UNIVERSITÉ DE LA RÉUNION

DE LA RÉUNION
Institut Universitaire de Technologie
Réseaux et Télécommunications

Rapport

SUPERVISION DE LA SÉCURITÉ

Présenté par

Hoareau Anthony

BOYER Arthur

Encadré par

M. Fontaine

M. Montegu

BUT 3 Réseaux & Télécoms

Année universitaire 2025-2026

Janvier 2026

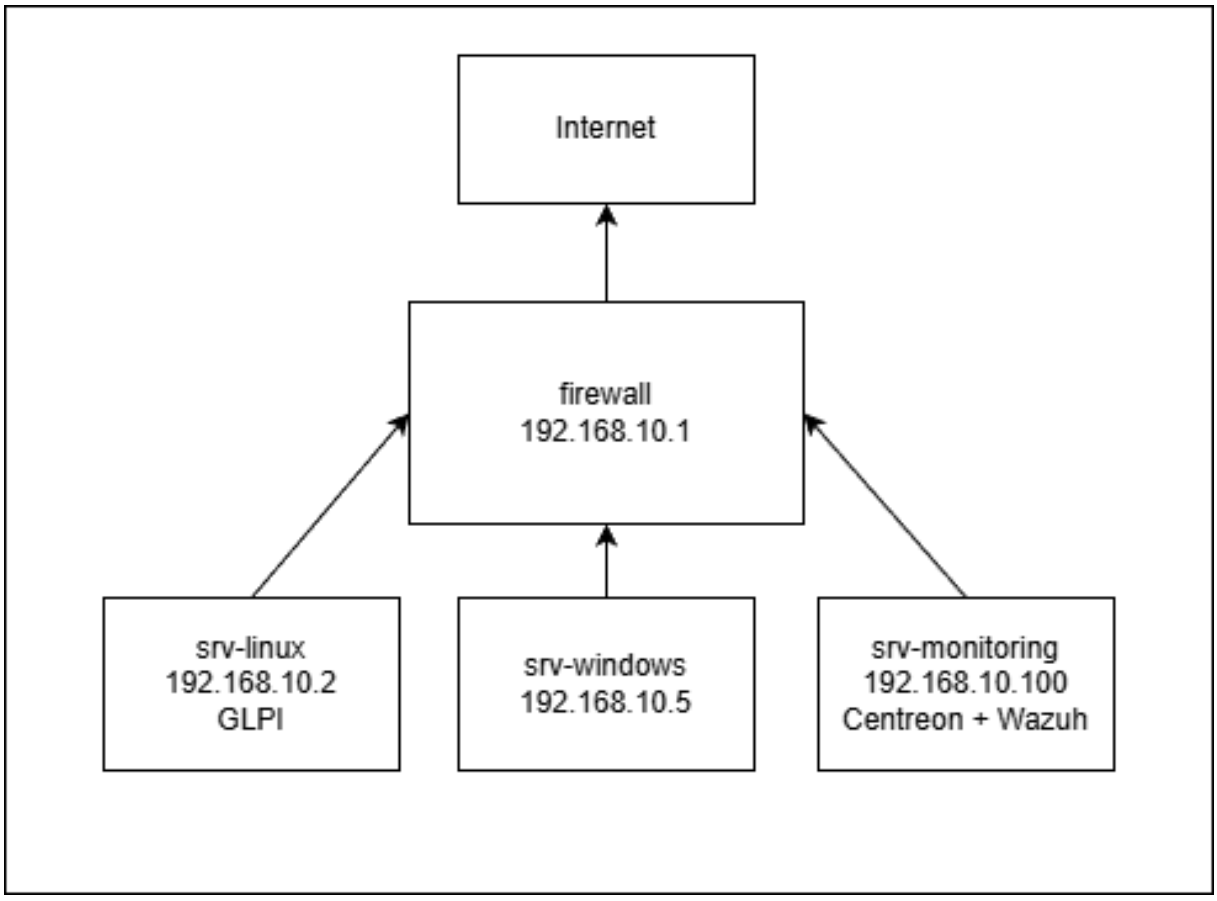
Table des matières

1. Contexte

1.1. Architechture Visé

- Un serveur Linux (services + supervision agent)
- un serveur Windows (agent + logs)
- un pare-feu / routeur (SNMP)
- un serveur de supervision central (Zabbix ou Centreon)
- un SIEM Wazuh centralisé avec agents sur chaque machine
- Un outil de ticketing (GLPI ou Jira Cloud) pour remonter les alertes Supervision et Wazuh

1.2. Schémas Réseaux



1.3. Objectifs

- Concevoir une architecture de supervision réseau et système.
- Configurer SNMP, agents et sondes de supervision.
- Collecter et centraliser les logs pour analyse.
- Corréler les événements dans un SIEM (Wazuh).
- Détecter un incident simulé (attaque brute-force SSH).
- Mise en place d'une gestion des événements basés sur ITIL et ITSM

Machine	Adresse IP
srv-linux	192.168.10.2
srv-windows	192.168.10.5
firewall	192.168.10.1
srv-monitoring	192.168.10.100

2. Configurations

2.1. Windows Server

2.1.1. SNMP Installation

Pour installer SNMP, nous avons suivie les étapes suivantes:
- Ouvrez le Gestionnaire de serveur
- Cliquez sur Gérer –> Ajouter des rôles et fonctionnalités
- Avancez jusqu'à l'étape Fonctionnalités
- Faites défiler vers le bas et cochez Service SNMP
- Cliquez sur Suivant puis Installer

2.1.2. SNMP Configuration

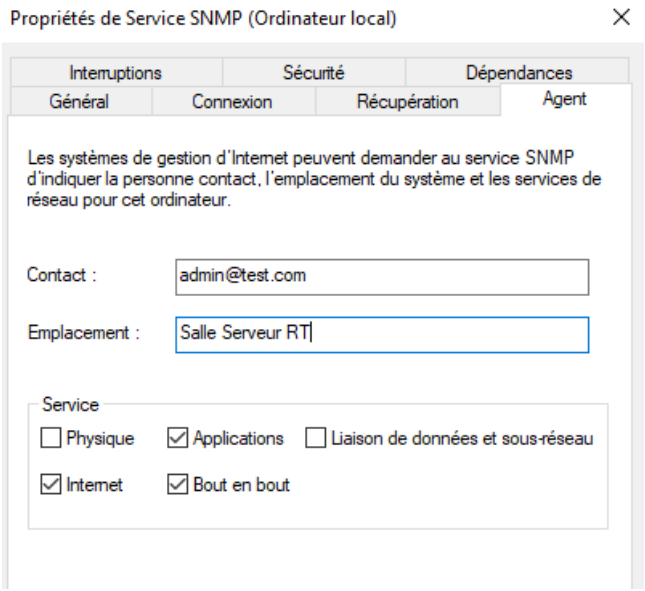
Une fois installé, SNMP se configure via la console des services Windows (services.msc). Il n'y a pas d'application dédiée dans le menu Démarrer.

- Appuyez sur Win + R, tapez services.msc et validez.
- Cherchez le service nommé « Service SNMP ».
- Faites un clic droit dessus –> Propriétés.

Voici les trois onglets importants à configurer :

1. L'onglet « Agent »

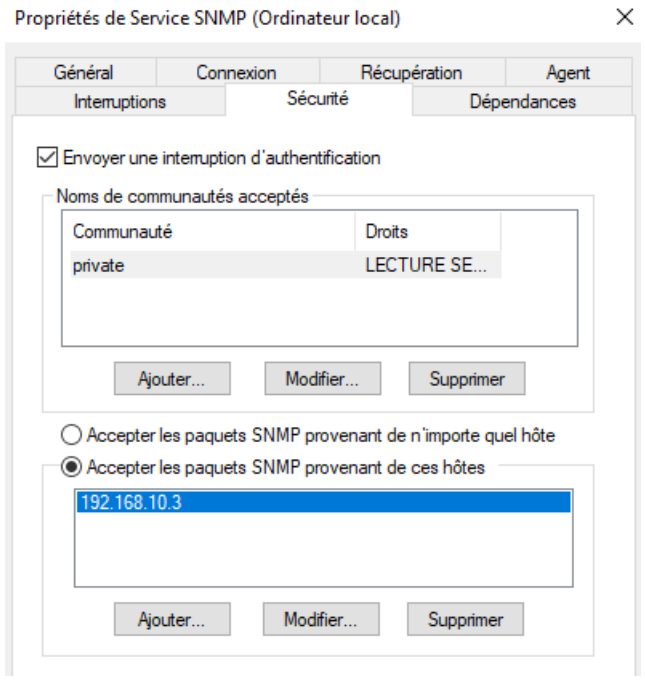
- Contact : Email ou nom de l'admin (ex: admin@test.com).
- Emplacement : Lieu physique du serveur (ex: Salle Serveur RT).
- Service : Cochez les cases « Applications », « Internet », « Bout en bout »



2. L'onglet « Sécurité »

C'est ici que vous configurez qui a le droit d'interroger ce serveur.

- Noms de communauté acceptés (Community Strings) :
  - Cliquez sur Ajouter.
  - Droit de communauté : Choisissez LECTURE SEULE.
  - Nom de communauté : C'est le « mot de passe » SNMP. Nous avons mis « private »
- Accepter les paquets SNMP provenant de ces hôtes :
  - Cochez Accepter les paquets SNMP provenant de ces hôtes.
  - Cliquez sur Ajouter et entrez l'adresse IP de votre serveur de supervision uniquement, ici pour nous ce sera « 192.168.10.3 »



2.2. OPNSense

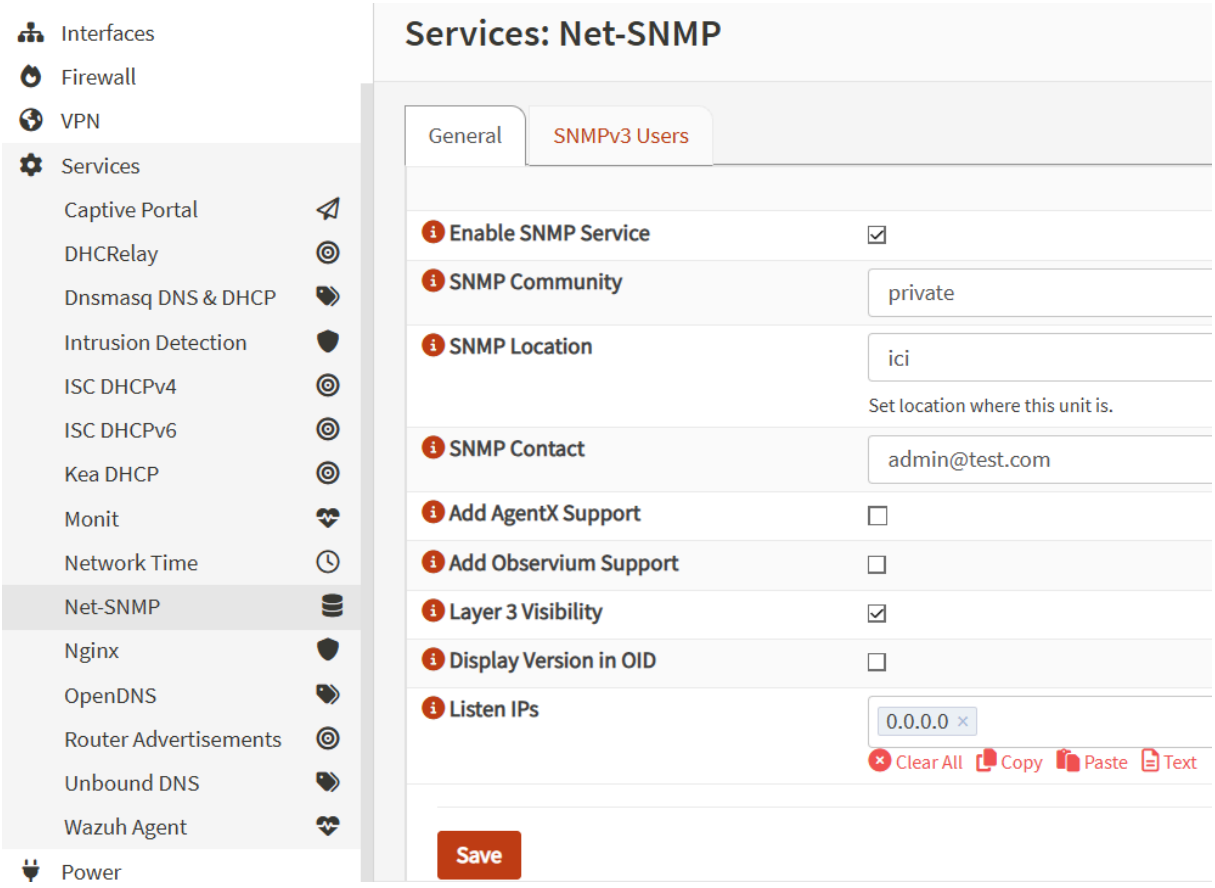
2.2.1. SNMP Installation

Pour installer SNMP, nous avons suivie les étapes suivantes:

- Pour installons le plugins « os-net-snmp » il faut ce rendre dans « System » –> « Firmware » –> « Plugins » –> cocher la case « Show community plugins » –> Rechercher « os-net-snmp » et l'installer

2.2.2. SNMP Configuration

- Ce rendre dans « Services » –> « Net-SNMP »
- Cocher la case « Enable SNMP Service »
- Compléter « SNMP Community » par « private »
- Compléter « SNMP Location » comme bon vous semble (représente le lieu physique du serveur)
- Compléter « SNMP Contact » comme bon vous semble (ex: admin@test.com)
- Cocher la case « Layer 3 Visibility »
- Cliquer sur « Save » pour sauvegarder la configuration



2.3. GLPI

2.3.1. Préparation de l'environnement pour GLPI

```
# Ceci est la suite des commande qui permet d'installer GLPI
sudo apt-get update && sudo apt-get upgrade
#Paquets principaux
sudo apt-get install apache2 php8.2-fpm mariadb-server
#Extensions nécessaires pour GLPI
sudo apt install php8.2-{curl,gd,intl,mysql,zip,bcmath,mbstring,xml,bz2}
#Extensions LDAP pour GLPI
sudo apt install php8.2-ldap
#La base de données pour GLPI
sudo mariadb-secure-installation
```

2.3.2. Installation de la base de données

```
#Pour se connecter à la base de données
sudo mysql -u root -p
```

```
# Création de la base de données
CREATE DATABASE db25_glpi;
GRANT ALL PRIVILEGES ON db25_glpi.* TO glpi_adm@localhost IDENTIFIED BY
"MotDePasseRobuste";
FLUSH PRIVILEGES;
EXIT
```

2.3.3. Installation de GLPI

```
#Nous l'installons depuis GitHub et L'archive sera téléchargée dans le
répertoire /tmp
cd /tmp
wget https://github.com/glpi-project/glpi/releases/download/11.0.4/glpi-11.0.4.
tgz
#Décompressions de l'archive .tgz dans le répertoire /var/www/
sudo tar -xzvf glpi-11.0.4.tgz -C /var/www/
#Définir l'utilisateur www-data correspondant à Apache2
sudo chown www-data /var/www/glpi/ -R
#Création du répertoire /etc/glpi qui va recevoir les fichiers de configuration
de GLPI
sudo mkdir /etc/glpi
sudo chown www-data /etc/glpi/
sudo mv /var/www/glpi/config /etc/glpi
sudo mkdir /var/lib/glpi
sudo chown www-data /var/lib/glpi/
sudo mv /var/www/glpi/files /var/lib/glpi
sudo mkdir /var/log/glpi
sudo chown www-data /var/log/glpi
#Création des fichiers de configuration
```

```
#1er fichier
sudo nano /var/www/glpi/inc/downstream.php
```

```
#Contenu du fichier downstream.php
<?php
define('GLPI_CONFIG_DIR', '/etc/glpi/');
if (file_exists(GLPI_CONFIG_DIR . '/local_define.php')) {
    require_once GLPI_CONFIG_DIR . '/local_define.php';
}
```

```
#2nd fichier
sudo nano /etc/glpi/local_define.php
```

```
#Contenu du fichier local_define.php
<?php
define('GLPI_VAR_DIR', '/var/lib/glpi/files');
define('GLPI_LOG_DIR', '/var/log/glpi');
```

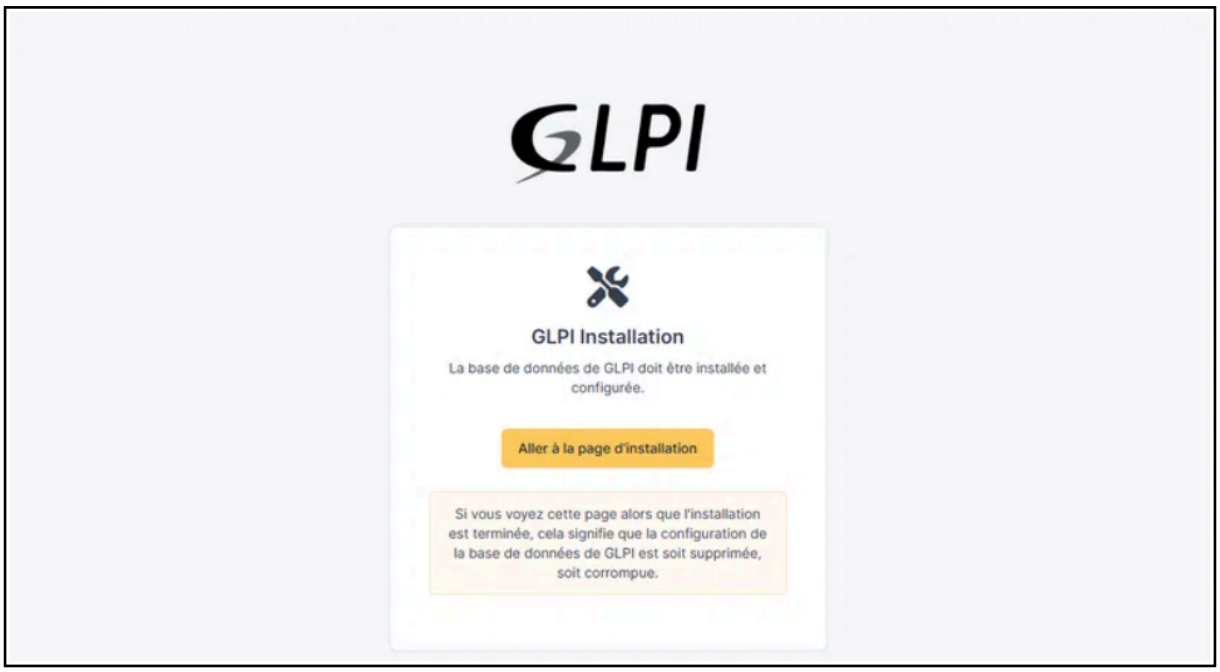
2.3.4. Installation d'Apache2 pour GLPI

```
#Nous allons créer un nouveau fichier de configuration qui va permettre de
configurer le VirtualHost dédié à GLPI.
sudo nano /etc/apache2/sites-available/supervision.fr.conf
```

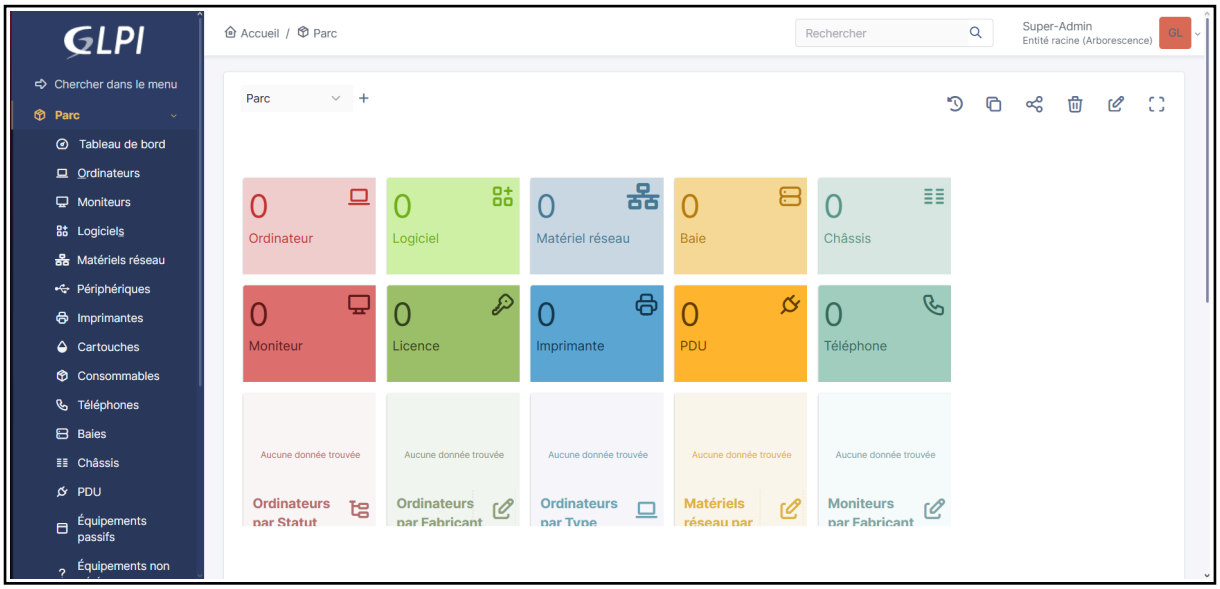
```
<VirtualHost *:80>
    ServerName supervision.fr
    DocumentRoot /var/www/glpi/public
    <Directory /var/www/glpi/public>
        Require all granted
        RewriteEngine On CalDAV, ...
        RewriteCond %{HTTP:Authorization} ^(.+)$
        RewriteRule .* - [E=HTTP_AUTHORIZATION:%{HTTP:Authorization}]
        RewriteCond %{REQUEST_FILENAME} !-f
        RewriteRule ^(.*)$ index.php [QSA,L]
    </Directory>
</VirtualHost>
```

```
#Activation du nouveau site
sudo a2ensite supervision.fr.conf
#Désactivation du site par défaut
sudo a2dissite 000-default.conf
#Activation du module Rewrite
sudo a2enmod rewrite
#Redémarrer le service Apache2
sudo systemctl restart apache2
```

Dans la barre de recherche de mon navigateur, je tape « supervision.fr ». Je me retrouve sur la page d'installation de GLPI.



Après avoir finis l'installation, je me trouve sur le dashboard de GLPI et prêt à l'utiliser

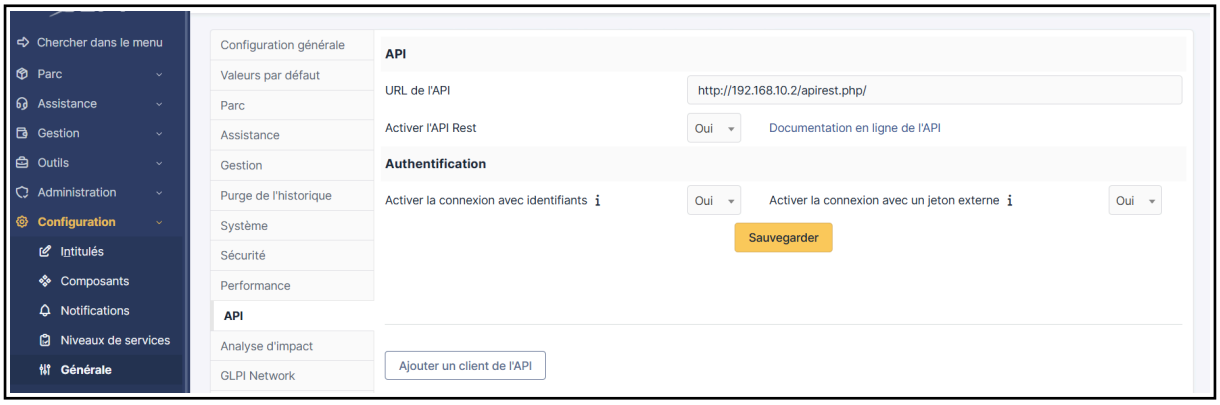


Après avoir finis l'installation, je me trouve sur le dashboard de GLPI et prêt à l'utiliser

2.4. Outil de Ticketing GLPI

2.4.1. Préparation de GLPI

- Activation de l'API : Dans Configuration > Générale > API, activation de l'accès REST.



- Génération des Tokens :
  - Création de « Client de l'API »
  - Création d'un App-Token pour identifier l'application Wazuh.

  - Récupération du User-Token pour l'utilisateur chargé de la création des tickets. (Administration > Utilisateur)

2.4.2. Développement du script d'intégration

- Création du script : Rédaction d'un script Python dans /var/ossec/integrations/custom-glpi.py.
- Gestion de l'API : Utilisation de la bibliothèque requests pour gérer l'authentification par sessions.
- Traitement des données : Extraction des champs (Niveau d'alerte, Description, Nom de l'agent) depuis le JSON envoyé par Wazuh.

```
#!/usr/bin/env python3
import sys
import json
import requests
import urllib3

urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

# --- CONFIGURATION ---
GLPI_URL = "http://192.168.10.2/apirest.php"

def main():
    if len(sys.argv) < 3:
        sys.exit(1)

    alert_file = sys.argv[1]
    try:
        keys = sys.argv[2].split(':')
        user_token = keys[0]
        app_token = keys[1]
    except IndexError:
        print("Erreur : Format api_key invalide dans ossec.conf")
        sys.exit(1)

    with open(alert_file) as f:
        alert_json = json.load(f)

    description = alert_json.get('rule', {}).get('description', 'Alerte Wazuh')
    level = alert_json.get('rule', {}).get('level', 0)
    agent = alert_json.get('agent', {}).get('name', 'Unknown')
    payload = {
        "input": {
            "name": f"[WAZUH] Niveau {level} - Agent: {agent}",
            "content": f"Alerte : {description}\nAgent : {agent}\nNiveau :
{level}",
            "urgency": 3,
            "status": 1
        }
    }

    # Utilisation des headers globaux pour toute la session
    with requests.Session() as s:
        s.headers.update({
            "Content-Type": "application/json",
            "Authorization": f"user_token {user_token}",
            "App-Token": app_token
        })

        try:
            # 1. Initialisation
            r_init = s.get(f"{GLPI_URL}/initSession", verify=False)
            res_init = r_init.json()

            if isinstance(res_init, dict) and 'session_token' in res_init:
                # 2. Ajout du token de session
```

```
                s.headers.update({"Session-Token": res_init['session_token']})

                # 3. Création du Ticket
                r_post = s.post(f"{GLPI_URL}/Ticket", json=payload, verify=False)
                print(f"Statut: {r_post.status_code}")
                print(f"Réponse: {r_post.text}")

                # 4. Fermeture
                s.get(f"{GLPI_URL}/killSession", verify=False)
            else:
                print(f"Erreur Auth: {res_init}")
        except Exception as e:
            print(f"Erreur Connexion: {str(e)}")

if __name__ == "__main__":
    main()
```

2.4.3. Configuration de Wazuh Manager

- Paramétrage de l'intégration : Modification du fichier ossec.conf pour déclarer le script.
- Définition des seuils : Choix d'un niveau d'alerte minimal (ex: Niveau 12) pour éviter la saturation de GLPI.

```
<integration>
  <name>custom-glpi.py</name>
  <api_key>
zP6rJPmeOYOe3wm5kJdHFptIyBVHewHMITSsc2ye:7rcJdNtIFLskSZpb7Rp3VBeCUjGDoYxrZ0Ebsdrn
</api_key>
  <level>10</level>
  <alert_format>json</alert_format>
</integration>
```

- Sécurisation : Attribution des droits d'exécution à l'utilisateur wazuh (chmod 755 et chown root:wazuh).

```
# Donne la propriété du script à root (propriétaire) et au groupe wazuh
sudo chown root:wazuh /var/ossec/integrations/custom-glpi.py

# Donne les droits de lecture et d'exécution au groupe wazuh
sudo chmod 750 /var/ossec/integrations/custom-glpi.py
```

2.5. Wazuh

2.5.1. Configuration et Installation

On va maintenant modifier le fichier ./config.yml

```
nodes:
  indexer:
    - name: node-1
      ip: "192.168.10.100"
  server:
    - name: wazuh-1
      ip: "192.168.10.100"
  dashboard:
    - name: dashboard
      ip: "192.168.10.100"
```

L'exécution du script wazuh-certs-tool.sh va nous permettre de créer un nouveau dossier wazuh-certificates dans le répertoire courant.Ce dossier contient les clés de chiffrement.

```
bash ./wazuh-certs-tool.sh -A
tar -cvf ./wazuh-certificates.tar -C ./wazuh-certificates/ .
rm -rf ./wazuh-certificates
```

Installation du paquet wazuh-indexer,wazuh-manager et wazuh-dashboard

```
sudo apt install wazuh-indexer wazuh-manager wazuh-dashboard -y
```

Nous allons a présent configurer notre indexer. nous allons juste renseigner le IP de notre serveur et le nom de l'indexer si nécessaires

```
nano /etc/wazuh-indexer/opensearch.yml
      network.host: "192.168.10.100"
```

Déploiement des certificats de l'indexer

```
NODE_NAME=node-1 mkdir /etc/wazuh-indexer/certs tar -xf ./wazuh-certificates.tar
-C /etc/wazuh-indexer/certs/ ./$NODE_NAME.pem ./$NODE_NAME-key.pem ./admin.pem ./
admin-key.pem ./root-ca.pem mv -n /etc/wazuh-indexer/certs/$NODE_NAME.pem /etc/
wazuh-indexer/certs/indexer.pem mv -n /etc/wazuh-indexer/certs/$NODE_NAME-
key.pem /etc/wazuh-indexer/certs/indexer-key.pem chmod 500 /etc/wazuh-indexer/
certs chmod 400 /etc/wazuh-indexer/certs/* chown -R wazuh-indexer:wazuh-indexer /
etc/wazuh-indexer/certs
```

Nous allons activer au démarrage le service wazuh-indexer, initier le cluster et puis le tester.

```
systemctl daemon-reload
systemctl enable wazuh-indexer
systemctl start wazuh-indexer

/usr/share/wazuh-indexer/bin/indexer-security-init.sh

curl -k -u admin:admin https://192.168.10.100:9200 curl -k -u admin:admin
https://192.168.10.100:9200/_cat/nodes?v
```

Configuration du (wazuh-manager). Redémarrage et vérification de l'état du service wazuh-manager

```
systemctl daemon-reload
systemctl enable wazuh-manager
systemctl start wazuh-manager
systemctl status wazuh-manager
```

Configuration du ficher filebeat.

```
apt install filebeat #Installation du paquet filebeat
curl -so /etc/filebeat/filebeat.yml https://packages.wazuh.com/4.14/tpl/wazuh/
filebeat/filebeat.yml #Téléchargement du fichier de confuguration filebeat

nano /etc/filebeat/filebeat.yml #Edition du fichier de configuration
      hosts: ["192.168.10.100:9200"]

filebeat keystore create #Création du fichier Keystore
```

Ajout de l'utilisateur et du mot de passe par défaut.

```
echo admin | filebeat keystore add username --stdin --force
echo admin | filebeat keystore add password --stdin --force
```

Téléchargement d'un modèle d'alerte

```
curl -so /etc/filebeat/wazuh-template.json https://raw.githubusercontent.com/
wazuh/wazuh/v4.14.2/extensions/elasticsearch/7.x/wazuh-template.json
chmod go+r /etc/filebeat/wazuh-template.json
```

Installation des modules supplémentaire filebeat.

```
curl -s https://packages.wazuh.com/4.x/filebeat/wazuh-filebeat-0.3.tar.gz | tar -
xvz -C /usr/share/filebeat/module
```

Déploiement des certificats.

```
NODE_NAME=node-1 mkdir /etc/filebeat/certs tar -xf ./wazuh-certificates.tar -C /
etc/filebeat/certs/ ./$NODE_NAME.pem ./$NODE_NAME-key.pem ./root-ca.pem mv -n /
etc/filebeat/certs/$NODE_NAME.pem /etc/filebeat/certs/filebeat.pem mv -n /etc/
```

```
filebeat/certs/$NODE_NAME-key.pem /etc/filebeat/certs/filebeat-key.pem chmod
500 /etc/filebeat/certs chmod 400 /etc/filebeat/certs/* chown -R root:root /etc/
filebeat/certs
```

Activation et redémarrage du service filebeat

```
systemctl daemon-reload
systemctl enable filebeat
systemctl start filebeat
```

Configuration du tableau de bord (dashbord). Nous allons modifier la variable opensearch.hosts: https://192.168.10.100:9200

```
nano /etc/wazuh-dashboard/opensearch_dashboards.yml
opensearch.hosts: https://192.168.10.100:9200
```

Déploiement des certificats du tableau de bord wazuh

```
NODE_NAME=node-1 mkdir /etc/wazuh-dashboard/certs tar -xf ./wazuh-
certificates.tar -C /etc/wazuh-dashboard/certs/ ./$NODE_NAME.pem ./$NODE_NAME-
key.pem ./root-ca.pem mv -n /etc/wazuh-dashboard/certs/$NODE_NAME.pem /etc/wazuh-
dashboard/certs/dashboard.pem mv -n /etc/wazuh-dashboard/certs/$NODE_NAME-
key.pem /etc/wazuh-dashboard/certs/dashboard-key.pem chmod 500 /etc/wazuh-
dashboard/certs chmod 400 /etc/wazuh-dashboard/certs/* chown -R wazuh-
dashboard:wazuh-dashboard /etc/wazuh-dashboard/certs
```

Et on restart enfin une dernière fois.

```
systemctl daemon-reload
systemctl enable wazuh-dashboard
systemctl start wazuh-dashboard
```

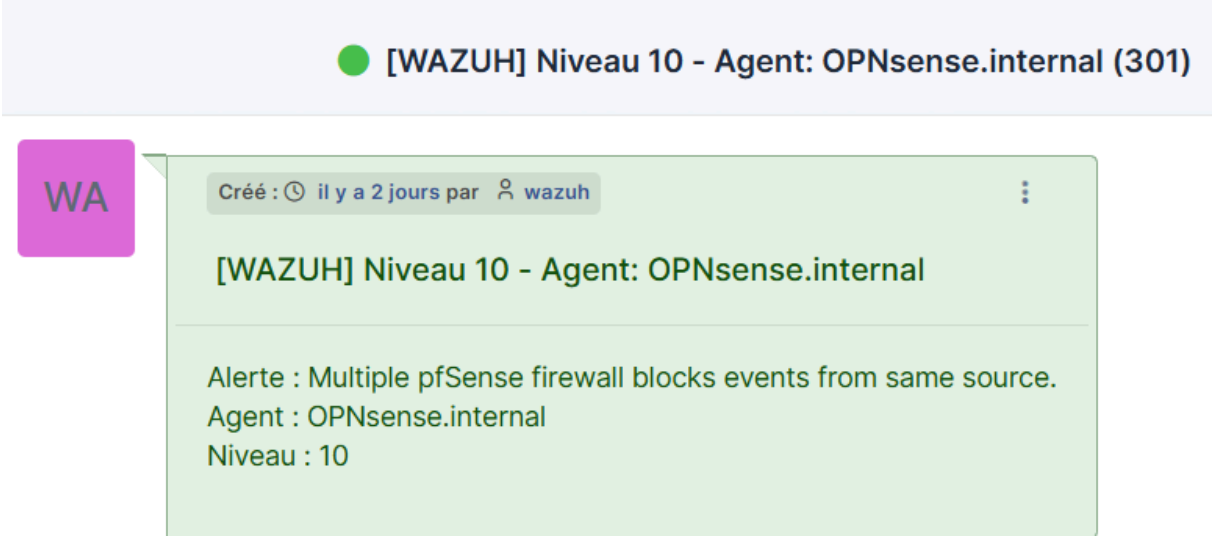
3. Test Finale

3.1. Simulation d'une attaque par brute-force

```
#Commande pour simuler l'attaque
for i in {1..20}; do    logger -p auth.info "sshd[1234]: Failed password for root
from 10.0.0.50 port 5678 ssh2"; done
```

3.2. Résultat dans GLPI

Le résultat après la commande ai été faites



Les tickets remonte bien dans GLPI


Accueil / Assistance / Tickets
Ajouter
Rechercher
Listes
Gabarits
Kanban global
Tickets attendant votre validation
807
Tickets
807
Tickets entrants
0
Tickets en attente
0
Tickets assignés
0
Tickets planifiés

Caractéristiques - Statut
est
Nouveau
règle
règle globale
groupe
Rechercher