, and print all data received from.

To read more about NetCat, you have to use the `man` command.

2.1.1 To do : Read the NetCat manual

Open a console window and input `man nc` or `man netcat`. Browse the content using the mouse wheel, or arrows keys.

Read carefully the manual.

2.1.2 Question : What's the command line to connect as a client to 1.2.3.4 host on port 80 in TCP mode ?

2.1.3 Question : What's the command line to connect as a client to 1.2.3.4 host on port 53 in UDP mode ?

2.1.4 Question : What's the command line to build a server host on port 25 in TCP mode ?

2.1.5 Question : What's the command line to build a server host on port 53 in UDP mode ?

2.2 UDP transport

At this time, we are going to analyse a UDP pseudo session. To help you to do the job, we will use an external host as a *echo server*. An *echo server* send back all you will send to it. The IP address of the echo server is **51.15.158.69**.

2.2.1 Question : What is the *UDP echo service* port number ?

2.2.2 Question : What is the command line to connect to the *echo server* using UDP?

2.2.3 Question : What is the Wireshark capture filter to only observe such connection ?

2.2.4 Todo : Implement the UDP client and analyse frames

1. Open Wireshark and run an advanced capture using the appropriate filter ;
2. Open a console window an run the client instance ;
3. At the client console, input `Hello from client` ;
4. Disconnect the client ($CTRL+C$) ;
5. Close the console window (optional)
6. Stop the capture.

2.2.5 Question : What happens when you input `Hello from client` and why ?

2.2.6 Question : How many frame can you see before the frame carrying `Hello from client` sent by the client?

2.2.7 Question : How many frame can you see after the frame carrying `Hello from client` sent back by the server?

2.2.8 Question : How many frame in total can you see in this capture ?

2.2.9 Question : Describe the frames can you see in this capture ?

2.2.10 Question : What is the server port number and why ?

2.2.11 Question : What is the client port number and why ?

2.3 TCP transport

Now, we are going to analyse a TCP session. We will use the same method as for UDP.

2.3.1 Question : What is the *TCP echo service* port number ?

2.3.2 Question : What is the command line to connect to the *echo server* using TCP?

2.3.3 Question : What is the Wireshark capture filter to only observe such connection ?

2.3.4 Todo : Implement the TCP client and analyse frames

1. Open Wireshark and run an advanced capture using the appropriate filter ;
2. Open a console window and run the client instance ;
3. At the client console, input `Hello from client` ;
4. Disconnect from client ($CTRL+C$) ;
5. Close the console window (optional)
6. Stop the capture.

2.3.5 Question : What happens when you input `Hello from client` and why ?

2.3.6 Question : How many frames can you see before the frame carrying `Hello from client` sent by the client ?

2.3.7 Question : What happens just after the `Hello from client` frame sent by the client?

2.3.8 Question : What happens just after the `Hello from client` frame sent back by the server?

2.3.9 Question : How many frame can you see after the frame carrying `Hello from client` sent back by the server?

2.3.10 Question : How many frame in total can you see in this capture ?

2.3.11 Question : Describe the frames can you see in this capture ?

2.3.12 To do : Enumerate the sequence number of all the client frames

In the detailed descriptions of the frames, look at *Seq* number in the TCP part. Look at *Sequence Number* field as raw and relative format.

2.3.13 Question : How can you describe this sequence ?

2.3.14 Question : How can you prove there were no lost frames ?

2.3.15 Question : What happens after each client send ?

2.3.16 Question : What happens after each server send ?

2.3.17 Question : What is the server port number and why ?

2.3.18 Question : What is the client port number and why ?

3 Conclusion of this practical work

3.0.1 Question : How can we know which service we want to join on a server ?

3.0.2 Question : How can we know which port a client is going to use to join on a server ?

3.1 About UDP

3.1.1 Question : Is UDP a secure way to transport data ?

3.1.2 Question : What should UDP use for, and why ?

3.2 About TCP

3.2.1 Question : Is TCP a secure way to transport data ?

3.2.2 Question : What should TCP use for, and why ?