



ANNEE ACADEMIQUE 2024-
2025

RAPPORT D'ETUDE INTRODUCTION A LA CYBERSECURITE

Thème : SNIFFING AVEC WIRESHARK (COMPRENDRE
COMMENT LA COMMUNICATION FONCTIONNE ENTRE
DEUX MACHINES)

Un travail de :

- ❖ EDZOA AHANDA MAKENZY BRYAN
- ❖ CHAMEN NKONGMENECK FRANK RYAN
- ❖ GAOUM NANKEU GUERAR

Étudiant en **Bachelor 1 A**

Sous l'encadrement de : **M. Berlin DJIONANG**



RAPPORT D'ETUDE INTRODUCTION A LA CYBERSECURITE

SNIFFING AVEC WIRESHARK (COMPRENDRE COMMENT LA COMMUNICATION FONCTIONNE ENTRE DEUX MACHINES)

Un travail de :

- ❖ EDZOA AHANDA MAKENZY BRYAN
- ❖ CHAMEN NKONGMENECK FRANK RYAN
- ❖ GAOUM NANKEU GUERAR

Étudiant en **Bachelor 1 A**

Sous l'encadrement de : **M. Berlin DJIONANG**

i. RESUME

Wireshark est un logiciel libre et open-source utilisé pour l'analyse des paquets réseau, considéré comme l'un des outils les plus puissants et populaires dans le domaine de la cybersécurité, de l'administration réseau et de la formation en protocoles de communication. Il permet de capturer en temps réel les données circulant sur une interface réseau (Wi-Fi, Ethernet, etc.) et de les analyser de manière détaillée. Cette capacité fait de Wireshark un outil idéal pour effectuer ce qu'on appelle le **sniffing**, c'est-à-dire l'interception et l'observation du trafic réseau. Lorsqu'un utilisateur lance une capture avec Wireshark, tous les paquets qui transitent par l'interface sélectionnée sont collectés et affichés dans une interface graphique intuitive, où chaque paquet peut être examiné ligne par ligne : on y retrouve les adresses IP source et destination, les ports, les protocoles utilisés (TCP, UDP, HTTP, DNS, etc.), les en-têtes et parfois même le contenu brut des données. Grâce à des filtres puissants, on peut isoler un type de trafic précis ou surveiller des échanges entre des machines spécifiques. Le sniffing avec Wireshark permet ainsi de diagnostiquer des problèmes réseau, d'identifier des failles de sécurité, de surveiller l'activité réseau d'un système, ou tout simplement d'apprendre comment fonctionnent les communications sur Internet. Cependant, cette pratique doit être utilisée avec prudence : intercepter des données sur un réseau sans autorisation est illégal et contraire à l'éthique. Il est donc essentiel de n'utiliser Wireshark que dans des environnements autorisés, comme un réseau personnel, un laboratoire de test ou un réseau d'entreprise avec permission explicite.

ii. ABSTRACT

Wireshark est un logiciel libre et open-source utilisé pour l'analyse des paquets réseau, considéré comme l'un des outils les plus puissants et populaires dans le domaine de la cybersécurité, de l'administration réseau et de la formation en protocoles de communication. Il permet de capturer en temps réel les données circulant sur une interface réseau (Wi-Fi, Ethernet, etc.) et de les analyser de manière détaillée. Cette capacité fait de Wireshark un outil idéal pour effectuer ce qu'on appelle le **sniffing**, c'est-à-dire l'interception et l'observation du trafic réseau. Lorsqu'un utilisateur lance une capture avec Wireshark, tous les paquets qui transitent par l'interface sélectionnée sont collectés et affichés dans une interface graphique intuitive, où chaque paquet peut être examiné ligne par ligne : on y retrouve les adresses IP source et destination, les ports, les protocoles utilisés (TCP, UDP, HTTP, DNS, etc.), les en-têtes et parfois même le contenu brut des données. Grâce à des filtres puissants, on peut isoler un type de trafic précis ou surveiller des échanges entre des machines spécifiques. Le sniffing avec Wireshark permet ainsi de diagnostiquer des problèmes réseau, d'identifier des failles de sécurité, de surveiller l'activité réseau d'un système, ou tout simplement d'apprendre comment fonctionnent les communications sur Internet. Cependant, cette pratique doit être utilisée avec prudence : intercepter des données sur un réseau sans autorisation est illégal et contraire à l'éthique. Il est donc essentiel de n'utiliser Wireshark que dans des environnements autorisés, comme un réseau personnel, un laboratoire de test ou un réseau d'entreprise avec permission explicite.

Table des matières

i. RESUME	2
ii. ABSTRACT	3
I. LE SNIFFING D'UN RESEAU	6
1. Le Réseau Informatique	6
1.1 Définition d'un réseau informatique	6
1.2 Lien entre le réseau et le sniffing	6
1.3 Les équipements du réseau.....	7
1.3.1 Équipements passifs.....	7
1.3.2 Équipements actifs	7
2. Les communications dans un réseau informatique.....	9
2.1 Définition et importance des communications réseau	9
2.2 Étapes d'établissement de la communication réseau	9
a. L'identification des adresses	9
b. L'établissement de la connexion.....	9
c. L'échange de données	9
2.3 Protocoles de communication importants.....	10
2.4 Itinéraire des paquets dans le réseau	10
2.5 Lien avec le sniffing.....	11
3. Le Sniffing d'un Réseau	12
3.1 Définition du sniffing	12
3.2 Objectifs du sniffing.....	12
4. Wireshark : l'outil de sniffing par excellence.....	12
4.1 CONCEPT DE BASE	12
4.2 Lien entre sniffing et sécurité.....	14

INTRODUCTION

Dans un monde numérique de plus en plus interconnecté, les réseaux informatiques sont au cœur de la communication et de l'échange de données. Que ce soit pour les communications personnelles, professionnelles ou la gestion de systèmes critiques, les réseaux assurent une fluidité indispensable dans l'accès aux informations. Cependant, avec cette interconnexion mondiale, la sécurité des données devient un enjeu majeur, car les informations peuvent être vulnérables à des attaques ou à des interceptions non autorisées.

Le **sniffing** réseau, ou écoute réseau, est une méthode qui permet d'intercepter et d'analyser les paquets de données circulant sur un réseau. Bien que cette technique puisse être utilisée dans des contextes légitimes tels que le diagnostic réseau, elle est souvent associée à des pratiques malveillantes lorsqu'elle est utilisée pour espionner des informations sensibles. Dans ce cadre, des outils comme **Wireshark** se révèlent indispensables. **Wireshark** est un logiciel open-source de capture et d'analyse de paquets réseau qui permet aux administrateurs réseau, aux ingénieurs en sécurité et aux chercheurs de réaliser des analyses approfondies du trafic réseau et de détecter d'éventuelles vulnérabilités.

Ce rapport se concentre sur l'importance du sniffing dans le diagnostic et la surveillance des réseaux, en particulier en utilisant **Wireshark**. Nous allons explorer le rôle des équipements actifs et passifs, le fonctionnement des protocoles de communication réseau, ainsi que la manière dont Wireshark peut être utilisé pour capturer, analyser et interpréter les données échangées sur un réseau.

I. LE SNIFFING D'UN RESEAU

1. Le Réseau Informatique

1.1 Définition d'un réseau informatique

Un **réseau informatique** est un système d'interconnexion entre plusieurs équipements (ordinateurs, imprimantes, serveurs, etc.) permettant de partager des données et des ressources. Les réseaux peuvent être **locaux (LAN)**, lorsqu'ils couvrent une petite zone (bâtiment, entreprise), ou **étendus (WAN)** lorsqu'ils s'étendent sur de grandes distances, comme Internet.

Un réseau fonctionne grâce à l'échange de **paquets de données**, qui sont des unités d'information envoyées entre les appareils selon des règles bien précises appelées **protocoles**. Pour bien comprendre les enjeux liés au **sniffing**, il est essentiel de savoir comment ces réseaux sont structurés et quels équipements les composent.

1.2 Lien entre le réseau et le sniffing

Le **sniffing** est une technique d'écoute réseau qui consiste à **capturer les paquets de données** circulant dans un réseau. Pour qu'un sniffing soit efficace, il doit s'appuyer sur une **bonne compréhension de la structure du réseau**, des types de câblage utilisés, ainsi que du rôle des différents équipements.

En effet :

- Sur un réseau **mal structuré ou non sécurisé**, un attaquant peut facilement intercepter des données sensibles en positionnant un outil de sniffing comme **Wireshark** à un endroit stratégique.
- Les **équipements passifs**, comme les câbles, ne protègent pas les données qu'ils transportent : toute interception physique permet de lire les signaux.
- Les **équipements actifs**, s'ils ne sont pas bien configurés (ex. : switchs non sécurisés, routeurs mal protégés), peuvent également laisser passer des paquets vers des interfaces non autorisées, ce qui facilite le sniffing.

Ainsi, la connaissance des **composants d'un réseau** est un **prérequis indispensable** pour analyser ou sécuriser ce réseau, notamment en utilisant des outils de capture comme Wireshark.

1.3 Les équipements du réseau

Les composants du réseau se répartissent en deux grandes familles : les **équipements passifs** et les **équipements actifs**.

1.3.1 Équipements passifs

Ce sont les éléments **non intelligents** du réseau. Ils assurent simplement la **transmission physique** du signal, sans traitement ni modification des données.

Exemples d'équipements passifs :

- **Câbles réseau (RJ45, fibre optique)** : Assurent la liaison physique entre les équipements.
- **Panneaux de brassage** : Organisent les connexions dans une baie réseau.
- **Prises murales RJ45** : Permettent de brancher des équipements réseau via un câble.
- **Baies de brassage** : Supportent l'ensemble des équipements réseau (passifs et actifs).

Ces éléments **ne filtrent pas** les informations : toute personne connectée physiquement au réseau peut potentiellement intercepter les données qui y circulent, surtout en mode "promiscuous" (écoute totale).

1.3.2 Équipements actifs

Ces équipements sont **alimentés électriquement** et jouent un rôle **central dans la gestion du trafic** réseau. Ils peuvent acheminer, filtrer, et parfois analyser les paquets.

Exemples d'équipements actifs :

- **Switch (commutateur)** : Envoie les paquets au bon destinataire en se basant sur l'adresse MAC.
- **Routeur** : Oriente les paquets entre différents réseaux.
- **Point d'accès Wi-Fi** : Fournit un accès sans fil au réseau.
- **Serveurs** : Fournissent des services (web, mail, base de données...).
- **Pare-feux (firewalls)** : Contrôlent les accès autorisés et non autorisés.

Lien avec le sniffing :

- Un switch bien configuré peut limiter le sniffing car il envoie les paquets uniquement à la bonne destination.

- En revanche, un réseau Wi-Fi mal protégé ou un pare-feu mal paramétré peut laisser passer des paquets vers un outil comme Wireshark, facilitant l’espionnage du réseau.

Avant de pouvoir analyser ou sécuriser un réseau à l’aide de Wireshark, il est fondamental de **comprendre l’architecture réseau**, de **connaître les différents types d’équipements** et de **savoir comment les données circulent**. Cela permet non seulement de mieux interpréter les paquets capturés, mais aussi de repérer rapidement les anomalies ou les failles potentielles

2. Les communications dans un réseau informatique

2.1 Définition et importance des communications réseau

Dans un réseau informatique, la **communication** désigne l'échange de données entre deux ou plusieurs équipements (ordinateurs, imprimantes, serveurs, smartphones, etc.). Ces échanges sont rendus possibles grâce à des **protocoles de communication**, qui définissent **comment** les données doivent être structurées, envoyées, acheminées, et reçues.

Chaque fois que tu ouvres une page web, envoies un message, ou télécharges un fichier, des **paquets de données** sont envoyés d'un point à un autre. Comprendre comment cette communication s'établit et circule dans le réseau est essentiel pour détecter, analyser ou même intercepter ces échanges à l'aide d'un outil comme **Wireshark**.

2.2 Étapes d'établissement de la communication réseau

La communication entre deux machines suit plusieurs **étapes structurées** :

a. L'identification des adresses

Chaque appareil possède :

- Une **adresse MAC** (physique, unique à la carte réseau)
- Une **adresse IP** (logique, utilisée pour identifier l'appareil dans le réseau)

Les protocoles comme **ARP (Address Resolution Protocol)** permettent de faire le lien entre ces deux adresses. Wireshark permet de capturer ces requêtes ARP, souvent utilisées par les attaquants pour faire de l'**ARP Spoofing** (technique de sniffing).

b. L'établissement de la connexion

Pour une communication fiable (comme lors du chargement d'un site), une **connexion TCP** est souvent utilisée. Elle passe par trois étapes :

1. **SYN** : Le client envoie une demande de connexion.
2. **SYN-ACK** : Le serveur répond en acceptant.
3. **ACK** : Le client confirme et la communication commence.

C'est le célèbre "**handshake TCP à 3 étapes**", observable très clairement dans Wireshark.

c. L'échange de données

Une fois la connexion établie, les données circulent sous forme de **paquets**, contenant :

- Une en-tête (adresse source/destination, numéro de séquence...)
- Un contenu (payload)

Ces paquets peuvent être interceptés si le réseau est mal sécurisé, d'où l'utilité du sniffing pour vérifier les failles potentielles.

d. La terminaison de la connexion

Une fois les données échangées, la connexion est fermée via un processus **FIN / ACK**, pour libérer les ressources.

2.3 Protocoles de communication importants

Voici quelques **protocoles réseau** que l'on retrouve fréquemment dans Wireshark, et leur rôle dans la communication :

Protocole	Fonction principale	Visibilité dans Wireshark
TCP	Communication fiable (web, mail...)	Très fréquent
HTTP/HTTPS	Accès aux sites web	HTTPS est chiffré, HTTP lisible
DNS	Résolution des noms de domaine	Oui
DHCP	Attribution automatique des IP	Oui
ICMP	Messages de contrôle (ping)	Oui
ARP	Résolution d'adresse MAC	Très utile pour le sniffing

À noter : le sniffing peut révéler des **informations sensibles** si les protocoles ne sont pas chiffrés (ex : HTTP, FTP, Telnet).

2.4 Itinéraire des paquets dans le réseau

Lorsqu'un paquet est envoyé, il suit un **chemin déterminé** entre la source et la destination. Ce chemin peut inclure :

- **Switchs** : guident les paquets vers le bon destinataire à l'intérieur du réseau local.
- **Routeurs** : font transiter les paquets entre différents réseaux.
- **Passerelles** : permettent la sortie du réseau local vers Internet.
- **Pare-feux** : analysent les paquets pour bloquer ou autoriser le passage.

Dans Wireshark, chaque paquet affiche toutes ces étapes sous forme de couches (**modèle OSI**) :

- Couche 1 : Physique
- Couche 2 : Liaison (ex. Ethernet)
- Couche 3 : Réseau (ex. IP)
- Couche 4 : Transport (ex. TCP/UDP)
- Couche 7 : Application (ex. HTTP, DNS)

Ces **couches permettent d'analyser finement** le contenu d'un paquet capturé. Un sniffing bien mené avec Wireshark peut ainsi révéler **l'itinéraire complet d'un paquet**, les équipements traversés, et même parfois les données qu'il transporte.

2.5 Lien avec le sniffing

Comprendre le **fonctionnement des communications réseau** permet de :

- **Localiser les points sensibles** du réseau (où placer un sniffing pour intercepter les paquets)
- **Interpréter les données capturées avec Wireshark**
- **Détecter les anomalies** (paquets suspects, protocoles non sécurisés, connexions non sollicitées)
- **Prévoir des contre-mesures** (chiffrement, filtrage, détection d'intrusions)

Par exemple, si une communication utilise le protocole **HTTP**, un attaquant peut voir en clair les identifiants ou les mots de passe échangés. Un protocole chiffré comme **HTTPS** rend cela impossible. Le sniffing devient donc un **outil d'audit** et de **sécurité** précieux lorsqu'il est utilisé de manière légale et éthique.

3. Le Sniffing d'un Réseau

3.1 Définition du sniffing

Le **sniffing**, aussi appelé **espionnage réseau**, est une technique qui consiste à **intercepter et analyser les paquets de données** qui circulent dans un réseau informatique. L'objectif peut être l'analyse du trafic à des fins de diagnostic, de surveillance ou encore d'attaque, selon l'intention de l'utilisateur.

Un sniffeur (ou sniffing tool) fonctionne en **mettant l'interface réseau de la machine en mode "promiscuous"**, ce qui signifie qu'elle peut lire tous les paquets qui circulent sur le réseau, même ceux qui ne lui sont pas destinés.

Il existe deux types de sniffing :

- **Sniffing passif** : l'attaquant se contente de lire les paquets sans perturber le réseau. Cette technique est difficile à détecter.
- **Sniffing actif** : l'attaquant envoie des paquets pour provoquer des réactions (ex : ARP Spoofing) afin de rediriger ou capturer les communications. Ce type est plus risqué, mais plus puissant.

3.2 Objectifs du sniffing

Le sniffing peut être utilisé à des fins :

- **Légitimes** : analyse de performance, recherche de pannes, supervision de réseau (par les administrateurs).
- **Malveillantes** : vol de mots de passe, espionnage, attaque man-in-the-middle (MITM), etc.

4. Wireshark : l'outil de sniffing par excellence

Wireshark est un **analyseur de protocoles réseau** (ou **sniffer réseau**) **libre et open-source**, utilisé pour **capturer, visualiser et analyser en détail le trafic réseau**. Il fonctionne en interceptant les **paquets de données** qui circulent sur une interface réseau (Ethernet, Wi-Fi, etc.) et permet de voir **en temps réel** ou en différé tout ce qui se passe sur le réseau au niveau des communications entre machines.

Créé à l'origine sous le nom **Ethereal** en 1998, Wireshark est aujourd'hui utilisé dans le monde entier par les **administrateurs réseau**, les **experts en cybersécurité**, les **enseignants**, et les **étudiants** pour diagnostiquer des problèmes, détecter des attaques, ou simplement apprendre comment les réseaux fonctionnent.

4.1 CONCEPT DE BASE

1. Paquet (Packet)

Un **paquet** est la plus petite unité de données envoyée sur un réseau. Chaque paquet contient :

- Des **en-têtes** (informations de contrôle : adresses IP, port, protocole...)
- Une **charge utile** (le contenu réel, comme un message, une requête, un fichier, etc.)

2. Interface réseau (Network Interface)

C'est le point de connexion entre l'ordinateur et le réseau (ex : Wi-Fi, Ethernet). Wireshark écoute cette interface pour capturer le trafic.

3. Capture (Capture)

Il s'agit du processus par lequel Wireshark intercepte les paquets réseau. On peut démarrer/arrêter la capture à tout moment.

4. Filtrage (Filtering)

Wireshark permet de **filtrer les paquets** selon des critères précis (ex : ip.addr == 192.168.1.1, http, tcp.port == 80) afin de cibler ce qu'on veut analyser.

5. Protocole réseau (Network Protocol)

Un protocole est une **règle de communication** utilisée entre deux machines. Wireshark prend en charge des **centaines de protocoles** comme :

- **TCP/IP** : Transmission Control Protocol / Internet Protocol
- **http/https** : utilisé pour le web
- **DNS** : résolution de noms de domaine
- **ICMP** : Messages de contrôle (ping)
- **ARP** : Résolution d'adresse MAC
- **DHCP** : Attribution automatique des IP
- **UDP** : Communication rapide mais non fiable (vidéo, jeux...)

6. Sniffing (Reniflage réseau)

C'est le fait d'**écouter passivement** le trafic réseau pour observer et analyser les données. C'est très utile pour :

- Identifier les problèmes de performance
- Détecter des intrusions
- Comprendre comment une application fonctionne

a. Fonctionnalités principales :

- Capture de paquets en temps réel.
- Filtrage des paquets selon l'adresse IP, le protocole, le port, etc.
- Analyse des protocoles : TCP, UDP, ICMP, DNS, HTTP, etc.
- Détection des erreurs et des anomalies dans le trafic.
- Visualisation en couches (modèle OSI).

b. Utilisation de Wireshark dans le sniffing

1. **Lancement de la capture** : on choisit une interface réseau à surveiller (Wi-Fi, Ethernet).
2. **Filtrage des paquets** : on utilise des filtres (ex : ip.addr == 192.168.1.1) pour ne voir que les données intéressantes.
3. **Analyse du contenu** : chaque paquet peut être analysé en profondeur pour détecter des requêtes, réponses, erreurs, etc.

c. Avantages de Wireshark

- Interface graphique intuitive.
- Outil très complet et utilisé dans le monde professionnel.
- Idéal pour les tests de sécurité et les audits réseau.

d. Limites

- Ne capture que le trafic visible par la machine : sur un réseau switché, il faut parfois utiliser des techniques supplémentaires pour sniffer efficacement (ex : port mirroring).
- Peut générer de grandes quantités de données.

4.2 Lien entre sniffing et sécurité

Le sniffing révèle l'importance de **protéger les données qui circulent** sur un réseau. Parmi les bonnes pratiques pour se prémunir contre les attaques par sniffing, on peut citer :

- L'utilisation de **protocoles chiffrés** (HTTPS, SSH, SFTP...).
- La **segmentation du réseau** et l'isolation des zones sensibles.
- L'utilisation de **VPN** pour créer des tunnels sécurisés.
- La **surveillance active** du trafic par les administrateurs.

Ainsi, Wireshark peut être un outil d'attaque s'il est mal utilisé, mais aussi un **outil de protection** lorsqu'il est employé à bon escient pour détecter les failles et renforcer la sécurité du réseau.

4.3 Filtrage des paquets

Le **filtrage des paquets** est une technique utilisée dans les outils d'analyse réseau comme **Wireshark** pour **sélectionner uniquement les paquets qui nous intéressent** parmi tous ceux capturés sur le réseau. Cela se fait en utilisant des **filtres**, c'est-à-dire des expressions ou des mots-clés (par exemple `arp, http, ip.addr == 192.168.1.1, etc.`).

Grâce au filtrage, l'utilisateur peut :

- Isoler un type spécifique de protocole (ex : ARP, TCP, DNS...),
- Observer uniquement les communications entre deux adresses précises,
- Gagner du temps en évitant de lire des centaines de paquets inutiles.

C'est un outil **essentiel pour une analyse rapide et efficace** lors du sniffing d'un réseau.

4.4 Capture des paquets

4.5 Sniffing du protocole TCP

4.6 Définition du protocole TCP

TCP (Transmission Control Protocol) est un **protocole de communication** utilisé dans les réseaux informatiques pour permettre l'échange de données entre deux machines de manière **fiable, ordonnée et sans perte**. Il fait partie de la **suite de protocoles TCP/IP**, utilisée sur Internet.

4.7 Fonctionnement de TCP

1. Connexion orientée (Connection-oriented)

Avant de transmettre des données, TCP établit une **connexion** entre les deux machines via le mécanisme appelé **"Three-Way Handshake"** (poignée de main en trois étapes) :

- **SYN** : le client envoie une requête de connexion (SYN).
- **SYN-ACK** : le serveur répond qu'il accepte la connexion (SYN + ACK).
- **ACK** : le client confirme la réception (ACK).

✓ Une fois cette phase terminée, la communication peut commencer.

2. Transmission fiable et ordonnée

- Les données sont **divisées en segments**.
- Chaque segment est **numéroté** (Numéro de séquence).
- Si un segment est perdu ou endommagé, il est **renvoyé automatiquement**.
- Les données sont **réassemblées** dans le bon ordre à la réception.

3. Contrôle de flux

TCP ajuste le **volume de données envoyées** en fonction de la capacité du récepteur (fenêtre glissante).

4. Contrôle de congestion

TCP réduit le débit en cas de **saturation du réseau** (pour éviter les surcharges).

5. Fermeture de la connexion

Après l'échange de données, TCP **termine la connexion** avec un autre échange à 4 étapes (FIN, ACK, etc.).

Avantages de TCP

- ✓ **Fiabilité** : retransmission des paquets perdus.
- ✓ **Ordre garanti** : les segments arrivent dans le bon ordre.
- ✓ **Pas de duplication** de données.
- ✓ **Contrôle de flux** et **contrôle de congestion**.
- ✓ Convient pour des **applications critiques** : email, web (HTTP/HTTPS), transfert de fichiers (FTP), etc.

Inconvénients de TCP

- ! **Plus lent** que d'autres protocoles (à cause de la vérification, des accusés de réception).
- ! **Consommation de ressources** plus élevée (mémoire, CPU).
- ! Moins adapté aux applications temps réel (latence trop élevée).

Relation entre TCP et UDP

TCP (Transmission Control Protocol) et UDP (User Datagram Protocol) sont **deux protocoles de transport** appartenant à la **même couche du modèle TCP/IP** :

↳ la **couche transport**.

1. Ils ont le même objectif général

Les deux servent à **transporter des données** d'un point A à un point B dans un réseau (comme Internet), mais chacun le fait à sa manière.

Ils sont comme **deux méthodes différentes pour livrer un colis** :

- TCP : méthode sécurisée avec signature, suivi, et confirmation de livraison.
- UDP : méthode rapide sans suivi, sans assurance.

2. Ils fonctionnent au-dessus de la couche IP

Tous les deux utilisent **l'adresse IP** et les ports pour envoyer des données :

- L'**IP** s'occupe du routage entre machines.
- Le **protocole de transport** (TCP ou UDP) s'occupe de comment les données sont transférées une fois le chemin trouvé.

Donc, TCP et UDP dépendent tous les deux de **l'infrastructure IP**, mais agissent différemment **au-dessus de cette couche**.

3. Ils peuvent être utilisés par des applications différentes

Certaines applications réseau peuvent choisir d'utiliser **soit TCP, soit UDP** en fonction de leurs besoins :

- Le choix dépend des **exigences de l'application** : fiabilité, vitesse, latence...

Donc, dans une même session réseau, **plusieurs flux de données** peuvent coexister, certains en TCP, d'autres en UDP.

4. Ils sont tous les deux des "protocoles frères"

Ils ne sont **ni en opposition, ni concurrents**. Ce sont **deux solutions complémentaires** au sein de la même **architecture réseau (TCP/IP)**.

Tu peux les voir comme **deux outils dans une boîte à outils** : chacun est utile selon la tâche à accomplir.

En résumé :

TCP et UDP sont deux protocoles de transport dans la suite TCP/IP.

Ils ont le **même rôle général** (transporter des données), mais ils utilisent des **méthodes différentes**. Ils **travaillent tous les deux avec IP**, mais servent **des besoins différents** dans le réseau.

- **TCP** = Fiabilité avant tout.
- **UDP** = Vitesse avant tout.

Comparaison TCP vs UDP

Caractéristique	TCP	UDP
Type de connexion	Orientée connexion	Non orientée connexion
Fiabilité	✓ Oui	✗ Non
Ordre des données	✓ Garanti	✗ Non garanti
Contrôle de flux	✓ Oui	✗ Non
Rapidité	✗ Moins rapide	✓ Très rapide
Applications typiques	Web, Email, FTP	Streaming, VoIP, jeux en ligne

Dans la capture ci-dessous, nous avons appliqué un filtre **TCP** dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole **TCP** (Transmission Control Protocol). TCP est un protocole de transport qui assure une **transmission fiable et ordonnée des données** entre deux machines. Il est utilisé par de nombreux services comme le web (HTTP/HTTPS), les e-mails ou encore le transfert de fichiers.

Lors d'une communication TCP, on peut observer trois grandes phases dans Wireshark : l'établissement de la connexion (appelé **handshake en trois étapes**), l'échange de données, puis la fermeture de la connexion. Grâce au filtrage **TCP**, nous pouvons facilement analyser ces phases, suivre les échanges entre deux machines, détecter des erreurs de transmission ou repérer des connexions suspectes.

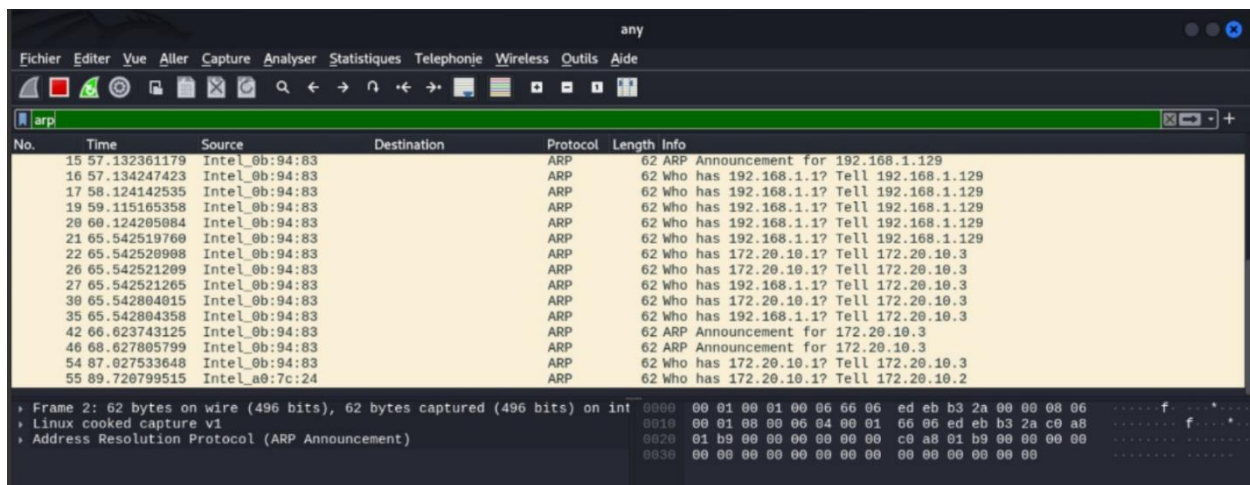
Tout comme pour ARP, le filtrage des paquets TCP permet de **gagner du temps**, de **se concentrer sur un type de trafic précis**, et d'**identifier les problèmes plus facilement** dans le cadre d'une analyse réseau.

D'autres exemple du filtrage des paquets

Protocole ARP

Sur la capture ci-dessous, nous avons appliqué un filtre ARP dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole ARP (Address Resolution Protocol). Ce protocole joue un rôle essentiel dans les communications réseau : il permet à une machine de découvrir l'adresse MAC associée à une adresse IP sur le réseau local. Lorsqu'un ordinateur souhaite envoyer des données à une adresse IP précise, il utilise ARP pour demander : « Qui a cette adresse IP ? » Le dispositif concerné répond avec son adresse MAC, permettant ainsi la communication.

Le filtrage des paquets est très important pendant une session de sniffing, car il permet de se concentrer uniquement sur un type de trafic. Cela simplifie l'analyse, réduit le bruit inutile et permet de détecter plus rapidement les anomalies ou problèmes réseau. Dans notre exemple, le filtrage ARP nous aide à observer uniquement les requêtes d'identification d'adresses, ce qui est Utile pour diagnostiquer des problèmes de communication locale ou de configuration d'adresses IP.



The image shows a Wireshark packet capture window with the filter 'arp' applied. The packet list shows 15 ARP packets. The packet details pane shows the structure of an ARP Announcement packet.

No.	Time	Source	Destination	Protocol	Length	Info
15	57.132361179	Intel_0b:94:83		ARP	62	ARP Announcement for 192.168.1.129
16	57.134247423	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 192.168.1.129
17	58.124142535	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 192.168.1.129
19	59.115165358	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 192.168.1.129
20	60.124205084	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 192.168.1.129
21	65.542519760	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 192.168.1.129
22	65.542520908	Intel_0b:94:83		ARP	62	Who has 172.20.10.1? Tell 172.20.10.3
26	65.542521209	Intel_0b:94:83		ARP	62	Who has 172.20.10.1? Tell 172.20.10.3
27	65.542521265	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 172.20.10.3
30	65.542804015	Intel_0b:94:83		ARP	62	Who has 172.20.10.1? Tell 172.20.10.3
35	65.542804358	Intel_0b:94:83		ARP	62	Who has 192.168.1.1? Tell 172.20.10.3
42	66.623743125	Intel_0b:94:83		ARP	62	ARP Announcement for 172.20.10.3
46	68.627805799	Intel_0b:94:83		ARP	62	ARP Announcement for 172.20.10.3
54	87.027533648	Intel_0b:94:83		ARP	62	Who has 172.20.10.1? Tell 172.20.10.3
55	89.720799515	Intel_a0:7c:24		ARP	62	Who has 172.20.10.1? Tell 172.20.10.2

Frame 2: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface
Linux cooked capture v1
Address Resolution Protocol (ARP Announcement)

0000 00 01 00 01 00 06 06 06 ed eb b3 2a 00 00 08 06
0010 00 01 00 00 06 04 00 01 06 06 ed eb b3 2a c0 a8
0020 01 b9 00 00 00 00 00 00 c0 a8 01 b9 00 00 00 00
0030 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Figure 1: ARP

Protocole ICMP

Dans la capture ci-dessous, nous avons appliqué un filtre **ICMP** dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole **ICMP** (Internet Control Message Protocol). ICMP est un protocole utilisé pour **envoyer des messages de diagnostic et de contrôle** entre les appareils d'un réseau. Le cas le plus courant est la commande **ping**, qui utilise ICMP pour tester si une machine est joignable.

Lorsque l'on envoie un ping, deux types de paquets ICMP sont généralement observés :

- **Echo request** : une machine envoie un paquet pour tester la connexion.
- **Echo reply** : la machine cible répond, confirmant sa disponibilité.

Grâce au filtrage **ICMP** nous pouvons observer uniquement ces messages de test dans le trafic réseau. Cela permet de vérifier si une machine répond bien aux requêtes, de **détecter une panne de communication**, ou encore de **repérer une attaque par ping flood** (envoi massif de requêtes ICMP).

Le filtrage ICMP est donc très utile pour diagnostiquer rapidement **l'état des connexions réseau** et la disponibilité des machines.

Protocole http

Dans la capture ci-dessous, nous avons appliqué un filtre `http` dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole HTTP (HyperText Transfer Protocol). HTTP est un protocole de communication utilisé pour **échanger des données entre un client (généralement un navigateur web) et un serveur web**. Il est à la base de la navigation sur Internet.

Lorsqu'un utilisateur accède à un site web, plusieurs requêtes HTTP peuvent être envoyées pour récupérer :

- Le **contenu HTML** de la page,
- Des **images, scripts ou feuilles de style**,
- D'autres ressources nécessaires à l'affichage du site.

En analysant les paquets HTTP avec Wireshark, on peut voir les **requêtes (GET, POST)** envoyées par le navigateur, ainsi que les **réponses du serveur** (avec les codes de statut comme 200 OK, 404 Not Found, etc.).

Le filtrage `http` permet donc d'isoler ces échanges web et d'analyser le comportement d'un site ou d'un utilisateur. Cela peut servir à:

- **Diagnostiquer un problème d'accès à un site,**
- **Étudier la structure d'un site web,**
- **Repérer des transmissions de données sensibles non sécurisées.**

Protocole DNS

Dans la capture ci-dessous, nous avons appliqué un filtre **DNS** dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole DNS (Domain Name System). Le protocole DNS sert à **traduire les noms de domaine (comme www.google.com) en adresses IP** (comme 142.250.185.68), afin que les ordinateurs puissent les utiliser pour communiquer.

Lorsqu'un utilisateur entre une adresse web dans son navigateur, une **requête DNS est envoyée** à un serveur DNS pour demander l'adresse IP associée à ce nom. Le serveur **DNS** répond ensuite avec l'adresse correspondante.

Dans Wireshark, les paquets DNS apparaissent généralement sous deux formes :

- **DNS Query (requête)** : le client demande l'adresse IP d'un nom de domaine.
- **DNS Response (réponse)** : le serveur retourne l'adresse IP correspondante.

Le filtrage dns permet donc de se concentrer uniquement sur ces échanges de résolution de noms. Cela est utile pour :

- **Diagnostiquer des problèmes d'accès à Internet,**
- **Vérifier si un nom de domaine est bien résolu,**
- **Détecter des requêtes DNS suspectes ou non autorisées.**

Protocole DHCP

Dans la capture ci-dessous, nous avons appliqué un filtre **DHCP** dans Wireshark. Ce filtre permet d'afficher uniquement les paquets utilisant le protocole DHCP (Dynamic Host Configuration Protocol). Ce protocole est utilisé pour **attribuer automatiquement des adresses IP et d'autres paramètres réseau** (comme la passerelle, le serveur DNS, etc.) aux machines qui se connectent au réseau.

Lorsqu'un appareil (comme un ordinateur ou un smartphone) se connecte à un réseau, il utilise DHCP pour obtenir automatiquement une configuration. Cette communication suit généralement **quatre étapes**, appelées **DORA**:

1. **Discover** : le client cherche un serveur DHCP.
2. **Offer** : le serveur propose une adresse IP.
3. **Request** : le client demande officiellement cette adresse.
4. **Acknowledge** : le serveur confirme l'attribution.

Grâce au filtrage **DHCP**, nous pouvons observer ces étapes et analyser :

- Si un appareil obtient bien une adresse IP,
- Si le serveur DHCP fonctionne correctement,
- S'il y a des **conflits ou problèmes d'attribution d'adresse**.

Le filtrage DHCP est donc très utile pour **diagnostiquer les problèmes de connectivité réseau**, surtout lorsqu'un appareil n'arrive pas à se connecter à Internet ou reste sans adresse IP.