

Network sniffing

Lucas Savioz
Youcef Ouarhlent
Jordan Reinaldo

Sommaire du projet

Partie 1

Introduction à Wireshark

Concepts de base en analyse réseau

Analyse des paquets réseau

3.1 Installation de wireshark

3.2 Analyse des paquets sur le réseau Alcasar

3.3 Installation wireshark (Windows)

Capture des paquets réseau

4.1 Paquets ARP

4.2 Paquets UDP

4.3 Paquets TCP

4.4 Mise en place d'un diagramme pour décrire le mécanisme de connexion

Désencapsuler les trames pour retrouver les différentes couches du modèle OSI

5.1 Désencapsulation des trames

5.2 Identification des couches OSI

Partie 2

1. Analyse des protocoles spécifiques sur un réseau local

2. Installation et configuration de la VM Serveur

2.1 Serveur DHCP

2.2 Capture des paquets

3. Installation et configuration de la VM Client



Partie 1

1. Introduction à Wireshark

Wireshark est un outil de capture et d'analyse de paquets open source. Il produit une analyse détaillée de la pile de protocoles réseau. Il permet de filtrer le trafic pour des opérations de dépannage du réseau, d'enquêter sur des problèmes de sécurité et d'analyser les protocoles réseau.

Étant donné que Wireshark vous permet d'afficher les informations sur les paquets, il peut être utilisé comme un outil de reconnaissance. Il capture le trafic du réseau local et stocke les données ainsi obtenues pour permettre leur analyse hors ligne. Wireshark est capable de capturer le trafic Ethernet, Bluetooth, sans fil (IEEE. 802.11), Token Ring, Frame Relay et plus encore.

Wireshark utilise la bibliothèque logicielle Qt pour l'implémentation de son interface utilisateur et pcap pour la capture des paquets ; il fonctionne sur de nombreux environnements compatibles UNIX comme GNU/Linux, FreeBSD, NetBSD, OpenBSD ou Mac OSX, mais également sur Microsoft Windows. Il existe aussi entre autres une version en ligne de commande nommée TShark. Ces programmes sont distribués gratuitement sous la licence GNU General Public License.

2. Concepts de base en analyse réseau

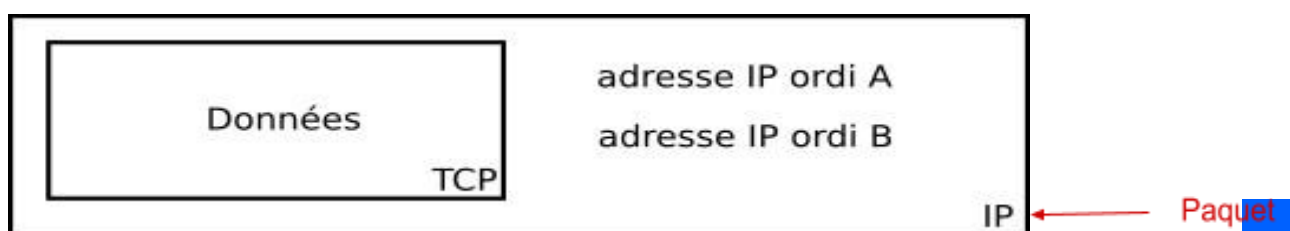
C'est quoi une trame ? C'est quoi un paquet ?

Les paquets IP ne peuvent pas transiter sur un réseau tel quel, ils vont eux aussi être encapsulés avant de pouvoir "voyager" sur le réseau. L'encapsulation des paquets IP produit ce que l'on appelle une trame.

Si vous utilisez un réseau filaire avec des câbles Ethernet (avec des prises RJ45), la trame sera de type Ethernet (ce qui est le cas pour le réseau d'un lycée). Si vous utilisez un réseau sans fil Wifi, la trame sera de type Wifi.

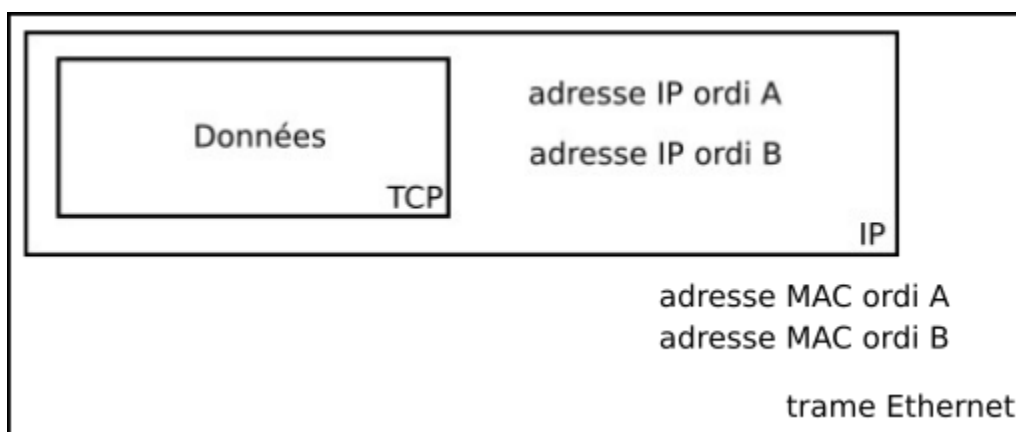
En fait, la trame Wifi ressemble beaucoup à la trame Ethernet, on peut même dire que la trame Wifi est la variante sans-fil de la trame Ethernet, afin de simplifier les choses, dans la suite, nous évoquerons uniquement la trame Ethernet en ayant à l'esprit que ce qui est dit sur la trame Ethernet est aussi valable pour la trame Wifi.

Nous savons que le paquet IP contient les adresses IP de l'émetteur et du récepteur :





Le paquet IP étant encapsulé par la trame Ethernet, les adresses IP ne sont plus directement disponibles (il faut désencapsuler le paquet IP pour pouvoir lire ces adresses IP), nous allons donc trouver un autre type d'adresse qui permet d'identifier l'émetteur et le récepteur : l'adresse MAC (Media Access Control) aussi appelée adresse physique.



Qu'est-ce que le format pcap/pcapng ?

pcap (« packet capture ») :

Format de capture, les fichiers de type pcap contiennent les captures d'un trafic on l'utilise pour une analyse ultérieure. Les fichiers .pcap sont les plus courants et sont généralement compatibles avec un large éventail d'analyseurs de réseau et d'autres outils.

pcapng (packet capture next generation) :

Le format ".pcapng" reprend le format simple ".pcap" avec de nouveaux champs et de nouvelles fonctionnalités. Chaque fichier pcapng contient plusieurs blocs ou données, qui contiennent différents types d'informations. Des exemples de blocs incluent le bloc d'en-tête de section, le bloc de description d'interface, le bloc de paquets amélioré, le bloc de paquets simple, le bloc de résolution de nom et le bloc de statistiques d'interface.

Ces blocs peuvent être utilisés pour reconstituer les paquets capturés dans des données reconnaissables. Pcap-NG est conçu pour le format de capture de paquets Pcap (.PCAP) précédent. Wireshark a commencé à utiliser le format de fichier Pcap-NG dans la version 1.8.



3. Analyse des paquets réseau

3.1 Installation de Wireshark (Linux)

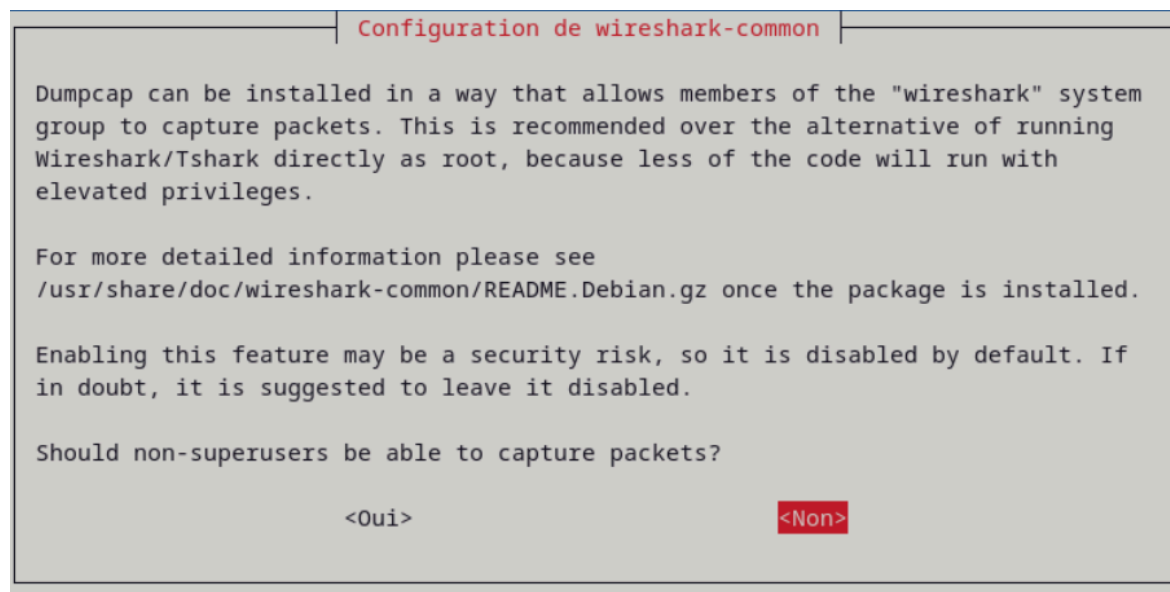
Faire une mise à jour avant de commencer :

```
root@debianServer:~# apt update && sudo apt upgrade
```

Procédez à l'installation :

```
root@debianServer:~# apt install wireshark
```

Pendant l'installation, une invite peut sembler demander si les non-super-utilisateurs doivent être autorisés à exécuter Wireshark. Cette décision dépend des autorisations du système nécessaires pour faire fonctionner la demande et doit être évaluée en tenant compte de nos exigences de sécurité. Nous mettons “**Oui**” puis nous allons configurer qui a droit à son utilisation.



```
server@debianServer:~$ sudo usermod -aG wireshark server
```

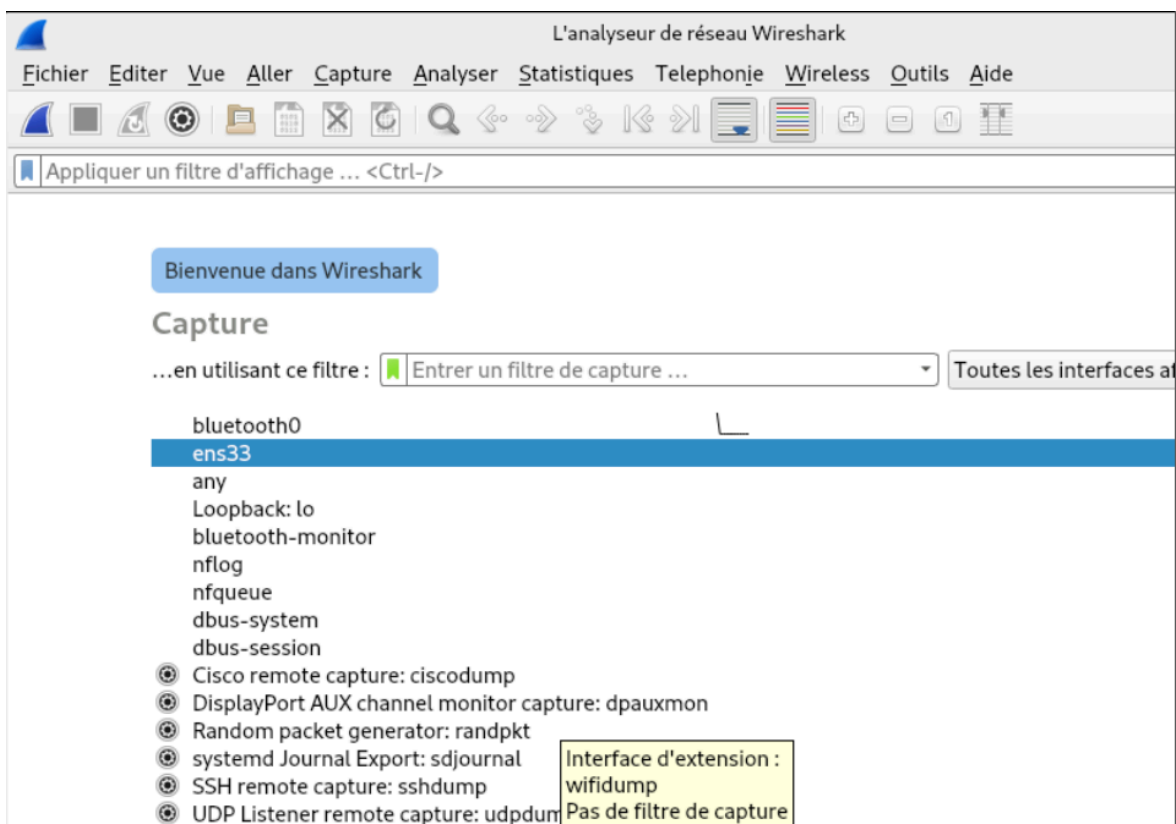
Une fois fait nous vérifions les membres du groupe wireshark :



```
server@debianServer:~$ getent group wireshark
wireshark:x:122:server
```

Ensuite lancer la commande **wireshark** sur le terminal avec le membre du groupe pour analyser les réseaux :

```
server@debianServer:/$ wireshark
** (wireshark:6709) 15:49:18.915435 [GUI WARNING] -- QSocketNotifier: Can only be used
with threads started with QThread
```

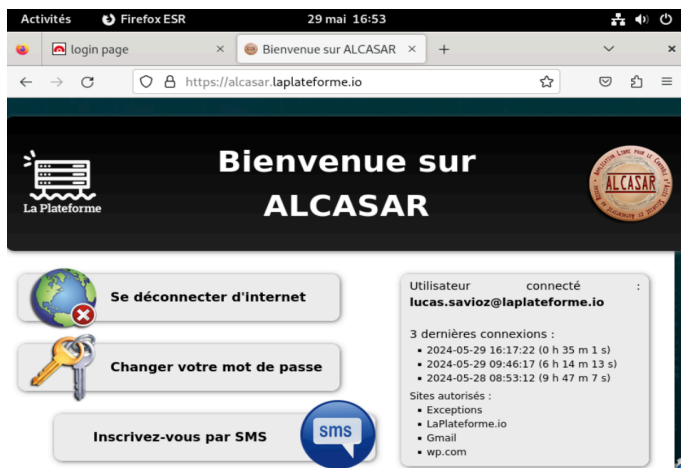


Nous allons sélectionner notre carte réseau “ens33” pour analyser son trafic.



3.2 Analyse des paquets sur le réseau Alcasar

Connexion à la page :



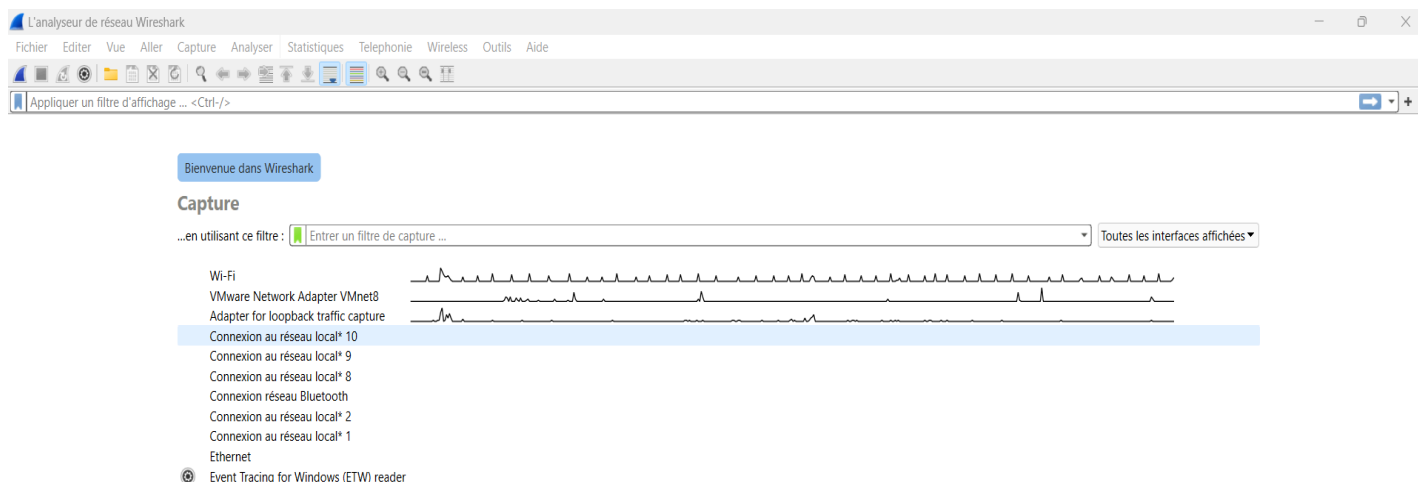
Nous écoutons les informations transmises par la page entre le réseau et la machine.

3.3 Installation wireshark (Windows)

Télécharger la dernière version de wireshark via le site officiel et suivre les étapes :

<https://www.wireshark.org/download.html>

Une fois cela fait, ouvrir wireshark :



Nous pouvons constater différents types de réseau dans notre environnement. A l'école La Plateforme nous utilisons le wifi pour nous connecter, donc pour capturer les connexions au portail Alcasar nous sélectionnons "wifi".

Vérifier votre adresse IPv4 de la machine windows ainsi que celle de la passerelle du réseau La Plateforme pour comprendre les captures de paquets.

```
PS C:\Users\savio> ipconfig

Configuration IP de Windows

Carte réseau sans fil Connexion au réseau local* 1 :

    Statut du média. . . . . : Média déconnecté
    Suffixe DNS propre à la connexion. . . :

Carte réseau sans fil Connexion au réseau local* 2 :

    Statut du média. . . . . : Média déconnecté
    Suffixe DNS propre à la connexion. . . :

Carte Ethernet VMware Network Adapter VMnet8 :

    Suffixe DNS propre à la connexion. . . :
    Adresse IPv6 de liaison locale. . . . : fe80::20e5:22ad:ccdf:7620%12
    Adresse IPv4. . . . . : 192.168.203.1
    Masque de sous-réseau. . . . . : 255.255.255.0
    Passerelle par défaut. . . . . :

Carte réseau sans fil Wi-Fi :

    Suffixe DNS propre à la connexion. . . : laplateforme.io
    Adresse IPv6 de liaison locale. . . . : fe80::b61:a61f:f10d:3ddf%8
    Adresse IPv4. . . . . : 10.10.26.241
    Masque de sous-réseau. . . . . : 255.255.0.0
    Passerelle par défaut. . . . . : 10.10.0.1
```

4. Capture et analyse des paquets réseau

Pour être précis dans la recherche nous mettons en filtre l'adresse ip de passerelle La Plateforme "10.10.0.1".

Pour effectuer nos analyses nous utiliserons "ncat". Ncat est un utilitaire de mise en réseau bourré de fonctionnalités qui lit et écrit des données à travers les réseaux à partir de la ligne de commande. Ncat a été écrit pour le projet Nmap en tant que ré-exécution considérablement améliorée vénérable Netcat. Il utilise tant TCP que UDP pour la communication et sont conçus pour être fiables un outil dorsal pour fournir instantanément une connectivité réseau à



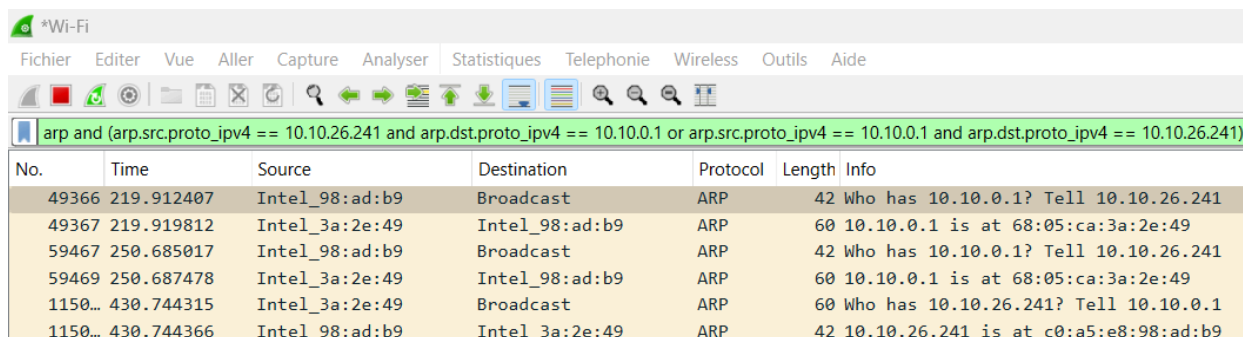
d'autres applications et utilisateurs. Ncat fonctionne non seulement avec IPv4 et IPv6, mais aussi fournit à l'utilisateur un nombre pratiquement illimité d'utilisations potentielles.

4.1 Paquets ARP :

Pour être précis dans notre recherche nous allons mettre un filtre pour n'afficher que notre adresse IP et l'adresse de La Plateforme pour les requêtes ARP :

arp and (arp.src.proto_ipv4 == 10.10.26.241 and arp.dst.proto_ipv4 == 10.10.0.1 or arp.src.proto_ipv4 == 10.10.0.1 and arp.dst.proto_ipv4 == 10.10.26.241)

Capture des paquets ARP :



The screenshot shows the Wireshark interface with the following details:

- Filter: `arp and (arp.src.proto_ipv4 == 10.10.26.241 and arp.dst.proto_ipv4 == 10.10.0.1 or arp.src.proto_ipv4 == 10.10.0.1 and arp.dst.proto_ipv4 == 10.10.26.241)`
- Table of captured packets:

No.	Time	Source	Destination	Protocol	Length	Info
49366	219.912407	Intel_98:ad:b9	Broadcast	ARP	42	Who has 10.10.0.1? Tell 10.10.26.241
49367	219.919812	Intel_3a:2e:49	Intel_98:ad:b9	ARP	60	10.10.0.1 is at 68:05:ca:3a:2e:49
59467	250.685017	Intel_98:ad:b9	Broadcast	ARP	42	Who has 10.10.0.1? Tell 10.10.26.241
59469	250.687478	Intel_3a:2e:49	Intel_98:ad:b9	ARP	60	10.10.0.1 is at 68:05:ca:3a:2e:49
1150...	430.744315	Intel_3a:2e:49	Broadcast	ARP	60	Who has 10.10.26.241? Tell 10.10.0.1
1150...	430.744366	Intel_98:ad:b9	Intel_3a:2e:49	ARP	42	10.10.26.241 is at c0:a5:e8:98:ad:b9

Analyse des paquets ARP :

```
> Frame 49366: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface \Device\NP
✓ Ethernet II, Src: Intel_98:ad:b9 (c0:a5:e8:98:ad:b9), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: Intel_98:ad:b9 (c0:a5:e8:98:ad:b9)
    Type: ARP (0x0806)
✓ Address Resolution Protocol (request)
  Hardware type: Ethernet (1)
  Protocol type: IPv4 (0x0800)
  Hardware size: 6
  Protocol size: 4
  Opcode: request (1)
  Sender MAC address: Intel_98:ad:b9 (c0:a5:e8:98:ad:b9)
  Sender IP address: 10.10.26.241
  Target MAC address: Xerox_00:00:00 (00:00:00:00:00:00)
  Target IP address: 10.10.0.1
```




Nous pouvons voir plusieurs éléments lors de notre analyse, comme le type du protocole, les adresses MAC des cartes réseau source et destination. Celles-ci communiquent et s'interrogent sur qui est qui ? Une fois les présentations faites et que tout est ok, alors le protocole.

4.2 Paquets UDP :

Sur notre machine hôte 10.10.24.40, on effectue la commande suivante pour ouvrir le port 1000 pour le protocole udp à l'adresse 10.10.0.1.

```
PS C:\Users\savio> ncat -u 10.10.0.1 1000
ceci est un test
```

No.	Time	Source	Destination	Protocol	Length	Info
34492	706.676092	10.10.24.40	10.10.0.1	UDP	59	54641 → 1000 Len=17
34493	706.678459	10.10.0.1	10.10.24.40	ICMP	87	Destination unreachable (Port unreachable)

On constate que le client (port 54641) établit une requête UDP vers le serveur 10.10.0.1 (port 1000).

Analyse du paquet UDP :

```
> Frame 34492: 59 bytes on wire (472 bits), 59 bytes captured (472 bits) on interface \Device\N
> Ethernet II, Src: Intel_98:ad:b9 (c0:a5:e8:98:ad:b9), Dst: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
> Internet Protocol Version 4, Src: 10.10.24.40, Dst: 10.10.0.1
v User Datagram Protocol, Src Port: 54641, Dst Port: 1000
  Source Port: 54641
  Destination Port: 1000
  Length: 25
  Checksum: 0x2c67 [unverified]
  [Checksum Status: Unverified]
  [Stream index: 3568]
  > [Timestamps]
  UDP payload (17 bytes)
v Data (17 bytes)
  Data: 636563692065737420756e20746573740a
  [Length: 17]
```

L'analyse de ce paquet nous montre plusieurs éléments concernant l'onglet UDP :

- Le port source qui a émit
- Se port de destination
- Sa longueur
- Son checksum
- Ainsi que ses données

On peut vérifier son contenu dans son chiffrement hexadécimal :

0000	68 05 ca 3a 2e 49 c0 a5 e8 98 ad b9 08 00 45 00	h...I... ..E.
0010	00 2d b8 05 00 00 80 11 00 00 0a 0a 18 28 0a 0a(..
0020	00 01 d5 71 03 e8 00 19 2c 67 63 65 63 69 20 65	...q.... ,gceci e
0030	73 74 20 75 6e 20 74 65 73 74 0a	st un te st.



4.3 Paquets TCP :

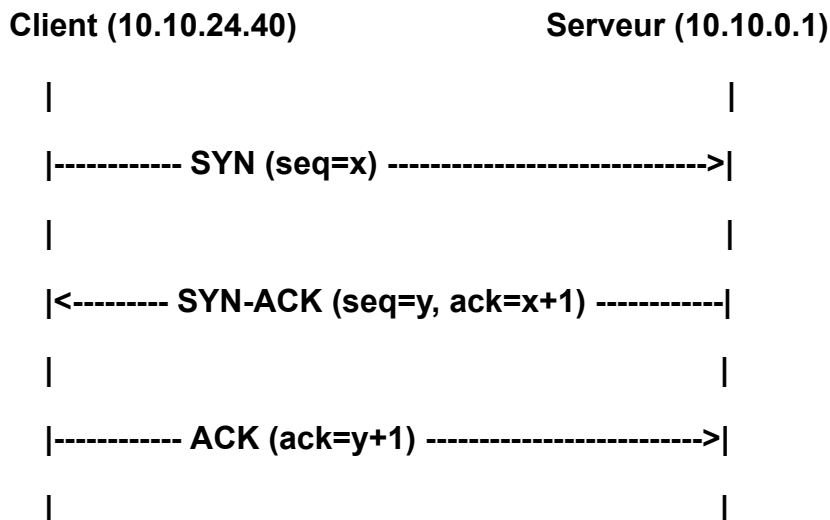
L'établissement d'une connexion TCP est basée sur un échange de trois messages ("three-way handshake"). En effet, chaque partie doit envoyer un message **SYN** qui propose un numéro de séquence pour la numération des données échangées, et qui doit être suivi par un message d'acquiescement **ACK** de l'autre côté.

No.	Time	Source	Destination	Protocol	Length	Info
742	10.227577	10.10.24.40	10.10.0.1	TCP	66	55286 → 3991 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
743	10.238363	10.10.0.1	10.10.24.40	TCP	66	3991 → 55286 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM WS=128
744	10.238464	10.10.24.40	10.10.0.1	TCP	54	55286 → 3991 [ACK] Seq=1 Ack=1 Win=131328 Len=0
745	10.239525	10.10.24.40	10.10.0.1	TLSv1.3	1168	Client Hello (SNI=alcasar.laplatforme.io)

Cette capture nous permet d'analyser les informations suivantes :

- Le protocole TCP, les ports utilisés par chacun ainsi que leur destination.
- Le temps du processus.
- Client : IP 10.10.24.40 initie la synchronisation avec le SYN.
- Serveur : IP 10.10.0.1 répond avec SYN-ACK pour vérifier la synchronisation.
- Client : IP 10.10.24.40 envoie ACK pour établir la validation de la connexion.

4.4 Mise en place d'un diagramme pour décrire le mécanisme de connexion





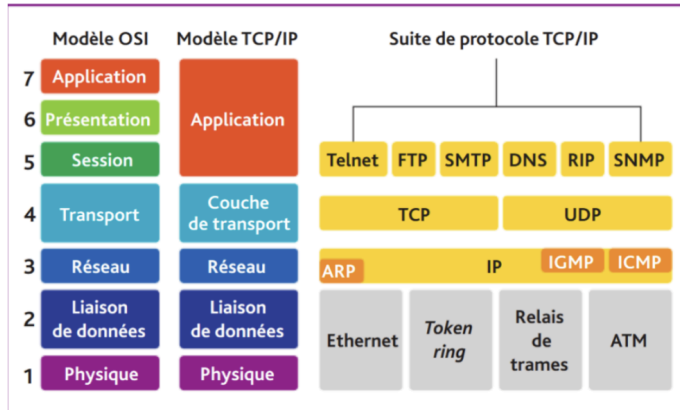
Ce diagramme décrit bien le Three-way handshake du protocole TCP. Chaque partie doit envoyer un message sync qui propose un numéro de séquence pour la numération des données échangées, et qui doit être suivi par un message d'acquittement ack de l'autre côté.

5. Désencapsuler les trames pour retrouver les différentes couches du modèle OSI

5.1 Désencapsulation des trames :

```
Frame 20: 606 bytes on wire (4848 bits), 606 bytes captured (4848 bits) on interface ens33, id 0
- Ethernet II, Src: VMware_a2:cd:b6 (00:0c:29:a2:cd:b6), Dst: VMware_ef:22:58 (00:50:56:ef:22:58)
  - Destination: VMware_ef:22:58 (00:50:56:ef:22:58)
  - Source: VMware_a2:cd:b6 (00:0c:29:a2:cd:b6)
  - Type: IPv4 (0x0800)
- Internet Protocol Version 4, Src: 192.168.76.130, Dst: 44.228.249.3
- Transmission Control Protocol, Src Port: 60548, Dst Port: 80, Seq: 1, Ack: 1, Len: 552
- Hypertext Transfer Protocol
- HTML Form URL Encoded: application/x-www-form-urlencoded
  - Form item: "uname" = "coucou"
  - Form item: "pass" = "mince"
```

5.2 Identification des couches OSI :



11 Modèle OSI et protocoles TCP/IP.

Quelles sont les adresses MAC sources, les IP sources et les adresses MAC sources, les IP destinations des données capturées ?

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe80::20c:29ff:fea2::ff02::12	ff02::12	ICMPv6	70	Router Solicitation from 00:0c:29:a2:cd:b6
2	0.936302791	192.168.197.4	192.168.76.2	DNS	80	Standard query 0xc1db A appstream.debian.org
3	0.936611587	192.168.197.4	192.168.76.2	DNS	80	Standard query 0xe0de AAAA appstream.debian.org
4	1.993147438	192.168.197.4	192.168.76.2	DNS	93	Standard query 0x1e16 A 3.debian.pool.ntp.org.localdomain
5	1.993307391	192.168.197.4	192.168.76.2	DNS	93	Standard query 0xd9d0 AAAA 3.debian.pool.ntp.org.localdomain
6	3.912836498	192.168.197.1	239.255.255.250	SSDP	174	M-SEARCH * HTTP/1.1
7	5.351649572	192.168.197.132	193.52.136.2	NTP	90	NTP Version 4, client
8	5.377308918	193.52.136.2	192.168.197.132	NTP	90	NTP Version 4, server
9	5.942556554	192.168.197.4	192.168.76.2	DNS	80	Standard query 0xc1db A appstream.debian.org
10	5.942815516	192.168.197.4	192.168.76.2	DNS	80	Standard query 0xe0de AAAA appstream.debian.org
11	7.000607771	192.168.197.4	192.168.76.2	DNS	81	Standard query 0xf43 A 0.debian.pool.ntp.org
12	7.001208928	192.168.197.4	192.168.76.2	DNS	81	Standard query 0xab5f AAAA 0.debian.pool.ntp.org
13	8.79467412	fe80::20c:29ff:fea2::ff02::12	ff02::12	ICMPv6	70	Router Solicitation from 00:0c:29:a2:cd:b6
14	9.471357676	192.168.197.132	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipp._tcp.local, "QM" question PTR _ipps._tcp.local, "QM" question
15	9.472585277	192.168.197.1	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipp._tcp.local, "QM" question PTR _ipps._tcp.local, "QM" question
16	9.746308496	fe80::20c:29ff:fea2::ff02::12	ff02::12	MDNS	107	Standard query 0x0000 PTR _ipps._tcp.local, "QM" question PTR _ipp._tcp.local, "QM" question
17	10.571476788	Vmware_0a:fc:33	Vmware_e0:9c:0b	ARP	60	who has 192.168.197.27 Tell 192.168.197.132
18	10.571477539	Vmware_e0:9c:0b	Vmware_0a:fc:33	ARP	60	192.168.197.2 is at 00:50:56:e0:9c:0b
19	10.948878663	192.168.197.4	192.168.76.2	DNS	92	Standard query 0x95c7 A appstream.debian.org.localdomain
20	10.949167400	192.168.197.4	192.168.76.2	DNS	92	Standard query 0xcbf4 AAAA appstream.debian.org.localdomain
21	11.230940169	fe80::20c:29ff:fea2::ff02::12	ff02::12	MDNS	101	Standard query 0x0000 PTR _nmea-0183._tcp.local, "QM" question
22	11.231769477	192.168.197.4	224.0.0.251	MDNS	81	Standard query 0x0000 PTR _nmea-0183._tcp.local, "QM" question
23	11.232662920	192.168.197.1	224.0.0.251	MDNS	81	Standard query 0x0000 PTR _nmea-0183._tcp.local, "QM" question
24	12.000872517	192.168.197.4	192.168.76.2	DNS	81	Standard query 0xf43 A 0.debian.pool.ntp.org
25	12.007827281	192.168.197.4	192.168.76.2	DNS	81	Standard query 0xab5f AAAA 0.debian.pool.ntp.org
26	15.952543600	192.168.197.4	192.168.76.2	DNS	92	Standard query 0x95c7 A appstream.debian.org.localdomain
27	15.952803523	192.168.197.4	192.168.76.2	DNS	92	Standard query 0xcbf4 AAAA appstream.debian.org.localdomain
28	17.013052104	192.168.197.4	192.168.76.2	DNS	93	Standard query 0x9971 A 0.debian.pool.ntp.org.localdomain

Frame 18: 87 bytes on wire (696 bits), 87 bytes captured (696 bits) on interface ens3, id 0

Ethernet II, Src: Vmware_c0:00:08 (00:50:56:c0:00:08), Dst: IPv4mcast_fb (01:00:5e:00:00:fb)

Destination: IPv4mcast_fb (01:00:5e:00:00:fb)

Source: Vmware_c0:00:08 (00:50:56:c0:00:08)

Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 192.168.197.1, Dst: 224.0.0.251

User Datagram Protocol, Src Port: 53949, Dst Port: 5353

Multicast Domain Name System (mDNS)

Les différentes lignes représentent:

- le numéro unique de séquence du paquets
- le moment ou le paquet a été capturé
- la source (adresse ip ou MAC de l'émetteur du paquet)
- la destination (adresse ip ou MAC du récepteur du paquet)
- le protocole utilisée (permet de voir la nature du trafic réseau)
- affichage de la longueur du paquet



Partie 2

1. Analyse des protocoles spécifiques sur un réseau local

Nous allons désormais faire nos tests d'écoute sur une VM serveur et une VM client en NAT.

Configuration du réseau en NAT sur VMware :

The image shows two overlapping windows from the VMware interface. The top window is titled 'Virtual Network Editor' and contains a table with the following data:

Name	Type	External Connection	Host Connection	DHCP	Subnet Address
VMnet8	NAT	NAT	Connected	Enabled	192.168.203.0

The bottom window is titled 'NAT Settings' and displays the following configuration:

Network: vmnet8
Subnet IP: 192.168.203.0
Subnet mask: 255.255.255.0
Gateway IP: 192 . 168 . 203 . 2

Nous installerons nos deux vm sous debian 12.

2 . Installation et configuration de la VM Serveur

Procéder à l'installation classique d'une vm debian 12 :

The image shows a terminal window from the Debian installer. The title bar reads '[[!]] Créer les utilisateurs et choisir les mots de passe'. The text in the terminal is as follows:

Un compte d'utilisateur va être créé afin que vous puissiez disposer d'un compte différent de celui du superutilisateur (« root »), pour l'utilisation courante du système.

Veuillez indiquer le nom complet du nouvel utilisateur. Cette information servira par exemple dans l'adresse d'origine des courriels émis ainsi que dans tout programme qui affiche ou se sert du nom complet. Votre propre nom est un bon choix.

Nom complet du nouvel utilisateur :

Server

At the bottom, there are two buttons: '<Revenir en arrière' and '<Continuer>'.



[[!]] Créer les utilisateurs et choisir les mots de passe

Veuillez choisir un identifiant (« login ») pour le nouveau compte. Votre prénom est un choix possible. Les identifiants doivent commencer par une lettre minuscule, suivie d'un nombre quelconque de chiffres et de lettres minuscules.

Identifiant pour le compte utilisateur :

server

<Revenir en arrière> <Continuer>

Pour ce serveur nous utiliserons les identifiants et mots de passe suivant suivants :

Identifiant : server

Mot de passe : server

Sur la VM Serveur on installe et configure les services réseau DHCP, DNS, FTP.

2.1 Serveur DHCP

Vérifions l'adress IP de notre machine client avant l'installation et le paramétrage du server :

```
client@debian:~$ ip route
default via 192.168.1.1 dev ens33 proto dhcp src 192.168.1.1
6 metric 100
192.168.1.0/24 dev ens33 proto kernel scope link src 192.168
.1.16 metric 100
```

Installation du serveur DHCP sur la machine server



On se rend dans le fichier “ /etc/dhcp/dhcpd.conf “

```
GNU nano 7.2          dhcpd.conf
# Définir le temps de bail par défaut et maximum
default-lease-time 600;
max-lease-time 7200;

# Déclaration du sous-réseau
subnet 192.168.76.0 netmask 255.255.255.0 {
    range 192.168.76.100 192.168.76.150;    # Plage d'adresses IP à attribuer
    option routers 192.168.76.2;           # Adresse IP de la passerelle
    option subnet-mask 255.255.255.0;      # Masque de sous-réseau
    option broadcast-address 192.168.76.255; # Adresse de diffusion
    option domain-name-servers 8.8.8.8, 8.8.4.4; # Serveurs DNS
    option domain-name "local";           # Nom de domaine
}
```

Puis “/etc/default/isc-dhcp-server”

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

# 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

# Additional options to start dhcpd with.
# Don't use options -cf or -pf here; use DHCPD_CONF/ DHCPD_PID instead
#OPTIONS=""

# On what interfaces should the DHCP server (dhcpd) serve DHCP requests?
# Separate multiple interfaces with spaces, e.g. "eth0 eth1".
INTERFACESv4="ens33"
INTERFACESv6=""
```

/etc/network/interfaces



```
GNU nano 7.2 interfaces
auto ens33
iface ens33 inet static
    address 192.168.76.4
    netmask 255.255.255.0
    gateway 192.168.76.2
    dns-nameservers 8.8.8.8 8.8.4.4
```

VM Client : Configurez la VM client pour obtenir une adresse IP via DHCP.

/etc/network/interfaces

```
GNU nano 7.2 interfaces
auto ens33
iface ens33 inet dhcp
```

Adresse IP de la machine client après activation du serveur DHCP.

```
jordan@debian:/etc/network$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:0a:fc:33 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.76.101/24 brd 192.168.76.255 scope global dynamic ens33
        valid_lft 537sec preferred_lft 537sec
    inet6 fe80::20c:29ff:fe0a:fc33/64 scope link
        valid_lft forever preferred_lft forever
```

Capture des paquets DHCP Lancez Wireshark sur la VM client pour capturer les paquets DHCP lors de l'initialisation de la connexion réseau.



*ens33									
Fichier Editer Vue Aller Capture Analyser Statistiques Telephone Wireless Outils Aide									
dhcpc									
No.	Time	Source	Destination	Protocol	Length	Info			
63	183.652146827	192.168.76.100	192.168.76.4	DHCP	342	DHCP Request - Transaction ID 0x1e023750			
64	183.654988340	192.168.76.4	192.168.76.100	DHCP	342	DHCP ACK - Transaction ID 0x1e023750			
133	317.915713972	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request - Transaction ID 0xa40ebd53			
134	317.919466398	192.168.76.4	192.168.76.101	DHCP	342	DHCP ACK - Transaction ID 0xa40ebd53			
797	493.454942067	192.168.76.100	192.168.76.4	DHCP	342	DHCP Request - Transaction ID 0xae8e9b5			
798	493.459719819	192.168.76.4	192.168.76.100	DHCP	342	DHCP ACK - Transaction ID 0xae8e9b5			
859	558.014091138	192.168.76.101	192.168.76.4	DHCP	342	DHCP Request - Transaction ID 0xa40ebd53			
860	558.016088701	192.168.76.4	192.168.76.101	DHCP	342	DHCP ACK - Transaction ID 0xa40ebd53			

Frame 859: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface ens33, id 0

Ethernet II, Src: VMware_0a:fc:33 (00:0c:29:0a:fc:33), Dst: VMware_a2:cd:b6 (00:0c:29:a2:cd:b6)

Destination: VMware_a2:cd:b6 (00:0c:29:a2:cd:b6)

Source: VMware_0a:fc:33 (00:0c:29:0a:fc:33)

Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 192.168.76.101, Dst: 192.168.76.4

0100 = Version: 4

.... 0101 = Header Length: 20 bytes (5)

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

0000 00.. = Differentiated Services Codepoint: Default (0)

.... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)

Total Length: 328

Identification: 0x4022 (16418)

010. = Flags: 0x2, Don't fragment

0. = Reserved bit: Not set

..1. = Don't fragment: Set

0000 00 0c 29 a2 cd b6 00 0c 29 0a fc 33 08 00 45 00 ..).....):3-E-

0010 01 48 40 22 40 00 40 11 df c8 c9 a8 4c 05 c0 a8 -H0000:....Le-

0020 4c 04 00 44 00 43 01 34 ed 00 01 01 00 00 a4 0e L-D-C-4.....

0030 bd 53 00 00 00 00 c0 a8 4c 05 00 00 00 00 00 ..S.....Le.....

0040 00 00 00 00 00 00 00 0c 29 0a fc 33 00 00 00 00):3.....

0050 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

0060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

0070 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

0080 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

0090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00a0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00b0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00c0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00d0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00e0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

00f0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

0100 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00Le.....

3. Installation et configuration de la VM Client

Sur la VM Client on va interagir avec les services installés sur la vm serveur en faisant des requêtes DNS, télécharger des fichiers via FTP.

3.1 Requêtes DNS

Analyse des requêtes DNS avec Wireshark

No.	dns	Source	Destination	Protocol	Length	Info
	dnsserver	192.168.197.132	8.8.8.8	DNS	85	Standard query 0xf7fb SRV _http._tcp.deb.debian.org
12	41.437077792	192.168.197.132	8.8.4.4	DNS	85	Standard query 0xf7fb SRV _http._tcp.deb.debian.org
15	41.458522066	8.8.4.4	192.168.197.132	DNS	129	Standard query response 0xf7fb SRV _http._tcp.deb.debian.org SRV 10 1 80 debian.map.fastlydns.net
16	41.459618740	192.168.197.132	8.8.8.8	DNS	84	Standard query 0x2a95 A debian.map.fastlydns.net
17	41.459618931	192.168.197.132	8.8.8.8	DNS	84	Standard query 0x3893 AAAA debian.map.fastlydns.net
18	41.467715144	8.8.8.8	192.168.197.132	DNS	100	Standard query response 0x2a95 A debian.map.fastlydns.net A 199.232.82.132
19	41.469984216	8.8.8.8	192.168.197.132	DNS	112	Standard query response 0x3893 AAAA debian.map.fastlydns.net AAAA 2a04:4e42:54::644
38	48.896248253	192.168.197.132	8.8.8.8	DNS	80	Standard query 0xf644 A appstream.debian.org
39	48.896417573	192.168.197.132	8.8.8.8	DNS	80	Standard query 0x0246 AAAA appstream.debian.org
40	48.929834931	8.8.8.8	192.168.197.132	DNS	185	Standard query response 0x0246 AAAA appstream.debian.org CNAME static.debian.org AAAA 2001:67c:2564:a119::7...



*ens33						
Fichier Editor Vue Aller Capture Analyser Statistiques Telephone Wireless Outils Aide						
mdns						
No.	Time	Source	Destination	Protocol	Length	Info
15	8.569397503	192.168.197.1	224.0.0.251	MDNS	81	Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
16	8.560227746	fe80::a0be:d7b6:5bb...	ff02::fb	MDNS	101	Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
17	8.560779673	fe80::a0be:d7b6:5bb...	ff02::fb	MDNS	139	Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
18	8.561152075	192.168.197.1	224.0.0.251	MDNS	119	Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
19	8.561570232	192.168.197.1	224.0.0.251	MDNS	81	Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
20	8.561919487	fe80::a0be:d7b6:5bb...	ff02::fb	MDNS	101	Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
21	8.562370993	fe80::a0be:d7b6:5bb...	ff02::fb	MDNS	139	Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
22	8.562809696	192.168.197.1	224.0.0.251	MDNS	119	Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
25	8.698453855	192.168.197.4	224.0.0.251	MDNS	87	Standard query 0x0000 PTR _ipp._tcp.local, "QM" question PTR _ipps._tcp.local, "QM" question
26	8.706850500	192.168.197.2	224.0.0.251	MDNS	792	Standard query response 0x0000 PTR _ipp._tcp.local, "QU" question PTR Brother MFC-L9570CDW series...
28	8.808602600	192.168.197.2	224.0.0.251	MDNS	1514	Standard query response 0x0000 PTR _ipp._tcp.local, "QM" question PTR _ipps._tcp.local, "QM" question
Frame 15: 81 bytes on wire (648 bits), 81 bytes captured (648 bits) on interface eth0, 0 bytes on interface eth0						
Ethernet II, Src: VMware_c0:00:08 (00:50:56:00:00:08), Dst: 01:00:5e:00:00:01 (01:00:5e:00:00:01)						
Internet Protocol Version 4, Src: 192.168.197.1, Dst: 224.0.0.251						
User Datagram Protocol, Src Port: 5353, Dst Port: 5353						
Multicast Domain Name System (query)						

Téléchargement de fichiers via FTP :

apt install -y proftpd

Configuration de PROFTPD

nano /etc/proftpd/conf.d/ftp-perso.conf

```
GNU nano 7.2 proftpd.conf
# Disable MultilineRFC2228 per https://github.com/proftpd/proftpd/issues/1085
# MultilineRFC2228on
defaultServer on
showSymlinks on

TimeoutNoTransfer 600
TimeoutStalled 600
TimeoutIdle 1200

displayLogin welcome.msg
displayChdir .message true
listOptions "-l"

denyFilter \*.*/

# Use this to jail all users in their homes
# DefaultRoot ~

# Users require a valid shell listed in /etc/shells to login.
# Use this directive to release that constrain.
```



```
PS C:\Users\archi> ftp 192.168.197.4
Connecté à 192.168.197.4.
220 ProFTPD Server (IT-CONNECT) [192.168.197.4]
200 UTF-8 activÃ©
Utilisateur (192.168.197.4:(none)) : itconnect
331 Mot de passe requis pour itconnect
Mot de passe :

230 Utilisateur itconnect authentifié
ftp> |
```

3.2 Requêtes FTP

Analyse des téléchargement via FTP avec Wireshark .

ftp

No.	Time	Source	Destination	Protocol	Length	Info
17	7.240329787	192.168.197.4	192.168.197.1	FTP	103	Response: 220 ProFTPD Server (IT-CONNECT) [192.168.197.4]
18	7.243791146	192.168.197.1	192.168.197.4	FTP	68	Request: OPTS UTF8 ON
20	7.244161605	192.168.197.4	192.168.197.1	FTP	73	Response: 200 UTF-8 activé
30	12.392117831	192.168.197.1	192.168.197.4	FTP	70	Request: USER itconnect
34	12.646444394	192.168.197.4	192.168.197.1	FTP	94	Response: 331 Mot de passe requis pour itconnect
36	14.322971119	192.168.197.1	192.168.197.4	FTP	65	Request: PASS root
38	14.395386503	192.168.197.4	192.168.197.1	FTP	94	Response: 230 Utilisateur itconnect authentifié

Partie 3

La commande tshark -i ens33 nous permet d'enregistrer une capture ;

tshark: "Outils de capture et d'analyse de paquets réseau"

-i ens33: "Spécifie l'interface réseau à utiliser pour la capture"



```
jordan@debian:~/Documents$ sudo tshark -i ens33
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:3420) 11:03:08.688593 [Main MESSAGE] -- Capture started.
** (tshark:3420) 11:03:08.689325 [Main MESSAGE] -- File: "/tmp/wireshark_ens33CNX702.pcapng"
 1 0.000000000 192.168.197.1 → 239.255.255.250 SSDP 217 M-SEARCH * HTTP/1.1
 2 1.012337126 192.168.197.1 → 239.255.255.250 SSDP 217 M-SEARCH * HTTP/1.1
 3 2.014525122 192.168.197.1 → 239.255.255.250 SSDP 217 M-SEARCH * HTTP/1.1
 4 3.017260698 192.168.197.1 → 239.255.255.250 SSDP 217 M-SEARCH * HTTP/1.1
 5 17.674154948 192.168.197.1 → 224.0.0.251 MDNS 85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QU" question
 6 17.676327387 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 105 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QU" question
 7 18.675438429 192.168.197.1 → 224.0.0.251 MDNS 85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" question
 8 18.676616897 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 105 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" question
 9 75.765694927 fe80::a0be:d7b6:5bb7:209a → ff02::16 ICMPv6 90 Multicast Listener Report Message v2
10 75.765899752 192.168.197.1 → 224.0.0.22 IGMPv3 60 Membership Report / Leave group 224.0.0.252
11 75.799998857 fe80::a0be:d7b6:5bb7:209a → ff02::16 ICMPv6 90 Multicast Listener Report Message v2
12 75.800332155 192.168.197.1 → 224.0.0.22 IGMPv3 60 Membership Report / Join group 224.0.0.252 for any sources
13 75.809641691 fe80::a0be:d7b6:5bb7:209a → ff02::16 ICMPv6 90 Multicast Listener Report Message v2
14 75.810173822 192.168.197.1 → 224.0.0.22 IGMPv3 60 Membership Report / Leave group 224.0.0.252
15 75.813354089 fe80::a0be:d7b6:5bb7:209a → ff02::16 ICMPv6 90 Multicast Listener Report Message v2
16 75.813968146 192.168.197.1 → 224.0.0.22 IGMPv3 60 Membership Report / Join group 224.0.0.252 for any sources
17 75.816833891 192.168.197.1 → 224.0.0.251 MDNS 81 Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
18 75.818450699 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 101 Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
19 75.820165342 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 139 Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
20 75.821825112 192.168.197.1 → 224.0.0.251 MDNS 81 Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
21 75.823019276 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 101 Standard query 0x0000 ANY LAPTOP-H9B096GI.local, "QM" question
22 75.824189018 fe80::a0be:d7b6:5bb7:209a → ff02::fb MDNS 139 Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
23 75.826079267 192.168.197.1 → 224.0.0.251 MDNS 119 Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
24 75.826466556 192.168.197.1 → 224.0.0.251 MDNS 119 Standard query response 0x0000 AAAA fe80::a0be:d7b6:5bb7:209a A 192.168.197.1
25 75.855556300 192.168.197.1 → 224.0.0.22 IGMPv3 60 Membership Report / Join group 224.0.0.252 for any sources
26 75.855738122 fe80::a0be:d7b6:5bb7:209a → ff02::16 ICMPv6 90 Multicast Listener Report Message v2
```

```
jordan@debian: ~
ned: Permission denied.

tshark:
root@debian:/home/jordan/Documents# tshark -i ens33 -f "tcp port 80" -w http_capture.pcap
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
tshark: The file to which the capture would be saved ("http_capture.pcap") could not be opened: Permission denied.

tshark:
root@debian:/home/jordan/Documents# sudo chmod o+w /home/jordan/Documents
root@debian:/home/jordan/Documents# tshark -i ens33 -f "tcp port 80" -w http_capture.pcap
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:3461) 11:15:25.520189 [Main MESSAGE] -- Capture started.
** (tshark:3461) 11:15:25.521529 [Main MESSAGE] -- File: "http_capture.pcap"
1766 ^C
tshark:
root@debian:/home/jordan/Documents# ls
http_capture.pcap
root@debian:/home/jordan/Documents# sudo wireshark
** (wireshark:3976) 11:17:03.888969 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

http_capture.pcap

Fichier Éditer Vue Aller Capture Analyser Statistiques Téléphonie Wireless Outils

Appliquer un filtre d'affichage ... <Ctrl-/>

No.	Time	Source	Destination	Protocol
331	40.020327241	192.168.76.101	138.199.14.21	TCP
332	40.024642331	192.168.76.101	138.199.14.21	HTTP
333	40.024972825	192.168.76.101	138.199.14.21	HTTP
334	40.024983961	138.199.14.21	192.168.76.101	TCP
335	40.025287945	192.168.76.101	138.199.14.21	HTTP
336	40.025317901	138.199.14.21	192.168.76.101	TCP
337	40.025591275	192.168.76.101	138.199.14.21	HTTP
338	40.025626565	138.199.14.21	192.168.76.101	TCP
339	40.026232368	138.199.14.21	192.168.76.101	TCP
340	40.031630247	192.168.76.101	138.199.14.21	HTTP
341	40.032032562	138.199.14.21	192.168.76.101	TCP
342	40.036847743	138.199.14.21	192.168.76.101	HTTP
343	40.036881910	192.168.76.101	138.199.14.21	TCP
344	40.038589842	192.168.76.101	138.199.14.21	TCP
345	40.041258763	138.199.14.55	192.168.76.101	TCP
346	40.041406686	138.199.14.21	192.168.76.101	HTTP

Frame 1: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0

Ethernet II, Src: VMware_0a:fc:33 (00:0c:29:0a:fc:33), Dst: 08:00:27:00:00:00

Internet Protocol Version 4, Src: 192.168.76.101, Destination: 138.199.14.21

Transmission Control Protocol, Src Port: 56500, Dst Port: 80



```
FOOTMERCATO

jordan@debian: ~
root@debian:/home/jordan/Documents# sudo wireshark
** (wireshark:3976) 11:17:03.888969 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
root@debian:/home/jordan/Documents# wireshark
** (wireshark:4135) 11:19:37.610200 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
** (wireshark:4135) 11:19:46.169149 [Capture MESSAGE] -- Capture Start ...
** (wireshark:4135) 11:19:46.267382 [Capture MESSAGE] -- Capture started
** (wireshark:4135) 11:19:46.267592 [Capture MESSAGE] -- File: "/tmp/wireshark_ens33IS4W02.pcapng"
** (wireshark:4135) 11:22:06.391272 [Capture MESSAGE] -- Capture Stop ...
** (wireshark:4135) 11:22:06.526685 [Capture MESSAGE] -- Capture stopped.
root@debian:/home/jordan/Documents# sudo tshark -i ens33 -f "udp port 53" -w dns_capture.pcap
Running as user "root" and group "root". This could be dangerous.
Capturing on 'ens33'
** (tshark:4547) 11:22:53.043142 [Main MESSAGE] -- Capture started.
** (tshark:4547) 11:22:53.045705 [Main MESSAGE] -- File: "dns_capture.pcap"
218 ^C
tshark:
root@debian:/home/jordan/Documents# wireshark
** (wireshark:4709) 11:23:34.419603 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

dns_capture.pcap

No.	Time	Source	Destination	Protocol
1	0.000000000	192.168.76.101	8.8.8.8	DNS
2	0.000256668	192.168.76.101	8.8.8.8	DNS
3	0.070881776	8.8.8.8	192.168.76.101	DNS
4	0.070882458	8.8.8.8	192.168.76.101	DNS
5	0.189433150	192.168.76.101	8.8.8.8	DNS
6	0.189826229	192.168.76.101	8.8.8.8	DNS
7	0.197865728	8.8.8.8	192.168.76.101	DNS
8	0.197866371	8.8.8.8	192.168.76.101	DNS
9	0.390777112	192.168.76.101	8.8.8.8	DNS
10	0.390990247	192.168.76.101	8.8.8.8	DNS
11	0.400267793	192.168.76.101	8.8.8.8	DNS
12	0.402922163	8.8.8.8	192.168.76.101	DNS
13	0.405743301	8.8.8.8	192.168.76.101	DNS
14	0.405743778	8.8.8.8	192.168.76.101	DNS
15	0.416197573	192.168.76.101	8.8.8.8	DNS
16	0.416581878	192.168.76.101	8.8.8.8	DNS

Frame 1: 72 bytes on wire (576 bits), 72 bytes captured on interface ens33, 72 bytes from 192.168.76.101 to 8.8.8.8 on interface ens33
Ethernet II, Src: VMware_0a:fc:33 (00:0c:29:0a:fc:33), Dst: 08:00:00:00:00:00
Internet Protocol Version 4, Src: 192.168.76.101, Destination: 8.8.8.8
User Datagram Protocol, Src Port: 49706, Dst Port: 53
Domain Name System (query)

Exemple de commande :

```
sudo tshark -i ens33 -y "dns" -w dns_capture.pcap
```

tshark : Outil de capture et d'analyse paquet reseau

-i ens33 : spécifie l'interface réseau que l'on veut capturer

-y dns : Filtre d'affichage qui capture uniquement les paquets dns

-w dns_capture.pcap : Enregistre



The terminal window shows the following commands and output:

```
jordan@debian: ~  
218 ^C  
tshark:  
root@debian:/home/jordan/Documents# wireshark  
** (wireshark:4709) 11:23:34.419603 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'  
root@debian:/home/jordan/Documents# sudo tshark -i ens33 -Y "dns" -w dns_capture2e.pcap  
Running as user "root" and group "root". This could be dangerous.  
tshark: Display filters aren't supported when capturing and saving the captured packets.  
root@debian:/home/jordan/Documents# ls  
dns_capture.pcap  http_capture.pcap  
root@debian:/home/jordan/Documents# sudo tshark -i ens33 -w full_capture.pcap  
Running as user "root" and group "root". This could be dangerous.  
Capturing on 'ens33'  
** (tshark:5114) 11:31:32.384023 [Main MESSAGE] -- Capture started.  
** (tshark:5114) 11:31:32.386300 [Main MESSAGE] -- File: "full_capture.pcap"  
1193 ^C  
tshark:  
root@debian:/home/jordan/Documents# tshark -r full_capture.pcap -Y "dns" -w dns_capture_filtered.pcap  
Running as user "root" and group "root". This could be dangerous.  
root@debian:/home/jordan/Documents# sudo wireshark  
** (wireshark:5231) 11:32:32.521967 [GUI WARNING] -- QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
```

The Wireshark interface shows the following table of captured packets:

No.	Time	Source	Destination	Protocol	Length	Info
109	10.724047289	192.168.76.101	8.8.8.8	DNS	78	Standard query 0xc3bc3 A st
110	10.724595182	192.168.76.101	8.8.8.8	DNS	74	Standard query 0x90b4 A eu
111	10.724739796	192.168.76.101	8.8.8.8	DNS	74	Standard query 0x4ca9 AAA
112	10.725967851	8.8.8.8	192.168.76.101	DNS	115	Standard query response 0x
113	10.726601002	192.168.76.101	8.8.8.8	DNS	80	Standard query 0x695a A st
114	10.726761323	192.168.76.101	8.8.8.8	DNS	80	Standard query 0x1657 AAA
115	10.735471852	8.8.8.8	192.168.76.101	DNS	110	Standard query response 0x
116	10.735472208	8.8.8.8	192.168.76.101	DNS	134	Standard query response 0x
117	10.738067808	8.8.8.8	192.168.76.101	DNS	129	Standard query response 0x
118	10.738068285	8.8.8.8	192.168.76.101	DNS	132	Standard query response 0x
119	10.738554322	192.168.76.101	8.8.8.8	DNS	76	Standard query 0x1527 A ww
120	10.738926193	192.168.76.101	8.8.8.8	DNS	76	Standard query 0xa322 AAA
121	10.769810099	8.8.8.8	192.168.76.101	DNS	96	Standard query response 0x
122	10.769810550	8.8.8.8	192.168.76.101	DNS	131	Standard query response 0x
123	10.778224101	8.8.8.8	192.168.76.101	DNS	92	Standard query response 0x
124	10.778224563	8.8.8.8	192.168.76.101	DNS	106	Standard query response 0x

Voici un exemple de script filtrant les paquets DNS.

```
#!/bin/bash  
  
# Nom de l'interface réseau  
INTERFACE="ens33"  
  
# Durée de la capture en secondes  
DURATION=10  
  
# Fichiers de capture  
FULL_CAPTURE="full_capture1.pcap"  
DNS_CAPTURE="dns_capture_filtered1.pcap"  
  
# Capturer tout le trafic pendant la durée spécifiée  
echo "Capturing traffic on $INTERFACE for $DURATION seconds..."  
sudo tshark -i $INTERFACE -a duration:$DURATION -w $FULL_CAPTURE  
  
# Filtrer les paquets DNS du fichier de capture complet  
echo "Filtering DNS packets..."  
tshark -r $FULL_CAPTURE -Y "dns" -w $DNS_CAPTURE  
  
echo "Capture complete. DNS packets saved to $DNS_CAPTURE."
```