Wireshark

Description

Wireshark est un outil open source de capture et d'analyse de paquets réseau. Cet outil est principalement utilisé pour réaliser des captures de trafic réseau sur les interfaces d'une machine. Ces captures peuvent ensuite être enregistrées dans des fichiers PCAP (extension .pcap) pour analyse.

Installation

- Sous Linux (Debian/Ubuntu) :

```
sudo apt-get install wireshark
```

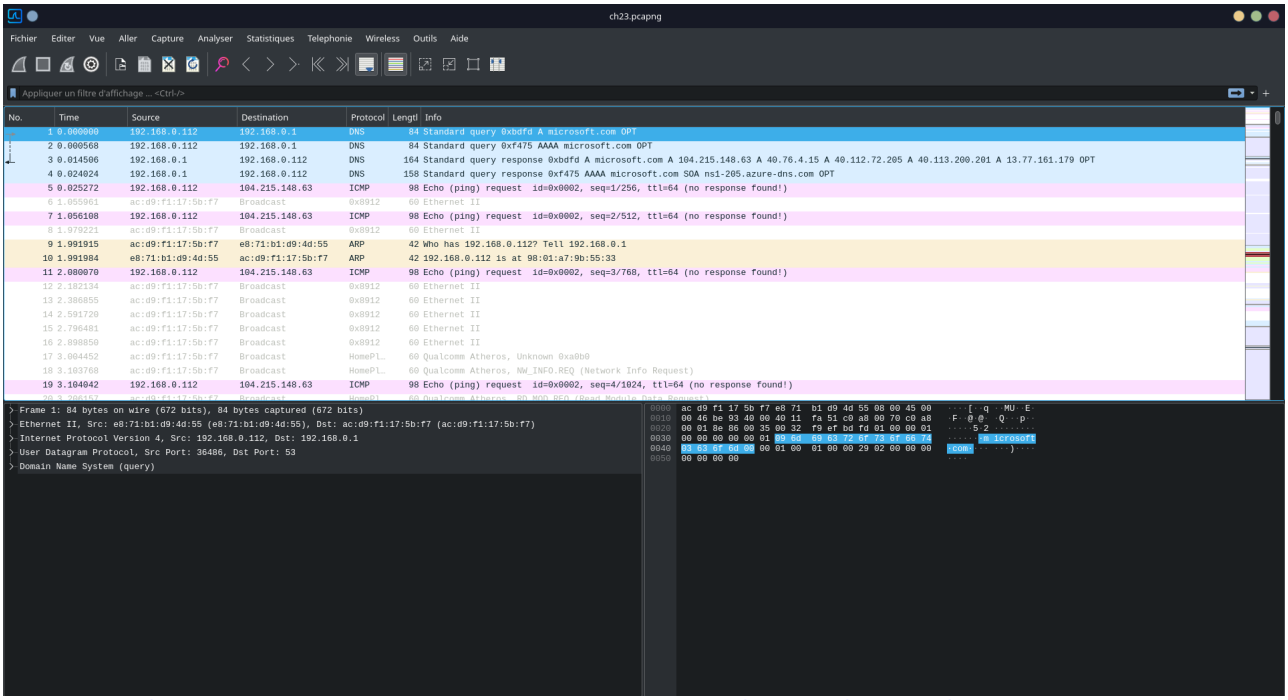
- Sous Windows : vous pouvez télécharger l'installateur depuis le site officiel (https://www.wireshark.org/download.html) et suivre les instructions d'installation.

Cas d'utilisation

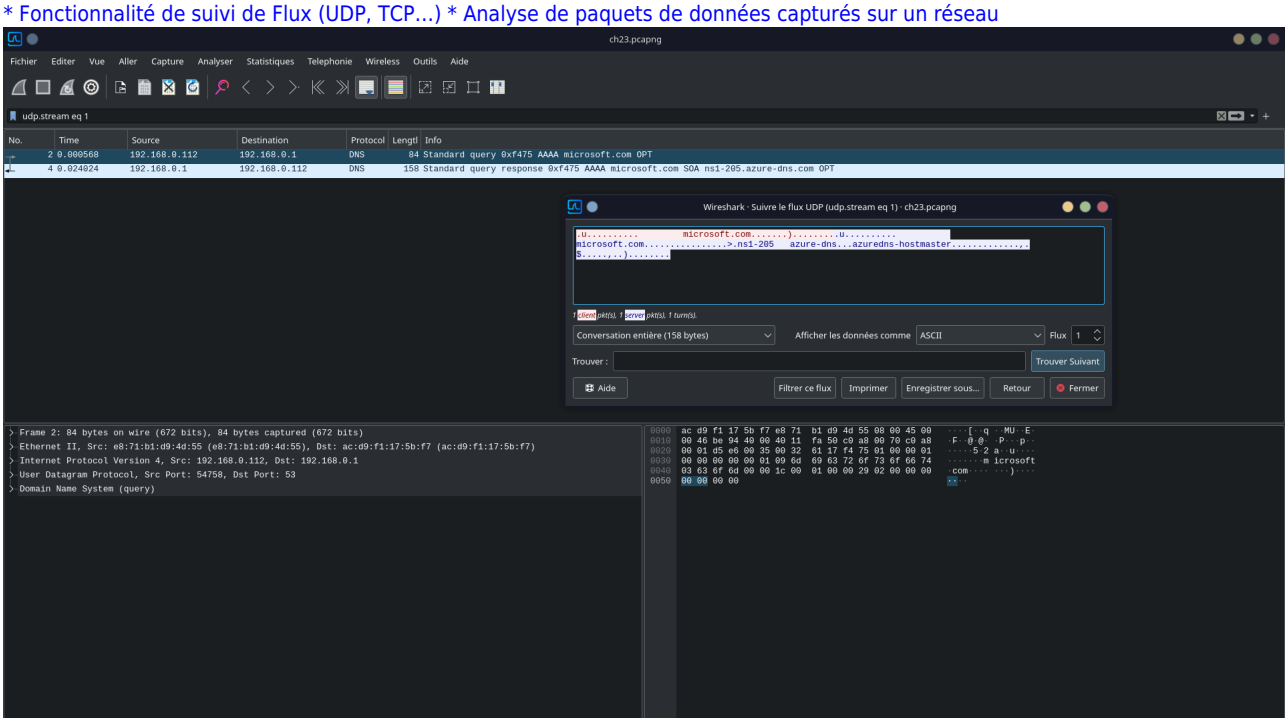
- **Dépannage réseau** : pour identifier et résoudre les problèmes de réseau tels que les pertes de paquets, les retards et les erreurs de configuration ;
- **Sécurité réseau** : dans le cadre de la détection d'intrusion, de l'analyse de logiciels malveillants et de la surveillance de la sécurité réseau pour détecter les flux liés à des activités suspectes ;

Fonctionnalités principales

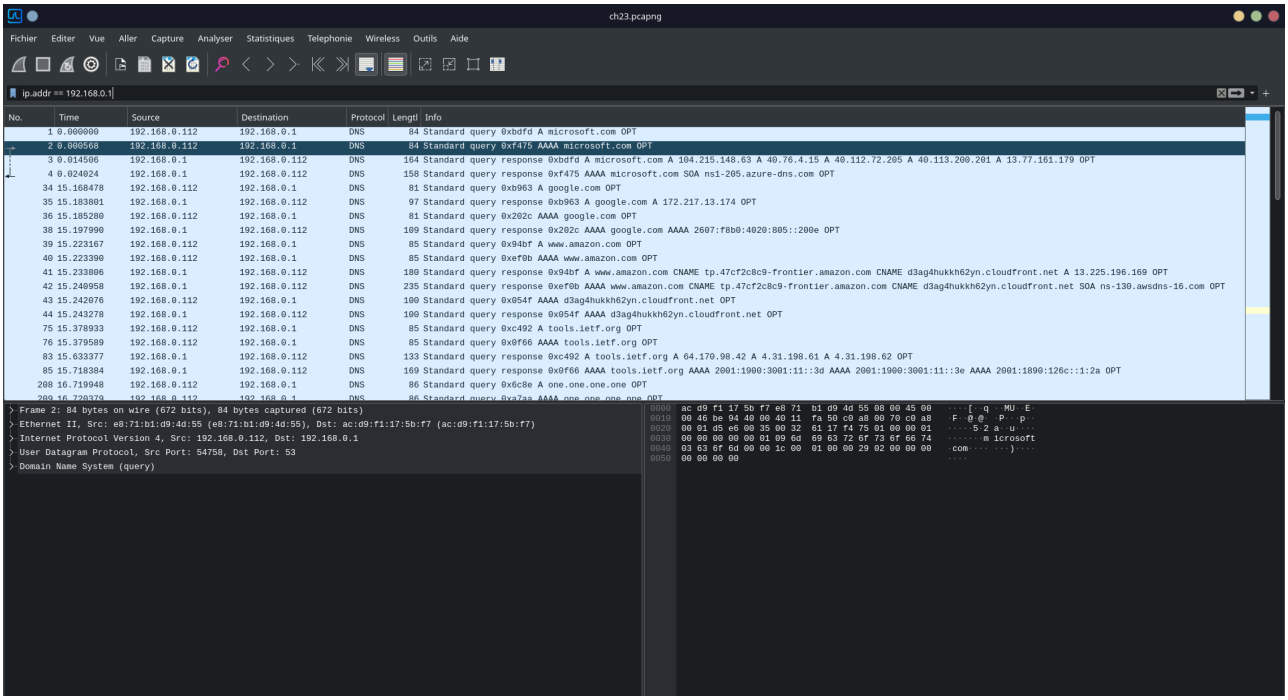
- **Filtrage avancé** : permet de filtrer les paquets en fonction de critères tels que l'adresse IP, le port, le protocole, etc. pour se concentrer sur des éléments spécifiques.
- **Analyse détaillée** : offre une analyse approfondie des paquets, y compris l'inspection des en-têtes, le suivi des flux de données, la reconstitution de sessions, etc.
- **Support de nombreux protocoles** : prend en charge un large éventail de protocoles réseau, tels que TCP, UDP, HTTP, DNS, FTP, et bien d'autres.
- **Exportation de données** : les résultats de l'analyse peuvent être exportés dans divers formats tels que CSV, XML et PCAP.
- Graphiques et statistiques **: permet de génèrer des graphiques et des statistiques pour visualiser les performances du réseau. ===== TShark ===== TShark est une version CLI de Wireshark qui permet les mêmes actions d'enregistrement et d'analyse de captures réseau mais au travers de la ligne de commande uniquement. TShark peut se montrer utile dans de nombreux cas : * Traitements automatisés via des scripts ; * Extraction de données ; * Usage sur serveurs ; * … ===== Exemple d'exploitation ou d'utilisation ===== Supposons que vous êtes confronté à des problèmes de latence sur un réseau local, vous pouvez lancer Wireshark et sélectionner l'interface réseau à surveiller. En appliquant un filtre pour ne capturer que le trafic pertinent et en filtrant le trafic vers ou depuis une adresse IP spécifique par exemple, vous pourrez, en analysant les résultats, identifier les sources de latence telles que des retards de transmission ou des pertes de paquets. * Analyse de paquets de données capturés sur un réseau**



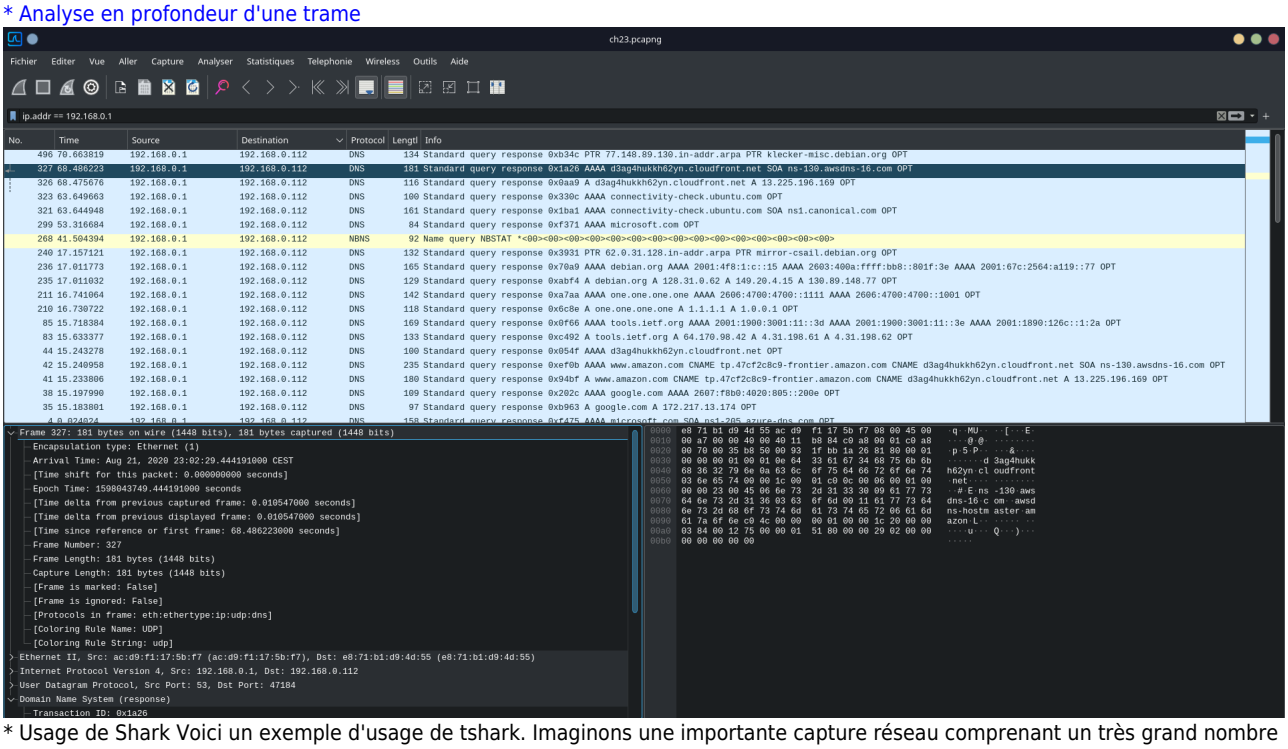
* Fonctionnalité de suivi de Flux (UDP, TCP...) * Analyse de paquets de données capturés sur un réseau



* Filtrage des paquets par adresse IP * Analyse de paquets de données capturés sur un réseau



* Analyse en profondeur d'une trame



* Usage de Shark Voici un exemple d'usage de tshark. Imaginons une importante capture réseau comprenant un très grand nombre de paquets, mais seuls les protocoles DNS et HTTP nous intéressent. Nous pouvons extraire les paquets comprenant ces protocoles dans une nouvelle capture en utilisant Tshark : <code class="“bash”> tshark -r capture_example.pcapng -Y 'http || dns' -w out.pcapng </code> Nous pouvons ensuite lire la capture avec tshark :
<code class="“bash”> tshark -r out.pcapng </code> <code> 1 0.000000000 192.168.121.3 → 192.168.202.49 DNS 103 Standard query response 0x6f3d A perdu.com A 104.21.5.178 A 172.67.133.176 2 0.000000209 192.168.121.3 → 192.168.202.49 DNS 127 Standard query response 0x8802 AAAA perdu.com AAAA 2606:4700:3033::6815:5b2 AAAA 2606:4700:3037::ac43:85b0 3 0.009499422 192.168.5.74 → 104.21.5.178 HTTP 141 GET / HTTP/1.1 4 0.377068826 104.21.5.178 → 192.168.5.74 HTTP 73 Continuation </code> ===== References ===== URL : * https://www.wireshark.org/docs/ * https://www.wireshark.org/docs/man-pages/tshark.html ====== Retour fiches outils ====== * Cyber fiches outils