GOVINDIN Devendra
BTS SIO- option SISR
2024-2026

Documentation Technique : Analyse de Trame avec Wireshark

L'outil Wireshark a été utilisé dans deux contextes techniques distincts au cours de ma formation. D'une part, pour valider les flux réseau et le cloisonnement de l'infrastructure Vénus dans le cadre du projet E6 (NIDS Suricata). D'autre part, lors d'un TP de cybersécurité visant à démontrer la vulnérabilité du protocole HTTP face à l'interception de données d'authentification.

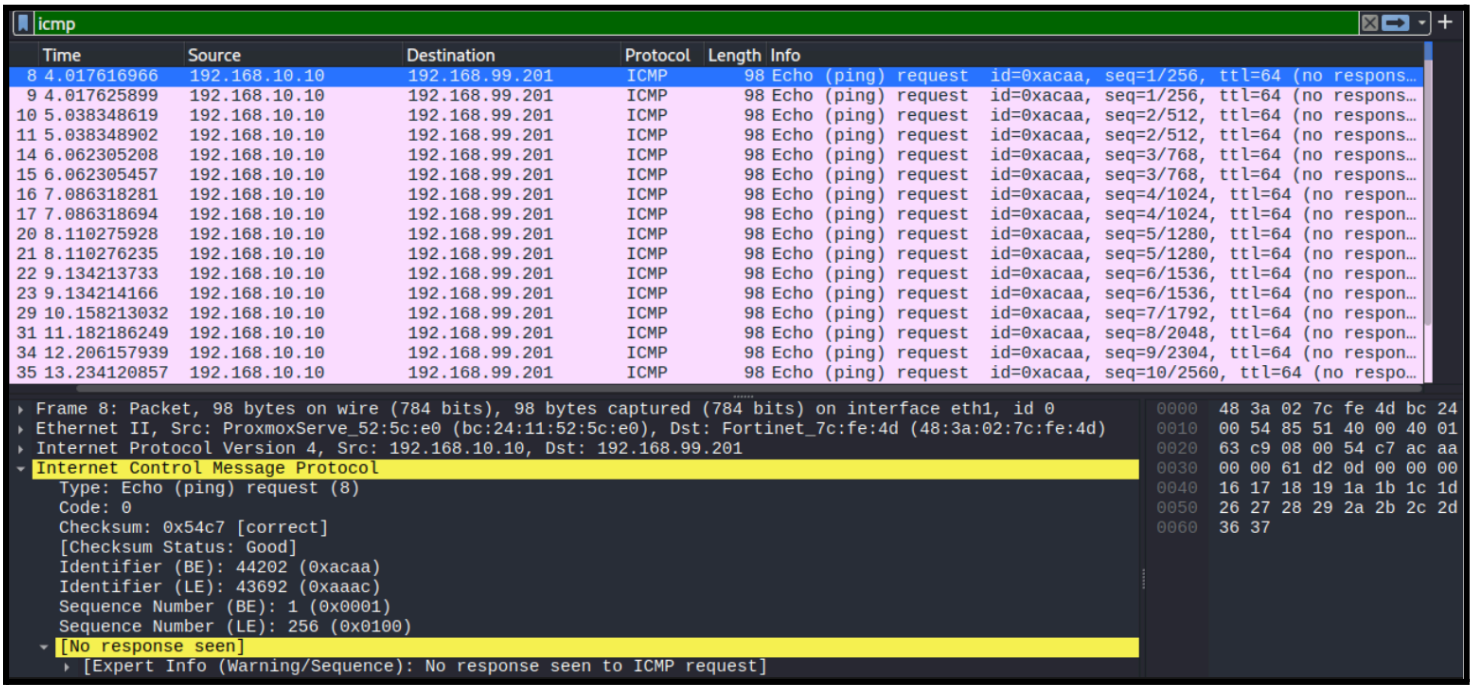
1. Contexte Projet Vénus : Analyse du trafic ICMP inter-VLAN

Dans le cadre du déploiement de la sonde Suricata sur l'infrastructure Vénus, Wireshark a été utilisé pour capturer et analyser les tentatives de communication entre différentes zones réseau. *(Documentation complète du NIDS - Projet 2 E6)*

1.1. Validation du cloisonnement réseau

La capture réseau illustre une tentative de balayage ICMP (Ping sweep) initiée depuis une machine utilisateur du VLAN 10 (IP source 192.168.10.10) à destination d'un serveur critique situé dans le VLAN 99 (IP de destination 192.168.99.201). Un poste de travail standard n'ayant aucune raison légitime de vérifier la connectivité des serveurs d'infrastructure, ce trafic est considéré comme suspect.

L'analyse de la trame via Wireshark confirme que le pare-feu remplit son rôle de ségrégation : les requêtes "Echo (ping) request" restent sans réponse, comme l'indique le message d'avertissement "No response seen to ICMP request".

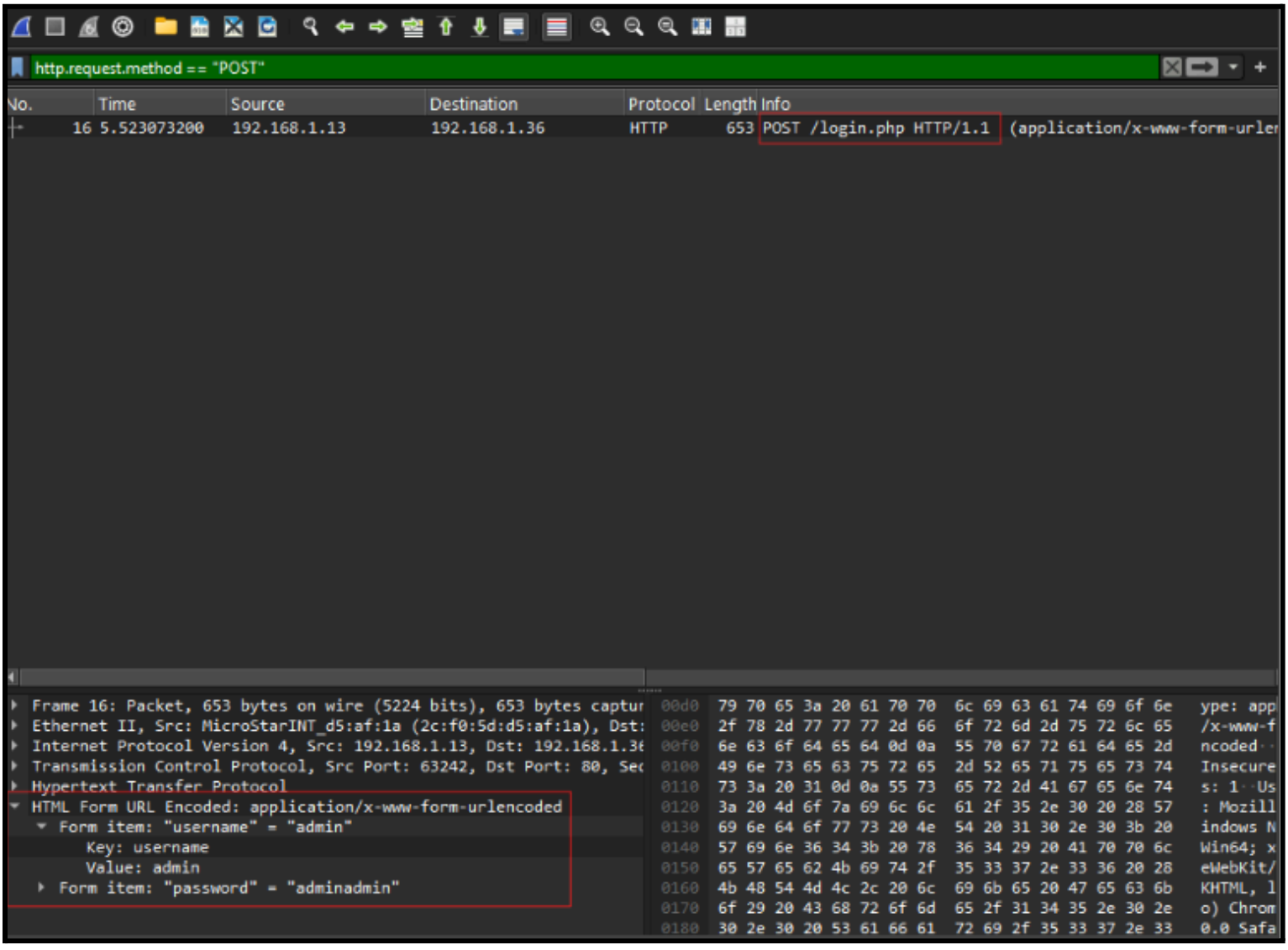


2. Contexte TP Cybersécurité : HTTP vs HTTPS

Ce second cas d'usage de Wireshark illustre une vulnérabilité applicative classique : l'interception d'identifiants de connexion transitant en clair sur un réseau local.

2.1. Interception des identifiants en HTTP

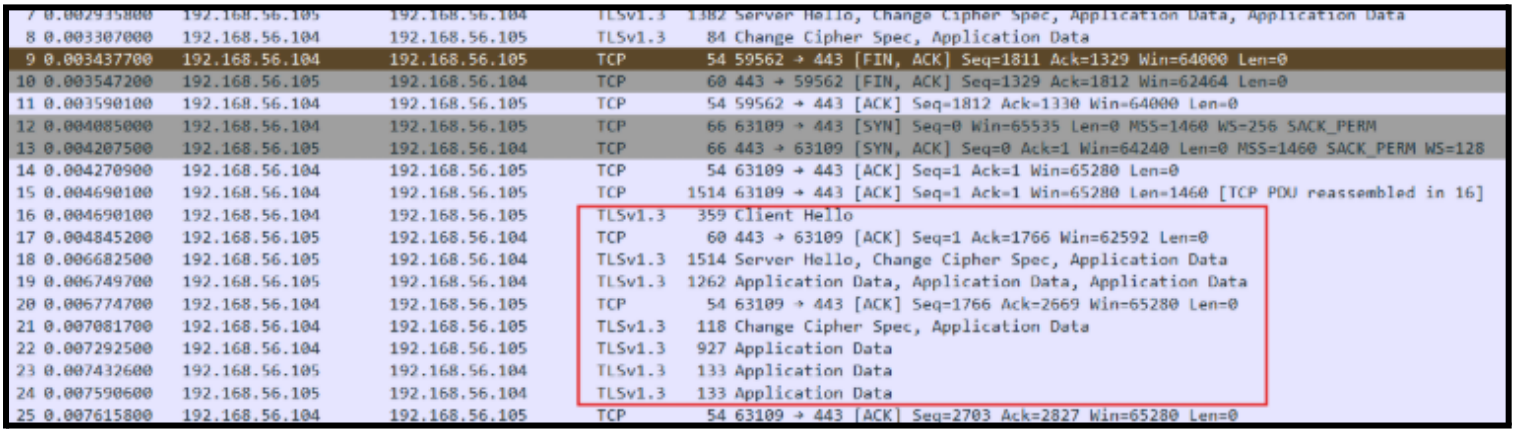
Lors de la première phase du TP, un formulaire d'authentification a été hébergé sur un serveur web Apache classique (port 80). La capture Wireshark, filtrée avec l'expression `http.request.method == "POST"`, permet d'isoler la requête d'authentification. L'analyse du paquet démontre que les variables envoyées par l'utilisateur sont totalement lisibles : la section "HTML Form URL Encoded" expose le nom d'utilisateur et le mot de passe en texte clair. Cette démonstration prouve qu'un attaquant positionné sur le réseau local peut facilement compromettre des comptes utilisateurs.

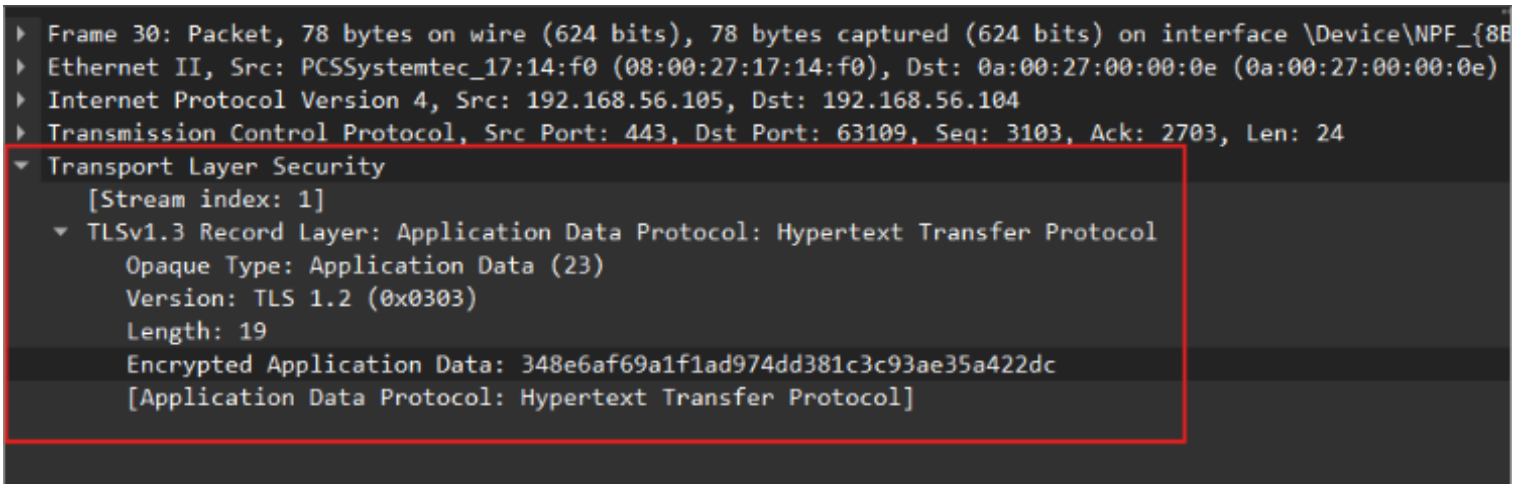


2.2. Preuve de sécurisation par TLS (HTTPS)

Pour corriger cette faille, un certificat auto-signé a été généré via OpenSSL et déployé sur le serveur Apache pour chiffrer les échanges sur le port 443. Une nouvelle capture Wireshark a été réalisée (filtre `tcp.port == 443`) lors de la soumission du même formulaire.

L'analyse de cette nouvelle trame montre l'absence totale de la section "HTML Form URL Encoded". À la place, les identifiants sont encapsulés sous la mention "TLSv1.3 Application Data" et remplacés par un bloc nommé "Encrypted Application Data". L'outil Wireshark confirme ainsi visuellement que le chiffrement empêche toute lecture des données sensibles lors de leur transit sur le réseau.





Preuve visuelle : plus de section « HTML Form URL Encoded » avec `username` / `password`. Tout est remplacé par « Encrypted Application Data ».

Comparaison avec HTTP

HTTP (Partie 1) :

```
├── Protocol : HTTP
├── Port : 80
└── Contenu : username=admin & password=adminadmin (clair)
```

HTTPS (Partie 2) :

```
├── Protocol : TLSv1.3
├── Port : 443
└── Contenu : Encrypted Application Data (illisible)
```