

WireSkark v2

sommaire :

Partie 0 : Introduction

1. Présentation de Wireshark
2. Concepts de base (Trame, Paquet et fichier de capture)

Partie 1 : Initiation à Wireshark et Modèle OSI sur l'Alcazar

1. Pratique avec Wireshark (Capture des trames ARP , UDP TCP et désencapsulation)
2. Analyse des en-têtes (Adresse MAC/IP , Spécification Protocole ARP, UDP,TCP)
3. Mécanisme de connexion TCP (Diagramme des étapes)

Partie 2 : Capture Ciblée de Protocoles

1. Mise en place d'un lab réseau avec 2 VM (serveur + client en NAT)
2. Capture et analyse des protocoles (DHCP, DNS, mDNS, FTP, SMB, HTTPS/TLSv1.2)
3. Sauvegarde des paquets pertinents

Partie 3 : Automatisation avec Tshark

1. Installation et prise en main de Tshark
2. Commandes de capture avancées
3. Redirection et traitement des données

Partie 0 : Introduction

1. Présentation de Wireshark

Wireshark est un analyseur de protocoles réseau open source, utilisé pour capturer, inspecter et analyser le trafic réseau en temps réel. C'est un outil essentiel pour les administrateurs réseau, les pentesters et les développeurs, permettant de diagnostiquer des problèmes de connectivité, d'analyser des attaques ou de comprendre le fonctionnement des protocoles.

Fonctionnalités principales :

- Capture en direct du trafic (Ethernet, Wi-Fi, etc.).
- Analyse approfondie des paquets (TCP, UDP, HTTP, DNS, etc.).
- Filtres puissants pour isoler des flux spécifiques.
- Décodage de protocoles (y compris chiffrés avec les clés appropriées).
- Support multiplateforme (Windows, Linux, macOS).

Cas d'usage :

- Dépannage réseau.
- Investigation de cyberattaques.
- Reverse engineering de communications.
- Éducation sur les protocoles réseau.

Wireshark est un outil puissant, mais complexe : son utilisation nécessite des bases en réseaux pour interpréter correctement les données.

Installation sous linux :

Commande bash :

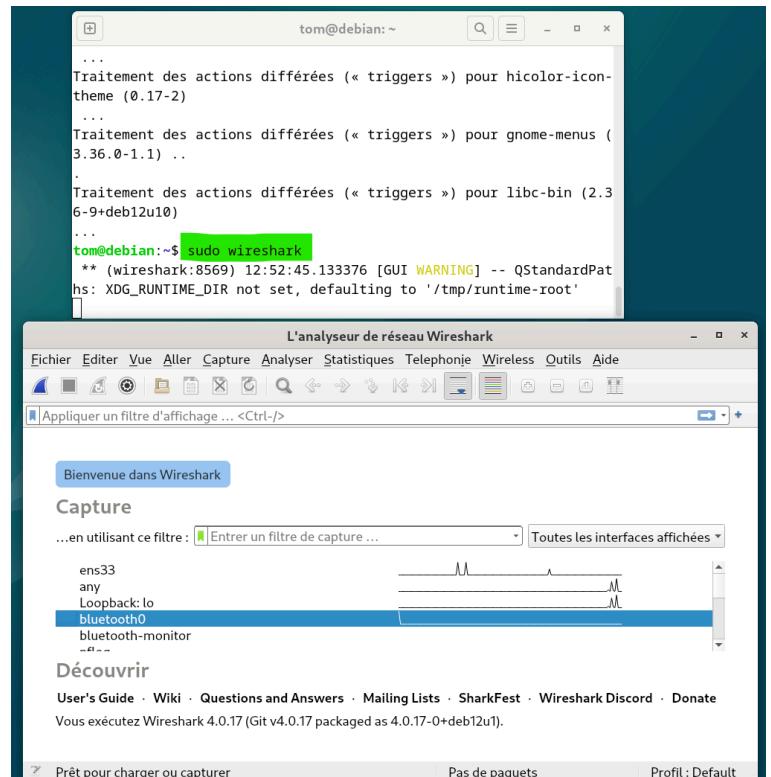
sudo apt update

sudo apt install wireshark

Pour lancer Wireshark en root :

sudo wireshark

```
tom@debian:~$ sudo apt update
[sudo] Mot de passe de tom :
Atteint :1 http://deb.debian.org/debian bookworm InRelease
Atteint :2 http://deb.debian.org/debian bookworm-updates InRelease
Atteint :3 http://security.debian.org/debian-security bookworm-security InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
tom@debian:~$ sudo apt install wireshark
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
 libbcg729-0 libc-ares2 libdouble-conversion3 liblua5.2-0 libmd4c0
 libminizip1 libpcre2-16-0 libqt5core5a libqt5dbus5 libqt5gui5
 libqt5multimedia5 libqt5multimedia5-plugins libqt5multimediasettings5
 libqt5multimediacwidgets5 libqt5network5 libqt5printsupport5 libqt5qml5
 libqt5qmlmodels5 libqt5quick5 libqt5svg5 libqt5waylandclient5
 libqt5waylandcompositor5 libqt5widgets5 libsmi2l dbus libspeexdsp1
 libwireshark-data libwireshark16 libwiretap13 libwsutil14 libxcb-xinerama0
 libxcb-xinput0 qt5-gtk-platformtheme qttranslations5-l10n qtwayland5
 wireshark-common wireshark-qt
```



Concepts de base

Différence entre trame (L2) et paquet (L3) :

- **Trame :**

Une trame est une unité de données de la couche liaison (couche 2 /Layer2 du modèle OSI). Elle inclut une en-tête, une charge utile (données) et une séquence de fin de trame (FCS). La trame est encapsulée dans les protocoles de la couche liaison, comme Ethernet.

- **Paquet :**

Un paquet est une unité de données de la couche réseau (couche 3 /Layer3 du modèle OSI). Il inclut un en-tête contenant des informations de routage et une charge utile (données). Les paquets sont encapsulés dans les protocoles de la couche réseau, comme IP

Formats PCAP/PCAPNG :

- **Format pcap :**

Le format de capture de paquets (pcap) est un format de fichier standard pour l'enregistrement des données capturées par les analyseurs réseau. Il stocke les paquets capturés de manière séquentielle et est largement utilisé par les outils d'analyse comme Wireshark.

- **Format pcapng :**

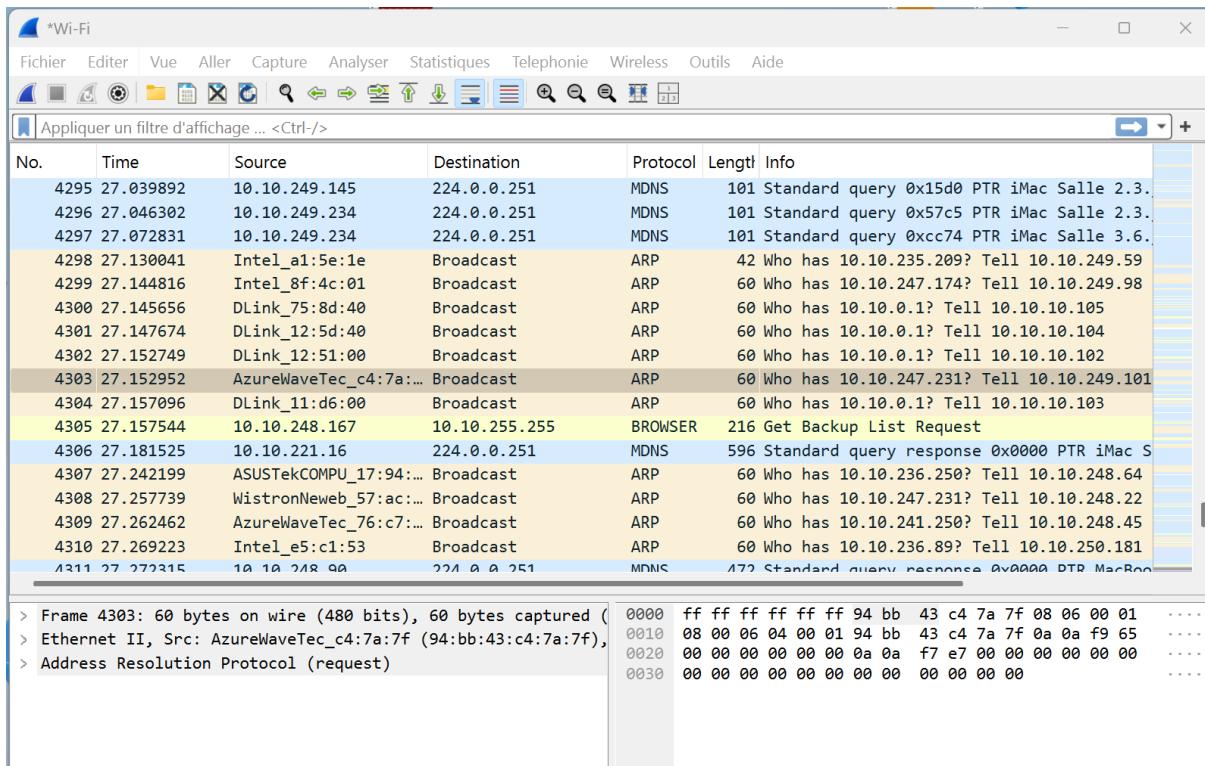
Le format pcap Next Generation (pcapng) est une version améliorée de pcap, offrant des fonctionnalités supplémentaires telles que le support multi-interface, des options de métadonnées étendues et une meilleure structuration des fichiers de capture.

Partie 1 : Initiation à Wireshark et Modèle OSI sur L'alcazar

1. Pratique avec Wireshark

a. Capture de trames

1. Lancez Wireshark en root
2. Sélectionnez l'interface connectée à Alcasar (généralement eth0 ou wlan0)
3. Cliquez sur "Start" pour commencer la capture



Je peux voir le trafic de l'alcazar en direct

b. Filtrage des protocoles

Dans la barre de filtre (en vert) :

Je tape les raccourcis suivants:

● ARP : arp

The screenshot shows a Wireshark capture window titled "arp". The list of captured frames includes:

- Frame 271: ARP request from ActionsMicro_42:01:... to Broadcast (ARP)
- Frame 277: ARP request from WistronNeweb_57:ac:... to Broadcast (ARP)
- Frame 280: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 287: ARP request from AzureWaveTec_77:68:... to Broadcast (ARP)
- Frame 288: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 289: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 290: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 291: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 292: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 293: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 294: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 295: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 296: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 297: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 298: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)
- Frame 299: ARP request from c2:5e:dc:19:8f:88 to Broadcast (ARP)

Details for frame 288:

- Type:** ARP (0x0806)
- [Stream index:** 105]
- Padding:** 00000000000000000000000000000000

Address Resolution Protocol (request) details:

- Hardware type:** Ethernet (1)
- Protocol type:** IPv4 (0x0800)
- Hardware size:** 6
- Protocol size:** 4
- Opcode: request (1)**
- Sender MAC address:** c2:5e:dc:19:8f:88 (c2:5e:dc:19:8f:88)
- Sender IP address:** 10.10.247.107
- Target MAC address:** 00:00:00_00:00:00 (00:00:00:00:00:00)
- Target IP address:** 10.10.250.229

● UDP : udp

The screenshot shows a Wireshark capture window titled "udp". The list of captured frames includes:

- Frame 4: DNS query from 10.10.245.8 to 224.0.0.251 (MDNS)
- Frame 5: DNS response from 224.0.0.251 to 10.10.245.8 (MDNS)
- Frame 6: DNS query from fe80::b8d4:4b01:a01... to ff02::fb (MDNS)
- Frame 7: DNS query from fe80::b8d4:4b01:a01... to ff02::fb (MDNS)
- Frame 11: DNS query from 10.10.249.238 to 224.0.0.251 (MDNS)
- Frame 13: DNS query from 10.10.247.220 to 224.0.0.251 (MDNS)
- Frame 14: DNS query from fe80::b504:46e2:13b... to ff02::fb (MDNS)
- Frame 15: DNS query from 10.10.251.8 to 224.0.0.251 (MDNS)
- Frame 16: DNS query from fe80::c69:2026:6d3c... to ff02::fb (MDNS)
- Frame 18: DNS query from fe80::425:3a75:e201... to ff02::fb (MDNS)
- Frame 19: DNS query from 10.10.234.186 to 224.0.0.251 (MDNS)
- Frame 20: DNS query from fe80::c50:ab76:be0b... to ff02::fb (MDNS)
- Frame 21: DNS query from 10.10.250.226 to 224.0.0.251 (MDNS)
- Frame 24: DNS query from 10.10.250.11 to 224.0.0.251 (MDNS)
- Frame 25: DNS query from fe80::18f6:6f8b:7de... to ff02::fb (MDNS)
- Frame 26: DNS query from 10.10.250.198 to 239.255.255.250 (SSDP)
- Frame 28: DNS query from 10.10.249.238 to 224.0.0.251 (MDNS)
- Frame 29: DNS query from 10.10.249.238 to 224.0.0.251 (MDNS)

Details for frame 29:

- Type:** IPv4 (0x0800)
- [Stream index:** 2]
- > Internet Protocol Version 4, Src: 10.10.245.8, Dst: 224.0.0.251**
- > User Datagram Protocol, Src Port: 5353, Dst Port: 5353**

● TCP : tcp

The screenshot shows a Wireshark capture window titled "tcp". The list of captured frames includes:

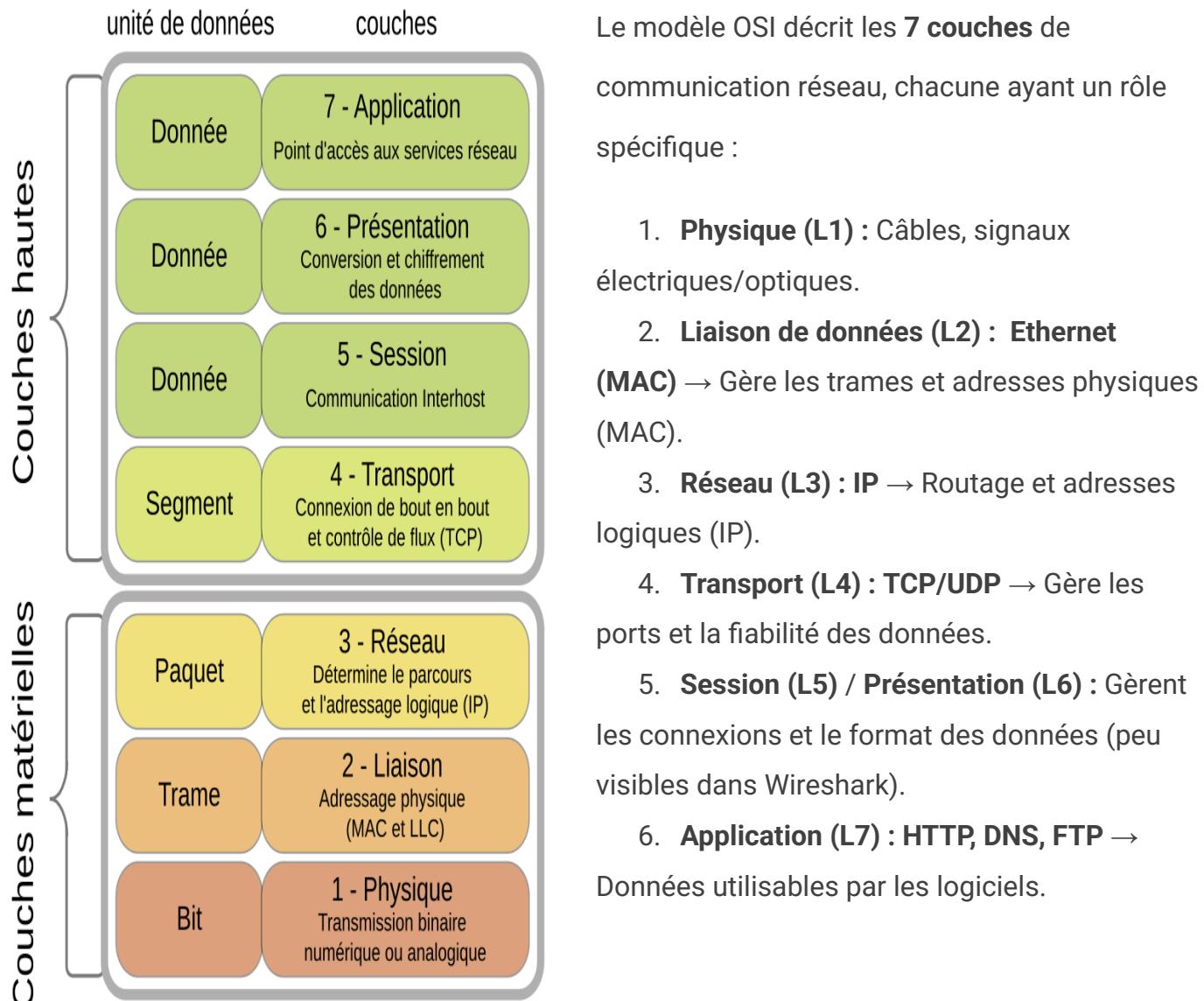
- Frame 1145: TLS handshake from 10.10.249.59 to 10.10.249.59 (TCP)
- Frame 1156: TLS handshake from 10.10.249.59 to 10.10.249.59 (TCP)
- Frame 1308: TLS handshake from 10.10.249.59 to 10.10.249.59 (TCP)
- Frame 1309: TLS handshake from 10.10.249.59 to 10.10.249.59 (TCP)
- Frame 1310: TLS handshake from 10.10.249.59 to 10.10.249.59 (TLSv1.2)
- Frame 1311: TLS handshake from 10.10.249.59 to 10.10.249.59 (TLSv1.2)
- Frame 1312: TLS handshake from 10.10.249.59 to 10.10.249.59 (TCP)
- Frame 1558: TLS handshake from 10.10.249.59 to 10.10.248.188 (TCP)

Details for frame 1312:

- Type:** TCP (0x0803)
- Sequence Number:** 4293
- Acknowledgment Number:** 16292
- Flags:** [RST, ACK]
- Sequence Range:** Seq=2 Ack=25 Win=256 Len=0

c. Désencapsulation OSI

📌 Modèle OSI (Open Systems Interconnection)



Désencapsulation dans Wireshark

Wireshark permet d'analyser une **trame réseau** en "désencapsulant" chaque couche :

1. **Ouvrir une capture** (ou en démarrer une nouvelle).
2. **Sélectionner une trame** dans la liste.
3. **Observer le détail des couches** dans la fenêtre du milieu :

Le modèle OSI décrit les **7 couches** de communication réseau, chacune ayant un rôle spécifique :

1. **Physique (L1)** : Câbles, signaux électriques/optiques.
2. **Liaison de données (L2) : Ethernet (MAC)** → Gère les trames et adresses physiques (MAC).
3. **Réseau (L3) : IP** → Routage et adresses logiques (IP).
4. **Transport (L4) : TCP/UDP** → Gère les ports et la fiabilité des données.
5. **Session (L5) / Présentation (L6)** : Gèrent les connexions et le format des données (peu visibles dans Wireshark).
6. **Application (L7) : HTTP, DNS, FTP** → Données utilisables par les logiciels.

- a. **Couche 2 (Ethernet) :**
 - i. Adresses **MAC source/destination**.
 - ii. Type de protocole (ex : IPv4).
- b. **Couche 3 (IP) :**
 - i. Adresses **IP source/destination**.
 - ii. TTL, protocole (ex : TCP).
- c. **Couche 4 (TCP/UDP) :**
 - i. Ports **source/destination**.
 - ii. Numéros de séquence/acquittement (TCP).
- d. **Couche 7 (Application) :**
 - i. Données brutes (ex : requête **HTTP**, réponse **DNS**, **Données**).



2. Analyse des en-têtes

Différence trame/paquet et format PCAP

Une trame opère au niveau 2 (liaison) et contient des adresses MAC, tandis qu'un paquet opère au niveau 3 (réseau) et contient des adresses IP. Le format PCAP/PCAPNG est le format standard pour enregistrer des captures réseau.

Analyse des adresses

Pour voir les adresses MAC et IP :

1. Sélectionnez une trame
2. Développez la section "Ethernet II" pour les MAC

```
✗ Ethernet II, Src: HuiZhouGaosh_3b:38:f7 (64:57:25:3b:38:f7), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: HuiZhouGaosh_3b:38:f7 (64:57:25:3b:38:f7)
  Type: IPv4 (0x0800)
  [Stream index: 3]
```

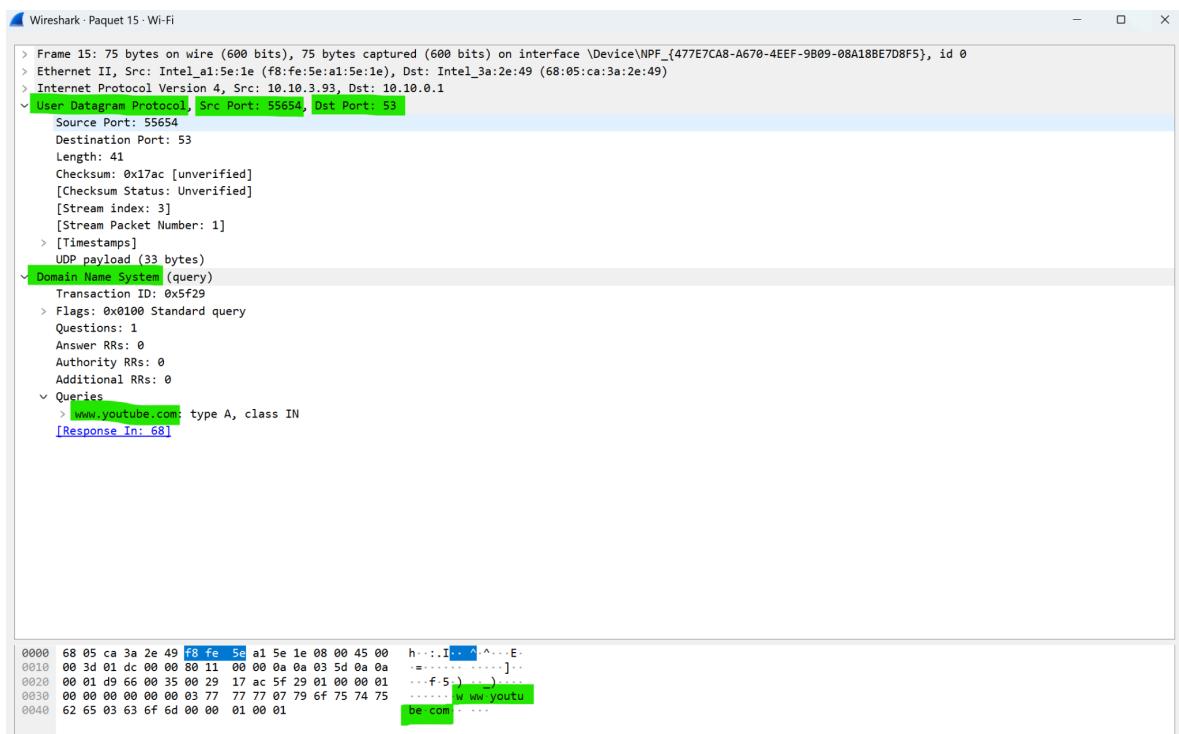
3. Développez la section "Internet Protocol" pour les IP

```
✗ Internet Protocol Version 4, Src: 10.10.6.13, Dst: 255.255.255.255
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 148
    Identification: 0xa556 (42326)
  > 010. .... = Flags: 0x2, Don't fragment
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: UDP (17)
    Header Checksum: 0x84ec [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.10.6.13
    Destination Address: 255.255.255.255
    [Stream index: 8]
```

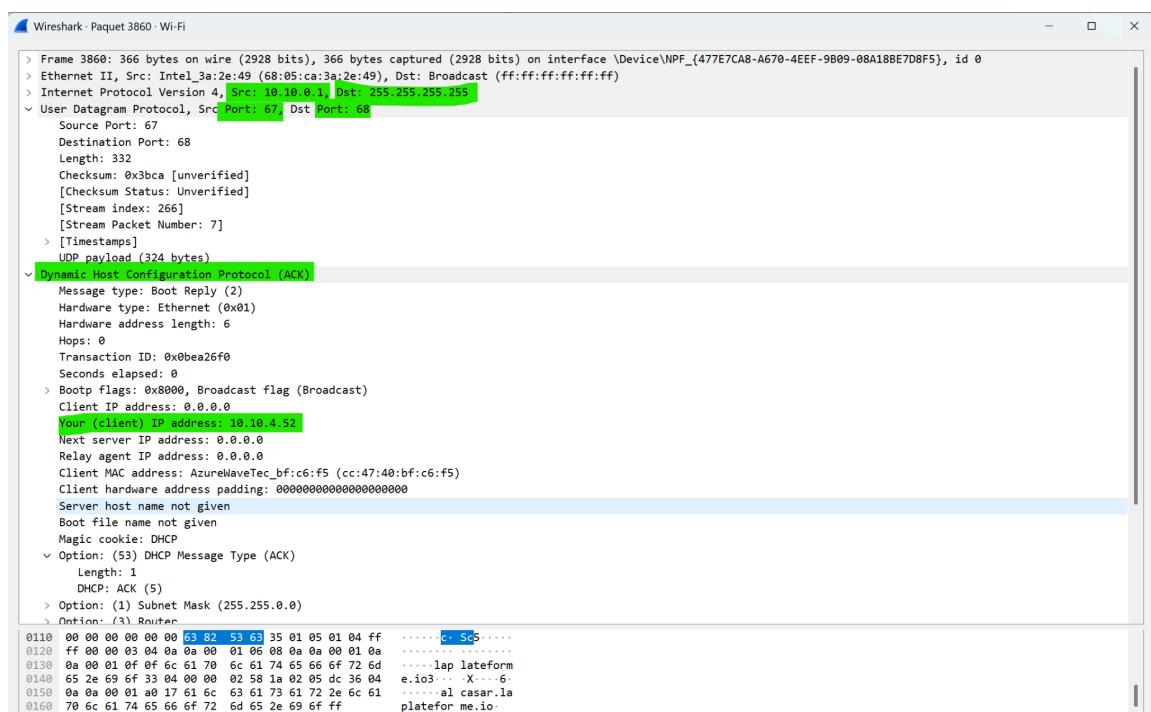
Identification d'autres trames

Exemples courants :

- DNS : Résolution de noms (udp port 53)



- DHCP : Attribution d'adresses IP : UDP (port 68 -67) DHCP ACK (réponse d'attribution IP)



Analyse hexadécimale

1. Sélectionnez une trame
 2. La partie inférieure montre les données en hexadécimal
 3. Comparez avec les spécifications des protocoles :
 - o ARP : Commence par 0001 (Ethernet) 0800 (IPv4)

Address Resolution Protocol (reply/gratuitous ARP)															
Hardware type: Ethernet (1)															
Protocol type: IPv4 (0x0800)															
00	ff	ff	ff	ff	ff	e4	24	6c	96	75	af	08	06	00 01\$1.u.....
10	08	00	06	04	00	02	e4	24	6c	96	75	af	0a	0a 01 d8\$1.u.....
20	00	00	00	00	00	00	0a	0a	01	d8	00	00	00	00 00 00
30	00	00	00	00	00	00	00	00	00	00	00	00	00	00 00 00 00
▼ Address Resolution Protocol (reply/gratuitous ARP)															
Hardware type: Ethernet (1)															
Protocol type: IPv4 (0x0800)															
0000	ff	ff	ff	ff	ff	ff	e4	24	6c	96	75	af	08	06 00 01\$1.u.....
0010	08	00	06	04	00	02	e4	24	6c	96	75	af	0a	0a 01 d8\$1.u.....
0020	00	00	00	00	00	00	0a	0a	01	d8	00	00	00	00 00 00
0030	00	00	00	00	00	00	00	00	00	00	00	00	00	00 00 00 00

- UDP : Structure simple (ports + longueur + checksum)

- TCP : Flags (SYN, ACK, etc.) dans le 13ème octet

Wireshark · Paquet 2306 · Wi-Fi

> Frame 2306: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{477E7CA8-A671-4A8C-BB8D-000000000000
> Ethernet II, Src: Intel_a1:5e:1e (f8:fe:5e:a1:5e:1e), Dst: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
> Internet Protocol Version 4, Src: 10.10.3.93, Dst: 142.250.201.46
< Transmission Control Protocol, Src Port: 6260, Dst Port: 443, Seq: 5914, Ack: 2791, Len: 0

Source Port: 6260
Destination Port: 443
[Stream index: 0]
[Stream Packet Number: 29]
> [Conversation completeness: Incomplete (28)]
[TCP Segment Len: 0]
Sequence Number: 5914 (relative sequence number)
Sequence Number (raw): 375481210
[Next Sequence Number: 5914 (relative sequence number)]
Acknowledgment Number: 2791 (relative ack number)
Acknowledgment number (raw): 1035508045
0101 = Header Length: 20 bytes (5)
> Flags: 0x010 (ACK)
Window: 255
[Calculated window size: 255]
[Window size scaling factor: -1 (unknown)]
Checksum: 0x65aa [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]

0000	68 05 ca 3a 2e 49 f8 fe 5e a1 5e 1e 08 00 45 00	h...I.. ^.^.. E.
0010	00 28 d9 7c 40 00 80 06 00 00 0a 0a 03 5d 8e fa	.(@.... .).
0020	c9 2e 18 74 01 bb 16 61 63 7a 3d b8 99 4d 50 10	.t...a cz... MP..
0030	00 ff 65 aa 00 00	...e...

< Transmission Control Protocol, Src Port: 443, Dst Port: 6342, Seq: 0, Ack: 1, Len: 0

Source Port: 443
Destination Port: 6342
[Stream index: 22]
[Stream Packet Number: 2]
> [Conversation completeness: Complete, WITH_DATA (31)]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 1035349829
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 2626156416
1000 = Header Length: 32 bytes (8)
> Flags: 0x012 (SYN, ACK)
Window: 26883
[Calculated window size: 26883]
Checksum: 0x8790 [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK perm:
> [Timestamps]
> [SEQ/ACK analysis]

0000	f8 fe 5e a1 5e 1e 68 05 ca 3a 2e 49 08 00 45 00	..^.^..h...I.. E.
0010	00 34 00 00 40 00 f4 06 23 dc 22 c2 32 bf 0a 0a	4...@... #.^..2...
0020	03 5d 01 bb 18 c6 3d b6 2f 45 9c 87 f7 80 80 12	...]..... /E.....
0030	69 03 87 90 00 00 02 04 05 b4 01 01 04 02 01 03	i.....

```

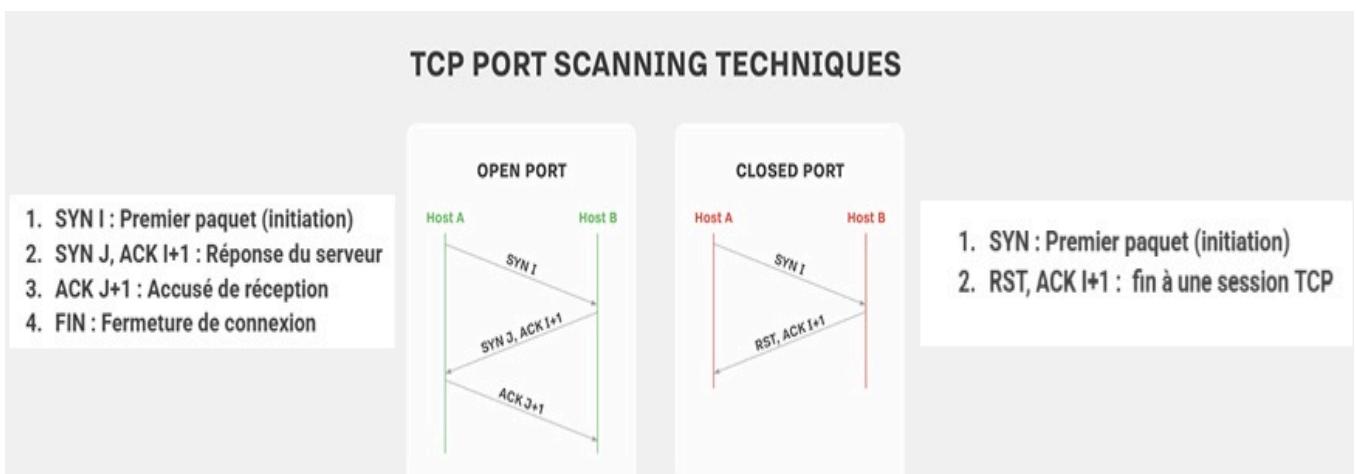
> Ethernet II, Src: Intel PRO (00:0c:29:2e:1e), Dst: Intel PRO (00:0c:29:2e:47)
> Internet Protocol Version 4, Src: 10.10.3.93, Dst: 95.101.110.203
> Transmission Control Protocol, Src Port: 6345, Dst Port: 80, Seq: 112, Ack: 189, Len: 0
    Source Port: 6345
    Destination Port: 80
    [Stream index: 23]
    [Stream Packet Number: 9]
    > [Conversation completeness: Complete, WITH_DATA (63)]
        [TCP Segment Len: 0]
        Sequence Number: 112      (relative sequence number)
        Sequence Number (raw): 2924545668
        [Next Sequence Number: 113      (relative sequence number)]
        Acknowledgment Number: 189      (relative ack number)
        Acknowledgment number (raw): 56285012
        0101 .... = Header Length: 20 bytes (5)
    > Flags: 0x011 (FIN, ACK)
        Window: 255
        [Calculated window size: 65280]
        [Window size scaling factor: 256]
        Checksum: 0xdbb1 [unverified]
        [Checksum Status: Unverified]
        Urgent Pointer: 0
    > [Timestamps]

0000  68 05 ca 3a 2e 49 f8 fe  5e a1 5e 1e 08 00 45 00  h...I..^.^...E.
0010  00 28 f6 9f 40 00 80 06  00 00 0a 0a 03 5d 5f 65  -(..@... ....]_e
0020  6e cb 18 c9 00 50 ae 51  06 84 03 5a d7 54 50 11  n....P.Q ...Z.TP.
0030  00 ff db b1 00 00          ..... .

```

3. Mécanisme de connexion TCP

Diagramme de connexion TCP :



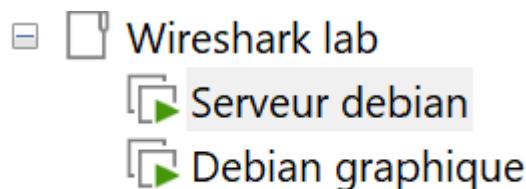
Le diagramme illustre les trois étapes essentielles du mécanisme de connexion TCP entre un client et un serveur.

- 1. SYN (Synchronize) :** ○ Client → Serveur : Le client envoie un paquet SYN pour initier la connexion, incluant un numéro de séquence initial (SEQ. Client).
- 2. SYN-ACK (Synchronize-Acknowledge) :** ○ Serveur → Client : Le serveur reçoit le paquet SYN et répond avec un paquet SYN-ACK. Ce paquet contient le numéro de séquence initial du serveur (SEQ. Serveur) et un accusé de réception pour le SYN du client (SEQ. Client + 1).
- 3. ACK (Acknowledge) :** ○ Client → Serveur : Le client accuse réception du paquet SYN-ACK en envoyant un paquet ACK. Ce paquet contient l'accusé de réception pour le SYN-ACK du serveur (SEQ. Serveur + 1) et le numéro de séquence du client (SEQ. Client + 1).

À ce stade, la connexion est établie, et le client peut optionnellement envoyer les premières données. Ce processus assure que les deux parties sont synchronisées et prêtes à échanger des données de manière fiable

Partie 2 : Capture Ciblée de Protocoles

1. Mise en place d'un lab réseau



J'installe et configure :

- **une WM Debian en graphique avec Wireshark pour voir les protocoles les données**

Une fois la VM installée, il faut installer Wireshark :

Shell:

sudo apt update

sudo apt install wireshark

et donner les droits à l'user pour qu'il puisse faire des captures.

Shell:

sudo usermod -aG wireshark \$USER

reboot

- **une VM serveur sous Debian avec les services (DHCP, DNS, DNS, SSL , FTP , SMB, HTTPS/TLSv1.2)**

j'installe les services :

shell:

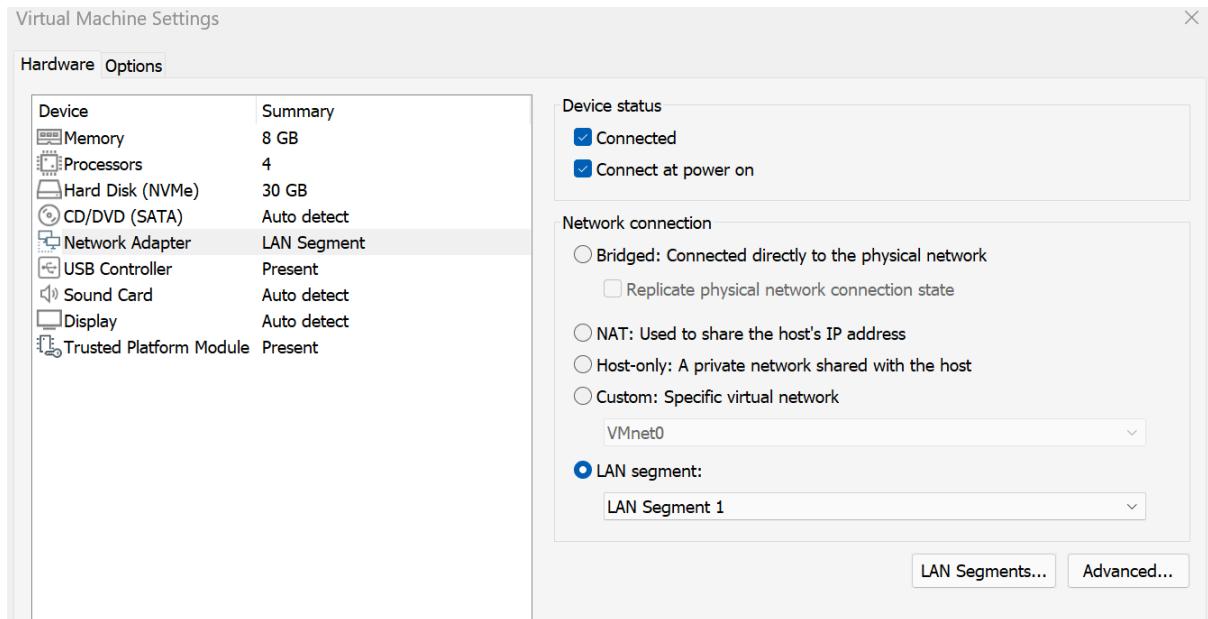
sudo apt upgrade

sudo apt install isc-dhcp-server bind9 vsftpd samba apache2 openssl

sudo apt-get install bind9 dnsutils

sudo apt install apache2

Je crée une LAN pour isoler mes deux VM qui permet de voir le trafic qu'entre les deux machines en local. (qui je mets qu'une fois que j'ai fini de tout installer et paramétré)



Je pense à vérifier que mes machines sont bien sur le même réseau

je peux passer à la configuration des services :

A. DHCP - Attribution d'IP

bash:

```
sudo nano /etc/default/isc-dhcp-server # Ubuntu/Debian
```

```
GNU nano 7.2          /etc/default/isc-dhcp-server *
# Defaults for isc-dhcp-server (sourced by /etc/init.d/isc-dhcp->

# Path to dhcpd's config file (default: /etc/dhcp/dhcpd.conf).
#DHCPDv4_CONF=/etc/dhcp/dhcpd.conf
#DHCPDv6_CONF=/etc/dhcp/dhcpd6.conf

# Path to dhcpd's PID file (default: /var/run/dhcpd.pid).
#DHCPDv4_PID=/var/run/dhcpd.pid
#DHCPDv6_PID=/var/run/dhcpd6.pid
```

Je modifie INTERFACESv4

Fichier à éditer : `/etc/dhcp/dhcp.conf`

`sudo nano /etc/dhcp/dhcp.conf`

exemple

```
# Options globales
default-lease-time 600;
max-lease-time 7200;
authoritative;

# Sous-réseau à desservir
subnet 192.168.20.130 netmask 255.255.255.0 {
    range 192.168.20.130 192.168.20.150;
    option routers 192.168.20.141;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "serveurdebian.local";
}
```

```

GNU nano 7.2          /etc/dhcp/dhcpd.conf *
# dhcpd.conf
#
# Sample configuration file for ISC dhcpcd
#
# Options globales
default-lease-time 600;
max-lease-time 7200;
authoritative;

# Sous-réseau à desservir
subnet 192.168.20.130 netmask 255.255.255.0 {
    range 192.168.20.130 192.168.20.150;
    option routers 192.168.20.141;
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "serveurdebian.local";
}

}

```

redémarrer le services

shell:

sudo systemctl restart isc-dhcp-server

B. DNS - Résolution de noms

Fichier : /etc/bind/named.conf.options

Ce fichier contient les options globales de BIND.

configuration :

```

options {
    directory "/var/cache/bind";

```

```
recursion yes;

allow-query { any; };

forwarders {

    8.8.8.8;

    8.8.4.4;

};

dnssec-validation auto;

listen-on { any; };

listen-on-v6 { any; };

};
```

Fichier : `/etc/bind/named.conf.local`

Définir les zones DNS dans `/etc/bind/named.conf.local` et créer les fichiers de zones.

conf

```
zone "serveurdebian.local" {

    type master;

    file "/etc/bind/zones/db.example.local";

};po
```

redémarrer le services

shell:

```
sudo systemctl restart bind9
```

C. mDNS - Multicast DNS

Installer **avahi-daemon** :

shell:

```
sudo apt install avahi-daemon
```

```
sudo systemctl start avahi-daemon
```

Cela permet à la machine d'être découverte en **.local.**

D. FTP - Transfert non chiffré

Installer et configurer **vsftpd**.

Modifier **/etc/vsftpd.conf** :

```
anonymous_enable=NO      # Désactive l'accès anonyme  
local_enable=YES        # Permet aux utilisateurs locaux de se connecter  
write_enable=YES        # Autorise l'écriture  
chroot_local_user=YES   # Restreint les utilisateurs à leur répertoire home  
local_umask=022         # Permissions des nouveaux fichiers
```

```

GNU nano 7.2                               /etc/vsftpd.conf *

anonymous_enable=NO      # Désactive l'accès anonyme
local_enable=YES         # Permet aux utilisateurs locaux de se connecter
write_enable=YES          # Autorise l'écriture
chroot_local_user=YES    # Restreint les utilisateurs à leur répertoire home
local_umask=022           # Permissions des nouveaux fichiers

# Example config file /etc/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.
#
#
# Run standalone?  vsftpd can run either from an inetd or as a standalone
# daemon started from an initscript.
listen=NO
#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "loopback" address (::) will accept connections from both IPv6

```

Créer un utilisateur pour tester :

sudo adduser testftp

sudo systemctl restart tftp

```

tom@debian:~$ ftp 192.168.20.141
Connected to 192.168.20.141.
220 (vsFTPd 3.0.3)
Name (192.168.20.141:tom): stom
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 

```

E. SMB - Partage Windows

sur le serveur :

installer samba

```
sudo apt install samba
```

Modifier `/etc/samba/smb.conf` pour ajouter un partage :

```
sudo nano /etc/samba/smb.conf
```

[partage]

```
path = /srv/samba/partage
browseable = yes
read only = no
valid users = nom_utilisateur
```

```
GNU nano 7.2                               /etc/samba/smb.conf *
[partage]
path = /srv/samba/partage
browseable = yes
read only = no
valid users = nom_utilisateur
```

Créer le dossier :

shell:

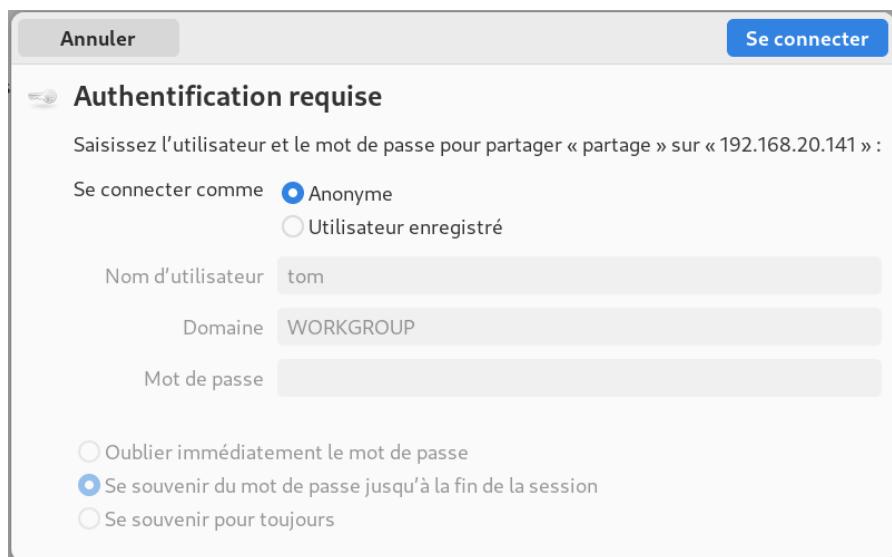
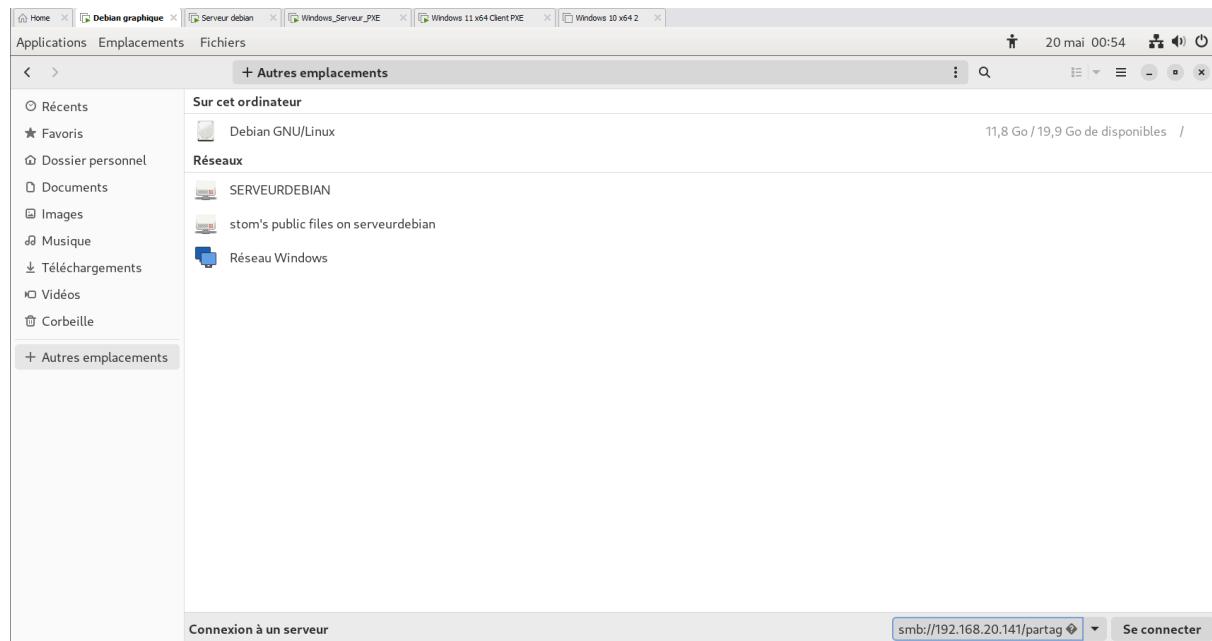
```
sudo mkdir -p /srv/samba/partage
```

```
sudo chmod 777 /srv/samba/partage
```

```
sudo systemctl restart smbd
```

maintenant je peux me connecter avec mon client:

smb://192.168.20.141/partage



F. HTTPS avec TLSv1.2

• Étape 1 : Installer Apache

sudo apt update

sudo apt install apache2

- **Étape 2 : Activer HTTPS avec SSL/TL**

-Créer un certificat SSL auto-signé (test):

```
sudo mkdir /etc/apache2/ssl
```

```
sudo openssl req -x509 -nodes -days 365 -newkey rsa:2048 \
```

**-keyout /etc/apache2/ssl/apache.key **

-out /etc/apache2/ssl/apache.crt

Répondez aux questions (Common Name = nom de domaine ou IP locale, ex: **localhost** ou **127.0.0.1**).

```
----  
Country Name (2 letter code) [AU]:FR  
State or Province Name (full name) [Some-State]:France  
Locality Name (eg, city) []:Marseille  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:laplateforme  
Organizational Unit Name (eg, section) []:lab  
Common Name (e.g. server FQDN or YOUR name) []:serveurdebian  
Email Address []:  
stom@serveurdebian:~$ |
```

-Configurer le virtual host HTTPS

Créez ou éditez ce fichier :

```
sudo nano /etc/apache2/sites-available/default-ssl.conf
```

Assurez-vous qu'il contient :

```
<VirtualHost *:443>
```

```
    ServerAdmin stom@serveurdebian.local
```

```
    DocumentRoot /var/www/html
```

```
    SSLEngine on
```

```
    SSLCertificateFile /etc/apache2/ssl/apache.crt
```

```
    SSLCertificateKeyFile /etc/apache2/ssl/apache.key
```

```
<Directory /var/www/html>
```

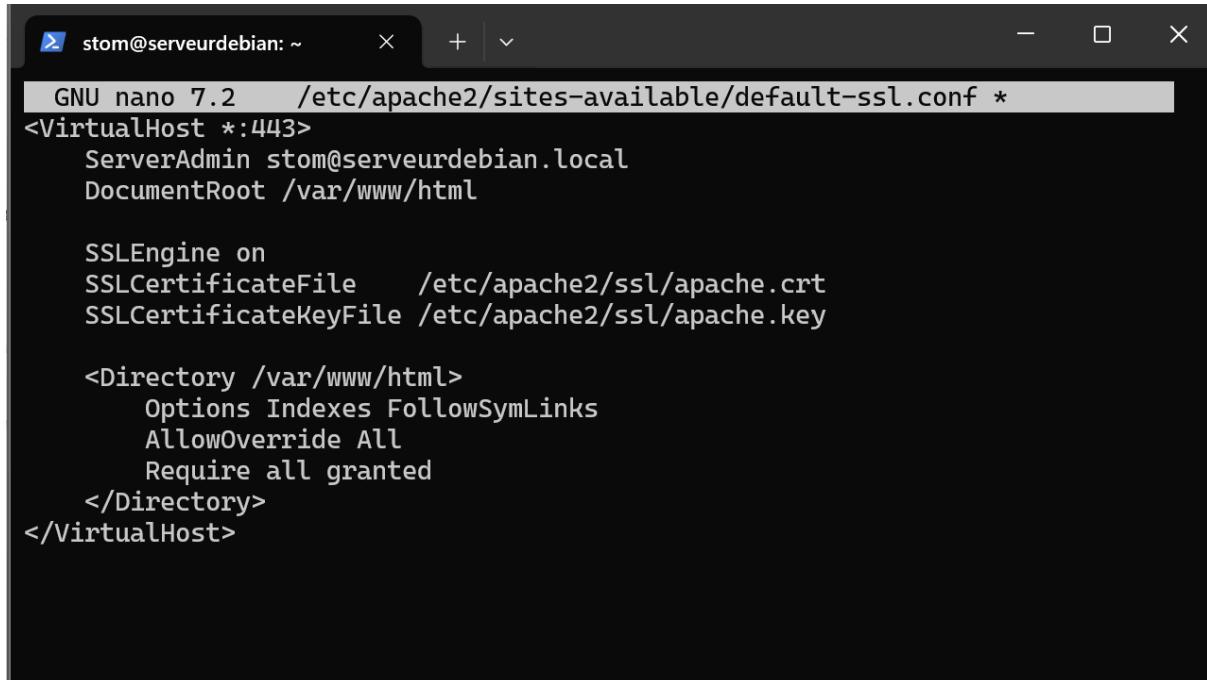
```
    Options Indexes FollowSymLinks
```

```
AllowOverride All
```

```
Require all granted
```

```
</Directory>
```

```
</VirtualHost>
```



```
GNU nano 7.2      /etc/apache2/sites-available/default-ssl.conf *
<VirtualHost *:443>
    ServerAdmin stom@serveurdebian.local
    DocumentRoot /var/www/html

    SSLEngine on
    SSLCertificateFile      /etc/apache2/ssl/apache.crt
    SSLCertificateKeyFile  /etc/apache2/ssl/apache.key

    <Directory /var/www/html>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

- Activer les modules et le site SSL:

```
sudo a2enmod ssl
```

```
sudo a2ensite default-ssl
```

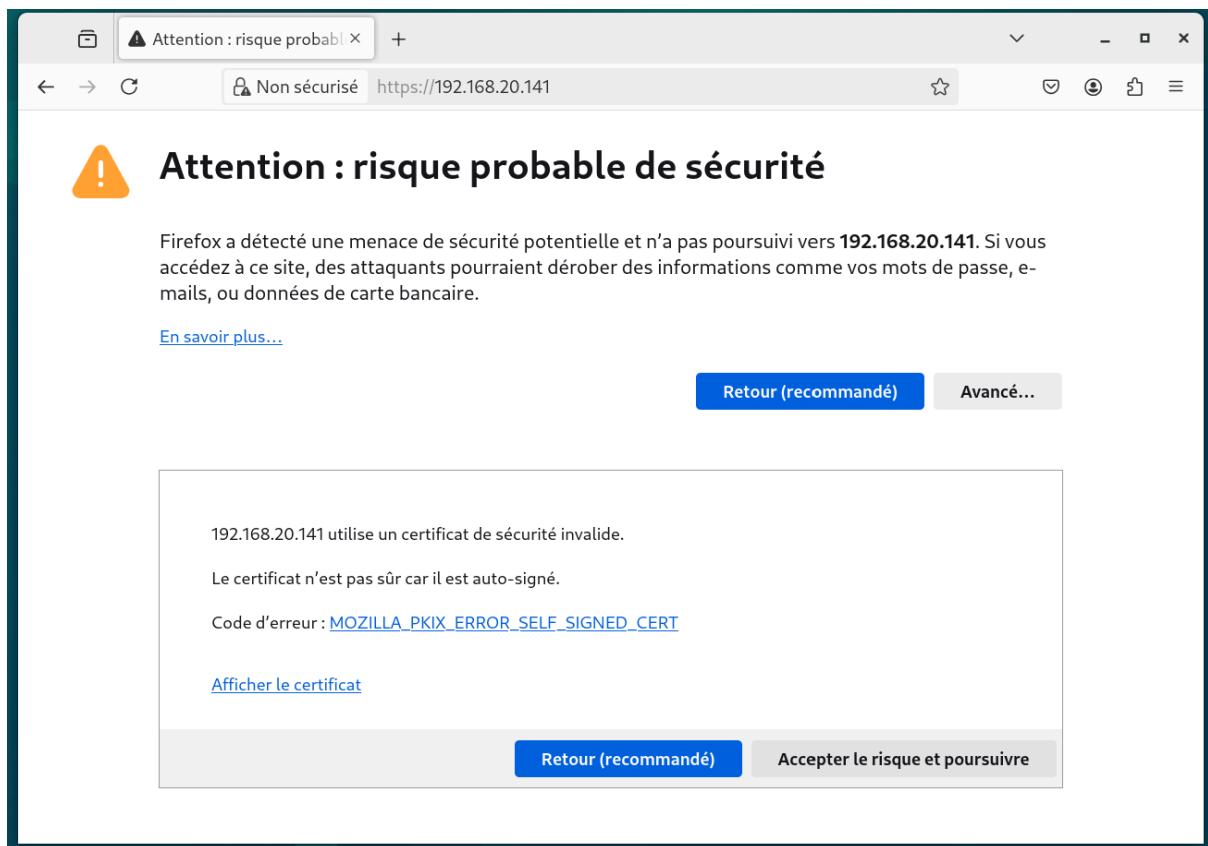
```
sudo systemctl restart apache2
```

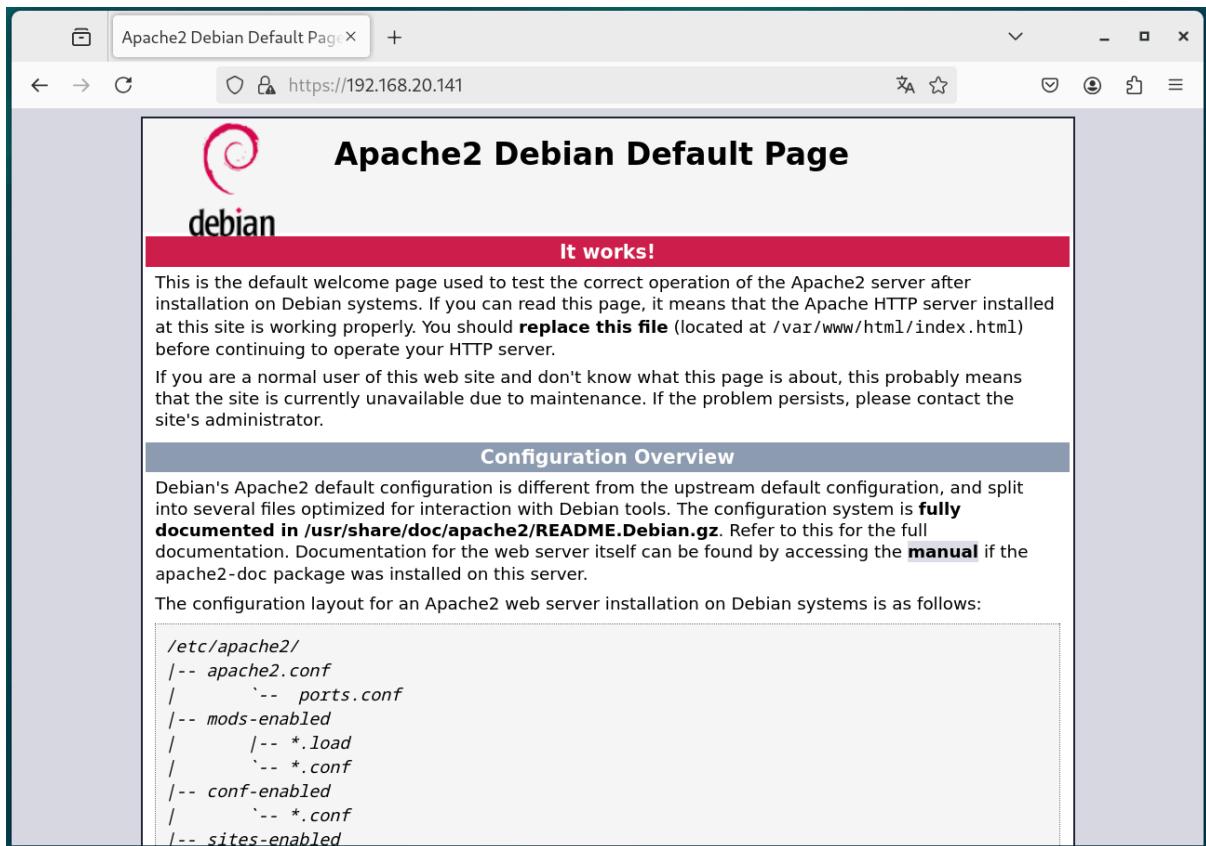
```
stom@serveurdebian:~$ sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and crea
te self-signed certificates.
To activate the new configuration, you need to run:
  systemctl restart apache2
stom@serveurdebian:~$ sudo a2ensite default-ssl
Enabling site default-ssl.
To activate the new configuration, you need to run:
  systemctl reload apache2
stom@serveurdebian:~$ sudo systemctl restart apache2
```

- **Étape 3 : Tester la connexion HTTPS**

Ouvrez votre navigateur et allez sur :

<https://192.168.20.141> ou <https://serveurdebian.local>





La connexion fonctionne.

2. Capture et analyse des protocoles

o DHCP (Attribution d'IP)

connecter une nouvelle machine au serveur

filtre dhcp

No.	Time	Source	Destination	Protocol	Length	Info
3136	-57.303695	0.0.0.0	255.255.255.255	DHCP	389	DHCP Discover - Transaction ID 0x5d9036e2
3137	-57.300325	192.168.20.133	255.255.255.255	DHCP	346	DHCP Offer - Transaction ID 0x5d9036e2
3155	-53.348382	0.0.0.0	255.255.255.255	DHCP	401	DHCP Request - Transaction ID 0x5d9036e2
3156	-53.344296	192.168.20.133	255.255.255.255	DHCP	346	DHCP ACK - Transaction ID 0x5d9036e2
3159	-53.341117	192.168.20.137	192.168.20.133	DHCP	389	proxyDHCP Request - Transaction ID 0x31159591
3167	-53.303491	192.168.20.133	192.168.20.137	DHCP	391	proxyDHCP ACK - Transaction ID 0x31159591
4539	-47.336239	192.168.20.137	192.168.20.133	DHCP	348	proxyDHCP Request - Transaction ID 0x40e20100
4541	-47.313681	192.168.20.133	192.168.20.137	DHCP	388	proxyDHCP ACK - Transaction ID 0x40e20100

DHCP Discover

Description : Le client, qui n'a pas encore d'adresse IP, envoie une requête DHCP Discover pour trouver un serveur DHCP sur le réseau.

- Adresse de Destination : (diffusion).

DHCP Offer

Description : Un serveur DHCP répond à la requête Discover en proposant une adresse IP au client.

- Adresse IP du Serveur: Par exemple, 192.168.20.133
- Adresse IP Proposée au Client: Par exemple, 192.168.20.137

DHCP Request

Description: Le client accepte l'offre en envoyant une requête

DHCP Request pour la confirmer.

- Adresse Source du Client: 0.0.0.0.
- Adresse de Destination: 255.255.255.255 (diffusion).

DHCP ACK

Description: Le serveur DHCP confirme l'attribution de l'adresse IP au client en envoyant une réponse

DHCP Request

Description: Le client accepte l'offre en envoyant une requête

DHCP Request pour la confirmer.

- Adresse Source du Client: **192.168.20.137**
- Adresse de Destination: **192.168.20.133**

DHCP ACK.

Description: L'adresse est acquise

- Adresse IP du Serveur: Par exemple, **192.168.20.133**.
- Adresse IP Attribuée au Client: Par exemple, **192.168.20.137**

- **DNS (Résolution de noms)**

filtre dns

Faire une requête DNS (ex: `dig`, `nslookup`)

501867	5226.371733	192.168.20.136	192.168.20.133	DNS	79 Standard query 0x0002 A pxe.local.pxe.local
501868	5226.373488	192.168.20.133	192.168.20.136	DNS	143 Standard query response 0x0002 No such name A pxe.local.pxe.l...
501869	5226.374286	192.168.20.136	192.168.20.133	DNS	79 Standard query 0x0003 AAAA pxe.local.pxe.local
501870	5226.375595	192.168.20.133	192.168.20.136	DNS	143 Standard query response 0x0003 No such name AAAA pxe.local.px...
501871	5226.376470	192.168.20.136	192.168.20.133	DNS	69 Standard query 0x0004 A pxe.local
501872	5226.377646	192.168.20.133	192.168.20.136	DNS	85 Standard query response 0x0004 A pxe.local A 192.168.20.133
501873	5226.378324	192.168.20.136	192.168.20.133	DNS	69 Standard query 0x0005 AAAA pxe.local
501874	5226.379476	192.168.20.133	192.168.20.136	DNS	124 Standard query response 0x0005 AAAA pxe.local SOA srv-pxe.pxe...
501878	5234.794840	192.168.20.136	192.168.20.133	DNS	74 Standard query 0xc88c A wpad.pxe.local
501879	5234.796783	192.168.20.133	192.168.20.136	DNS	138 Standard query response 0xc88c No such name A wpad.pxe.local ...

Les adresses sont :

Clients: 192.168.20.136

Serveurs : 192.168.20.133

qui s'inverse à tour de rôle

- **DNS Standard Query**

- **Source:** 192.16820.136
- **Destination:** 192.168.20.133
- **Description:**

Le client, dont l'adresse IP est 192.168.20.136, envoie une requête DNS de type A pour le domaine pxe.local au serveur DNS à l'adresse 192.168.20.133. La requête demande l'adresse IPv4 associée à ce domaine.

- **Détail:** La requête est identifiée par l'ID 0x0002.

- **DNS Standard Query Response**

- **Source:** 192.168.20.133
- **Destination:** 192.16820.136
- **Description:** Le serveur DNS (192.168.20.133) répond à la requête du client. Il retourne l'adresse IPv4 associée à www.example.com, qui est 192.168.20.133
- **Détail:** La réponse est identifiée par le même ID (0x002) et contient un enregistrement de type A pour pxe.local.

- **DNS Standard Query (AAAA)**

- **Source:** 192.168.20.136
- **Destination:** 192.168.20.133
- **Description:** Le client (192.16820.136) envoie une requête DNS de type AAAA pour pxe.local au serveur DNS (192.168.20.133). Cette requête demande l'adresse IPv6 du domaine.

- **DNS Standard Query Response (AAAA)**

- **Source:** 192.168.20.133
- **Destination:** 192.168.20.136
- **Description:** Le serveur DNS (192.168.65.133) répond à la requête du client. Il indique qu'il n'y a pas d'enregistrement AAAA pour www.example.com et retourne un enregistrement SOA (Start of Authority)

Résumé DNS

- Requêtes de type A: Le client demande l'adresse IPv4 de pxe.local et reçoit 192.168.65.133 en réponse.
 - Requêtes de type AAAA: Le client demande l'adresse IPv6 de local.pxe, mais reçoit un enregistrement SOA indiquant qu'il n'y a pas d'adresse IPv6 disponible pour ce domaine.
 - Options DNS Étendues (EDNS0): Le client utilise EDNS0 pour des fonctionnalités avancées, et le serveur répond en utilisant également EDNS0.
 - **mDNS (DNS multicast, ex: découverte de périphériques locaux)**
- filtrer mdns**

je me connecte a un fichier de partages depuis ma machine client

Time	Source	Destination	Type	Content
25759 1696.6851929...	fe80::20c:29ff:feab...	ff02::fb	MDNS	101 Standard query 0x0000 PTR _nmea-0183._tcp.local, "QM" question
25760 1696.6854637...	192.168.20.139	224.0.0.251	MDNS	81 Standard query 0x0000 PTR _nmea-0183._tcp.local, "QM" question
25761 1716.4527953...	192.168.20.139	224.0.0.251	MDNS	149 Standard query 0x0000 SRV stom's public files on serveurdebian._webdav._tcp.local, "Q...
25762 1716.4537802...	192.168.20.1	224.0.0.251	MDNS	149 Standard query 0x0000 SRV stom's public files on serveurdebian._webdav._tcp.local, "Q...
25763 1717.4123562...	fe80::20c:29ff:feab...	ff02::fb	MDNS	203 Standard query 0x0000 AAAA serveurdebian.local, "QM" question SRV stom's public files...
25764 1733.0928584...	fe80::20c:29ff:fe1f...	ff02::fb	MDNS	231 Standard query 0x0000 PTR _ftp._tcp.local, "QM" question PTR _nfs._tcp.local, "QM" qu...
25765 1733.0928593...	192.168.20.141	224.0.0.251	MDNS	211 Standard query 0x0000 PTR _ftp._tcp.local, "QM" question PTR _nfs._tcp.local, "QM" qu...
25766 1733.0941832...	192.168.20.1	224.0.0.251	MDNS	211 Standard query 0x0000 PTR _ftp._tcp.local, "QU" question PTR _nfs._tcp.local, "QU" qu...
25767 1734.6360304...	fe80::20c:29ff:fe1f...	ff02::fb	MDNS	194 Standard query 0x0000 SRV stom's public files on serveurdebian._webdav._tcp.local, "Q...
25768 1734.6360308...	192.168.20.141	224.0.0.251	MDNS	148 Standard query 0x0000 SRV stom's public files on serveurdebian._webdav._tcp.local, "Q...
25769 1734.6360309...	192.168.20.141	224.0.0.251	MDNS	187 Standard query response 0x0000 SRV, cache flush 0 0 37919 serveurdebian.local AAAA, c...
25770 1734.6360310...	192.168.20.1	224.0.0.251	MDNS	148 Standard query 0x0000 SRV stom's public files on serveurdebian._webdav._tcp.local, "Q...
25771 1734.6360311...	192.168.20.1	224.0.0.251	MDNS	187 Standard query response 0x0000 SRV, cache flush 0 0 37919 serveurdebian.local AAAA, c...
25772 1734.6594084...	fe80::20c:29ff:fe1f...	ff02::fb	MDNS	191 Standard query response 0x0000 AAAA, cache flush fe80::20c:29ff:fe1f:d300 SRV, cache ...
25773 1736.3998336...	192.168.20.141	192.168.20.254	DHCP	333 DHCP Request - Transaction ID 0xd698bc91
25774 1736.3998346...	192.168.20.254	192.168.20.141	DHCP	342 DHCP ACK - Transaction ID 0xd698bc91

MDNS (Multicast DNS)

MDNS est un protocole utilisé pour la résolution de noms de domaine dans des réseaux locaux sans avoir besoin d'un serveur DNS centralisé.

Il permet aux appareils sur un réseau local de s'interroger les uns les autres pour résoudre des noms d'hôtes en adresses IP.

- Fonctionnalité Principale:** MDNS permet à un appareil de demander l'adresse IP associée à un nom d'hôte local (comme "imprimante.local") et de recevoir une réponse directement des autres appareils sur le réseau qui connaissent cette information.
- Utilisation:** Couramment utilisé dans des environnements où il n'y a pas de serveur DNS centralisé, comme à la maison ou dans des petits bureaux. MDNS est souvent utilisé pour la découverte de services réseau comme les imprimantes, les services de partage de fichiers, et les appareils multimédia.

mDNS Query (Question)

- **Description :** Un client envoie une requête mDNS pour résoudre un nom de domaine en adresse IP ou pour découvrir des services sur le réseau local.
- **Adresse de Destination :** 224.0.0.251 (multicast, pour atteindre tous les appareils prenant en charge mDNS).
- **Exemple :**
 - Requête PTR pour `_ftp._tcp.local` ou `_nfs._tcp.local`.
 - Requête SRV pour découvrir des services comme `stom's public files on server/debian_webdev._tcp.local`.
 -

mDNS Response (Réponse)

- **Description :** Un appareil ou service répond à la requête mDNS en fournissant les informations demandées (adresse IP, nom d'hôte, détails de service).
- **Adresse Source :** L'adresse IP de l'appareil/service répondant (par exemple, `192.168.20.141`).
- **Exemple :**
 - Réponse SRV avec des détails de service (port, nom d'hôte).
 - Réponse AAAA avec une adresse IPv6 (par exemple, `fe80::20c:29ff:fe1f:d396`).
 -

Caractéristiques Clés :

- **Multicast :** Les requêtes/réponses utilisent l'adresse multicast `224.0.0.251` pour une communication locale sans serveur DNS centralisé.
- **Cache Flush :** Les réponses peuvent inclure un indicateur "cache flush" pour forcer la mise à jour des caches des autres appareils.
- **Services :** Permet la découverte de services (par exemple, partage de fichiers, imprimantes) via des enregistrements PTR, SRV, et TXT.

- **FTP (Transfert de fichiers non chiffré)**

filtre ftp

Connexion FTP avec FileZilla ou ligne de commande.

commande:

ftp 192.168.20.141

No.	Time	Source	Destination	Protocol	Length	Info
28495	6359.3378534...	192.168.20.141	192.168.20.139	FTP	86	Response: 220 (vsFTPD 3.0.3)
28501	6375.4099679...	192.168.20.139	192.168.20.141	FTP	76	Request: USER tom
28503	6375.4113252...	192.168.20.141	192.168.20.139	FTP	100	Response: 331 Please specify the password.
28505	6379.6113546...	192.168.20.139	192.168.20.141	FTP	77	Request: PASS poiу
28507	6383.0358239...	192.168.20.141	192.168.20.139	FTP	88	Response: 530 Login incorrect.
28509	6388.2293386...	192.168.20.139	192.168.20.141	FTP	72	Request: QUIT
28511	6388.2313506...	192.168.20.141	192.168.20.139	FTP	80	Response: 221 Goodbye.
28519	6389.5314491...	192.168.20.141	192.168.20.139	FTP	86	Response: 220 (vsFTPD 3.0.3)
28523	6393.4354539...	192.168.20.139	192.168.20.141	FTP	77	Request: USER stom
28525	6393.4365326...	192.168.20.141	192.168.20.139	FTP	100	Response: 331 Please specify the password.
28527	6396.0492726...	192.168.20.139	192.168.20.141	FTP	77	Request: PASS poiу
28529	6396.1413241...	192.168.20.141	192.168.20.139	FTP	89	Response: 230 Login successful.
28531	6396.1417781...	192.168.20.139	192.168.20.141	FTP	72	Request: SYST
28533	6396.1423630...	192.168.20.141	192.168.20.139	FTP	85	Response: 215 UNIX Type: L8
28534	6396.1425323...	192.168.20.139	192.168.20.141	FTP	72	Request: FEAT
28535	6396.1431258...	192.168.20.141	192.168.20.139	FTP	81	Response: 211-Features:
28536	6396.1431260...	192.168.20.141	192.168.20.139	FTP	87	Response: EPRT
28538	6396.1433918...	192.168.20.141	192.168.20.139	FTP	110	Response: PASV

▶ Frame 28523: 77 bytes on wire (616 bits), 77 bytes captured (616 bits) on interface eth0, Intel/Profinet Adapter, link layer type: Ethernet II (0x0806), source: VMware_ab:64:f4 (00:0c:29:a0:64:f4), destination: 192.168.20.141 (00:0c:29:1f:d3:00)	0000 00 0c 29 1f d3 00 00 0c 29 ab 64 f4 08 00 45 10 ..).....)-d-E-
▶ Internet Protocol Version 4, Src: 192.168.20.139, Dst: 192.168.20.141 (0.0.0.0), Len: 616	0010 00 3f dd bc 40 00 40 06 b2 83 c0 a8 14 8b c0 a8 ..?..@..
▶ Transmission Control Protocol, Src Port: 33420, Dst Port: 21 (0.0.0.0), Len: 600	0020 14 8d 82 92 00 15 46 0e a4 06 64 59 56 0c 80 18 ..F..dYV.. . . .
▶ File Transfer Protocol (FTP)	0030 40 00 aa 9a 00 00 01 01 08 0a 57 ca d5 e8 b8 ca @.....W.. . .
[Current working directory:]	0040 79 45 55 53 45 52 20 73 74 6f 6d 0d 0a yEUSER s tom..

Détails des Paquets

- 1. Paquet 1 : Réponse du serveur FTP avec le message d'accueil "220 (vsFTPD 3.0.3)".**
- 2. Paquet 2 : Requête USER avec "stom" comme nom d'utilisateur envoyé en clair.**
- 3. Paquet 3 : Réponse du serveur "331 Please specify the password.", demandant le mot de passe.**
- 4. Paquet 4 : Requête PASS avec "poiу" comme mot de passe envoyé en clair.**
- 5. Paquet 5 : Réponse "230 Login successful.", indiquant une connexion réussie.**

6. Paquets suivants : Contiennent des commandes et réponses FTP supplémentaires (SYST, FEAT, EPRT, PASV).

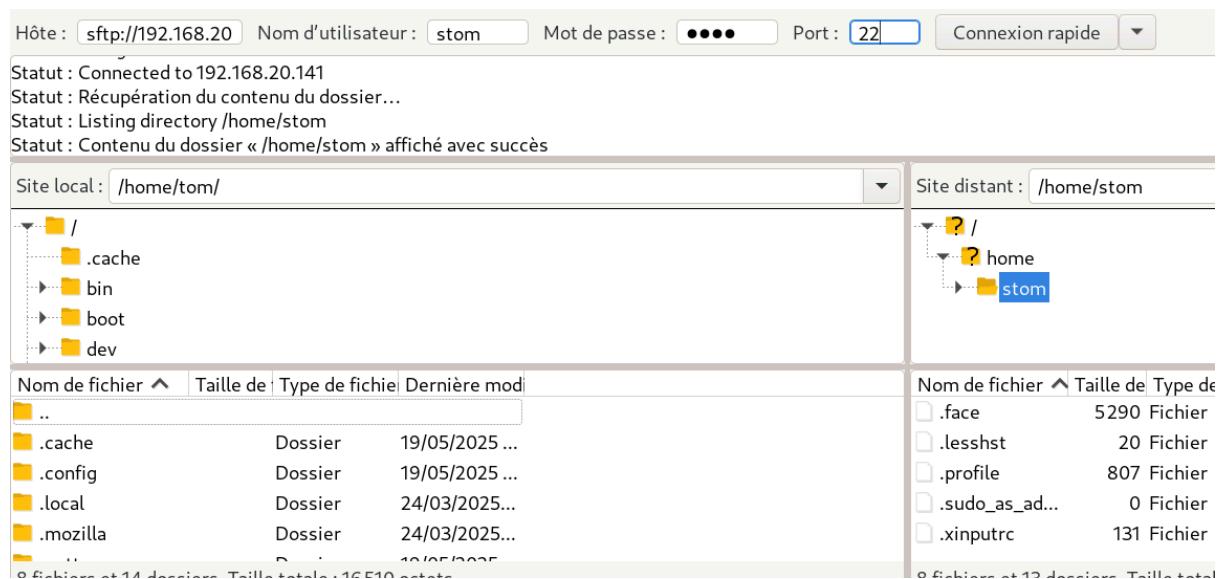
Interprétation Données Sensibles :

- Les informations de connexion, y compris le nom d'utilisateur (ted) et le mot de passe (123), sont visibles en clair dans les paquets capturés. Récupération de Données :
- Oui, il est possible de récupérer des données sensibles, y compris les identifiants de connexion, car elles ne sont pas chiffrées.

Sécurité

: • L'absence de chiffrement signifie que toute personne ayant accès au réseau peut intercepter et lire ces informations.

○ SFTP (avec filezilla)



28746	7229.6083568...	192.168.20.139	192.168.20.141	SSHv2	92 Client: Protocol (SSH-2.0-FileZilla_3.63.0)
28748	7229.6391642...	192.168.20.141	192.168.20.139	SSHv2	106 Server: Protocol (SSH-2.0-OpenSSH_9.2p1 Debian-2+deb12u6)
28750	7229.6414380...	192.168.20.141	192.168.20.139	SSHv2	1202 Client: Key Exchange Init
28752	7229.6454290...	192.168.20.139	192.168.20.141	SSHv2	1442 Client: Key Exchange Init
28753	7229.645881...	192.168.20.139	192.168.20.141	SSHv2	114 Client: Elliptic Curve Diffie-Hellman Key Exchange Init
28755	7229.6554373...	192.168.20.141	192.168.20.139	SSHv2	598 Server: Elliptic Curve Diffie-Hellman Key Exchange Reply, New Keys, Encrypted pack
28757	7231.4573240...	192.168.20.139	192.168.20.141	SSHv2	134 Client: New Keys, Encrypted packet (plaintext_len=36), Unknown (140)
28758	7231.4588535...	192.168.20.141	192.168.20.139	SSHv2	118 Encrypted packet (plaintext_len=36), Unknown (115)[Malformed Packet]
28760	7231.4716800...	192.168.20.139	192.168.20.141	SSHv2	134 Encrypted packet (plaintext_len=52), Unknown (158)[Malformed Packet]
28761	7231.4820168...	192.168.20.141	192.168.20.139	SSHv2	118 Encrypted packet (plaintext_len=36), Unknown (165)[Malformed Packet]
28762	7231.4826982...	192.168.20.139	192.168.20.141	SSHv2	330 Encrypted packet (plaintext_len=164), Unknown (167)[Malformed Packet]
28764	7231.5580555...	192.168.20.141	192.168.20.139	SSHv2	102 Encrypted packet (plaintext_len=20), Unknown (235)[Malformed Packet]
28765	7231.5592108...	192.168.20.139	192.168.20.141	SSHv2	118 Encrypted packet (plaintext_len=36)[Malformed Packet]
28767	7231.6539845...	192.168.20.141	192.168.20.139	SSHv2	694 Server: Encrypted packet (plaintext_len=612), Unknown (171)
28769	7231.6982621...	192.168.20.141	192.168.20.139	SSHv2	118 Encrypted packet (plaintext_len=36), Unknown (168)[Malformed Packet]
28771	7231.6986763...	192.168.20.139	192.168.20.141	SSHv2	202 Encrypted packet (plaintext_len=68), Unknown (124)[Malformed Packet]
28773	7231.7011453...	192.168.20.141	192.168.20.139	SSHv2	138 Encrypted packet (plaintext_len=20)[Malformed Packet]
28774	7231.7015780...	192.168.20.139	192.168.20.141	SSHv2	118 Encrypted packet (plaintext_len=36), Unknown (156)[Malformed Packet]
28775	7231.7114158...	192.168.20.141	192.168.20.139	SSHv2	422 Encrypted packet (plaintext_len=340), Disconnect[Malformed Packet]
28776	7231.7116218...	192.168.20.139	192.168.20.141	SSHv2	118 Encrypted packet (plaintext_len=36), Unknown (175)[Malformed Packet]
28777	7231.7127188...	192.168.20.141	192.168.20.139	SSHv2	150 Encrypted packet (plaintext_len=68), Unknown (124)[Malformed Packet]
28778	7231.7156650...	192.168.20.139	192.168.20.141	SSHv2	134 Client: Encrypted packet (plaintext_len=52), Unknown (134)
28779	7231.7167621...	192.168.20.141	192.168.20.139	SSHv2	150 Server: Encrypted packet (plaintext_len=68), Unknown (102)
28780	7231.7169688...	192.168.20.139	192.168.20.141	SSHv2	134 Encrypted packet (plaintext_len=52), Unknown (211)[Malformed Packet]
28781	7231.7184843...	192.168.20.141	192.168.20.139	SSHv2	118 Server: Encrypted packet (plaintext_len=36), Unknown (133)
28782	7231.7184843...	192.168.20.139	192.168.20.141	SSHv2	274 Encrypted packet (plaintext_len=52), Unknown (211)[Malformed Packet]

Détails des Paquets SFTP (FileZilla)

1. Paquets 2014b et 28748 :

- **Client (192.168.20.139) : Annonce la version du protocole SSH (SSH-2.0-FilledAlla_5.63.0).**
- **Serveur (192.168.20.141) : Répond avec sa version (SSH-2.0-OpenSSH_9.2p1 Debian-24de01206).**

2. Paquets 28750 et 28752 :

- **Échange des paramètres de Key Exchange Init entre le serveur et le client pour établir une connexion sécurisée.**

3. Paquets 28753 et 28755 :

- **Client : Initie un échange de clés Elliptic Curve Diffie-Hellman.**
- **Serveur : Répond avec une clé de chiffrement, active le chiffrement (New Keys), et envoie un paquet chiffré.**

4. Paquets 28756 à 28781 :

- **Communication chiffrée après l'établissement de la session SSH. Les paquets sont marqués comme "Encrypted packet" avec des tailles de plaintext indiquées (ex: plaintext_len=36).**
- **Certains paquets sont annotés comme "[Walformed Packet]" ou "Unknown", suggérant des commandes ou données internes au SFTP (non lisibles sans la clé de session).**

Interprétation Données Sensibles :

- Aucune donnée sensible en clair : Contrairement à FTP, SFTP chiffre toute la communication (y compris identifiants et fichiers transférés).
 - Les paquets après le Key Exchange sont illisibles sans la clé de session.

Récupération de Données :

- **✗ Impossible de récupérer des identifiants ou fichiers directement depuis les paquets capturés (chiffrement SSH).**

Sécurité :

-  Conforme aux bonnes pratiques :
 - Utilisation de SSH-2.0 avec ECDH (échange de clés sécurisé).
 - Chiffrement des données dès l'établissement de la session.
 -  Remarque : La présence de "Unknown" peut indiquer des extensions ou versions spécifiques de logiciels (à vérifier pour des vulnérabilités connues).

○ **SMB (Partage de fichiers Windows)**

filtre smb

Se connecter à un partage SMB (`smbclient //ip/share`).

No.	Time	Source	Destination	Protocol	Length	Info
25791	1774.8742647...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
26089	2494.5307009...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
26241	3213.2682215...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
26622	3935.3905590...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
26984	4456.7711794...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
27144	5375.0615088...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
27917	6994.9116284...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
28648	6817.1847601...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
28865	7538.7731145...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
29097	8259.8755024...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
33571	8831.8946568...	192.168.20.141	192.168.20.255	BROWSER	263	Host Announcement SERVEURDEBIAN, Workstation, Server, Print Queue Server, Xenix Serve...
33720	8838.9884065...	192.168.20.141	192.168.20.255	BROWSER	238	Browser Election Request
33721	8840.9110743...	192.168.20.141	192.168.20.255	BROWSER	238	Browser Election Request
33722	8842.9135470...	192.168.20.141	192.168.20.255	BROWSER	238	Browser Election Request
33728	8844.9161132...	192.168.20.141	192.168.20.255	BROWSER	238	Browser Election Request
33729	8846.9187257...	192.168.20.141	192.168.20.255	BROWSER	238	Browser Election Request
33738	8854.9342898...	192.168.20.141	192.168.20.255	BROWSER	226	Request Announcement
33739	8854.9356552...	192.168.20.141	192.168.20.255	BROWSER	263	Local Master Announcement SERVEURDEBIAN, Workstation, Server, Print Queue Server, Xen...
33740	8854.9352672...	192.168.20.141	192.168.20.255	BROWSER	256	Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Enum
34138	8981.3805327...	192.168.20.1	192.168.20.255	BROWSER	243	Host Announcement DESKTOP-DVJ6KFA, Workstation, Server, NT Workstation
34139	8981.3807680...	192.168.20.141	192.168.20.255	BROWSER	263	Local Master Announcement SERVEURDEBIAN, Workstation, Server, Print Queue Server, Xen...
34140	8981.3807689...	192.168.20.141	192.168.20.255	BROWSER	256	Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Enum
34650	9171.4921910...	192.168.20.141	192.168.20.255	BROWSER	263	Local Master Announcement SERVEURDEBIAN, Workstation, Server, Print Queue Server, Xen...
34651	9171.4929297...	192.168.20.141	192.168.20.255	BROWSER	256	Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Enum
35045	9411.6896548...	192.168.20.141	192.168.20.255	BROWSER	263	Local Master Announcement SERVEURDEBIAN, Workstation, Server, Print Queue Server, Xen...
L	35046.9411.6996671...	192.168.20.141	192.168.20.255	BROWSER	256	Domain/Workgroup Announcement WORKGROUP, NT Workstation, Domain Enum

Analyse des Paquets SMB (BROWSER)

Détails des Paquets BROWSER

Paquets 25791 à 29697:

- **Source:** 192.168.20.1 (probablement un contrôleur de domaine ou serveur principal)
- **Destination:** Broadcast (192.168.20.255)
- **Protocole:** BROWSER (Service de découverte réseau Windows)
- **Contenu:**
 - "Host Announcement" pour la machine "DESKTOP-DVJ&KFA"
 - Rôles déclarés: Workstation, Server, NT Workstation
 - Longueur fixe: 243 octets
 - Intervalle régulier (~300-500 unités de temps entre chaque annonce)

Paquets 29731 à 35946:

- **Source:** 192.168.20.141 (une autre machine du réseau)
- **Destination:** Broadcast (192.168.20.255)
- **Protocole:** BROWSER
- **Contenu:**
 - "Browser Election Request" pour la même machine "DESKTOP-DVJ&KFA"
 - Rôles étendus: Workstation, Server, Print Queue Server, Xenix Server
 - Longueur légèrement différente: 238 octets
 - Envois très rapprochés (suggère une compétition pour le rôle de maître navigateur)

Interprétation Technique

Comportement Normal:

- Les annonces de host sont normales pour un réseau Windows (toutes les 12 minutes environ)
- L'élection de navigateur fait partie du fonctionnement SMB pour désigner qui maintient la liste des partages réseau

Éléments Notables:

⚠️ Conflit Potentiel:

- Deux machines (192.168.20.1 et 192.168.20.141) annoncent le même nom "DESKTOP-DVJ&KFA"
- Cela pourrait indiquer:
 - Un clone de machine virtuelle
 - Une configuration réseau incorrecte
 - Une tentative d'usurpation (moins probable)

⚠️ Trafic Broadcast Excessif:

- Les paquets d'élection sont envoyés en rafale (potentiel symptôme de problème réseau)

Sécurité:

- 🔒 Données exposées:
 - Noms de machine et rôles visibles en clair
 - Pas d'informations d'authentification visibles dans ces paquets
- ⚠️ Risques Potentiels:
 - Découverte facile des machines réseau par un attaquant
 - Possibilité de perturbation du service en interférant avec les élections de navigateur

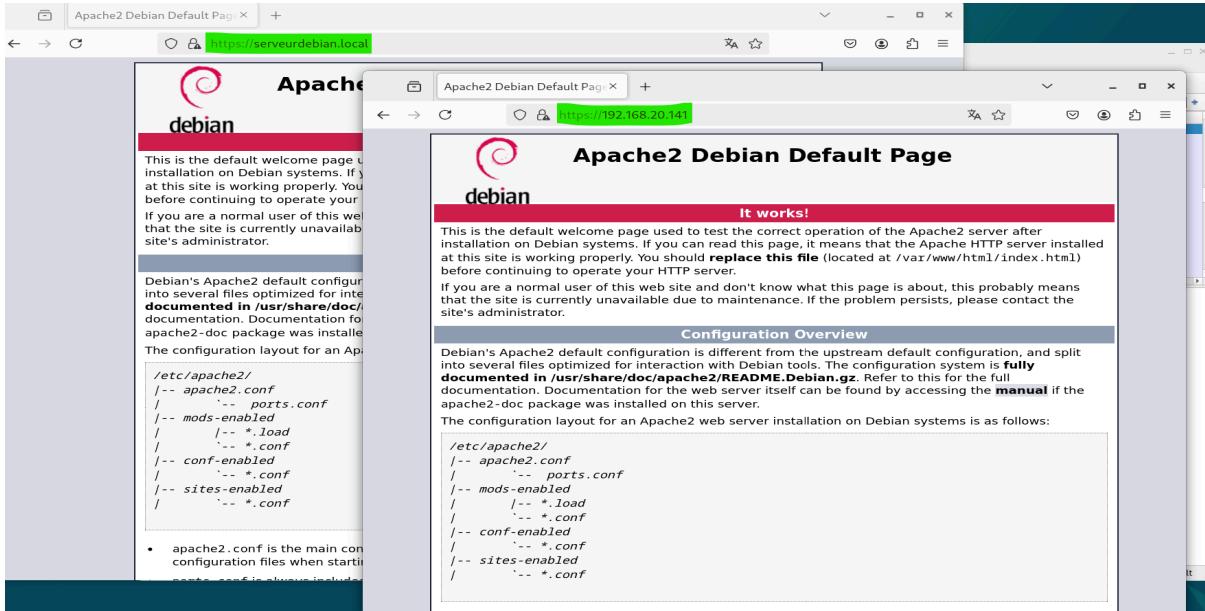
Recommandations:

1. Vérifier la configuration des deux machines partageant le même nom
2. Surveiller les élections de navigateur fréquentes (peut indiquer une instabilité réseau)
3. Si possible, limiter le trafic BROWSER aux segments réseau nécessaires

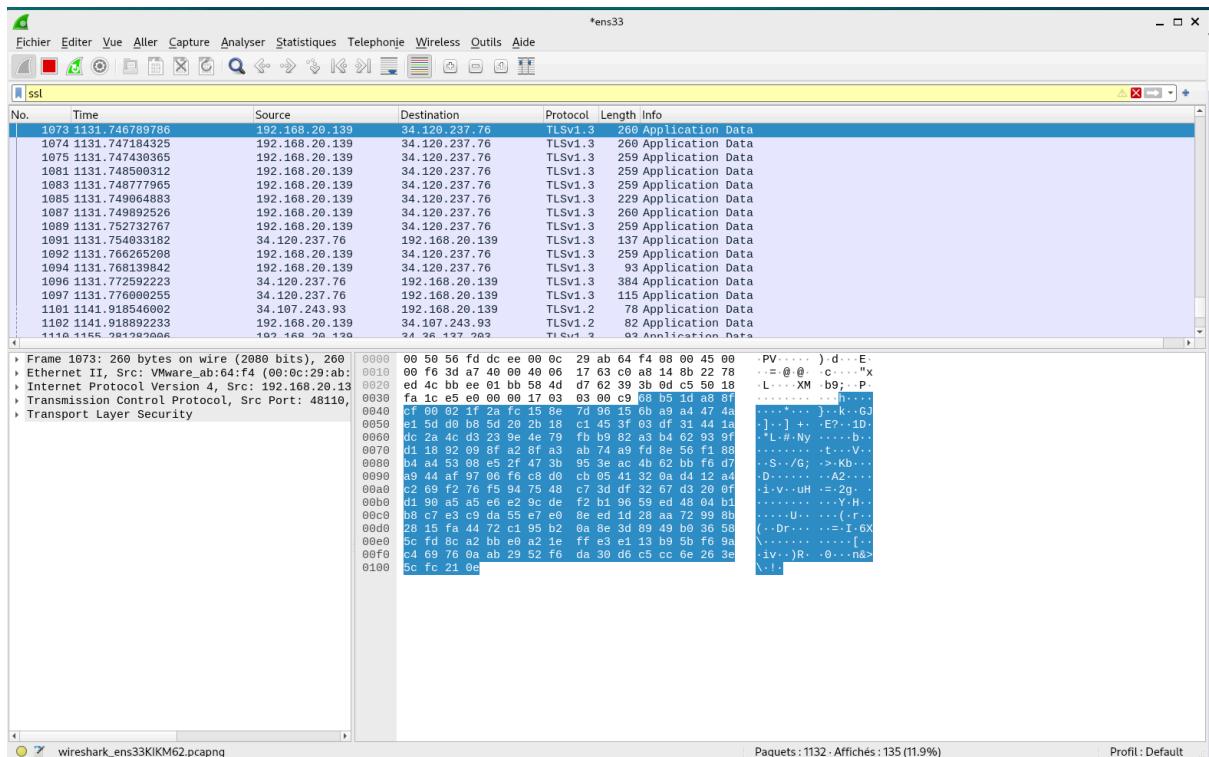
- **HTTPS/TLSv1.2 (Chiffrement SSL/TLS)**

Utiliser un navigateur pour aller sur <https://192.168.20.141> ou

<https://serveurdebian.local/>



wireshark: filtre tls ou ssl



Analyse de Connexion HTTPS/TLS avec Apache

Détails des Paquets HTTPS (Apache)

Établissement de la Connexion TLS 1.2/1.3

Paquets d'Initialisation:

- **Client (192.168.20.139) → Serveur (34.120.237.76):**
 - "Client Hello" avec versions TLS supportées (dont TLS 1.2/1.3)
 - Suites cryptographiques proposées (ex: ECDHE-ECDSA-AES256-GCM-SHA384)
 - Client Random (32 bytes)

Réponse du Serveur:

- **Serveur (34.120.237.76) → Client (192.168.20.139):**
 - "Server Hello" sélectionnant TLS 1.3 (visible dans votre capture)
 - Certificat SSL du serveur (Apache)
 - Server Random (32 bytes)
 - Paramètres de chiffrement sélectionnés

Échange de Clés (Key Exchange)

- **Client → Serveur:**
 - Clé pré-maîtresse chiffrée avec la clé publique du certificat
 - Changement de spécification de chiffrement
- **Serveur → Client:**
 - "New Session Ticket" pour la reprise de session
 - Activation du chiffrement ("Change Cipher Spec")

Communication Chiffrée

Paquets 1074 à 1083:

- Application Data (taille ~260-269 bytes)
- Tous les paquets sont marqués "TLSv1.3" et "Application Data"
- Données entièrement chiffrées avec AES-GCM ou autre algorithme négocié

Interprétation Technique

Sécurité

- Chiffrement fort (TLS 1.3 avec suites modernes)
- Perfect Forward Secrecy (ECDHE utilisé)
- Certificat valide (non visible mais implicite dans l'établissement TLS)

Données Sensibles

- Impossible d'extraire:
 - Headers HTTP
 - Cookies de session
 - Données POST (formulaires)
 - URLs demandées (après le SNI initial)

Particularités Apache

- Configuration visible dans la négociation:
 - Préférence pour TLS 1.3 si disponible (rétrocompatible avec 1.2)
 - Suite cryptographique probablement configurée via `SSLCipherSuite` dans `httpd.conf`
 - Pas de vulnérabilités apparentes (pas de SSLv3, TLS 1.0/1.1)

Remarque:

Votre capture montre bien que:

1. Apache préfère TLS 1.3 quand disponible (plus sécurisé que 1.2)
2. Toute la charge utile HTTP est chiffrée après le handshake
3. Les tailles de paquets sont cohérentes avec du trafic HTTPS classique

Risques à Mettre en Évidence

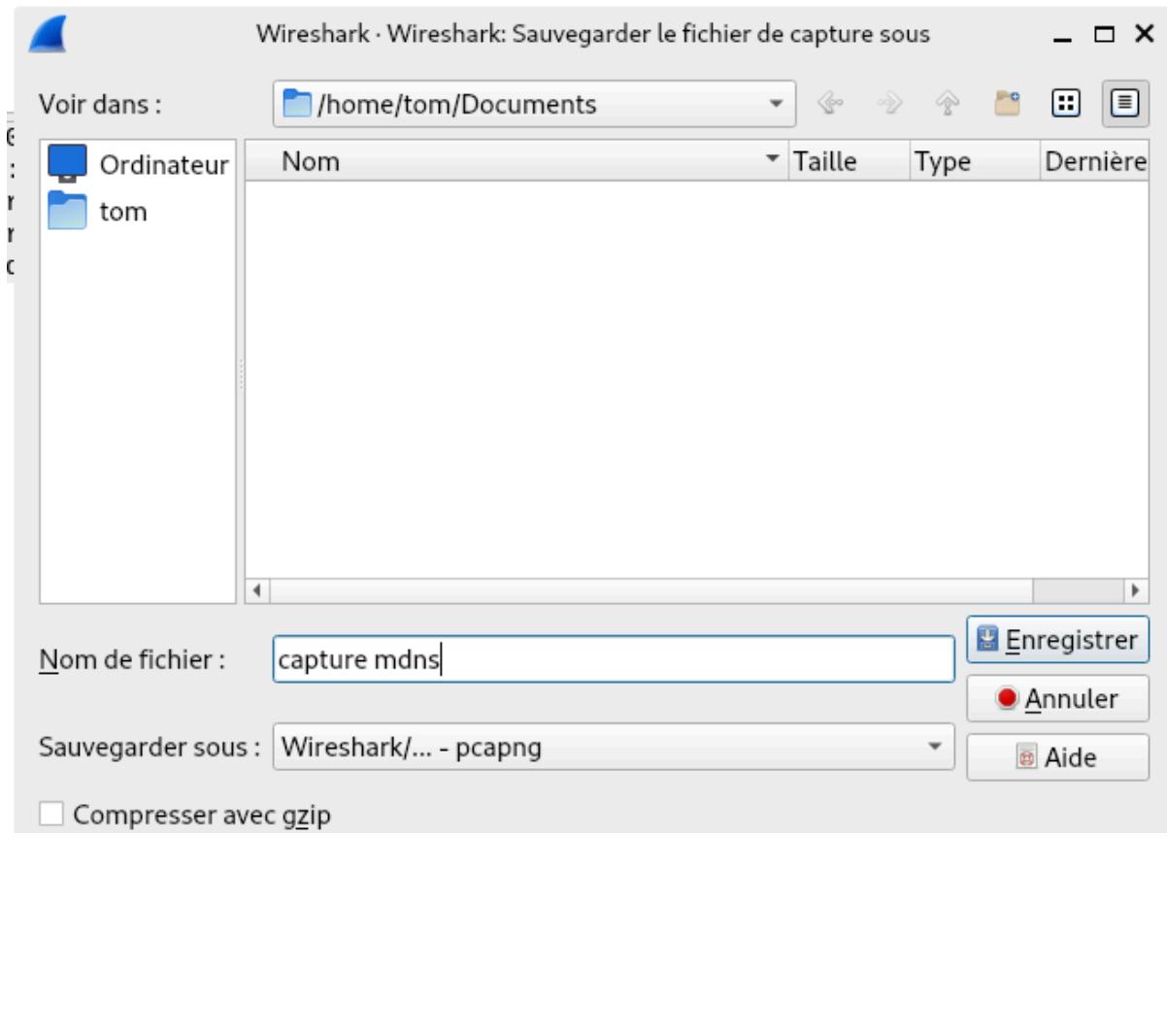
- **FTP : identifiants et fichiers passent en clair → interceptables.**
- **SMB : dépend de la config, mais peut exposer des fichiers ou credentials.**
- **HTTPS : communications chiffrées, comparer avec http et ftp**
- **mDNS : possible fuite d'informations sur les périphériques locaux.**

3. Sauvegarde des paquets pertinents

- **Export des captures au format PCAPNG**

je peux filtrer les paquets que je veux enregistrer pour les analyses plus tard ou les garder.

Je peux également les enregistrés sous différents formats si besoin



Partie 3 : Automatisation avec Tshark

- Installation et prise en main de Tshark

- Commande : `sudo apt install tshark`

```
tom@debian:~$ sudo apt install tshark
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Le paquet suivant a été installé automatiquement et n'est plus nécessaire :
  linux-image-6.1.0-32-amd64
Veuillez utiliser « sudo apt autoremove » pour le supprimer.
Les NOUVEAUX paquets suivants seront installés :
  tshark
0 mis à jour, 1 nouvellement installés, 0 à enlever et 2 non mis à jour.
Il est nécessaire de prendre 163 ko dans les archives.
Après cette opération, 406 ko d'espace disque supplémentaires sero
```

Pour démarrer tshark:

sudo tshark -i ens33

```
tom@debian:~$ sudo tshark -i ens33
[sudo] Mot de passe de tom :
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:5649) 23:28:14.352235 [Main MESSAGE] -- Capture started.
** (tshark:5649) 23:28:14.352355 [Main MESSAGE] -- File: "/tmp/wireshark_ens33711X62.pcapng"
 1 0.000000000 fe80::20c:29ff:fe1f:d300 → ff02::fb      MDNS 155 Standard query 0x0000 AAAA
serveurdebian.local, "QM" question SRV stom's public files on serveurdebian._webdav._tcp.local
, "QM" question
 2 0.000281009 fe80::20c:29ff:fe1f:d300 → ff02::fb      MDNS 191 Standard query response 0x0
000 SRV, cache flush 0 0 37919 serveurdebian.local AAAA, cache flush fe80::20c:29ff:fe1f:d300
^Ctshark:
2 packets captured
```

- **Commandes de capture avancées**

- **Exemples :**

```
root@debian:/home/tom# sudo tshark -i ens33 -f "tcp port 80" -w httpcap.pcapng
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
```

```
# Capturer du HTTP
tshark -i eth0 -f "tcp port 80" -w http_capture.pcapng
```

```
# Filtrer les requêtes DNS
```

- `tshark -i eth0 -Y "dns" -T fields -e dnsqry.name`
- **Options clés :**
 - `-i` : Interface réseau
 - `-f` : Filtre BPF (Berkeley Packet Filter)
 - `-Y` : Filtre d'affichage
 - `-w` : Sauvegarde dans un fichier

Example de capture:

```
sudo tshark -i ens33 -Y 'ftp.request.command == "USER" || ftp.request.command == "PASS"' -T fields -e frame.time -e ip.src -e ftp.request.command -e ftp.request.arg -W ftplog.pcapng
```

🔍 Explication des options :

Option	Description
<code>-i eth0</code>	Interface réseau à écouter (remplace par ton interface réelle)
<code>-Y 'ftp.request.command == "USER"</code>	
<code>-T fields</code>	Affichage sous forme de champs
<code>-e frame.time</code>	Affiche le timestamp
<code>-e ip.src</code>	IP source de la requête
<code>-e ftp.request.command</code>	Affiche la commande FTP (<code>USER</code> ou <code>PASS</code>)
<code>-e ftp.request.arg</code>	Affiche l'argument de la commande (nom d'utilisateur ou mot de passe)

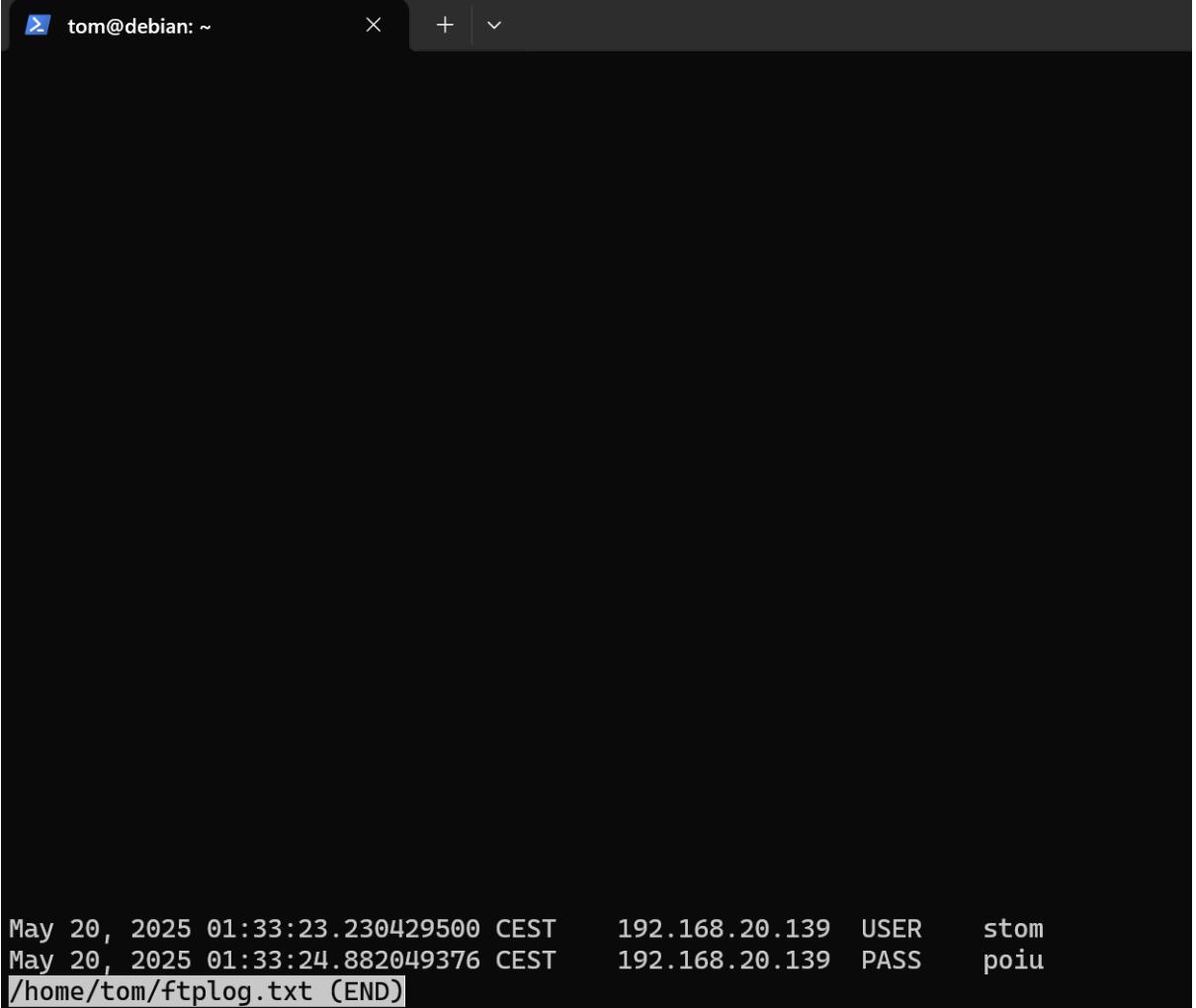
```
root@debian:/home/tom# sudo tshark -i ens33 -Y 'ftp.request.command == "USER" || ftp.request.command == "PASS"' -T fields -e frame.time -e ip.src -e ftp.request.command -e ftp.request.arg
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:6651) 01:20:11.153821 [Main MESSAGE] -- Capture started.
** (tshark:6651) 01:20:11.154135 [Main MESSAGE] -- File: "/tmp/wireshark_ens334EXQ62.pcapng"
^Ctshark:
0 packets captured
root@debian:/home/tom# sudo tshark -i ens33 -Y 'ftp.request.command == "USER" || ftp.request.command == "PASS"' -T fields -e frame.time -e ip.src -e ftp.request.command -e ftp.request.arg
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:6680) 01:20:39.340357 [Main MESSAGE] -- Capture started.
** (tshark:6680) 01:20:39.340544 [Main MESSAGE] -- File: "/tmp/wireshark_ens330EHX62.pcapng"
May 20, 2025 01:21:00.426959157 CEST    192.168.20.139  USER      stom
May 20, 2025 01:21:02.683560856 CEST    192.168.20.139  PASS      poi
|
```

en modifiant ma ligne de commande et en rajoutant **-e ftp.request.arg**
> ftelog.txt

cela me permet de l'enregistrer dans un fichier .txt

```
sudo tshark -i ens33 -Y 'ftp.request.command == "USER" ||
ftp.request.command == "PASS"' -T fields -e frame.time -e ip.src -e
ftp.request.command -e ftp.request.arg > ftelog.txt
```

```
root@debian:/home/tom# sudo tshark -i ens33 -Y 'ftp.request.command == "USER" || ftp.request.command == "PASS"' -T fields -e frame.time -e ip.src -e ftp.request.command -e ftp.request.arg > ftelog.txt
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:6892) 01:37:16.156539 [Main MESSAGE] -- Capture started.
** (tshark:6892) 01:37:16.156689 [Main MESSAGE] -- File: "/tmp/wireshark_ens33CAJQ62.pcapng"
4 ^C
tshark:
root@debian:/home/tom# open ftelog.txt
```



A screenshot of a terminal window titled "tom@debian: ~". The window shows a single line of text from an FTP log file:

```
May 20, 2025 01:33:23.230429500 CEST 192.168.20.139 USER stom  
May 20, 2025 01:33:24.882049376 CEST 192.168.20.139 PASS poiu  
/home/tom/ftplog.txt (END)
```

 **Ce qui se passe quand un utilisateur se connecte en FTP :**

- Un client FTP se connecte à un serveur FTP non chiffré (FTP simple, pas FTPS/SFTP).
- Le client envoie :
 - La commande **USER nom_utilisateur**

- Puis PASS `mot_de_passe`
- **tshark** intercepte ces deux commandes FTP si elles passent par l'interface que tu surveilles.
- Tu verras apparaître immédiatement dans le terminal :
-

 **Attention importante :**

- Cette méthode ne fonctionne que sur du FTP non chiffré.
- Le FTP classique (port 21) transmet les identifiants en clair → c'est pourquoi **tshark** peut les voir.
- Si le serveur utilise FTPS ou SFTP, les données sont chiffrées, et tu ne pourras rien capturer en clair.

● Redirection et traitement des données

Exemple :

commande:

```
tshark -i eth0 -Y "ftp" | grep "USER\|PASS" #  
Extraction des identifiants FTP
```

-i eth0: Capture sur l'interface réseau eth0 (peut être ens33, wlan0, etc.)

-Y "ftp": Filtre d'affichage Wireshark : ne montre que les paquets FTP

grep "USER|PASS": Permet d'enregistré les nom user et les mots de passe non chiffré dans FTP

Conclusion

- Synthèse des apprentissages
- Importance du filtrage pour l'analyse efficace
- Risques de sécurité (ex: FTP non chiffré vs. HTTPS)
- Applications pratiques (Dépannage réseau, audit de sécurité)