Lab Work No. 02
Protocol Analyzer – Wireshark

1. Aim

In this lab work, we will observe some network protocols "in action," interacting and exchanging messages in a real execution using a software tool called a Sniffer.
The sniffer software we will use in our lab is **Wireshark**.
The aims of this lab are:

- To be able to use the Wireshark protocol analyzer: capture data packets traveling through the network, observe, and analyze them.
- To discover the general characteristics and encapsulation of the "TCP/IP" model protocols.

2. Sniffer Software

A **sniffer software** is capable of intercepting, recording, and analyzing data traveling within a network. It allows capturing the data flow passing through the network and displaying the content of each field that constitutes this data according to the specifications of each protocol used.
It is important to note that data is structured into a set of fields, each containing specific information.

3. Wireshark

Wireshark is an open-source protocol analysis software, or "packet sniffer," used for network troubleshooting, network analysis, protocol development, education, reverse engineering, and even hacking.
Wireshark is cross-platform and runs on Windows, macOS, Linux, Solaris, and FreeBSD. It supports 759 protocols.
Wireshark can be downloaded from:
http://www.wireshark.org/download.html
The official documentation is available at:
http://www.wireshark.org/docs/wsug_html_chunked/index.html

3.1. Working Principle

Wireshark captures **Ethernet layer** frames on a computer, allowing it to capture all messages sent (or received) by all protocols running on that computer. This is because communication follows an **encapsulation principle**, meaning that all upper-layer protocols are ultimately encapsulated within an **Ethernet frame**.
Thanks to its packet analyzer, which understands the structure of messages exchanged by different protocols, Wireshark can display the content of each field in a message depending on the protocol used.

- ***Example:*** Suppose we use a browser to visit the website of the **University Center of Mila**.
  - At the **Application layer**, we use the **HTTP protocol**.
  - The **HTTP message** is encapsulated in **TCP or UDP messages**, which in turn are encapsulated in **IP packets**.
  - The **IP packets** are further encapsulated in **Ethernet frames** before being transmitted over the physical medium.

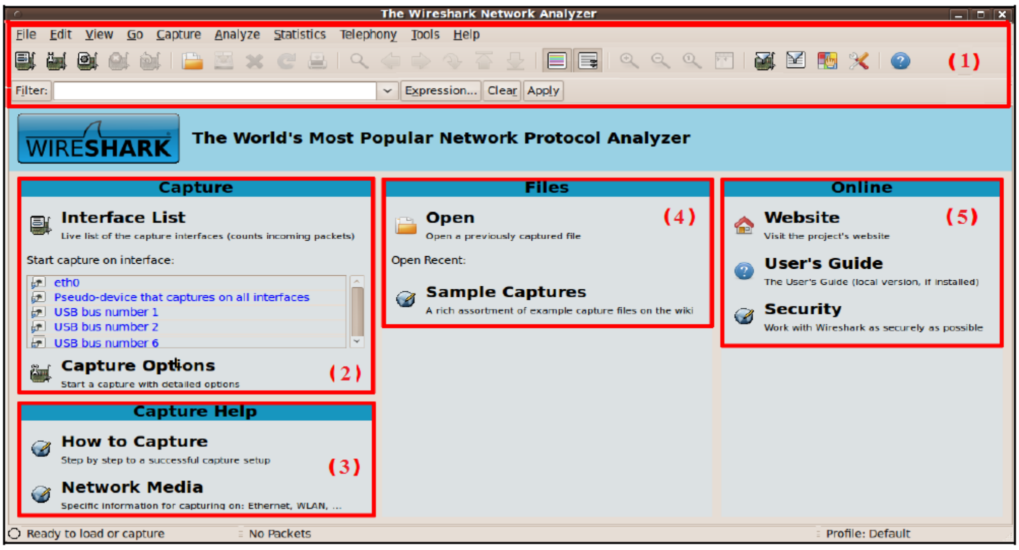
    Wireshark captures an Ethernet frame and can analyze its structure:
  - Identifying the **IP datagram** encapsulated inside.
  - Extracting the **TCP segment** from the IP datagram.
  - Finally, extracting and displaying the **HTTP message**.

3.2. Main Interface

Wireshark allows analyzing:

- **Recorded traffic** from an external file.
- **Live network traffic** (requires administrator privileges).

When launching **Wireshark**, the main interface consists of **five zones**:



(1) *Command Panel:*

The top part of this panel contains the standard drop-down menus, the most interesting ones are:

- **File:** Save or open a captured data file.
- **Capture:** Start and stop data capture.

Below the drop-down menus, there are quick launch icons for the tasks presented in these menus. Then, there is the Filter field to filter the captured packets to be displayed. There is also the "Expression" button right next to it to define a more complex filter expression.

(2) *Network Interfaces and Quick Capture Launch:*

Displays all active network adapters, including virtual interfaces (created by VirtualBox, etc.).

(3) *Packet Capture Help:*

Provides guidance on capturing and analyzing network data.

(4) *Recent Capture Files:*

Displays a list of recently recorded captures for quick access.

(5) *Online Help and User Manual:*

Contains links to Wireshark's main website and user guide.

3.3. Capturing Packets with Wireshark

The capture process consists of three steps:

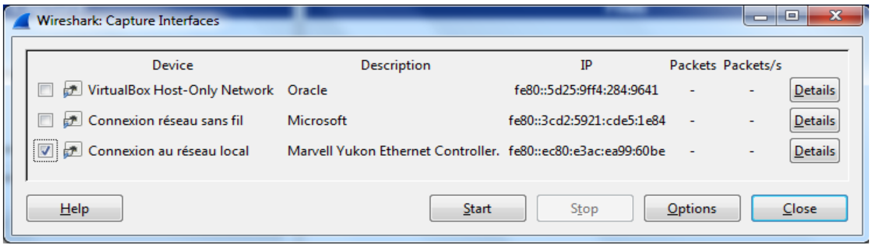
(1) Selecting the **network interface** Wireshark will use (a computer may have multiple interfaces).
(2) Starting the **capture** process.
(3) Stopping the **capture**.

- **How to Start a Capture?**

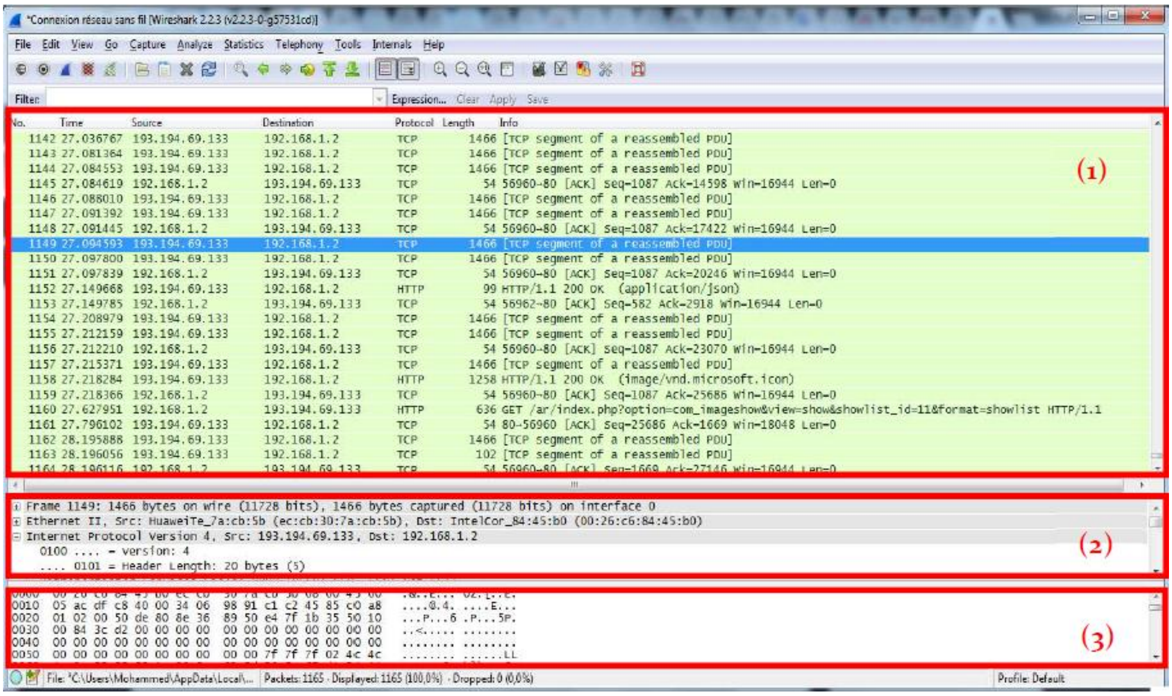
➢ **Method 1:** Using the **"Capture"** section:
  - Select a network interface from the **"Interface List"**.
  - Click **"Start"**: 

➢ **Method 2:** Using the **Command Panel**, either by:
  - Navigate to **Capture → Interfaces**.
  - Click the **first icon** in the quick launch toolbar, titled **"Interface List"**
  - Use the keyboard shortcut **Ctrl + I**.

The following window opens:



Simply select the desired interface and then click the "Start" button to begin the capture.
After selecting an interface and starting the capture, Wireshark captures **all packets sent/received** and displays a window like this:



The interface consists of three areas:

(1) ***The top area:*** displays the list of captured packets.
(2) ***The middle area:*** displays the details of a selected (highlighted) packet from the top area.
(3) ***The bottom area:*** displays the content of the selected packet from the top area in ASCII and hexadecimal.

The capture can be stopped simply by clicking the **"Stop the running live capture"** button in the quick launch icon bar in the **"Control Panel"**: 

3.4. Analyzing Captured Packets

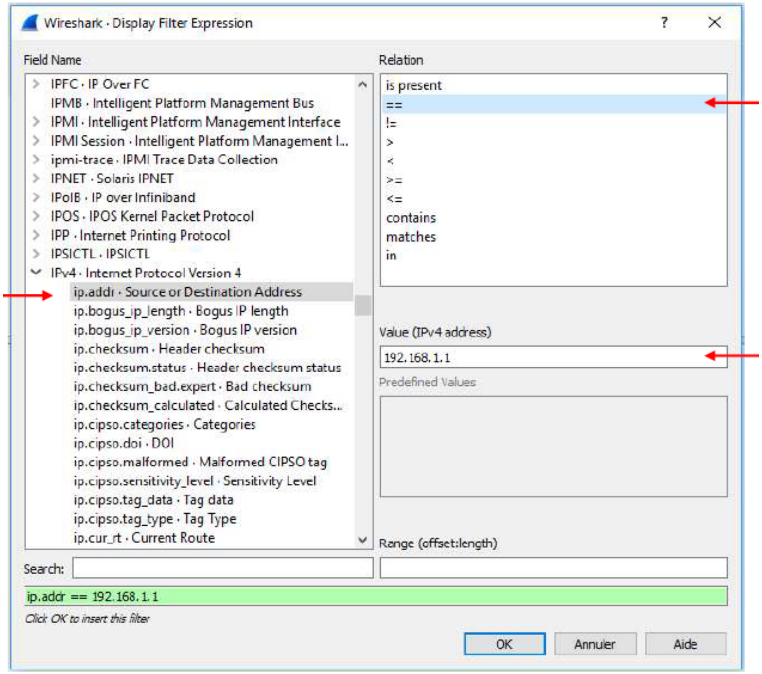
In the capture window, the information displayed in a lower area corresponds to what has been selected in the upper area. Thus, if you want to see the details of a captured packet, simply select it in area (1), and its encapsulation information will be displayed in area (2), along with its hexadecimal content in area (3).

3.4.1. Filtering Captured Packets

It can be observed that there are many types of packets captured and displayed by Wireshark due to the large number of protocols supported by this software. To search within the captured data and display only the traffic of interest, filters are used. The general syntax of a simple filter expression is:

	Protocol	**.**	**Parameter**	**Operator**	**Value**
Example:	ip	.	addr	==	192.168.1.1

The filter expression can be defined visually by clicking on the **"Expression"** button next to the **"Filter"** field in the control panel. To define the filter expression, follow the steps illustrated in the following figure:



(1) Select the protocol, and for the parameter, simply expand the list of parameters by clicking on **">"** in front of the protocol and choose one.
(2) Then, select the comparison operator.
(3) Finally, enter the desired value in the value field.

Throughout the steps, the expression is built at the bottom of the window. Once finished, click **"OK"**.
More complex filter expressions can be defined using logical operators. The format is:

	Expression	**logical operator**	**expression**	**.....**
Example:	TCP	||	UDP	

- **Primary comparison operators:**

English	C-Like	Description	Example
eq	==	Equal	`ip.src == 10.10.10.100`
ne	!=	Not equal	`ip.src != 10.10.10.100`
gt	>	Greater than	`ip.ttl > 250`
lt	<	Less Than	`ip.ttl < 10`
ge	>=	Greater than or equal to	`ip.ttl >= 0xFA`
le	<=	Less than or equal to	`ip.ttl <= 0xA`

- **Primary logical operators:**

English	C-Like	Description	Example
and	&&	Logical AND	`(ip.src == 10.10.10.100) AND (ip.src == 10.10.10.111)`
or	||	Logical OR	`(ip.src == 10.10.10.100) OR (ip.src == 10.10.10.111)`
not	!	Logical NOT	`!(ip.src == 10.10.10.222)` **Note:** Usage of `!=value` is deprecated; using it could provide inconsistent results. Using the `!(value)` style is suggested for more consistent results.

- **Examples of filter expressions:**
  - ip.addr == 192.168.1.1 → Displays packets with source/destination **IP 192.168.1.1**.
  - tcp || ip || dns → Displays **TCP, IP, or DNS** traffic.
  - ip.src == 192.168.1.1 && ip.dst != 172.16.10.2 → Shows packets from **192.168.1.1** excluding destination **IP 172.16.10.2**.
  - eth.addr == 00:2F:4C:01:23:6C → Shows traffic from the **MAC address 00:2F:4C:01:23:6C**

3.4.2. Captured Packet List

In the list of captured packets displayed in area (1), each line corresponds to a captured packet. It contains:

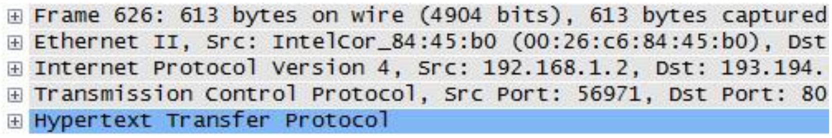
- The packet number assigned by Wireshark,
- The time the packet was captured,
- The source and destination addresses of the packet,
- The protocol type, and protocol-specific information contained in the packet.

The list of captured packets can be sorted by any of these categories by clicking on the corresponding column name.

3.4.3. Packet Details

If a packet is selected in the **top area**, the **middle area** provides details about the packet as well as its different encapsulation levels.

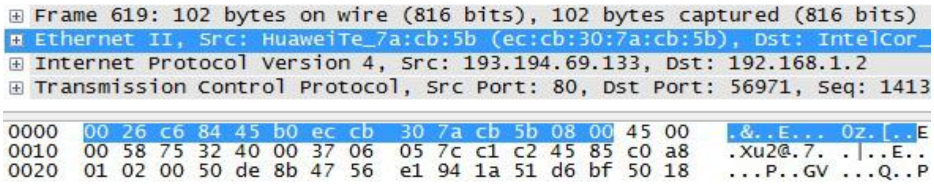
- **Example:** If an HTTP packet is selected in **the top area**, **the middle area** displays something similar to this:



- Each entry corresponds to an encapsulation level. The encapsulation order is read from bottom to top. It is essentially the **de-encapsulation** order that is displayed.
- Wireshark captures a sequence of bytes. It extracts the frame from this sequence, then extracts the IP packet from the frame, followed by the TCP (or UDP) message from the IP packet, and finally, the HTTP message.
- Clicking the **"+"** in front of each encapsulation level allows you to view all its fields. Additionally, some fields can also be expanded. For example, expanding the **Ethernet** entry (data link layer) reveals the **"Destination," "Source,"** and **"Type"** fields. The **"Destination"** and **"Source"** fields can, in turn, be expanded.

3.4.4. Packet Content in Hexadecimal

- The **bottom area** displays frame data in **hexadecimal**.
- Clicking on a protocol layer (in **the middle area)** highlights the corresponding **byte portion**.
- Conversely, clicking on any byte in **the bottom area** highlights the corresponding field in **the middle area**.



4. Required Work

1. Start a packet capture in Wireshark using the Ethernet interface.
2. List five different protocols used by your computer.
3. Apply the following display filters:
   (a) Packets with a TCP destination port of 25.
   (b) All TCP messages except those with source/destination port 80.
   (c) Only packets sent by your machine.
   (d) All except ICMP packets.
4. Perform a "ping" command while capturing packets.
   - Identify the protocol used and its network layer.
5. Visit www.centre-univ-mila.dz in a browser while capturing.
   (a) Identify the protocol used by the browser.
   (b) Apply a filter to show only HTTP messages.
   (c) Select an HTTP packet and answer:
      - Which layer does HTTP belong to?
      - What transport protocol does HTTP use?
      - What is the encapsulation structure?
   (d) Identify the two types of HTTP messages exchanged.
   (e) Measure response time for the HTTP request.
   (f) Find the server's IP address.
6. Save the final capture.
7. Open the saved capture file.