RÉPUBLIQUE DU SÉNÉGAL
UN PEUPLE-UN BUT-UNE FOI
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
DIRECTION GÉNÉRALE DE L'ENSEIGNEMENT SUPÉRIEUR

RAPPORT DE SUPERVISION AVANCÉE D'UN SYSTÈME D'INFORMATION

Présenté par :
Nick Alix AIGOUEKE BEKALE

Supervisé par :
M. Mbaye SAMB

TABLES DE MATIERES

TP1 : Questions de cours

Objectif :

L'objectif de ce TP est de comprendre les concepts fondamentaux liés à la supervision avancée d'un système d'information et au fonctionnement d'un SIEM (Security Information and Event Management). Il vise à définir la supervision avancée, expliquer son importance, décrire ce qu'est un SIEM avec ses entités, et illustrer son fonctionnement à travers un schéma.

Architecture

- **Système d'information supervisé** : Ensemble des ressources informatiques (serveurs, applications, réseaux, bases de données) à surveiller.

- **Outils de supervision** : Logiciels ou agents (ex. : Wazuh, Nagios) déployés pour collecter des données sur les performances, les événements, et les incidents.

- **SIEM (Security Information and Event Management)** :

    - **Composants principaux** :
        - **Collecteurs/Agents** : Collectent les logs et événements (ex. : Wazuh Agent sur les hôtes).
        - **Serveur SIEM** : Centralise, normalise, et analyse les données (ex. : Wazuh Manager).
        - **Base de données/Indexer** : Stocke les logs et événements pour analyse (ex. : Wazuh Indexer).
        - **Interface utilisateur** : Permet de visualiser les alertes et rapports (ex. : Wazuh Dashboard sur https://localhost:443).
    - **Flux de données** :
        - Les agents collectent les logs (système, réseau, applications).
        - Les données sont envoyées au serveur SIEM pour corrélation et analyse.
        - Les alertes sont générées et affichées dans l'interface utilisateur.

- **Système de notification** : Intégration avec des outils comme Slack pour envoyer des alertes en temps réel.

1. Donner la définition de la supervision avancée d'un système d'information.

La supervision avancée d'un système d'information (SI) consiste à surveiller, analyser et gérer en temps réel les performances, la sécurité et les événements d'un SI. Elle utilise des outils automatisés (comme des SIEM) pour collecter, corréler et analyser les données issues des équipements, applications et réseaux, afin de détecter des anomalies, des incidents de sécurité ou des dysfonctionnements, et d'y répondre pro activement. Elle intègre des fonctionnalités comme l'analyse prédictive, la corrélation d'événements et la gestion des alertes.

2. Expliquer l'importance de la supervision d'un système d'information.

La supervision d'un SI est cruciale pour :

- Assurer la disponibilité : Détecter et résoudre rapidement les pannes pour minimiser les interruptions de service.
- Renforcer la sécurité : Identifier les menaces (intrusions, malwares) via l'analyse des logs et des comportements anormaux.
- Optimiser les performances : Surveiller l'utilisation des ressources (CPU, mémoire, réseau) pour garantir une efficacité optimale.
- Conformité réglementaire : Fournir des rapports d'audit pour répondre aux exigences légales (RGPD, ISO 27001).
- Prise de décision : Fournir des tableaux de bord et des analyses pour anticiper les besoins d'évolution du SI.

3. Donner la définition et les entités d'un SIEM.

Définition : Un SIEM (Security Information and Event Management) est une solution qui combine la gestion des informations de sécurité (SIM) et la gestion des événements de

sécurité (SEM). Il collecte, stocke, corrèle et analyse les logs et événements provenant de diverses sources dans un SI pour détecter les incidents de sécurité et faciliter leur réponse.

Entités principales :

- Sources de données : Équipements (serveurs, pare-feu, routeurs), applications, bases de données, systèmes de détection d'intrusion (IDS/IPS).
- Agents : Logiciels installés sur les systèmes surveillés pour collecter et transmettre les logs.
- Collecteur de logs : Module centralisant les données brutes.
- Moteur de corrélation : Analyse et corrèle les événements pour identifier les menaces.
- Base de données : Stockage des logs et des événements pour analyse et audit.
- Interface utilisateur : Tableau de bord pour visualiser les alertes, rapports et analyses.

4. Expliquer le fonctionnement d'un SIEM avec un schéma à l'appui.

Fonctionnement :

- **Collecte :** Les agents ou connecteurs récupèrent les logs (syslog, journaux Windows, etc.) des sources du SI.
- **Normalisation :** Les données brutes sont formatées pour être exploitables.
- **Corrélation :** Le moteur analyse les événements en temps réel, applique des règles pour détecter des schémas suspects (ex. : multiples tentatives de connexion échouées).
- **Alerte :** Les incidents détectés génèrent des notifications (email, SMS, intégrations comme Slack).
- **Stockage et analyse :** Les logs sont archivés pour des analyses forensic ou des audits.
- Visualisation : Les résultats sont affichés via des tableaux de bord (graphiques, alertes).

TP2 : Mise en place d'un SIEM avec Wazuh

Objectif

L'objectif de ce TP est de mettre en place un système SIEM avec Wazuh pour superviser un environnement informatique. Cela inclut la compréhension de l'architecture détaillée de Wazuh et de ses entités, l'installation et la configuration de Wazuh sur un serveur Ubuntu 20, l'intégration de deux machines en tant qu'agents (Windows 10 et Ubuntu 20), et l'analyse des résultats obtenus via l'interface Wazuh pour valider le bon fonctionnement de la supervision.

1. Donner l'architecture détaillée de Wazuh et ses entités.

Architecture de Wazuh :

Wazuh est une plateforme open-source de sécurité et de supervision qui combine des fonctionnalités de SIEM et de HIDS (Host Intrusion Detection System). Son architecture est modulaire et repose sur trois composants principaux :

- **Wazuh Server :** Gère la collecte, l'analyse et le stockage des données. Il inclut :
- **Manager :** Centralise la gestion des agents et applique les règles de corrélation.
- **Filebeat :** Transfère les données vers Elasticsearch.
- **API RESTful :** Permet l'interaction avec le serveur.
- **Wazuh Agent :** Logiciel léger installé sur les systèmes surveillés (Windows, Linux, etc.) pour collecter les logs, surveiller les fichiers, détecter les anomalies et exécuter des commandes de réponse.
- **Elastic Stack :** Intégré pour le stockage, l'analyse et la visualisation :
- **Elasticsearch :** Stocke et indexe les logs.
- **Kibana ou Wazuh Dashboard :** Interface web pour visualiser les alertes, rapports et tableaux de bord.

Entités :

- Agents (installés sur les hôtes).
- Manager (serveur central).
- Base de données Elasticsearch.
- Interface Kibana.

2. Installer et configurer Wazuh sur Ubuntu 20.

- **Installer les dépendances :**

apt install curl apt-transport-https lsb-release gnupg2 -y

```
root@nospi-visiotech:/home/nospi# apt install curl apt-transport-https lsb-release gnupg2 -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
curl est déjà la version la plus récente (8.5.0-2ubuntu10.6).
apt-transport-https est déjà la version la plus récente (2.7.14build2).
lsb-release est déjà la version la plus récente (12.0-2).
lsb-release passé en « installé manuellement ».
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
```

- **Ajouter le dépôt Wazuh :**

curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -

echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list

sudo apt update

```
root@nospi-visiotech:/home/nospi# curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -
   echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
deb https://packages.wazuh.com/4.x/apt/ stable main
root@nospi-visiotech:/home/nospi# apt update
```

- **Installer Wazuh Manager :**

apt install wazuh-manager -y

```
root@nospi-visiotech:/home/nospi# apt install wazuh-manager -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  docker-ce-rootless-extras slirp4netns
```

systemctl enable wazuh-manager

systemctl start wazuh-manager

```
root@nospi-visiotech:/home/nospi# systemctl enable wazuh-manager
root@nospi-visiotech:/home/nospi# systemctl start wazuh-manager
root@nospi-visiotech:/home/nospi#
```

```
root@nospi-visiotech:/home/nospi# systemctl status wazuh-manager
● wazuh-manager.service - Wazuh manager
     Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled; preset: enabled)
     Active: active (running) since Mon 2025-05-05 22:17:38 GMT; 29s ago
    Process: 53886 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 166 (limit: 14119)
     Memory: 1.6G (peak: 1.6G)
        CPU: 1min 27.806s
     CGroup: /system.slice/wazuh-manager.service
             ├─53948 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─53949 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─53950 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─53953 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─53956 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─53997 /var/ossec/bin/wazuh-authd
             ├─54010 /var/ossec/bin/wazuh-db
```

- **Installer Elastic Stack :**

Ajouter le dépôt Elastic :

curl -s https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list

sudo apt update

```
root@nospi-visiotech:/home/nospi# curl -s https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
     echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-7.x.list
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
deb https://artifacts.elastic.co/packages/7.x/apt stable main
root@nospi-visiotech:/home/nospi# apt update
Atteint :1 http://sn.archive.ubuntu.com/ubuntu noble InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu noble-updates InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu noble-backports InRelease
Atteint :4 http://security.ubuntu.com/ubuntu noble-security InRelease
```

Installer Elasticsearch :

apt install elasticsearch -y

```
root@nospi-visiotech:/home/nospi# apt install elasticsearch -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
  docker-ce-rootless-extras slirp4netns
Veuillez utiliser « apt autoremove » pour les supprimer.
Les NOUVEAUX paquets suivants seront installés :
  elasticsearch
0 mis à jour, 1 nouvellement installés, 0 à enlever et 161 non mis à jour.
Il est nécessaire de prendre 325 Mo dans les archives.
Après cette opération, 542 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 elasticsearch amd64 7.17.28 [325 MB]
14% [1 elasticsearch 55,1 MB/325 MB 17%]
```

systemctl enable elasticsearch

systemctl start elasticsearch

systemctl status elasticsearch

```
root@nospi-visiotech:/home/nospi# systemctl daemon-reload
root@nospi-visiotech:/home/nospi# systemctl enable elasticsearch.service
Synchronizing state of elasticsearch.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable elasticsearch
Created symlink /etc/systemd/system/multi-user.target.wants/elasticsearch.service → /usr/lib/systemd/system/elasticsearch.service.
root@nospi-visiotech:/home/nospi# systemctl start elasticsearch.service
```

```
root@nospi-visiotech:/home/nospi# systemctl status elasticsearch.service
● elasticsearch.service - Elasticsearch
     Loaded: loaded (/usr/lib/systemd/system/elasticsearch.service; enabled; preset: enabled)
     Active: active (running) since Mon 2025-05-05 22:29:00 GMT; 9s ago
       Docs: https://www.elastic.co
   Main PID: 56670 (java)
      Tasks: 68 (limit: 14119)
     Memory: 6.2G (peak: 6.2G)
        CPU: 2min 49.735s
     CGroup: /system.slice/elasticsearch.service
             ├─56670 /usr/share/elasticsearch/jdk/bin/java -Xshare:auto -Des.networkaddress.cache.ttl=60 -Des.networka
             └─56860 /usr/share/elasticsearch/modules/x-pack-ml/platform/linux-x86_64/bin/controller

mai 05 22:26:39 nospi-visiotech systemd[1]: Starting elasticsearch.service - Elasticsearch...
mai 05 22:27:01 nospi-visiotech systemd-entrypoint[56670]: mai 05, 2025 10:27:01 PM sun.util.locale.provider.LocalePr
mai 05 22:27:01 nospi-visiotech systemd-entrypoint[56670]: WARNING: COMPAT locale provider will be removed in a future
```

- Installer Filebeat :

apt install filebeat -y

```
root@nospi-visiotech:/home/nospi# apt install filebeat -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets suivants ont été installés automatiquement et ne sont plus nécessaires :
```

curl -so /etc/filebeat/filebeat.yml https://packages.wazuh.com/4.x/filebeat/filebeat.yml

systemctl enable filebeat

systemctl start filebeat

```
root@nospi-visiotech:/home/nospi# curl -so /etc/filebeat/filebeat.yml https://packages.wazuh.com/4.x/filebeat/filebeat.yml
systemctl enable filebeat
systemctl start filebeat
Synchronizing state of filebeat.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable filebeat
root@nospi-visiotech:/home/nospi#
```

- **Installer Kibana :**

sudo apt install kibana=7.17.13 -y

```
root@nospi-visiotech:/tmp# apt install kibana=7.17.13
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  kibana
0 mis à jour, 1 nouvellement installés, 0 à enlever et 162 non mis à jour.
Il est nécessaire de prendre 307 Mo dans les archives.
Après cette opération, 826 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 kibana amd64 7.17.13 [307 MB]
28% [1 kibana 106 MB/307 MB 35%]                                                4 486 kB/s 44s
```

systemctl enable kibana

systemctl start kibana

```
root@nospi-visiotech:/home/nospi#
root@nospi-visiotech:/home/nospi# systemctl enable kibana
Synchronizing state of kibana.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable kibana
Created symlink /etc/systemd/system/multi-user.target.wants/kibana.service → /etc/systemd/system/kibana.service.
root@nospi-visiotech:/home/nospi# systemctl start kibana
root@nospi-visiotech:/home/nospi#
```

Installer le plugin wazuh kibana

```
root@nospi-visiotech:/tmp# sudo /usr/share/kibana/bin/kibana-plugin install https://packages.wazuh.com/4.x/ui/kibana/wazuh_kibana-4.5.4_7.17.13-1.z
ip
Attempting to transfer from https://packages.wazuh.com/4.x/ui/kibana/wazuh_kibana-4.5.4_7.17.13-1.zip
Transferring 36404504 bytes....................
Transfer complete
Retrieving metadata from plugin archive
Extracting plugin archive
Extraction complete
Plugin installation complete
root@nospi-visiotech:/tmp#
```

- Configurer Wazuh avec Elastic** :

Modifier /etc/filebeat/filebeat.yml pour inclure l'adresse du serveur Elasticsearch.

```
root@nospi-visiotech: /home/nospi (ssh)  ⌘1
filebeat.inputs:
- type: log
  enabled: true
  paths:
    - /var/ossec/logs/alerts/alerts.json

output.elasticsearch:
  hosts: ["http://localhost:9200"]

setup.kibana:
  host: "http://localhost:5601"
```

- Redémarrer les services :

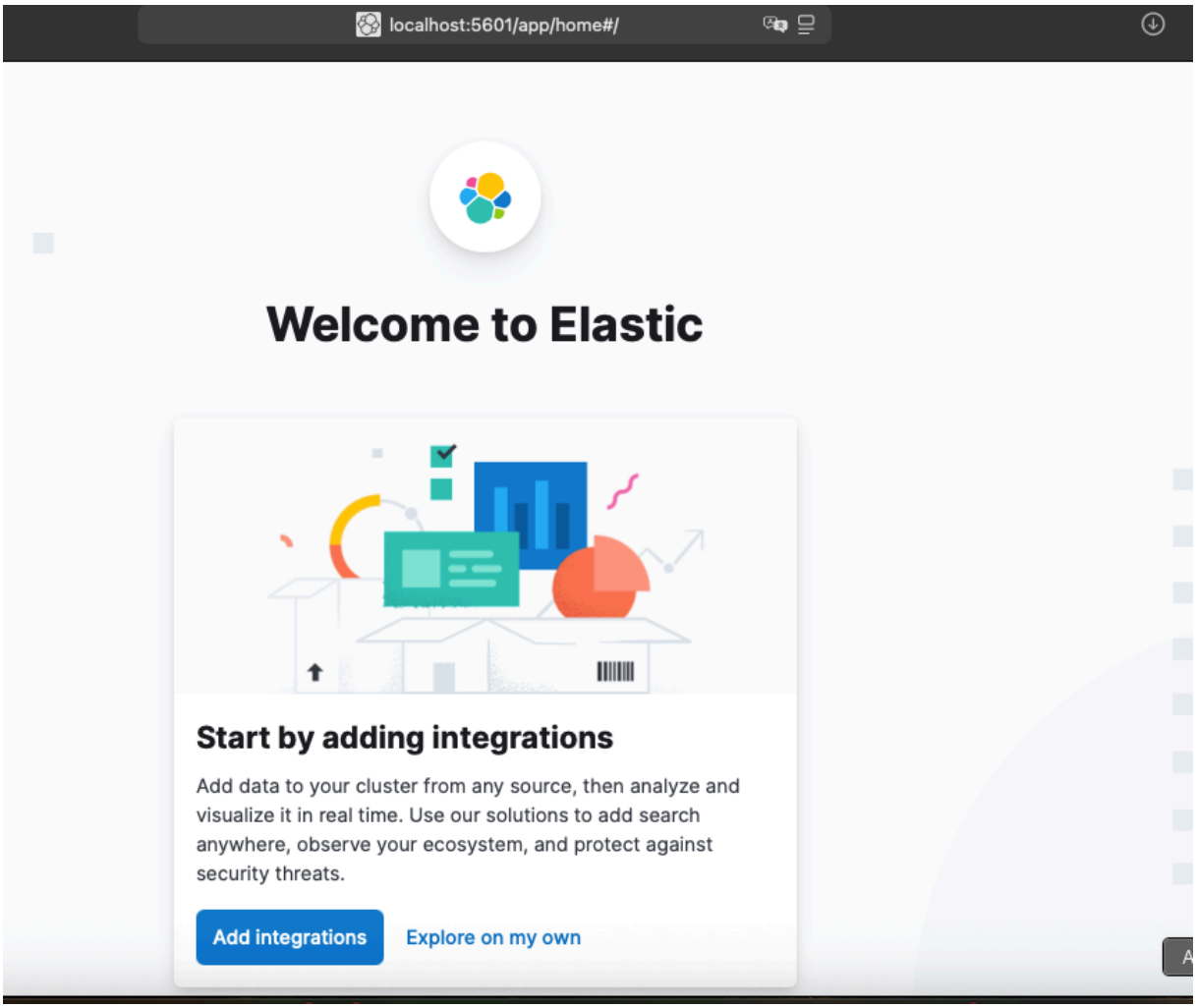
systemctl restart wazuh-manager filebeat kibana

```
root@nospi-visiotech: /home/nospi (ssh)  ⌘1  nospi@nospi-visiot
root@nospi-visiotech:/home/nospi# systemctl restart wazuh-manager filebeat kibana
root@nospi-visiotech:/home/nospi#
```

systemctl restart wazuh-manager filebeat kibana

```
root@nospi-visiotech:/home/nospi# systemctl status wazuh-manager filebeat kibana
● wazuh-manager.service - Wazuh manager
     Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled; preset: enabled)
     Active: active (running) since Mon 2025-05-05 23:24:47 GMT; 1min 47s ago
    Process: 61805 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 171 (limit: 14119)
     Memory: 1.7G (peak: 1.7G)
        CPU: 2min 9.405s
     CGroup: /system.slice/wazuh-manager.service
             ├─61901 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─61902 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─61903 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─61906 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
```

- Accéder à Kibana :
- Ouvrir un navigateur : http://localhost:5601



3. Intégrer deux machines en tant qu'agents (Windows 10 et Ubuntu 20).

Sur le serveur Wazuh :

- Générer une clé pour les agents :

/var/ossec/bin/manage_agents

- Choisir `A` pour ajouter un agent, entrer un nom (ex. : `win10-agent`, `ubuntu-agent`) et l'IP.

```
root@nospi-visiotech:/home/nospi# /var/ossec/bin/manage_agents


****************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: A

- Adding a new agent (use '\q' to return to the main menu).
  Please provide the following:
   * A name for the new agent: win10-agent
   * The IP Address of the new agent: 192.168.1.50
Confirm adding it?(y/n): y
Agent added with ID 001.


****************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: A
```

```
Choose your action: A,E,L,R or Q: A

- Adding a new agent (use '\q' to return to the main menu).
  Please provide the following:
   * A name for the new agent: ubuntu-agent
   * The IP Address of the new agent: 192.168.1.51
Confirm adding it?(y/n): y
Agent added with ID 002.


****************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: Q

manage_agents: Exiting.
root@nospi-visiotech:/home/nospi#
```

- Extraire la clé générée avec `E`.

```
root@nospi-visiotech:/home/nospi# /var/ossec/bin/manage_agents


****************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: E

Available agents:
   ID: 001, Name: win10-agent, IP: 192.168.1.50
   ID: 002, Name: ubuntu-agent, IP: 192.168.1.51
Provide the ID of the agent to extract the key (or '\q' to quit): 001

Agent key information for '001' is:
MDAxIHdpbjEwLWFnZW50IDE5Mi4xNjguMS41MCAxN2JiOWY5YjExNDhkNThmOGMzMTBiNTNmZWQzNTE5MTgzYzdjNDI4MDJjYmIwNTI1MGJhMjQzYjQ2YmVlNmVk

** Press ENTER to return to the main menu.
```

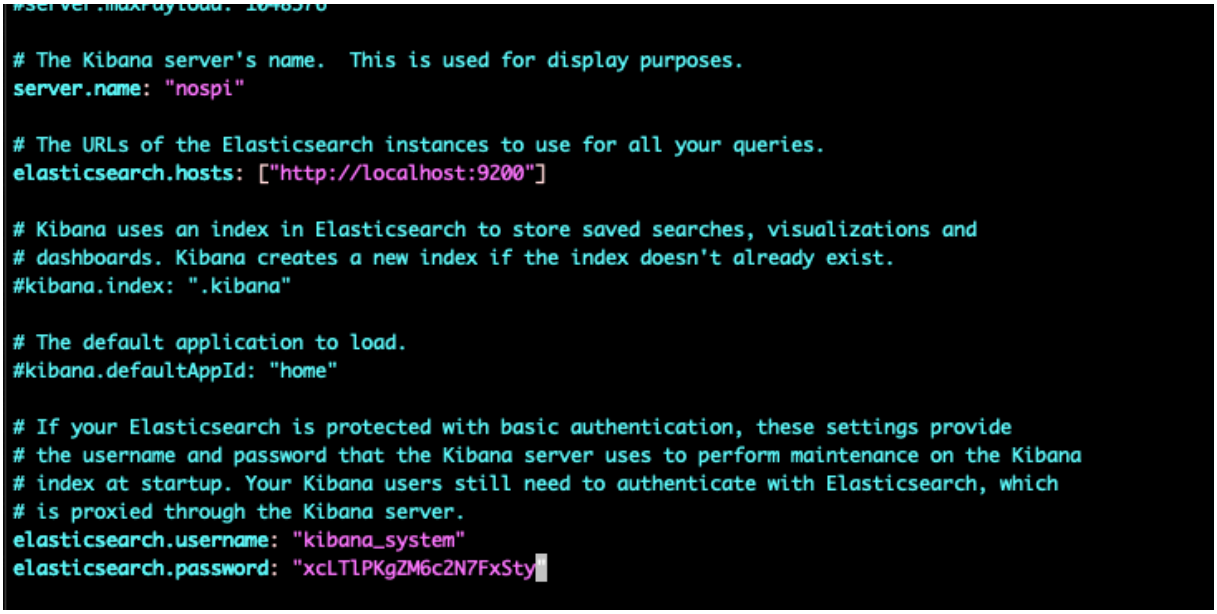
```
****************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
****************************************
   (A)dd an agent (A).
   (E)xtract key for an agent (E).
   (L)ist already added agents (L).
   (R)emove an agent (R).
   (Q)uit.
Choose your action: A,E,L,R or Q: E

Available agents:
   ID: 001, Name: win10-agent, IP: 192.168.1.50
   ID: 002, Name: ubuntu-agent, IP: 192.168.1.51
Provide the ID of the agent to extract the key (or '\q' to quit): 002

Agent key information for '002' is:
MDAyIHVidW50dS1hZ2VudCAxOTIuMTY4LjEuNTEgNzIzYTc1ZDRmYTY4NWJiY2Y3ZWQ0ZWE0NTgyNGNkNzJhOGViMDkzZDk4MzY1NGM3NjU5YWQ5NTczM2VhZDg2MA==

** Press ENTER to return to the main menu.
```

- On récupère le mot de passe des utilisateurs

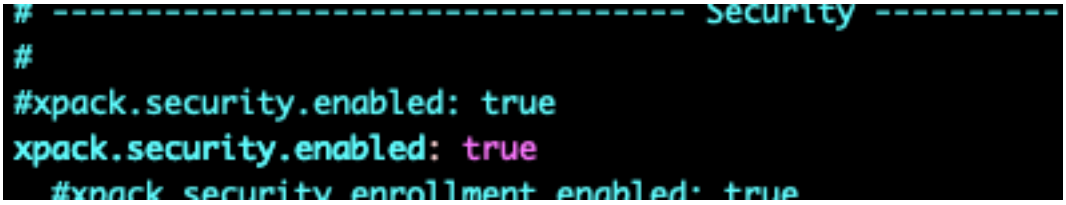
```
root@nospi-visiotech:/home/nospi# sudo /usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logstash_system,beats_system,remote_monitoring_user.
The passwords will be randomly generated and printed to the console.
Please confirm that you would like to continue [y/N]y


Changed password for user apm_system
PASSWORD apm_system = sZxQesG2U9obXDVNZsGp

Changed password for user kibana_system
PASSWORD kibana_system = xcLTlPKgZM6c2N7FxSty

Changed password for user kibana
PASSWORD kibana = xcLTlPKgZM6c2N7FxSty

Changed password for user logstash_system
PASSWORD logstash_system = 86KbilivVfJYXTXd9pqt

Changed password for user beats_system
PASSWORD beats_system = eFULV5rhaglnJaPv4m1s

Changed password for user remote_monitoring_user
PASSWORD remote_monitoring_user = Puyzd5VGKetjjj8HU4dG

Changed password for user elastic
PASSWORD elastic = PFcBH1ujIlPG9HO0GdjP

root@nospi-visiotech:/home/nospi#
```

Rappels

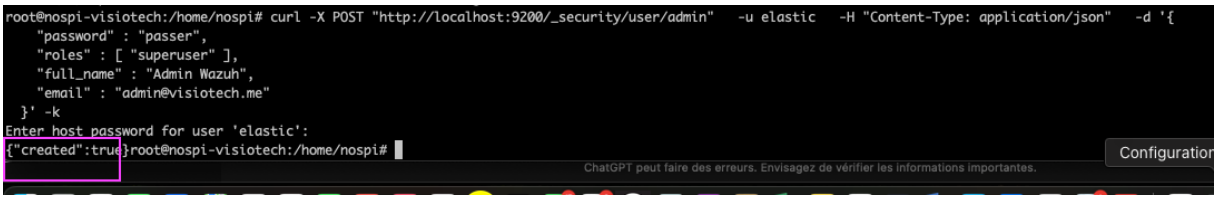
On édite le fichier /etc/kibana/kibana.yml pour inclure l'utilisateur kybana system



On active la security dans elasticsearch dans /etc/elasticsearch/elasticsearch.yml

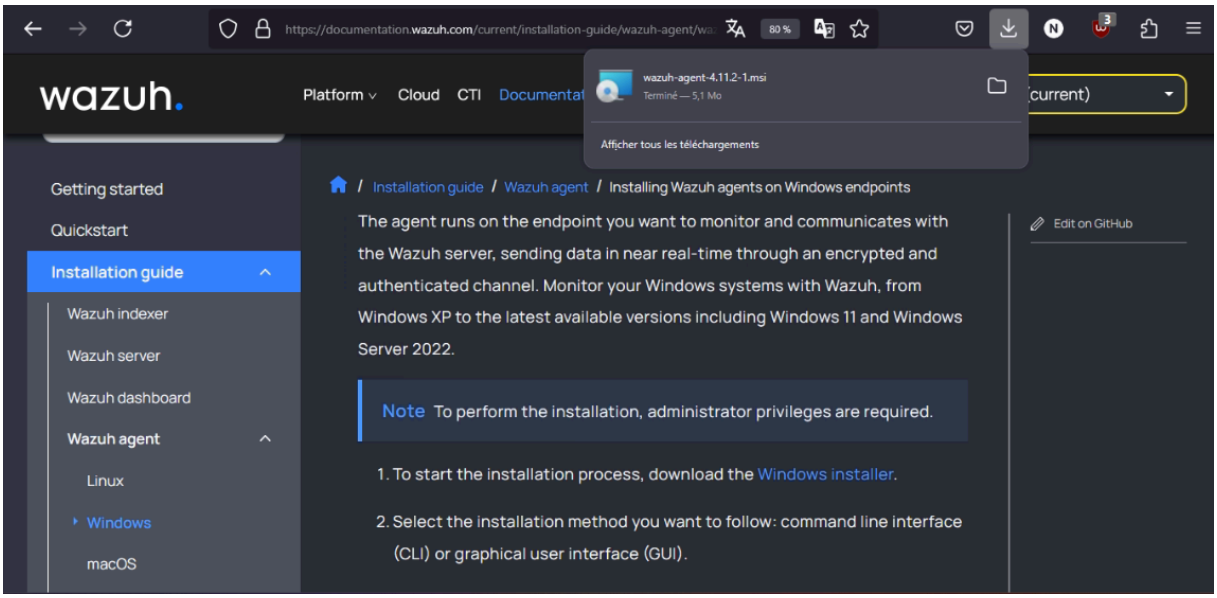


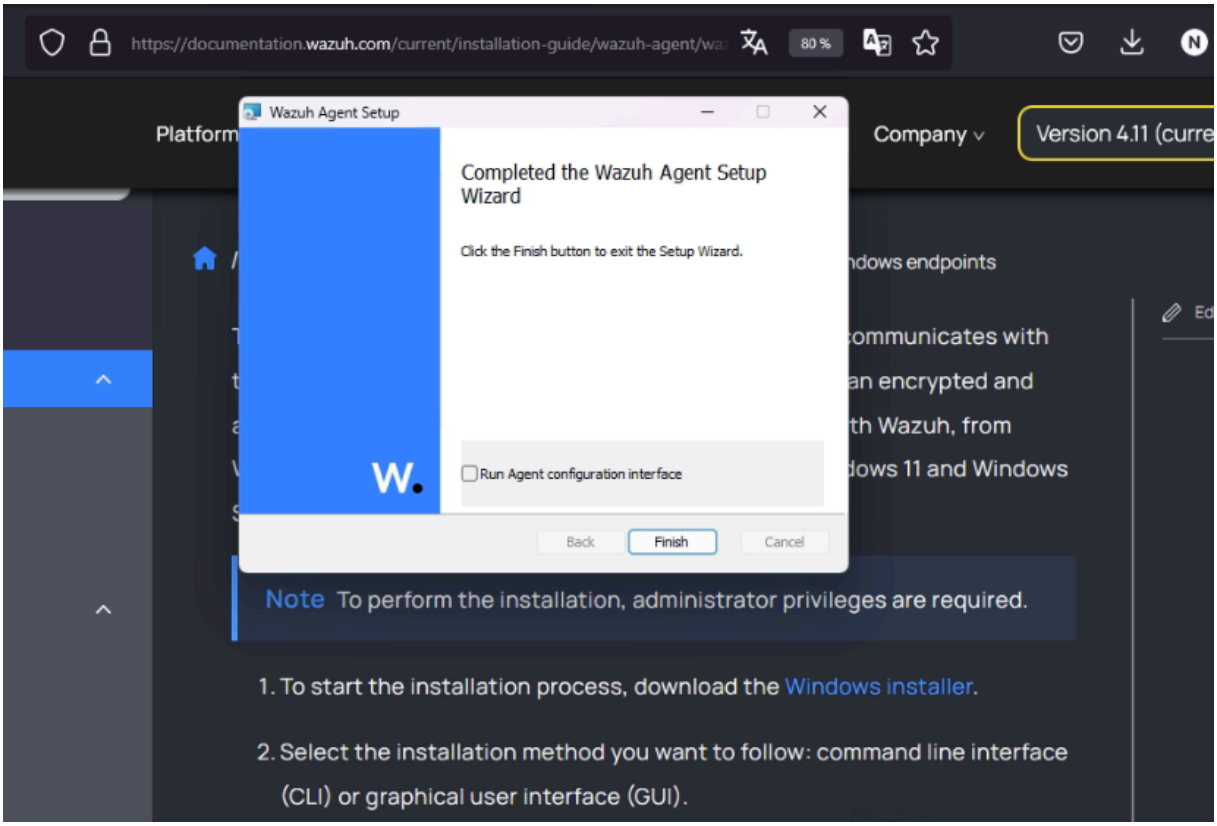
On créer l'utilisateur admin et on s'aisit le mot de passe de l'utlisateur elastic



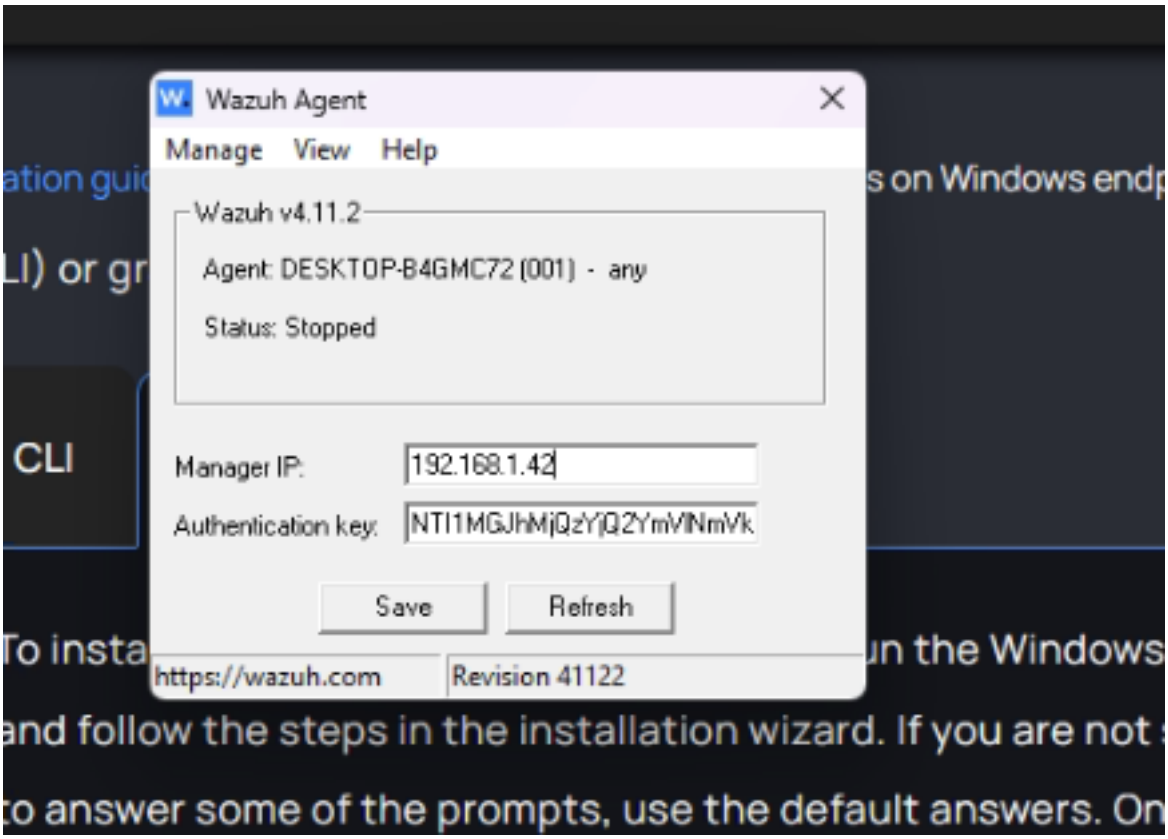
Sur Windows 10 :

Télécharger l'agent Wazuh : `https://packages.wazuh.com/4.x/windows/wazuh-agent-4.x.x.msi`.

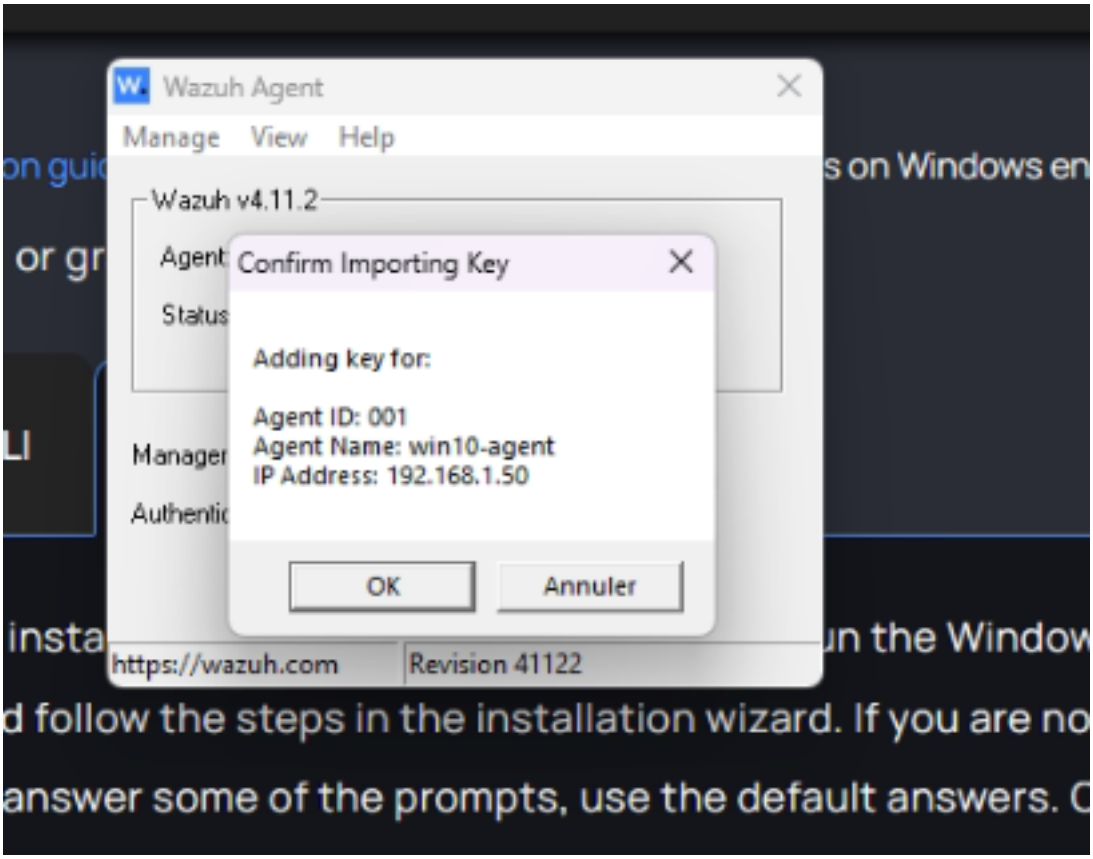




Configurer via l'interface graphique :



On clique sur save



Sur Ubuntu 20 :

- Ajouter le dépôt Wazuh :

curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -

echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list

sudo apt update

```
root@nospi-server:/home/nospi# curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -
OK
root@nospi-server:/home/nospi#    echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/s
ources.list.d/wazuh.list
deb https://packages.wazuh.com/4.x/apt/ stable main
root@nospi-server:/home/nospi#
root@nospi-server:/home/nospi# apt update
Atteint :1 http://sn.archive.ubuntu.com/ubuntu focal InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu focal-updates InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu focal-backports InRelease
Atteint :4 http://sn.archive.ubuntu.com/ubuntu focal-security InRelease
```

- Installer l'agent :

sudo apt install wazuh-agent -y

```
root@nospi-server:/home/nospi# apt install wazuh-agent -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  wazuh-agent
0 mis à jour, 1 nouvellement installés, 0 à enlever et 62 non mis à jour.
Il est nécessaire de prendre 11,1 Mo dans les archives.
Après cette opération, 39,7 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 https://packages.wazuh.com/4.x/apt stable/main amd64 wazuh-agent amd64 4.11.2-1 [11,1 MB]
11,1 Mo réceptionnés en 3s (3 195 ko/s)
Préconfiguration des paquets...
```

- Configurer l'agent :

sudo /var/ossec/bin/manage_agents

Choisir `I` et coller la clé.

```
root@nospi-server:/home/nospi# /var/ossec/bin/manage_agents


************************************
* Wazuh v4.11.2 Agent manager.         *
* The following options are available: *
************************************
   (I)mport key from the server (I).
   (Q)uit.
Choose your action: I or Q: I

* Provide the Key generated by the server.
* The best approach is to cut and paste it.
*** OBS: Do not include spaces or new lines.

Paste it here (or '\q' to quit): MDAyIHVidW50dS1hZ2VudCAxOTIuMTY4LjEuNTEgNzIzYTc1ZDRmYTY4NWJiY2Y3ZWQ0ZWE0NTgyNGNkN
zJhOGViMDkzZDk4MzY1NGM3NjU5YWQ5NTczM2VhZDg2MA==

Agent information:
   ID:002
   Name:ubuntu-agent
   IP Address:192.168.1.51

Confirm adding it?(y/n):
```

Modifier `/var/ossec/etc/ossec.conf` pour pointer vers l'IP du serveur :

```
<!--
  Wazuh - Agent - Default configuration for ubuntu 20.04
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <client>
    <server>
      <address>192.168.1.42</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>ubuntu, ubuntu20, ubuntu20.04</config-profile>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
    <crypto_method>aes</crypto_method>
  </client>
```

Redémarrer l'agent :

sudo systemctl restart wazuh-agent

```
root@nospi-server:/home/nospi# systemctl restart wazuh-agent
root@nospi-server:/home/nospi#
```

On vérifie l'intégration des deux machines dans le serveur Wazuh

```
root@nospi-visiotech:/usr/share/kibana# /var/ossec/bin/agent_control -l

Wazuh agent_control. List of available agents:
   ID: 000, Name: nospi-visiotech (server), IP: 127.0.0.1, Active/Local
   ID: 001, Name: win10-agent, IP: 192.168.1.50, Active
   ID: 002, Name: ubuntu-agent, IP: 192.168.1.51, Never connected
   ID: 003, Name: nospi-server, IP: any, Active

List of agentless devices:

root@nospi-visiotech:/usr/share/kibana#
```

Voir les agents sur les Dashboard

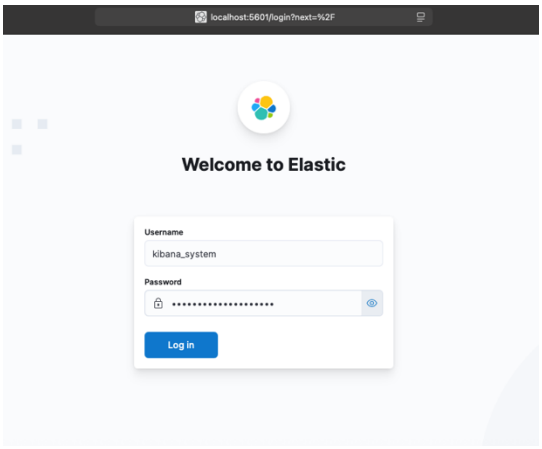
Pour le Dashboard kibana

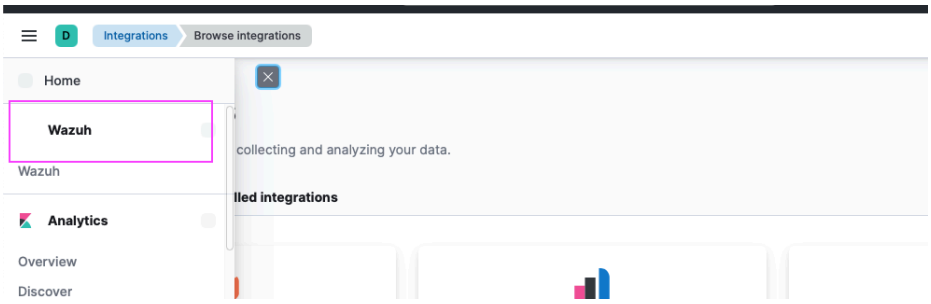
On installe le plugin Wazuh

```
root@nospi-visiotech:/usr/share/kibana# git clone https://github.com/wazuh/wazuh-dashboard-plugins.git
Clonage dans 'wazuh-dashboard-plugins'...
remote: Enumerating objects: 141702, done.
remote: Counting objects: 100% (2108/2108), done.
remote: Compressing objects: 100% (995/995), done.
remote: Total 141702 (delta 1554), reused 1185 (delta 1110), pack-reused 139594 (from 2)
Réception d'objets: 100% (141702/141702), 59.01 Mio | 2.31 Mio/s, fait.
Résolution des deltas: 100% (99335/99335), fait.
root@nospi-visiotech:/usr/share/kibana#
root@nospi-visiotech:/usr/share/kibana# cd wazuh-dashboard-plugins
root@nospi-visiotech:/usr/share/kibana/wazuh-dashboard-plugins#
```

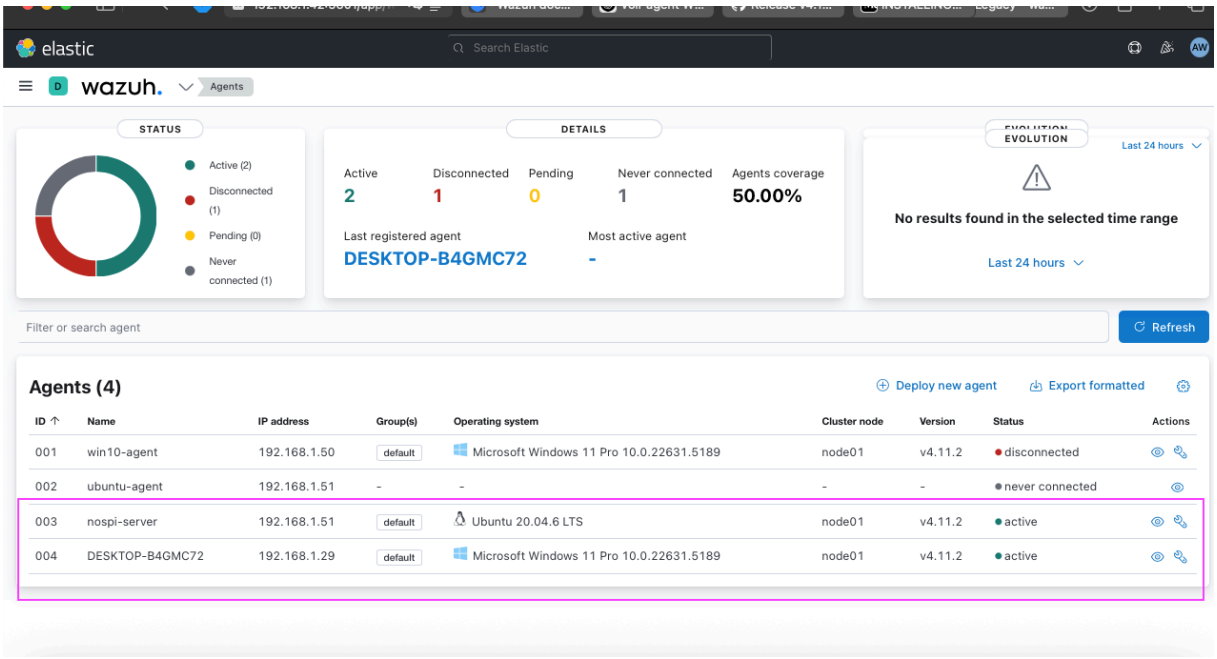
On édite wazuh.yml

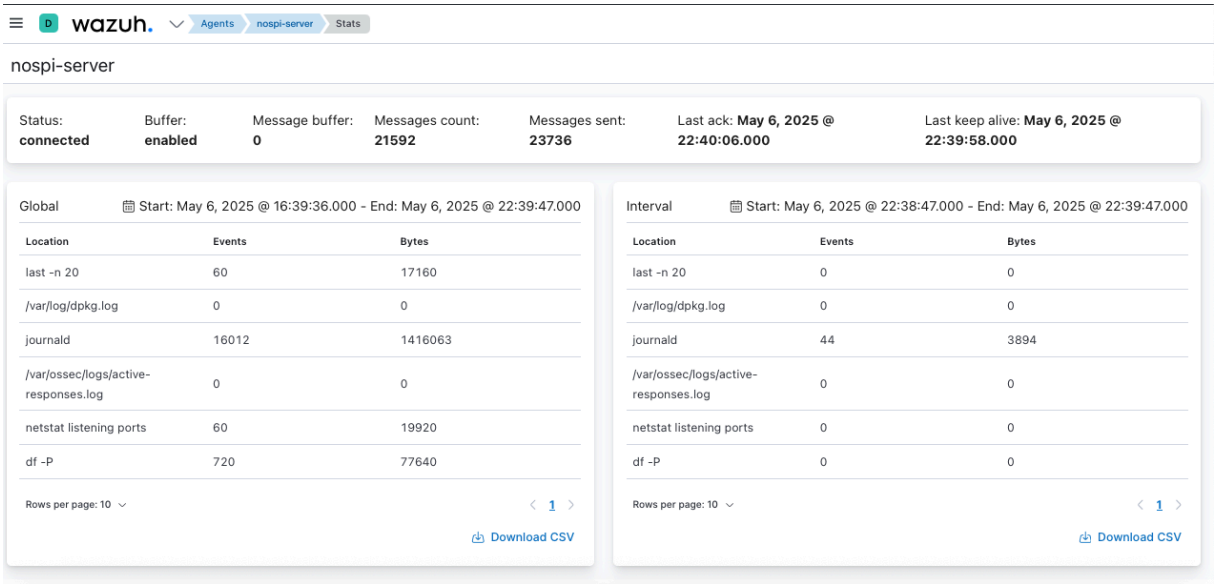
```
hosts:
  - default:
      url: https://192.168.1.42
      port: 55000
      username: wazuh-wui
      password: wazuh-wui
      run_as: false
```

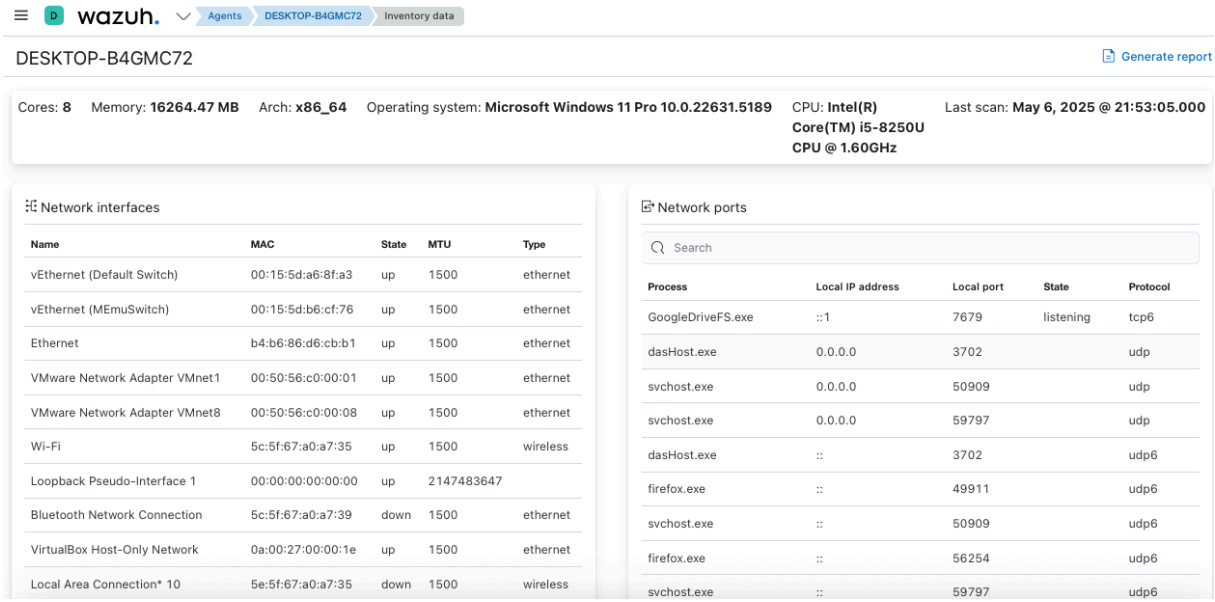




Connexion des agents : Dans Kibana, sous l'onglet « Wazuh » > « Agents », les deux agents (Windows 10 et Ubuntu 20) apparaissent avec le statut « Active ».







Pour le Dashboard Wazuh-Dashboard

On peut aussi le faire depuis wazuh-dashboard pour éviter kabana

On édite le fichier config.yml pour inclure l'ip du serveur

```
    - name: node-1
      ip: "192.168.1.42"
    #- name: node-2
    #  ip: "<indexer-node-ip>"
    #- name: node-3
    #  ip: "<indexer-node-ip>"

  # Wazuh server nodes
  # If there is more than one Wazuh server
  # node, each one must have a node_type
  server:
    - name: wazuh-1
      ip: "192.168.1.42"
    #  node_type: master
    #- name: wazuh-2
    #  ip: "<wazuh-manager-ip>"
    #  node_type: worker
    #- name: wazuh-3
    #  ip: "<wazuh-manager-ip>"
    #  node_type: worker

  # Wazuh dashboard nodes
  dashboard:
    - name: dashboard
      ip: "192.168.1.42"
```

On génère les fichiers de config

```
root@nospi-visiotech:~# bash wazuh-install.sh --generate-config-files
08/05/2025 01:43:12 INFO: Starting Wazuh installation assistant. Wazuh version: 4.11.2 (x86_64/AMD64)
08/05/2025 01:43:12 INFO: Verbose logging redirected to /var/log/wazuh-install.log
08/05/2025 01:43:34 INFO: Verifying that your system meets the recommended minimum hardware requirements.
08/05/2025 01:43:34 INFO: --- Configuration files ---
08/05/2025 01:43:34 INFO: Generating configuration files.
08/05/2025 01:43:36 INFO: Generating the root certificate.
08/05/2025 01:43:36 INFO: Generating Admin certificates.
08/05/2025 01:43:38 INFO: Generating Wazuh indexer certificates.
08/05/2025 01:43:39 INFO: Generating Filebeat certificates.
08/05/2025 01:43:40 INFO: Generating Wazuh dashboard certificates.
08/05/2025 01:43:42 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificates, and passwords necessary for installation.
root@nospi-visiotech:~#
```

On install wazuh-indexer

```
root@nospi-visiotech:~# bash wazuh-install.sh --wazuh-indexer node-1 --overwrite
08/05/2025 02:03:49 INFO: Starting Wazuh installation assistant. Wazuh version: 4.11.2 (x86_64/AMD64)
08/05/2025 02:03:49 INFO: Verbose logging redirected to /var/log/wazuh-install.log
08/05/2025 02:04:12 INFO: Verifying that your system meets the recommended minimum hardware requirements.
08/05/2025 02:04:43 INFO: Wazuh repository added.
08/05/2025 02:04:44 INFO: --- Wazuh indexer ---
08/05/2025 02:04:44 INFO: Starting Wazuh indexer installation.
08/05/2025 02:05:38 INFO: Wazuh indexer installation finished.
08/05/2025 02:05:38 INFO: Wazuh indexer post-install configuration finished.
08/05/2025 02:05:38 INFO: Starting service wazuh-indexer.
08/05/2025 02:06:38 INFO: wazuh-indexer service started.
08/05/2025 02:06:38 INFO: Initializing Wazuh indexer cluster security settings.
08/05/2025 02:06:45 INFO: Wazuh indexer cluster initialized.
08/05/2025 02:06:45 INFO: Installation finished.
root@nospi-visiotech:~#
```

On initilialise le cluster

```
root@nospi-visiotech:~# bash wazuh-install.sh --start-cluster
08/05/2025 02:07:41 INFO: Starting Wazuh installation assistant. Wazuh version: 4.11.2 (x86_64/AMD64)
08/05/2025 02:07:41 INFO: Verbose logging redirected to /var/log/wazuh-install.log
08/05/2025 02:08:03 INFO: Verifying that your system meets the recommended minimum hardware requirements.
08/05/2025 02:08:19 INFO: Wazuh indexer cluster security configuration initialized.
08/05/2025 02:08:31 INFO: Updating the internal users.
08/05/2025 02:08:41 INFO: A backup of the internal users has been saved in the /etc/wazuh-indexer/internalusers-backup folder.
08/05/2025 02:09:25 INFO: Wazuh indexer cluster started.
root@nospi-visiotech:~#
```

Exécutez la commande suivante pour confirmer que l'installation a réussi.

```
root@nospi-visiotech:~# tar -axf wazuh-install-files.tar wazuh-install-files/wazuh-passwords.txt -O | grep -P "\'admin\'" -A 1
    indexer_username: 'admin'
    indexer_password: '8RohQnKBIbPCDYWtjhAY6?WIY+Sy9N+J'
root@nospi-visiotech:~#
root@nospi-visiotech:~# curl -k -u admin:8RohQnKBIbPCDYWtjhAY6?WIY+Sy9N+J https://192.168.1.42:9200
{
  "name" : "node-1",
  "cluster_name" : "wazuh-indexer-cluster",
  "cluster_uuid" : "3XL8cu5STn-I63znoJq6pA",
  "version" : {
    "number" : "7.10.2",
    "build_type" : "deb",
    "build_hash" : "e5a68d19815af94a9883fead7927edb40181f32d",
    "build_date" : "2025-03-26T19:08:40.098412Z",
    "build_snapshot" : false,
    "lucene_version" : "9.11.1",
    "minimum_wire_compatibility_version" : "7.10.0",
    "minimum_index_compatibility_version" : "7.0.0"
  },
  "tagline" : "The OpenSearch Project: https://opensearch.org/"
}
root@nospi-visiotech:~#
```

On install wazuh-server

```
root@nospi-visiotech:~# bash wazuh-install.sh --wazuh-server wazuh-1
08/05/2025 05:25:11 INFO: Starting Wazuh installation assistant. Wazuh version: 4.11.2 (x86_64/AMD64)
08/05/2025 05:25:11 INFO: Verbose logging redirected to /var/log/wazuh-install.log
08/05/2025 05:25:34 INFO: Verifying that your system meets the recommended minimum hardware requirements.
08/05/2025 05:25:53 INFO: Wazuh repository added.
08/05/2025 05:25:53 INFO: --- Wazuh server ---
08/05/2025 05:25:53 INFO: Starting the Wazuh manager installation.
08/05/2025 05:29:38 INFO: Wazuh manager installation finished.
08/05/2025 05:29:39 INFO: Wazuh manager vulnerability detection configuration finished.
08/05/2025 05:29:39 INFO: Starting service wazuh-manager.
08/05/2025 05:30:17 INFO: wazuh-manager service started.
08/05/2025 05:30:17 INFO: Starting Filebeat installation.
08/05/2025 05:33:54 INFO: Filebeat installation finished.
08/05/2025 05:34:00 INFO: Filebeat post-install configuration finished.
08/05/2025 05:34:09 INFO: The filebeat.yml file has been updated to use the Filebeat Keystore username and password.
08/05/2025 05:35:08 INFO: Starting service filebeat.
08/05/2025 05:35:19 INFO: filebeat service started.
08/05/2025 05:35:19 INFO: Installation finished.
root@nospi-visiotech:~#
```

On install wazuh-dashboard

```
root@nospi-visiotech:~# bash wazuh-install.sh --wazuh-dashboard dashboard
08/05/2025 09:57:12 INFO: Starting Wazuh installation assistant. Wazuh version: 4.11.2 (x86_64/AMD64)
08/05/2025 09:57:12 INFO: Verbose logging redirected to /var/log/wazuh-install.log
08/05/2025 09:57:36 INFO: Verifying that your system meets the recommended minimum hardware requirements.
08/05/2025 09:57:36 INFO: Wazuh web interface port will be 443.
08/05/2025 09:57:57 INFO: Wazuh repository added.
08/05/2025 09:57:57 INFO: --- Wazuh dashboard ----
08/05/2025 09:57:57 INFO: Starting Wazuh dashboard installation.
08/05/2025 10:02:44 INFO: Wazuh dashboard installation finished.
08/05/2025 10:02:45 INFO: Wazuh dashboard post-install configuration finished.
08/05/2025 10:02:45 INFO: Starting service wazuh-dashboard.
08/05/2025 10:02:51 INFO: wazuh-dashboard service started.
08/05/2025 10:03:49 INFO: Initializing Wazuh dashboard web application.
08/05/2025 10:03:51 INFO: Wazuh dashboard web application initialized.
08/05/2025 10:03:51 INFO: --- Summary ---
08/05/2025 10:03:51 INFO: You can access the web interface https://192.168.1.42:443
    User: admin
    Password: 8RohQnKBIbPCDYWtjhAY6?WIY+Sy9N+J
08/05/2025 10:03:51 INFO: Installation finished.
root@nospi-visiotech:~#
```

Les informations de connexion sont affichées (A conserver)

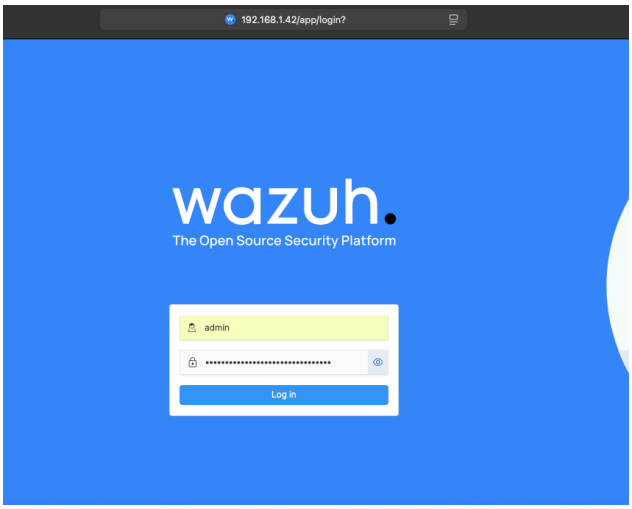
User: admin

Password: 8RohQnKBIbPCDYWtjhAY6?WIY+Sy9N+J

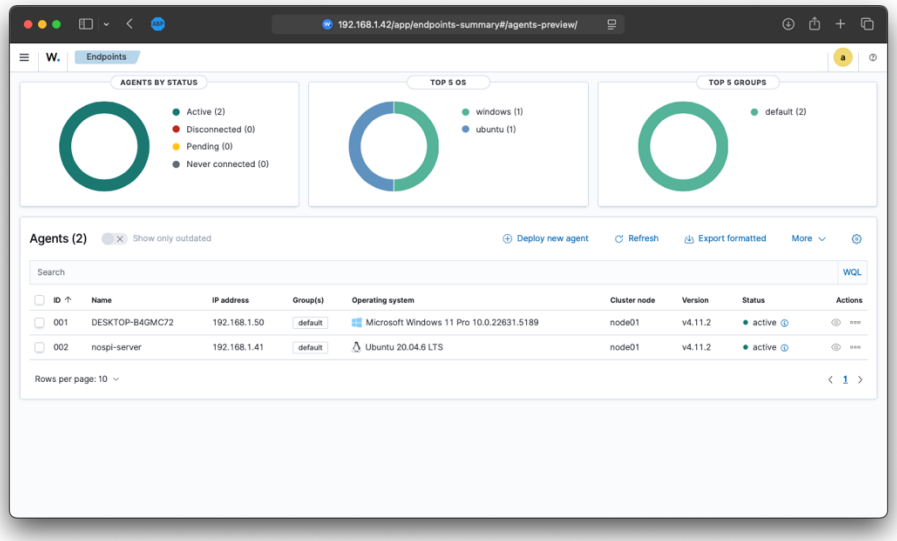
On peut vérifier que nos agents sont actifs en ligne de commande

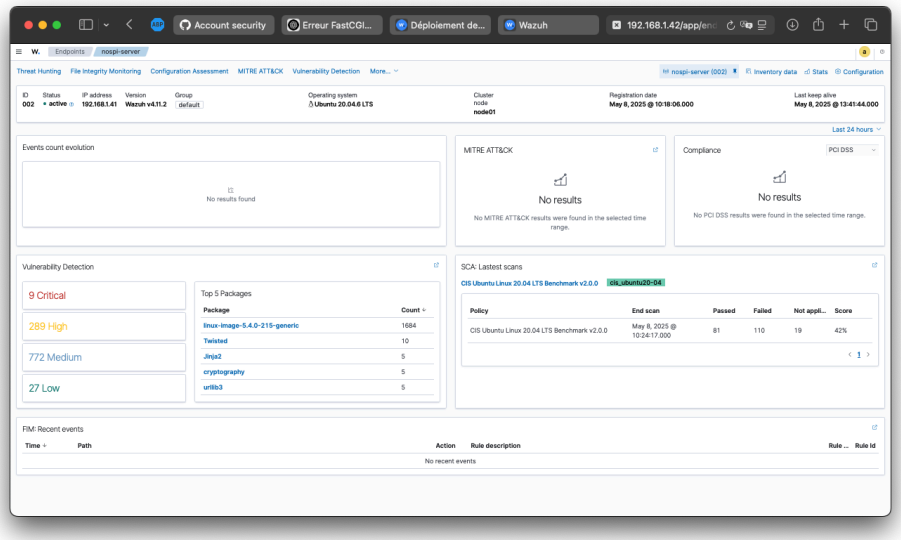
```
root@nospi-visiotech:~# /var/ossec/bin/agent_control -l

Wazuh agent_control. List of available agents:
   ID: 000, Name: nospi-visiotech (server), IP: 127.0.0.1, Active/Local
   ID: 001, Name: DESKTOP-B4GMC72, IP: any, Active
   ID: 002, Name: nospi-server, IP: any, Active

List of agentless devices:

root@nospi-visiotech:~#
```

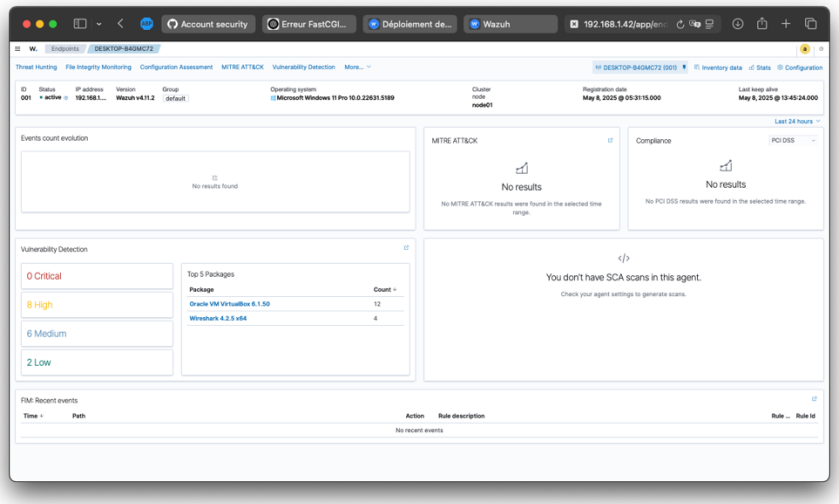
On se connecte à l'interface avec l'utilisateur admin qui a été généré

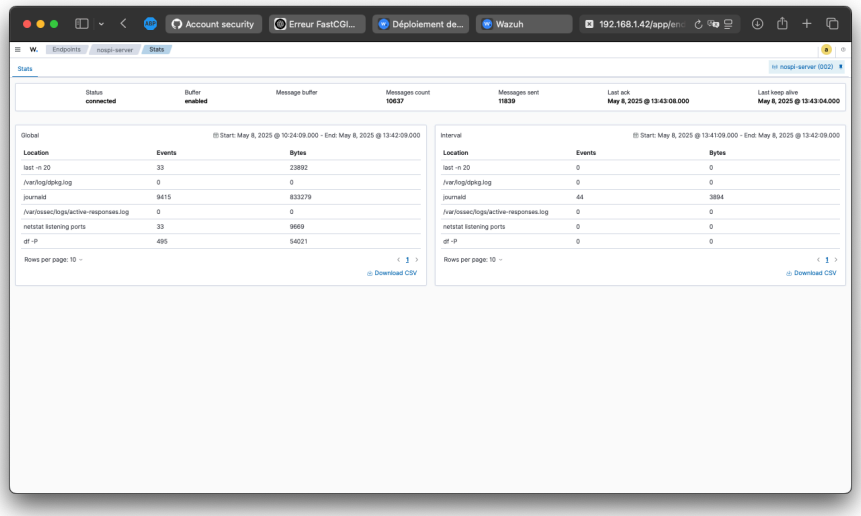


On constate bien nos deux agents ubuntu et windows









4. Commenter les résultats obtenus.

- **Alertes :** Les événements de sécurité (ex. : tentatives de connexion SSH sur Ubuntu, modifications de fichiers sur Windows) sont visibles dans « Security Events ».

- **Logs collectés :** Les journaux système (syslog pour Ubuntu, journaux d'événements Windows) sont centralisés et consultables.

- **Tableaux de bord :** Les graphiques montrent les types d'alertes, leur gravité et leur fréquence. Par exemple, des alertes de niveau 3 (faible) pour des erreurs mineures, ou de niveau 12 pour des menaces critiques.

- **Performance :** La latence dans la collecte dépend de la charge réseau et des performances du serveur. Une configuration correcte garantit une détection en temps réel.

TP4 : Notification Wazuh avec Slack

1. Présenter l'objectif et l'architecture du TP.

Objectif : Configurer Wazuh pour envoyer des notifications d'alertes de sécurité à un canal Slack, permettant une réponse rapide aux incidents.

Architecture :

Wazuh Manager : Génère les alertes basées sur les événements collectés.
Slack Webhook : URL sécurisée pour envoyer des messages à un canal Slack.
Filebeat/Elasticsearch : Stocke les alertes avant transmission.
Slack : Application recevant les notifications via un canal dédié.

2. Identifier les cas d'utilisation de Slack.

Notification en temps réel : Informer les équipes de sécurité des incidents critiques (ex. : détection de malware).
Collaboration : Faciliter la communication entre les administrateurs pour coordonner les réponses.
Automatisation : Intégrer Slack avec d'autres outils (ex. : ticketing) pour déclencher des workflows.
Audit : Garder une trace des alertes dans un canal Slack pour analyse ultérieure.

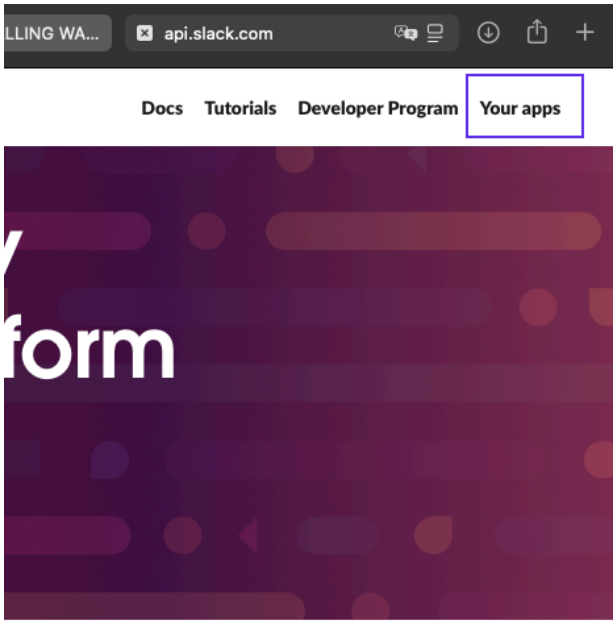
3. Envoyer un message Slack via la commande curl

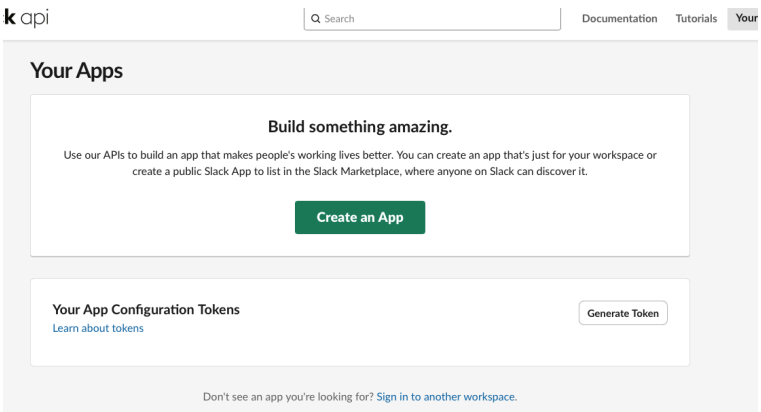
Cette étape consiste à tester manuellement l'envoi d'un message à Slack en utilisant une URL de webhook Slack via la commande curl.

Créer un webhook Slack :

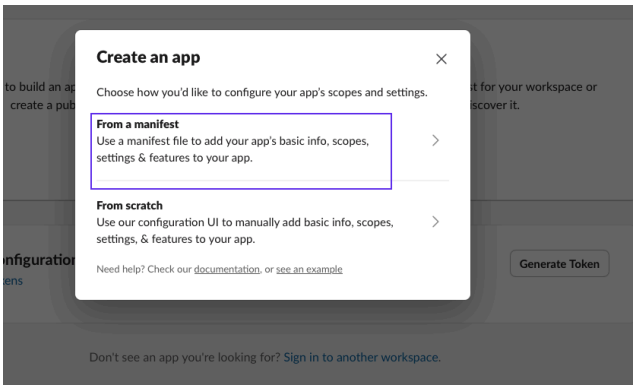
- Allez sur le site de Slack (https://api.slack.com/) et connectez-vous à votre espace de travail.

- Créez une application Slack :
  - Cliquez sur "Your Apps" > "Create New App".

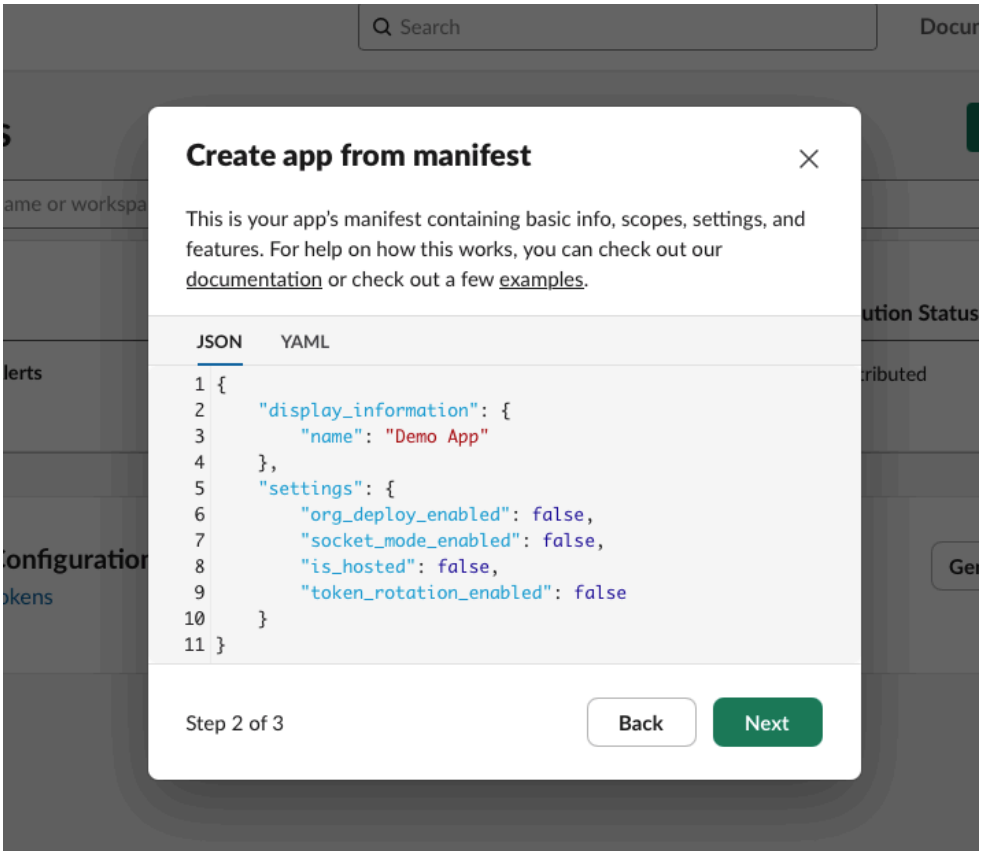




Choisir from a manifest



Créer un fichier manifeste :



On Crée un fichier (par exemple, slack_manifest.json) sur le serveur Wazuh (`nospi-visiotech`):

vim slack_manifest.json

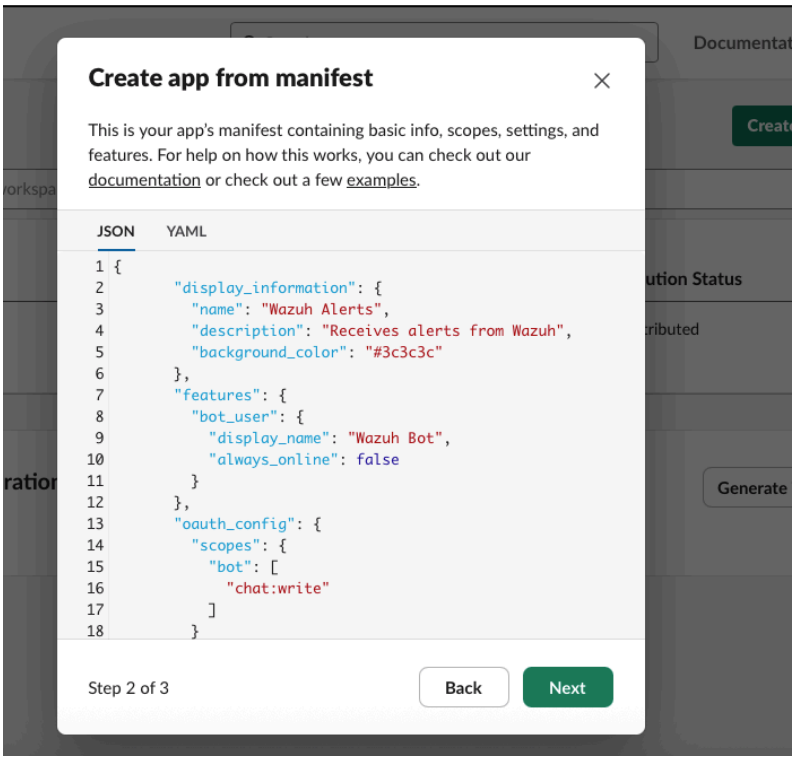
On ajoute une configuration minimale pour ton application Slack.

```
{
  "display_information": {
    "name": "Wazuh Alerts",
    "description": "Receives alerts from Wazuh",
    "background_color": "#3c3c3c"
  },
  "features": {
    "bot_user": {
```

```
      "display_name": "Wazuh Bot",
      "always_online": false
    }
  },
  "oauth_config": {
    "scopes": {
      "bot": [
        "chat:write"
      ]
    }
  },
  "settings": {
    "org_deploy_enabled": false,
    "socket_mode_enabled": false,
    "token_rotation_enabled": false
  }
}
```

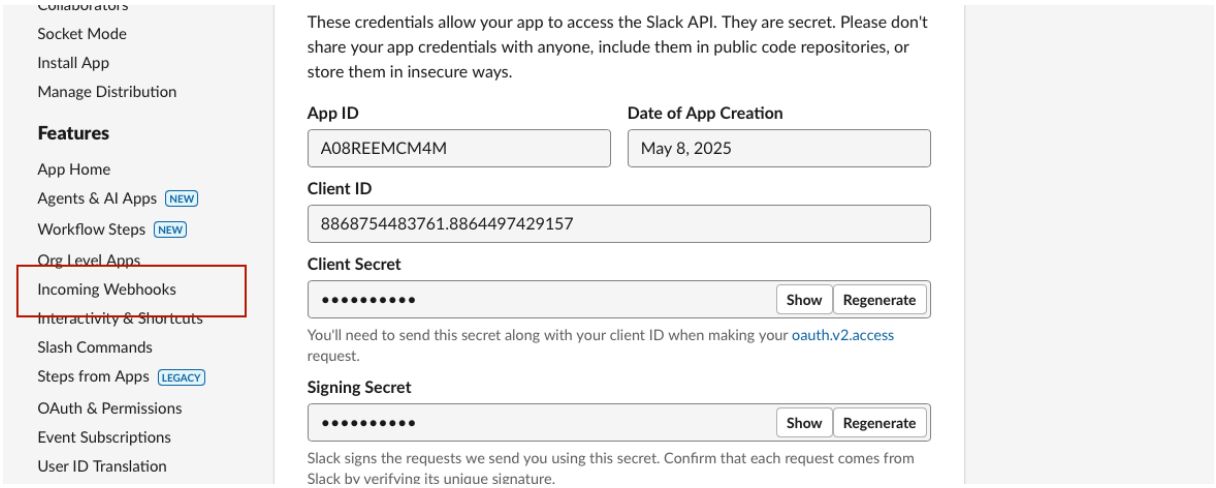
On copie et colle le contenu du fichier slack_manifest.json dans le champ prévu par Slack.

- Clique sur "Next", vérifie le résumé, puis clique sur "Create".

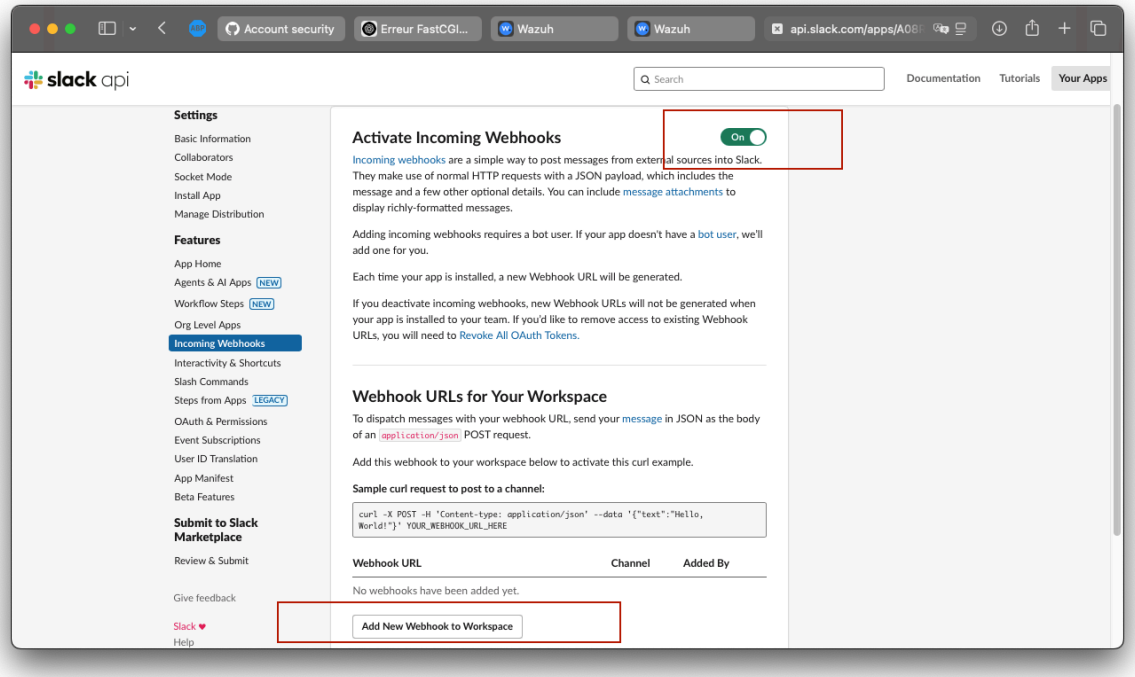


Activer les "Incoming Webhooks" :

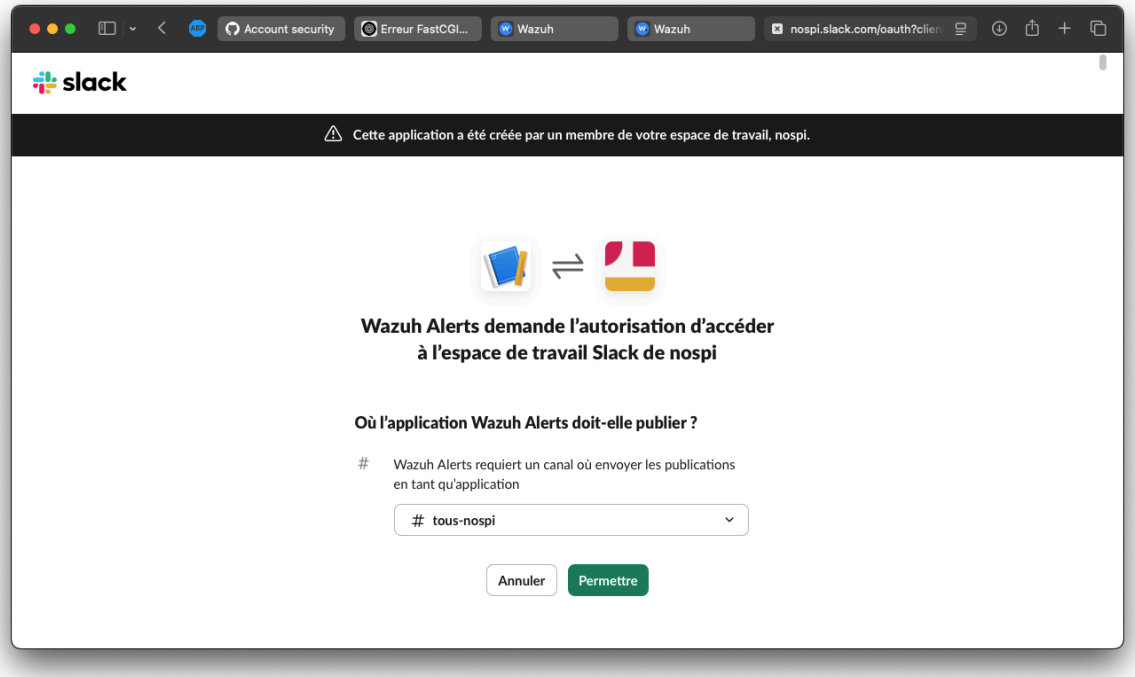
- Une fois l'application créée, va dans "Incoming Webhooks" dans le menu de gauche.
- Active les webhooks ("Activate Incoming Webhooks").

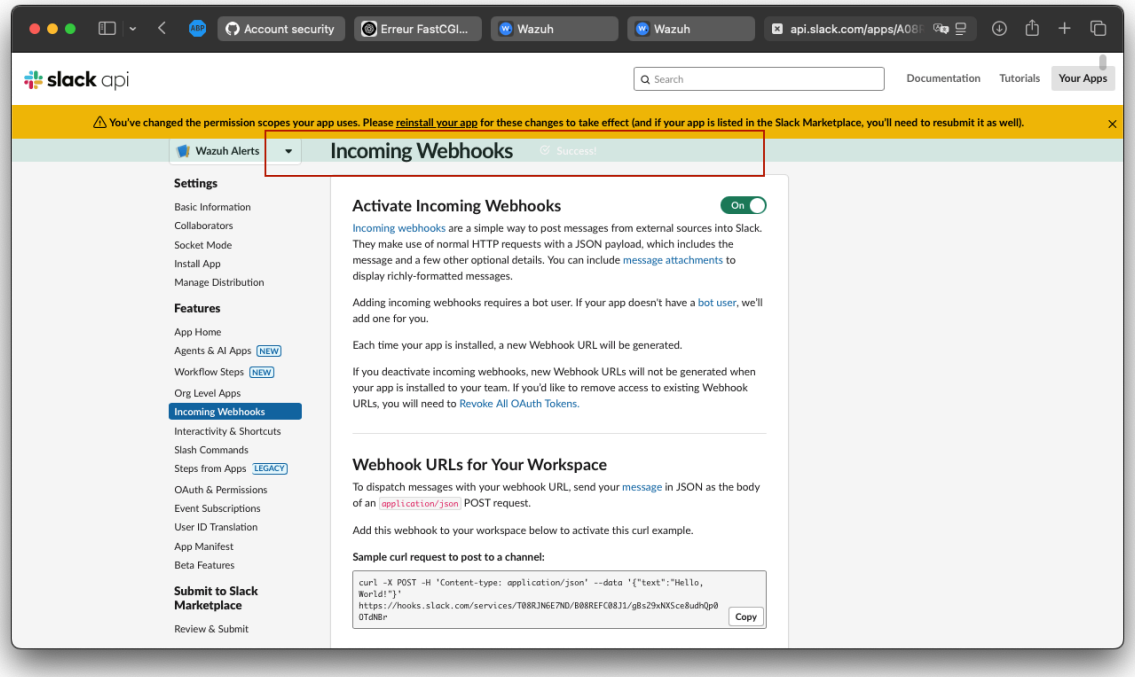


- Clique sur "Add New Webhook to Workspace", sélectionne le canal (ex. : `#wazuh-alerts`), et valide.

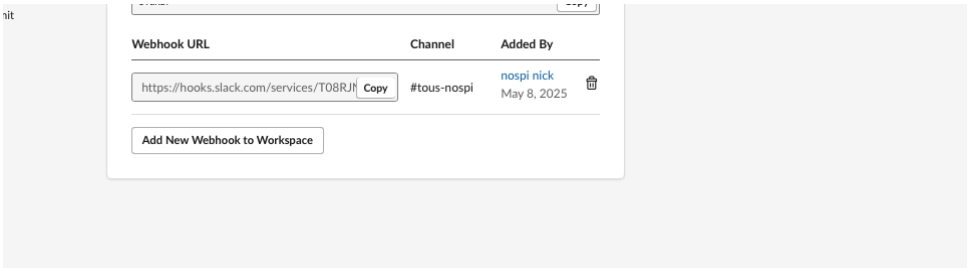


Choisir le canal et cliqué sur permettre





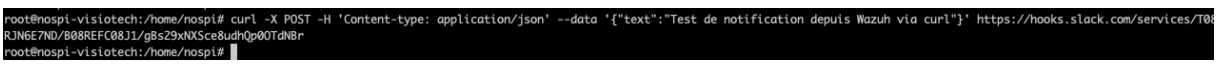
- Copie l'URL du webhook générée (ex. : https://hooks.slack.com/services/TXXXX/BXXXX/XXXX).



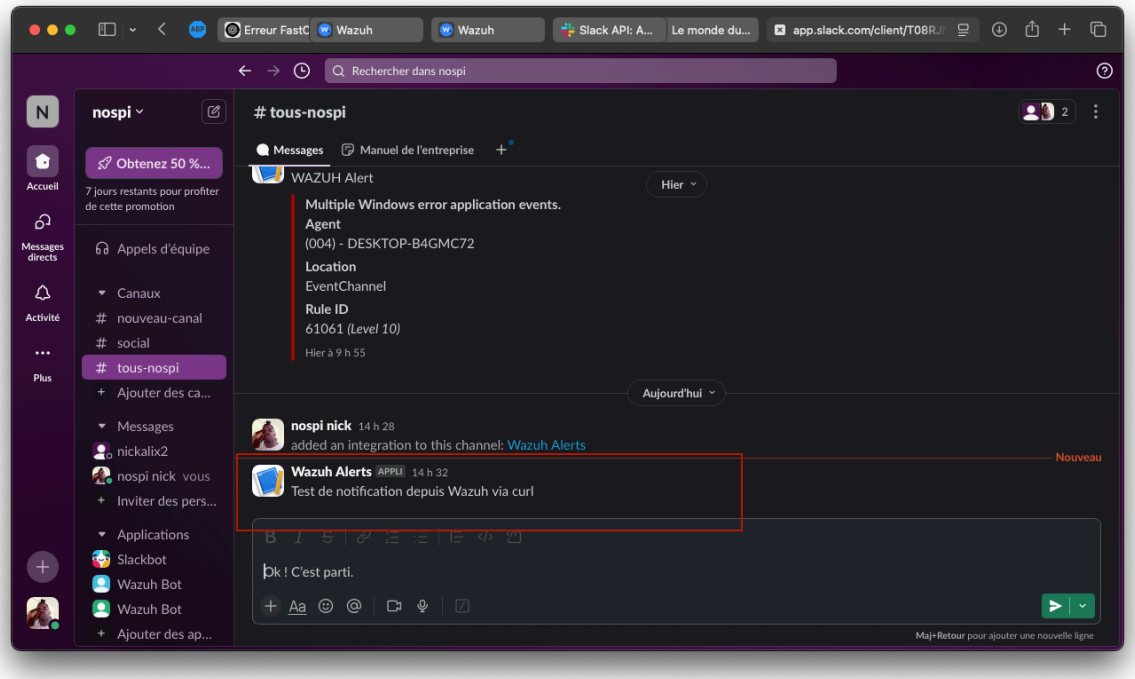
- Tester l'envoi d'un message avec `curl` :

- Sur le serveur Wazuh exécutez la commande suivante pour envoyer un message de test à Slack :

curl -X POST -H 'Content-type: application/json' --data '{"text":"Test de notification depuis Wazuh via curl"}' <SLACK_WEBHOOK_URL>



Si la commande réussit, vous devriez voir le message "Test de notification depuis Wazuh via curl" apparaître dans le canal Slack choisi.



4. Configurer Slack avec Wazuh pour recevoir les notifications

Maintenant que le webhook fonctionne, nous allons configurer Wazuh pour envoyer automatiquement les alertes à Slack.

- Modifier la configuration de Wazuh :

Éditez le fichier de configuration Wazuh (`/var/ossec/etc/ossec.conf`) pour ajouter l'intégration Slack :

```
vim /var/ossec/etc/ossec.conf
```

- Ajoutez le bloc suivant à la fin du fichier, dans la section `<ossec_config>`

```
<ossec_config>

  <integration>
```

<name>slack</name>
<hook_url><SLACK_WEBHOOK_URL></hook_url>
<level>3</level>
<alert_format>json</alert_format>
</integration>

</ossec_config>

```
<integration>
  <name>slack</name>
  <hook_url>https://hooks.slack.com/services/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</hook_url>
  <level>3</level>
  <alert_format>json</alert_format>
</integration>
```

- **<level>3</level>** signifie que seules les alertes de niveau 3 ou supérieur (alertes general) seront envoyées à Slack. On peut ajuster le seuil
- **<alert_format>json</alert_format>** est requis pour que Wazuh envoie les alertes dans un format compatible avec Slack.

- Redémarrer Wazuh Manager** :

Redémarrez le service Wazuh pour appliquer les modifications et vérifiez que le service est actif :

systemctl restart wazuh-manager

systemctl status wazuh-manager

```
root@nospi-visiotech:/home/nospi# systemctl restart wazuh-manager
root@nospi-visiotech:/home/nospi# systemctl status wazuh-manager
● wazuh-manager.service - Wazuh manager
     Loaded: loaded (/usr/lib/systemd/system/wazuh-manager.service; enabled; preset: enabled)
     Active: active (running) since Thu 2025-05-08 14:43:55 GMT; 2min 27s ago
    Process: 258104 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 180 (limit: 14119)
     Memory: 5.5G (peak: 5.5G swap: 67.7M swap peak: 67.7M)
        CPU: 4min 34.831s
     CGroup: /system.slice/wazuh-manager.service
             ├─182824 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─182825 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─182828 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─182831 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258166 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258167 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258168 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258171 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258174 /var/ossec/framework/python/bin/python3 /var/ossec/api/scripts/wazuh_apid.py
             ├─258197 /var/ossec/bin/wazuh-integratord
             ├─258218 /var/ossec/bin/wazuh-authd
             ├─258234 /var/ossec/bin/wazuh-db
             ├─258247 /var/ossec/bin/wazuh-execd
             ├─258261 /var/ossec/bin/wazuh-analysisd
```

5. Vérifier la réception d'un message SIEM

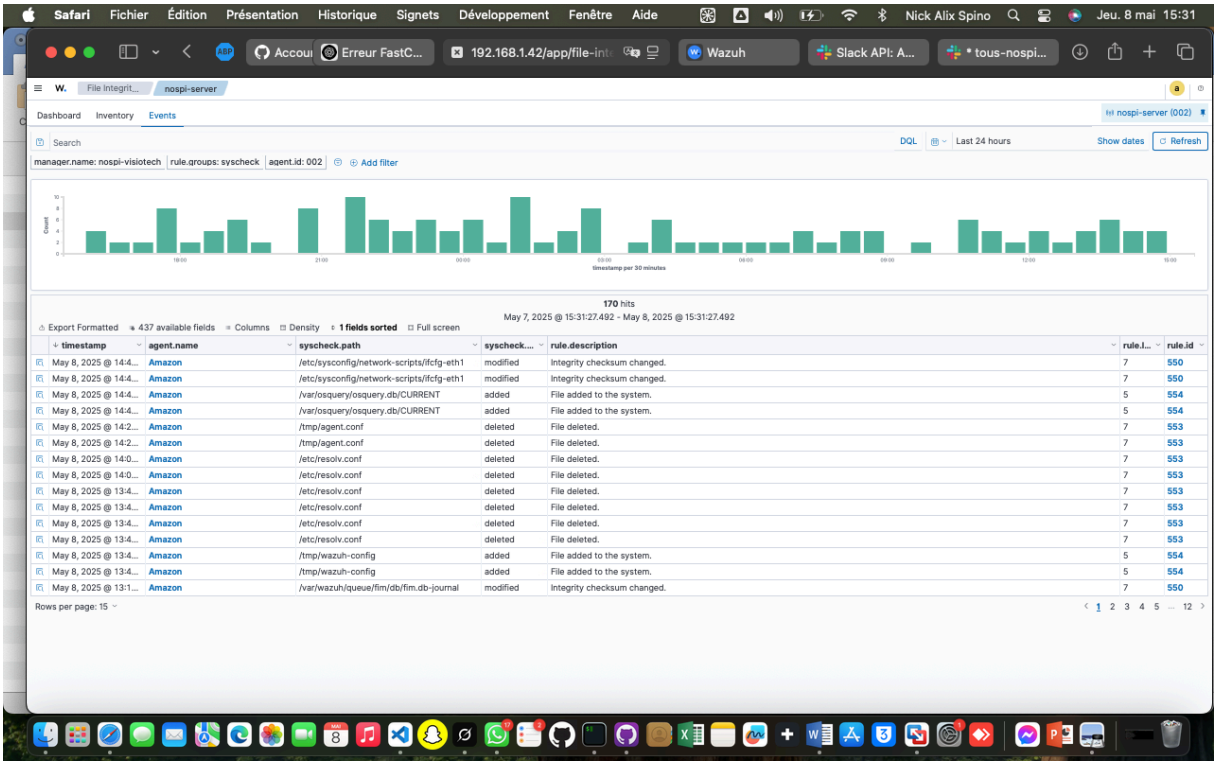
Pour vérifier que Wazuh envoie bien les notifications à Slack, nous allons générer une alerte qui déclenchera une notification.

Vérifier la réception d'un message SIEM

Comme nous avons mis level 3 au redémarrage Slack recevra les notifications automatiquement

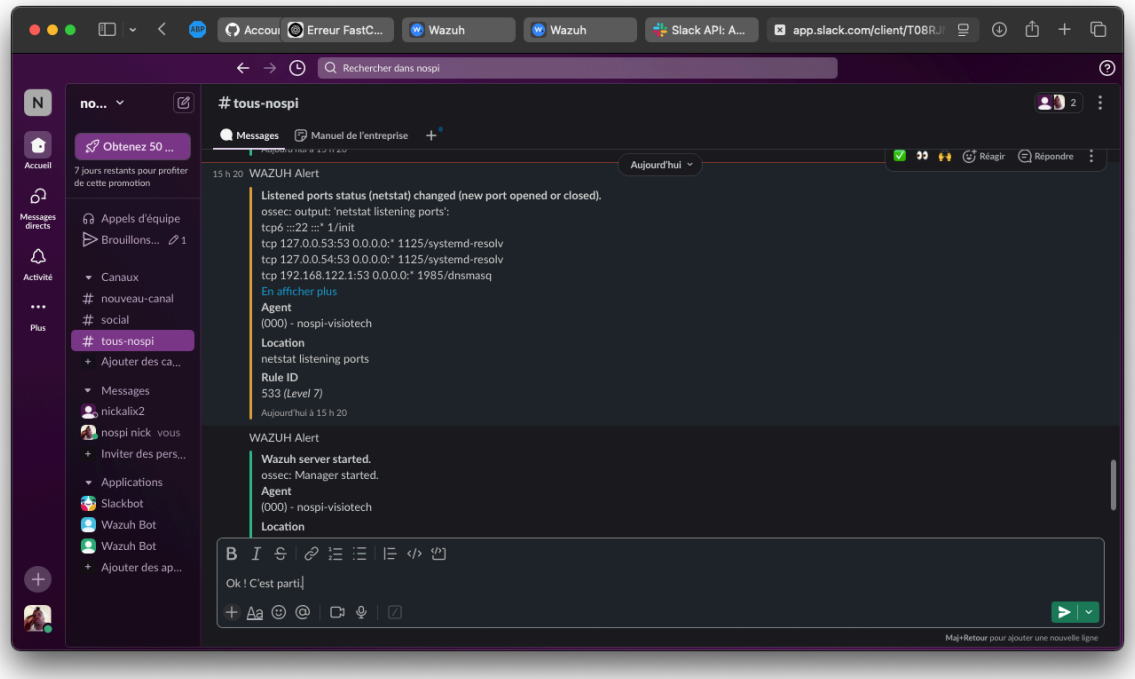
Vérifier dans le Wazuh Dashboard :

- Accède au Wazuh Dashboard via ton navigateur sur https://localhost.
- Se Connecter avec les identifiants.
- Aller dans **Security Events**.



Vérifier dans Slack :

- Va dans le canal Slack configuré (ex. : #wazuh-alerts).
- On devrait voir des messages contenant les détails de l'alerte, par exemple :



TP5 : Preuve de concept de Wazuh

Objectif

L'objectif est de démontrer la capacité de Wazuh à surveiller l'intégrité des fichiers sur un système (via le module File Integrity Monitoring, FIM) en détectant les modifications, ajouts ou suppressions de fichiers dans des répertoires spécifiés. Cela permet de repérer des activités suspectes, comme des modifications malveillantes par un attaquant.

Architecture

Wazuh Manager : Centralise la collecte des données d'intégrité et analyse les alertes. Il est installé sur ton serveur (`nospi-visiotech`).
Wazuh Agent : Déployé sur les machines surveillées (ex. : le serveur lui-même et les agents Windows 10/Ubuntu 20). Il surveille les fichiers en temps réel et envoie les données au Manager.
Wazuh Indexer : Stocke les événements d'intégrité dans une base de données interne
Wazuh Dashboard : Interface web sur `https://localhost:443` pour visualiser les alertes FIM.
Slack : Recevra les notifications des modifications détectées via un webhook configuré.

1. Surveillance de l'intégrité des fichiers

Vérifier la configuration FIM existante :

Ouvre le fichier de configuration Wazuh sur le Manager :

vim /var/ossec/etc/ossec.conf

Cherche la section **<syscheck>** pour voir les répertoires surveillés. Par défaut, Wazuh surveille des répertoires comme **`/etc`, `/bin`, `/usr/bin`,** etc. Exemple typique :

```
<!-- Directories to check  (perform all possible verifications) -->
<directories check_all>/etc,/usr/bin,/usr/sbin</directories>
<directories check_all>/bin,/sbin,/boot</directories>

<!-- Files/directories to ignore -->
<ignore>/etc/mtab</ignore>
```

3. **Générer une modification de fichier** :

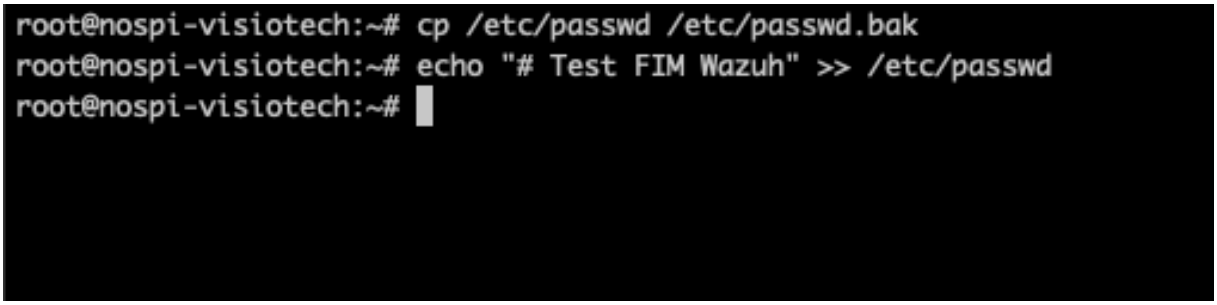
- Sur le serveur (`nospi-visiotech`) ou un agent (ex. : Ubuntu 20), modifie un fichier surveillé :

- Sauvegarde d'abord :

cp /etc/passwd /etc/passwd.bak

- Ajoute une ligne :

echo "# Test FIM Wazuh" >> /etc/passwd



2. Détection d'attaque par force brute avec réponse active

Vérifier la surveillance des logs SSH

Wazuh doit surveiller les logs d'authentification pour détecter les tentatives de connexion échouées.

- Vérifier la configuration des logs :

Ouvre le fichier de configuration Wazuh sur le Manager :

vim /var/ossec/etc/ossec.conf

- Chercher la section `<localfile>` et assure-toi que `/var/log/auth.log` est surveillé. Par défaut, cela devrait déjà être inclus :

<localfile>

<log_format>syslog</log_format>
<location>/var/log/auth.log</location>
</localfile>

Vérifier la règle de détection des attaques par force brute

Wazuh a des règles intégrées pour détecter les attaques par force brute sur SSH (règles dans /var/ossec/ruleset/rules/).

Vérifier la règle SSH :

- Les règles SSH sont dans `/var/ossec/ruleset/rules/0095-sshd_rules.xml`. Ouvre ce fichier pour confirmer :

ls /var/ossec/ruleset/rules/0095-sshd_rules.xml

```
root@nospi-visiotech:~# ls /var/ossec/ruleset/rules/0095-sshd_rules.xml
/var/ossec/ruleset/rules/0095-sshd_rules.xml
root@nospi-visiotech:~#
```

- Cherche une règle comme celle-ci (exemple pour une attaque par force brute) :

```
<rule id="5712" level="10" frequency="8" timeframe="120" ignore="60">
  <if_matched_sid>5710</if_matched_sid>
  <same_source_ip />
  <description>sshd: brute force trying to get access to the system. Non existent user.</description>
  <mitre>
    <id>T1110</id>
  </mitre>
  <group>authentication_failures,gdpr_IV_35.7.d,gdpr_IV_32.2,hipaa_164.312.b,nist_800_53_SI.4,nist_800_53_AU.14,nist_800_53_AC.7,pci_dss_11.4,pci_dss_10.2.4,pci_
1,tsc_CC6.8,tsc_CC7.2,tsc_CC7.3,</group>
</rule>

<rule id="5713" level="6">
```

Explication :

La règle 5710 détecte une tentative de connexion échouée.

- La règle 5712 (niveau 10) déclenche une alerte si 6 tentatives échouées (`frequency="8"`) sont détectées dans un délai de 120 secondes (timeframe="120").

Configurer la réponse active

Wazuh peut exécuter une action (réponse active) lorsqu'une attaque par force brute est détectée, comme bloquer l'IP de l'attaquant avec `iptables`.

Activer la réponse active :

- Ouvre le fichier de configuration Wazuh :

vim /var/ossec/etc/ossec.conf

Ajoute ou modifie la section `<active-response>` pour bloquer les IP malveillantes :

```
<active-response>
  <command>firewall-drop</command>
  <location>local</location>
  <rules_id>5712</rules_id>
  <timeout>300</timeout>
</active-response>
```

Explications:

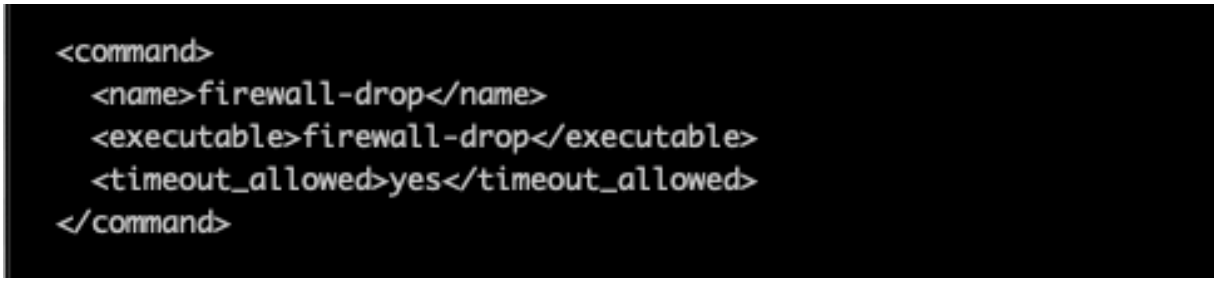
- **command :** Utilise la commande intégrée `firewall-drop` pour bloquer l'IP.
- **location :** Applique la réponse sur la machine locale (`nospi-visiotech`).
- **rules_id :** Associe cette réponse à la règle `5712` (attaque par force brute).
- **timeout :** Bloque l'IP pendant 300 secondes (5 minutes).

Wazuh fournit une commande intégrée `firewall-drop` qui utilise `iptables`. Vérifie sa configuration :

vim /var/ossec/etc/ossec.conf

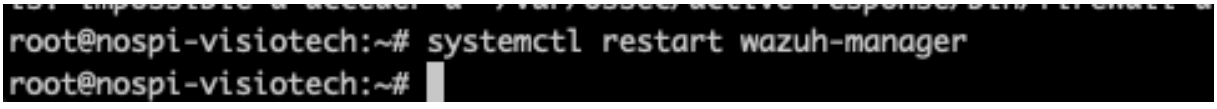
- Cherche la section <command> :

<command>
<name>firewall-drop</name>
<executable>firewall-drop.sh</executable>
<timeout_allowed>yes</timeout_allowed>
</command>



Redémarrer Wazuh :

systemctl restart wazuh-manager



Simuler une attaque par force brute

- Sur une autre machine essaie de te connecter à `nospi-visiotech` via SSH :

ssh alix@<IP_de_nospi-visiotech>

- Répète plusieurs fois avec un mot de passe incorrect.

```
nospi@nospi-server:~$ ssh alix@192.168.1.42
The authenticity of host '192.168.1.42 (192.168.1.42)' can't be established.
ECDSA key fingerprint is SHA256:9VNXlTn4vGM8PGQP7whw4Ius2G5hG8JAoOY6vnm/EG8.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.42' (ECDSA) to the list of known hosts.
alix@192.168.1.42's password:
Permission denied, please try again.
alix@192.168.1.42's password:
Permission denied, please try again.
alix@192.168.1.42's password:
alix@192.168.1.42: Permission denied (publickey,password).
nospi@nospi-server:~$ _
```

Vérifier la détection dans Wazuh Dashboard

Accède au Wazuh Dashboard sur `https://localhost:443`.

Va dans Security Events.

Cherche une alerte avec la règle ` ("SSHD brute force attack").

May 8, 2025 @ 13:5...	Amazon	T1021	Lateral Movement	sshd: Reverse lookup error (bad ISP or attack).
May 8, 2025 @ 13:5...	Amazon	T1021	Lateral Movement	sshd: Reverse lookup error (bad ISP or attack).
May 8, 2025 @ 13:4...	Amazon	T1021	Lateral Movement	sshd: Possible attack on the ssh server (or version gathering).
May 8, 2025 @ 13:4...	Amazon	T1021	Lateral Movement	sshd: Possible attack on the ssh server (or version gathering).
May 8, 2025 @ 13:2...	Amazon	T1110 T102	Credential Access, Lateral M...	sshd: brute force trying to get access to the system.
May 8, 2025 @ 13:2...	Amazon	T1110 T102	Credential Access, Lateral M...	sshd: brute force trying to get access to the system.
May 8, 2025 @ 13:1...	Amazon	T1114	Collection	Postfix: Sender domain is not found (450: Requested mail action not taken).
May 8, 2025 @ 13:1...	Amazon	T1114	Collection	Postfix: Sender domain is not found (450: Requested mail action not taken).
May 8, 2025 @ 13:0...	Amazon	T1107 T148	Defense Evasion, Impact	File deleted.

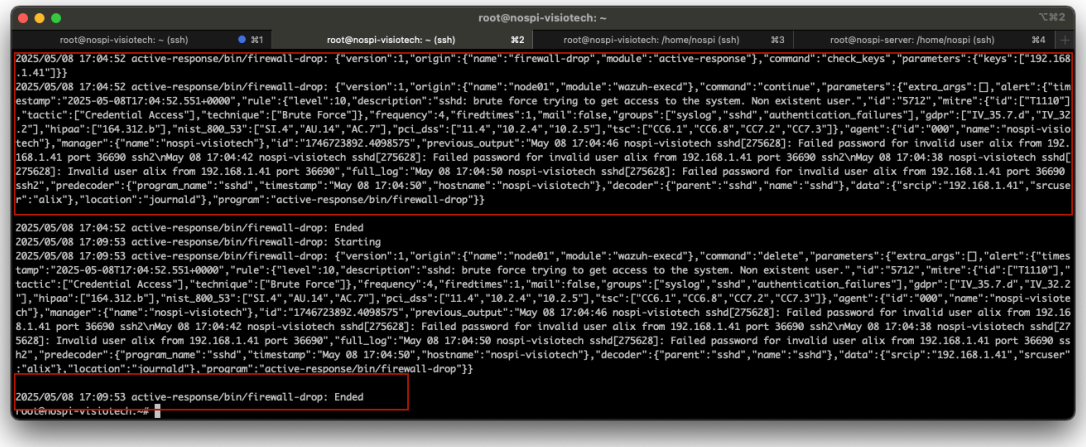
Rows per page: 15

Vérifier la réponse active

Logs de réponse active :

- Vérifie les logs Wazuh pour confirmer l'exécution de la réponse active :

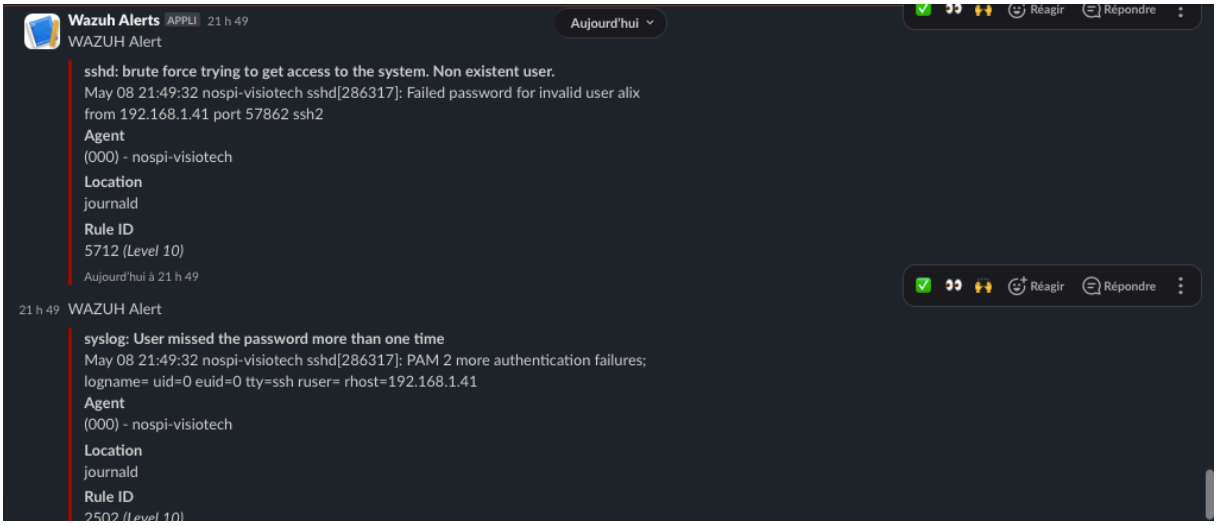
  cat /var/ossec/logs/active-responses.log



Vérifier la notification dans Slack

Allez dans le canal Slack (ex. : #wazuh-alerts).

Tu devrais voir une notification pour l'alerte de niveau 10 :

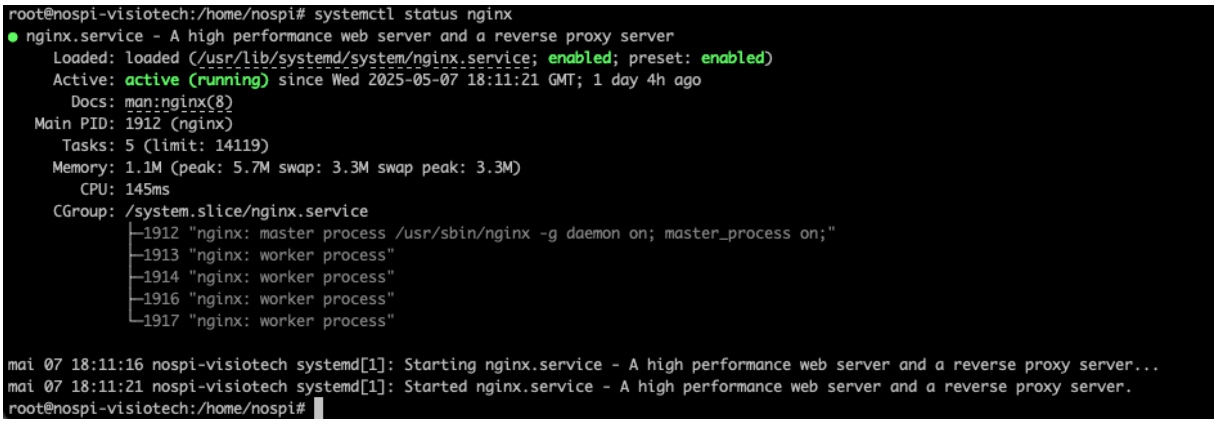


3. Détection d'injection SQL

Vérifier l'installation et la configuration de Nginx ou apache2

S'assurer que Nginx est installé et en cours d'exécution :

systemctl status nginx



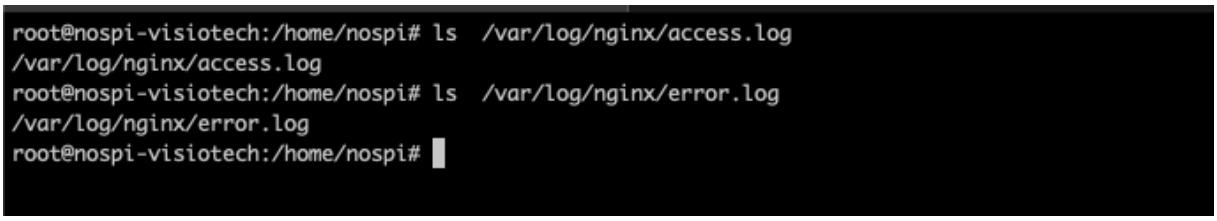
- Vérifie les logs par défaut de Nginx :

- Logs d’accès : `/var/log/nginx/access.log`

Logs d’erreur : `/var/log/nginx/error.log`

ls /var/log/nginx/access.log

ls /var/log/nginx/error.log



Configurer Wazuh pour surveiller les logs Nginx

Wazuh doit être configuré pour lire les logs Nginx et appliquer des règles pour détecter les injections SQL.

Modifier `ossec.conf` :

vim /var/ossec/etc/ossec.conf

On verifie que y’a section `<localfile>` pour surveiller les logs Nginx :

```
<localfile>
  <log_format>plain</log_format>
  <location>/var/log/nginx/access.log</location>
</localfile>
<localfile>
  <log_format>plain</log_format>
  <location>/var/log/nginx/error.log</location>
</localfile>
```

```
<localfile>
  <log_format>apache</log_format>
  <location>/var/log/nginx/access.log</location>
</localfile>

<localfile>
  <log_format>apache</log_format>
  <location>/var/log/nginx/error.log</location>
</localfile>
```

Vérifier les règles d'injection SQL

Wazuh inclut des règles pour détecter les injections SQL dans les logs web (fichier `/var/ossec/ruleset/rules/0245-web_rules.xml`).

```
root@nospi-visiotech:/home/nospi# ls /var/ossec/ruleset/rules/0245-web_rules.xml
/var/ossec/ruleset/rules/0245-web_rules.xml
root@nospi-visiotech:/home/nospi#
```

Vérifier les règles :

vim /var/ossec/ruleset/rules/0245-web_rules.xml

- Cherche une règle comme celle-ci :

```
<rule id="31103" level="7">
  <if_sid>31100,31108</if_sid>
  <url>=select%20|select+|insert%20|%20from%20|%20where%20|union%20|</url>
  <url>union+|where+|null,null|xp_cmdshell</url>
  <description>SQL injection attempt.</description>
  <mitre>
    <id>T1190</id>
  </mitre>
  <group>attack,sql_injection,pci_dss_6.5,pci_dss_11.4,pci_dss_6.5.1,gdpr_IV_35.7.d,nist_800_53_SA.11,nist_800_53_SI.4,tsc_CC6.6,tsc_CC7.1,tsc_CC8.1,tsc
_CC6.1,tsc_CC6.8,tsc_CC7.2,tsc_CC7.3,</group>
</rule>
```

Explication :

- La règle `31103` (niveau 7) détecte des motifs courants d'injection SQL (ex. : `'`, `;`, `SELECT`) dans les logs web.
- Elle est basée sur la règle parente `31100 & 31108` (activité web suspecte).

Simuler une tentative d'injection SQL

Créer une page web vulnérable

Crée un fichier PHP dans le répertoire par défaut de Nginx (`/var/www/html`) :

```
sudo nano /var/www/html/test.php
```

- Ajoute un formulaire vulnérable à l'injection SQL :

```
<?php
// Connexion basique (pour démonstration)
$conn = new mysqli("localhost", "root", "", "test_db");
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "GET") {
    $id = $_GET['id'];
    $query = "SELECT * FROM users WHERE id = '$id'";
    $result = $conn->query($query);
    if ($result) {
        echo "Résultat : " . $result->num_rows . " lignes trouvées.<br>";
    } else {
        echo "Erreur : " . $conn->error . "<br>";
    }
}
$conn->close();
?>
<!DOCTYPE html>
<html>
<body>
    <h2>Formulaire de recherche</h2>
    <form method="get" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);
?>">
        ID : <input type="text" name="id">
```

<input type="submit" value="Rechercher">

</form>

</body>

</html>

```php
<?php
$conn = new mysqli("localhost", "nick", "passer", "test_db");
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

if ($_SERVER["REQUEST_METHOD"] == "GET") {
    $id = $_GET['id'];
    $query = "SELECT * FROM users WHERE id = '$id'";
    $result = $conn->query($query);
    if ($result) {
        echo "Résultat : " . $result->num_rows . " lignes trouvées.<br>";
    } else {
        echo "Erreur : " . $conn->error . "<br>";
    }
}
$conn->close();
?>
<!DOCTYPE html>
<html>
<body>
    <h2>Formulaire de recherche</h2>
    <form method="get" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
```

Explications :

Ce code crée un formulaire simple où l'utilisateur entre un `id`.

La requête SQL utilise directement l'entrée utilisateur sans sanitisation, simulant une vulnérabilité.

On aura besoin d'une base de données `test_db` avec une table `users`. ```bash

Dans MySQL :

CREATE DATABASE test_db;

USE test_db;

CREATE TABLE users (id INT);

INSERT INTO users (id) VALUES (1);

EXIT;

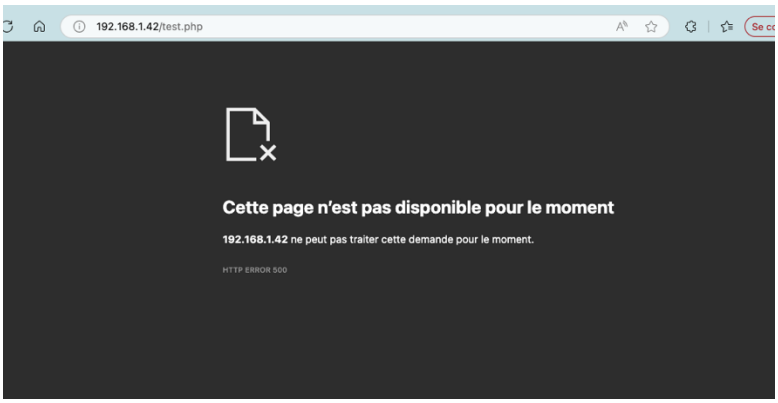
```
MariaDB [(none)]>     CREATE DATABASE test_db;
Query OK, 1 row affected (0,037 sec)

MariaDB [(none)]>     USE test_db;
Database changed
MariaDB [test_db]>     CREATE TABLE users (id INT);
Query OK, 0 rows affected (0,722 sec)

MariaDB [test_db]>     INSERT INTO users (id) VALUES (1);
Query OK, 1 row affected (0,122 sec)

MariaDB [test_db]>     EXIT;
Bye
root@nospi-visiotech:/var/www/html#
```

Tester le formulaire

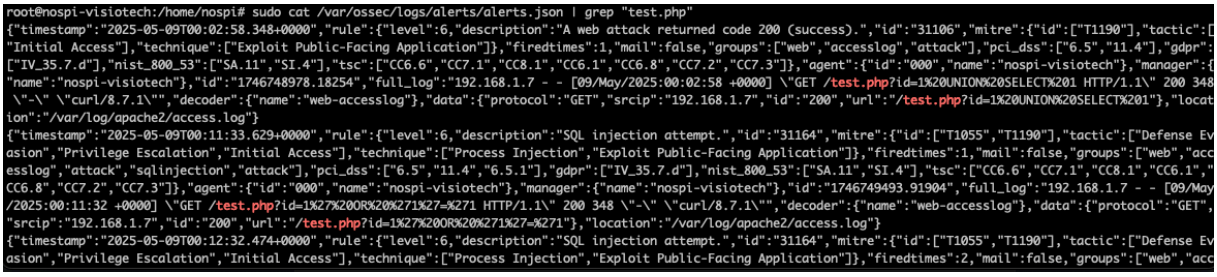
Accède à la page via ton navigateur : `http://192.168.1.42/test.php`

Entre une valeur valide (ex. : `1`) et soumets. On devrait voir un résultat.

- Entre une injection SQL (ex. : `1' OR '1'='1`) et soumets. Cela devrait générer une erreur ou un comportement inattendu dans les logs.



on vérifie les logs pour voir que l'injection est détecté



Vérifier la notification dans Slack

Va dans ton canal Slack (ex. : `#wazuh-alerts`).

4. Surveillance des événements Docker

- Vérifie que Docker fonctionne :

docker run hello-world

```
root@nospi-visiotech:/home/nospi# docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

root@nospi-visiotech:/home/nospi#
```

Configurer Wazuh pour surveiller les logs Docker

Docker utilise généralement `journald` pour journaliser ses événements sur Ubuntu, mais les logs peuvent aussi être dans `/var/lib/docker/` ou configurés différemment. Nous allons surveiller les logs via `journald`.

Vérifier les logs Docker :

- Vérifie les événements Docker dans `journald` :

journalctl -u docker.service

```
root@nospi-visiotech:/home/nospi# journalctl -u docker.service
mai 03 10:35:01 nospi-visiotech dockerd[2232]: time="2025-05-03T10:35:00.695749040Z" level=info msg="Processing signal 'terminated'"
mai 03 10:35:01 nospi-visiotech dockerd[2232]: time="2025-05-03T10:35:00.912396839Z" level=info msg="stopping event stream following graceful
mai 03 10:35:01 nospi-visiotech dockerd[2232]: time="2025-05-03T10:35:01.078287723Z" level=info msg="Daemon shutdown complete"
mai 03 10:35:00 nospi-visiotech systemd[1]: Stopping docker.service - Docker Application Container Engine...
mai 03 10:35:01 nospi-visiotech systemd[1]: docker.service: Deactivated successfully.
mai 03 10:35:01 nospi-visiotech systemd[1]: Stopped docker.service - Docker Application Container Engine.
mai 03 10:35:01 nospi-visiotech systemd[1]: docker.service: Consumed 2min 37.336s CPU time, 148.1M memory peak, 0B memory swap peak.
-- Boot 44ea90cb2a6d414c9e1200072f4a35f2 --
mai 03 10:59:35 nospi-visiotech systemd[1]: Starting docker.service - Docker Application Container Engine...
mai 03 10:59:59 nospi-visiotech dockerd[2280]: time="2025-05-03T10:59:59.020241028Z" level=info msg="Starting up"
```

Modifier ossec.conf :

- Ouvre le fichier de configuration Wazuh :

vim /var/ossec/etc/ossec.conf

Ajoute une section <localfile> pour surveiller les logs Docker via `journald` :

```
<localfile>
  <log_format>syslog</log_format>
  <location>/var/log/syslog</location>
</localfile>
```

systemctl restart wazuh-manager

Vérifier les règles Docker

Wazuh inclut des règles pour Docker dans /var/ossec/ruleset/rules/0455-docker_rules.xml.

Simuler un événement Docker

Lance un conteneur pour générer un événement info :

docker run -d --name test-container nginx

Arrête-le pour générer un autre événement :

sudo docker stop test-container

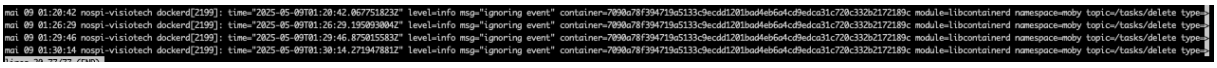
```
root@nospi-visiotech:/var/www/html# docker stop test-container
test-container
root@nospi-visiotech:/var/www/html#
```

Vérifier les logs Docker

Vérifie les logs dans journald :

journalctl -u docker.service

```
2025-05-09T01:29:46.877388+00:00 nospi-visiotech dockerd[2199]: time="2025-05-09T01:29:46.875015583Z" level=info msg="ignoring event" container=7090a78f394719a5133c9ecdd1201bad4eb6a4cd9edca31c720c332b2172189c module=libcontainerd namespace=moby topic=/tasks/delete type="*events.TaskDelete"
2025-05-09T01:30:14.273597+00:00 nospi-visiotech dockerd[2199]: time="2025-05-09T01:30:14.271947881Z" level=info msg="ignoring event" container=7090a78f394719a5133c9ecdd1201bad4eb6a4cd9edca31c720c332b2172189c module=libcontainerd namespace=moby topic=/tasks/delete type="*events.TaskDelete"
```



Vérifier la notification dans Slack

aller dans le canal Slack (ex. : #wazuh-alerts).
Tu devrais voir une notification pour l'événement Docker :

5. Détection des processus non autorisés

Configurer Wazuh pour surveiller les processus

Wazuh peut surveiller les processus via l'analyse des logs système ou en utilisant le module FIM pour détecter les exécutions.

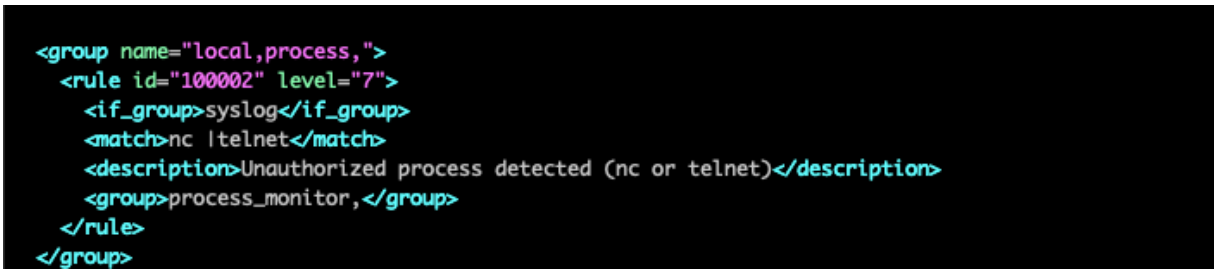
Ajoute une règle personnalisée pour détecter un processus spécifique (ex. : `nc`).

On modifie `/var/ossec/etc/rules/local_rules.xml` :

vim /var/ossec/etc/rules/local_rules.xml

- Ajoute cette règle :

```
<group name="local,process,">
  <rule id="100002" level="7">
    <if_group>syslog</if_group>
    <match>nc |telnet</match>
    <description>Unauthorized process detected (nc or telnet)</description>
    <group>process_monitor,</group>
  </rule>
</group>
```



Explication :

- Détecte `nc` ou `telnet` dans les logs système.
- Niveau 7 pour que l'alerte soit visible et envoyée à Slack.

- Redémarre Wazuh :

systemctl restart wazuh-manager

Simuler un processus non autorisé

Lance un processus comme `nc` (Netcat, souvent utilisé pour des activités malveillantes) :

apt install netcat-traditional -y

```
root@nospi-visiotech:/home/nospi# sudo apt install netcat-traditional -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les NOUVEAUX paquets suivants seront installés :
  netcat-traditional
0 mis à jour, 1 nouvellement installés, 0 à enlever et 189 non mis à jour.
```

nc -l 12345 &

```
root@nospi-visiotech:/home/nospi# nc -l 12345 &
[1] 338518
root@nospi-visiotech:/home/nospi#
```

- Cela démarre `nc` en écoute sur le port 12345.

Vérifier les logs

Vérifie que le processus est logué dans `/var/log/syslog` :

tail -f /var/log/syslog | grep nc

Vérifier la détection dans Wazuh Dashboard

- Accède au Wazuh Dashboard (`https://localhost:443`).

- Cherche une alerte avec la règle `100002` (niveau 7) : "Unauthorized process detected (nc or telnet)".

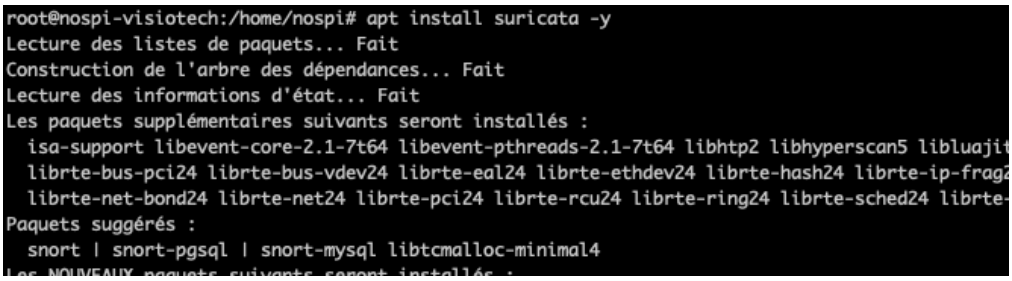
Vérifier la notification dans Slack

6. Intégration de l'IDS réseau

Installer Suricata

Installe Suricata sur `nospi-visiotech` :

apt install suricata -y



Configure Suricata pour surveiller l'interface réseau (ex. : ` enp1s0 `) :

vim /etc/suricata/suricata.yaml

Cherche la section `af-packet` et ajuste :

```
af-packet:
  - interface: enp1s0
    threads: 1
    cluster-id: 99
    cluster-type: cluster_flow
```

```
# Linux high speed capture support
af-packet:
  - interface: enp1s0
    # Number of receive threads. "auto" uses the number of cores
    threads: 1
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible value are:
    #  * cluster_flow: all packets of a given flow are sent to the same socket
    #  * cluster_cpu: all packets treated in kernel by a CPU are sent to the same socket
    #  * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
    #  socket. Requires at least Linux 3.14.
    #  * cluster_ebpf: eBPF file load balancing. See doc/userguide/capture-hardware/ebpf-xdp.rst for
    #  more info.
    # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
    # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
    # cluster_rollover has been deprecated; if used, it'll be replaced with cluster_flow.
    cluster-type: cluster_flow
    # In some fragmentation cases, the hash can not be computed. If "defrag" is set
```

- Active les règles par défaut :

suricata-update

```
root@nospi-visiotech:/var/www/html#   suricata-update
9/5/2025 -- 02:20:00 - <Info> -- Using data-directory /var/lib/suricata.
9/5/2025 -- 02:20:00 - <Info> -- Using Suricata configuration /etc/suricata/suricata.yaml
9/5/2025 -- 02:20:00 - <Info> -- Using /etc/suricata/rules for Suricata provided rules.
9/5/2025 -- 02:20:01 - <Info> -- Found Suricata version 7.0.3 at /usr/bin/suricata.
9/5/2025 -- 02:20:01 - <Info> -- Loading /etc/suricata/suricata.yaml
9/5/2025 -- 02:20:01 - <Info> -- Disabling rules for protocol pgsql
9/5/2025 -- 02:20:01 - <Info> -- Disabling rules for protocol modbus
9/5/2025 -- 02:20:01 - <Info> -- Disabling rules for protocol dnp3
9/5/2025 -- 02:20:01 - <Info> -- Disabling rules for protocol enip
9/5/2025 -- 02:20:01 - <Info> -- No sources configured, will use Emerging Threats Open
9/5/2025 -- 02:20:01 - <Info> -- Fetching https://rules.emergingthreats.net/open/suricata-7.0.3/emerging.rules.tar.gz.
 100% - 4910485/4910485
9/5/2025 -- 02:20:04 - <Info> -- Done.
49/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/app-layer-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/decoder-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/dhcp-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/dnp3-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/dns-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/files.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/http-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/ipsec-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/kerberos-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/modbus-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/nfs-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/ntp-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/smb-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/smtp-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/stream-events.rules
9/5/2025 -- 02:20:04 - <Info> -- Loading distribution rule file /etc/suricata/rules/tls-events.rules
9/5/2025 -- 02:20:05 - <Info> -- Ignoring file rules/emerging-deleted.rules
49/5/2025 -- 02:20:13 - <Info> -- Loaded 58867 rules.
9/5/2025 -- 02:20:14 - <Info> -- Disabled 14 rules.
```

Configurer Wazuh pour lire les logs Suricata

- Les logs Suricata sont généralement dans `/var/log/suricata/eve.json`.
- Ajoute une section dans `/var/ossec/etc/ossec.conf` :

vim /var/ossec/etc/ossec.conf

Ajoute :

<localfile>
<log_format>json</log_format>
<location>/var/log/suricata/eve.json</location>
</localfile>

```
<localfile>
   <log_format>json</log_format>
   <location>/var/log/suricata/eve.json</location>
 </localfile>
```

systemctl restart wazuh-manager

Lancer Suricata

Démarre Suricata sur l'interface réseau :

suricata -c /etc/suricata/suricata.yaml -i enp1s0

```
root@nospi-visiotech:/var/www/html# suricata -c /etc/suricata/suricata.yaml -i enp1s0
i: suricata: This is Suricata version 7.0.3 RELEASE running in SYSTEM mode
```

4. Simuler une activité réseau suspecte

- Génère du trafic réseau pour déclencher une alerte Suricata (ex. : scan de ports avec `nmap`)
:

- Depuis une autre machine, scanne `nospi-visiotech` :

nmap 192.168.1.42

```
nickalixspino@MacBook-Pro-de-Nick-3 ~ % nmap 192.168.1.42
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-09 02:25 GMT
Nmap scan report for 192.168.1.42
Host is up (0.0022s latency).
Not shown: 993 closed tcp ports (conn-refused)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
443/tcp   open  https
7070/tcp  open  realserver
8888/tcp  open  sun-answerbook
9200/tcp  open  wap-wsp
12345/tcp open  netbus

Nmap done: 1 IP address (1 host up) scanned in 1.08 seconds
nickalixspino@MacBook-Pro-de-Nick-3 ~ %
```

5. Vérifier les logs Suricata

- Vérifie les logs :

tail -f /var/log/suricata/eve.json

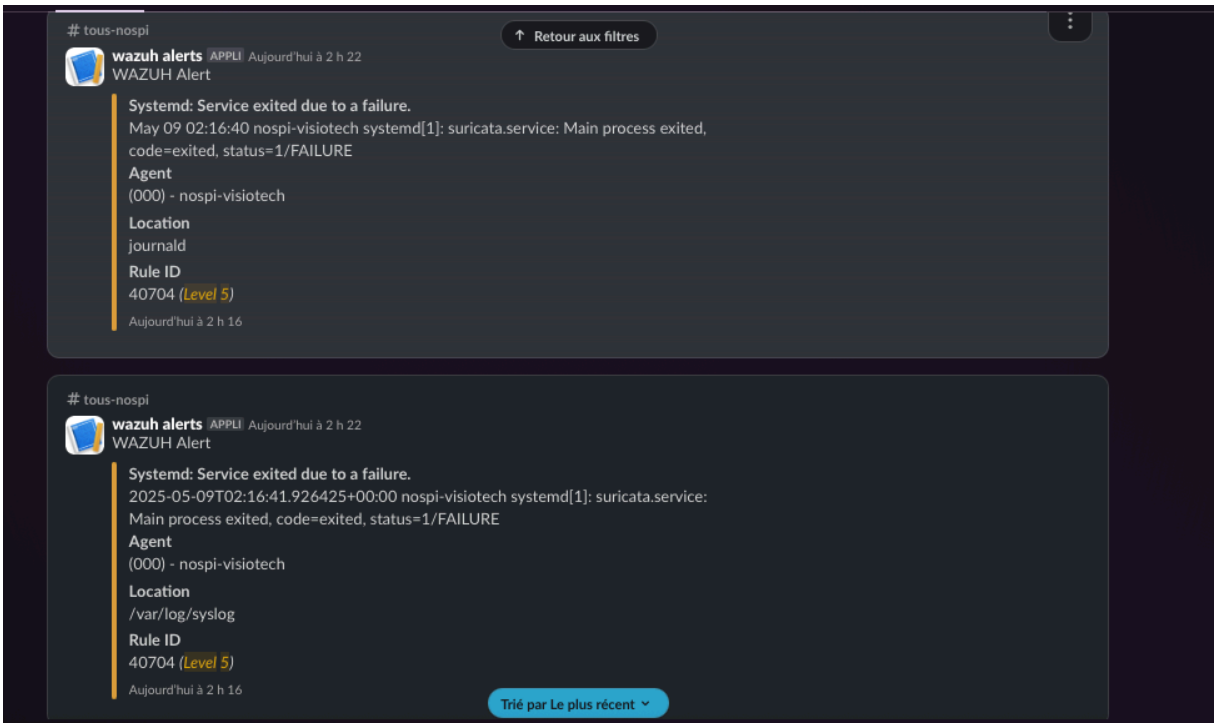
- On devrait voir des événements JSON, par exemple :

{"timestamp":"2025-05-09T01:30:00Z","event_type":"alert","alert":{"signature":"Port scan detected"}}

Vérifier la notification dans Slack

- Va dans ton canal Slack (ex. : `#wazuh-alerts`).

- On devrait voir une notification (les alertes Suricata sont souvent de niveau 5 ou plus) :



7. Détection d'une attaque de Shellshock

Vérifier la vulnérabilité Bash

- Shellshock affecte les anciennes versions de Bash (< 4.3). Vérifie ta version :

bash –version

```
root@nospi-visiotech:/home/nospi# bash --version
GNU bash, version 5.2.21(1)-release (x86_64-pc-linux-gnu)
Copyright (C) 2022 Free Software Foundation, Inc.
Licence GPLv3+ : GNU GPL version 3 ou ultérieure <http://gnu.org/licenses/gpl.html>

Ceci est un logiciel libre ; vous être libre de le modifier et de le redistribuer.
AUCUNE GARANTIE n'est fournie, dans les limites permises par la loi.
```

Si la version est inférieure à 4.3, On peut simuler l'attaque.

Configurer Wazuh pour surveiller les logs

Wazuh peut détecter Shellshock via les logs d'accès Apache si une attaque est simulée via une requête HTTP.

- Assure-toi que `/var/log/apache2/access.log` est surveillé :

<localfile>

<log_format>apache</log_format>

<location>/var/log/apache2/access.log</location>

</localfile>

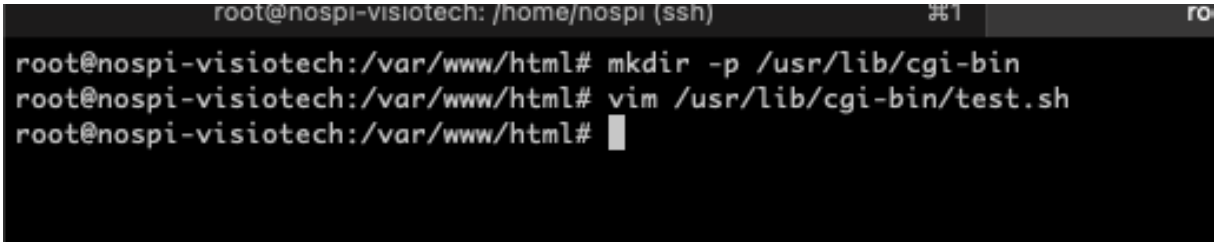
(Déjà configuré précédemment.)

Simuler une attaque Shellshock

- Crée un script CGI vulnérable dans `/usr/lib/cgi-bin/` :

vim mkdir -p /usr/lib/cgi-bin

vim /usr/lib/cgi-bin/test.sh



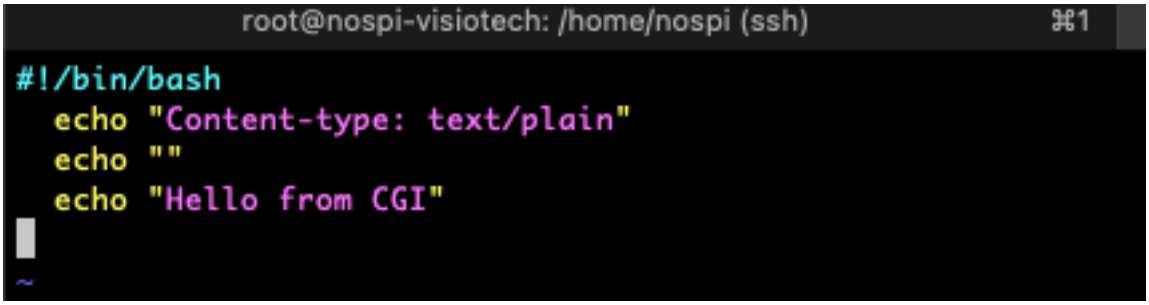
- Ajoute :

#!/bin/bash

echo "Content-type: text/plain"

echo ""

echo "Hello from CGI"



- Rends-le exécutable :

chmod +x /usr/lib/cgi-bin/test.sh

```
root@nospi-visiotech:/var/www/html# chmod +x /usr/lib/cgi-bin/test.sh
root@nospi-visiotech:/var/www/html#
```

- Redémarre Apache2 :

systemctl restart apache2

- Simule l'attaque avec `curl` sur une autre machine :

curl -H "User-Agent: () { :;}; /bin/cat /etc/passwd" http://192.168.1.42/cgi-bin/test.sh

```
root@nospi-server:/home/nospi#   curl -H "User-Agent: () { :;}; /bin/cat /etc/passwd" http://192.168.1.42/cgi-bin/test.sh
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL was not found on this server.</p>
<hr>
<address>Apache/2.4.58 (Ubuntu) Server at 192.168.1.42 Port 80</address>
</body></html>
root@nospi-server:/home/nospi#
```

- Cela envoie une requête malveillante qui exploite Shellshock pour exécuter `/bin/cat /etc/passwd`.

Vérifier les logs

- Vérifie les logs d'accès :

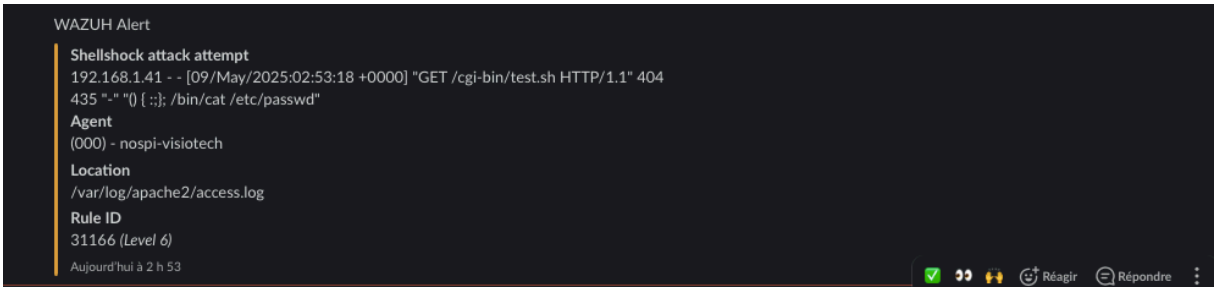
tail -f /var/log/apache2/access.log

- On devrait voir la requête suspecte.

```
192.168.1.41 - - [09/May/2025:02:53:18 +0000] "GET /cgi-bin/test.sh HTTP/1.1" 404 435 "-" "() { :;}; /bin/cat /etc/passwd"
```

Vérifier la notification dans Slack

Allez dans le canal Slack (ex. : `#wazuh-alerts`).



8. Détection des vulnérabilités

Activer le module de vulnérabilités

Assure-toi que le module est activé dans `/var/ossec/etc/ossec.conf` :

```
vim /var/ossec/etc/ossec.conf
```

vérifie :

```
<vulnerability-detector>
  <enabled>yes</enabled>
  <interval>1d</interval> <!-- Vérifie tous les jours -->
  <ignore> <!-- Ignore des packages si besoin -->
  </ignore>
</vulnerability-detector>
```

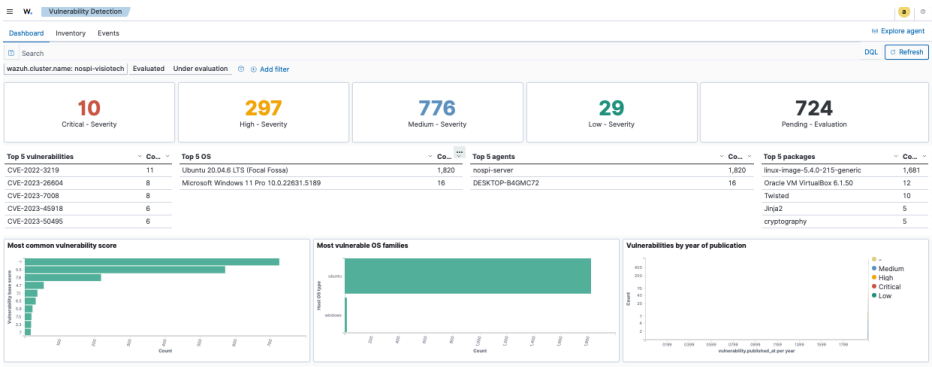
```
<vulnerability-detection>
  <enabled>yes</enabled>
  <index-status>yes</index-status>
  <feed-update-interval>60m</feed-update-interval>
</vulnerability-detection>
```

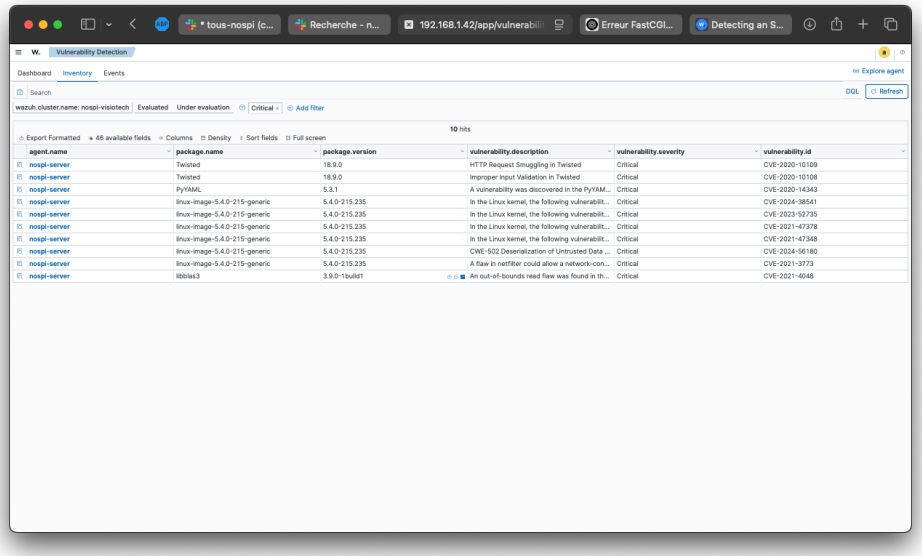
Vérifier les vulnérabilités

- Aucun test simulé n’est nécessaire ; Wazuh scannera automatiquement les logiciels installés.
- Attendre un peu (jusqu’à l’intervalle de 1 jour, ou force un scan manuel).

Vérifier la détection dans Wazuh Dashboard

- Accède au Wazuh Dashboard (`https://localhost:443`).
- Va dans Vulnerabilities.
- Cherche des vulnérabilités listées (ex. : logiciels obsolètes comme une vieille version d’Apache).

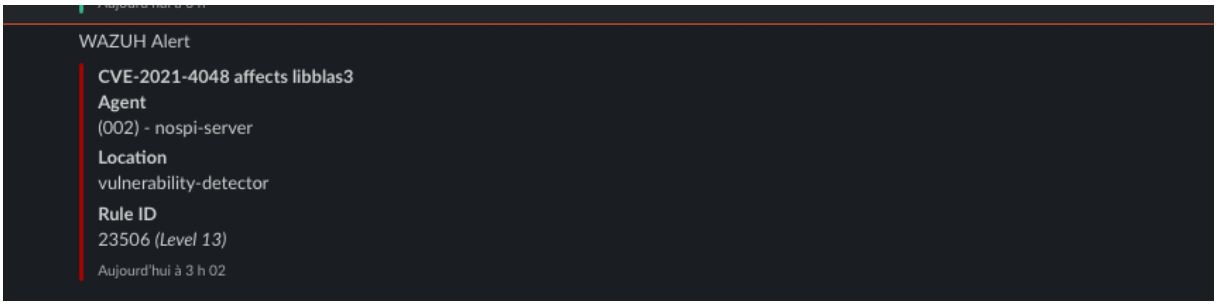




Vérifier la notification dans Slack

Aller dans le canal Slack (ex. : `#wazuh-alerts`).

- On devrait voir une notification pour une vulnérabilité détectée (niveau variable, souvent 7 ou plus) :



TP6 : Prometheus et Grafana

1. Expliquer le rôle des deux logiciels

Prometheus : C'est un outil de surveillance et d'alerte open-source qui collecte des métriques (données temporelles) à partir de systèmes ou d'applications via un modèle de "scraping" HTTP. Il stocke ces métriques dans une base de données temporelle et permet de les interroger avec son langage PromQL.
Grafana : C'est une plateforme de visualisation qui se connecte à des sources de données comme Prometheus pour créer des tableaux de bord interactifs. Elle permet de visualiser les métriques sous forme de graphiques et de configurer des alertes.

2. Présenter l'objectif et l'architecture du TP (utilisation de 3 machines)

Objectif

L'objectif de ce TP est de mettre en place une solution de surveillance avec Prometheus et Grafana pour collecter, analyser et visualiser des métriques provenant de trois machines (un serveur et deux agents). Cela inclut l'installation de Prometheus et Grafana, la configuration des agents (Linux et Windows) pour exporter des métriques, l'interconnexion des agents avec le serveur, la surveillance spécifique d'Apache2, Docker, et MySQL, ainsi que l'analyse d'une attaque DDoS simulée via les tableaux de bord Grafana.

Architecture

Serveur (Ubuntu 20) : Héberge Prometheus pour collecter et stocker les métriques, et Grafana pour visualiser les données. Collecte également ses propres métriques via un node_exporter.
Agent Linux (Ubuntu 20) : Exécute un node_exporter pour exposer les métriques système (CPU, mémoire, etc.) et envoie ces données au serveur Prometheus.
Agent Windows (Windows 10) : Exécute un windows_exporter pour exposer les métriques système et envoie ces données au serveur Prometheus.
Interconnexion réseau : Les agents exposent leurs métriques sur des ports (ex. : 9100 pour node_exporter, 9182 pour windows_exporter), et Prometheus les "scrape" via HTTP.

➢ **Applications surveillées :**

Apache2 : Sur le serveur ou un agent, surveillé avec un exporter Prometheus.

Docker : Sur le serveur ou un agent, surveillé avec un exporter Docker.

MySQL : Sur le serveur ou un agent, surveillé avec un exporter MySQL.

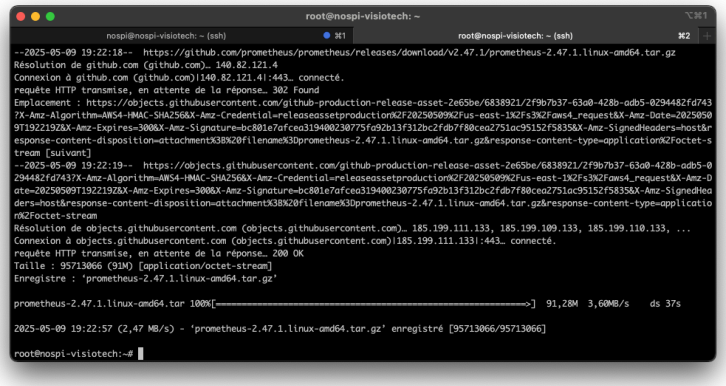
Grafana : Se connecte à Prometheus pour afficher les métriques sous forme de tableaux de bord.

3. Installer et configurer Prometheus et vérifier qu'il récupère les données sur le serveur local

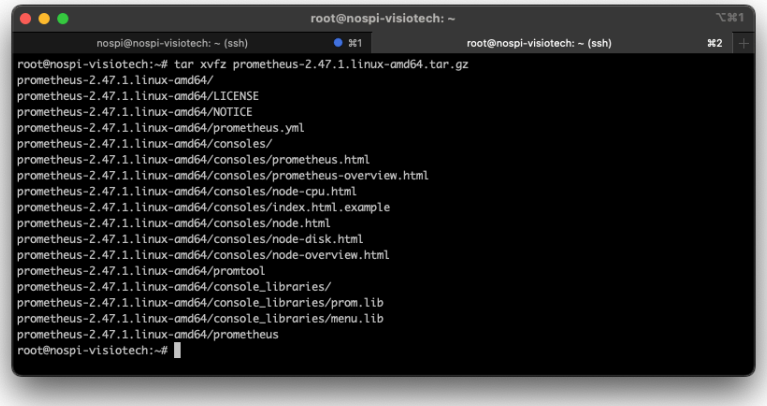
Installation de Prometheus

Sur le server visiotech :

wget https://github.com/prometheus/prometheus/releases/download/v2.47.1/prometheus-2.47.1.linux-amd64.tar.gz



tar xvfz prometheus-2.47.1.linux-amd64.tar.gz



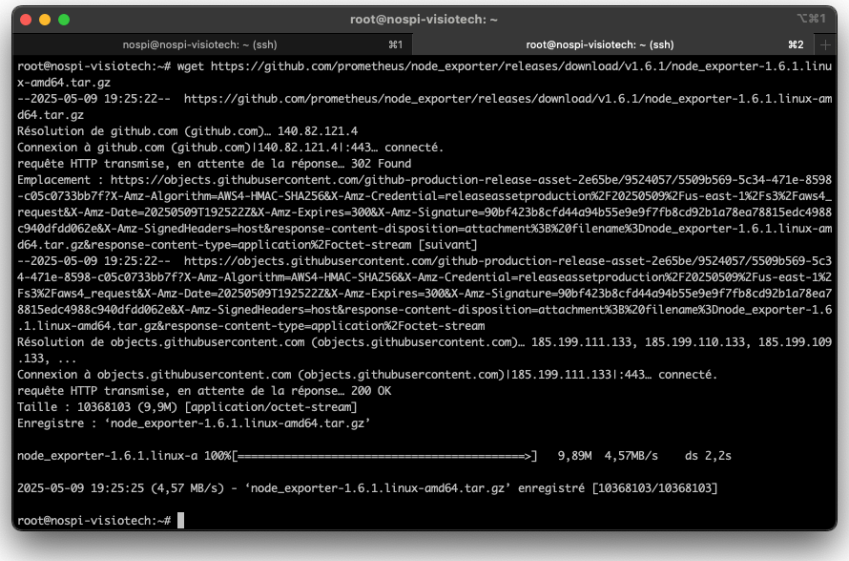
mv prometheus-2.47.1.linux-amd64 /opt/prometheus

```
root@nospi-visiotech:~# mv prometheus-2.47.1.linux-amd64 /opt/prometheus
root@nospi-visiotech:~#
```

Installer node_exporter sur le serveur

node_exporter collecte les métriques système (CPU, mémoire, etc.) :

wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz



tar xvfz node_exporter-1.6.1.linux-amd64.tar.gz

```
root@nospi-visiotech:~# tar xvfz node_exporter-1.6.1.linux-amd64.tar.gz
node_exporter-1.6.1.linux-amd64/
node_exporter-1.6.1.linux-amd64/NOTICE
node_exporter-1.6.1.linux-amd64/node_exporter
node_exporter-1.6.1.linux-amd64/LICENSE
root@nospi-visiotech:~#
```

mv node_exporter-1.6.1.linux-amd64 /opt/node_exporter

```
root@nospi-visiotech:~# mv node_exporter-1.6.1.linux-amd64 /opt/node_exporter
root@nospi-visiotech:~#
```

/opt/node_exporter/node_exporter &

```
root@nospi-visiotech:~# /opt/node_exporter/node_exporter &
[1] 364498
root@nospi-visiotech:~# ts=2025-05-09T19:30:47.783Z caller=node_exporter.go:180 level=info msg="Starting node_exporter" version="(version=1.6.1, branch=HEAD, r
a8233f3ffb55afcedbb63b8d84)"
ts=2025-05-09T19:30:47.784Z caller=node_exporter.go:181 level=info msg="Build context" build_context="(go=go1.20.6, platform=linux/amd64, user=root@586879db11c
:52, tags=netgo osusergo static_build)"
ts=2025-05-09T19:30:47.784Z caller=node_exporter.go:183 level=warn msg="Node Exporter is running as root user. This exporter is designed to run as unprivileged
uired."
ts=2025-05-09T19:30:47.786Z caller=filesystem_common.go:111 level=info collector=filesystem msg="Parsed flag --collector.filesystem.mount-points-exclude" flag=
tials/.+|sys|var/lib/docker/.+|var/lib/containers/storage/.+)($|/)
ts=2025-05-09T19:30:47.786Z caller=filesystem_common.go:113 level=info collector=filesystem msg="Parsed flag --collector.filesystem.fs-types-exclude" flag=^(au
group2?|configfs|debugfs|devpts|devtmpfs|fusectl|hugetlbfs|iso9660|mqueue|nsfs|overlay|proc|procfs|pstore|rpc_pipefs|securityfs|selinuxfs|squashfs|sysfs|tracef
ts=2025-05-09T19:30:47.791Z caller=diskstats_common.go:111 level=info collector=diskstats msg="Parsed flag --collector.diskstats.device-exclude" flag=^(ram|loo
nvme\d+n\d+p)\d+$
ts=2025-05-09T19:30:47.791Z caller=node_exporter.go:110 level=info msg="Enabled collectors"
ts=2025-05-09T19:30:47.791Z caller=node_exporter.go:117 level=info collector=arp
```

- Vérifie que node_exporter fonctionne (port 9100) :

curl http://localhost:9100/metrics

Il doit y avoir un résultat semblable à la capture

```
# TYPE promhttp_metric_handler_errors_total counter
promhttp_metric_handler_errors_total{cause="encoding"} 0
promhttp_metric_handler_errors_total{cause="gathering"} 0
# HELP promhttp_metric_handler_requests_in_flight Current number of scrapes being served.
# TYPE promhttp_metric_handler_requests_in_flight gauge
promhttp_metric_handler_requests_in_flight 1
# HELP promhttp_metric_handler_requests_total Total number of scrapes by HTTP status code.
# TYPE promhttp_metric_handler_requests_total counter
promhttp_metric_handler_requests_total{code="200"} 1
promhttp_metric_handler_requests_total{code="500"} 0
promhttp_metric_handler_requests_total{code="503"} 0
root@nospi-visiotech:~#
```

Configurer Prometheus

Éditer le fichier de configuration de Prometheus :

vim /opt/prometheus/prometheus.yml

- On decommente pour scraper le serveur local et on modifie l'adresse et le job name (facultatif) :

```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

# Alertmanager configuration
alerting:
  alertmanagers:
    - static_configs:
        - targets:
          # - alertmanager:9093

# Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
rule_files:
  # - "first_rules.yml"
  # - "second_rules.yml"

# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
    - job_name: "prometheus server_local"

    # metrics_path defaults to '/metrics'
    # scheme defaults to 'http'.

      static_configs:
       - targets: ["localhost:9100"]
```

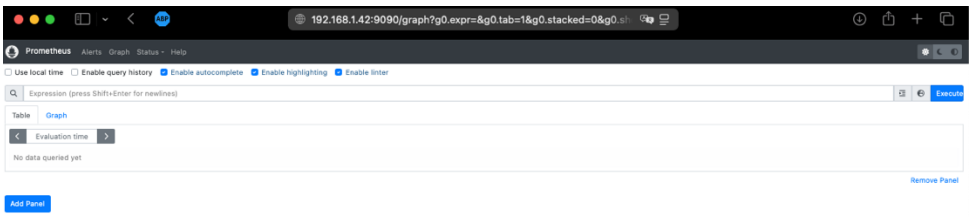
- Lance Prometheus :

/opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml &

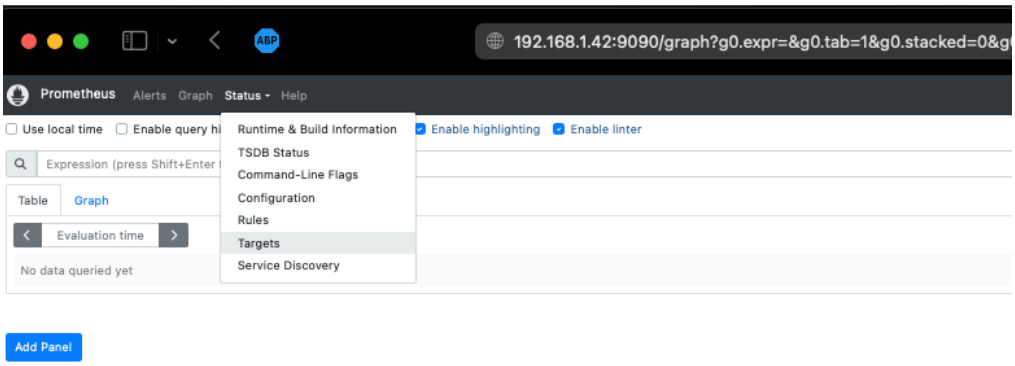
```
root@nospi-visiotech:~# /opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml &
[2] 364639
root@nospi-visiotech:~# ts=2025-05-09T19:44:43.300Z caller=main.go:539 level=info msg="No time or size retention was set so using the default time retention" duration=15d
ts=2025-05-09T19:44:43.301Z caller=main.go:583 level=info msg="Starting Prometheus Server" mode=server version="(version=2.47.1, branch=HEAD, revision=c4d1a8beff37cc004f1dc4ab9d2e73193f51aaeb)"
ts=2025-05-09T19:44:43.301Z caller=main.go:588 level=info build_context="(go=go1.21.1, platform=linux/amd64, user=root@4829330363be, date=20231004-10:31:16, tags=netgo,builtinassets,stringlabels)"
ts=2025-05-09T19:44:43.301Z caller=main.go:589 level=info host_details="(Linux 6.11.0-25-generic #25~24.04.1-Ubuntu SMP PREEMPT_DYNAMIC Tue Apr 15 17:20:50 UTC 2 x86_64 nospi-visiotech (none))"
ts=2025-05-09T19:44:43.301Z caller=main.go:590 level=info fd_limits="(soft=1048576, hard=1048576)"
ts=2025-05-09T19:44:43.301Z caller=main.go:591 level=info vm_limits="(soft=unlimited, hard=unlimited)"
ts=2025-05-09T19:44:43.308Z caller=web.go:566 level=info component=web msg="Start listening for connections" address=0.0.0.0:9090
ts=2025-05-09T19:44:43.310Z caller=main.go:1024 level=info msg="Starting TSDB ..."
ts=2025-05-09T19:44:43.319Z caller=tls_config.go:274 level=info component=web msg="Listening on" address=[::]:9090
ts=2025-05-09T19:44:43.319Z caller=tls_config.go:277 level=info component=web msg="TLS is disabled." http2=false address=[::]:9090
ts=2025-05-09T19:44:43.329Z caller=head.go:600 level=info component=tsdb msg="Replaying on-disk memory mappable chunks if any"
ts=2025-05-09T19:44:43.329Z caller=head.go:681 level=info component=tsdb msg="On-disk memory mappable chunks replay completed" duration=4.892µs
ts=2025-05-09T19:44:43.329Z caller=head.go:689 level=info component=tsdb msg="Replaying WAL, this may take a while"
ts=2025-05-09T19:44:43.330Z caller=head.go:760 level=info component=tsdb msg="WAL segment loaded" segment=0 maxSegment=0
ts=2025-05-09T19:44:43.331Z caller=head.go:797 level=info component=tsdb msg="WAL replay completed" checkpoint_replay_duration=166.777µs wal_replay_duration=1.167589ms wbl_replay_duration=436ns total_replay_duration=1
.4531ms
ts=2025-05-09T19:44:43.340Z caller=main.go:1045 level=info fs_type=EXT4_SUPER_MAGIC
ts=2025-05-09T19:44:43.340Z caller=main.go:1048 level=info msg="TSDB started"
ts=2025-05-09T19:44:43.340Z caller=main.go:1229 level=info msg="Loading configuration file" filename=/opt/prometheus/prometheus.yml
ts=2025-05-09T19:44:43.342Z caller=main.go:1266 level=info msg="Completed loading of configuration file" filename=/opt/prometheus/prometheus.yml totalDuration=1.633547ms db_storage=1.768µs remote_storage=3.024µs web_h
andler=924ns query_engine=1.693µs scrape=632.642µs scrape_sd=76.726µs notify=102.737µs notify_sd=52.86µs rules=2.542µs tracing=10.409µs
ts=2025-05-09T19:44:43.342Z caller=main.go:1009 level=info msg="Server is ready to receive web requests."
ts=2025-05-09T19:44:43.342Z caller=manager.go:1009 level=info component="rule manager" msg="Starting rule manager..."
```

Vérifier la collecte

- Accède à l'interface web de Prometheus (port 9090) :
    - Ouvre http://192.168.1.42:9090 dans le navigateur.



- On va dans "Status" > "Targets" et vérifie que server_local est "UP".

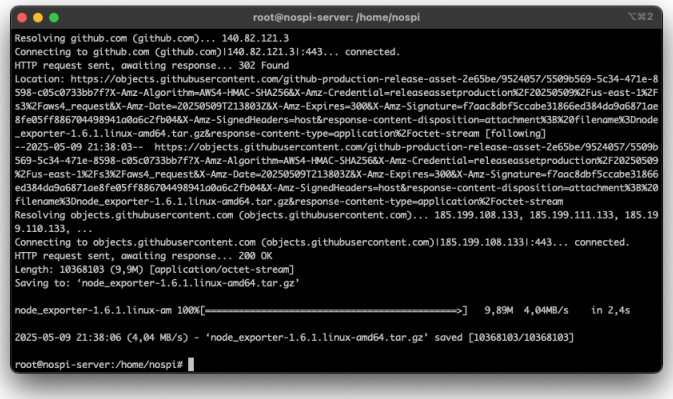


4. Installer et configurer les nœuds sur deux agents (Linux et Windows) et vérifier que les données sont récupérées en local

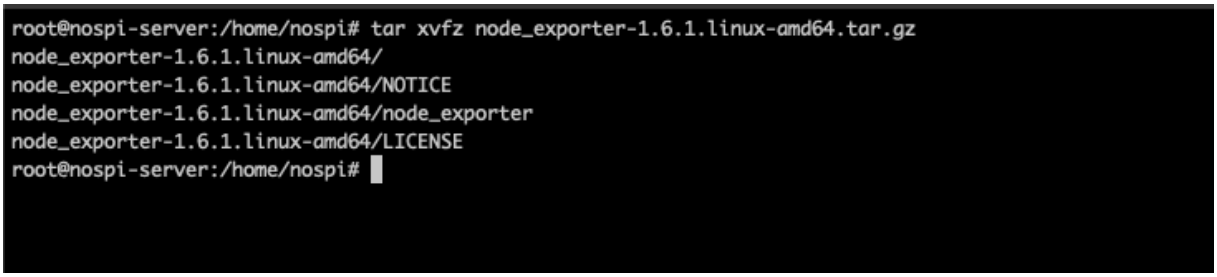
Agent Linux (Ubuntu 20)

- Installe node_exporter :

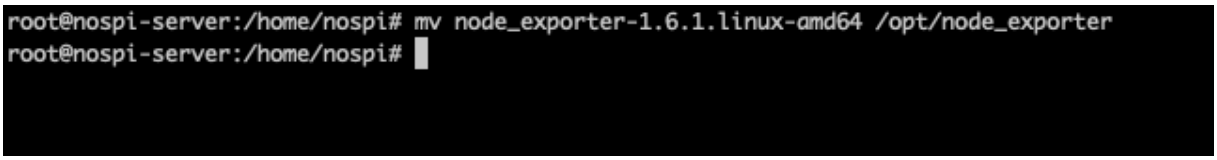
wget https://github.com/prometheus/node_exporter/releases/download/v1.6.1/node_exporter-1.6.1.linux-amd64.tar.gz



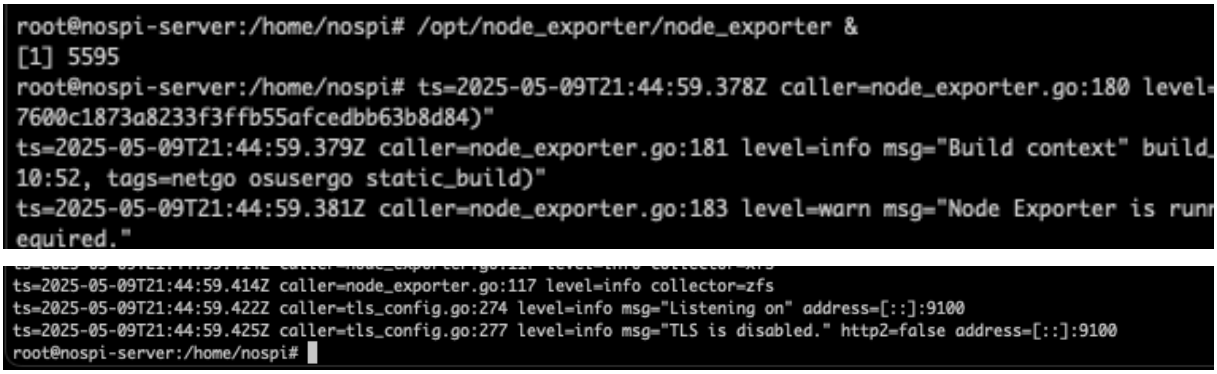
tar xvfz node_exporter-1.6.1.linux-amd64.tar.gz



mv node_exporter-1.6.1.linux-amd64 /opt/node_exporter

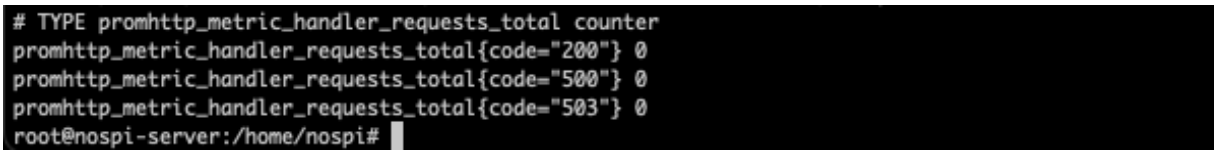


/opt/node_exporter/node_exporter &



- Vérifie localement (port 9100) :

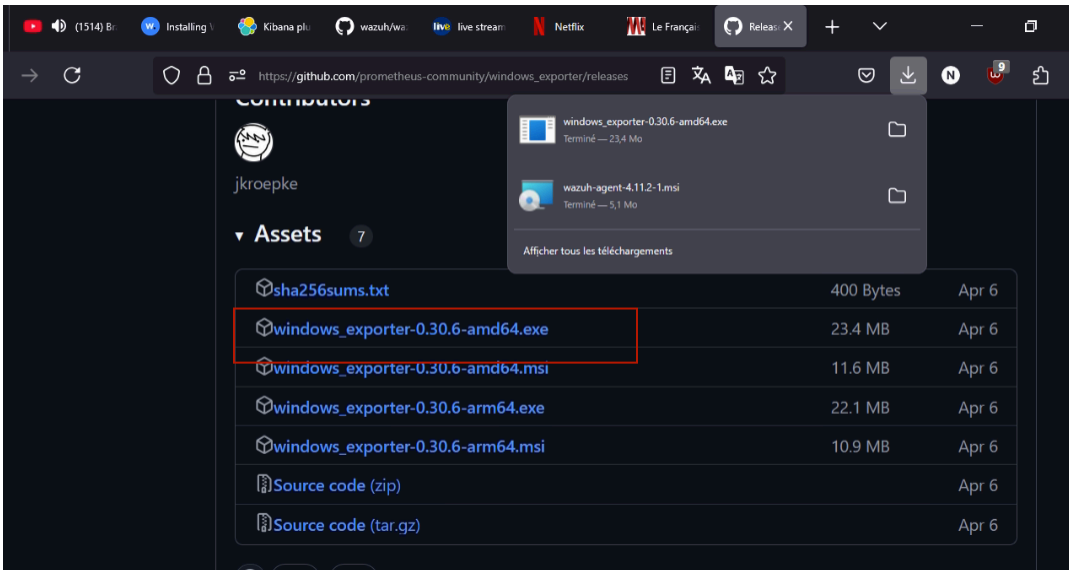
curl http://localhost:9100/metrics



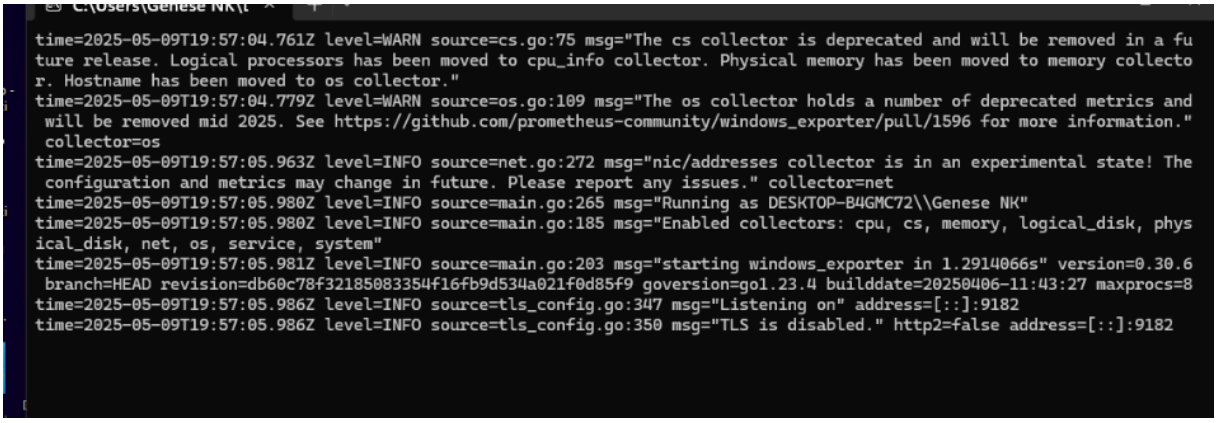
Agent Windows (Windows 11)

- Télécharge windows_exporter :

- Télécharge le fichier MSI depuis https://github.com/prometheus-community/windows_exporter/releases (ex. : windows_exporter-0.30.6-amd64.msi) derniere version stable.



- On installe :



- Vérifie localement (port 9182) :
  - Ouvre une invite de commande et utilise :

curl http://localhost:9182/metrics

```
PS C:\Users\Genese NK> curl http://localhost:9182/metrics


StatusCode        : 200
StatusDescription : OK
Content           : # HELP go_build_info Build information about the main Go module.
                    # TYPE go_build_info gauge
                    go_build_info{checksum="",path="github.com/prometheus-community/windows_exporter",version="(devel)"
                    } 1
                    # HEL...
RawContent        : HTTP/1.1 200 OK
                    Process-Start-Time-Unix: 1746820624
                    Transfer-Encoding: chunked
                    Content-Type: text/plain; version=0.0.4; charset=utf-8; escaping=underscores
                    Date: Fri, 09 May 2025 19:58:09 GMT

                    #...
Forms             : {}
Headers           : {[Process-Start-Time-Unix, 1746820624], [Transfer-Encoding, chunked], [Content-Type, text/plain;
                    version=0.0.4; charset=utf-8; escaping=underscores], [Date, Fri, 09 May 2025 19:58:09 GMT]}
Images            : {}
InputFields       : {}
Links             : {}
ParsedHtml        : mshtml.HTMLDocumentClass
RawContentLength  : 371857
```

5. Assurer l'interconnexion entre les agents (Linux et Windows) et montrer les données recueillies sur le serveur Prometheus

Configurer Prometheus pour scraper les agents

- Sur le serveur, modifie /opt/prometheus/prometheus.yml :

vim /opt/prometheus/prometheus.yml

- On ajoute les agents :

```
- job_name: 'agent_linux'
  static_configs:
    - targets: ['192.168.1.49:9100']

- job_name: 'agent_windows'
  static_configs:
    - targets: ['192.168.1.50:9182']
```

- Redémarre Prometheus :

pkill prometheus

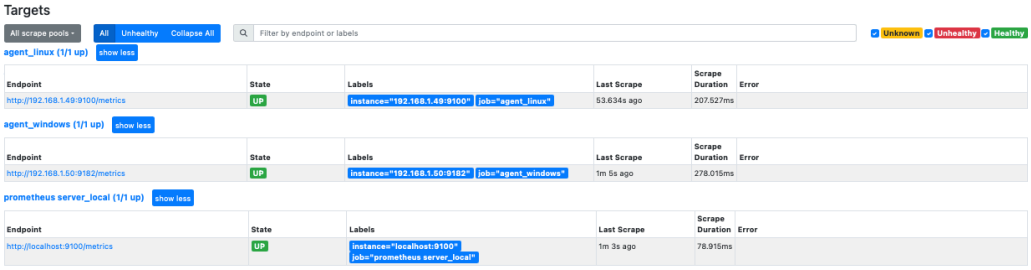
/opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml &

```
root@nospi-visiotech:/home/nospi# pkill prometheus
root@nospi-visiotech:/home/nospi# vim /opt/prometheus/prometheus.yml
root@nospi-visiotech:/home/nospi# /opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml &
[1] 367070
root@nospi-visiotech:/home/nospi# ts=2025-05-09T21:59:02.819Z caller=main.go:539 level=info msg="No time or size retention was set so using the default
e retention" duration=15d
ts=2025-05-09T21:59:02.819Z caller=main.go:583 level=info msg="Starting Prometheus Server" mode=server version="(version=2.47.1, branch=HEAD, revision=
a8beff37cc004f1dc4ab9d2e73193f51aaeb)"
ts=2025-05-09T21:59:02.819Z caller=main.go:588 level=info build_context="(go=go1.21.1, platform=linux/amd64, user=root@4829330363be, date=20231004-10:31
, tags=netgo,builtinassets,stringlabels)"
ts=2025-05-09T21:59:02.819Z caller=main.go:589 level=info host_details="(Linux 6.11.0-25-generic #25~24.04.1-Ubuntu SMP PREEMPT_DYNAMIC Tue Apr 15 17:20
 UTC 2 x86_64 nospi-visiotech (none))"
ts=2025-05-09T21:59:02.819Z caller=main.go:590 level=info fd_limits="(soft=1048576, hard=1048576)"
```

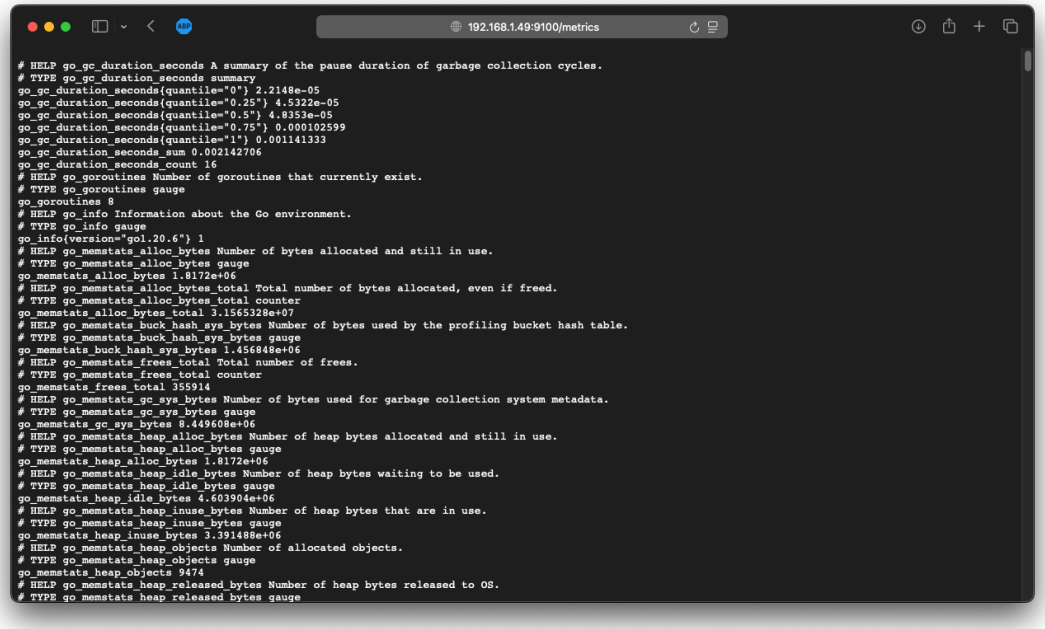
Vérifier la collecte

Ouvre http://192.168.1.42:9090 > "Status" > "Targets".

Vérifie que l'agent_linux et agent_windows sont "UP".



Interroger une métrique (ex. : node_cpu_seconds_total pour Linux, windows_cpu_time_total pour Windows).



6. Configurer Prometheus comme service

D’abord tuer le processus Prometheus

```
root@nospi-visiotech:/home/nospi# pkill prometheus
ts=2025-05-09T22:13:32.399Z caller=main.go:859 level=warn msg="Received SIGTERM, exiting gracefully..."
ts=2025-05-09T22:13:32.399Z caller=main.go:883 level=info msg="Stopping scrape discovery manager..."
ts=2025-05-09T22:13:32.399Z caller=main.go:897 level=info msg="Stopping notify discovery manager..."
ts=2025-05-09T22:13:32.399Z caller=manager.go:1023 level=info component="rule manager" msg="Stopping rule manager..."
ts=2025-05-09T22:13:32.399Z caller=manager.go:1033 level=info component="rule manager" msg="Rule manager stopped"
ts=2025-05-09T22:13:32.399Z caller=main.go:934 level=info msg="Stopping scrape manager..."
ts=2025-05-09T22:13:32.399Z caller=main.go:893 level=info msg="Notify discovery manager stopped"
ts=2025-05-09T22:13:32.399Z caller=main.go:879 level=info msg="Scrape discovery manager stopped"
ts=2025-05-09T22:13:32.400Z caller=main.go:926 level=info msg="Scrape manager stopped"
root@nospi-visiotech:/home/nospi# ts=2025-05-09T22:13:32.494Z caller=notifier.go:603 level=info component=notifier msg="Stopping notification manager..."
ts=2025-05-09T22:13:32.494Z caller=main.go:1155 level=info msg="Notifier manager stopped"
ts=2025-05-09T22:13:32.494Z caller=main.go:1167 level=info msg="See you next time!"

[1]+  Fini                  /opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml
root@nospi-visiotech:/home/nospi#
```

Créer un fichier de service systemd :

vim /etc/systemd/system/prometheus.service

Ajouter :

```
[Unit]
Description=Prometheus Monitoring
Wants=network-online.target
After=network-online.target

[Service]
User=root
ExecStart=/opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml
Restart=always

[Install]
WantedBy=multi-user.target
~
~
```

Active le service :

systemctl daemon-reload

systemctl start prometheus

systemctl enable prometheus

```
root@nospi-visiotech:/home/nospi# vim /etc/systemd/system/prometheus.service
root@nospi-visiotech:/home/nospi# systemctl daemon-reload
root@nospi-visiotech:/home/nospi# systemctl start prometheus
root@nospi-visiotech:/home/nospi# systemctl enable prometheus
Created symlink /etc/systemd/system/multi-user.target.wants/prometheus.service → /etc/systemd/system/prometheus.service.
root@nospi-visiotech:/home/nospi#
```

Redémarrer & Vérifier :

systemctl status prometheus

```
root@nospi-visiotech:/home/nospi# systemctl restart prometheus
root@nospi-visiotech:/home/nospi# systemctl status prometheus
● prometheus.service - Prometheus Monitoring
     Loaded: loaded (/etc/systemd/system/prometheus.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-05-09 22:14:52 GMT; 24s ago
   Main PID: 368948 (prometheus)
      Tasks: 10 (limit: 14119)
     Memory: 30.9M (peak: 31.6M)
        CPU: 493ms
     CGroup: /system.slice/prometheus.service
             └─368948 /opt/prometheus/prometheus --config.file=/opt/prometheus/prometheus.yml

mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.124Z caller=head.go:681 level=info component=tsdb msg="On-disk memory mappable >
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.124Z caller=head.go:689 level=info component=tsdb msg="Replaying WAL, this may >
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.126Z caller=head.go:760 level=info component=tsdb msg="WAL segment loaded" segm>
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.126Z caller=head.go:797 level=info component=tsdb msg="WAL replay completed" ch>
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.135Z caller=main.go:1045 level=info fs_type=EXT4_SUPER_MAGIC
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.135Z caller=main.go:1048 level=info msg="TSDB started"
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.135Z caller=main.go:1229 level=info msg="Loading configuration file" filename=/>
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.137Z caller=main.go:1266 level=info msg="Completed loading of configuration fil>
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.137Z caller=main.go:1009 level=info msg="Server is ready to receive web request>
mai 09 22:14:53 nospi-visiotech prometheus[368948]: ts=2025-05-09T22:14:53.138Z caller=manager.go:1009 level=info component="rule manager" msg="Starting r>
lines 1-20/20 (END)
```

7. Configurer Grafana pour visualiser les données des agents collectées par le serveur Prometheus et commenter les résultats obtenus

Installer grafana sur le serveur visiotech :

Installer les prerequis

apt-get install -y apt-transport-https software-properties-common wget

```
root@nospi-visiotech:/home/nospi# apt-get install -y apt-transport-https software-properties-common wget
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
apt-transport-https est déjà la version la plus récente (2.7.14build2).
software-properties-common est déjà la version la plus récente (0.99.49.2).
```

Import the GPG key:

mkdir -p /etc/apt/keyrings/

wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

```
root@nospi-visiotech:/home/nospi# sudo mkdir -p /etc/apt/keyrings/
wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null
root@nospi-visiotech:/home/nospi#
```

echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

```
root@nospi-visiotech:/home/nospi# echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.l
ist.d/grafana.list
deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main
root@nospi-visiotech:/home/nospi#
```

echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com beta main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

```
root@nospi-visiotech:/home/nospi# echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com beta main" | sudo tee -a /etc/apt/sources.lis
t.d/grafana.list
deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com beta main
root@nospi-visiotech:/home/nospi#
```

apt update

```
root@nospi-visiotech:/home/nospi# apt update
Atteint :1 http://sn.archive.ubuntu.com/ubuntu noble InRelease
Atteint :2 http://sn.archive.ubuntu.com/ubuntu noble-updates InRelease
Atteint :3 http://sn.archive.ubuntu.com/ubuntu noble-backports InRelease
Atteint :4 http://security.ubuntu.com/ubuntu noble-security InRelease
Atteint :5 https://deb.anydesk.com all InRelease
Atteint :6 https://dl.google.com/linux/chrome/deb stable InRelease
Atteint :7 https://ngrok-agent.s3.amazonaws.com buster InRelease
Réception de :8 https://apt.grafana.com stable InRelease [7 660 B]
```

apt install grafana

```
root@nospi-visiotech:/home/nospi# sudo apt install grafana
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  musl
Les NOUVEAUX paquets suivants seront installés :
  grafana musl
0 mis à jour, 2 nouvellement installés, 0 à enlever et 155 non mis à jour.
Il est nécessaire de prendre 175 Mo/175 Mo dans les archives.
Après cette opération, 649 Mo d'espace disque supplémentaires seront utilisés.
```

```
info: Pas de création du répertoire personnel « /usr/share/grafana ».
### NOT starting on installation, please execute the following statements to configure grafana to start automatically using syste
 sudo /bin/systemctl daemon-reload
 sudo /bin/systemctl enable grafana-server
### You can start grafana-server by executing
 sudo /bin/systemctl start grafana-server
Traitement des actions différées (« triggers ») pour man-db (2.12.0-4build2) ...
root@nospi-visiotech:/home/nospi#
root@nospi-visiotech:/home/nospi#
```

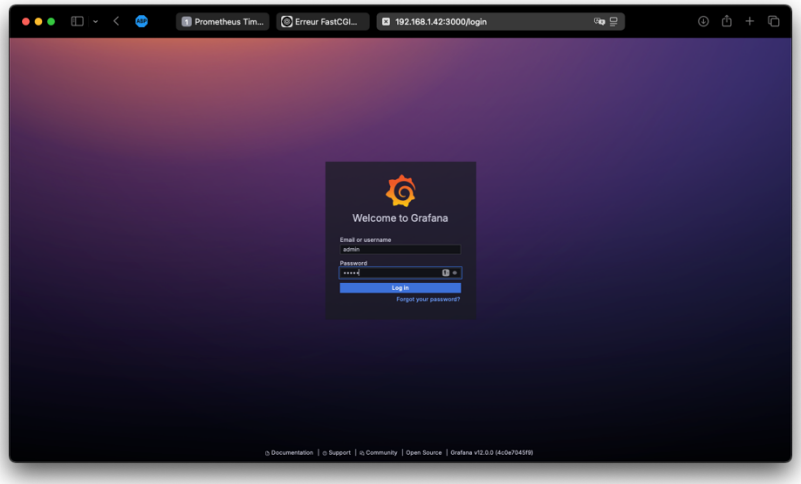
Activer grafana & Vérifier le status

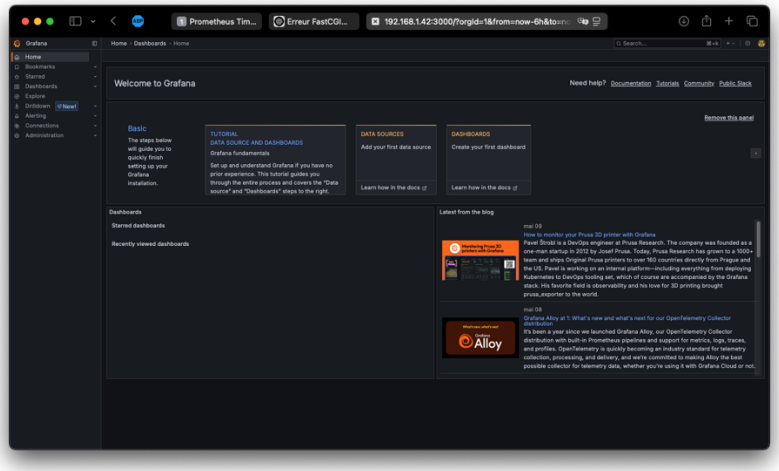
```
root@nospi-visiotech:/home/nospi# systemctl daemon-reload
root@nospi-visiotech:/home/nospi# systemctl enable grafana-server
Synchronizing state of grafana-server.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable grafana-server
Created symlink /etc/systemd/system/multi-user.target.wants/grafana-server.service → /usr/lib/systemd/system/grafana-server.service.
root@nospi-visiotech:/home/nospi# systemctl start grafana-server
root@nospi-visiotech:/home/nospi#
```

```
root@nospi-visiotech:/home/nospi# systemctl status grafana-server
● grafana-server.service - Grafana instance
     Loaded: loaded (/usr/lib/systemd/system/grafana-server.service; enabled; preset: enabled)
     Active: active (running) since Fri 2025-05-09 22:51:32 GMT; 1min 6s ago
       Docs: http://docs.grafana.org
   Main PID: 377921 (grafana)
      Tasks: 8 (limit: 14119)
     Memory: 53.6M (peak: 54.2M)
        CPU: 7.411s
     CGroup: /system.slice/grafana-server.service
             └─377921 /usr/share/grafana/bin/grafana server --config=/etc/grafana/grafana.ini --pidfile=/run/grafana/grafana-server.pid --packa

mai 09 22:52:38 nospi-visiotech grafana[377921]: logger=migrator t=2025-05-09T22:52:38.969925149Z level=info msg="Executing migration" id="Rena
mai 09 22:52:39 nospi-visiotech grafana[377921]: logger=migrator t=2025-05-09T22:52:39.035749614Z level=info msg="Migration successfully execut
```

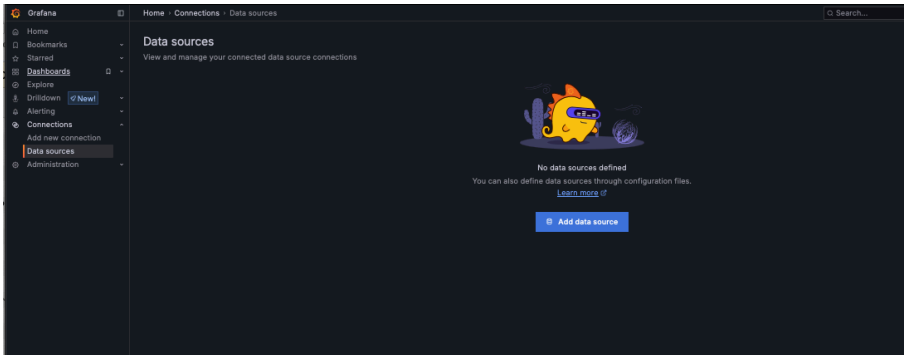
Configurer Grafana

- Accéder à Grafana sur http://192.168.1.42:3000 (identifiants par défaut : admin/admin).

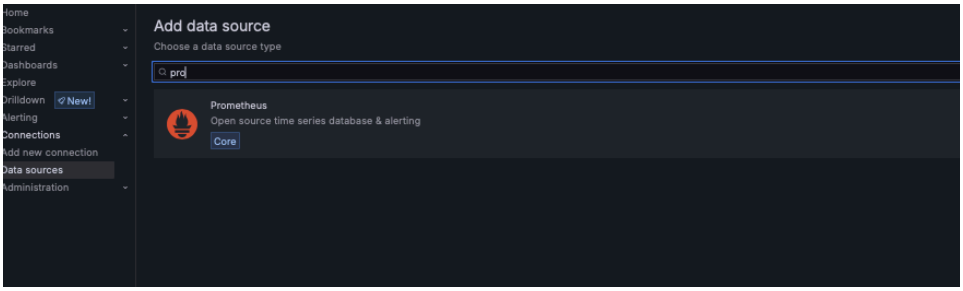




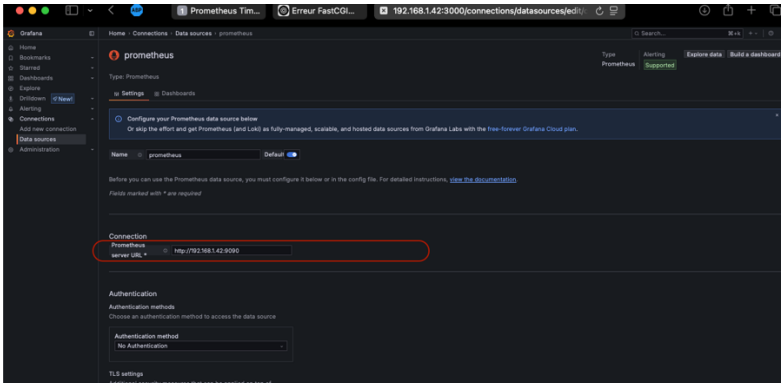
- Ajoute Prometheus comme source de données :
    - Va dans "Configuration" > "Data Sources" > "Add data source".



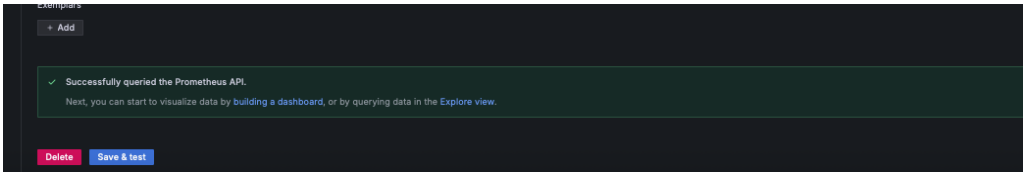
    - Sélectionne "Prometheus".



    - Configure l'URL : http://192.168.1.42:9090.



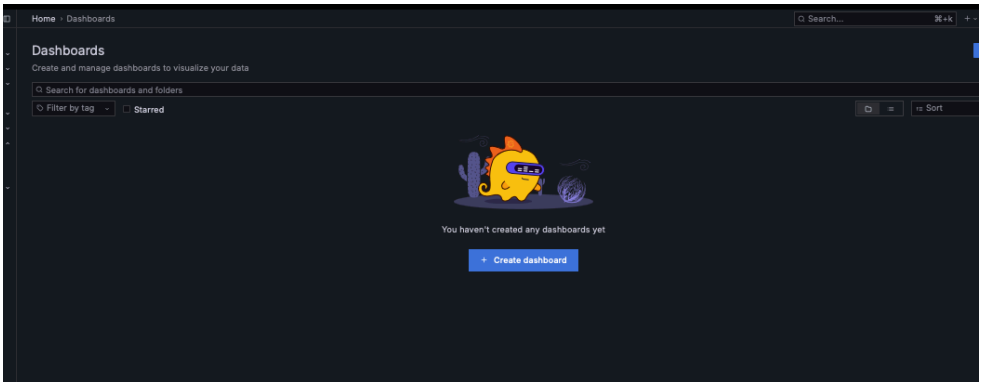
    - Sauvegarder.



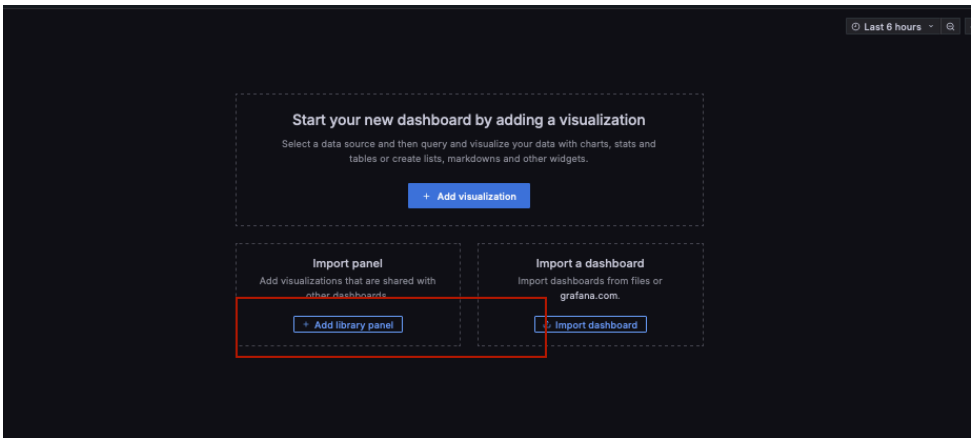
Créer un tableau de bord

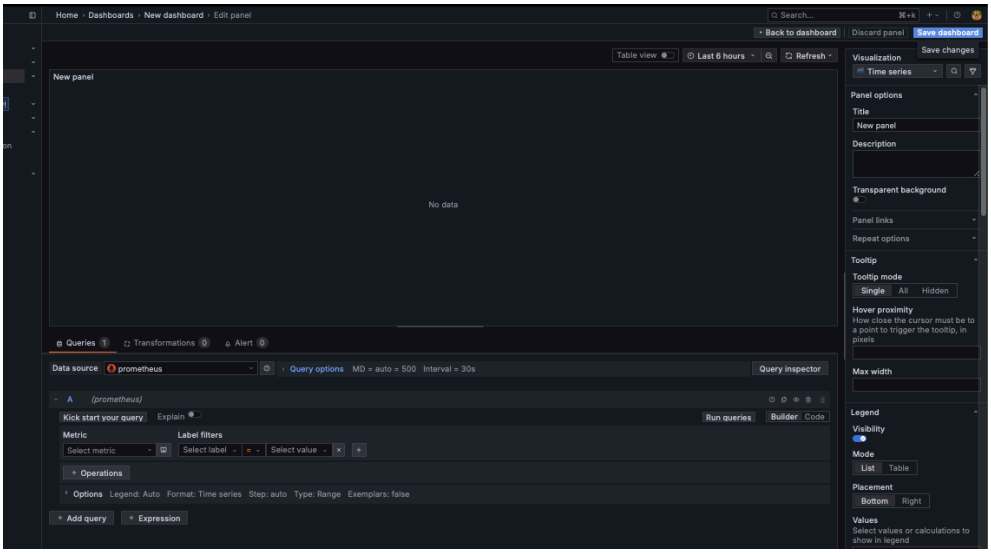
- Crée un tableau de bord :

Va dans "Create" > "Dashboard" > "Add new panel".

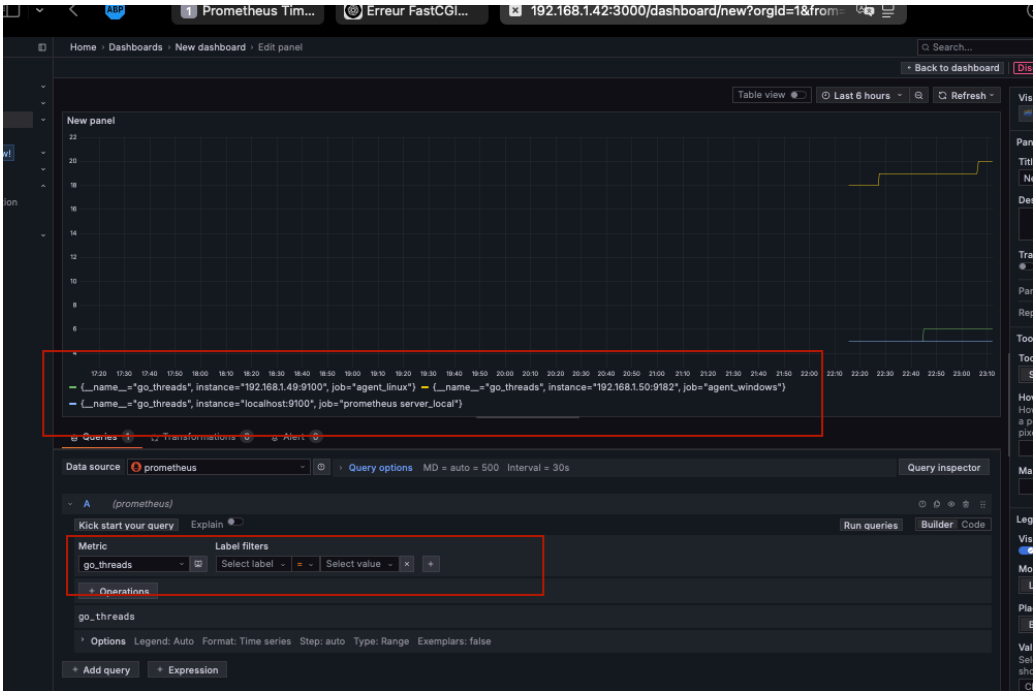


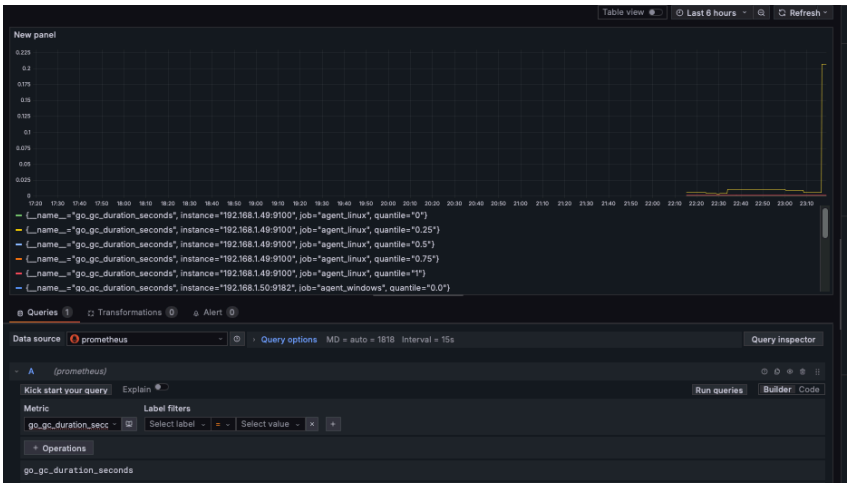
Cliquer plutôt sur add visualisation

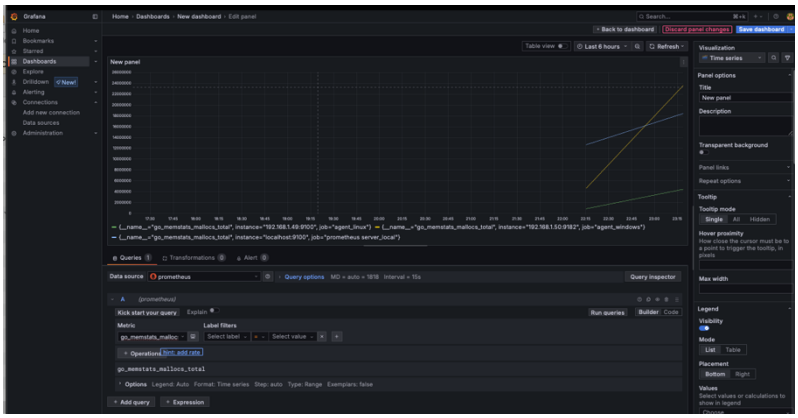




Ajoute une métrique, par exemple go_threads pour l'agent Linux.







Commentaires sur les résultats

- Les graphiques montrent l'utilisation CPU des agents Linux et Windows en temps réel.
- Si les métriques sont cohérentes (pas de données manquantes), l'interconnexion fonctionne correctement.
- Les tableaux de bord permettent une visualisation claire des performances des systèmes.