

NETWORK SNIFFING

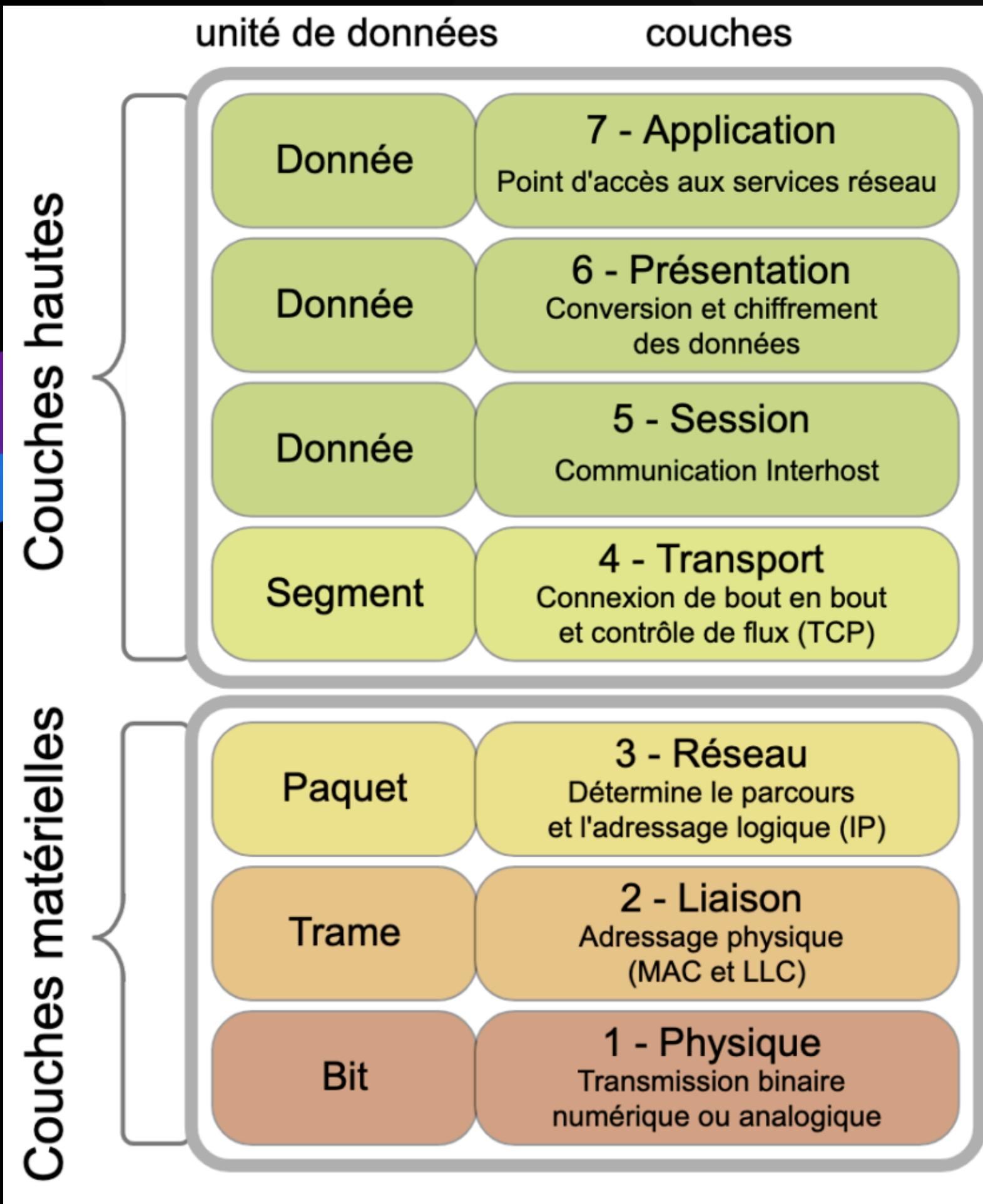
Crystal, Sofian, Cyril

Wireshark c'est quoi ?

Wireshark est un logiciel d'analyse de protocole réseau, il capture et inspecte les données circulant dans un réseau informatique en temps réel :

- Il capture les paquets
- Il fait une analyse détailler des différents protocoles et des paquets détectés
- Il filtre selon les besoins de son utilisateur
- Il a une interface graphique intuitive

Modèle OSI



La couche application fournit l'interface pour un utilisateur final qui utilise la machine en question, il charge une application comme les mais par exemple

La couche présentation prépare les données qui seront affiché à l'utilisateur. Elle va, en autre, gérer la compression et la décompression, d'encodage et de décodage

La couche session permet d'ouvrir et de fermer la communication entre deux machines, assure que les données soient transferés, retransmettre si elles sont incomplètes

La couche session permet d'ouvrir et de fermer la communication entre deux machines, assure que les données soient transferés, retransmettre si elles sont incomplètes en utilisant différents protocoles

La couche réseau (pas utile sur un même réseau) décompose les données de l'expéditeur et les réassemble pour le destinataire par la création de paquets

La couche liaison assure le transfère de données de noeud à noeud directement connecté et corrige les erreurs de la couche physique. Elle comprend deux sous couches, contrôle d'accès (MAC) et de liaison logique (LLC)

Cette couche gère les caractéristiques électrique, logique et physique d'un système



Différents protocoles réseaux

Protocol TCP

Le protocole TCP (TRANSMISSION CONTROL PROTOCOL) est un des principaux protocoles de la couche transport du modèle OSI.

Il est utilisé pour assurer une transmission fiable et ordonné en utilisant un système de numérotation de séquence afin de transmettre dans le bon ordre, il gère aussi le flux pour éviter une surcharge et s'adapte à la capacité de réception du récepteur.

Il fonctionne en 3 étapes :

SYN : envoi d'un segment de synchronisation pour établir la connexion

SYN-ACK : Le récepteur répond avec un autre segment de synchronisation et accusé de réception

ACK : Confirmation de l'accusé de réception
c'est alors que la connexion est établie.

Protocol ARP

Ce protocole de résolution d'adresse sert à traduire une adresse IP à une adresse MAC dans un réseau local car elle diffère en longueur et en format et change constamment, il est donc nécessaire pour la communication entre deux machines et assurer qu'un paquet soit bien envoyé au bon endroit.

Protocol UDP

Ce protocole (USER DATAGRAM PROTOCOL) est utilisé pour la communication pour les applications sensibles au temps comme les jeux vidéo, lecture de vidéos ou recherches des noms de domaines (DNS)

Il ne passe par une connexion fermé avec la destination avant le transfère de données, il a pas de vérification de réception de données ni d'ordre de réception de données ce qui fait gagné du temps lors du transféré. Il peut donc avoir des pertes de paquets lors de la transmission et est moins sécurisé ce qui peut facilité les attaques DDOS

Trames et paquet

Faisons une métaphore pour bien comprendre la différence

Un paquet serait le contenu et l'adresse d'une lettre postale, elle contient donc les données principales et les informations sur la destination finale.

La trame serait l'enveloppe qui contient la lettre (les données) ainsi que d'autres informations nécessaires à l'acheminement entre deux points intermédiaires comme un centre de tri postal. Elle contient donc les adresses MAC et des informations de contrôle pour l'acheminement à des points différents sur un réseau.

Format pcap/pcapng

PCAP

Le format pcap est relativement simple et consiste en une série d'en-tête de paquets suivis des données des paquets capturés.

Il est principalement utilisé pour capturer des paquets sur un réseau pour une analyse ultérieure.

Pris en charge par la plupart des outils d'analyse de paquets

PCAPNG

Ce format est plus avancé et plus flexible que le format pcap, il permet de capturer plus d'informations sur la capture comme les métadonnées, des statistiques et des commentaires

Il est composé de différents blocks :

[Section Header Block \(SHB \)](#)

Contient des informations globales de la capture

[Interface Description Block \(IDB \)](#)

Fournit des informations sur les interfaces réseau utilisées lors de la capture

[Enhanced Packet Block \(EBP \)](#)

Contient des données des paquets capturés avec des informations supplémentaires par rapport à pcap

[Simple Packet Block \(SPB \)](#)

Contient les données des paquets de manière plus simple

[Name Resolution Block \(NRB \)](#)

Permet de stocker des informations de résolutions de noms

Décryptage d'une trame

Adresse MAC de destination est :

SagemcomBroa_7e...

Il s'agit d'un appareil fabriqué par l'entreprise Sagemcom Broadband, une entreprise d'équipement de télécommunications y compris les modem, routeur et d'autres appareils de réseau

L'adresse MAC source est :

Intel_7c...

Il s'agit de ma machine dont le fabricant est l'entreprise Intel corporation

Le protocol utilisé pour cette trame est l'IPV4

```
Frame 116: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NP
Ethernet II, Src: Intel_7c:c2:cf (40:74:e0:7c:c2:cf), Dst: SagemcomBroa_7e:e2:94 (08:d5:9d:7e:e2:94)
  Destination: SagemcomBroa_7e:e2:94 (08:d5:9d:7e:e2:94)
  Source: Intel_7c:c2:cf (40:74:e0:7c:c2:cf)
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src: 192.168.0.17, Dst: 104.16.102.112
Transmission Control Protocol, Src Port: 59919, Dst Port: 443, Seq: 2953, Ack: 391, Len: 0
```

Ip de destination est 104.16.102.112 qui appartient à CLOUDFLARE.inc qui est un services d'optimisation de distribution de contenu de site web et de protection contre les attaques DDOS

Ip source est 192.168.0.17 qui est la mienne

D'autres exemples de trame

QUIC
QUIC

Le protocole QUIC est basé sur le protocole UDP avec sa rapidité mais est aussi combiné avec les fonctionnalité du protocol TLS qui est plus sécurisé notamment pour les service de steaming sécurisé et permet le chiffrement bout à bout

SSDP
SSDP

Ce protocole permet de découvrir les services et les appareils disponible sur le réseaux, il permet donc d'annoncer les différents services et appareils qui sera utilisé lors d'une capture de trame

Il y 3 étapes :

L'annonce : NOTIFY : qui permet aux appareils de signalé leurs présences

aux autres présent dans le réseau

Recherche de services : M-SEARCH :

Cette requête permet de découverrir les services utilisé et disponible dans le réseau

La réponse : RESPONSE :

Une fois la requête envoyé, il y a une réponse pour les services sollicités avec des information supplémentaire (URL, localisation, le serveur utilisé)

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

TLSv1.2

Le protocole TLSv1.2 est utilisé pour sécuriser la communication sur internet en chiffrant les données et en assurant l'authentification des deux parties impliquées.

Il est principalement utilisé pour les connexions sécurisées comme l'HTTPS

Partie 3

- En écoutant des échanges FTP sans TLS, que remarquez-vous dans les paquets ? Est-il possible de récupérer des données sensibles de connexion ?
- En est-il de même avec les échanges SSL ?
- Réponse:
- L'utilisation de FTP sans TLS expose les utilisateurs à des risques de sécurité importants, permettant à quiconque interceptant les paquets de récupérer des données sensibles. En revanche, l'utilisation de FTPS (FTP sécurisé avec SSL/TLS) protège ces informations en les chiffrant, rendant leur interception et exploitation beaucoup plus difficiles. Toujours préférer les protocoles sécurisés pour protéger les informations sensibles sur le réseau.

Pour capturer les paquets des différents protocoles listés (DHCP, DNS, mDNS, SSL, FTP, SMB, HTTPS, TLSv1.2) pendant une durée de 60 secondes et enregistrer les résultats au format CSV, utiliser tshark avec les options appropriées. Voici comment le faire pour chaque protocole :

DHCP : Les paquets DHCP utilisent les ports 67 et 68 (UDP).

DNS : Les paquets DNS utilisent le port 53 (UDP et TCP).

mDNS : Les paquets mDNS utilisent le port 5353 (UDP).

SSL/TLS (y compris HTTPS et TLSv1.2) : Les paquets SSL/TLS utilisent généralement le port 443 (TCP).

FTP : Les paquets FTP utilisent les ports 20 et 21 (TCP).

SMB : Les paquets SMB utilisent les ports 139 et 445 (TCP).

Expliquer les Options Utilisées avec les commandes:

- i ens33: spécifie l'interface réseau à utiliser.
- f "filter expression": utilise une expression de filtre BPF (Berkeley Packet Filter) pour capturer uniquement les paquets des ports spécifiés.
- a duration:60: capture les paquets pendant 60 secondes.
 - T fields: affiche des champs spécifiques.
 - e field_name: spécifie les champs à afficher.
- E header=y -E separator=: ajoute une ligne d'en-tête et utilise la virgule comme séparateur dans le fichier CSV.

Capturer des paquets DHCP:

```
sudo tshark -i ens33 -f "udp port 67 or udp port 68" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e udp.srcport -e udp.dstport -e bootp -E header=y -E separator=, > dhcp_capture.csv
```

Capturer des paquets DNS:

```
sudo tshark -i ens33 -f "udp port 53" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e udp.srcport -e udp.dstport -e dns -E header=y -E separator=, > dns_capture.csv
```

Capturer des paquets mDNS:

```
sudo tshark -i ens33 -f "udp port 5353" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e udp.srcport -e udp.dstport -e mdns -E header=y -E separator=, > mdns_capture.csv
```

Capturer des paquets SSL/TLS (port 443):

```
sudo tshark -i ens33 -f "tcp port 443" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e ssl -E header=y -E separator=, > ssl_capture.csv
```

Capturer des paquets FTP:

```
sudo tshark -i ens33 -f "tcp port 21" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e ftp -E header=y -E separator=, > ftp_capture.csv
```

Capturer des paquets SMB:

```
sudo tshark -i ens33 -f "tcp port 445" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e smb -E header=y -E separator=, > smb_capture.csv
```

Capturer des paquets HTTPS (port 443):

```
sudo tshark -i ens33 -f "tcp port 443" -a duration:60 -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e tls -E header=y -E separator=, > https_capture.csv
```

Capturer des paquets TLSv1.2:

```
sudo tshark -i ens33 -f "tcp port 443" -a duration:60 -Y "tls.handshake.version == 0x0303" -T fields -e frame.number -e frame.time -e ip.src -e ip.dst -e tcp.srcport -e tcp.dstport -e tls -E header=y -E separator=, > tlsv1_2_capture.csv
```

**Lire le fichier CSV:
Utilisation de cat, less ou nano exemple_capture.csv**

exemple de capture dns_capture.csv:

root@Wireshark:~# cat dns_capture.csv

frame.number,frame.time,ip.src,ip.dst,udp.srcport,udp.dstport,dnsqry.name,dns.resp.name,dns.resp.type,dns.resp.class

Explication de chaque ligne capturée:

frame.number: Le numéro de séquence du paquet dans la capture.

frame.time: L'heure à laquelle le paquet a été capturé.

ip.src: L'adresse IP source du paquet.

ip.dst: L'adresse IP de destination du paquet.

udp.srcport: Le port source UDP du paquet.

udp.dstport: Le port de destination UDP du paquet.

dnsqry.name: Le nom de la requête DNS effectuée dans le paquet.

dns.resp.name: Le nom de réponse DNS dans le paquet (s'il y en a).

dns.resp.type: Le type de réponse DNS (par exemple, A pour une résolution d'adresse IPv4).

dns.resp.class: La classe de réponse DNS (généralement IN pour Internet).

On peut ouvrir le fichier CSV avec un tableur comme LibreOffice Calc ou Microsoft Excel pour une meilleure lisibilité.

RESOURCE PAGE

