RAPPORT DE SUIVI DE PROJET

Projet fil rouge Althea Systems

Modernisation de l'infrastructure réseau et sécurité

Mise en place d'une supervision de sécurité (SIEM / IDS-IPS)

Rédigé par : Dylan Signing Kemlekeu Boing

Formation : Bachelor Cybersécurité, INGESTIS Paris

Méthodologie de gestion de projet : Trello (Kanban)

Date : Octobre 2026

1. Contexte et objectifs du projet

Le projet fil rouge consiste à concevoir et déployer la modernisation de l'infrastructure réseau et sécurité d'Althea Systems, éditeur de solutions SaaS pour le secteur de la santé, réparti sur deux sites (Paris et Genève). Le périmètre couvre notamment le réseau, la virtualisation, la sécurité opérationnelle (SOC/SIEM/EDR-XDR), la messagerie et les outils collaboratifs, le service de fichiers, ainsi que la continuité d'activité (PRA/PCA).

Ce rapport présente l'état d'avancement du volet « Sécurité opérationnelle », avec la mise en place d'une chaîne de supervision (SIEM) et de détection d'intrusion (IDS/IPS), déployée et testée sur un environnement de laboratoire représentatif de l'architecture cible.

2. Méthodologie de gestion de projet

Conformément au cadrage retenu pour ce projet, le suivi est assuré via Trello plutôt que Jira. Le tableau du projet est organisé en listes représentant le flux de traitement (Backlog, Sprint en cours, En revue, Terminé), avec des étiquettes par domaine technique (Réseau & Sécurité, SOC/SIEM/EDR-XDR, Messagerie, Fichiers, etc.). Chaque exigence du cahier des charges a été déclinée en cartes, dotées d'une checklist de définition de « terminé », permettant une traçabilité directe entre le cahier des charges et le travail réalisé.

Le présent document couvre la réalisation concrète des cartes du domaine « Sécurité opérationnelle (SOC/SIEM/EDR-XDR) ».

3. Environnement de laboratoire

Afin de valider les choix techniques avant leur formalisation dans l'étude, un environnement de laboratoire a été mis en place :

- Une machine virtuelle Debian 12 (« althea », 192.168.2.134), hébergeant les briques de supervision Linux (Docker, SIEM, IDS/IPS).
- Un serveur Windows Server avec le rôle Active Directory Domain Services (192.168.2.135), représentant l'annuaire d'entreprise.

Cette séparation reproduit la logique de l'architecture cible, où les services de supervision collectent les événements provenant à la fois des systèmes Windows (annuaire, postes) et Linux (serveurs applicatifs, réseau).

4. Mise en place du SIEM

4.1 Choix initial : Wazuh

Le choix technique initial s'est porté sur Wazuh, solution SIEM open source déjà pratiquée dans un contexte de formation précédent. Les premières tentatives d'installation sur la VM Debian 12 ont toutefois rencontré des blocages liés à l'indisponibilité du paquet dans les dépôts de cette version de Debian, ainsi que des difficultés de déploiement des agents.

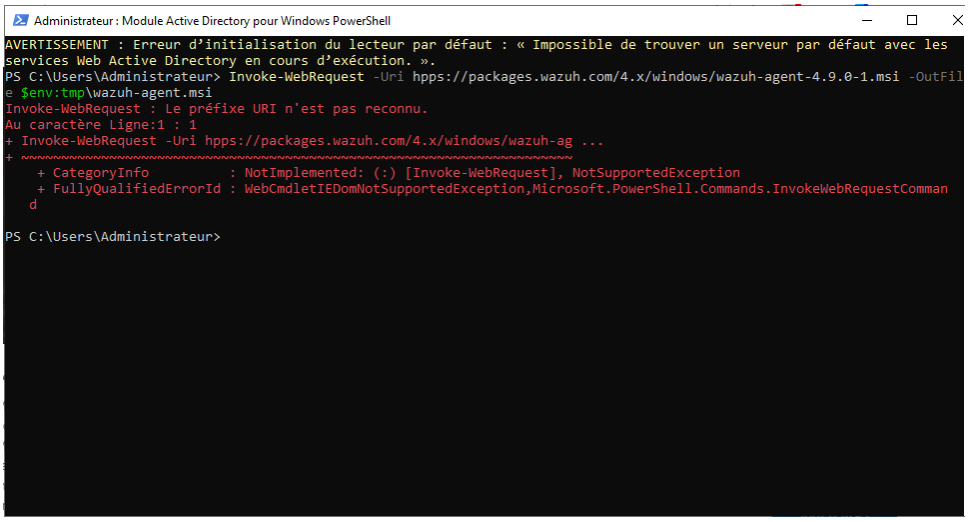


Figure 1 : Erreur lors du téléchargement de l'agent Wazuh sur un poste Windows (faute de frappe dans l'URL HTTPS).

Face à la récurrence de ces blocages et à l'enjeu de délai du projet, une réorientation technique a été actée : le SIEM central repose désormais sur Graylog, solution également open source, déployée par conteneurs Docker, ce qui permet de s'affranchir des contraintes de disponibilité des paquets natifs rencontrées avec Wazuh.

4.2 Déploiement de Graylog (Docker Compose)

La stack Graylog a été déployée via Docker Compose, avec trois services : MongoDB (métadonnées), OpenSearch (indexation des journaux) et Graylog (moteur et interface). Le déploiement initial a nécessité plusieurs corrections successives : installation du moteur Docker sur Debian 12, correction d'un dépôt APT mal configuré, puis résolution d'une erreur de configuration (URI externe HTTP invalide) qui empêchait le démarrage du service.

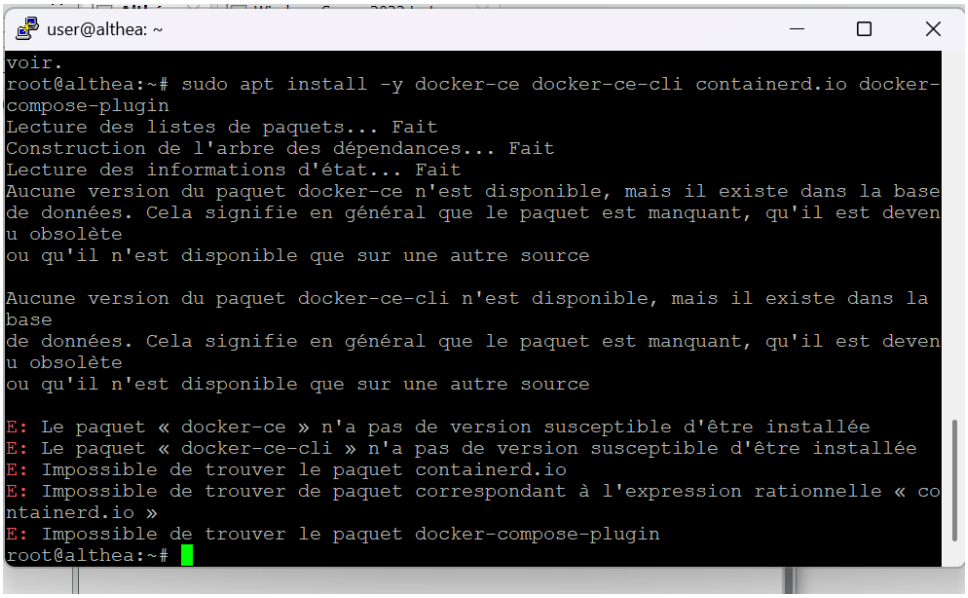


Figure 2 : Échec initial d'installation de Docker : le dépôt APT n'était pas encore correctement enregistré.

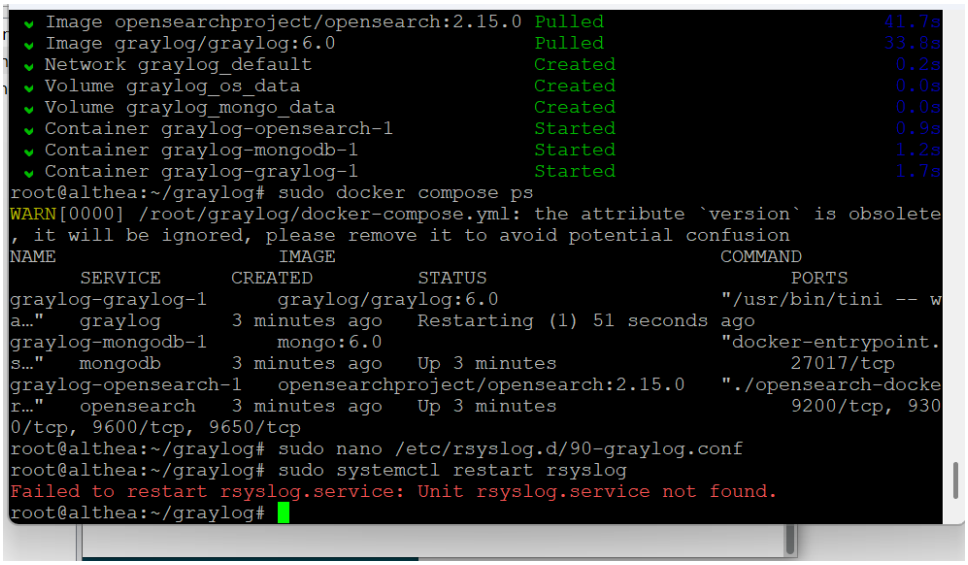


Figure 3 : Premier déploiement de la stack Graylog : les conteneurs MongoDB et OpenSearch démarrent, le conteneur Graylog redémarre en boucle.

L'analyse des journaux du conteneur a révélé une variable d'environnement mal renseignée (adresse IP du serveur laissée sous forme de paramètre non substitué). Une seconde anomalie, plus profonde, a ensuite été identifiée : le conteneur Graylog ne parvenait pas à résoudre le nom d'hôte du service MongoDB, le fichier de configuration ne déclarant pas explicitement l'URI de connexion à la base. Le fichier a été entièrement réécrit afin de garantir une configuration saine, avec les variables GRAYLOG_MONGODB_URI et GRAYLOG_ELASTICSEARCH_HOSTS correctement définies.

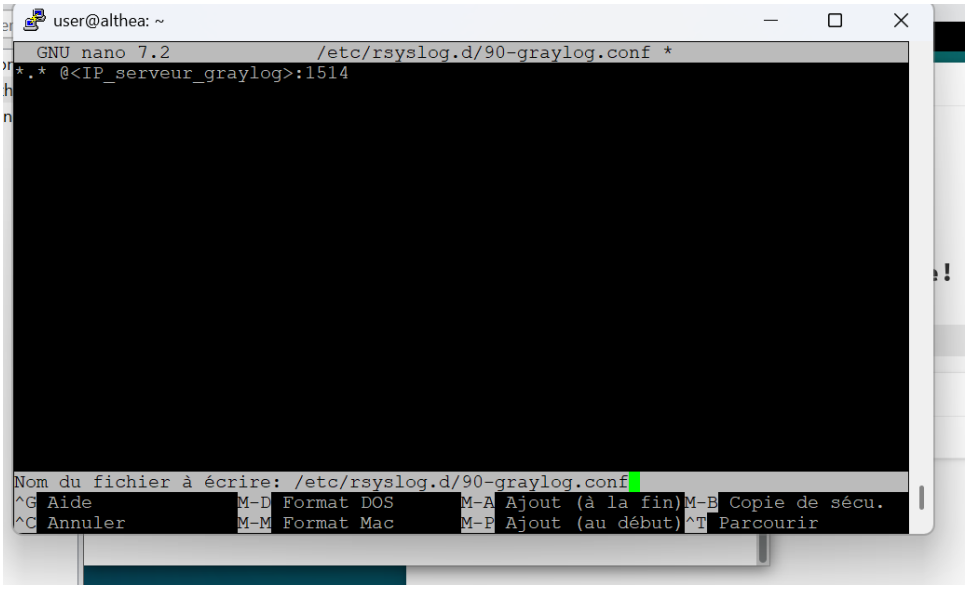


Figure 4 : Correction du fichier de configuration rsyslog, avec une valeur de paramètre restée non substituée (erreur identifiée et corrigée).

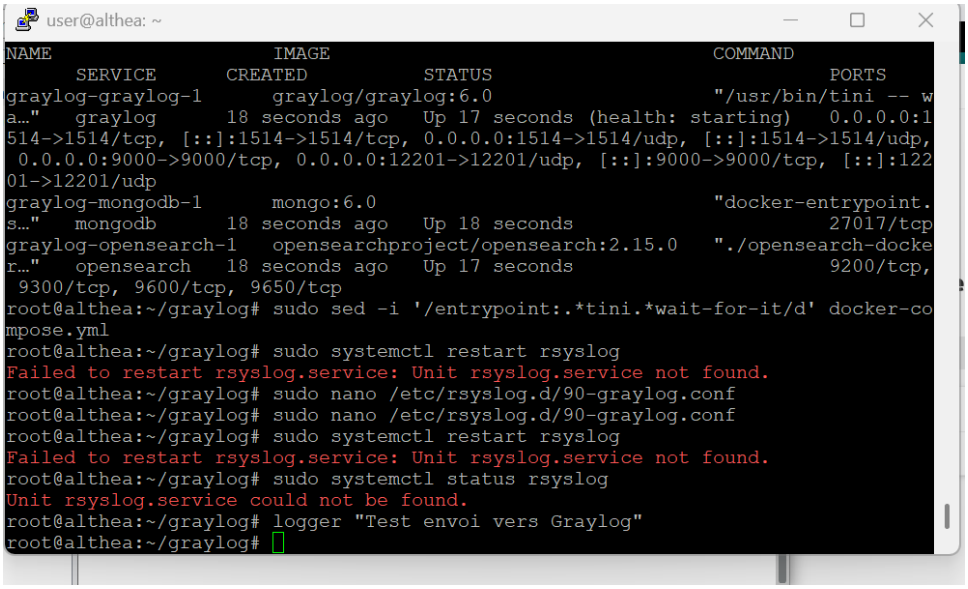


Figure 5 : Le service rsyslog, nécessaire à la remontée des journaux, n'était pas installé sur la VM : installation réalisée a posteriori.

4.3 Résolution de l'accès à l'interface web

Une fois le conteneur stabilisé, l'accès à l'interface web (port 9000) restait indisponible depuis le poste d'administration Windows. Un diagnostic méthodique (vérification du conteneur, du port en écoute côté VM, puis de la connectivité réseau côté poste Windows) a permis d'écarter une cause réseau : le test de connectivité confirmait que le port n'était tout simplement pas encore exposé par le conteneur, celui-ci n'ayant pas terminé son initialisation.

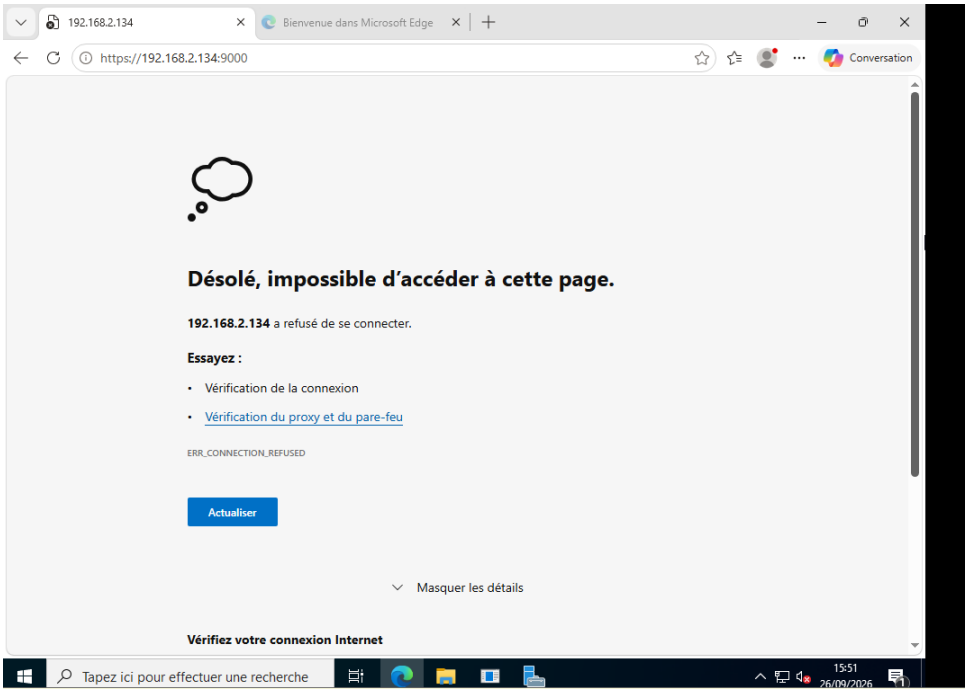


Figure 6 : Tentative d'accès à l'interface Graylog : connexion refusée, le service n'était pas encore pleinement opérationnel.

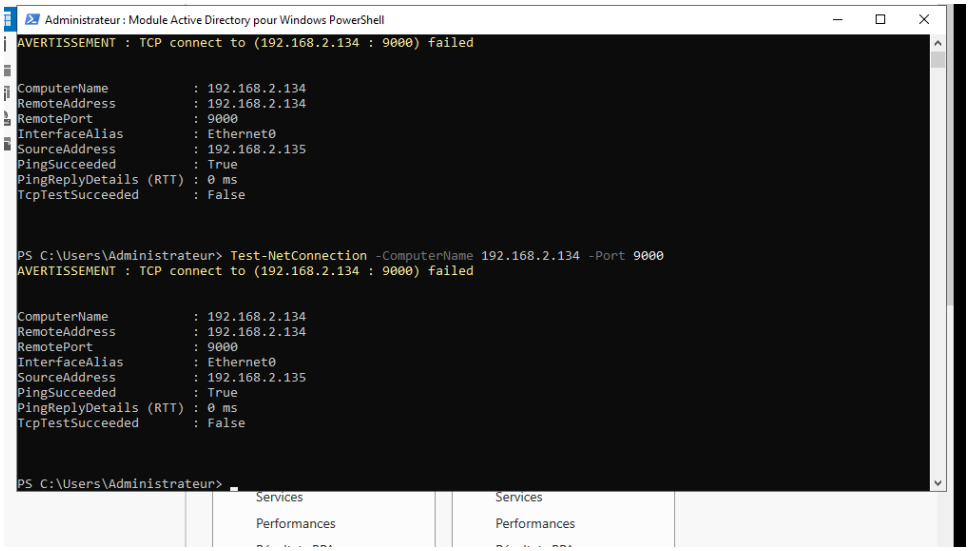


Figure 7 : Test de connectivité réseau (PowerShell) : la résolution ping aboutit, mais le port TCP 9000 reste fermé, confirmant un problème côté service plutôt que réseau.

Après recréation complète du conteneur avec la configuration corrigée, l'interface Graylog est devenue accessible et la connexion avec le compte administrateur a pu être validée.

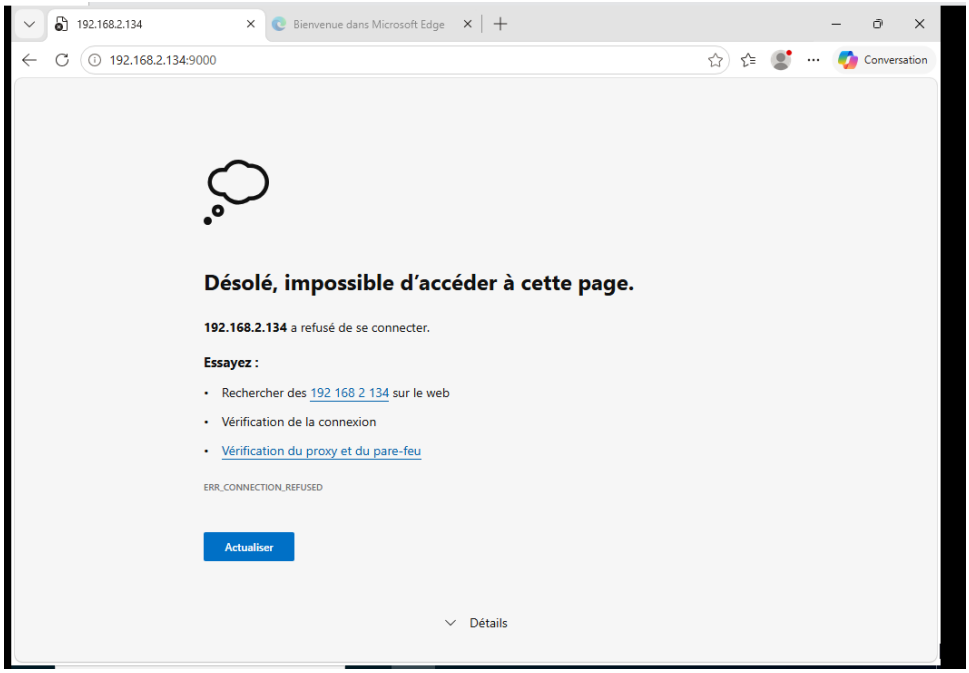


Figure 8 : Page d'authentification Graylog, accessible une fois le service correctement configuré.

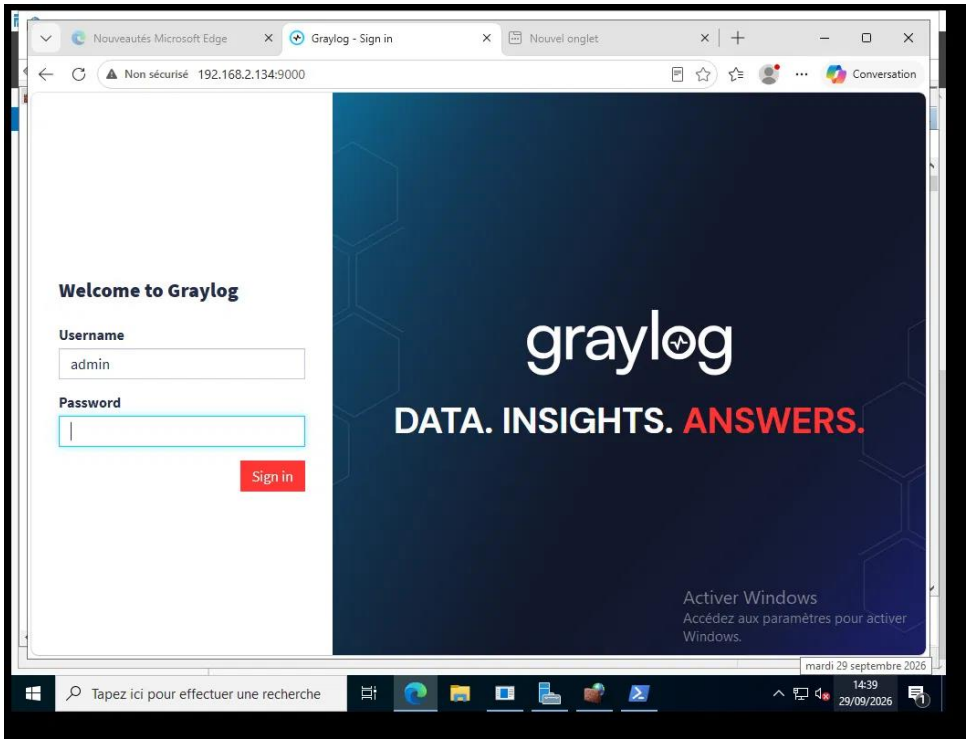


Figure 9 : Connexion réussie à l'interface Graylog.

5. Mise en place de l'IDS/IPS (Suricata)

Le volet détection réseau repose sur Suricata. Le paquet natif Suricata s'étant révélé indisponible dans les dépôts Debian 12 (retiré de la branche « oldstable »), la même approche que pour Graylog a été retenue : un déploiement par conteneur Docker, en mode réseau « host » afin de donner à la sonde un accès direct au trafic de l'interface physique de la VM.

Après correction du nom de l'interface réseau à surveiller, le conteneur a démarré correctement. La mise à jour des règles de détection (jeu de règles Emerging Threats Open) a ensuite été exécutée avec succès : plus de 53 000 règles ont été chargées sans erreur, et le moteur a généré sa première alerte de test, confirmant le bon fonctionnement de la détection.

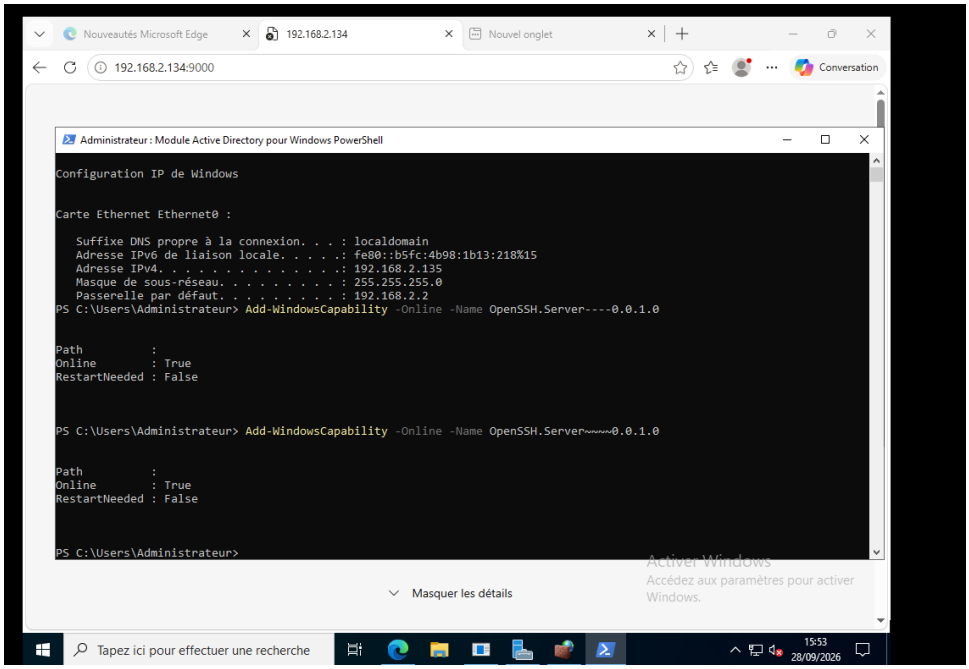


Figure 10 : Installation du client OpenSSH sur le serveur Windows, nécessaire à l'établissement d'un tunnel sécurisé vers l'interface Graylog.

6. Intégration de Suricata dans Graylog

La dernière étape en cours consiste à acheminer les journaux générés par Suricata (format JSON, fichier eve.json) vers Graylog, via rsyslog, afin de centraliser la détection réseau dans le même outil de supervision que les autres sources.

6.1 Vérification de l'état du serveur Graylog

Le tableau de bord « Nodes » confirme que le serveur Graylog fonctionne normalement (nœud actif, traitement des messages activé), avec toutefois un débit d'entrée/sortie nul au moment de la capture, ce qui corrobore l'absence de résultats observée lors des recherches.

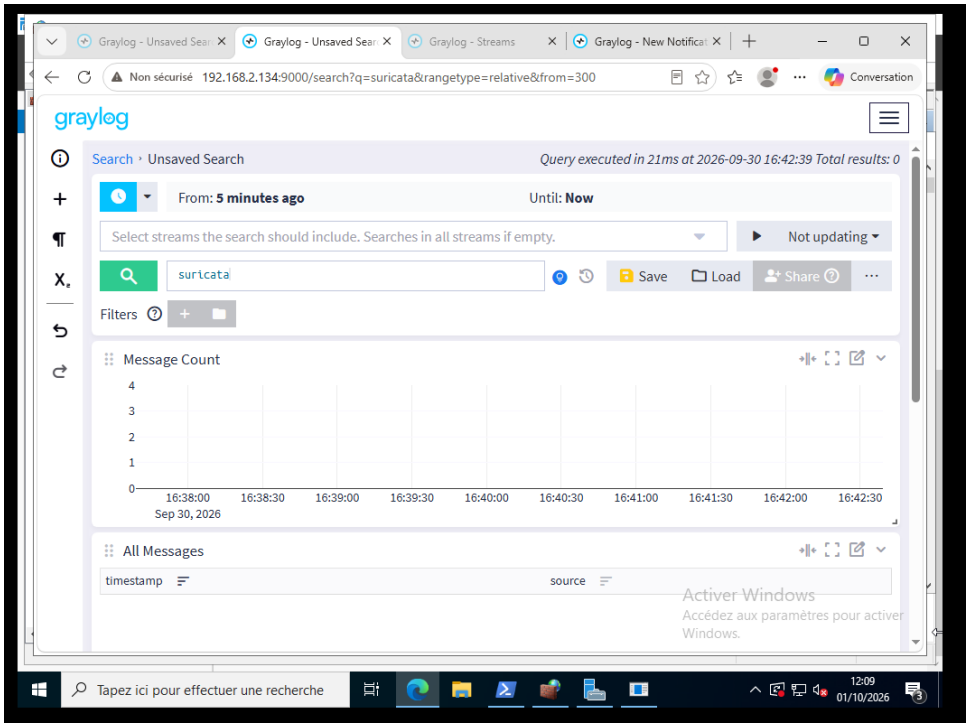


Figure 11 : Tableau de bord « System Overview » : le serveur traite ponctuellement des messages (débit de 4 messages/s au moment du test).

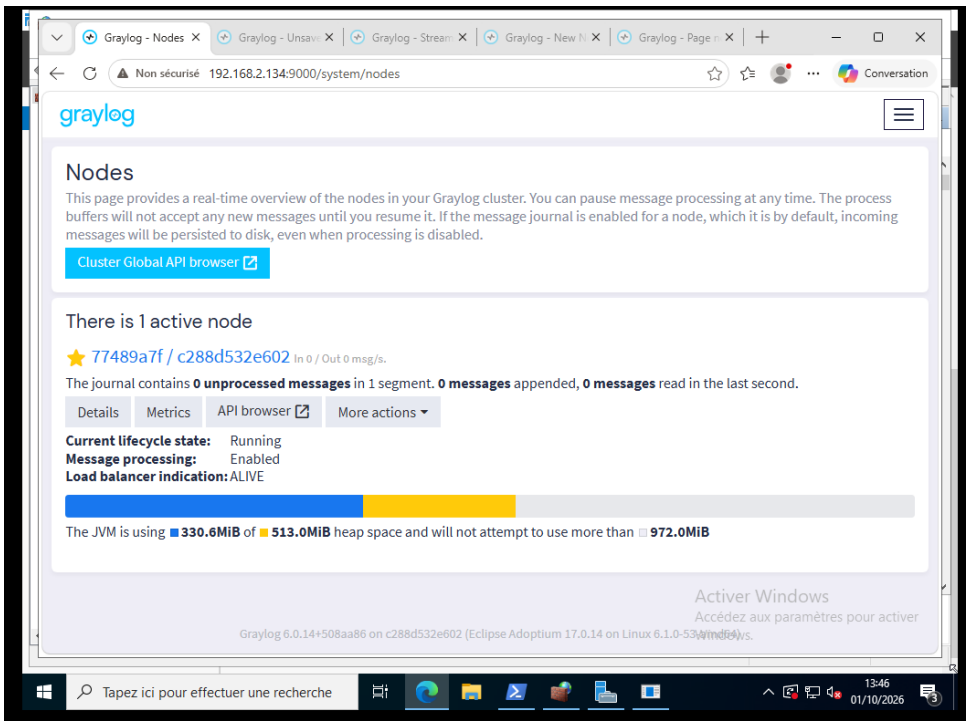


Figure 12 : Page « Nodes » : nœud actif et sain, mais débit d'entrée/sortie à 0 message/s au moment de la vérification.

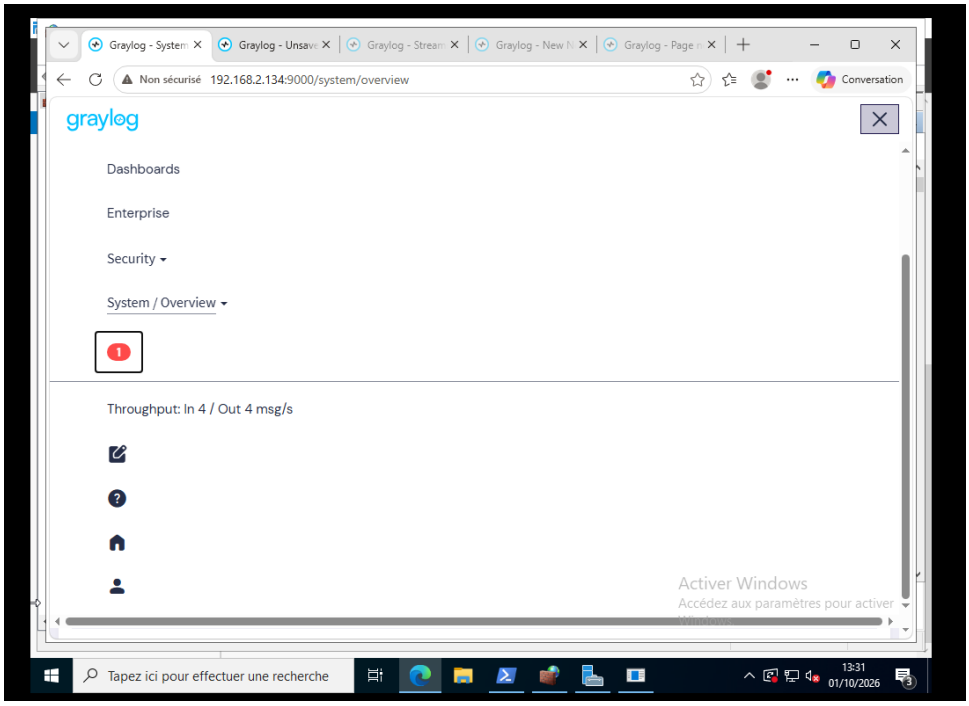


Figure 13 : Journal des notifications système Graylog, sans anomalie bloquante relevée pour l'instance.

6.2 Incident identifié : taille maximale des messages rsyslog

L'investigation a mis en évidence la cause racine de l'absence des journaux Suricata dans Graylog : les journaux rsyslog du système indiquaient de manière répétée l'erreur « message too long », rsyslog appliquant par défaut une limite de 8 192 octets par message. Or les lignes JSON générées par Suricata (en particulier les événements de type statistiques) dépassent régulièrement cette taille, ce qui entraînait leur rejet silencieux avant même leur transmission à Graylog.

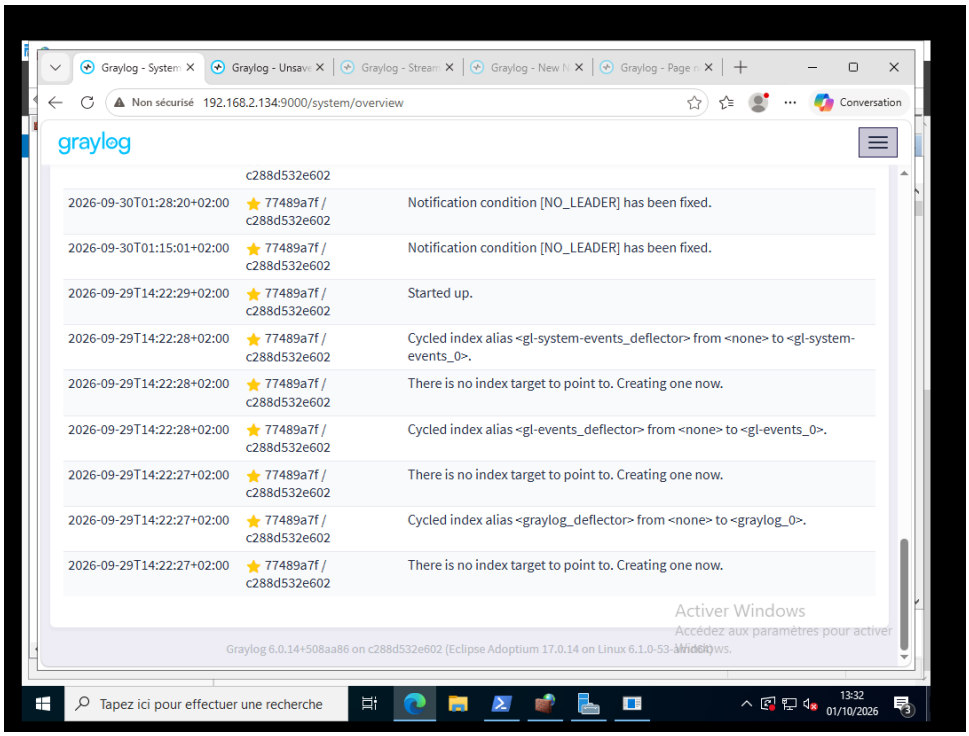


Figure 14 : Recherche « suricata » dans Graylog : aucun résultat, alors que la sonde Suricata fonctionne normalement côté système.

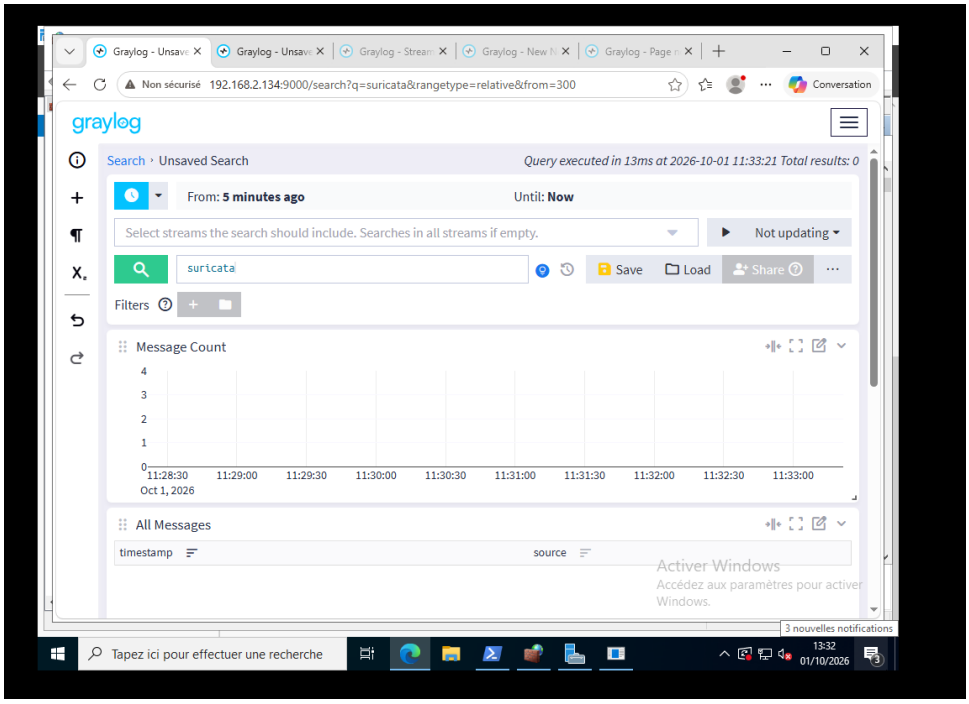


Figure 15 : Confirmation de l'absence de résultats sur une fenêtre de recherche élargie.

Le correctif appliqué consiste à relever explicitement cette limite dans la configuration globale de rsyslog (directive $MaxMessageSize, portée à 64 kilooctets), seule modification suffisante pour permettre le transit des messages volumineux générés par Suricata.

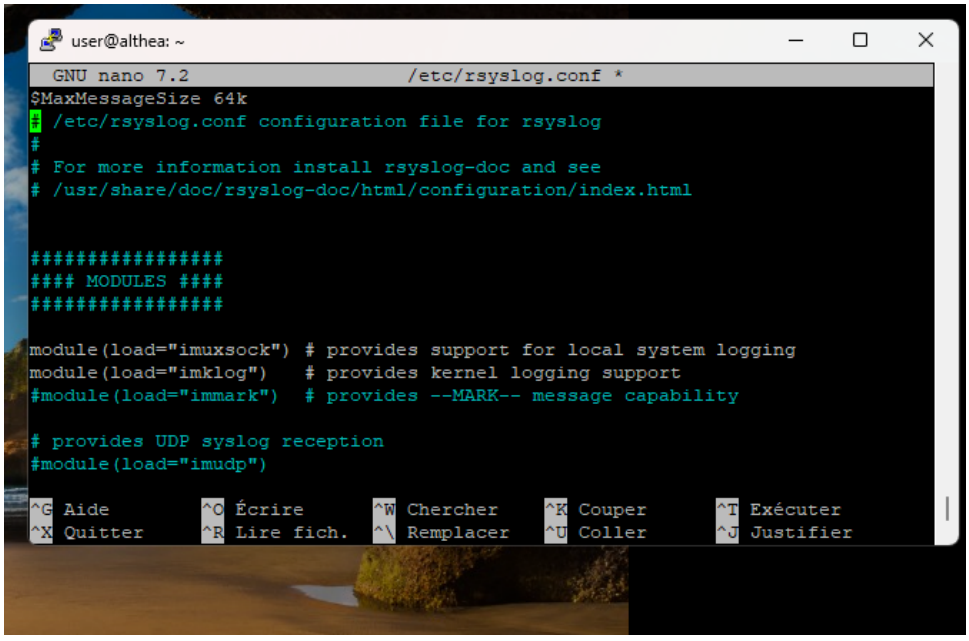


Figure 16 : Ajout de la directive $MaxMessageSize 64k en tête du fichier de configuration rsyslog.

Ce correctif a été appliqué et rsyslog redémarré avec succès, sans réapparition de l'erreur depuis la modification. La vérification finale de la bonne réception des journaux Suricata dans Graylog (recherche sur les champs applicatifs du flux JSON, puis finalisation du Stream et de la règle d'alerte dédiés) reste l'action en cours à ce stade.

7. Bilan et prochaines étapes

À ce stade du projet, les réalisations suivantes sont actées et vérifiées :

- Pivot argumenté de Wazuh vers Graylog pour le SIEM, documenté et fonctionnel.
- Déploiement de la stack Graylog (MongoDB, OpenSearch, Graylog) opérationnel, interface web accessible et authentification validée.
- Déploiement de Suricata en conteneur, règles de détection chargées, moteur de détection validé par un test positif.
- Cause racine de l'absence de remontée des journaux Suricata identifiée et corrigée (limite de taille des messages rsyslog).

Les actions suivantes restent à finaliser :

- Confirmer la réception effective des journaux Suricata dans Graylog après correctif, par une recherche ciblée sur le flux applicatif.
- Créer l'extracteur JSON, le Stream dédié et la règle d'alerte (notification e-mail) pour les événements de détection Suricata, selon le même modèle que celui prévu pour la détection de force brute SSH.
- Documenter la procédure finale dans le tableau Trello (cartes et checklists du domaine SOC/SIEM/EDR-XDR), en reprenant les correctifs validés dans ce rapport.

L'ensemble de la démarche (diagnostics, causes racines identifiées, correctifs appliqués et validés) a été conservé afin d'alimenter la documentation méthodologique attendue dans le rendu final du projet.

Fait à Argenteuil, le 16 octobre 2026.

Signature :

Dylan Signing Kemlekeu Boing
Étudiant en Bachelor Cybersécurité,
INGESTIS Paris