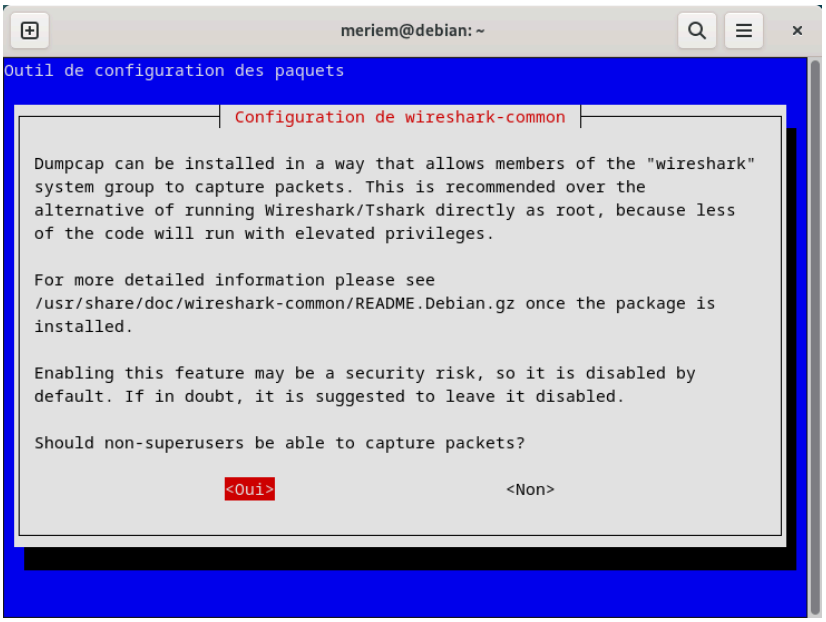
Partie 1

Définition wireshark :

Wireshark est un logiciel de capture et d'analyse de paquets réseau. Il permet d'observer en temps réel les données circulant sur un réseau, utile pour diagnostiquer des problèmes ou analyser la sécurité.

Installation de wireshark sur linux :

```
meriem@debian:~$ apt install wireshark
```



Ajout de l'utilisateur dans le groupe WireShark :

```
root@debian:~# usermod -a -G wireshark meriem
```

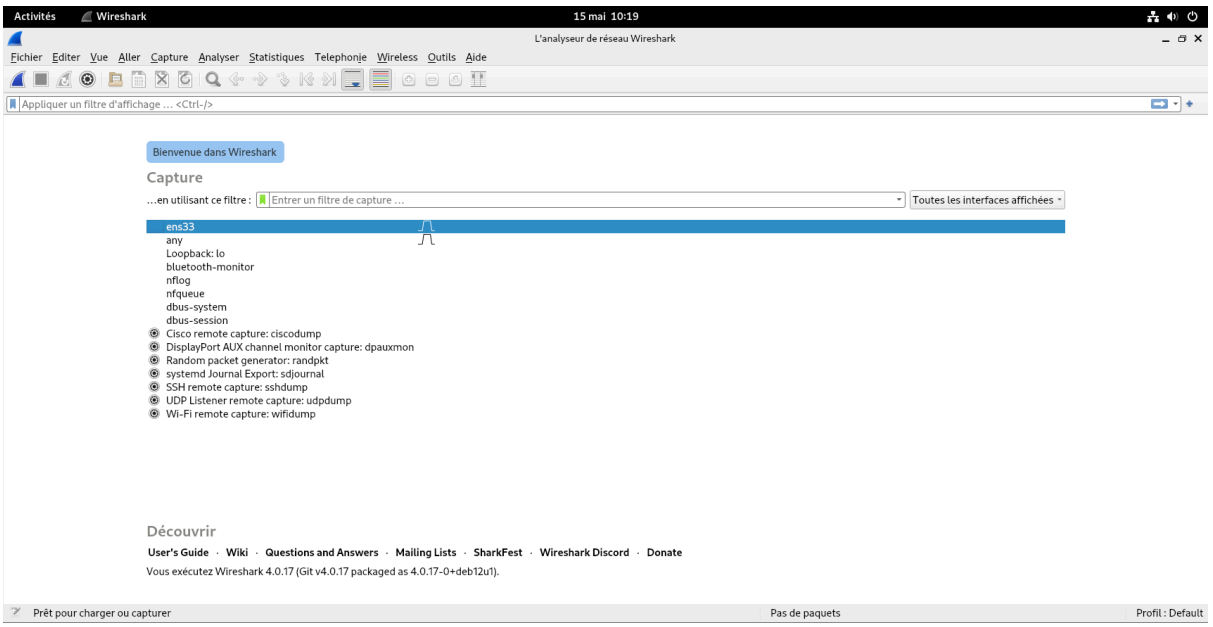
Il faut ensuite redémarrer la machine.

```
root@debian:~# reboot
```

Ouvrire wireshark avec la commande suivante :

```
meriem@debian:~$ wireshark
```

Interface de wireshark :



Quelle est la différence entre une trame et un paquet ? Qu'est-ce que le format pcap/pcapng ?

*** Trame : C'est l'unité de données échangée au niveau de la couche liaison de données (couche 2 du modèle OSI). Une trame contient généralement les informations nécessaires à la communication entre deux appareils directement connectés (comme les adresses MAC, des informations de contrôle, etc.). Elle est utilisée dans les réseaux locaux**

(Ethernet, par exemple).

*** Paquet : C'est l'unité de données échangée au niveau de la couche réseau (couche 3 du modèle OSI). Un paquet contient les données envoyées entre des hôtes sur des réseaux différents et inclut des informations de routage, comme l'adresse IP source et destination.**

Pour résumer : la trame est utilisée pour la transmission locale entre deux dispositifs proches, tandis que le paquet permet de transporter les données entre des réseaux distants.

Trame	Paquet
Unité de donées de la couche liaison (couche 2 du modèle OSI)	Unitaire de donneres de la couche droite (couche 3 du modèle OSI)
Contient les adresses MAC (source et destination)	Contient les adresses IP (source et destination)
Encapsule généralement un paquet	Peut encapsuler un segment (par exemple, TCP/UDP)
Sert au transport des données sur un jourme roi local	Sert à l'ament des données entre lieux différents
Exemple : trame Ethernet, trame Wi-Fi	Exemple: paquet IP

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

PCAP (Capture de paquets) et PCAPNG (PCAP Next Generation) sont des formats de fichiers utilisés pour stocker des captures de trafic.

*** PCAP : Format historique utilisé pour enregistrer des paquets capturés sur un rayon. Chaque entrée contient un horodatage, la longueur et les données du paquet. Il est largement utilisé par des outils comme Wireshark et tcpdump .**

***PCAPNG : Format plus récent et évolué, conçu pour surmonter les limitations du format PCAP. Il permet notamment de :**

*** De stocker des paquets issus de différents types de liaisons (Ethernet, Wi-Fi, etc.) dans un jour fichier.**

*** D'outre des comptes rendus (commentaires, informations sur l'interface de capture, etc.).**

***De fusionner facilement plusieurs fichiers de capture.**

***De gérer des annotations et des informations supplémentaires sur casque paquet.**

PCAPNG est devenu le format par défaut dans Wireshark depuis 2012, car il offre plus de flexibilité et d'informations pour l'analyse réseau.
Résumé :

***La traite appartient à la couche liaison (adresse MAC), le paquet à la couche droite (adresse IP).**

***Les formats PCAP/PCAPNG servent à stocker des captures de trafic réseau, PCAPNG et plus flexible du format PCAP.**

Définition des paquets TCP / ARP / UDP :

Paquet TCP (Transmission Control Protocol) : Un paquet de données utilisé pour la transmission fiable entre ordinateurs sur un réseau. Il garantit la réception correcte des données.

Paquet ARP (Address Resolution Protocol) : Un paquet qui sert à associer une adresse IP à une adresse MAC sur un réseau local, permettant ainsi à un dispositif de localiser l'adresse physique d'un autre appareil en fonction de son adresse IP.

Paquet UDP (User Datagram Protocol) : Un paquet qui permet de transmettre des données de manière rapide, sans garantir leur réception, leur ordre ou l'absence d'erreurs.

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

Pour obtenir les adresses MAC sources, les IP sources, les adresses MAC destinations, et les IP destinations des données capturées, nous

devrons analyser les paquets capturés à l'aide d'un outil comme Wireshark. Ces informations sont présentes dans les différentes couches des paquets :

1. Adresses MAC sources et destinations (couche liaison de données - Ethernet)

Les adresses MAC source et MAC destination se trouvent dans l'en-tête Ethernet de chaque trame.

***MAC source : C'est l'adresse physique de l'émetteur de la trame exemple pour paquet ARP :**

```
Sender MAC address: VMware_6c:40:b2 (00:0c:29:6c:40:b2)
```

***MAC destination : C'est l'adresse physique du récepteur de la trame exemple pour le paquet ARP:**

```
Target MAC address: VMware_e9:9c:a8 (00:50:56:e9:9c:a8)
```

Ces informations sont visibles dans le frame détails sous la section Ethernet II dans Wireshark.

2. IP sources et destinations (couche réseau - IP)

Les adresses IP source et IP destination se trouvent dans l'en-tête IP de chaque paquet (au niveau de la couche 3).

***IP source : C'est l'adresse IP de l'émetteur du paquet exemple pour le paquet TCP :**

192.168.153.129

***IP destination : C'est l'adresse IP du destinataire du paquet exemple pour le paquet TCP :**

142.251.168.84

Ces informations sont visibles dans le frame détails sous la section Internet Protocol dans Wireshark.

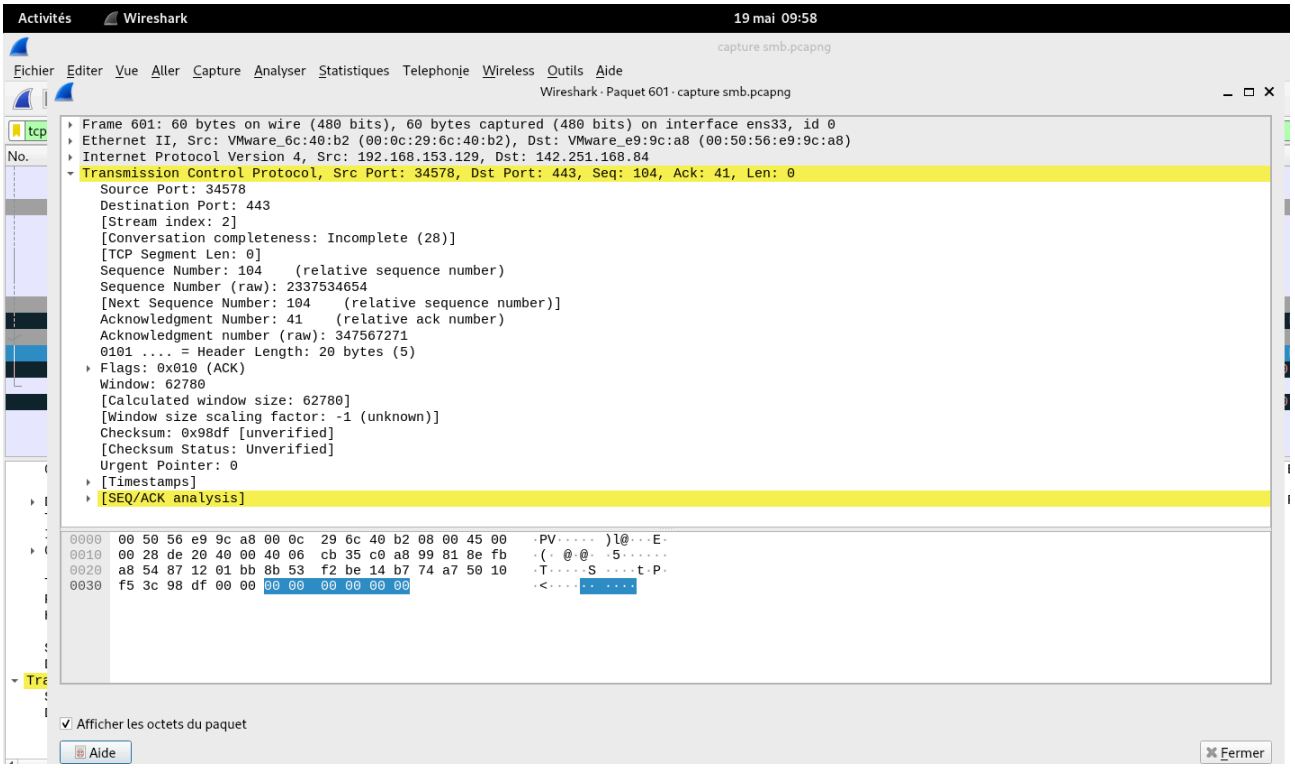
Comment obtenir ces informations dans Wireshark ?

1. **Ouvre Wireshark et commence la capture de paquets.**

2. **Une fois la capture en cours, nous sélectionnons un paquet pour voir les détails.**

3. **Dans le panneau du bas (détails du paquet), regarde les différentes couches du paquet :**

    * **Pour les adresses MAC, sous la section Ethernet II**
    * **Pour les adresses IP, sous la section Internet Protocol.**



<u>Explications des champs principaux dans le paquet capturé</u> :

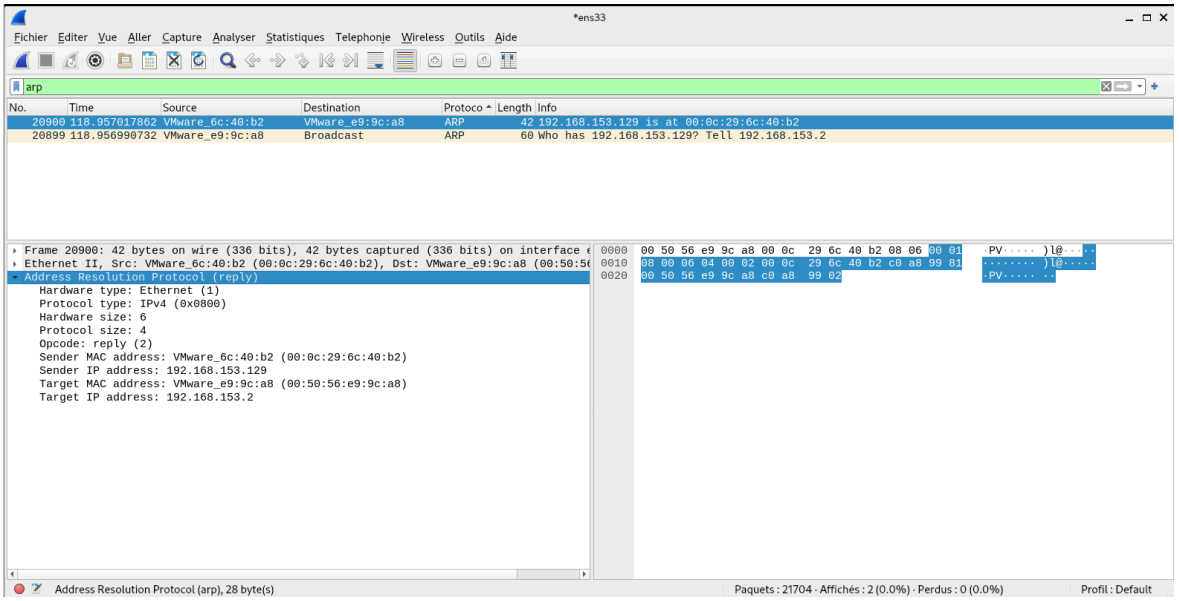
- ***MAC source :*** `00:29:6c:40:b2`
- ***MAC destination :*** `00:50:56:e9:9c:a8`
- ***IP source :*** `192.168.153.129`
- ***IP destination :*** `142.251.168.84`
- ***Source Port : 34578 (client)***
- ***Destination Port : 443 (serveur, HTTPS)***
- ***Flags : 0x010 (ACK) → ici, le paquet montré correspond à l'étape d'acquittement (ACK)***
- ***Sequence Number et Acknowledgment Number : utilisés pour assurer la fiabilité du transfert de données.***
- ***Ce mécanisme garantit que les deux parties sont prêtes à communiquer et synchronise les numéros de séquence pour un échange fiable des données***

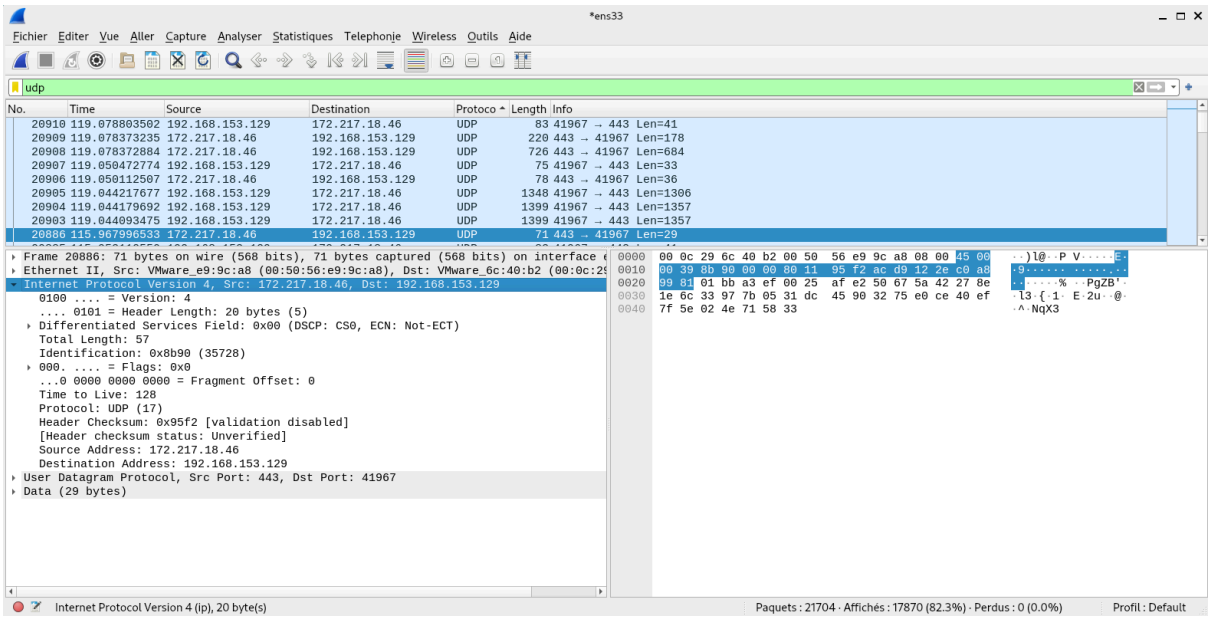
Décrivez le mécanisme de connexion avec un diagramme.

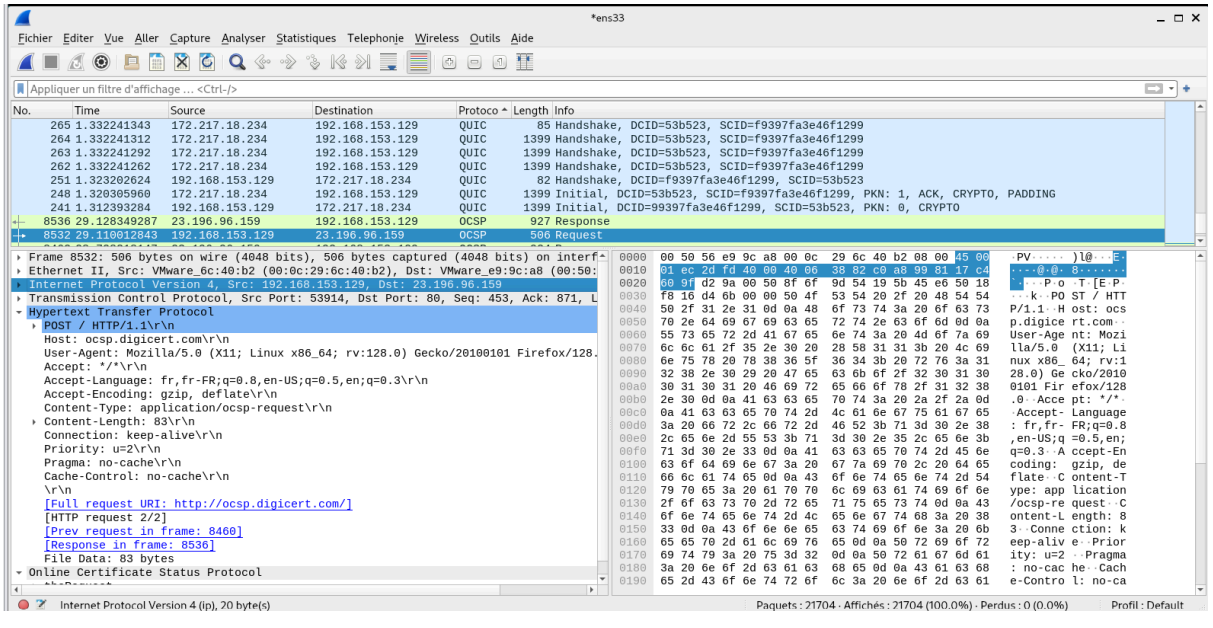
```text
Client                          Serveur
   |      SYN (Seq=x)             |
   |----------------------------->|
   |                              |
   |      SYN-ACK (Seq=y, Ack=x+1)|
   |<-----------------------------|
   |                              |
   |      ACK (Ack=y+1)           |
   |----------------------------->|
```

X = 104 , Y = 41

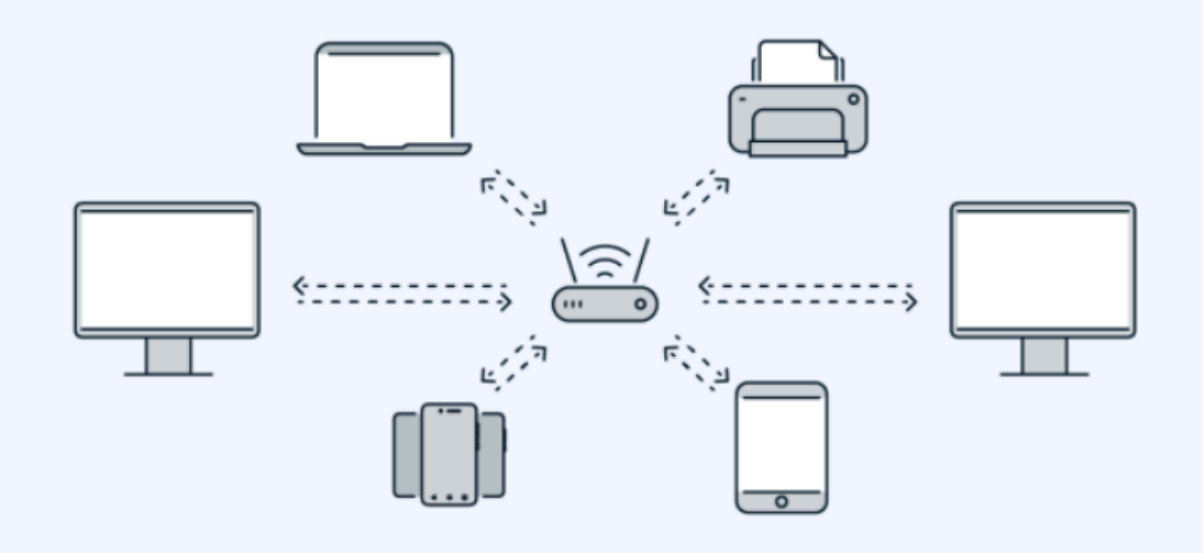
Documentez-vous sur le sujet et faites quelques tests pour n'afficher que les trames qui vous intéressent.




*ens33
Fichier Editer Vue Aller Capture Analyser Statistiques Telephonie Wireless Outils Aide
udp
No. Time Source Destination Protoco Length Info
20910 119.078803502 192.168.153.129 172.217.18.46 UDP 83 41967 → 443 Len=41
20909 119.078373235 172.217.18.46 192.168.153.129 UDP 220 443 → 41967 Len=178
20908 119.078372884 172.217.18.46 192.168.153.129 UDP 726 443 → 41967 Len=684
20907 119.050472774 192.168.153.129 172.217.18.46 UDP 75 41967 → 443 Len=33
20906 119.050112507 172.217.18.46 192.168.153.129 UDP 78 443 → 41967 Len=36
20905 119.044217677 192.168.153.129 172.217.18.46 UDP 1348 41967 → 443 Len=1306
20904 119.044179692 192.168.153.129 172.217.18.46 UDP 1399 41967 → 443 Len=1357
20903 119.044093475 192.168.153.129 172.217.18.46 UDP 1399 41967 → 443 Len=1357
20886 115.967996533 172.217.18.46 192.168.153.129 UDP 71 443 → 41967 Len=29
Frame 20886: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface
Ethernet II, Src: VMware_e9:9c:a8 (00:50:56:e9:9c:a8), Dst: VMware_6c:40:b2 (00:0c:2
Internet Protocol Version 4, Src: 172.217.18.46, Dst: 192.168.153.129
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 57
Identification: 0x8b90 (35728)
000. .... = Flags: 0x0
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: UDP (17)
Header Checksum: 0x95f2 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.217.18.46
Destination Address: 192.168.153.129
User Datagram Protocol, Src Port: 443, Dst Port: 41967
Data (29 bytes)
0000 00 0c 29 6c 40 b2 00 50 56 e9 9c a8 08 00 45 00
0010 00 39 8b 90 00 00 80 11 95 f2 ac d9 12 2e c0 a8
0020 99 81 01 bb a3 ef 00 25 af e2 50 67 5a 42 27 8e
0030 1e 6c 33 97 7b 05 31 dc 45 90 32 75 e0 ce 40 ef
0040 7f 5e 02 4e 71 58 33
Internet Protocol Version 4 (ip), 20 byte(s)
Paquets : 21704 · Affichés : 17870 (82.3%) · Perdus : 0 (0.0%)
Profil : Default


*ens33
Fichier Editer Vue Aller Capture Analyser Statistiques Telephonie Wireless Outils Aide
Appliquer un filtre d'affichage … <Ctrl-/>
No. Time Source Destination Protoco Length Info
265 1.332241343 172.217.18.234 192.168.153.129 QUIC 85 Handshake, DCID=53b523, SCID=f9397fa3e46f1299
264 1.332241312 172.217.18.234 192.168.153.129 QUIC 1399 Handshake, DCID=53b523, SCID=f9397fa3e46f1299
263 1.332241292 172.217.18.234 192.168.153.129 QUIC 1399 Handshake, DCID=53b523, SCID=f9397fa3e46f1299
262 1.332241262 172.217.18.234 192.168.153.129 QUIC 1399 Handshake, DCID=53b523, SCID=f9397fa3e46f1299
251 1.323202624 192.168.153.129 172.217.18.234 QUIC 82 Handshake, DCID=f9397fa3e46f1299, SCID=53b523
248 1.320305960 172.217.18.234 192.168.153.129 QUIC 1399 Initial, DCID=53b523, SCID=f9397fa3e46f1299, PKN: 1, ACK, CRYPTO, PADDING
241 1.312393284 192.168.153.129 172.217.18.234 QUIC 1399 Initial, DCID=99397fa3e46f1299, SCID=53b523, PKN: 0, CRYPTO
8536 29.128349287 23.196.96.159 192.168.153.129 OCSP 927 Response
8532 29.110012843 192.168.153.129 23.196.96.159 OCSP 506 Request
Frame 8532: 506 bytes on wire (4048 bits), 506 bytes captured (4048 bits) on interf
Ethernet II, Src: VMware_6c:40:b2 (00:0c:29:6c:40:b2), Dst: VMware_e9:9c:a8 (00:50:
Internet Protocol Version 4, Src: 192.168.153.129, Dst: 23.196.96.159
Transmission Control Protocol, Src Port: 53914, Dst Port: 80, Seq: 453, Ack: 871, L
Hypertext Transfer Protocol
POST / HTTP/1.1\r\n
Host: ocsp.digicert.com\r\n
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.
Accept: */*\r\n
Accept-Language: fr,fr-FR;q=0.8,en-US;q=0.5,en;q=0.3\r\n
Accept-Encoding: gzip, deflate\r\n
Content-Type: application/ocsp-request\r\n
Content-Length: 83\r\n
Connection: keep-alive\r\n
Priority: u=2\r\n
Pragma: no-cache\r\n
Cache-Control: no-cache\r\n
\r\n
[Full request URI: http://ocsp.digicert.com/]
[HTTP request 2/2]
[Prev request in frame: 8460]
[Response in frame: 8536]
File Data: 83 bytes
Online Certificate Status Protocol
0000 00 50 56 e9 9c a8 00 0c 29 6c 40 b2 08 00 45 00
0010 01 ec 2d fd 40 00 40 06 38 82 c0 a8 99 81 17 c4
0020 60 9f d2 9a 00 50 8f 6f 9d 54 19 5b 45 e6 50 18
0030 f8 16 d4 6b 00 00 50 4f 53 54 20 2f 20 48 54 54
0040 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 6f 63 73
0050 70 2e 64 69 67 69 63 65 72 74 2e 63 6f 6d 0d 0a
0060 55 73 65 72 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69
0070 6c 6c 61 2f 35 2e 30 20 28 58 31 31 3b 20 4c 69
0080 6e 75 78 20 78 38 36 5f 36 34 3b 20 72 76 3a 31
0090 32 38 2e 30 29 20 47 65 63 6b 6f 2f 32 30 31 30
00a0 30 31 30 31 20 46 69 72 65 66 6f 78 2f 31 32 38
00b0 2e 30 0d 0a 41 63 63 65 70 74 3a 20 2a 2f 2a 0d
00c0 0a 41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65
00d0 3a 20 66 72 2c 66 72 2d 46 52 3b 71 3d 30 2e 38
00e0 2c 65 6e 2d 55 53 3b 71 3d 30 2e 35 2c 65 6e 3b
00f0 71 3d 30 2e 33 0d 0a 41 63 63 65 70 74 2d 45 6e
0100 63 6f 64 69 6e 67 3a 20 67 7a 69 70 2c 20 64 65
0110 66 6c 61 74 65 0d 0a 43 6f 6e 74 65 6e 74 2d 54
0120 79 70 65 3a 20 61 70 70 6c 69 63 61 74 69 6f 6e
0130 2f 6f 63 73 70 2d 72 65 71 75 65 73 74 0d 0a 43
0140 6f 6e 74 65 6e 74 2d 4c 65 6e 67 74 68 3a 20 38
0150 33 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 6b
0160 65 65 70 2d 61 6c 69 76 65 0d 0a 50 72 61 67 6d
0170 61 3a 20 6e 6f 2d 63 61 63 68 65 0d 0a 43 61 63
0180 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20 6e 6f 2d 63
0190 61
Internet Protocol Version 4 (ip), 20 byte(s)
Paquets : 21704 · Affichés : 21704 (100.0%) · Perdus : 0 (0.0%)
Profil : Default

Partie 2



Installation des services qui nous permettront d'écouter quelques-uns des protocoles suivants :

Protocole	Commande principale d'installation
DHCP	`sudo apt install isc-dhcp-server`
DNS	`sudo apt install bind9`
mDNS	`sudo apt install avahi-daemon`
SSL/TLS	(via Apache/FTP/Autre)
FTP	`sudo apt install vsftpd` ou `proftpd`
SMB	`sudo apt install samba`
HTTPS	(via Apache/Nginx + SSL)

Installation du service DHCP :

```
root@zedel:/home/zedel# apt install isc-dhcp-server
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 :
  policycoreutils selinux-utils
Paquets suggérés :
  isc-dhcp-server-ldap ieee-data
Les NOUVEAUX paquets suivants seront installés :
  isc-dhcp-server policycoreutils selinux-utils
0 mis à jour, 3 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 1 766 ko dans les archives.
Après cette opération, 7 818 ko d'espace disque supplémentaires seront utilisés.
Souhaitez-vous continuer ? [O/n] o
```

Configuration du fichier /etc/dhcp/dhcpd.conf :

```
# Configuration de base
authoritative;
default-lease-time 600;
max-lease-time 7200;

# Déclaration du sous-réseau
subnet 192.168.12.0 netmask 255.255.255.0 {
    range 192.168.12.100 192.168.12.200;          # Plage DHCP
    option routers 192.168.12.2;                  # Passerelle par défaut
    option subnet-mask 255.255.255.0;
    option domain-name-servers 8.8.8.8, 8.8.4.4;  # DNS Google
    option broadcast-address 192.168.12.255;      # Adresse de broadcast
}
```

paquet HTTP :

Qu'est-ce que mDNS ?

Le mDNS (Multicast DNS) permet à des appareils sur un même réseau local de se découvrir et de résoudre des noms d'hôtes sans serveur DNS central. Il utilise le port UDP 5353 et la diffusion multicast pour répondre aux requêtes de résolution de noms sur le réseau local.

FTP sans TLS : peut-on récupérer des données sensibles ?

Oui, lors d'un échange FTP sans chiffrement (FTP simple sur port 21), toutes les données, y compris les identifiants (nom d'utilisateur, mot de passe), transitent en clair dans les paquets réseau. En capturant ce trafic avec Wireshark, il est donc possible de lire en texte clair les mots de passe et autres informations sensibles échangées lors de l'authentification.

Est-ce aussi le cas avec SSL/TLS ?

Non. Quand les échanges FTP ou HTTP sont protégés par SSL/TLS (FTPS, HTTPS), les données sont chiffrées : il n'est plus possible de lire les identifiants ou le contenu des échanges dans les paquets capturés, sauf si tu disposes de la clé privée du serveur ou utilises un proxy d'interception SSL/TLS.

Résumé :

- ***En FTP non sécurisé, les identifiants sont visibles en clair dans les paquets capturés.***
- ***En FTP sécurisé (SSL/TLS), les identifiants et données sont chiffrés et donc protégés.***

Installation de tshark :

```
zedel@zedel:~$ sudo apt install tshark
[sudo] Mot de passe de zedel :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  tshark
0 mis à jour, 1 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 163 ko dans les archives.
Après cette opération, 406 ko d'espace disque supplémentaires seront utili
Réception de :1 http://deb.debian.org/debian bookworm/main amd64 tshark ar
163 ko réceptionnés en 0s (1 606 ko/s)
Sélection du paquet tshark précédemment désélectionné.
(Lecture de la base de données... 167289 fichiers et répertoires déjà inst
Préparation du dépaquetage de .../tshark_4.0.17-0+deb12u1_amd64.deb ...
Dépaquetage de tshark (4.0.17-0+deb12u1) ...
Paramétrage de tshark (4.0.17-0+deb12u1) ...
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ..
```

Commande tshark qui nous permet d'écouter et de capturer les paquets de quelques-uns de protocoles listés.

```
zedel@zedel:~$ tshark -i ens33 -Y "dhcp"
Capturing on 'ens33'
 ** (tshark:43941) 09:40:35.298208 [Main MESSAGE] -- Capture started.
 ** (tshark:43941) 09:40:35.298341 [Main MESSAGE] -- File: "/tmp/wireshark_ens33557W62.pcapng"
  109 106.984589209 192.168.12.134 → 192.168.12.133 DHCP 325 DHCP Request  - Transaction ID 0x4e39d118
  110 106.988847673 192.168.12.133 → 192.168.12.134 DHCP 342 DHCP ACK      - Transaction ID 0x4e39d118
  392 123.117000278      0.0.0.0 → 255.255.255.255 DHCP 342 DHCP Discover - Transaction ID 0x1a010d6e
  402 124.125310389 192.168.12.133 → 192.168.12.100 DHCP 342 DHCP Offer    - Transaction ID 0x1a010d6e
  406 124.126842606      0.0.0.0 → 255.255.255.255 DHCP 342 DHCP Request  - Transaction ID 0x1a010d6e
  407 124.130267485 192.168.12.133 → 192.168.12.100 DHCP 342 DHCP ACK      - Transaction ID 0x1a010d6e
```

Ici la commande utilisé est : 'tshark -i ens33 -Y "dhcp"

Détails des options utilisées par tshark :

- *-i ens33 = Cette option indique à tshark sur quelle interface réseau capturer les paquets.*
  - *Ici, **ens33** est le nom de l'interface réseau (par exemple, une carte Ethernet ou une interface virtuelle).*

- *-Y “dhcp” = Cette option applique un filtre d’affichage (display filter) pour n’afficher que les paquets correspondant au protocole DHCP.*
    - *Le filtre **"dhcp"** signifie que tshark ne montrera que les paquets analysés comme DHCP, même si d’autres paquets sont capturés.*
    - *Les filtres d’affichage utilisent la même syntaxe que dans Wireshark et sont très puissants pour cibler un protocole ou des champs précis.*

En d’autres termes, la commande serait :

- *tshark -i<nom interface réseau> -Y<protocole>.*