

# UTILISATION DE WIRESHARK

SERVICES ET TRANSPORT DE PAQUETS VERS LES  
APPLICATIONS

## Table des matières

|  |    |
|--|----|
| Table des matières .....   | 1  |
| 1 : La modélisation en couche : .....  | 2  |
| 1.1 : La modélisation en couche : .....  | 2  |
| 1.1.1 : Le modèle TCP/IP : .....   | 2  |
| 1.2 : Modèle et protocoles en couches : le processus d'encapsulation des données. .... | 2  |
| 1.3 : Exercice : .....   | 3  |
| 1.1.2 : Trame numéro 1 : .....   | 3  |
| 1.1.3 : Trame numéro 2 : .....   | 5  |
| 2 Prise en main de Wireshark : .....   | 6  |
| 1.4 : Analyse de trame : .....   | 6  |
| 1.1.4 : Etude de la trame 1 : .....  | 6  |
| 1.1.5 : Etude de la trame 2 : .....  | 9  |
| 1.5 : Capturer et analyser les données ICMP locales : .....                            | 11 |
| 1.1.6 : Examen des données capturées : .....   | 11 |
| 1.6 : Capturer et analyser les données ICMP locales : .....                            | 13 |
| 1.1.7 : Capture les données de l'interface : .....                                     | 13 |
| 1.1.8 : Examen et analyse des données à partir des hôtes distants : .....              | 14 |
| 1.1.9 : Analyse les données ICMPv6 locales .....                                       | 14 |
| 1.7 : Ettercap analyse des trames comme un attaquant simulation MAN in the middle..    | 15 |
| 3 Capture de trame liées à HTTP : .....  | 18 |
| 1.8 : MTU .....  | 19 |
| 4 Application pratiques de Wireshark HTTPS : .....                                     | 21 |
| 5 NetworkMiner : .....   | 23 |
| 1.9 : Première étude : .....   | 23 |
| 1.10 : Etude des différents outils : .....   | 25 |
| 6 Intelligence artificielle : .....  | 25 |
| 1.11 : ChatGPT : .....   | 25 |
| 1.12 : MistralAI le Chat : .....   | 27 |
| 1.13 : Autre méthode : .....   | 29 |
| 1.1.10 : Chat GPT .....  | 31 |
| 1.1.11 : Gemini .....  | 33 |

## UTILISATION DE WIRESHARK

|   |    |
|---|----|
| 1.1 : Vectra AI (Vectra AI - Advanced AI Security - Stop Cyberattacks Fast):..... | 34 |
| Conclusion :.....   | 35 |
| GitHub :.....   | 36 |
| Sources :.....  | 41 |

## 1 : La modélisation en couche :

### 1.1 : La modélisation en couche :

#### 1.1.1 : Le modèle TCP/IP :

1)

| Les couches TCP/IP :    | Protocoles connus  |
|-------------------------|--|
| Couche 4 (application)  | SMTP pour l'envoi de mails, HTTP(S) pour le transfert hypertexte, FTP pour le transport de fichiers, le DHCP pour coordonner l'allocation dynamique d'adresses, le DNS pour résoudre les noms de domaines en adresses IP et le SSH pour les connexions sécurisées. |
| Couche 3 (Transport)    | UDP envoie des informations mais sans garantie de réception et TCP envoie des informations avec une garantie de réception.   |
| Couche 2 (Internet)     | IP gère l'adressage et le routage des paquets (ICMP, ARP).   |
| Couche 1 (Accès réseau) | MAC identifie les équipements au niveau des réseaux locaux (câbles Ethernet, fibre, wifi, Bluetooth).  |

2) Les switches utilisent les couches 1 et 2 car ils ne se servent que des adresses MAC. Les routeurs ont besoin des couches 1, 2 et 3 pour connecter les réseaux entre eux.

### 1.2 : Modèle et protocoles en couches : le processus d'encapsulation des données.

| @MAC destination  | @MAC source        | IP source   | IP destination | Port source | Port Destination |
|-------------------|--------------------|-------------|----------------|-------------|------------------|
| 74:46:a0:9c:54:1e | 74:46:a0:9b:16 :cb | 192.168.1.2 | 192.168.1.1    | 80          | 49200            |

La trame est arrivée à destination et veut envoyer une confirmation à l'expéditeur. Il faut donc inverser les informations.

a)

## UTILISATION DE WIRESHARK

|   |              |              |      |                          |
|---|--------------|--------------|------|--------------------------|
| 112.7782977...  | 192.168.0.90 | 80.12.242.10 | SMTP | 87C: EHLO [192.168.0.90] |
| <ul style="list-style-type: none"> <li>Frame 11: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface 0</li> <li>Ethernet II, Src: RivetNet_d2:d4:13 (9c:b6:d0:d2:d4:13), Dst: Sagemcom_32:44:30 (68:15:90:32:44:30)</li> <li>Internet Protocol Version 4, Src: 192.168.0.90, Dst: 80.12.242.10</li> <li>Transmission Control Protocol, Src Port: 55990, Dst Port: 587, Seq: 1, Ack: 38, Len: 21</li> <li>Simple Mail Transfer Protocol</li> <li>Command Line: EHLO [192.168.0.90]\r\n</li> </ul> |              |              |      |                          |

La trame établit une connexion SMTP entre un poste (192.168.0.90) et un serveur (80.12.242.10) pour envoyer des mails. Il y a écrit Simple Mail Transfer Protocol, qui veut dire SMTP et le port 587 est utilisé pour envoyer des mails avec SMTP.

b)

| Couches du modèle TCP/IP | Protocole associé |
|--------------------------|-------------------|
| 4 - Application          | SMTP              |
| 3 - Transport            | TCP               |
| 2 - Internet             | IP                |
| 1 - Accès réseau         | Ethernet          |

c)

| @MAC destination       | @MAC source            | IP source    | IP destination | Port source | Port Destination |
|------------------------|------------------------|--------------|----------------|-------------|------------------|
| 9c :b6 :d0 :d2 :d4 :13 | 68 :15 :90 :32 :44 :30 | 192.168.0.90 | 80.12.242.10   | 55990       | 587              |

d)

| @MAC destination       | @MAC source            | IP source    | IP destination | Port source | Port Destination |
|------------------------|------------------------|--------------|----------------|-------------|------------------|
| 68 :15 :90 :32 :44 :30 | 9c :b6 :d0 :d2 :d4 :13 | 80.12.242.10 | 192.168.0.90   | 587         | 55990            |

1.3 : Exercice :

## 1.1.2 : Trame numéro 1 :

e)

## UTILISATION DE WIRESHARK

|  |   |
|--|---|
|  | <p>C'est le FTP (File Transfer Protocol).</p> |
|--|---|

Il sert à transporter des fichiers entre deux utilisateurs dans le protocole TCP/IP.

f)

|  |  |
|--|--|
|  | <p>C'est le TCP (Transmission Control Protocol).</p> |
|--|--|

Il permet une communication fiable entre les dispositifs d'un réseau.

g)

|  |   |
|--|---|
|  | <p>Le port 49681 est le port client ouvert.</p> |
|--|---|

h)

|  |  |
|--|--|
|  | <p>La longueur de l'en-tête TCP est de 20 bytes.</p> |
|--|--|

i)

## UTILISATION DE WIRESHARK

Frame 31: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0

Ethernet II, Src: Vmware\_b4:fb:04 (00:0c:29:b4:fb:04), Dst: Vmware\_7d:18:1a (00:0c:29:7d:18:1a)

Internet Protocol Version 4, Src: 192.168.100.1 (192.168.100.1), Dst: 192.168.100.2 (192.168.100.2)

Transmission Control Protocol, Src Port: 49681 (49681), Dst Port: 21 (21), Seq: 39, Ack: 273, Len: 20

Source Port: 49681 (49681)

Destination Port: 21 (21)

[Stream index: 0]

[TCP Segment Len: 20]

Sequence number: 39 (relative sequence number)

[Next sequence number: 59 (relative sequence number)]

Acknowledgment number: 273 (relative ack number)

Header Length: 20 bytes

... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)

Window size value: 255

[Calculated window size: 65280]

[Window size scaling factor: 256]

Checksum: 0x3a88 [validation disabled]

Urgent pointer: 0

[SEQ/ACK analysis]

File Transfer Protocol (FTP)

Développement des « flags »

... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)

000. .... = Reserved: Not set

...0 .... = Nonce: Not set

... 0... = Congestion window Reduced (cwr): Not set

... .0... = ECN-Echo: Not set

... ..0... = Urgent: Not set

... ...1... = Acknowledgment: Set

... ....1... = Push: Set

... .....0... = Reset: Not set

... .....0... = Syn: Not set

... .....0... = Fin: Not set

La taille de la fenêtre est de 255.

j)

Frame 31: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0

Ethernet II, Src: Vmware\_b4:fb:04 (00:0c:29:b4:fb:04), Dst: Vmware\_7d:18:1a (00:0c:29:7d:18:1a)

Internet Protocol Version 4, Src: 192.168.100.1 (192.168.100.1), Dst: 192.168.100.2 (192.168.100.2)

Transmission Control Protocol, Src Port: 49681 (49681), Dst Port: 21 (21), Seq: 39, Ack: 273, Len: 20

Source Port: 49681 (49681)

Destination Port: 21 (21)

[Stream index: 0]

[TCP Segment Len: 20]

Sequence number: 39 (relative sequence number)

[Next sequence number: 59 (relative sequence number)]

Acknowledgment number: 273 (relative ack number)

Header Length: 20 bytes

... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)

Window size value: 255

[Calculated window size: 65280]

[Window size scaling factor: 256]

Checksum: 0x3a88 [validation disabled]

Urgent pointer: 0

[SEQ/ACK analysis]

File Transfer Protocol (FTP)

Développement des « flags »

... 0000 0001 1000 = Flags: 0x018 (PSH, ACK)

000. .... = Reserved: Not set

...0 .... = Nonce: Not set

... 0... = Congestion window Reduced (cwr): Not set

... .0... = ECN-Echo: Not set

... ..0... = Urgent: Not set

... ...1... = Acknowledgment: Set

... ....1... = Push: Set

... .....0... = Reset: Not set

... .....0... = Syn: Not set

... .....0... = Fin: Not set

Il a un accusé de réception (ACK).

## 1.1.3 : Trame numéro 2 :

k)

| No.  | Time       | Source          | Destination     | Protocol | Length | Info  |
|------|------------|-----------------|-----------------|----------|--------|---|
| 3969 | 140.319044 | 192.168.229.250 | 192.168.224.57  | DNS      | 72     | Standard query 0x4afe A www.clown.fr                              |
| 3971 | 140.338887 | 192.168.229.250 | 192.168.224.58  | DNS      | 72     | Standard query 0x4afe A www.clown.fr                              |
| 3973 | 140.543070 | 192.168.224.58  | 192.168.229.250 | DNS      | 148    | Standard query response 0x4afe A www.clown.fr CNAME clown.fr A... |

> Frame 3969: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0

> Ethernet II, Src: Dell\_d8:26:8b (18:03:73:d8:26:8b), Dst: Vmware\_9c:b3:0d (08:50:56:9c:b3:0d)

> Internet Protocol Version 4, Src: 192.168.229.250, Dst: 192.168.224.57

> User Datagram Protocol, Src Port: 51530 (51530), Dst Port: 53 (53)

Source Port: 51530

Destination Port: 53

Length: 38

Checksum: 0x47bd [validation disabled]

[Stream index: 689]

Domain Name System query

C'est le DNS (Domain Name System)

12) UDP est un protocole léger qui n'a pas besoin de connexion, ce qui permet des résolutions DNS rapides. Les requêtes DNS sont dans la plupart des cas de petite taille et n'ont donc pas besoin de mécanismes de retransmission comme TCP.

l)



## UTILISATION DE WIRESHARK

| No.  | Time       | Source          | Destination     | Protocol | Length | Info                    |
|------|------------|-----------------|-----------------|----------|--------|-------------------------|
| 3969 | 140.319044 | 192.168.229.250 | 192.168.224.57  | DNS      | 72     | Standard query 0x4afe   |
| 3971 | 140.338887 | 192.168.229.250 | 192.168.224.58  | DNS      | 72     | Standard query 0x4afe   |
| 3973 | 140.543070 | 192.168.224.58  | 192.168.229.250 | DNS      | 148    | Standard query response |

|   |   |
|---|---|
| > | Frame 3969: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 0          |
| > | Ethernet II, Src: Dell_d8:26:8b (18:03:73:d8:26:8b), Dst: Vmware_9c:b3:0d (00:50:56:9c:b3:0d) |
| > | Internet Protocol Version 4, Src: 192.168.229.250, Dst: 192.168.224.57                        |
| > | User Datagram Protocol, Src Port: 51530 (51530), Dst Port: 53 (53)                            |
|   | Source Port: 51530  |
|   | Destination Port: 53  |
|   | Length: 38  |
| > | Checksum: 0x47bd [validation disabled]  |
|   | [Stream index: 689]   |
| > | Domain Name System (query)  |

C'est le port 51530, un port éphémère qui est utilisé pour établir une connexion éphémère entre deux postes. Il est libre après la connexion terminée et peut être attribué à d'autres connexions.

m) la longueur indiquée dans l'en-tête UDP correspond à la taille totale du segment UDP en octets. Length indique le nombre d'octets à lire et indique la fin du segment UDP.

## 2 Prise en main de Wireshark :

Wireshark est un outil d'analyse de trafic réseau permettant d'observer, de filtrer et d'interpréter en temps réel les données circulant sur un réseau. Pour un administrateur réseau, son utilisation consiste principalement à diagnostiquer des problèmes de communication, identifier les causes de pannes ou de ralentissements, et vérifier la conformité des protocoles employés. En capturant et décomposant les paquets, Wireshark offre une visibilité granulaire sur les échanges, aidant ainsi à détecter d'éventuelles failles de sécurité, à optimiser les performances et à assurer une meilleure maîtrise de l'infrastructure informatique.

a)

|   |  |
|---|--|
| <pre> Carte Ethernet vEthernet (Default Switch) :  Suffixe DNS propre à la connexion. . . . : Description. . . . . : Hyper-V Virtual Ethernet Adapter Adresse physique . . . . . : 00-15-5D-B9-10-53 DHCP activé. . . . . : Non Configuration automatique activée. . . . : Oui Adresse IPv6 de liaison locale. . . . . : fe80::be1d:29eb:a3ca:83b4%17(préféré) Adresse IPv4. . . . . : 172.19.48.1(préféré) Masque de sous-réseau. . . . . : 255.255.240.0 Passerelle par défaut. . . . . : IAID DHCPv6 . . . . . : 285218141 DUID de client DHCPv6. . . . . : 00-01-00-01-2C-91-20-A6-E0-73-E7-B2-64-D7 NetBIOS sur Tcpip. . . . . : Activé </pre> | <p>L'adresse physique est 00-15-5D-B9-10-53 et l'adresse IP est 172.19.48.1.</p> |
|---|--|

b) L'adresse IP de ma voisine (Shayma) est 172.31.1.106.

### 1.4 : Analyse de trame :

#### 1.1.4 : Etude de la trame 1 :

a)

Champs de la trame Ethernet :

|                        |                   |  |
|------------------------|-------------------|--|
| Adresse Ethernet (MAC) | ff:ff:ff:ff:ff:ff | > Frame 1: 42 bytes on wire (336 bits), 42 bytes captured  |
| Destination            |                   | > Ethernet II, Src: 09:ab:14:d8:05:48 (09:ab:14:d8:05:48), |
|                        |                   | > Destination: Broadcast (ff:ff:ff:ff:ff:ff)               |
|                        |                   | .....1. .... = LG bit: Locally adm                         |
|                        |                   | .....1. .... = IG bit: Group addre                         |
|                        |                   | > Source: 09:ab:14:d8:05:48 (09:ab:14:d8:05:48)            |
|                        |                   | Type: ARP (0x0806)   |
|                        |                   | [Stream index: 0]  |
|                        |                   | > Address Resolution Protocol (request)                    |


## UTILISATION DE WIRESHARK

|                         |   |   |
|-------------------------|---|---|
| Adresse Ethernet Source | 09 :ab :14 :d8 :05 :48  | <pre> &gt; Frame 1: 42 bytes on wire (336 bits), 42 bytes captured   Ethernet II, Src: 09:ab:14:d8:05:48 (09:ab:14:d8:05:48),     Destination: Broadcast (ff:ff:ff:ff:ff:ff)       ...1. .... = LG bit: Locally ad       ...1. .... = IG bit: Group addr     &gt; Source: 09:ab:14:d8:05:48 (09:ab:14:d8:05:48)       Type: ARP (0x0806)       [Stream index: 0]     &gt; Address Resolution Protocol (request)           </pre>  |
| EtherType               | ARP   | <pre> &gt; Frame 1: 42 bytes on wire (336 bits), 42 byte:   Encapsulation type: Ethernet (1)   Arrival Time: Mar 30, 2012 09:24:06.000000   UTC Arrival Time: Mar 30, 2012 07:24:06.00   Epoch Arrival Time: 1333092246.000000000   [Time shift for this packet: 0.000000000 s]   [Time delta from previous captured frame: {   [Time delta from previous displayed frame:   [Time since reference or first frame: 0.00   Frame Number: 1   Frame Length: 42 bytes (336 bits)   Capture Length: 42 bytes (336 bits)   [Frame is marked: False]   [Frame is ignored: False]   [Protocols in frame: eth:ethertype:arp]   [Coloring Rule Name: ARP]   [Coloring Rule String: arp]           </pre> |
| Données                 | Il recherche l'adresse MAC de l'appareil qui a 125.18.110.3 comme adresse IP. Il demande de répondre à son adresse IP | <pre> Info Who has 125.18.110.3? Tell 125.5.48.10  Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00) Target IP address: 125.18.110.3           </pre>   |

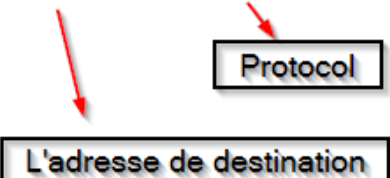
L'adresse MAC de destination est une adresse broadcast, destinée à toutes les machines du réseau pour soit découvrir des appareils sur ce réseau (ARP) soit demander des requêtes DHCP (pour obtenir une adresse IP). Ici c'est une requête ARP.

Les informations sont trouvables dans la zone 1 comme dans la zone 2 (qui donne tous les détails de la trame) :

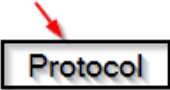
| Source            | Destination | Protocol | Length | Info                                   |
|-------------------|-------------|----------|--------|--|
| 09:ab:14:d8:05:48 | Broadcast   | ARP      | 42     | Who has 125.18.110.3? Tell 125.5.48.10 |



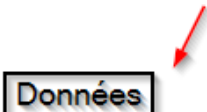
**L'adresse source**



**L'adresse de destination**



**Protocol**



**Données**

On voit une trame ARP avec une requête demandant qui possède l'adresse IP 125.18.110.3. L'adresse source 09: ab:14:d8:05:48 envoie une requête en broadcast. Cela signifie que l'hôte veut connaître l'adresse MAC correspondant à cette IP. Le protocole ARP est utilisé ici pour résoudre l'adresse IP en une adresse MAC. Cette opération est essentielle pour permettre la communication au niveau de la couche 2 (liaison).

Champs du datagramme ARP :

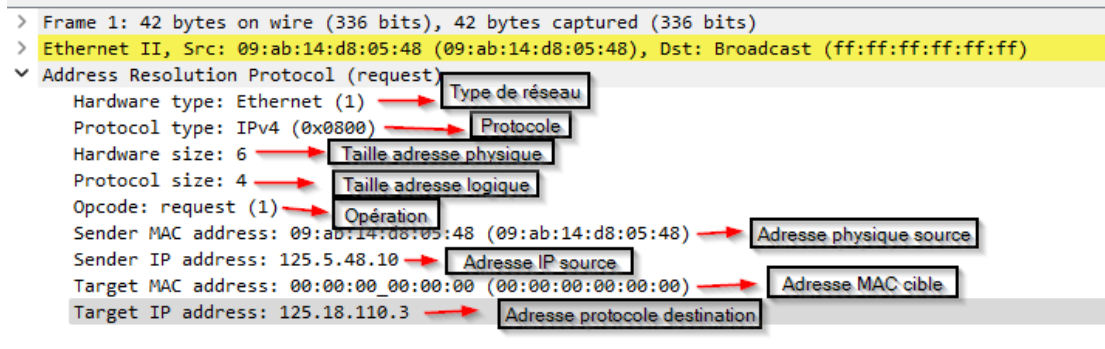
|                         |          |
|-------------------------|----------|
| Type de réseau          | Ethernet |
| Protocole               | IPv4     |
| Taille adresse physique | 6        |
| Taille adresse logique  | 4        |



## UTILISATION DE WIRESHARK

|                               |                        |
|-------------------------------|------------------------|
| Opération                     | 1 (ARP)                |
| Adresse physique source       | 09 :ab :14 :d8 :05 :48 |
| Adresse IP source             | 125.5.48.10            |
| Adresse MAC cible             | 00:00:00:00:00:00      |
| Adresse protocole destination | 125.18.110.3           |

Explication :



Cette capture montre une analyse détaillée de la requête ARP. On identifie le type de réseau (Ethernet), le protocole (IPv4), et les adresses physiques et IP. L'adresse MAC source 09:ab:14:d8:05:48 et l'adresse IP source 125.5.48.10 sont les identifiants de l'émetteur. L'adresse cible MAC est 00:00:00:00:00:00, car l'émetteur ne la connaît pas encore. L'adresse IP cible 125.18.110.3 est celle recherchée. Cette analyse permet de comprendre le fonctionnement de la résolution d'adresse pour établir une communication locale.

b) L'objet du message est de trouver l'adresse MAC et l'emplacement du poste 125.18.110.3. Le protocole IPv4 est cohérent car l'ARP fait le lien entre une adresse MAC et une adresse IP. IPv4 achemine les paquets avec les adresses IPv4, la trame contient l'adresse IP du destinataire, il va donc s'en servir pour acheminer les paquets vers le destinataire.

Vérification :

|                               |                        |                                       |
|-------------------------------|------------------------|---------------------------------------|
| Type de réseau                | Ethernet               | Cohérent                              |
| Protocole                     | IPv4                   | Cohérent                              |
| Taille adresse physique       | 6                      | Longueur standard                     |
| Taille adresse logique        | 4                      | Longueur standard                     |
| Opération                     | 1 (ARP)                | Cohérent                              |
| Adresse physique source       | 09 :ab :14 :d8 :05 :48 | Cohérent                              |
| Adresse IP source             | 125.5.48.10            | Cohérent                              |
| Adresse MAC cible             | 00:00:00:00:00:00      | Inconnu mais c'est le but de la trame |
| Adresse protocole destination | 125.18.110.3           | Adresse recherchée                    |

c) Une trame a une taille minimale de 64 octets mais la trame 1 fait 42 octets. Il manque le préambule (7 octets), le SFD (1 octets) et le FCS (4 octets) car ils sont générés automatiquement et ne sont pas capturés par Wireshark donc c'est normal.

## UTILISATION DE WIRESHARK

Le padding manque aussi, il permet à la trame d'atteindre 64 octets en rajoutant. Il en faut donc 10 octets en plus.

La norme IEEE 802.3 exige que les trames fassent au minimum 64 octets pour qu'elles soient traitées et reconnues correctement.

SFD (Start Frame Delimiter) marque le début effectif des données utiles d'une trame Ethernet.

FCS (Frame Check Sequence) détecte les erreurs qui peuvent survenir lors de la transmission des données sur le réseau.

### 1.1.5 : Etude de la trame 2 :

a) Champs de la trame :

|                                    |   |
|------------------------------------|---|
| Adresse Ethernet (MAC) Destination | 00:0f:1f:13:34:9a   |
| Adresse Ethernet Source            | 00 :01 :30 :4a :38 :00  |
| EtherType                          | Internet Control Message Protocol (ICMP).   |
| Données                            | LE ping a atteint son destinataire et la réponse est revenue correctement à son expéditeur. |

The image shows a Wireshark capture of an ICMP Echo (ping) reply. The packet list pane at the top shows packet 1 at time 0.000000, source 139.124.187.4, destination 172.16.203.109, protocol ICMP, length 98. The packet details pane shows the following layers:

- Ethernet II, Src: ExtremeNetwo\_4a:38:00 (00:01:30:4a:38:00), Dst: Dell\_13:34:9a (00:0f:1f:13:34:9a)
- Internet Protocol Version 4, Src: 139.124.187.4, Dst: 172.16.203.109
- Internet Control Message Protocol

The packet bytes pane shows the raw data in hexadecimal and ASCII. The data is as follows:

```

0000  00 0f 1f 13 34 9a 00 01 30 4a 38 00 00 00 00 00
0010  00 54 9c 1e 00 00 00 00 00 00 00 00 00 00 00
0020  cb 6d 00 00 f7 2b ea 30 00 02 02 00 08 09 0a 0b
0030  0c 0d 0e 0f 16 17 18 19 1a 1b 1c 1d 1e 1f
0040  26 27 28 29 2a 2b 2c 2d 2e 2f 36 37
0050
0060

```

Ici, il s'agit d'une trame ICMP, spécifiquement une réponse à un ping (Echo (ping) reply).

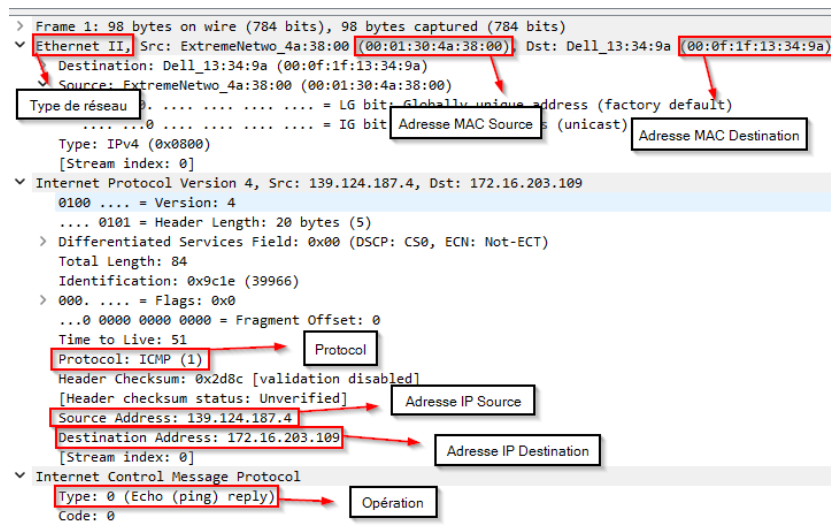
L'adresse IP source est 139.124.187.4 et l'adresse destination 172.16.203.109. Au niveau de la couche 2, l'adresse MAC source est 00:01:30:4a:38:00 et l'adresse MAC destination est 00:0f:1f:13:34:9a. Le protocole ICMP permet de diagnostiquer les problèmes de connectivité. Cette trame montre que le ping a réussi et que la réponse a bien été reçue.

Champs du datagramme :

## UTILISATION DE WIRESHARK

|                               |                              |
|-------------------------------|------------------------------|
| Type de réseau                | Ethernet II                  |
| Protocole                     | ICMP                         |
| Taille adresse physique       | Inconnu, il est de 6 octets  |
| Taille adresse logique        | Inconnu, il est de 4 octets. |
| Opération                     | Echo Reply (ping)            |
| Adresse physique source       | 00 :01 :30 :4a :38 :00       |
| Adresse IP source             | 139.124.187.4                |
| Adresse MAC cible             | 00 :0f :1f :13f34 :9a        |
| Adresse protocole destination | 172.16.203.109               |

Cette capture détaille encore la trame ICMP vue précédemment. On voit les adresses MAC source 00:01:30:4a:38:00 et destination 00:0f:1f:13:34:9a au niveau Ethernet II. L'adresse IP source 139.124.187.4 et l'adresse IP destination 172.16.203.109 sont clairement identifiées. Le protocole ICMP est utilisé avec le type 0, ce qui correspond à une réponse Echo (ping) reply. Cette capture permet de vérifier la connectivité réseau en confirmant que le paquet ICMP a bien été transmis et reçu.



Sur cette capture, nous avons une analyse du protocole Ethernet et IP. L'adresse MAC source est celle de "ExtremeNetwo" et celle de destination est "Dell", ce qui correspond à des équipements sur le réseau. En dessous, on remarque le protocole utilisé qui est ICMP (protocole de contrôle de messages Internet). L'adresse IP source et destination sont visibles, de même que l'opération effectuée, qui est un "Echo request" (ping). Cette capture permet de diagnostiquer un test de connectivité entre deux équipements à travers un réseau, en utilisant ICMP pour vérifier si un appareil répond à une requête de ping.

Les tailles d'adresses MAC et IP sont inconnues car la trame est mal configurée.

## UTILISATION DE WIRESHARK

```

▼ Destination: Dell_13:34:9a (00:0f:1f:13:34:9a)
  .... ..0. .... = LG bit: Globally unique address (factory default)
  .... ..0. .... = IG bit: Individual address (unicast)
▼ Source: ExtremeNetwo_4a:38:00 (00:01:30:4a:38:00)
  .... ..0. .... = LG bit: Globally unique address (factory default)
  .... ..0. .... = IG bit: Individual address (unicast)

```

Wireshark n'arrive pas à analyser correctement les adresses IP et MAC.

```

▼ Data (56 bytes)
  Data: c31f60470e37020008090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f202122232425262728292a2...
  [Length: 56]

```

De plus il y a des données (data) dans une trame de connectivité réseau, ce qui n'a rien à faire là. La trame essaie peut-être de faire passer des informations pour injecter du code malveillant. M Drogue n'aurait aucun objectif à infecter les postes de son réseau, celle-là est donc sans danger.

b) L'adresse MAC source change à chaque saut. Pour arriver à destination, elle a forcément dû passer par un routeur ou un switch. C'est donc l'adresse MAC de l'appareil qui a transmis le paquet à l'hôte destinataire et non celle de l'hôte source.

## 1.5 : Capturer et analyser les données ICMP locales :

### 1.1.6 : Examen des données capturées :

J'ai ping Shayma et je vois une série de requêtes ICMP entre deux adresses IP : 172.31.1.67 (source) et 172.31.1.106 (destination). Ces échanges montrent une communication de type ping pour vérifier la connectivité entre ces deux hôtes. La source et la destination alternent, indiquant que des requêtes et réponses ICMP se succèdent correctement. Cela permet de diagnostiquer le bon fonctionnement du réseau entre ces deux machines.

| No. | Time        | Source       | Destination  | Protoc |
|-----|-------------|--------------|--------------|--------|
| 1   | 36 1.836088 | 172.31.1.67  | 172.31.1.106 | ICMP   |
| 2   | 37 1.836531 | 172.31.1.106 | 172.31.1.67  | ICMP   |
| 3   | 47 2.853994 | 172.31.1.67  | 172.31.1.106 | ICMP   |
| 4   | 48 2.854655 | 172.31.1.106 | 172.31.1.67  | ICMP   |
| 5   | 82 3.868220 | 172.31.1.67  | 172.31.1.106 | ICMP   |
| 83  | 3.868744    | 172.31.1.106 | 172.31.1.67  | ICMP   |
| 96  | 4.875256    | 172.31.1.67  | 172.31.1.106 | ICMP   |
| 97  | 4.875884    | 172.31.1.106 | 172.31.1.67  | ICMP   |

```

> Frame 82: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: HP_b2:64:d7 (e0:73:e7:b2:64:d7), Dst: HP_b2:66:5f
▼ Internet Protocol Version 4, Src: 172.31.1.67, Dst: 172.31.1.106

```

Page

```

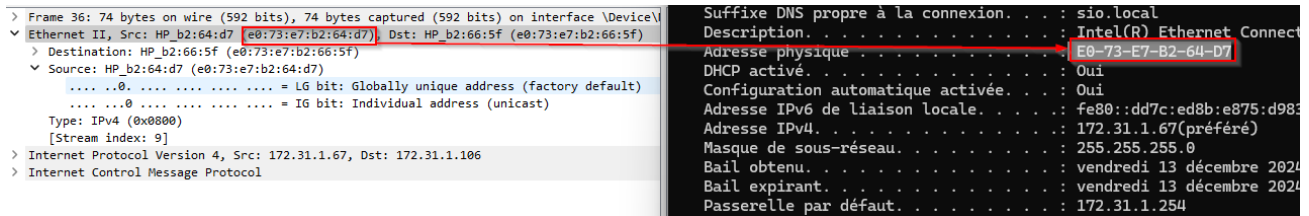
Carte Ethernet Ethernet :
  Suffixe DNS propre à la connexion. . . : sio.local
  Adresse IPv6 de liaison locale. . . . : fe80::dd7c:ed8b:e875:d983%19
  Adresse IPv4. . . . . : 172.31.1.67
  Masque de sous-réseau. . . . . : 255.255.255.0
  Passerelle par défaut. . . . . : 172.31.1.254

```

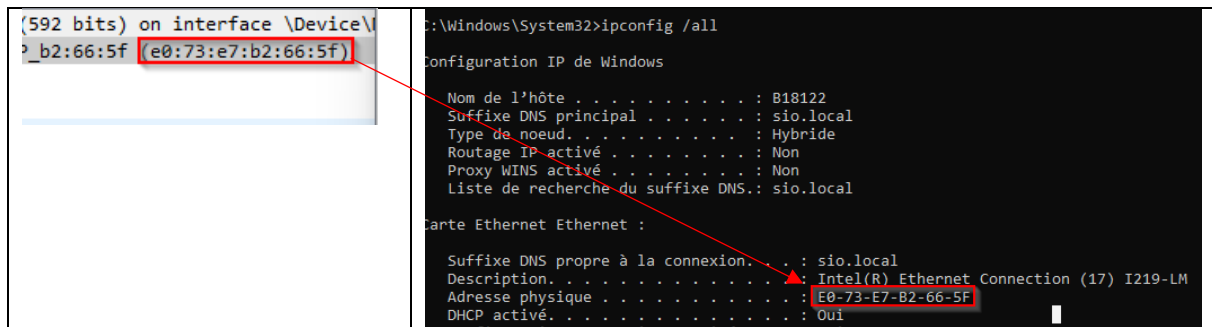
## UTILISATION DE WIRESHARK

Adresse IP de Shayma :172.31.1.106

a)



Sur cette capture, on voit bien que l'adresse MAC source e0:73:e7:b2:64:d7 correspond à mon adresse MAC source. Cela est logique puisque la requête ICMP (ping) provient de ma machine. Wireshark capture et analyse cette trame en direct depuis mon interface réseau, ce qui explique pourquoi l'adresse MAC source affichée est la mienne. Cela confirme que le ping est émis correctement depuis mon poste vers l'adresse 172.31.1.106.



L'adresse MAC e0:73:e7:b2:66:5f correspond à l'interface réseau de la machine 172.31.1.106. Cette information est vérifiée via la commande ipconfig /all,

## 1.6 : Capturer et analyser les données ICMP locales :

### 1.1.7 : Capture les données de l'interface :

```
C:\Users\uti029>ping ciscomadesimple.be

Envoi d'une requête 'ping' sur ciscomadesimple.be [145.239.37.162] avec 32 octets de données :
Réponse de 145.239.37.162 : octets=32 temps=12 ms TTL=54
Réponse de 145.239.37.162 : octets=32 temps=12 ms TTL=54
Réponse de 145.239.37.162 : octets=32 temps=13 ms TTL=54
Réponse de 145.239.37.162 : octets=32 temps=12 ms TTL=54

Statistiques Ping pour 145.239.37.162:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 12ms, Maximum = 13ms, Moyenne = 12ms

C:\Users\uti029>ping www.developpez.com

Envoi d'une requête 'ping' sur developpez.com [51.210.99.219] avec 32 octets de données :
Réponse de 51.210.99.219 : octets=32 temps=12 ms TTL=53
Réponse de 51.210.99.219 : octets=32 temps=12 ms TTL=53
Réponse de 51.210.99.219 : octets=32 temps=12 ms TTL=53
Réponse de 51.210.99.219 : octets=32 temps=12 ms TTL=53

Statistiques Ping pour 51.210.99.219:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 12ms, Maximum = 12ms, Moyenne = 12ms

C:\Users\uti029>ping www.reseaucerta.org

Envoi d'une requête 'ping' sur gravelines.reseaucerta.org [151.80.233.222] avec 32 octets de données :
Réponse de 151.80.233.222 : octets=32 temps=12 ms TTL=51
Réponse de 151.80.233.222 : octets=32 temps=86 ms TTL=51
Réponse de 151.80.233.222 : octets=32 temps=12 ms TTL=51
Réponse de 151.80.233.222 : octets=32 temps=12 ms TTL=51

Statistiques Ping pour 151.80.233.222:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 12ms, Maximum = 86ms, Moyenne = 30ms
```

On voit une commande ping. Les adresses IP et les temps de réponse sont affichés, avec une perte de paquets de 0%. Les adresses IP 145.239.37.162, 51.210.99.219, 151.80.233.222 correspondent aux serveurs ciscomadesimple.be, [www.developpez.com](http://www.developpez.com) et [www.reseaucerta.org](http://www.reseaucerta.org), ce qui montre une vérification de la connectivité vers des ressources extérieures à travers le réseau.



## UTILISATION DE WIRESHARK

## 1.1.8 : Examen et analyse des données à partir des hôtes distants :

| Emplacement | Adresse IP     | Adresse MAC             |
|-------------|----------------|-------------------------|
| 1           | 145.239.37.162 | 00 :0c :29 :b8 :25 :53  |
| 2           | 51.210.99.219  | 00 :0 c :29 :b8 :25 :53 |
| 3           | 151.80.233.222 | 00 :0 c :29 :b8 :25 :53 |

b) L'adresse MAC est toujours la même car c'est celle de la passerelle par défaut. Lorsqu'on envoie un ping à des adresses IP externes, le trafic doit passer par le routeur pour atteindre ces destinations. Le routeur relaie ensuite le paquet vers sa destination finale. C'est donc l'adresse MAC du routeur qui est utilisée comme adresse de destination dans chaque trame Ethernet.

## 1.1.9 : Analyse les données ICMPv6 locales

a) ICMPv6 est utilisé pour diagnostiquer et signaler les erreurs dans les réseaux IPv6

Frame 12: 90 bytes on wire (720 bits), 90 bytes captured (720 bits) on interface \Device\NPF\_{F412D550-3F7C-4244-9507-5E06D273C098}, id 0

Ethernet II, Src: HP\_b2:65:74:e0:73:e7:b2:65:74, Dst: IPv6mcast\_16 (33:33:00:00:00:16)

> Destination: IPv6mcast\_16 (33:33:00:00:00:16)

> Source: HP\_b2:65:74 (e0:73:e7:b2:65:74)

Type: IPv6 (0x86dd)

[Stream index: 4]

Internet Protocol Version 6, Src: fe80::bc3c:f584:7e72:9b48, Dst: ff02::16

0110 .... = Version: 6

.... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0, ECN: Not-ECT)

.... 0000 0000 0000 .... = Label: 0x000000

Payload Length: 36

Next Header: IPv6 Hop-by-Hop Option (0)

Hop Limit: 1

> Source Address: fe80::bc3c:f584:7e72:9b48

> Destination Address: ff02::16

[Stream index: 0]

> IPv6 Hop-by-Hop Option

Internet Control Message Protocol v6

Type: Multicast Listener Report Message v2 (143)

Code: 0

Checksum: 0xa5ba

[Checksum Status: Good]

Reserved: 0000

Number of Multicast Address Records: 1

> Multicast Address Record Changed to include: ff02::1:3

Adresse MAC source

Adresse MAC de destination

Adresse IP source (link-local)

Adresse IP destination (multicast)

Transport des informations supplémentaires

Gérer des multicasts spécifique

Erreur

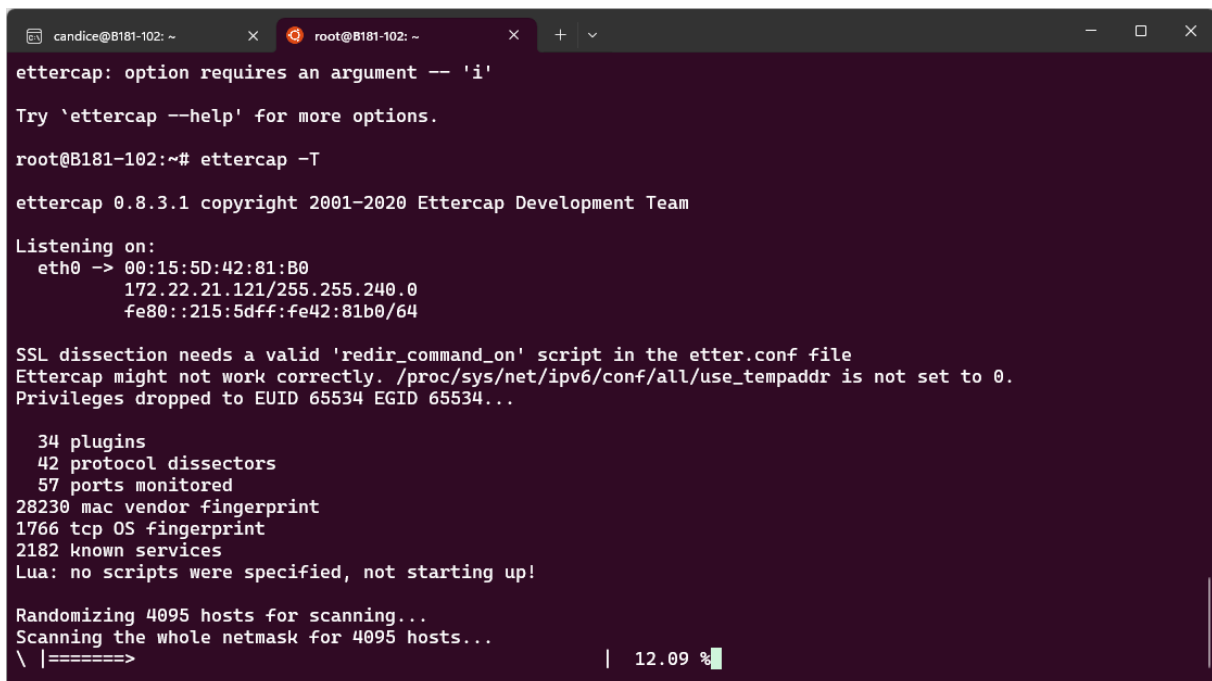
On peut observer l'adresse MAC source et de destination associée à des adresses IP en IPv6. Ici, le protocole ICMPv6 est utilisé pour le rapport multicast, ce qui est typiquement utilisé pour l'échange d'informations de groupe, comme les communications en temps réel ou le multicast vidéo. Le paquet montre un type "Multicast Listener Report Message v2", et les informations de transport sont également visibles dans l'analyse.

## UTILISATION DE WIRESHARK

b) Le préfixe link-local des adresses IPv6 est FE80 ::/10 car il a été défini par la RFC 4291 pour identifier les adresses link-local. Elles permettent aux appareils de communiquer dès qu'ils sont connectés à un réseau sans avoir besoin d'une configuration manuelle ou d'un serveur d'adresse.

```
[Stream index: 4]
▼ Internet Protocol Version 6, Src: fe80::bc3c:f584:7e72:9b48, Dst: ff02::16
    0110 .... = Version: 6
    > .... 0000 0000 .... = Traffic Class: 0x00 (DSCP: CS0)
```

## 1.7 : Ettercap analyse des trames comme un attaquant simulation MAN in the middle



```
candice@B181-102: ~
root@B181-102: ~
ettercap: option requires an argument -- 'i'
Try `ettercap --help' for more options.
root@B181-102:~# ettercap -T
ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Listening on:
  eth0 -> 00:15:5D:42:81:B0
         172.22.21.121/255.255.240.0
         fe80::215:5dff:fe42:81b0/64

SSL dissection needs a valid 'redir_command_on' script in the etter.conf file
Ettercap might not work correctly. /proc/sys/net/ipv6/conf/all/use_tempaddr is not set to 0.
Privileges dropped to EUID 65534 EGID 65534...

 34 plugins
 42 protocol dissectors
 57 ports monitored
28230 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!

Randomizing 4095 hosts for scanning...
Scanning the whole netmask for 4095 hosts...
\ |=====> | 12.09 %
```

Dans cette capture, on utilise ettercap pour scanner le réseau. Le scanne commence avec une plage d'adresses IP (172.22.21.121) et génère un message d'attente pour 4095 hôtes à analyser. Cette opération est utilisée pour détecter des dispositifs et services sur un réseau local.

## UTILISATION DE WIRESHARK

```

candice@B181-102: ~
root@B181-102: ~

Fri Dec 13 13:42:48 2024 [622342]
UDP 185.125.190.56:123 --> 172.22.21.121:51061 | (48)
| |=====| 57.92 %

Fri Dec 13 13:42:59 2024 [791551]
UDP 172.22.16.1:5353 --> 224.0.0.251:5353 | (40)
....._googlecast._tcp.local.....

Fri Dec 13 13:42:59 2024 [791664]
UDP fe80::771f:b8bd:6c99:a415:5353 --> ff02::fb:5353 | (40)
- |=====| 60.12 %

Fri Dec 13 13:43:00 2024 [721363]
UDP 172.22.16.1:5353 --> 224.0.0.251:5353 | (40)
....._googlecast._tcp.local.....

Fri Dec 13 13:43:00 2024 [721409]
UDP fe80::771f:b8bd:6c99:a415:5353 --> ff02::fb:5353 | (40)
| |=====| 64.57 %

Fri Dec 13 13:43:02 2024 [558349]
UDP 172.22.16.1:5353 --> 224.0.0.251:5353 | (40)
....._googlecast._tcp.local.....

Fri Dec 13 13:43:02 2024 [558395]
UDP fe80::771f:b8bd:6c99:a415:5353 --> ff02::fb:5353 | (40)
- |=====| 65.69 %

```

Dans cette capture, ettercap montre le trafic réseau en temps réel qu'il surveille. Les paquets UDP envoyés entre des adresses IP sont capturés. Cela indique que l'outil capture des requêtes multicast sur le réseau. Il est aussi possible de voir des pourcentages indiquant l'avancement de l'analyse des paquets. Cette analyse aide à surveiller la communication sur le réseau et à identifier des services actifs.

```

root@B181-102:~# ettercap -T -I

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

List of available Network Interfaces:

eth0    eth0
lo      Local Loopback
bluetooth-monitor  Bluetooth Linux Monitor

root@B181-102:~#

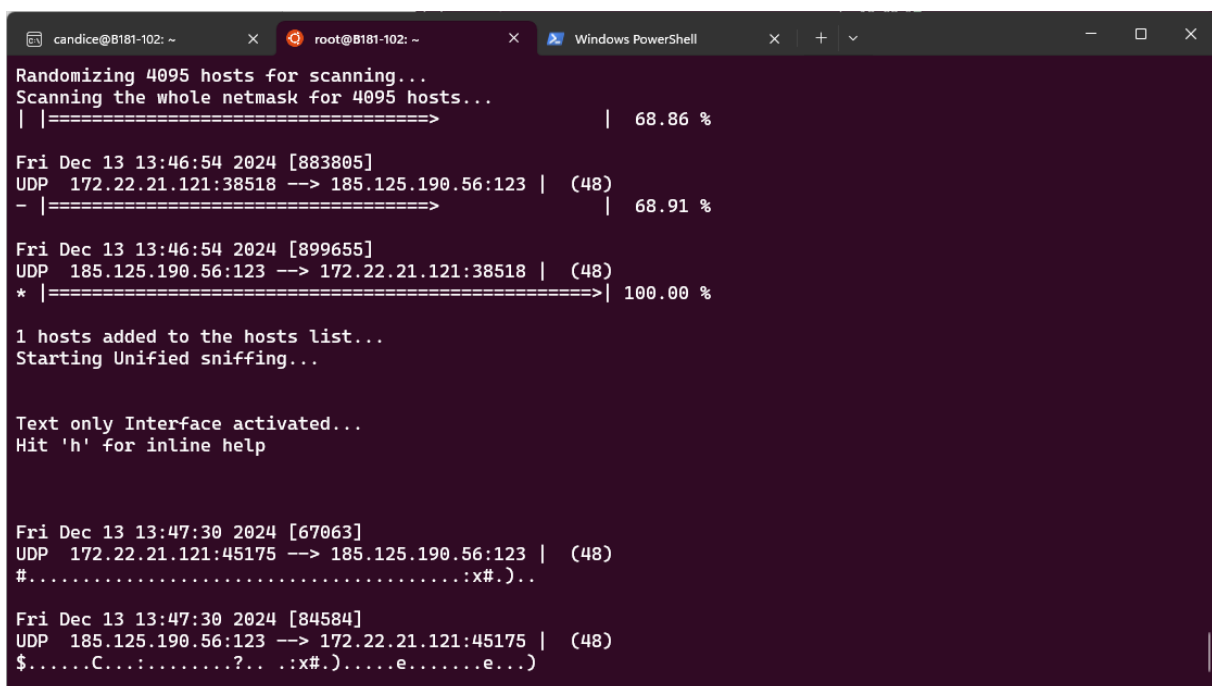
```

Dans cette troisième image, **ettercap** est à nouveau exécuté avec l'option -T -I pour lister les interfaces réseau disponibles sur le système. On voit que l'interface eth0 est listée comme étant disponible pour l'analyse. Cela montre comment l'outil permet à l'utilisateur de sélectionner et interagir avec différentes interfaces réseau.

## UTILISATION DE WIRESHARK

Lancer une attaque ARP Spoofing entre le serveur DHCP (172.31.1.4) et une machine cliente (par exemple, 172.31.1.59) :

- -T : Mode texte.
- -i eth0 : Utiliser l'interface eth0.
- -M arp:remote : Attaque ARP MITM.
- /172.31.1.4/ : Première cible (le serveur DHCP).
- /172.31.1.59/ : Deuxième cible (la machine cliente).



```

candice@B181-102: ~
root@B181-102: ~
Windows PowerShell

Randomizing 4095 hosts for scanning...
Scanning the whole netmask for 4095 hosts...
|=====| 68.86 %

Fri Dec 13 13:46:54 2024 [883805]
UDP 172.22.21.121:38518 --> 185.125.190.56:123 | (48)
-|=====| 68.91 %

Fri Dec 13 13:46:54 2024 [899655]
UDP 185.125.190.56:123 --> 172.22.21.121:38518 | (48)
*|=====| 100.00 %

1 hosts added to the hosts list...
Starting Unified sniffing...

Text only Interface activated...
Hit 'h' for inline help

Fri Dec 13 13:47:30 2024 [67063]
UDP 172.22.21.121:45175 --> 185.125.190.56:123 | (48)
#.....:x#.)..

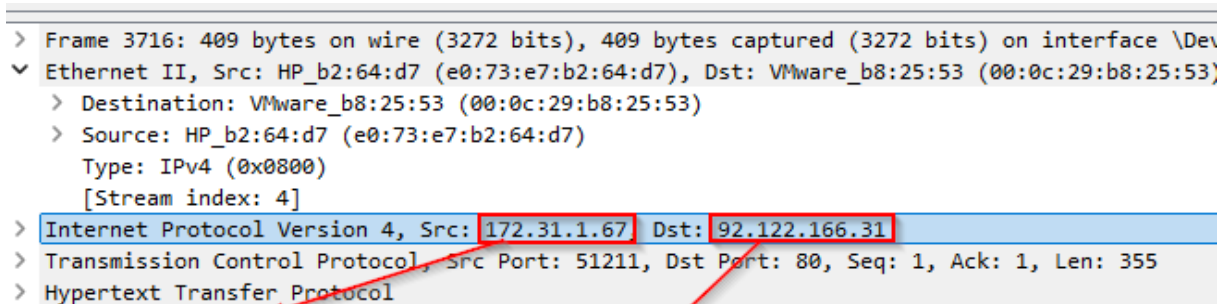
Fri Dec 13 13:47:30 2024 [84584]
UDP 185.125.190.56:123 --> 172.22.21.121:45175 | (48)
$.C...?..:x#.)...e...e...

```

Ici on a remplacé le dhcp originel par le nôtre nous sommes devenue le dhcp principale en remplaçant 172.31.1.4.

### 3 Capture de trame liées à HTTP :

a)



Mon adresse IP

L'adresse IP du site internet

b) Dans Statistiques puis conversation dans la barre de menu en haut, on voit apparaître les statistiques de la trame :

Wireshark · Conversations · Ethernet

Conversation Settings

☐ Résolution de nom

☐ Heure de début absolue

| Ethernet · 1      |                   | IPv4 · 1 | IPv6   | TCP · 1    | UDP |
|-------------------|-------------------|----------|--------|------------|-----|
| Adresse A         | Adresse B         | Paquets  | Octets | ID de flux | Pa  |
| e0:73:e7:b2:64:d7 | 00:0c:29:b8:25:53 | 4        | 3 ko   | 4          |     |

Il y a donc 4 paquets qui sont échangés.

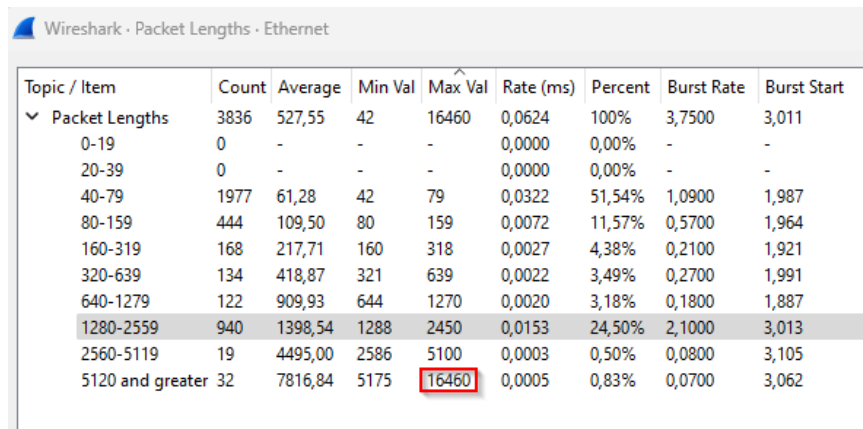
c) Il y a plusieurs trames car les données sont trop lourdes pour une seule trame, elles sont donc fragmentées en plusieurs trames.

Le protocole TCP a besoin de plusieurs trames, une de SYN (demande de connexion), une SYN-ACK (accepte de la demande) et une ACK (confirmation de l'acceptation).

Des paquets peuvent être perdus, le protocole demande de les re-envoyer.

## UTILISATION DE WIRESHARK

d) On voit dans statistiques puis taille maximale des trames reçues dans le menu en haut :



Wireshark - Packet Lengths - Ethernet

| Topic / Item     | Count | Average | Min Val | Max Val | Rate (ms) | Percent | Burst Rate | Burst Start |
|------------------|-------|---------|---------|---------|-----------|---------|------------|-------------|
| ▼ Packet Lengths | 3836  | 527,55  | 42      | 16460   | 0,0624    | 100%    | 3,7500     | 3,011       |
| 0-19             | 0     | -       | -       | -       | 0,0000    | 0,00%   | -          | -           |
| 20-39            | 0     | -       | -       | -       | 0,0000    | 0,00%   | -          | -           |
| 40-79            | 1977  | 61,28   | 42      | 79      | 0,0322    | 51,54%  | 1,0900     | 1,987       |
| 80-159           | 444   | 109,50  | 80      | 159     | 0,0072    | 11,57%  | 0,5700     | 1,964       |
| 160-319          | 168   | 217,71  | 160     | 318     | 0,0027    | 4,38%   | 0,2100     | 1,921       |
| 320-639          | 134   | 418,87  | 321     | 639     | 0,0022    | 3,49%   | 0,2700     | 1,991       |
| 640-1279         | 122   | 909,93  | 644     | 1270    | 0,0020    | 3,18%   | 0,1800     | 1,887       |
| 1280-2559        | 940   | 1398,54 | 1288    | 2450    | 0,0153    | 24,50%  | 2,1000     | 3,013       |
| 2560-5119        | 19    | 4495,00 | 2586    | 5100    | 0,0003    | 0,50%   | 0,0800     | 3,105       |
| 5120 and greater | 32    | 7816,84 | 5175    | 16460   | 0,0005    | 0,83%   | 0,0700     | 3,062       |

La taille maximale est de 16450.

La taille des trames est de 1518 octets pour les trames classiques et 9000 octets pour les trames jumbo, des trames pour des environnements spécifiques (centre de données). La plus grande est donc une trame jumbo.

## 1.8 : MTU

a)

```
C:\Users\uti029>ping www.google.fr -f -n 1 -l 1500
Envoi d'une requête 'ping' sur www.google.fr [142.251.143.195] avec 1500 octets de données :
Le paquet doit être fragmenté mais paramétré DF.

Statistiques Ping pour 142.251.143.195:
  Paquets : envoyés = 1, reçus = 0, perdus = 1 (perte 100%),
```

Le paramètre -f est utilisé pour activer le flag "Do Not Fragment" (DF), ce qui signifie que le paquet ne doit pas être fragmenté lors de son envoi. Le message d'erreur « Le paquet doit être fragmenté mais paramétré DF » indique que le paquet est trop volumineux pour passer dans un seul paquet IP et qu'il devrait être fragmenté, le MTU 1500 est donc trop grand pour passer, il faut le baisser.



## UTILISATION DE WIRESHARK

```
C:\Users\uti029>ping www.google.fr -f -n 1 -l 1452

Envoi d'une requête 'ping' sur www.google.fr [142.251.143.195] avec 1452 octets de données :
Réponse de 142.251.143.195 : octets=1452 temps=17 ms TTL=114

Statistiques Ping pour 142.251.143.195:
    Paquets : envoyés = 1, reçus = 1, perdus = 0 (perte 0%),
Durée approximative des boucles en millisecondes :
    Minimum = 17ms, Maximum = 17ms, Moyenne = 17ms

C:\Users\uti029>ping www.google.fr -f -n 1 -l 1453

Envoi d'une requête 'ping' sur www.google.fr [142.251.143.195] avec 1453 octets de données :
Le paquet doit être fragmenté mais paramétré DF.

Statistiques Ping pour 142.251.143.195:
    Paquets : envoyés = 1, reçus = 0, perdus = 1 (perte 100%),

C:\Users\uti029>|
```

Le paquet 1452 est plus petit que celui de 1500 octets et, en conséquence, il peut être transmis sans nécessiter de fragmentation, malgré le flag DF. Le ping réussit sans perte. Le paquet de 1453 octets ne peut pas être fragmenté en raison du paramètre DF et le MTU est trop grand, et donc une nouvelle perte de paquet se produit.

Le MTU optimal est donc 1452.

## 4 Application pratiques de Wireshark HTTPS :

10 Minute Mail - Free Anonymous Temporary email (un site sécurisé inconnu pour moi)

```
> Frame 2179: 1843 bytes on wire (14744 bits), 1843 bytes captured (14744 bits) on interface \Device\NPF_{F412D550-3F7C-4244-9507-5E06D}
> Ethernet II, Src: HP_b2:64:d7 (e0:73:e7:b2:64:d7), Dst: VMware_b8:25:53 (00:0c:29:b8:25:53)
> Internet Protocol Version 4, Src: 172.31.1.67, Dst: 162.247.243.29
> Transmission Control Protocol, Src Port: 58351, Dst Port: 443, Seq: 1, Ack: 1, Len: 1789
▼ Transport Layer Security
  ▼ TLSv1.3 Record Layer: Handshake Protocol: Client Hello
    Content Type: Handshake (22)
    Version: TLS 1.0 (0x0301)
    Length: 1784
    ▼ Handshake Protocol: Client Hello
      Handshake Type: Client Hello (1)
      Length: 1780
      > Version: TLS 1.2 (0x0303)
        Random: 252186b38b00e955438de5f35e618914d11f250ad4d24cc96b1fbe5d9a7a4688
        Session ID Length: 32
        Session ID: f92ed90074a8521a60824a5fb42abfe30dd08707b230cb95d84f0463a2d4e9c5
        Cipher Suites Length: 32
        ▼ Cipher Suites (16 suites)
          Cipher Suite: Reserved (GREASE) (0x2a2a)
          Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
          Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
          Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
          Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
          Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xccaa8)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
          Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
          Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
          Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
          Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
          Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
```

Cette capture montre le Client Hello dans un processus de handshake TLS. Le client initie la connexion avec le serveur en utilisant le protocole TLS 1.2 et inclut dans le message une liste de suites de chiffrement qu'il est prêt à accepter. Le paquet indique que le client supporte plusieurs suites de chiffrement, dont la suite de chiffrement TLS\_ECDHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384, qui est un choix populaire en raison de sa robustesse et de ses performances. Cela fait partie de l'échange qui permet d'établir une connexion sécurisée entre le client et le serveur.

## UTILISATION DE WIRESHARK

```

> Internet Protocol Version 4, Src: 13.107.21.239, Dst: 172.31.1.67
> Transmission Control Protocol, Src Port: 443, Dst Port: 58328, Seq: 5761, Ack: 1793, Len: 212
> [5 Reassembled TCP Segments (5972 bytes): #649(1440), #650(1440), #651(1440), #652(1440), #654(212)]
▼ Transport Layer Security
  ▼ TLSv1.2 Record Layer: Handshake Protocol: Multiple Handshake Messages
    Content Type: Handshake (22)
    Version: TLS 1.2 (0x0303)
    Length: 5967
    ▼ Handshake Protocol: Server Hello
      Handshake Type: Server Hello (2)
      Length: 102
      Version: TLS 1.2 (0x0303)
      Random: 6761271937720de3b4c244028537f99838aabbf17935bb81c6a163b95bba59991
      Session ID Length: 32
      Session ID: 8211000099bd5167b3bd0bb7f1f9c76e69a5ac8d71d88871102db2f45be296c6
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
      Compression Method: null (0)
      Extensions Length: 30
      > Extension: status_request (len=0)
      > Extension: session_ticket (len=0)
      > Extension: application_layer_protocol_negotiation (len=5)
      > Extension: extended_master_secret (len=0)
      > Extension: renegotiation_info (len=1)
      > Extension: server_name (len=0)
      [JA3S Fullstring: 771,49200,5-35-16-23-65281-0]
      [JA3S: 0f14538e1c9070becdad7739c67d6363]
    ▼ Handshake Protocol: Certificate
      Handshake Type: Certificate (11)
      Length: 3622
      Certificates Length: 3619
      ▼ Certificates (3619 bytes)
        Certificate Length: 2157
        > Certificate [..]: 3082086930820651a00302010202133300e258497850fc6287b76f2c000000e25849300d06092a864886f70d01010c0500305d310b3009060355040613025553311e301c060355040a1315...
          > SignedCertificate
            > algorithmIdentifier (sha384withRSAAEncryption)
              Padding: 0
              encrypted [..]: 6b0f1ed44e222955d608e1e6115c7d0b11b3c3dd314da6982235e26728c01e218169aeb7be8d993c676c040b9db9ceee9417e7f2d11e644b37dc03f0f2b3cd727f3987a238a257c9...
              Certificate Length: 1456
            > Certificate [..]: 308205ac30820494a003020102021009f96ce295555f24749eaf1e5dced49d300d06092a864886f70d01010c05003061310b300906035504061302555331153013060355040a130c446967...
          ▼ Handshake Protocol: Certificate Status

```

Dans cette capture, le serveur répond avec un message Server Hello. Le serveur confirme qu'il utilise également TLS 1.2 et a choisi une suite de chiffrement qui correspond à celle envoyée par le client. Ce message de réponse contient également un certificat numérique pour authentifier l'identité du serveur. Le certificat est vérifié pour établir une connexion sécurisée.

```

Wireshark - Suivre le flux TCP (tcp.stream eq 33) : Ethernet
...83...../...S.PA...R.V...e(... u.(.1.$q.SB...N^D..B''g... ^*T... ..+./...0...../..5.....:.....Di.....h2.....edge.microsoft.com..
...
...#...
...
...h2.http/1.1
...C\GD...A...F...X.k...[.....@... ..n...[.c'.O.h.....:Hno.....bqT-C.q...{...}]...;3..1
U... (RE.L...)<...xb... ..ab...I...mSa...<$%.&f.@...9...4)...4.y(d..B..
.CH#...[...m.#X.5[u.r.r.r.p.i.l.zh...+...]/... ..e..E.W)...!3d.....3.....C.M.....aYZ...3~ ..0.8...~.d.&{..2p@p=#c_
...X...8!6R@...-s.Z;...-6...nd.4j#D.ZP...6..a
...8S.(8.1...z...&...eU..+..+{ge...".X...d..2..%x...9..2.....h..a.zIk..9...%f.P..P.IP[...@k...S\...qT.a...S.uh9...Z...<.y...p/...t.&.t.C!.P...d9.6.Z...y".
V2.j.....)
{..L.U..M...A..lk.....CP...e...%...z..W..[f(...=.....{.@.W)...;3i3..8i ..o....Ay6..n..d...[...5.....S.....).E..I..v... "s.....Np.....W..J.L.7
".H.....hy..s ..O'3.P...<...{...+...3.(.....C.K....g
..V&...{...@...u...m...[...7q...C.../...^8i}[...@]6...0...w...$.+M...5p.....'..9f|. \..[.hmgz...].X...`U@..K.7vT....{.E.1..3.giR.....P1q*R.....~.#..).
...y...3]mc.OPQn...B.....s.....4[x.N. g...4.d..A..<R.n.z..6....E.....\S..9r..i3..ry.....68d.vC...d...C.U6..... V"...b.B.....%ce...
.T98.
..O.i (+...o.4..V.i2.(E..@0..+..h..5.....M.....&#.L. &....."TC8X;w...oo...".r.. D8[{(.k%h.....~...58T..{..T..8..TK.<.g...[c...f...&&.....t...7MB..oR9...\.
..U".G.B.M[.:K..C[410b.P1w.G.*\.(e...''r...S.(T8.....I..H.$i>.#].M..5..:.....[...ys.ER.....'.SP.
.P1N.x.i..Z...Fb.....m...6.L..@5h.....I..e.....i|Zj.3..#On6'.U.v..%...rj..h..>..n.....#Z].0.v.v.q.+<.c..I.x.{..pY.....V...X.2.v.....9.....k.....8...k.
...f.O.....+...*...

```

Ici, on voit un extrait de données capturées qui est la signature digitale TCP avec des données cryptées. Le contenu visible est une séquence de données illisibles car il est chiffré.

## 5 NetworkMiner :

NetworkMiner est un outil d'analyse réseau open-source qui analyse les hôtes et les sessions, récupère les fichiers transmis sur le réseau et visualise les certificats. Il est intuitif, analyse passivement (discrète) et extrait automatiquement les fichiers et mots de passe en clair. Il ne permet pas de capturer les données en temps réel, il faut entrer un fichier. pcap.

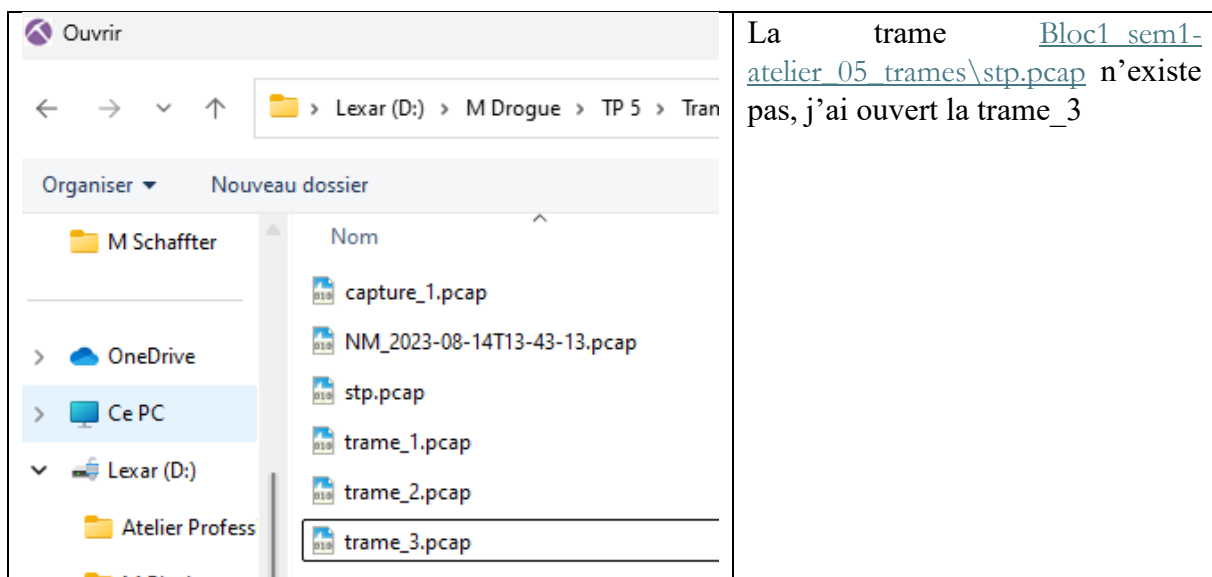
Installation :



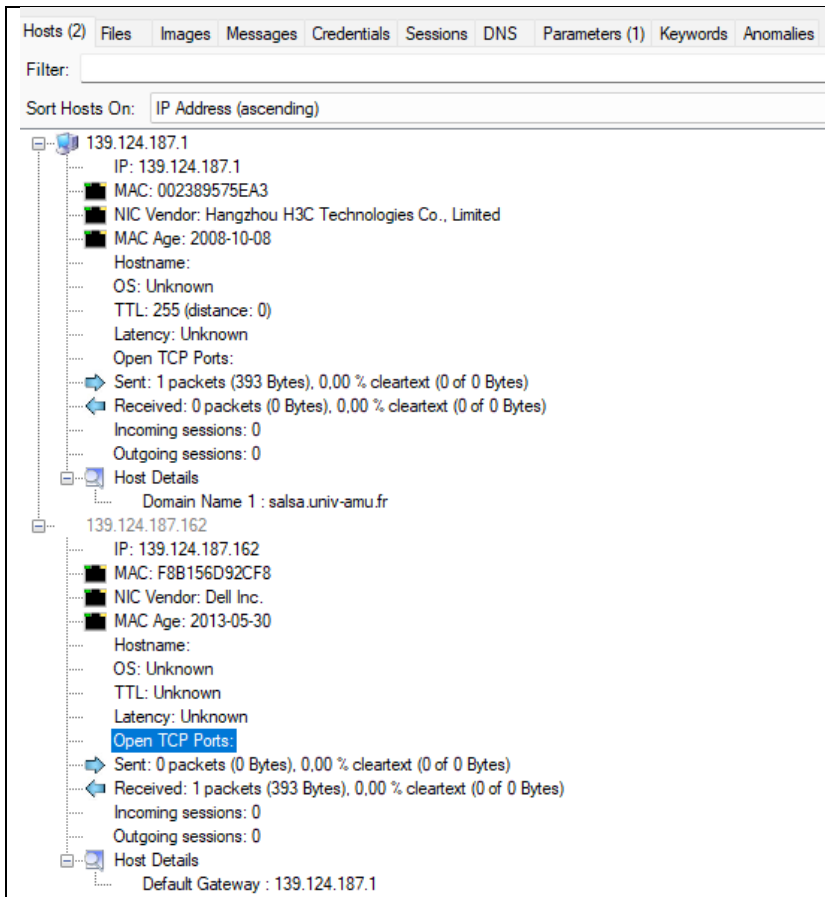
J'ai choisi la version gratuite sur le site officiel et sécurisé de netresec, le groupe qui a créé NetworkMiner. Les fonctionnalités que nous avons besoin sont comprises dedans.

### 1.9 : Première étude :

a)



## UTILISATION DE WIRESHARK



The screenshot shows the 'Hosts' pane in Wireshark. It lists two hosts:

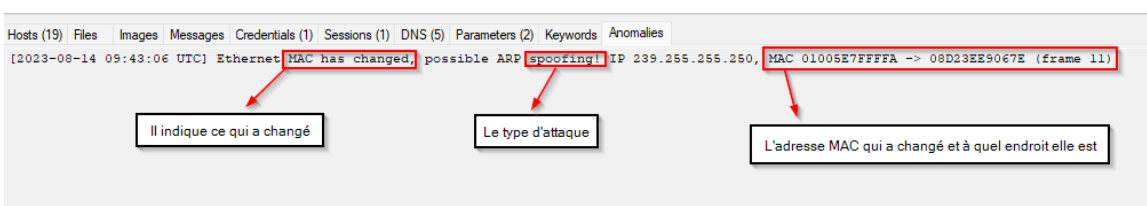
- 139.124.187.1**: IP: 139.124.187.1, MAC: 002389575EA3, NIC Vendor: Hangzhou H3C Technologies Co., Limited, MAC Age: 2008-10-08. It shows 1 packet sent and 0 received.
- 139.124.187.162**: IP: 139.124.187.162, MAC: F8B156D92CF8, NIC Vendor: Dell Inc., MAC Age: 2013-05-30. It shows 0 packets sent and 1 packet received.

Below the hosts list, the 'Host Details' section shows the domain name 'salsa.univ-amu.fr' and the default gateway '139.124.187.1'.

Nous n'avons pas de trame comme sur Wireshark, mais des adresses IP avec les détails. C'est plus facile à comprendre pour quelqu'un qui ne connaît rien ou qui a du mal dans l'analyse de trame.

Il y a une section pour voir les machines (hosts), une pour voir les fichiers (files), une pour les images, une pour les messages, une pour voir les identifiants d'authentification (Credentials), une pour les sessions, une pour les DNS, une pour les paramètres réseaux (Parameters), une pour des mots clés spécifique (Keywords) et une pour voir les anomalies.

b)



The screenshot shows a packet capture in Wireshark. The packet list pane displays a packet with the following details:

- Time: [2023-08-14 09:43:06 UTC]
- Interface: Ethernet
- Protocol: MAC has changed, possible ARP spoofing!
- IP: 239.255.255.250
- MAC: 01005E7FFFFA -> 08D23EE9067E (frame 11)

Annotations with arrows point to specific parts of the packet details:

- An arrow points to 'MAC has changed' with the text: 'Il indique ce qui a changé'.
- An arrow points to 'possible ARP spoofing!' with the text: 'Le type d'attaque'.
- An arrow points to the MAC address change '01005E7FFFFA -> 08D23EE9067E' with the text: 'L'adresse MAC qui a changé et à quel endroit elle est'.

Il y a un changement inattendu d'adresse MAC, ce qui peut être une attaque ARP spoofing. L'adresse MAC multicast est remplacée par une autre adresse sur le réseau (un poste dont l'hacker a accès).

Pendant une attaque ARP spoofing, l'hacker usurpe une adresse MAC dans la table ARP pour intercepter et rediriger le trafic réseau ou faire une attaque de type MITM.

C'est rassurant car c'est sur une adresse multicast et pas une passerelle par défaut, elle reste sur le réseau local sans en sortir. L'hacker peut perturber seulement les appareils sur le réseau. Le réseau ne sera pas complèCela montre comment l'outil permet à l'utilisateur de sélectionner et

## UTILISATION DE WIRESHARK

d'interagir avec différentes interfaces réseautement coupé. Il faut impérativement surveiller le réseau et identifier l'adresse MAC 08D23EE9067E pour régler ce problème.

### 1.10 : Etude des différents outils :

Synthese\_FTP.docx

## 6 Intelligence artificielle :

### 1.11 : ChatGPT :

Je vais utiliser GPT-4o (Chat-GPT) qui traite des fichiers et des outils Python qui permet d'analyser les fichiers. pcap (qui contiennent les trames).

|   |   |
|---|---|
| <p>Analyse ^</p> <pre>python  import pyshark  # Load the pcap file file_path = '/mnt/data/trame_1.pcap' capture = pyshark.FileCapture(file_path)  # Initialize a dictionary to store the analysis results pcap_analysis = {     "Total Packets": 0,     "Protocols": {},     "IP Addresses": {         "Source": {},         "Destination": {}     } }  # Analyze the pcap file for packet in capture</pre> <p>Toujours afficher les détails <input type="checkbox"/> Copier le code</p>  | <p>Chat GPT va utiliser Python pour analyser et extraire les données contenues dans la trame trame_1.pcap</p> |
| <pre>1032         restore_signals, 1033         gid, gids, uid, umask, 1034         start_new_session, process_group) 1035 except: 1036     # Cleanup if the child failed starting. 1037     for f in filter(None, (self.stdin, self.stdout, self.stderr)): 1038         os.close(f) 1039 1040 File /usr/local/lib/python3.11/subprocess.py:1953, in Popen._execute_child(self, args, executable, 1041     err_msg = os.strerror(errno_num) 1042     1952 if err_filename is not None: -&gt; 1953         raise child_exception_type(errno_num, err_msg, err_filename) 1044     1954 else: 1045         raise child_exception_type(errno_num, err_msg) 1046 FileNotFoundError: [Errno 2] No such file or directory: 'capinfos'</pre> <p>Je n'ai pas accès aux outils nécessaires pour analyser le fichier .pcap directement ici. Pour résoudre ce problème, je te propose deux solutions :</p> <ol style="list-style-type: none"> <li>1. <b>Utiliser Wireshark</b> localement pour analyser le fichier .pcap :             <ul style="list-style-type: none"> <li>• Télécharge Wireshark : <a href="#">Wireshark Download</a></li> <li>• Ouvre le fichier .pcap et utilise les filtres pour voir les protocoles, adresses IP, ports, et autres informations.</li> </ul> </li> <li>2. <b>Installer Scapy ou PyShark</b> sur ton environnement local :</li> </ol> | <p>Il n'arrive pas à analyser le fichier et me redirige vers Wireshark</p>                                    |

J'ai donc modifié le fichier directement en changeant l'extension en .txt pour qu'il soit lisible par Chat GPT :



## UTILISATION DE WIRESHARK

trame\_1.txt  
Document

analyse moi la trame et donne moi les infos

La trame analysée correspond à une requête **ARP** (Address Resolution Protocol). Voici les détails extraits de la trame :

### Contenu brut de la trame

yaml

 Copier le code

```
ffff ffff ffff 09ab 14d8 0548 0806 0001
0800 0604 0001 09ab 14d8 0548 7d05 300a
0000 0000 0000 7d12 6e03
```

J'ai eu les informations suivantes :

### Explication des champs

- Adresse de destination** : ffff ffff ffff
  - Broadcast** : la requête est envoyée à toutes les machines du réseau.
- Adresse MAC source** : 09:ab:14:d8:05:48
  - L'adresse MAC de l'expéditeur de la requête ARP.
- Type de protocole** : 0806
  - Ce champ indique que le protocole utilisé est **ARP**.
- Format matériel (Hardware Type)** : 0001
  - Indique qu'il s'agit d'une adresse MAC Ethernet.
- Format du protocole (Protocol Type)** : 0800
  - Indique le protocole IPv4.
- Longueur de l'adresse matérielle** : 06
  - La longueur de l'adresse MAC est de 6 octets.
- Longueur de l'adresse protocole** : 04
  - La longueur de l'adresse IPv4 est de 4 octets.
- Type d'opération** : 0001
  - C'est une requête ARP (1 = Request).

## UTILISATION DE WIRESHARK

## 9. Adresse MAC source : 09:ab:14:d8:05:48

- Répété pour le champ d'adresse source.

## 10. Adresse IP source : 125.5.48.10

- L'adresse IP de l'expéditeur.

## 11. Adresse MAC cible : 00:00:00:00:00:00

- Indique que l'adresse MAC de la cible est inconnue (d'où la requête ARP).

## 12. Adresse IP cible : 125.18.110.3

- L'adresse IP pour laquelle l'expéditeur cherche l'adresse MAC correspondante.

**Conclusion**

Il s'agit d'une **requête ARP** où l'émetteur demande l'adresse MAC associée à l'adresse IP 125.18.110.3.

Il a très bien analysé la trame, c'est l'intelligence artificielle la plus puissante et la plus développée aujourd'hui.

## 1.12 : MistralAI le Chat :

MistralAI le Chat est une intelligence artificielle française open-source et transparente mais elle est limitée et a moins de capacités que Chat GPT. Il ne peut pas lire les fichiers .pcap ni .txt, j'ai donc pris la trame\_1.txt et copier-coller le texte qu'il y avait dedans :

CO analyse moi la trame et donne moi les infos : ffff ffff ffff 09ab 14d8 0548 0806 0001  
0800 0604 0001 09ab 14d8 0548 7d05 300a  
0000 0000 0000 7d12 6e03

La trame que vous avez fournie semble être une séquence de données hexadécimales. Pour analyser cette trame, nous devons comprendre le contexte dans lequel elle est utilisée. Cependant, sans contexte spécifique, nous pouvons essayer de décomposer la trame en segments et voir si nous pouvons identifier des motifs ou des structures.

Voici la trame découpée en segments de 4 caractères hexadécimaux pour faciliter l'analyse :

```
ffff ffff ffff 09ab 14d8 0548 0806 0001
0800 0604 0001 09ab 14d8 0548 7d05 300a
0000 0000 0000 7d12 6e03
```

J'ai eu cette réponse :

## Analyse des segments

1. **ffff ffff ffff**
  - Ces valeurs peuvent représenter des marqueurs ou des valeurs de remplissage.
2. **09ab 14d8 0548**
  - Ces valeurs peuvent être des identifiants, des adresses ou des données spécifiques.
3. **0806 0001**
  - Ces valeurs peuvent représenter des commandes ou des paramètres.
4. **0800 0604 0001**
  - Ces valeurs peuvent également représenter des commandes ou des paramètres.
5. **09ab 14d8 0548**
  - Réapparition des mêmes valeurs, ce qui suggère une répétition ou une structure récurrente.
6. **7d05 300a**
  - Ces valeurs peuvent être des données spécifiques ou des marqueurs de fin de segment.
7. **0000 0000 0000**
  - Ces valeurs peuvent représenter des zéros de remplissage ou des valeurs par défaut.
8. **7d12 6e03**
  - Ces valeurs peuvent être des données spécifiques ou des marqueurs de fin de trame.

## Hypothèses possibles

1. En-tête et queue de trame
  - Les segments **ffff ffff ffff** et **0000 0000 0000** pourraient être des en-têtes ou des queues de trame.
2. Identifiants ou adresses
  - Les segments **09ab 14d8 0548** pourraient être des identifiants ou des adresses récurrents.
3. Commandes ou paramètres

## UTILISATION DE WIRESHARK

- Les segments **0806 0001** et **0800 0604 0001** pourraient être des commandes ou des paramètres spécifiques.

## 4. Données spécifiques

- Les segments **7d05 300a** et **7d12 6e03** pourraient être des données spécifiques ou des marqueurs de fin de segment.

## Conclusion

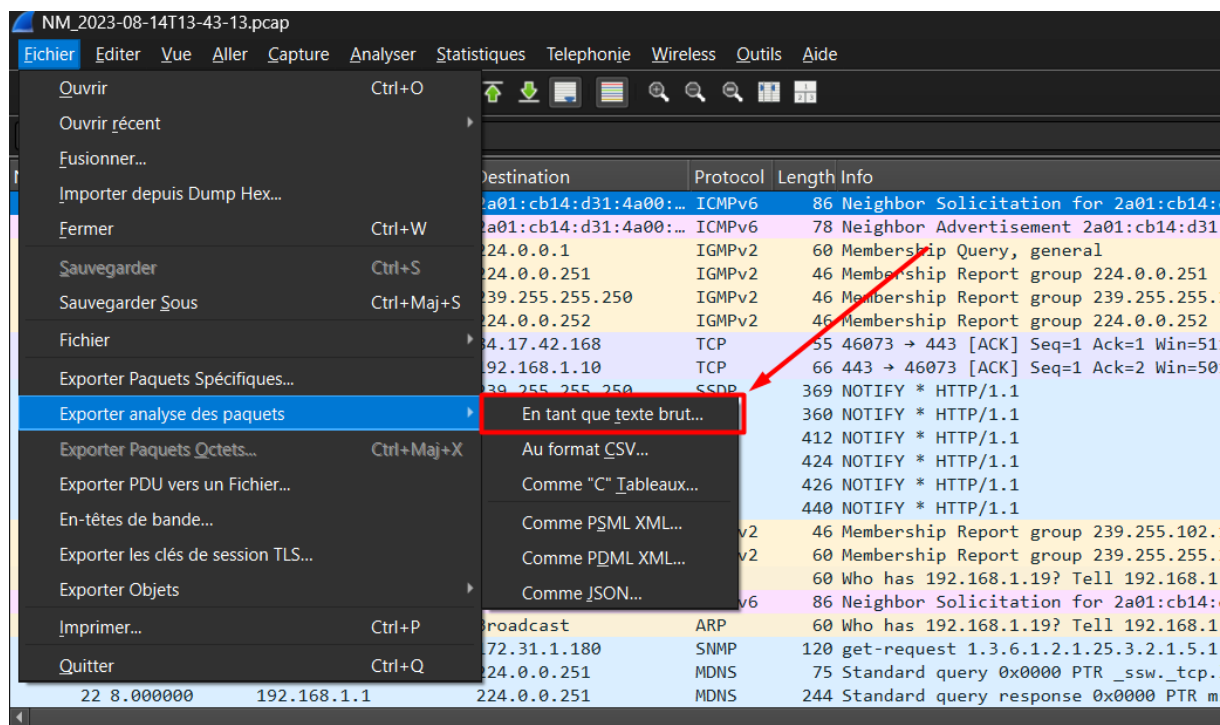
Sans contexte supplémentaire, il est difficile de fournir une analyse précise de la trame. Les segments identifiés peuvent représenter différentes parties d'un protocole de communication, mais une compréhension plus approfondie nécessiterait des informations sur le contexte d'utilisation de cette trame (par exemple, le type de protocole, l'application spécifique, etc.).

Si vous avez des informations supplémentaires sur le contexte ou l'application de cette trame, cela aiderait à fournir une analyse plus détaillée.

Ce qui est largement insuffisant car l'IA n'est pas aussi développée que Chat GPT. Elle est un grand cran en dessous de Chat GPT.

1.13 : Autre méthode :

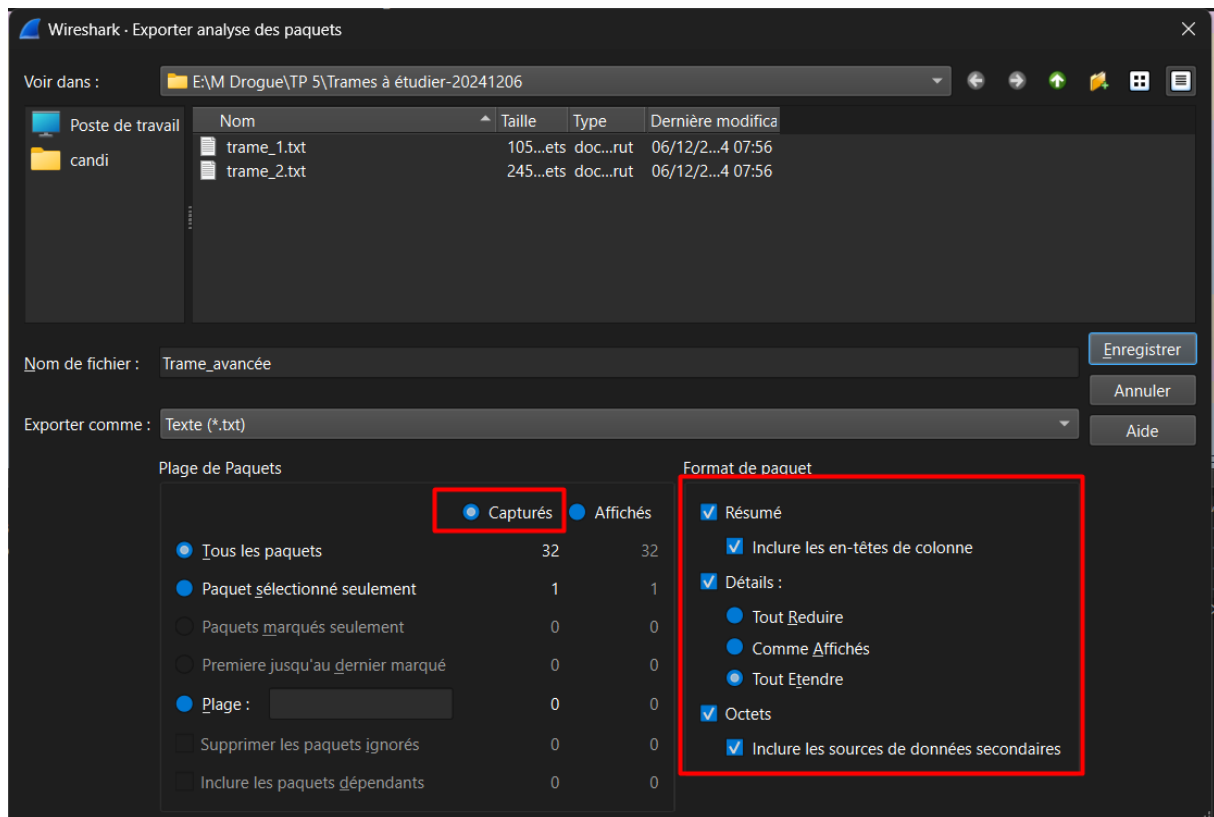
Nous allons utiliser cette fois la trame NM\_2023-08-14T13-43-13, plus complexe avec son changement d'adresse MAC.



Nous choisissons de l'exporter "en tant que texte brut". Cette option est utile lorsqu'on souhaite enregistrer une analyse de trame dans un format lisible. Ce qui permet de partager facilement les résultats ou d'analyser les paquets dans un autre outil de traitement de texte. C'est

## UTILISATION DE WIRESHARK

particulièrement utile pour les rapports d'analyse ou pour une inspection manuelle plus approfondie des paquets.

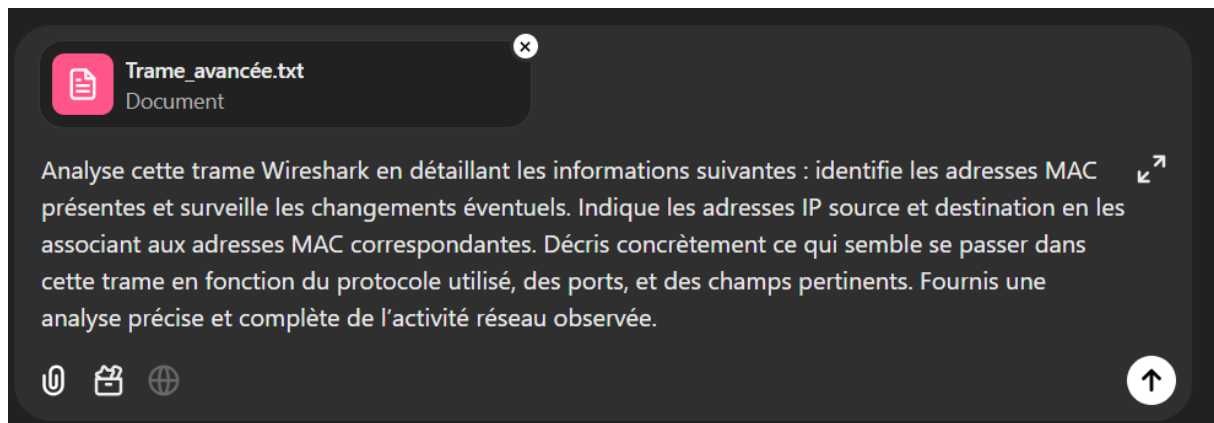


Nous avons choisi l'option "Capturés" pour n'exporter que les paquets capturés dans l'analyse, pas ceux qui ont été filtrés ou affichés uniquement. Ensuite, nous avons sélectionné le format d'exportation en texte et coché les options pour inclure un résumé et les en-têtes de colonnes, ce qui nous permet de récupérer des informations détaillées, mais structurées de manière lisible. Cela rend l'analyse plus complète et accessible pour un examen plus approfondi.

Prompt :

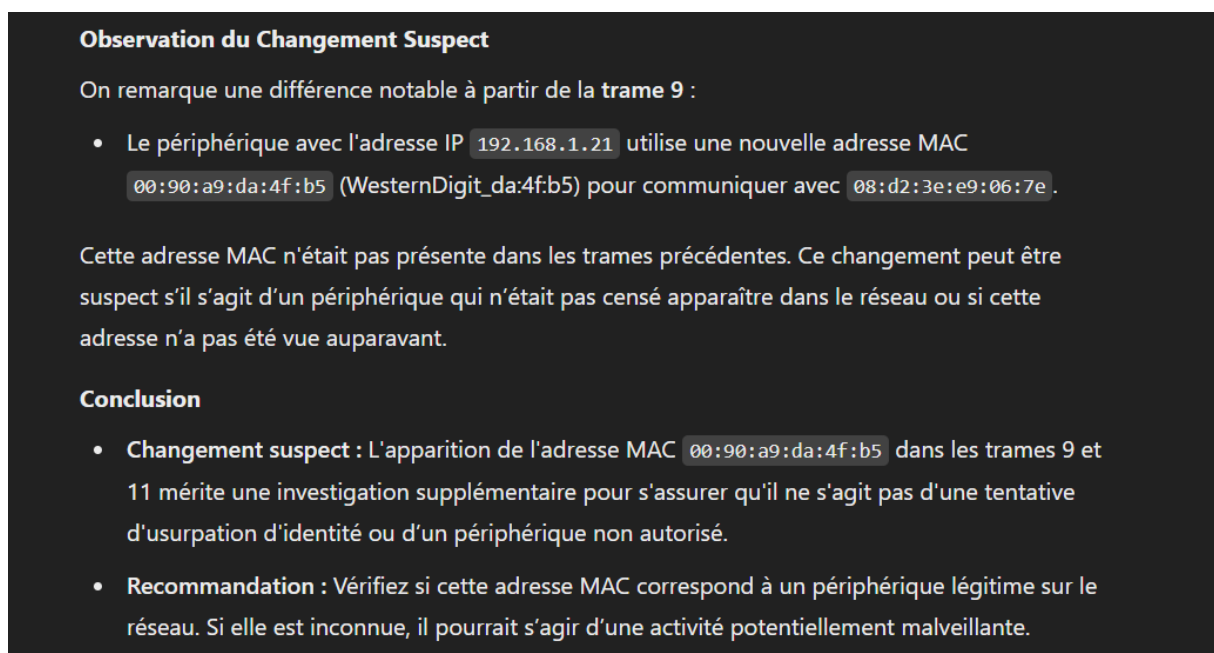
Analyse cette trame Wireshark en détaillant les informations suivantes : identifie les adresses MAC présentes et surveille les changements éventuels. Indique les adresses IP source et destination en les associant aux adresses MAC correspondantes. Décris concrètement ce qui semble se passer dans cette trame en fonction du protocole utilisé, des ports, et des champs pertinents. Fournis une analyse précise et complète de l'activité réseau observée.

### 1.1.10 : Chat GPT



Dans cette capture, nous allons identifier les adresses MAC, surveiller les changements dans les trames et lier ces adresses aux adresses IP source et destination. Le but est de fournir une analyse détaillée des protocoles utilisés, des ports et des champs pertinents pour déterminer l'activité réseau et voir d'éventuelles anomalies ou attaques.


Résultat :



Ici, l'observation se concentre sur un changement suspect d'adresse MAC dans la trame 9. Un périphérique avec l'adresse IP 192.168.1.21 utilise une nouvelle adresse MAC 00:90:a9:da:4f:b5, ce qui est anormal, car cette adresse MAC ne faisait pas partie des trames précédentes.



De quel type d'attaque peut-il s'agir ?

 **Types d'attaques potentielles associées à un changement suspect d'adresse MAC**

Le changement d'adresse MAC observé ( `00:90:a9:da:4f:b5` ) peut indiquer plusieurs types d'attaques réseau. Voici les scénarios les plus probables :

---

**1. Usurpation d'Adresse MAC (MAC Spoofing)**

- **Description :** Un attaquant modifie son adresse MAC pour imiter celle d'un périphérique légitime. Cela peut permettre à l'attaquant de :
  - Intercepter le trafic destiné au périphérique usurpé.
  - Bypasser les contrôles d'accès basés sur l'adresse MAC.
- **Impact potentiel :**
  - Interception de données sensibles circulant sur le réseau.
  - Détournement de session si des sessions actives sont redirigées vers l'attaquant.

Dans cette capture, plusieurs types d'attaques potentiellement associées à un changement d'adresse MAC sont présentées. Le changement d'adresse MAC observé pourrait signaler des attaques comme l'usurpation d'adresse MAC ou une attaque Man-in-the-Middle (MITM).

**2. Attaque Man-in-the-Middle (MITM)**

- **Description :** L'attaquant se positionne entre deux périphériques pour intercepter et potentiellement altérer les communications.
- **Scénario possible :**

L'attaquant utilise une adresse MAC usurpée pour tromper les périphériques du réseau et rediriger le trafic à travers lui.
- **Impact potentiel :**
  - **Espionnage du trafic,** y compris des informations d'authentification ou des données sensibles.
  - **Modification du trafic,** comme injecter des paquets malveillants.

Cette capture détaille l'attaque Man-in-the-Middle (MITM). L'attaquant se place entre deux périphériques pour intercepter et potentiellement manipuler les communications. Dans ce cas, l'attaque pourrait impliquer l'utilisation d'une adresse MAC usurpée pour rediriger le trafic réseau à travers l'attaquant.

## UTILISATION DE WIRESHARK

## 1.1.11 : Gemini

On lui donne en premier la trame car trop grosse pour aller avec un prompt :

| C | No. | Time     | Source                                | Destination                            | Protocol | Length | Info   | ▼ |
|---|-----|----------|---------------------------------------|--|----------|--------|--|---|
|   | 1   | 0.000000 | 2a01:cb14:d31:4a00:12b:873b:196e:e52b | 2a01:cb14:d31:4a00:7ec1:77ff:fe4b:17a0 | ICMPv6   | 86     | Neighbor Solicitation for 2a01:cb14:d31:4a00:7ec1:77ff:fe4b:17a0 from 08:d2:3e:e9:06:7e... |   |

Puis on lui donne prompte :

C

Analyse cette trame Wireshark en détaillant les informations suivantes : identifie les adresses MAC présentes et surveille les changements éventuels. Indique les adresses IP source et destination en les associant aux adresses MAC correspondantes. Décris concrètement ce qui...

▼

✦

Absolument ! Analyser une capture Wireshark, c'est un peu comme plonger dans les coulisses d'Internet. Cela permet de comprendre comment les données circulent entre les différents appareils.

**Pour répondre à ta demande, je vais te guider étape par étape:**

**1. Identifier les Adresses MAC et IP**

- **Adresses MAC:** Ce sont des adresses uniques attribuées à chaque interface réseau. Elles sont généralement représentées par une suite de 12 chiffres hexadécimaux séparés par des points (ex : 00:11:22:33:44:55).
- **Adresses IP:** Ce sont des adresses numériques attribuées à chaque appareil connecté à un réseau. Elles peuvent être de type IPv4 (ex : 192.168.1.1) ou IPv6 (ex : 2001:db8:85a3:0042:1000:8a2e:0370:7334).

Dans Wireshark, tu trouveras ces informations dans la section "Ethernet II" pour les adresses MAC et dans la section "Internet Protocol Version 4" ou "Internet Protocol Version 6" pour les adresses IP.

## UTILISATION DE WIRESHARK

**1. Identifier les Adresses MAC et IP**

- **Adresses MAC:** Ce sont des adresses uniques attribuées à chaque interface réseau. Elles sont généralement représentées par une suite de 12 chiffres hexadécimaux séparés par des points (ex : 00:11:22:33:44:55).
- **Adresses IP:** Ce sont des adresses numériques attribuées à chaque appareil connecté à un réseau. Elles peuvent être de type IPv4 (ex : 192.168.1.1) ou IPv6 (ex : 2001:db8:85a3:0042:1000:8a2e:0370:7334).

Dans Wireshark, tu trouveras ces informations dans la section "Ethernet II" pour les adresses MAC et dans la section "Internet Protocol Version 4" ou "Internet Protocol Version 6" pour les adresses IP.

**2. Suivre les Changements d'Adresses MAC**

- **Pourquoi ?** Les adresses MAC peuvent changer dans certains cas, par exemple lors de l'utilisation de DHCP ou si un appareil change d'interface réseau.
- **Comment ?** En observant plusieurs trames consécutives, tu pourras identifier si une adresse MAC est associée à différentes adresses IP ou si une adresse IP est associée à différentes adresses MAC.

**3. Associer Adresses MAC et IP**

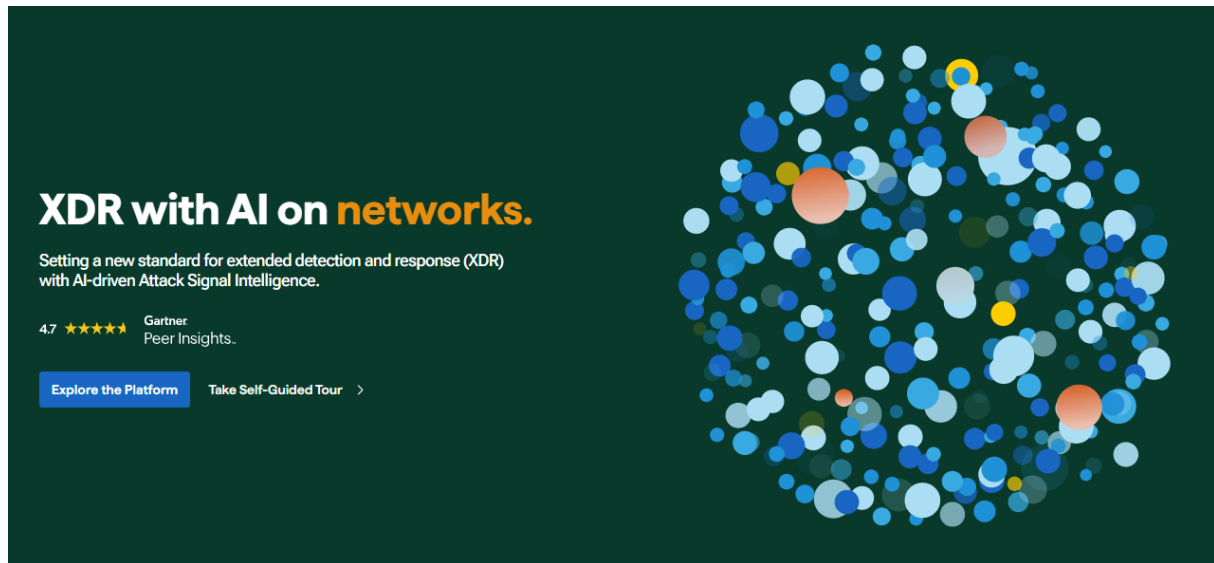
- **Table ARP:** Sur les réseaux locaux, le protocole ARP (Address Resolution Protocol)

Gemini explique pourquoi et comment les changements d'adresses MAC peuvent se produire. Le suivi des trames successives permet d'identifier si une adresse MAC est liée à plusieurs adresses IP ou si une adresse IP est associée à différentes adresses MAC, ce qui peut signaler un problème de réseau ou une tentative d'usurpation d'identité.

**1.1 : [Vectra AI \(Vectra AI - Advanced AI Security - Stop Cyberattacks Fast\)](#):**

C'est une plateforme spécialisée dans la détection et la réponse aux menaces. Elle est basée sur l'intelligence artificielle pour automatiser ça.

Elle a une détection en temps réel, des alertes plus pertinentes et une visibilité complète du réseau mais a un coût élevé et est compliqué à mettre en place.



## Conclusion :

L'utilisation de Wireshark et d'outils complémentaires comme Ettercap ou NetworkMiner permet une analyse approfondie des paquets réseau pour diagnostiquer les problèmes de communication et identifier les attaques potentielles. Dans le cadre de l'analyse des trames, il est crucial d'examiner les adresses MAC et IP, ainsi que les protocoles utilisés, pour détecter des anomalies comme le changement suspect d'adresses MAC, pouvant indiquer des tentatives d'usurpation d'identité ou des attaques de type Man-in-the-Middle (MITM).

Wireshark offre une visibilité complète sur les paquets, permettant de diagnostiquer des problèmes et de détecter des failles de sécurité. L'outil fournit aussi des informations cruciales sur la performance du réseau et les types de protocoles utilisés, comme ICMP, TCP, et UDP. En complément, les outils comme Ettercap peuvent simuler des attaques sur le réseau pour tester la sécurité et la résilience de l'infrastructure.

L'intégration de ces outils dans un cadre de gestion de réseau permet une meilleure maîtrise des risques liés à la sécurité et garantit un réseau plus stable et performant. L'importance de surveiller les réseaux en temps réel et d'analyser les trames pour identifier des comportements inhabituels est essentielle pour maintenir la sécurité et l'intégrité des données circulant sur le réseau.


## GitHub :

### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

*Required fields are marked with an asterisk (\*).*

Owner \*

 candicevl

Repository name \*


Wireshark-analyse-trames

✔ Wireshark-analyse-trames is available.


Great repository names are short and memorable. Need inspiration? How about [super-duper-telegram](#) ?

Description (optional)

Atelier pratique d'analyse de trames réseau avec Wireshark. Ce repository propose des captures de paquets, d

☒  Public

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore


.gitignore template: **None**


Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None**

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

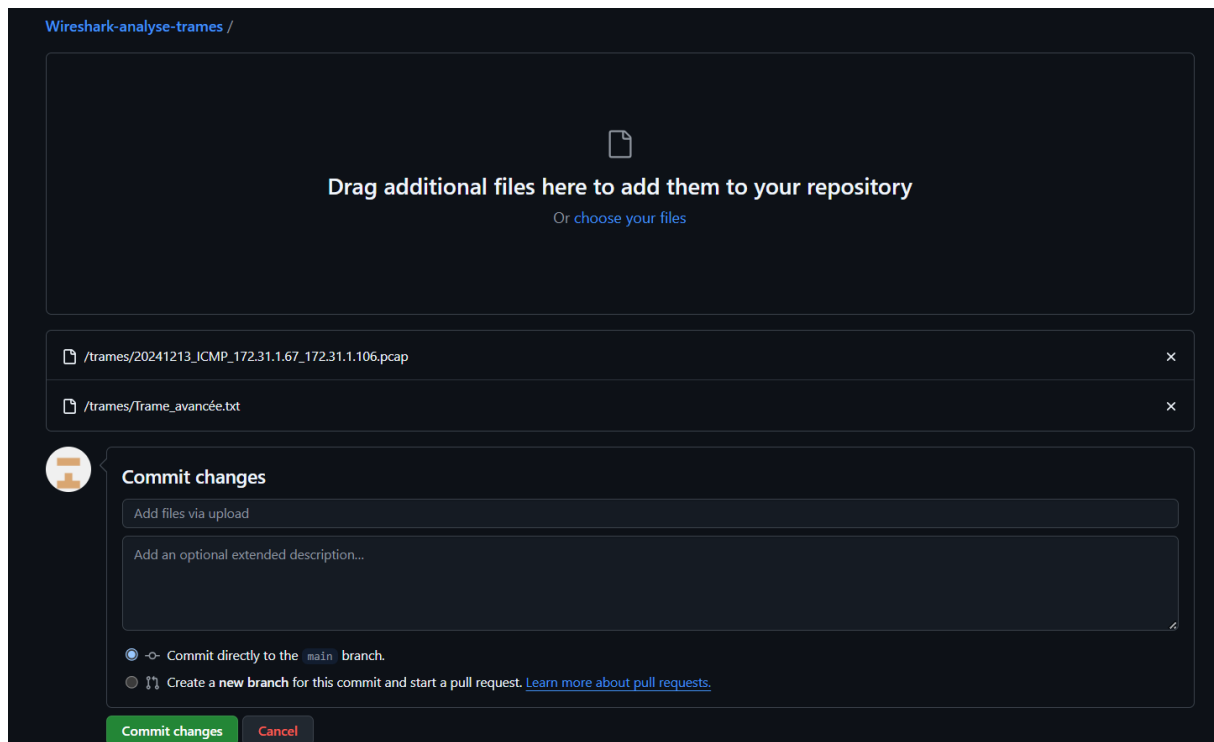
This will set  main as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

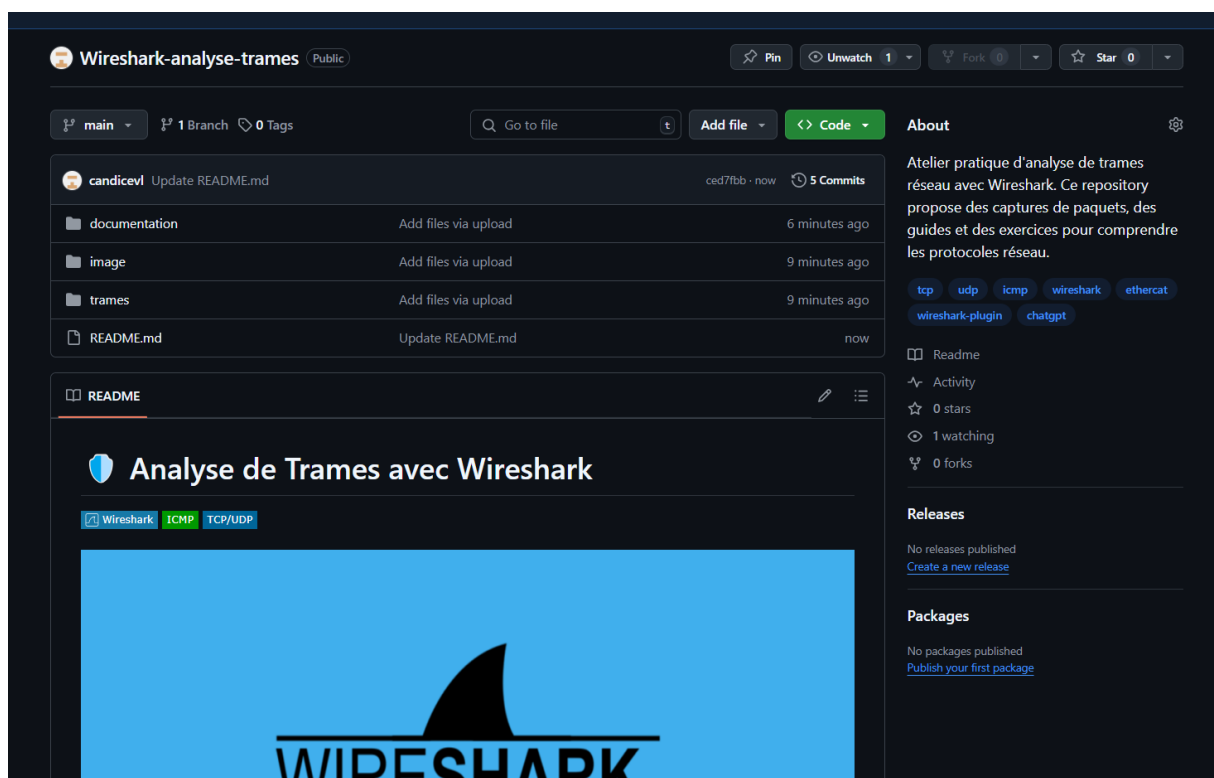
Create repository

Dans cette capture, nous voyons l'interface de création d'un nouveau repository sur GitHub. On a choisi le nom du repository "Wireshark-analyse-trames" comme le TP 5 et une description optionnelle est ajoutée. L'option "Public" a été sélectionnée, ce qui signifie que le repository sera accessible par tout le monde sur Internet.

## UTILISATION DE WIRESHARK



Cette capture montre l'ajout de fichiers dans le repository qu'on a créé. Les fichiers .pcap et .txt sont chargés, ce qui permet de les inclure dans le repository pour y ajouter les trames. L'option "Commit directly to the main branch" a été sélectionnée, ce qui envoie ces fichiers directement à la branche principale du repository. C'est plus simple quand on travaille seul et si les fichiers ne doivent pas être validés.



Ici, nous pouvons voir l'interface du repository GitHub après l'ajout des fichiers. Le fichier

## UTILISATION DE WIRESHARK

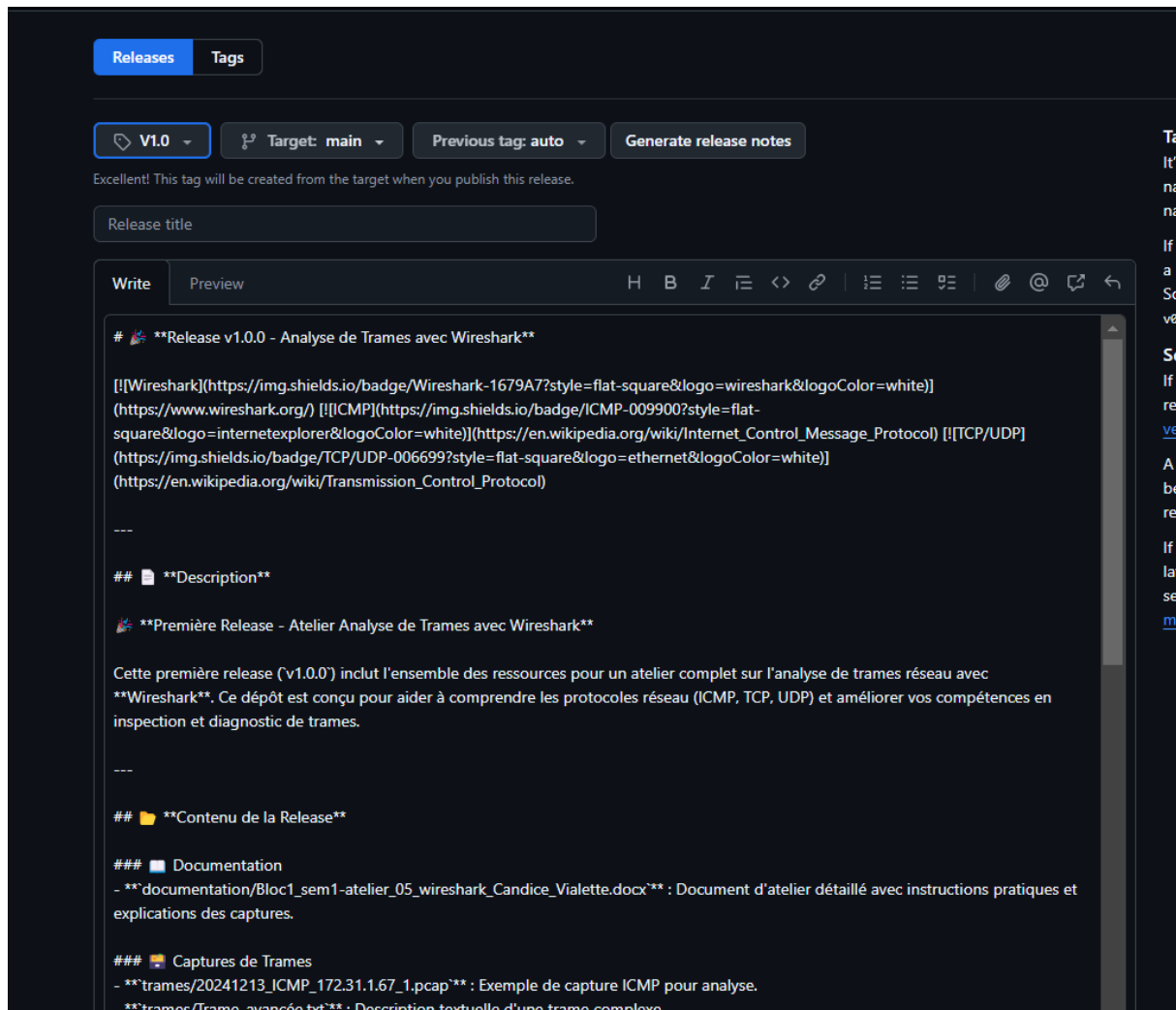
README est affiché, avec la description du projet "Analyse de Trames avec Wireshark" qui est maintenant visible. La structure du repository inclut des dossiers pour la documentation, les images et les captures de trames, et les fichiers sont organisés en conséquence. L'objectif est de rendre le projet bien structuré et facile à parcourir.



Cette capture montre la page du README (d'accueil) avec des informations supplémentaires sur le projet. Elle fournit une description détaillée du contenu du repository, en expliquant que ce dépôt est dédié à l'analyse des trames réseau avec Wireshark. Elle mentionne aussi que ce

## UTILISATION DE WIRESHARK

projet est conçu pour aider à développer des compétences en inspection et en diagnostic de trames réseau. Cette page d'introduction est essentielle pour toute personne visitant le repository, car elle définit clairement les objectifs et les ressources disponibles.



Dans cette capture, nous voyons la configuration de la première release d'un projet sur GitHub. Le nom de la release, avec une description détaillée sur ce que cette release inclut : un atelier complet pour l'analyse de trames réseau avec Wireshark. La release présente des badges pour les protocoles concernés (Wireshark, ICMP, TCP/UDP) et un lien vers des ressources externes.



## UTILISATION DE WIRESHARK

Releases / V1.0

V1.0 Pre-release Compare

candicevl released this in 1 minute V1.0 ced7fbb

## Release v1.0.0 - Analyse de Trames avec Wireshark

Wireshark ICMP TCP/UDP

### Description

Première Release - Atelier Analyse de Trames avec Wireshark

Cette première release ( `v1.0.0` ) inclut l'ensemble des ressources pour un atelier complet sur l'analyse de trames réseau avec **Wireshark**. Ce dépôt est conçu pour aider à comprendre les protocoles réseau (ICMP, TCP, UDP) et améliorer vos compétences en inspection et diagnostic de trames.

### Contenu de la Release

#### Documentation

- `documentation/Bloc1_sem1-atelier_05_wireshark_Candice_Vialette.docx` : Document d'atelier détaillé avec instructions pratiques et explications des captures.

#### Captures de Trames

- `trames/20241213_ICMP_172.31.1.67_1.pcap` : Exemple de capture ICMP pour analyse.
- `trames/Trame_avancée.txt` : Description textuelle d'une trame complexe.

#### Images

- `image/background.png` : Image de fond pour la présentation de l'atelier.

#### Guide

- `README.md` : Guide d'utilisation complet du dépôt avec instructions pour analyser les trames.

Ici, nous voyons le détail de la release 1.0.0 une fois publiée. La description et le contenu mentionnant les fichiers inclus comme le document d'atelier et les captures de trames pour l'analyse.

## Date de Release

Décembre 2024

### Assets 2

- Source code (zip)
- Source code (tar.gz)

## UTILISATION DE WIRESHARK

Cette capture montre une section où la date de la release est affichée, avec l'option de télécharger les fichiers sources sous deux formats (zip et tar.gz). Ces formats permettent aux utilisateurs de télécharger facilement l'ensemble du code source ou des fichiers associés à la release pour leur propre utilisation ou développement.

Lien vers mon GitHub : <https://github.com/candicevialette>

## Sources :

[Wireshark · Documentation](#)

[Wireshark User's Guide](#)

[IEEE 802.3 — Wikipédia](#)

[Qu'est-ce qu'une attaque MAN IN THE MIDDLE \(MITM\)? - IONOS](#)

[Les réseaux Ethernet: Le format des trames](#)

[Création d'un dépôt - Documentation GitHub](#)

[Surveillez votre réseau et surveillez votre bande passante avec NetworkMiner / les fenêtres | Nouvelles du monde de la technologie moderne!](#)