TP Wireshark

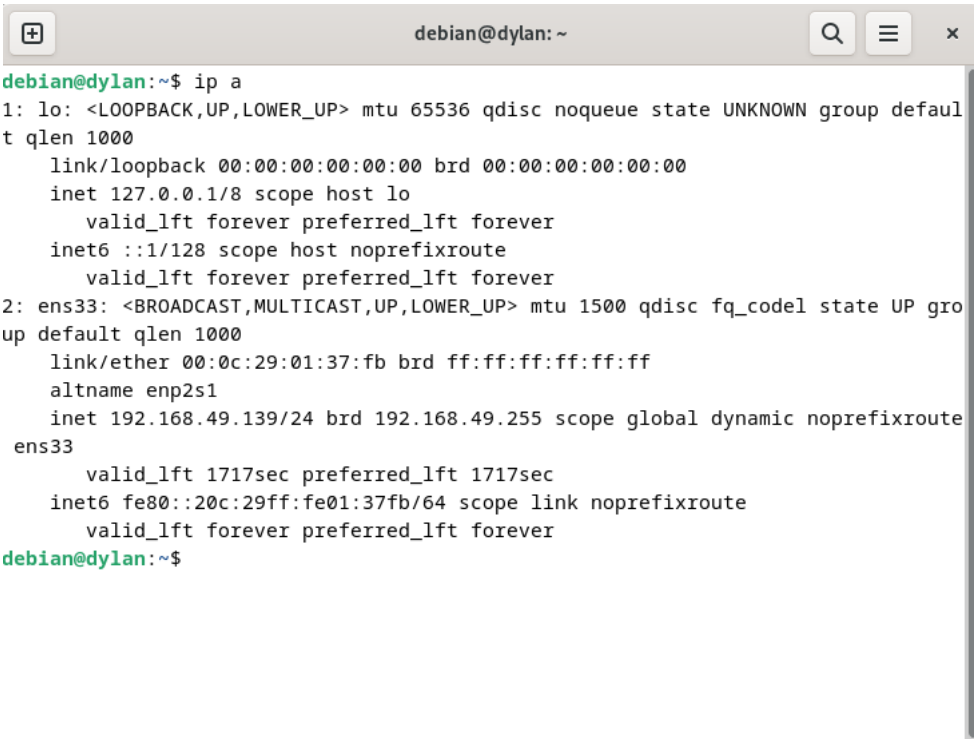
TP U6 SISR

SPINELLI Dylan
BTS SIO SISR | PARIS YNOV CAMPUS

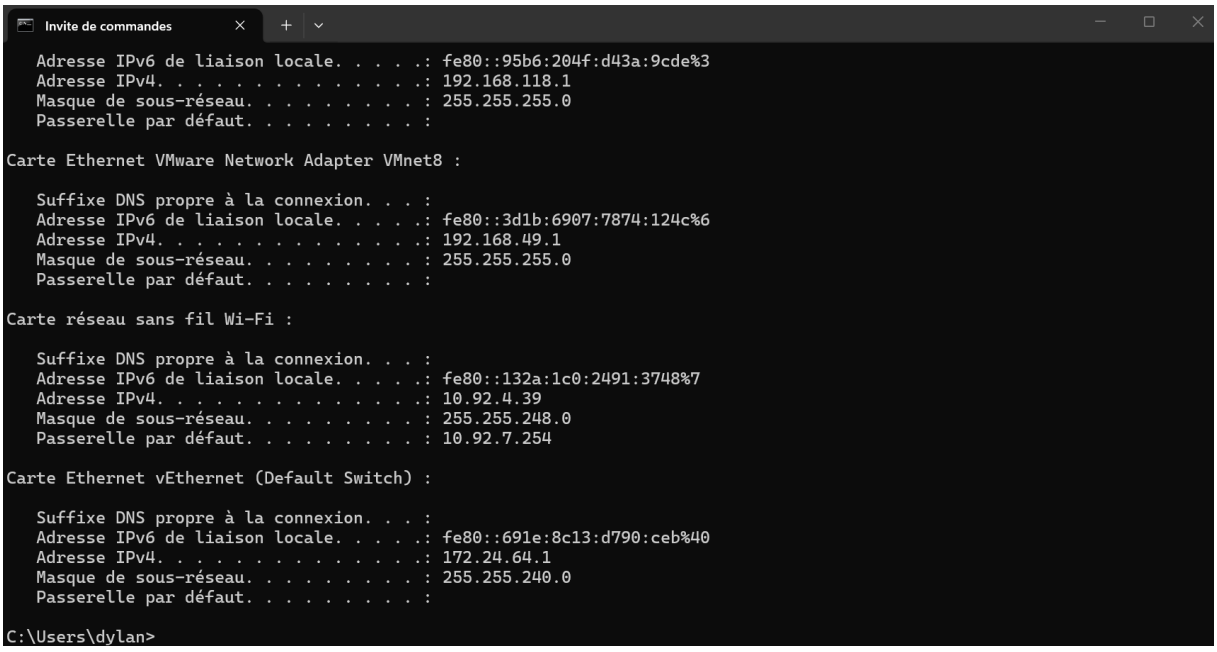
Table des matières

1-Capture et analyse du trafic dans Wireshark

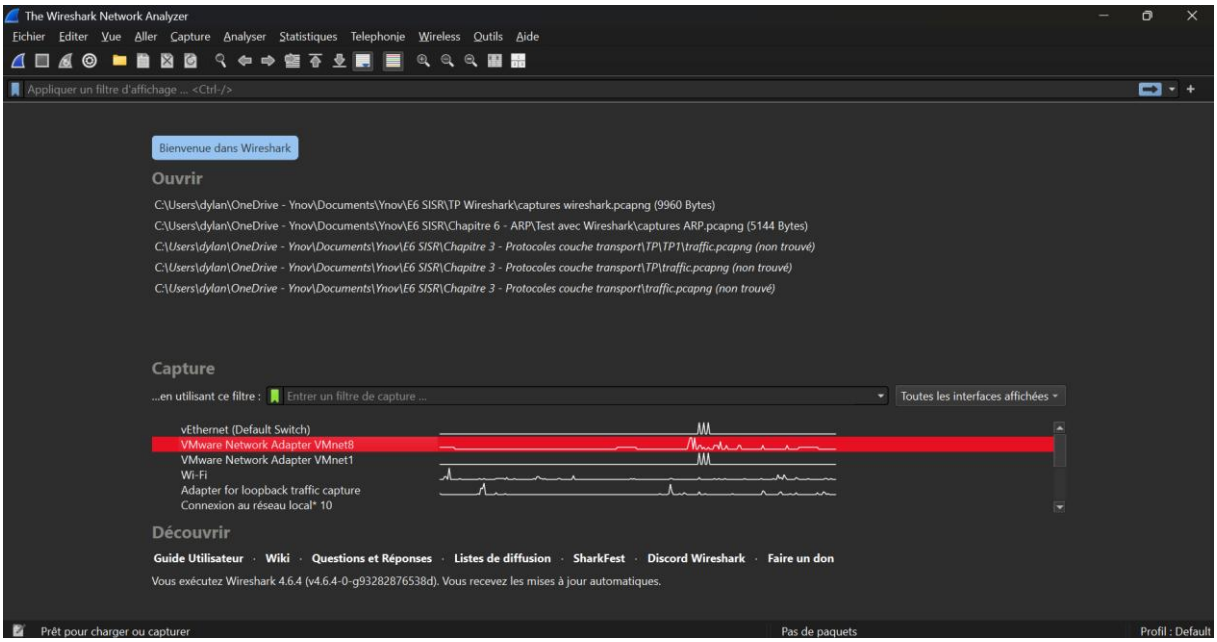
Voici l'adresse IP de notre machine virtuelle :



Voici l'interface réseau de la machine physique :

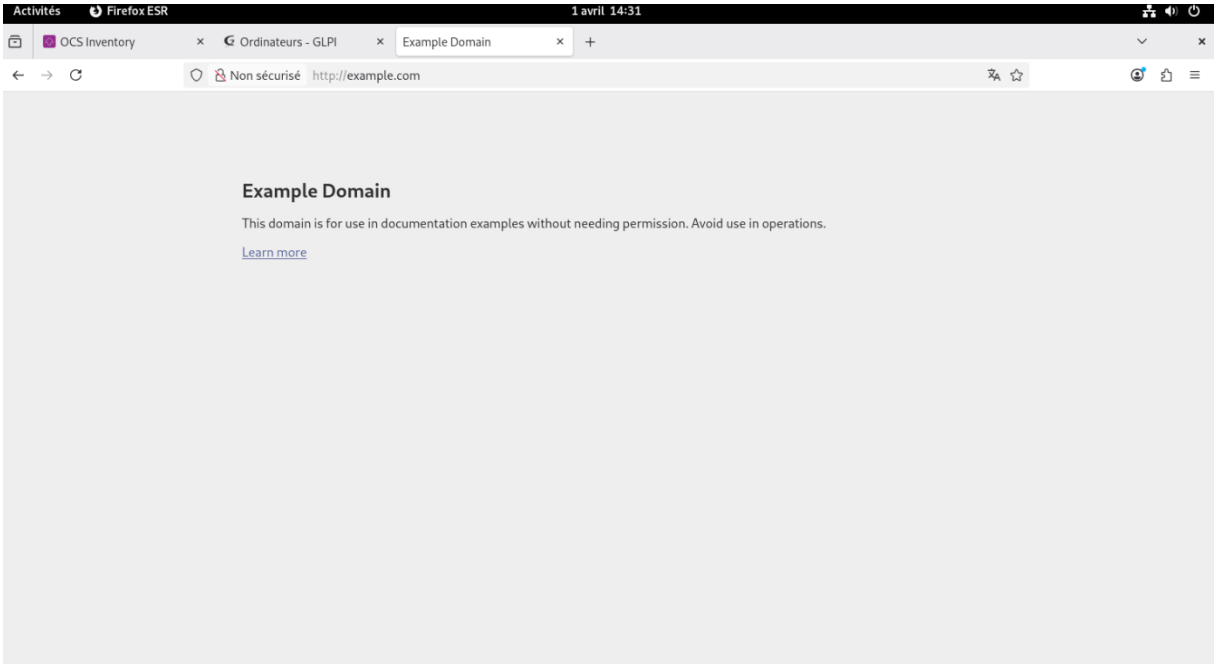


On sélectionne l'interface VMnet8 dans Wireshark qui a pour adresse IP 192.168.49.1 :



On démarre ensuite la capture du trafic dans Wireshark

On ouvre la page http://example.com afin d'analyser le trafic :



On ferme ensuite le navigateur web afin de pouvoir analyser le trafic lors de la fermeture de la connexion.

2-Analyse de la connexion TCP

Voici les principaux champs à connaitre pour le TCP :

Champ	Description
Source Port	Port de l'émetteur
Destination Port	Port du destinataire
Sequence Number	Numéro d'ordre du premier octet du segment
Acknowledgment Number	Prochain octet attendu par le récepteur
Flags (SYN, ACK, FIN, RST...)	Drapeaux de contrôle de la connexion
Window Size	Taille du tampon de réception (contrôle de flux)
Checksum	Vérification de l'intégrité du paquet
TCP Segment Length	Taille des données utiles du segment

L'établissement d'une connexion TCP se fait en 3 étapes :

• Étape 1 — SYN : Le client envoie un paquet avec le drapeau SYN activé. Il annonce son numéro de séquence initial (ISN).

• Étape 2 — SYN-ACK : Le serveur répond avec SYN + ACK. Il accuse réception du SYN client et annonce son propre ISN.

• Étape 3 — ACK : Le client accuse réception du SYN serveur. La connexion est Etablie

La fermeture propre d'une connexion TCP utilise 4 étapes (Four-Way Teardown) :

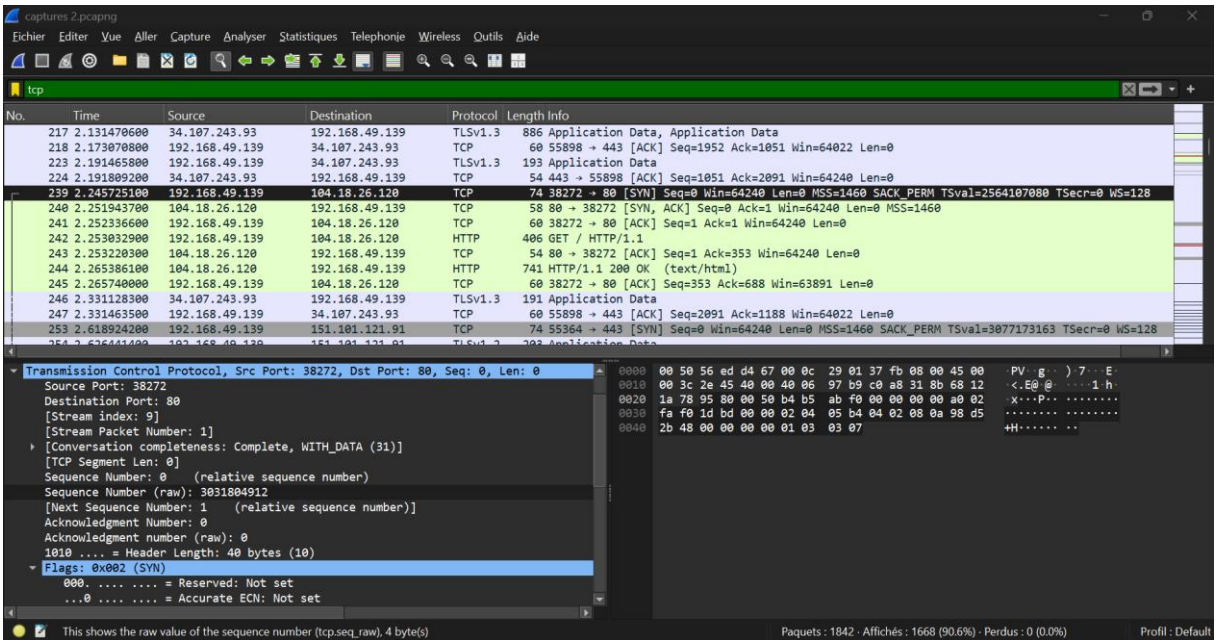
• L'hôte qui ferme envoie FIN + ACK

• L'autre hôte accuse réception avec ACK

• L'autre hôte envoie à son tour FIN + ACK

• Le premier hôte répond ACK. La connexion est fermée.

On regarde le premier segment TCP « SYN » :

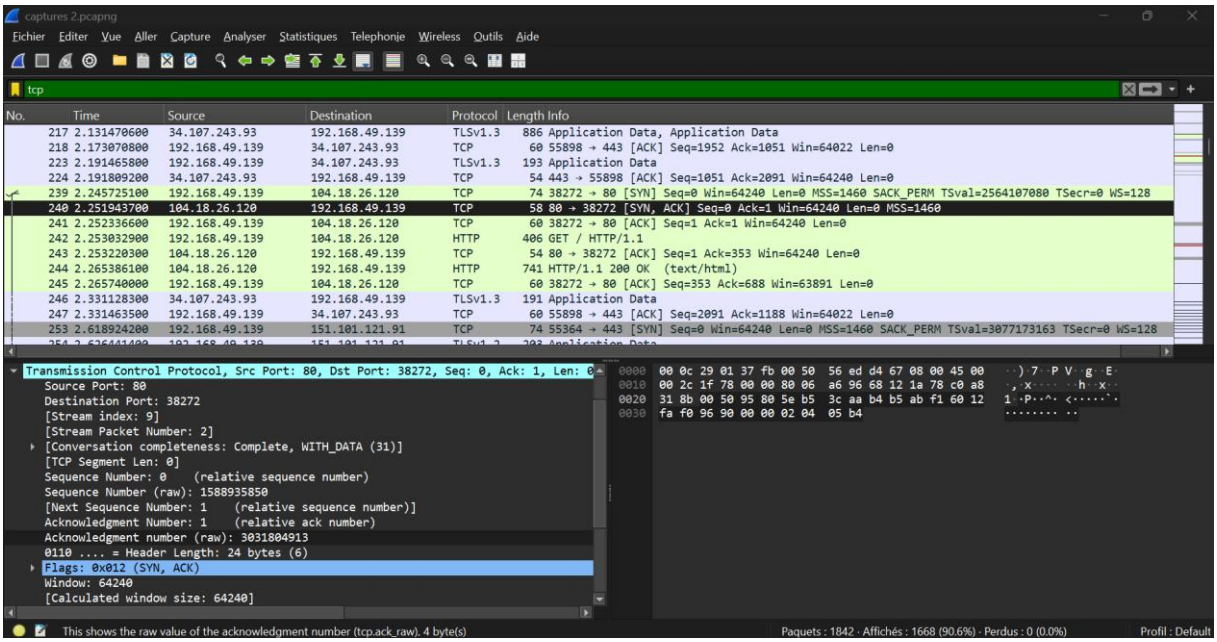


Le numéro de séquence du premier segment TCP « SYN » (Seq) = 3031804912

Le numéro relatif est = 0

Champ	Valeur observée	Signification
Source Port	38272	Port de l'émetteur
Destination Port	80	Port du destinataire
Sequence Number (raw)	3031804912	Numéro d'ordre du premier octet du segment
Header Length	40 bytes	Taille du header
Window Size	64240	Taille du tampon de réception (contrôle de flux)
Checksum	0x1dbd	Vérification de l'intégrité du paquet
TCP Segment Length	0	Taille des données utiles du segment

Analysons le segment suivant « SYN, ACK » :



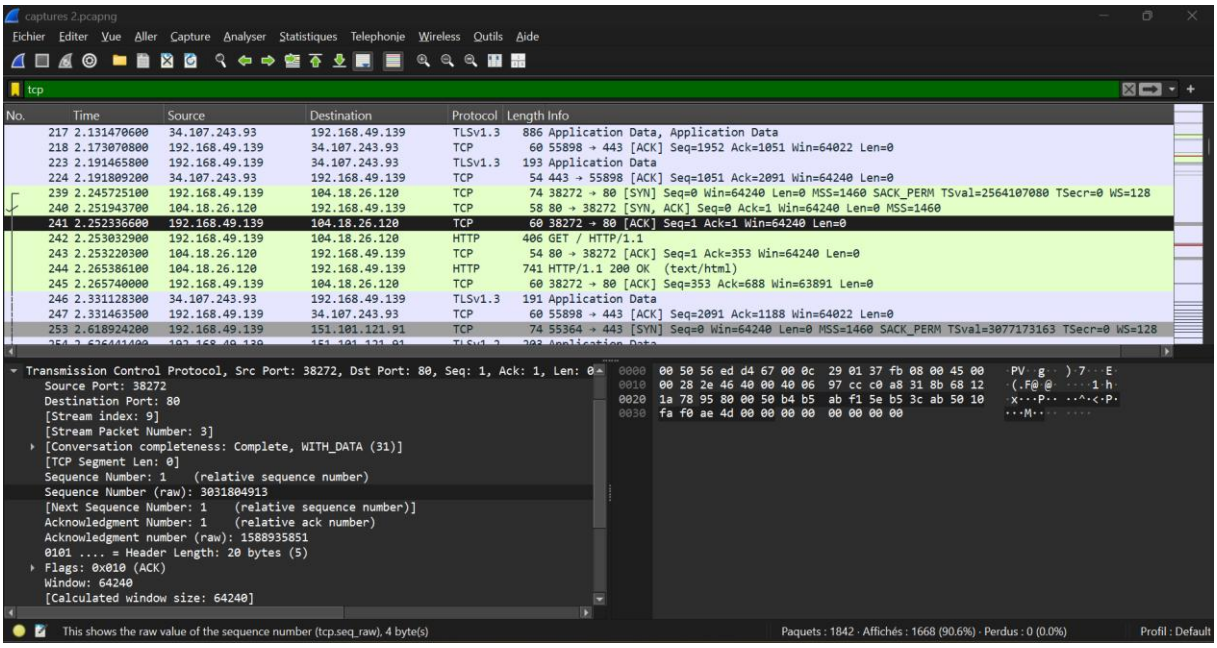
La valeur du champ acknowledgment number = 3031804913

La valeur est égale a celle du du segment « SYN » + 1. Cela signifie que le serveur distant à bien reçu le segment « SYN » que nous avons envoyé précédemment.

Le numéro de séquence initial (Seq) en valeur brute = 1588935850

Wireshark affiche 0 en valeur relative car maintenant que la connexion TCP est établie avec le 3 way handshake, le prochain segment contiendra les données que l'on souhaite envoyer ou recevoir.

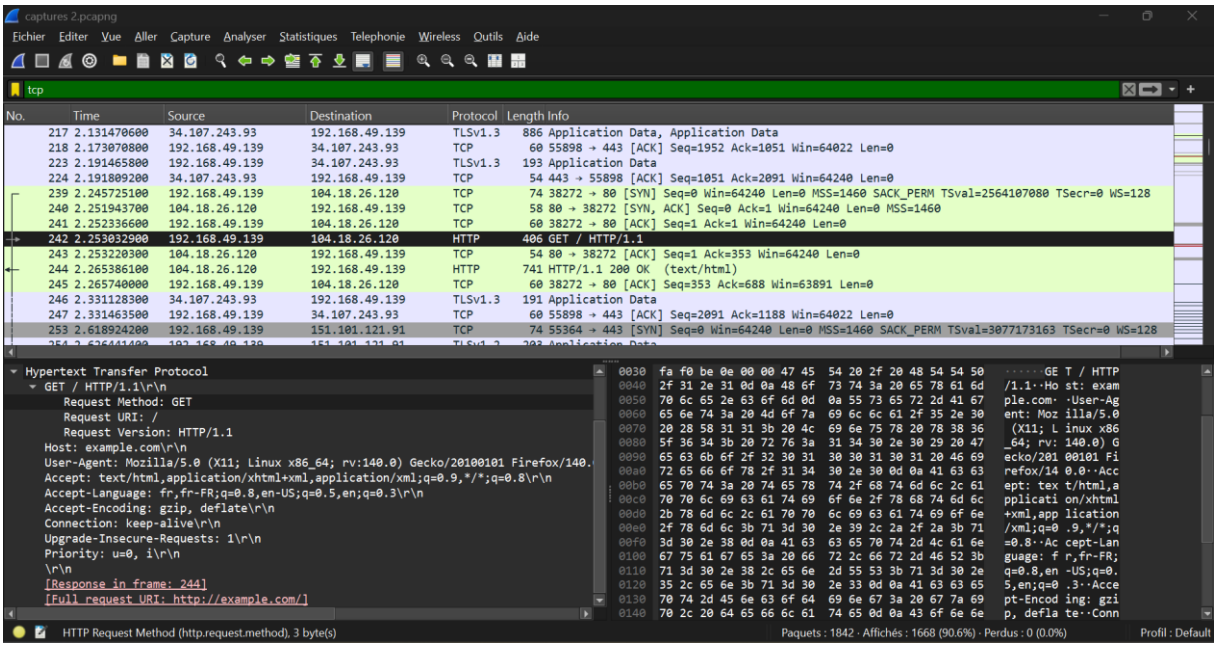
On trouve ensuite à la fin le segment TCP « ACK » :



Paquet	Etape	Drapeaux	IP source	IP destination	N° Sequence
9	SYN	SYN	192.168.49.139	104.18.26.120	3031804912
10	SYN-ACK	SYN-ACK	104.18.26.120	192.168.49.139	1588935850
11	ACK	ACK	192.168.49.139	104.18.26.120	3031804913

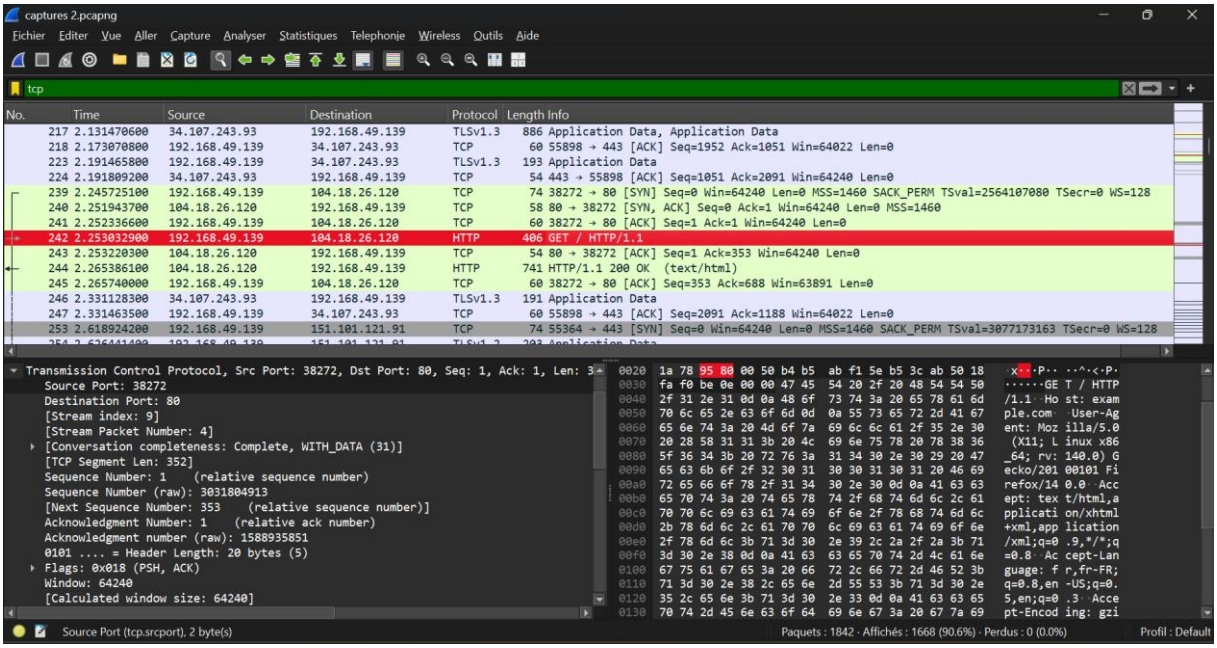
3-Analyse du trafic HTTP

Analysons la première requête HTTP « GET » :



La méthode HTTP utilisée est GET. La ressource demandée est « example.com »

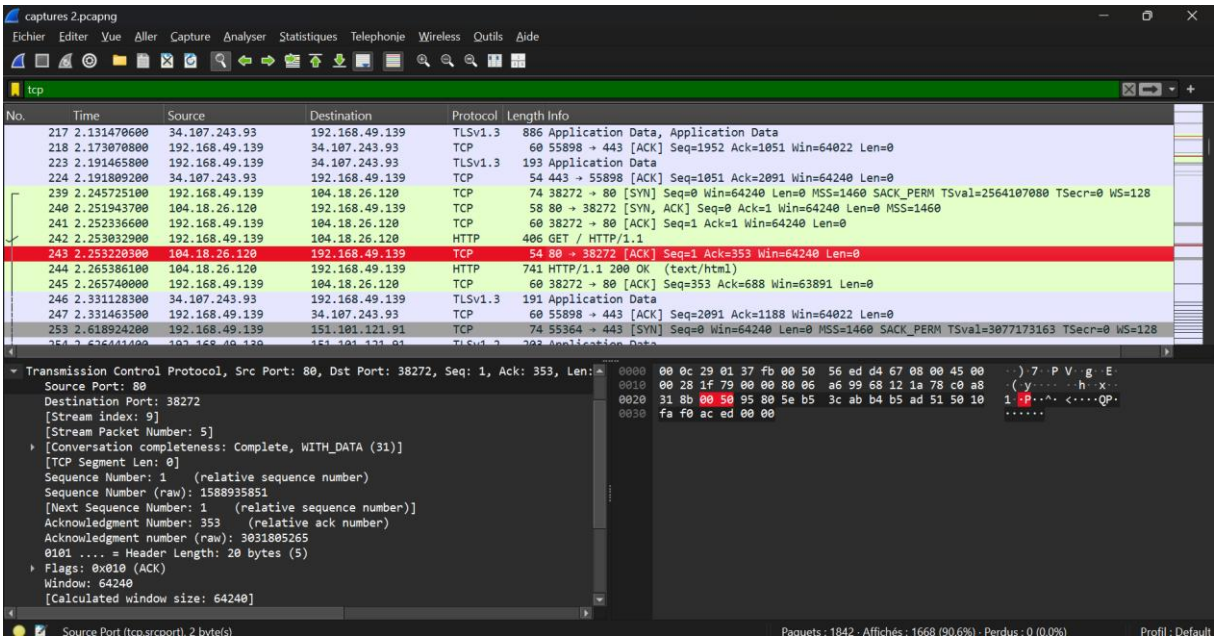
La version utilisée est HTTP/1.1



Le numéro de séquence (Seq) = 1

La taille du segment (TCP Segment Len) = 352

Le serveur renvois un segment « ACK » pour accuser réception de la requête « GET » :



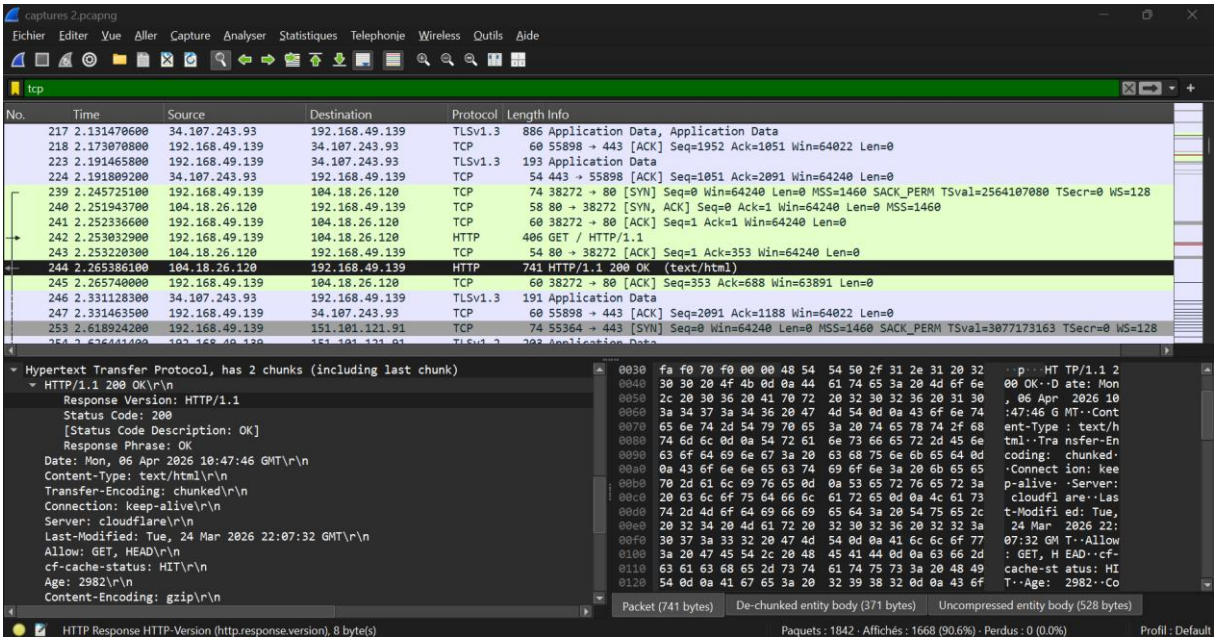
Dans l'ACK renvoyé par le serveur, le champ « Acknowledgment Number » à la valeur : 353

L'Acknowledgment Number (numéro d'acquittement) sert à indiquer à l'émetteur le prochain octet que le récepteur s'attend à recevoir. Il se calcule en prenant le numéro de séquence du segment reçu, auquel on ajoute la taille de ses données utiles (le *payload*).

Seq(reçu) + Len(reçu) = Ack(renvoyé)

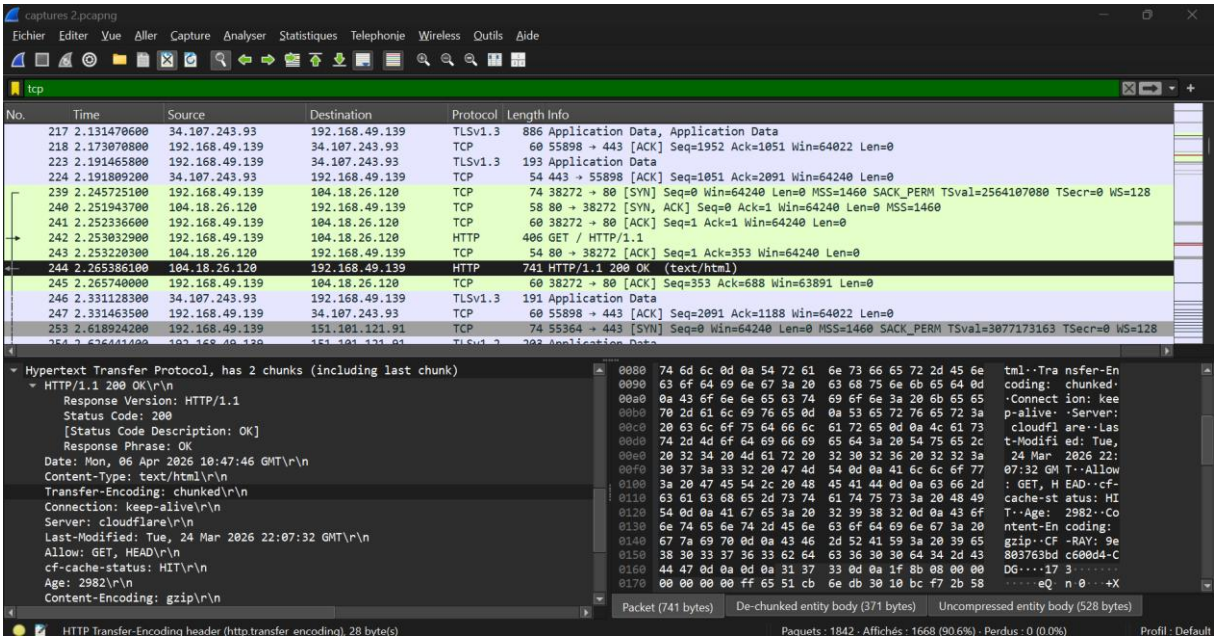
1 + 352 = 353

Le serveur envoie ensuite la réponse HTTP contenant le contenu de la page :

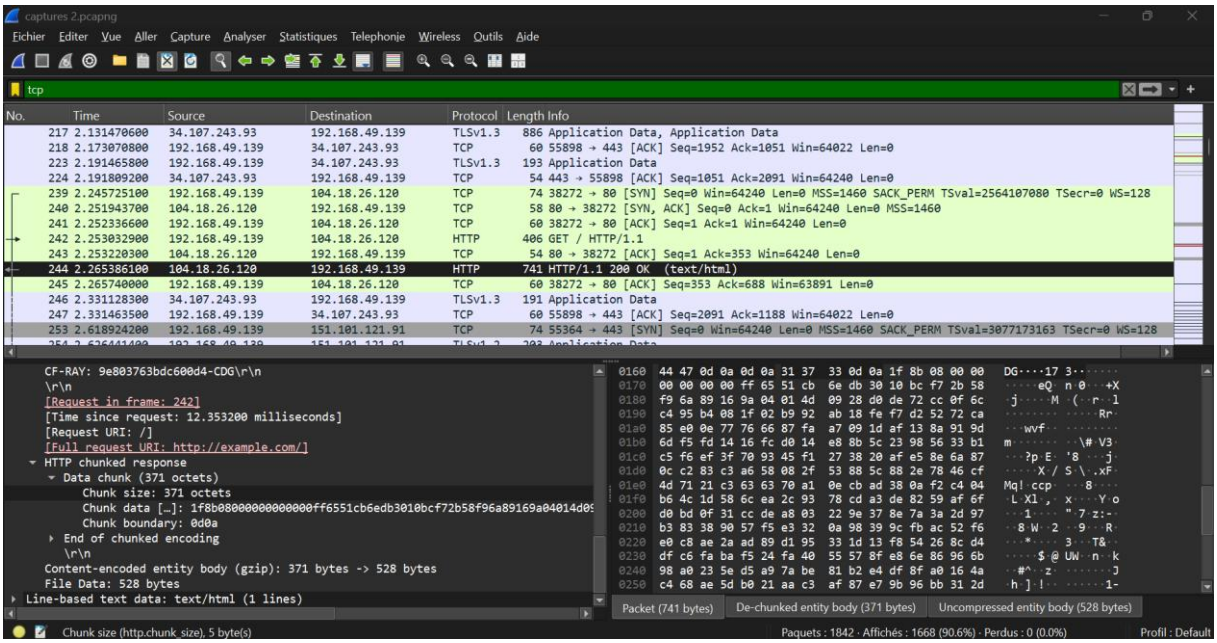


Le code renvoyé est bien « 200 OK »

Le contenu renvoyé par le serveur est envoyé par morceaux. Si on regarde « Tranfer-encoding : chunked » :

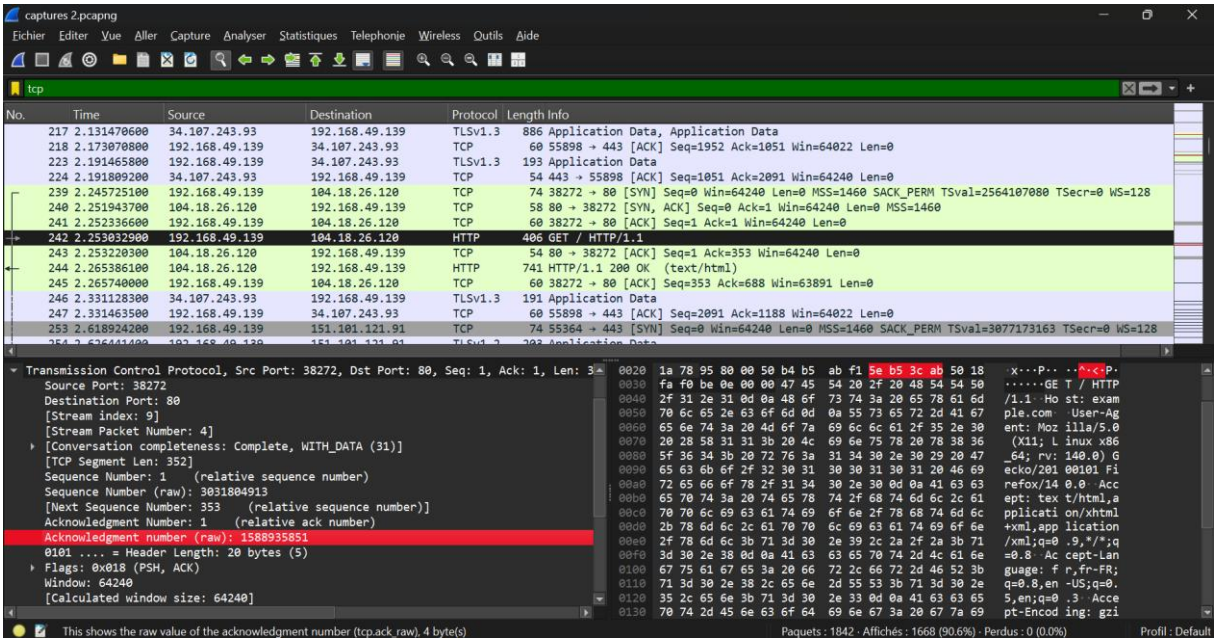


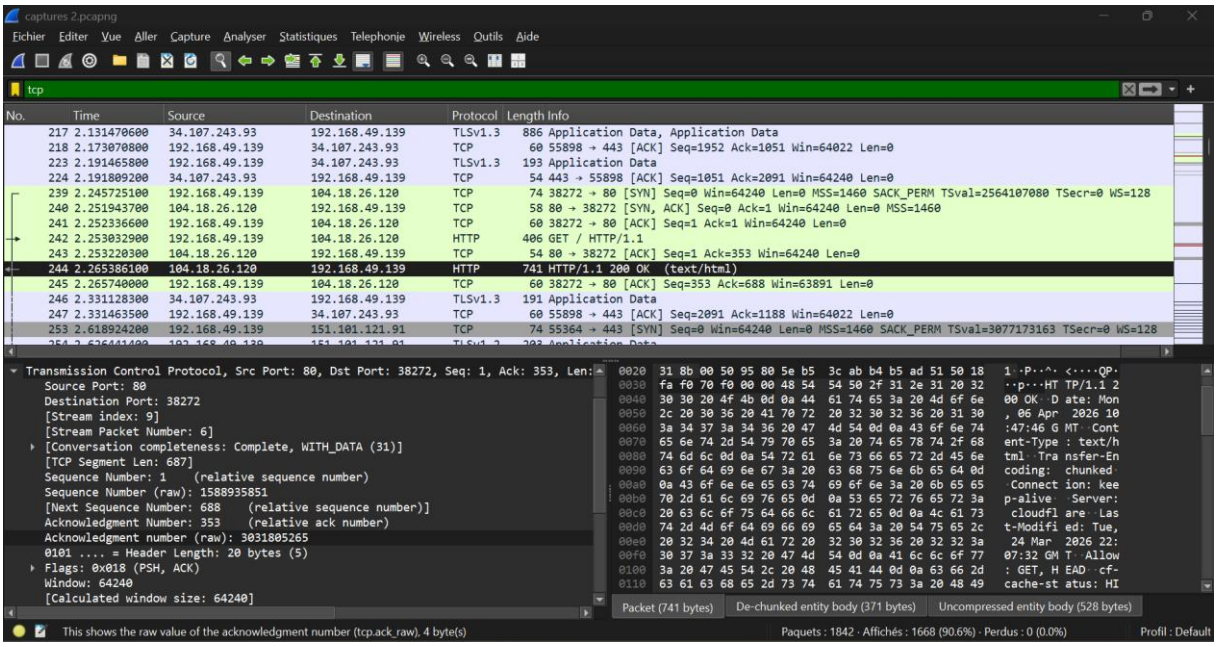
Si on regarde le menu « http chunked response », on voit de le serveur a envoyé un morceau de données de taille 371 octets :



En dessous, on voit le menu « File Data » qui nous indique que la taille totale des données envoyées est 528 bytes

Comparons les « Sequence number » et « acknowledgment number » entre la requête « GET » et la réponse du serveur :





Paquet HTTP	GET	200 OK
Sequence number relatif	1	1
Sequence number (raw)	3031804913	1588935851
Acknowledgment number relatif	1	353
Acknowledgment number (raw)	1588935851	3031805265

Schéma :

1. Envoi de la requête web

Requête HTTP GET

- Seq = 1 (Le client commence à parler)
- Ack = 1 (Le client confirme qu'il a reçu le SYN-ACK précédent)
- Len = 352 (La taille de la requête GET)

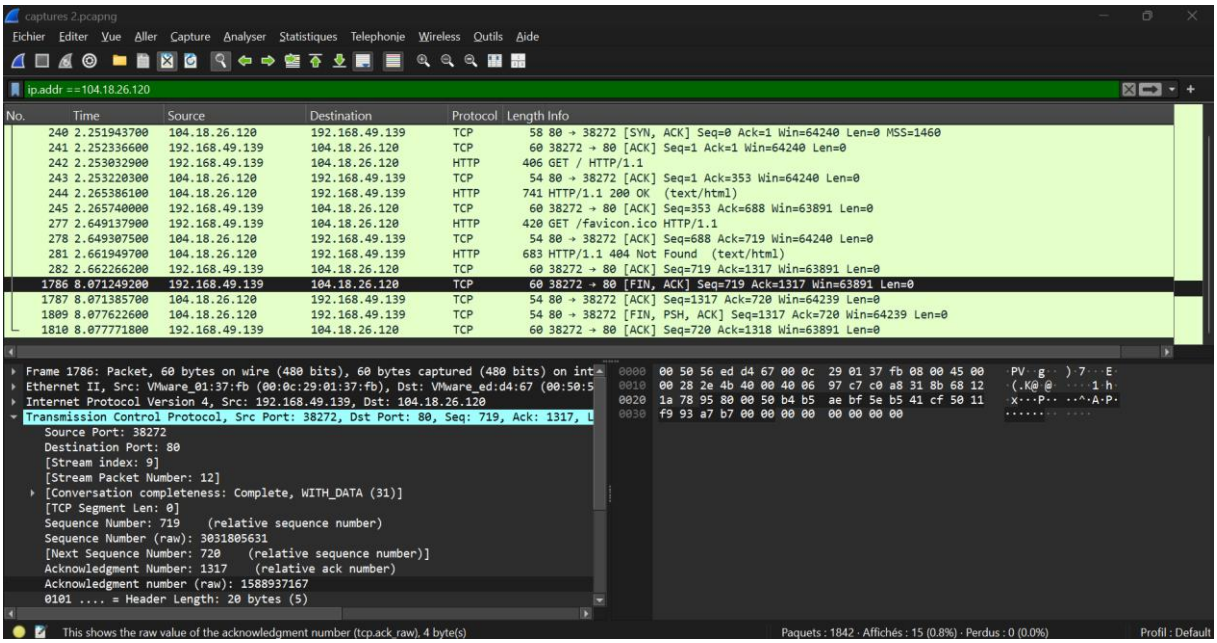
2. Traitement et réponse du serveur

Réponse HTTP 200 OK (Début des chunks)

- Seq = 1 (Le serveur commence à envoyer ses données web)
- Ack = 353 (Le serveur valide les 352 octets du GET : 1 + 352 = 353)

4- Analyse de la fermeture TCP

Analysons désormais la fermeture de la connexion TCP :

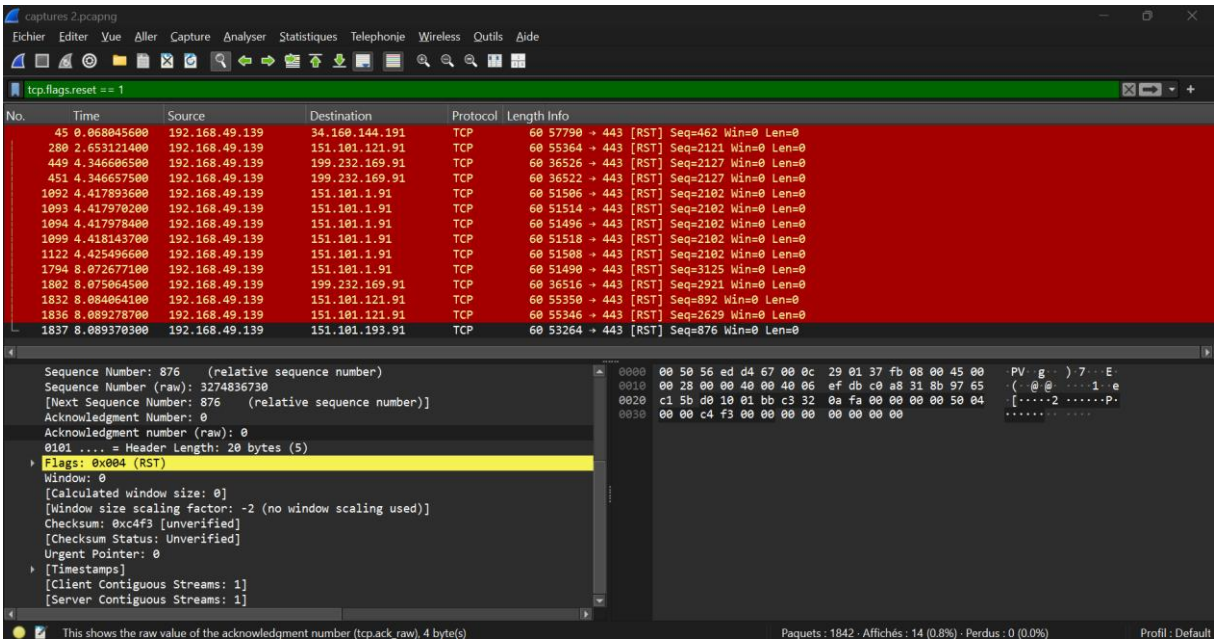


Paquet	Drapeau	Emetteur	Seq / Ack	Signification
#1786	FIN + ACK	Serveur	Seq relatif = 719 Seq (raw) = 3031805631 Ack relatif = 1317 Ack (raw) = 1588937167	Le serveur demande la fermeture
#1787	ACK	Client	Seq relatif = 1317 Seq (raw) = 1588937167 Ack relatif = 720 Ack (raw) = 3031805632	Le client confirme la fermeture du serveur
#1809	FIN + ACK	Client	Seq relatif = 1317 Seq (raw) = 1588937167 Ack relatif = 720 Ack (raw) = 3031805632	Le client ferme la connexion et la confirme
#1810	ACK	Serveur	Seq relatif = 720 Seq (raw) = 3031805632 Ack relatif = 1318 Ack (raw) = 1588937168	Le serveur confirme la fermeture du client

Le drapeau FIN incrémente le numéro d'acquittement de 1 comme le drapeau SYN car TCP doit garantir de manière absolue que les drapeaux SYN et FIN ont bien été reçus. C'est pour cela qu'ils doivent consommer un octet. Lors de l'acquittement, la présence de cet octet est nécessaire car elle prouve que l'autre machine a bien reçu l'information.

5-Cas des drapeaux RST

Le drapeau RST signifie une fermeture abrupte de connexion :

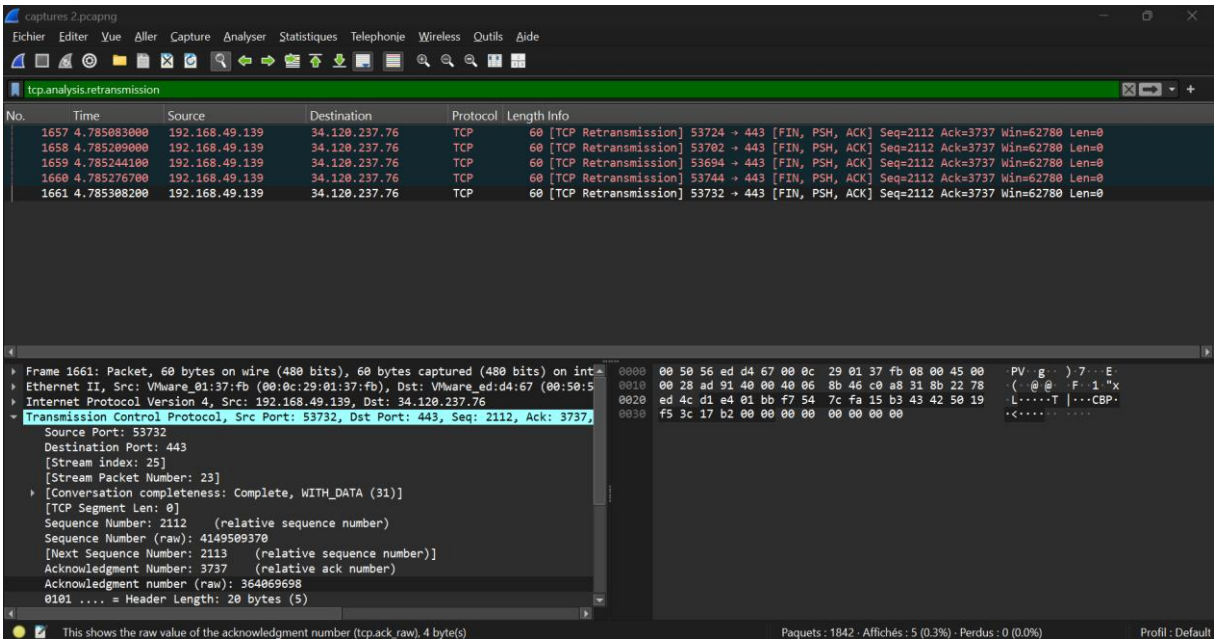


Nous avons plusieurs Drapeaux RST dans notre capture.

Dans un contexte normal, c'est le drapeau FIN qui met fin à une connexion. Le drapeau RST est plus brutal. Il peut survenir en cas de port fermé, de fermeture brutale de l'application, de blocage pare-feu, etc...

6-Analyse des retransmissions TCP

On utilise le filtre « tcp.analysis.retransmission » pour voir les retransmissions TCP :

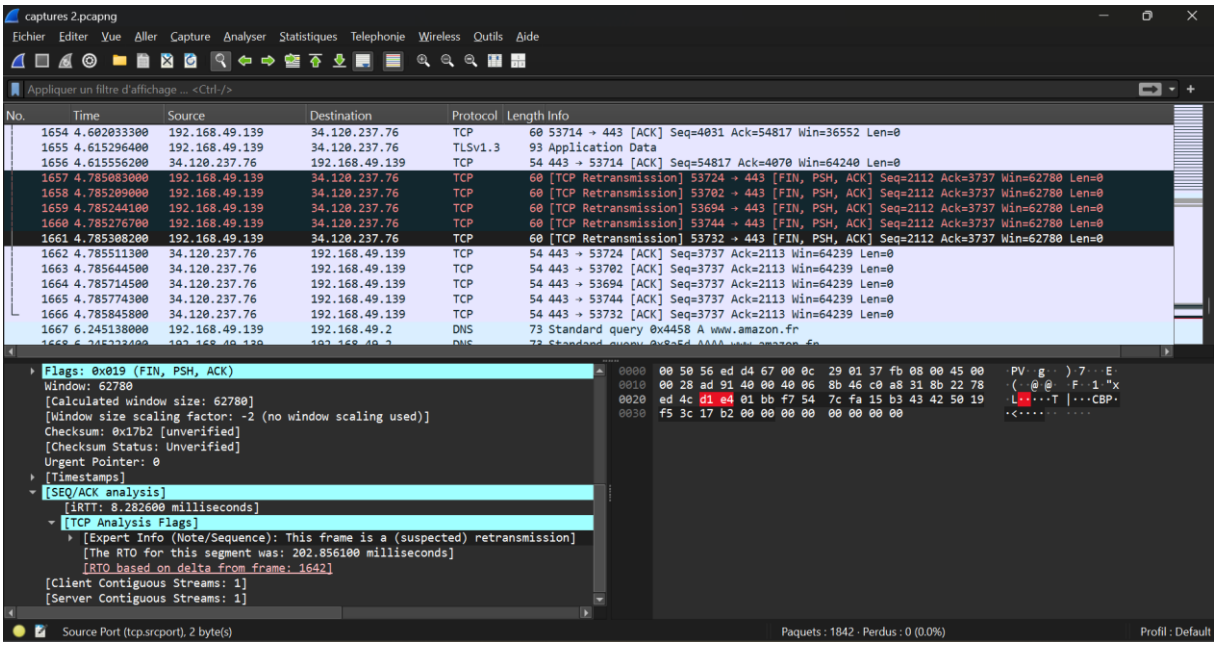


Wireshark les affiche en noir par défaut

Il existe deux types de retransmissions TCP :

Type	Mécanisme
Retransmission RTO	L'émetteur attend l'expiration d'un minuteur (RTO). Si aucun ACK n'est reçu avant, il retransmet le paquet.
Retransmission rapide	Si l'émetteur reçoit 3 ACK dupliqués consécutifs (dupthresh = 3), il retransmet immédiatement sans attendre le RTO.

On peut voir le RTO ainsi que les différents ACK envoyés :



Il s'agit retransmissions RTO car il n'y a pas de ACK dupliqués. Les retransmissions ont bien attendu la fin du RTO pour s'effectuer.

En temps normal, quand deux ordinateurs communiquent via TCP, le récepteur envoie un ACK (Acknowledgment / Accusé de réception) pour valider ce qu'il a reçu.

La subtilité de TCP, c'est que l'ACK indique le prochain paquet attendu, et non pas celui qui vient d'arriver.

- Si je reçois le paquet n°100, je renvoie un ACK 101

Le problème survient quand un paquet se perd, mais que les suivants arrivent. Imaginons qu'un serveur vous envoie les paquets 1, 2, 3, 4 et 5.

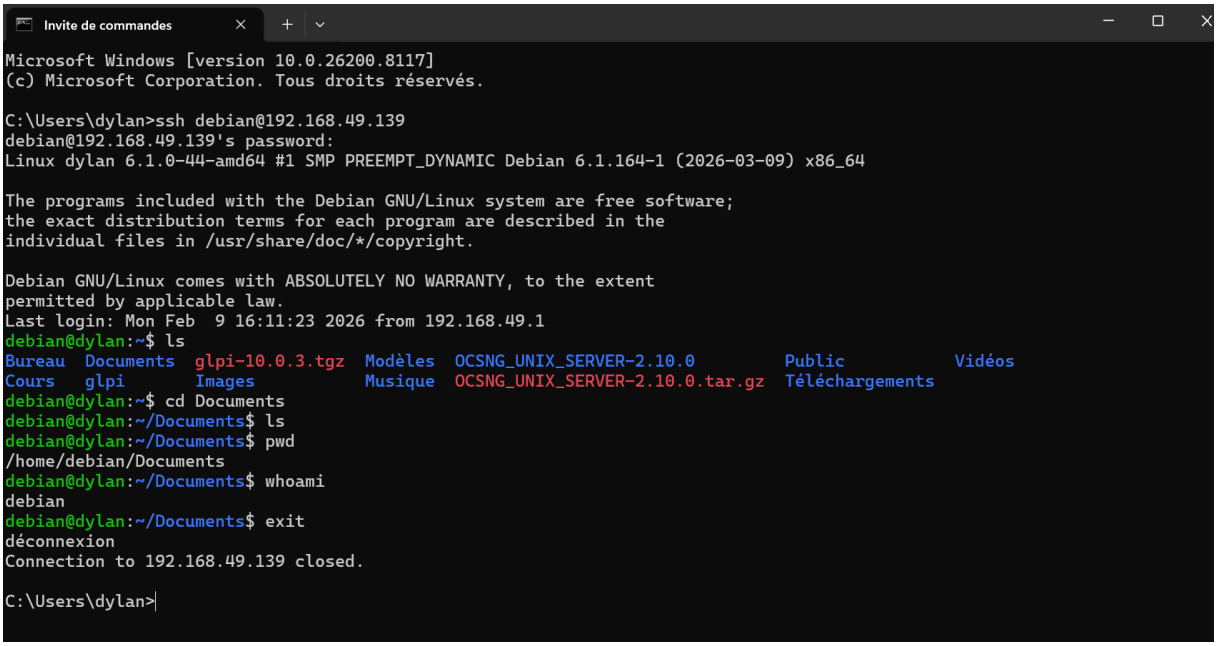
1. Vous recevez le paquet 1. Vous répondez : ACK 2 (J'attends le 2).
2. Le paquet 2 est perdu par le réseau.
3. Vous recevez le paquet 3. Vous ne pouvez pas accuser réception du 3, car il vous manque le 2 (TCP a besoin de tout dans l'ordre).
4. Votre ordinateur va donc répéter sa dernière demande valide : il renvoie un ACK 2. C'est cela, un ACK dupliqué (Dup ACK).

La retransmission rapide se déclenche à la réception de 3 ACK dupliqués consécutifs.

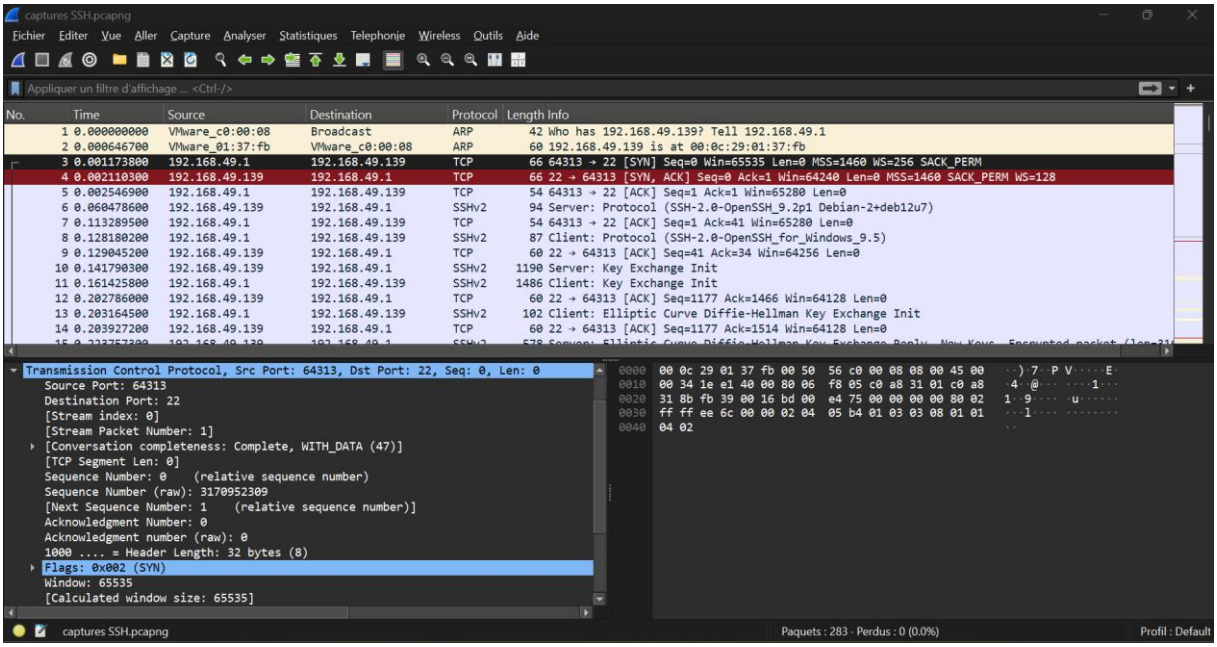
Au lieu d'attendre que le chronomètre (RTO) expire, le serveur renvoie immédiatement le paquet perdu au bout de 3 ACK

7-Analyse du trafic SSH

Connectons-nous en SSH à notre VM depuis notre machine physique et effectuons quelques commandes :

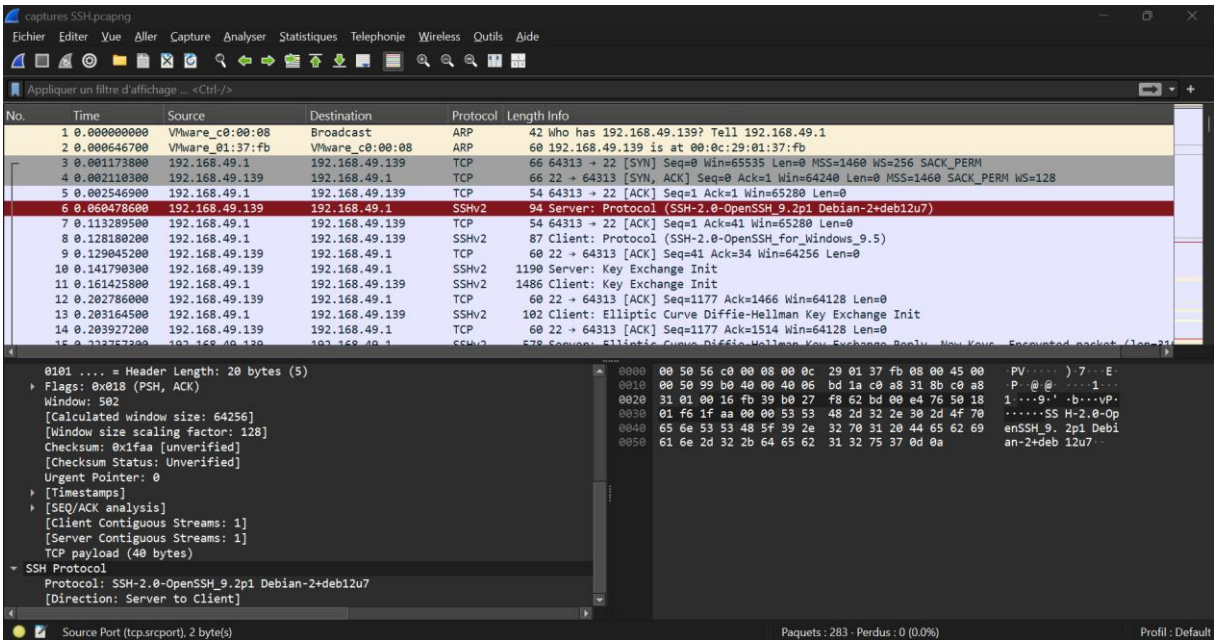


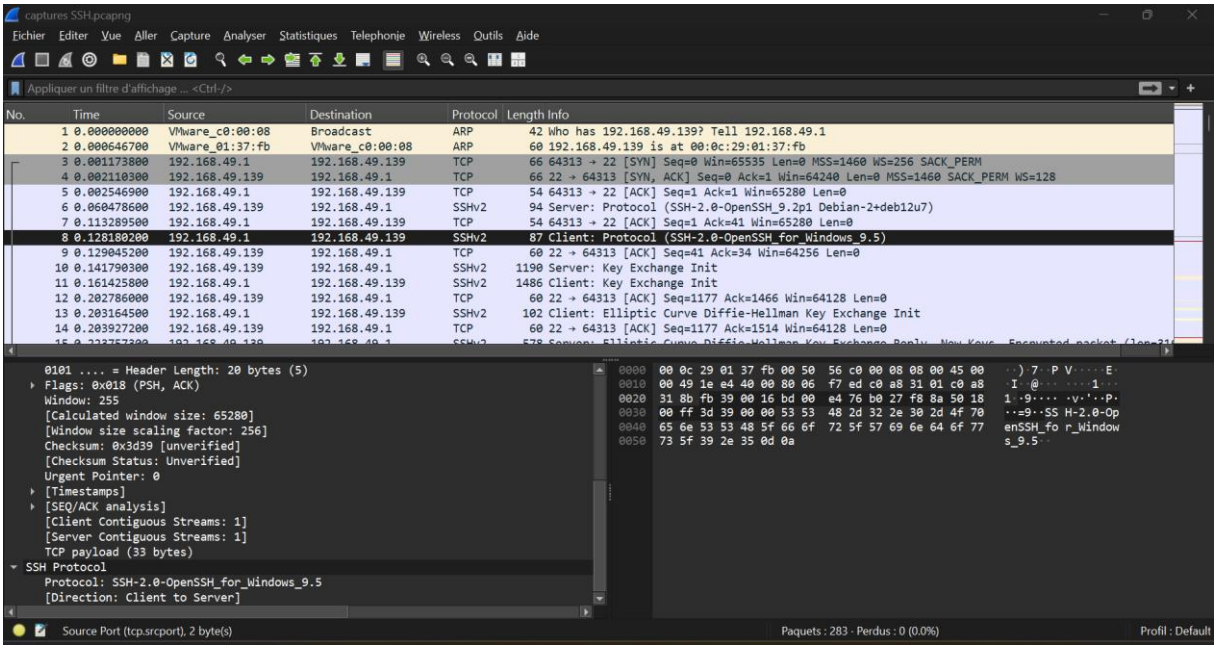
On peut ensuite observer le three-way handshake lors de la connexion SSH :



En effet, le protocole SSH utilise bien le TCP. Le port utilisé par le protocole SSH est 22

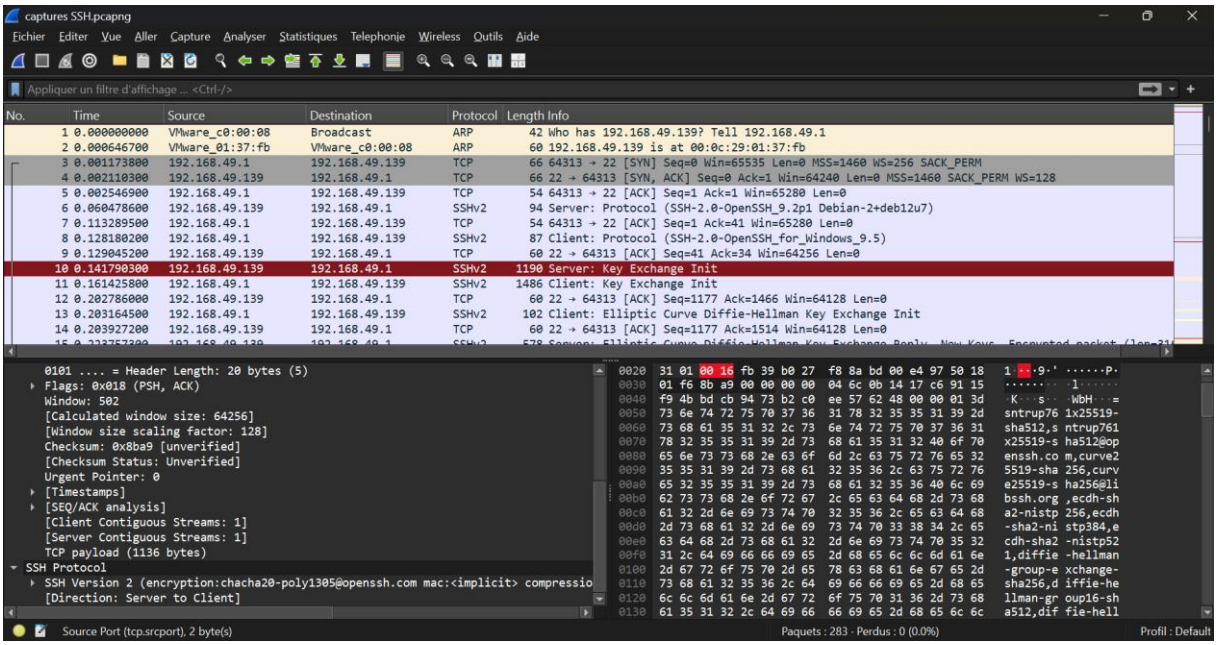
Regardons ensuite les 2 premier paquets SSH :





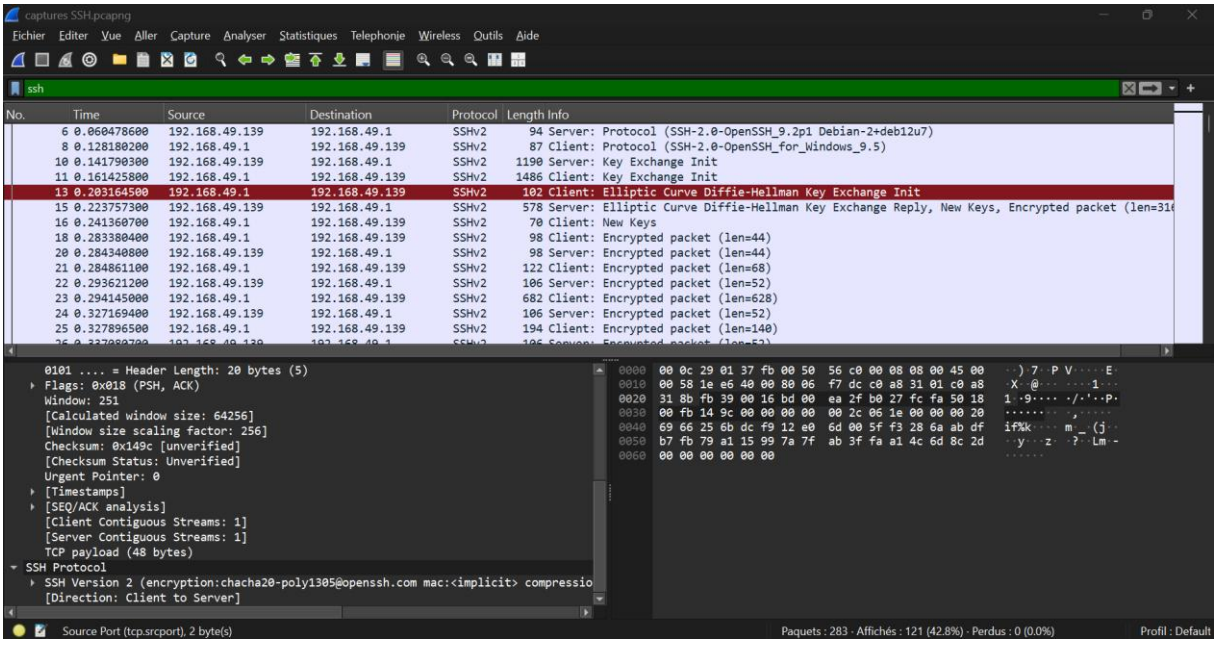
On voit les deux machines s'annoncer en clair. Les deux paquets annoncent la version utilisée par le client et le serveur : SSH 2.0 OpenSSH for Windows 9.5, SSH 2.0 OpenSSH for Debian

On peut ensuite observer les paquets « Key exchange unit » client / serveur :



Ces paquets correspondent à l'échange des clés de sécurité et indiquent l'algorithme de sécurité utilisé par les deux machines.

L'échange des clés Diffie-Hellman se produit ensuite :

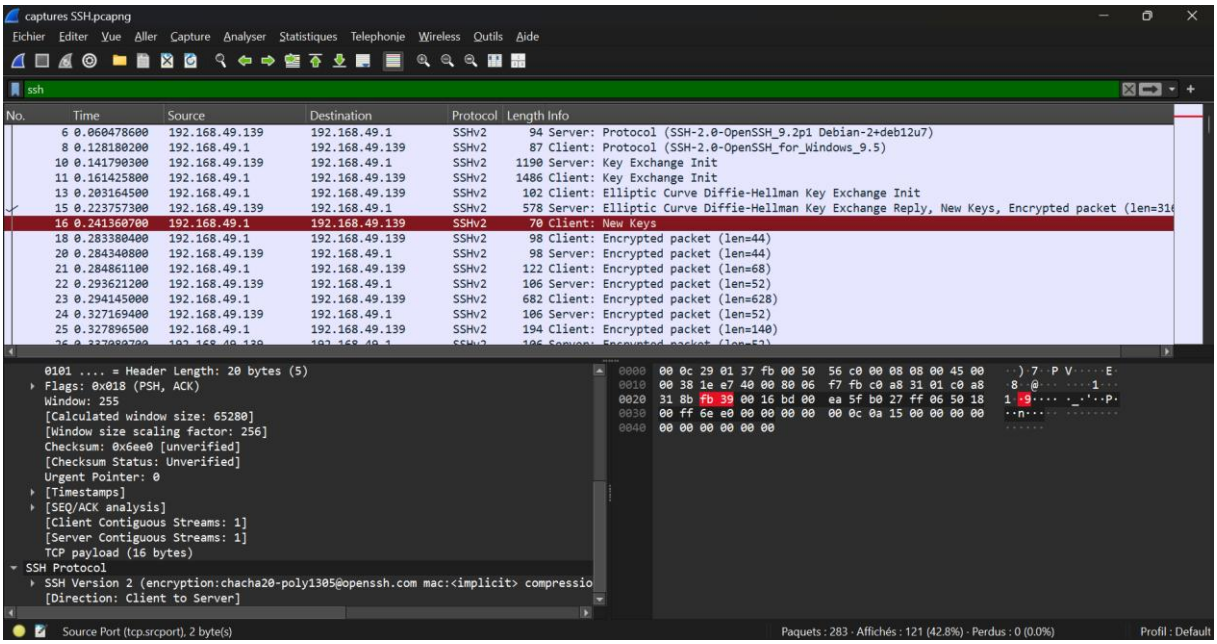


Nous observons deux paquets :

- Diffie-Hellman Key Exchange Init
- Diffie-Hellman Key Exchange Reply

C'est à ce moment que les machines vont créer ensemble un secret partagé sans jamais le faire transiter sur le réseau.

A partir du paquet suivant « New Keys », la connexion devient totalement chiffrée :



On peut observer tous les paquets suivants sont encryptés « Encrypted Packet »

Ces paquets encryptes correspondent aux commandes qui ont été transmises à la machine virtuelle en SSH : ls, cd, pwd, whoami

Suite à la commande « exit » entrée en SSH, on peut observer la fin de la communication TCP avec le drapeau « RST ACK » sur le port 22 (SSH) :