Wireshark

Wireshark c’est quoi ?

Wireshark est un analyseur de paquets libre et gratuit.
C’est un outil de capture et d’analyse de paquets.

Quelle est la différence entre une trame et un paquet ?

une trame permet de véhiculer le paquet sur le réseau .

Qu’est-ce que le format pcap/pcapng ?

Les fichiers PCAP appartiennent principalement à Wireshark de The Wireshark team. Les fichiers PCAP stockent les données réseau recueillies par le programme de capture du trafic réseau tcpdump. Ils sont utilisés pour analyser les réseaux, surveiller l'utilisation de la bande passante, identifier les serveurs DHCP malveillants, détecter les logiciels malveillants, les cyberattaques, la résolution DNS, la réponse aux incidents et le dépannage des problèmes de performance généraux. Un format successeur extensible de PCAP a été réalisé - PCAPNG.

- **Utilisation** : les fichiers PCAP sont créés par l'API, et lus à l'aide des outils d'analyse pris en charge. Le fichier contient un enregistrement des données du réseau qui sont visualisables par ces outils. Le plus populaire d'entre eux est Wireshark, mais il en existe de nombreux disponibles, à la fois basés sur le cloud et sur le client, et gratuits et open-source.
- **Compatibilité** : pcap est implémenté sur un système d'exploitation par le biais de différentes bibliothèques : sur les systèmes de type Unix, il se trouve dans la bibliothèque libcap, tandis que sur Windows, il s'agit de Npcap. Il est important de noter que pour les versions de Windows antérieures à 7, vous devez utiliser la bibliothèque WinPcap dépréciée.
- **Extensions connexes** : pcap utilise deux autres extensions autres que PCAP : CAP et DMP.

Les fichiers PCAPNG appartiennent principalement à Wireshark de The Wireshark team. PCAPNG est un format utilisé pour enregistrer les traces de paquets réseau capturés dans un fichier. Il a été conçu pour être un successeur extensible du format original PCAP utilisé par tcpdump et d'autres logiciels utilisant la bibliothèque libpcap. Actuellement, seul Wireshark peut lire et écrire des fichiers PCAPNG, alors que libpcap (et donc les logiciels qui l'utilisent) ne peuvent lire que certains de ces fichiers.

Buts : PCAPNG a été créé avec les objectifs suivants en tête :

- Extensibilité - La possibilité d'ajouter de nouvelles capacités standard au format au fil du temps, et la possibilité pour des tiers de pouvoir enrichir les informations intégrées dans le fichier avec des extensions propriétaires;
- Portabilité - Le fichier doit contenir toutes les informations nécessaires pour lire les données indépendamment du réseau, du matériel et du système d'exploitation de la machine qui a effectué la capture;
- Fusionner/Ajouter des données : La possibilité d'ajouter des données à la fin d'un fichier donné, alors que le fichier reste lisible.

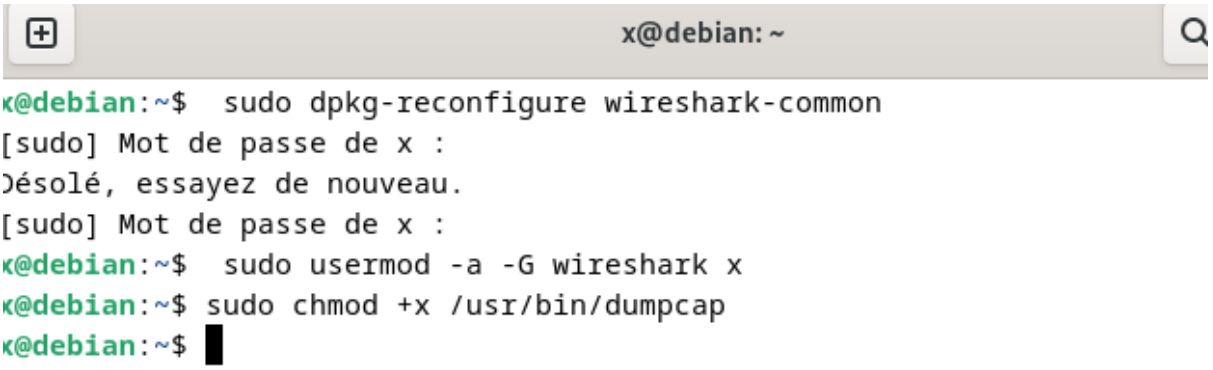
Structure : Un fichier de capture est organisé en blocs, qui sont annexés les uns aux autres pour former le fichier. Tous les blocs partagent un format commun, comportant les champs suivants dans l'ordre suivant : Type de bloc - une valeur unique non signée qui identifie le bloc ; Longueur totale du

bloc - une valeur non signée donnant la taille totale du bloc, en octets ; Corps du bloc - le contenu du bloc ; Longueur totale du bloc (encore) - ce champ est écrit deux fois pour permettre la navigation dans le fichier en arrière.

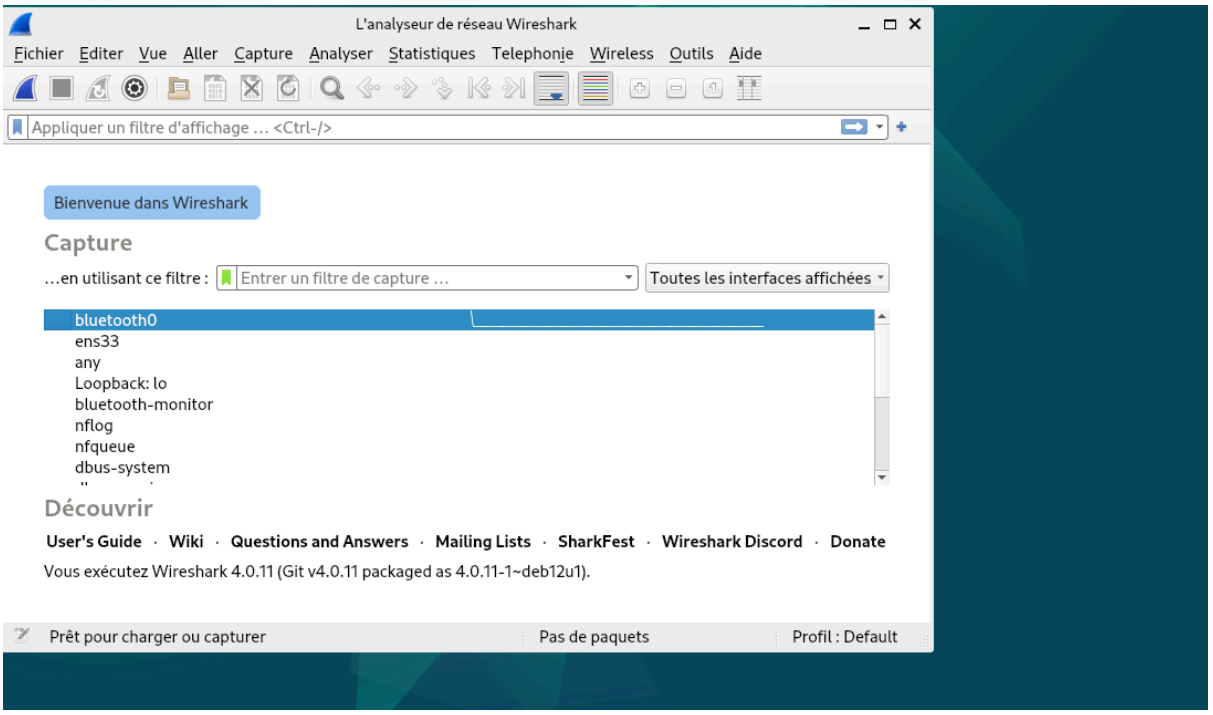
installation de wireshark sous linux

```
3  apt update
4  apt install wireshark
```

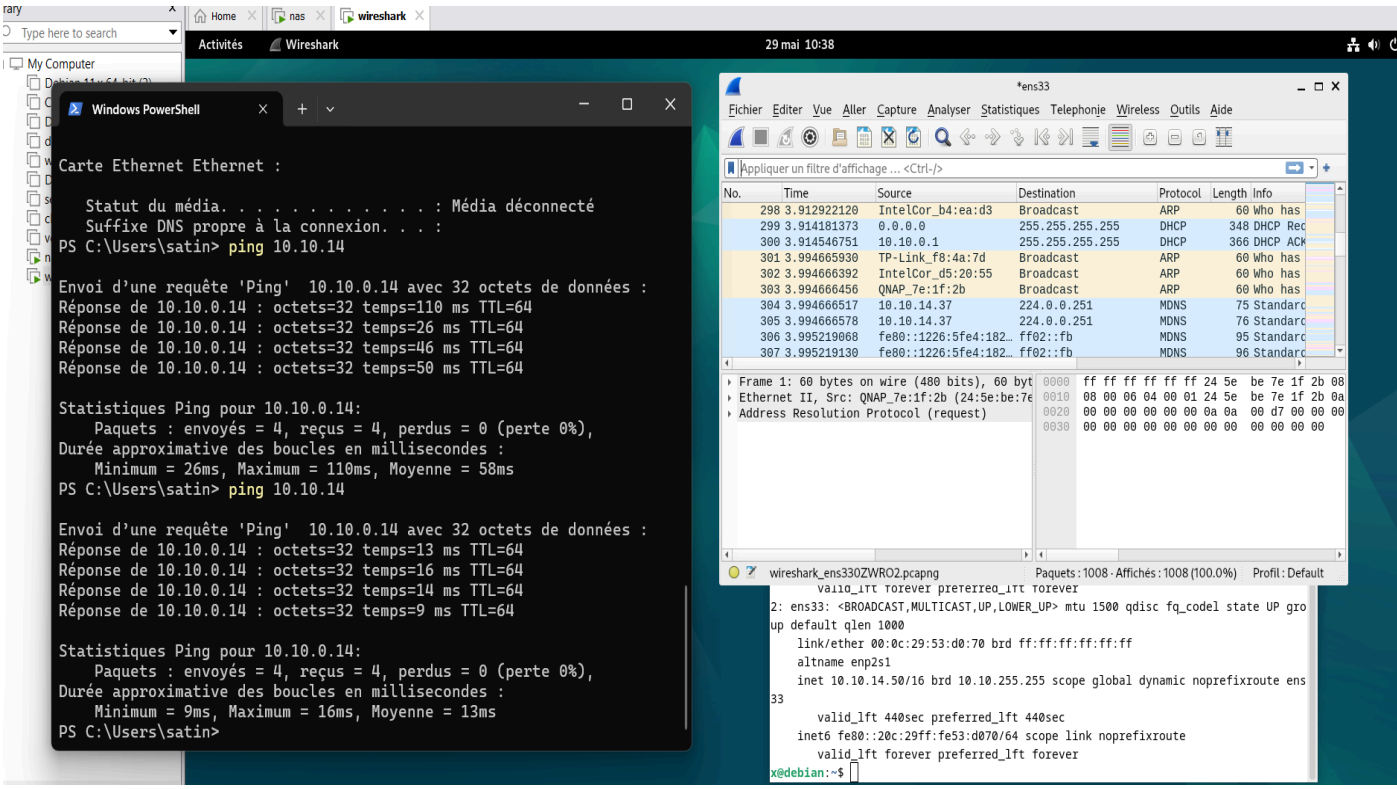
je tape cette commande pour donner les droits à mon utilisateur .



j'ouvre wireshark



je lance une capture , je ping la machine



Paquet ARP couche 2 liaison osi
paquet udp couche 4 transport
paquet tcp couche 4 transport

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

paquet tcp
adresse ip source/destination

819	45.298387	104.123.50.160	10.10.17.45	TCP	54 80 → 50720 [ACK] Seq=1 Ack=112 Win=64256 Len=0
820	45.300314	104.123.50.160	10.10.17.45	HTTP	241 HTTP/1.1 200 OK (text/plain)
821	45.300314	104.123.50.160	10.10.17.45	TCP	54 80 → 50720 [FIN, ACK] Seq=188 Ack=112 Win=64256 Len=0

address mac source/destination

```
Destination: AzureWaveTec_e3:3c:69 (2c:3b:70:e3:3c:69)
    Address: AzureWaveTec_e3:3c:69 (2c:3b:70:e3:3c:69)
    .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
Source: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
    Address: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
    .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
```

paquet udp
adresse ip source/destination

```
    692 39.657156     142.251.37.206        10.10.17.45           UDP         79 443 → 61932 Len=37
```

address mac source/destination

```
> Frame 692: 79 bytes on wire (632 bits), 79 bytes captured (632 bits) on interface \Device\NPF_{00A404
v Ethernet II, Src: Intel_3a:2e:49 (68:05:ca:3a:2e:49), Dst: AzureWaveTec_e3:3c:69 (2c:3b:70:e3:3c:69)
  v Destination: AzureWaveTec_e3:3c:69 (2c:3b:70:e3:3c:69)
      Address: AzureWaveTec_e3:3c:69 (2c:3b:70:e3:3c:69)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  v Source: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
      Address: Intel_3a:2e:49 (68:05:ca:3a:2e:49)
      .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
      .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
    Type: IPv4 (0x0800)
```

paquet arp
adresse ip source/destination

```
   8 0.883484     QNAP_7e:1f:2b     Broadcast           ARP         60 Who has 10.10.3.201? Tell 10.10.0.200
```

adresse mac source/destination

```
v Destination: Broadcast (ff:ff:ff:ff:ff:ff)
    Address: Broadcast (ff:ff:ff:ff:ff:ff)
    .... ..1. .... .... .... .... = LG bit: Locally administered address (this is NOT the factory defa
    .... ...1 .... .... .... .... = IG bit: Group address (multicast/broadcast)
v Source: QNAP_7e:1f:2b (24:5e:be:7e:1f:2b)
    Address: QNAP_7e:1f:2b (24:5e:be:7e:1f:2b)
    .... ..0. .... .... .... .... = LG bit: Globally unique address (factory default)
    .... ...0 .... .... .... .... = IG bit: Individual address (unicast)
  Type: ARP (0x0806)
  Padding: 000000000000000000000000000000000000
```

le protocole quic

QUIC est un protocole expérimental, créé par le moteur de recherche Google et présenté au public en 2013. Le nom signifie « **Q**uick **U**DP **I**nternet **C**onnections » (connexions Internet UDP rapides), car il permet l'envoi rapide de paquets simples via le protocole UDP (User Datagram Protocol) **sans connexion**. La raison du développement de QUIC était le désir de fournir une alternative aux solutions de sécurité TCP, HTTP/2 et TLS/SSL en développant la même protection mais avec **un délai de connexion et de transport réduit, et en permettant des connexions de multiplexage**.

Google a conçu QUIC afin que le protocole lui-même **contrôle la connexion**. Lors de la première prise de contact entre l'expéditeur et le destinataire, ils échangent les certificats et les clés nécessaires pour chiffrer les datagrammes envoyés. Les communications ultérieures éliminent cet échange, ce qui minimise la latence. Le protocole de chiffrement est la **version 1.3**

actuelle de TLS, optimisée en vitesse (standardisée en mars 2017), préférée à la solution de chiffrement interne. Concernant le **multiplexage**, le protocole QUIC suit le protocole SPDY développé par Google, qui fournit le modèle pour HTTP/2 : une connexion client-serveur unique pouvant être utilisée pour transmettre plusieurs flux de données, ce qui réduit considérablement le temps de chargement.

le protocole mdns
Multicast DNS (mDNS) est un service conçu pour aider à la résolution de noms dans les petits réseaux. Toutefois, le mDNS utilise une méthode différente de celle du DNS traditionnel : au lieu de solliciter un serveur de noms, **tous les participants au réseau sont directement adressés**. Le client correspondant envoie un multicast dans le réseau et demande à quel participant du réseau le nom d'hôte correspond. Le multicast est une forme de communication spécifique dans laquelle un seul message est adressé à un groupe de destinataires. Le groupe peut par exemple être constitué de l'ensemble du réseau ou d'un sous-réseau.

ARP (Address Resolution Protocol)

Spécificités

1. **Fonction** :
    - ARP est utilisé pour mapper une adresse IP à une adresse MAC (Media Access Control) sur un réseau local (LAN).
2. **Structure** :
    - **Hardware Type (2 bytes)** : Spécifie le type de réseau matériel (par exemple, Ethernet).
    - **Protocol Type (2 bytes)** : Spécifie le type de protocole de réseau (par exemple, IPv4).
    - **Hardware Size (1 byte)** : Longueur de l'adresse matérielle en octets.
    - **Protocol Size (1 byte)** : Longueur de l'adresse de protocole en octets.
    - **Opcode (2 bytes)** : Indique le type de message ARP (request ou reply).
    - **Sender MAC Address (6 bytes)** : Adresse MAC de l'expéditeur.
    - **Sender IP Address (4 bytes)** : Adresse IP de l'expéditeur.
    - **Target MAC Address (6 bytes)** : Adresse MAC du destinataire.
    - **Target IP Address (4 bytes)** : Adresse IP du destinataire.
3. **Utilisation** :
    - ARP est crucial pour la communication au sein d'un réseau local car il permet aux hôtes de découvrir l'adresse MAC correspondant à une adresse IP, permettant ainsi la transmission de paquets de données au bon destinataire sur le réseau local.

UDP (User Datagram Protocol)

Spécificités

1. **Fonction** :
    - UDP est un protocole de transport orienté datagramme, qui permet l'envoi de messages courts appelés datagrammes entre les hôtes d'un réseau IP sans nécessiter d'établissement de connexion préalable.
2. **Structure** :
    - **Source Port (2 bytes)** : Port source de l'application émettrice.
    - **Destination Port (2 bytes)** : Port destination de l'application réceptrice.
    - **Length (2 bytes)** : Longueur totale du datagramme UDP, incluant l'en-tête et les données.
    - **Checksum (2 bytes)** : Utilisé pour détecter les erreurs dans le datagramme.
3. **Utilisation** :
    - UDP est utilisé pour les applications qui nécessitent une transmission rapide et tolèrent une certaine perte de paquets, comme le streaming audio/vidéo, les jeux en ligne et les protocoles DNS.

TCP (Transmission Control Protocol)

Spécificités

1. **Fonction** :
    - TCP est un protocole de transport orienté connexion, qui assure une transmission fiable et ordonnée des paquets de données entre les hôtes sur un réseau IP.
2. **Structure** :
    - **Source Port (2 bytes)** : Port source de l'application émettrice.
    - **Destination Port (2 bytes)** : Port destination de l'application réceptrice.
    - **Sequence Number (4 bytes)** : Numéro de séquence du premier octet des données dans ce segment.
    - **Acknowledgment Number (4 bytes)** : Numéro de séquence du prochain octet que l'émetteur s'attend à recevoir.
    - **Data Offset (4 bits)** : Indique la taille de l'en-tête TCP.
    - **Reserved (3 bits)** : Réservé pour un usage futur.
    - **Flags (9 bits)** : Contrôle divers aspects de la connexion (SYN, ACK, FIN, RST, PSH, URG, etc.).
    - **Window Size (2 bytes)** : Taille de la fenêtre de réception.
    - **Checksum (2 bytes)** : Utilisé pour détecter les erreurs dans le segment.
    - **Urgent Pointer (2 bytes)** : Pointeur vers les données urgentes (si URG est fixé).
    - **Options (variable)** : Options TCP (par exemple, MSS, Window Scaling).
    - **Data (variable)** : Données transportées par le segment.
3. **Utilisation** :
    - TCP est utilisé pour les applications nécessitant une transmission fiable et sans erreur des données, comme le transfert de fichiers (FTP), les courriels (SMTP/IMAP) et la navigation web (HTTP/HTTPS).

Comparaison des Spécificités

- **Fiabilité** :
    - **ARP** : Non applicable (protocole de résolution d'adresse).
    - **UDP** : Non fiable, pas de garantie de livraison ou d'ordre.
    - **TCP** : Fiable, garantit la livraison et l'ordre des paquets.
- **Connexion** :
    - **ARP** : Sans connexion.
    - **UDP** : Sans connexion.
    - **TCP** : Orienté connexion.
- **Utilisation typique** :
    - **ARP** : Résolution d'adresse sur les réseaux locaux.
    - **UDP** : Applications en temps réel et tolérantes aux pertes (streaming, DNS).
    - **TCP** : Applications nécessitant fiabilité et intégrité (HTTP, FTP).

En résumé, chaque protocole a des spécificités et des utilisations distinctes adaptées à différents besoins de communication réseau. ARP est utilisé pour la résolution d'adresse sur un LAN, UDP pour des transmissions rapides et tolérantes aux erreurs, et TCP pour des transmissions fiables et ordonnées.

En écoutant des échanges FTP sans TLS, que remarquez-vous dans les

paquets ? Est-il possible de récupérer des données sensibles de connexion ?

dans les échanges ftp sans tls , on voit les identifiants et mots de passe en claire donc oui on peut récupérer des données sensibles.