UNIVERSITÉ NICE SOPHIA ANTIPOLIS
Département Réseaux & Télécoms Année Universitaire 2025-2026

Compte-rendu TP Wazuh

Présenté par : Morgan SALHI & Elouan ODRY

Date de rendu : 15/02/2026

NICE CÔTE D'AZUR

Questions Wazuh

IMPORTANT -> Pour chaque question :

- **Réponse en 1 à 2 phrase, screenshot à l'appui.**
- **Est-ce-que l'IA a été utilisée pour cette question ? Si oui, a-t-elle aidé ?**

Partie 1 – Intégration & Traitement initial de Wazuh

Intégration

Votre infrastructure actuelle a des conteneurs vulnérables et Suricata installé.

Il vous faut maintenant intégrer Wazuh (sous format Docker) à l'infrastructure existante, et déployer ses agents sur les machines existantes.

1. Comment avez-vous intégré les « agents » à chaque conteneur ? (ne fournir les screenshots que pour un seul conteneur) ? Comment avez-vous intégré la partie « serveur » ?

Le serveur Wazuh a été déployé via Docker et raccordé au réseau existant pour permettre la communication avec les conteneurs. Les agents ont été intégrés en installant le paquet `.deb` directement dans chaque conteneur, puis configurés avec l'adresse IP du manager avant d'être démarrés.

Connexion au réseau :

```
root@ubuntu:~# docker network connect single-node_default caching
```

(Preuve de la liaison du conteneur au réseau Docker du manager)

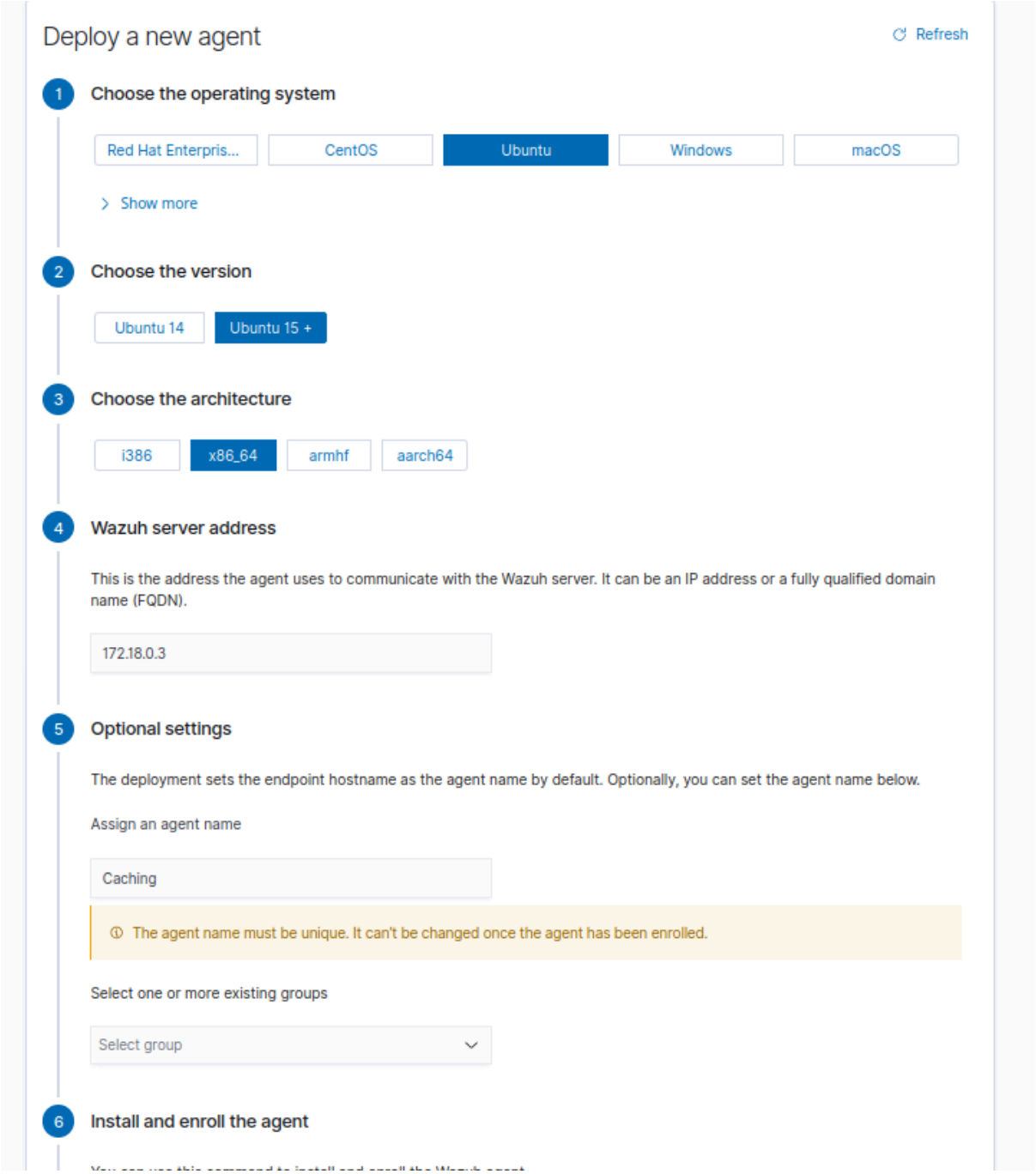
Test de connectivité :

```
root@ubuntu:~# docker exec -it caching ping -c 1 single-node-wazuh.manager-1
PING single-node-wazuh.manager-1 (172.18.0.3) 56(84) bytes of data.
64 bytes from single-node-wazuh.manager-1.single-node_default (172.18.0.3): icmp_seq=1 ttl=64 time=0.069 ms

--- single-node-wazuh.manager-1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.069/0.069/0.069/0.000 ms
```

(Ping vers le manager pour valider la communication)

Préparation du déploiement :



(Interface Wazuh montrant les paramètres de l'agent Ubuntu)

Installation de l'agent :

```
root@6135a8ecff7c:/# curl -so wazuh-agent.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.5.4-1_amd64.deb && sudo WAZUH_MANAGER='single-node-wazuh.manager
1' WAZUH_AGENT_NAME='Caching' dpkg -i ./wazuh-agent.deb
```

(Commande dpkg d'installation dans le conteneur)

Configuration IP :

```
<ossec_config>
  <client>
    <server>
      <address>172.18.0.3</address>
```

(Fichier `ossec.conf` avec l'adresse du serveur)

Lancement du service :

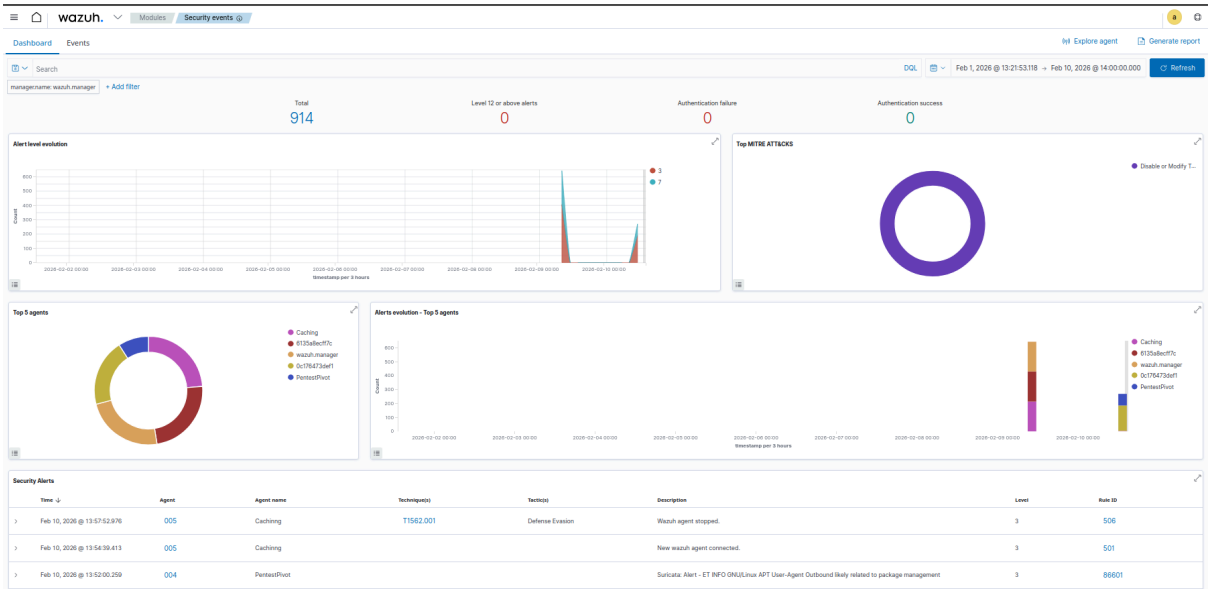
```
root@6135a8ecff7c:/# /var/ossec/bin/wazuh-control start
Starting Wazuh v4.5.4...
Started wazuh-execd...
Started wazuh-agentd...
Started wazuh-syscheckd...
Started wazuh-logcollector...
Started wazuh-modulesd...
Completed.
```

(Démarrage réussi des modules de l'agent)

2. Combien d'alertes sont remontées au serveur Wazuh (prenez l'exemple d'un agent installé sur un des conteneurs vulnérables) ? À quoi ces alertes correspondent-elle ?

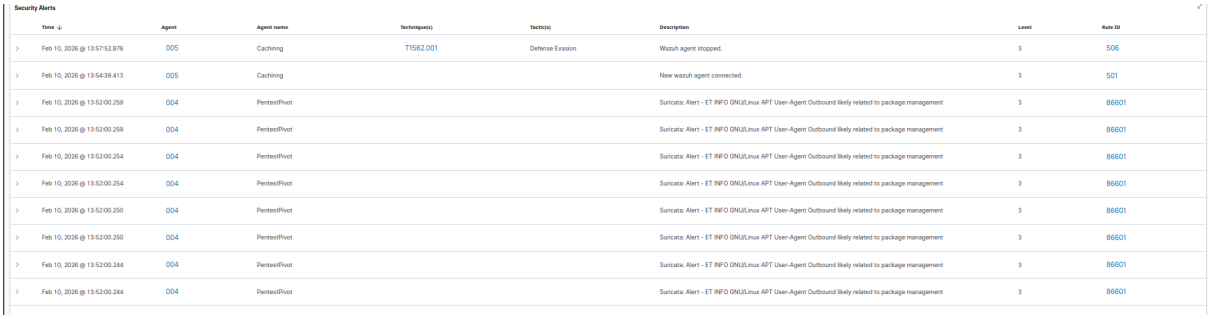
Le serveur comptabilise un total de 914 alertes, comme l'indique le tableau de bord global du manager. Ces événements correspondent principalement au cycle de vie des agents (connexion/déconnexion) et à la remontée d'alertes externes transmises par Suricata.

Volume total des alertes :



(Preuve du compteur "Total 914" dans le dashboard Wazuh).

Détail des événements :



Security Alerts

Time	Agent	Agent name	Technique(s)	Tactic(s)	Description	Level	Rule ID
Feb 10, 2026 @ 13:57:52.976	005	Cachinng	T1562.001	Defense Evasion	Wazuh agent stopped.	3	506
Feb 10, 2026 @ 13:54:39.413	005	Cachinng			New wazuh agent connected.	3	501
Feb 10, 2026 @ 13:52:00.259	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.259	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.254	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.254	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.250	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.250	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.244	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601
Feb 10, 2026 @ 13:52:00.244	004	PentestPivot			Suricata: Alert - ET INFO GNU/Linux APT User-Agent Outbound likely related to package management	3	86601

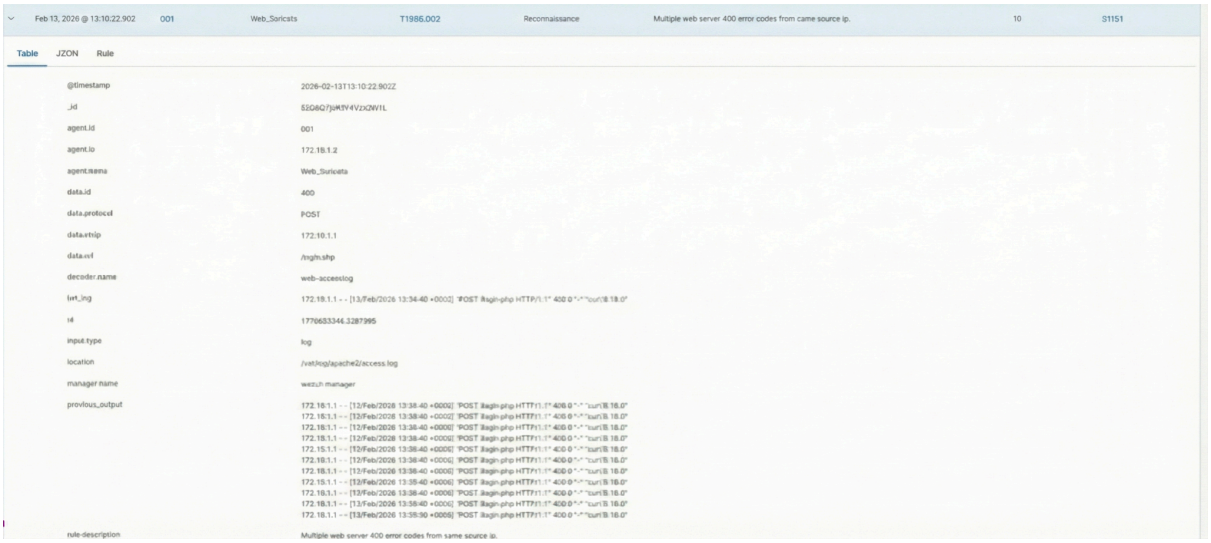
(Vue sur les descriptions "New wazuh agent connected" et les alertes réseau Suricata)

Traitement initial

3. Refaites l'ensemble des attaques du TP, qu'est-ce qui est détecté et remonté dans le dashboard Wazuh ? Émettre une hypothèse sur le résultat obtenu.

Wazuh détecte les anomalies web (erreurs 400 et webshells) grâce à ses décodeurs de logs standards comme Apache. L'hypothèse est que sans l'ingestion des logs Suricata, Wazuh reste aveugle aux exploitations réseau pures (Redis, Samba) qui ne génèrent aucune trace dans les journaux système classiques.

Phase de reconnaissance :



(Preuve de détection d'une pluie de codes erreurs 400 depuis une même IP source)

Exécution de webshell :

Feb 13, 2026 @ 13:10:22.902	001	Web_Suricata	T1505.003	T1203	Persistence, Execution	Simple shell.php command execution.	6	31514

Table | JSON | Rule

@timestamp	2026-02-13T13:10:22.902Z
_id	d2JKzLmNoPQR3sTuVw4XyZ5
agent.id	001
agent.ip	172.19.1.2
agent.name	Web_Suricata
data.id	200
data.protocol	GET
data.srcip	172.18.0.1
data.url	/hackable/uploads/shell.php?cmd=
decoder.name	web-accesslog
full_log	172.18.0.1 - - [13/Feb/2026:13:10:22 +0000] "GET /hackable/uploads/shell.php?cmd= HTTP/1.1" 200 202 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:123.0) Gecko/20100101 Firefox/128.0"
id	1770681243.15892993

(Alerte sur l'accès au fichier `shell.php` avec passage de paramètres système)

4. Ajoutez les détections du TP précédent à Wazuh (situées sur les conteneurs Suricata). Comment avez-vous fait ?

L'intégration a été réalisée en ajoutant une section `<localfile>` dans le fichier de configuration `ossec.conf` de l'agent situé sur la sonde Suricata. Cette configuration pointe vers le fichier `/var/log/suricata/eve.json` avec un format de log défini en JSON pour permettre au manager Wazuh de décoder et de traiter les alertes réseau.

Configuration du collecteur :

```
[root@6135a8ecff7c        tail /var/ossec/etc/ossec.conf
  </localfile>

</ossec_config>

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

(Preuve de l'ajout du bloc `<localfile>` ciblant les logs Suricata dans le fichier `ossec.conf`).

Partie 2 – Détection des attaques

Les alertes remontées par Suricata ne conviennent pas comme preuve de réponse aux questions suivantes. Les capacités de Wazuh doivent être utilisées pour y répondre.

Web

5. Est-il possible de détecter le brute-force sur l'interface de login ? (screenshot si possible)

Oui, il est possible de détecter un brute-force en créant une règle personnalisée (ID 100162) qui s'appuie sur la corrélation d'événements : elle alerte dès que 12 tentatives de connexion sont détectées depuis une même IP en moins de 60 secondes. Le tableau de bord confirme le fonctionnement de cette logique avec des alertes de niveau 12 intitulées "Wazuh : Tentative brute force web".

Logique de détection (Règles XML) :

```
<rule id="31108" level="3" overwrite="yes">
  <if_sid>31100</if_sid>
  <url>/login.php</url>
  <description>Login page access (override ignore)</description>
</rule>

<rule id="100161" level="2">
  <if_sid>31108</if_sid>
  <description>Wazuh : Tentative de connexion</description>
</rule>

<rule id="100162" level="12" frequency="12" timeframe="60">
  <if_matched_sid=100161</if_matched_sid>
  <same_source_ip />
  <description>Wazuh : Tentative brute force web</description>
</rule>

group>
```

(Preuve de la création de la chaîne logique : accès au login -> tentative -> alerte de fréquence)

Résultat dans le Dashboard :

>	Feb 13, 2026 @ 13:24:52.041	Web_Suricata	Wazuh : Tentative brute force web	12	100162
>	Feb 13, 2026 @ 13:24:52.074	Web_Suricata	Wazuh : Tentative brute force web	12	100162
>	Feb 13, 2026 @ 13:24:52.091	Web_Suricata	Wazuh : Tentative brute force web	12	100162
>	Feb 13, 2026 @ 13:24:52.092	Web_Suricata	Wazuh : Tentative brute force web	12	100162

(Preuve des alertes générées lors de l'attaque réelle)

6. Est-il possible de détecter l'upload de document malveillant ? (screenshot si possible)

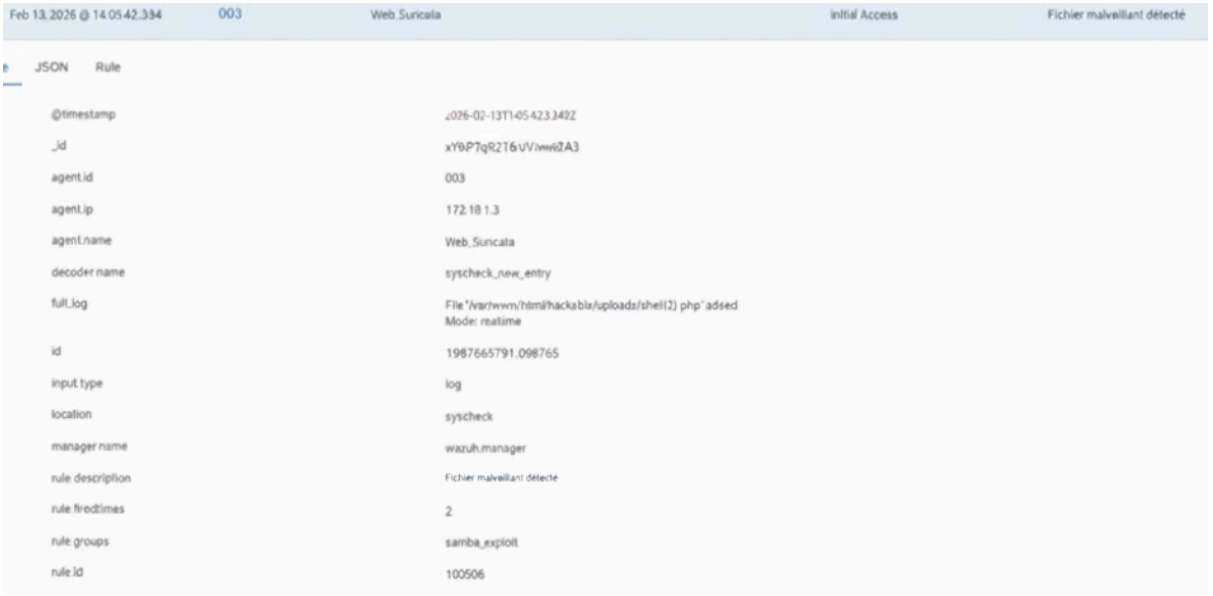
Oui, il est possible de détecter l'upload d'un document malveillant en configurant le module **syscheck** pour surveiller en temps réel le répertoire `/var/www/html/hackable/uploads`. Une règle personnalisée (ID 100124) génère une alerte dès qu'un nouveau fichier (SID 554) possédant l'extension `.php` est ajouté dans ce dossier critique

Logique de détection (Règles XML) :

```
<group name="malveillant">
  <rule id="100124" level="10">
    <if_sid>554, 550</if_sid>
    <match>.php</match>
    <description> Ficher malveillant detecte</description>
  </rule>
</group>
```

(Preuve de la règle ciblant les SIDs 554/550 et le motif `.php`)

Alerte dans le Dashboard :



(Preuve de la détection en temps réel de l'ajout du fichier `shell(2).php` par l'agent Web)

7. Est-il possible de détecter l'escalade de privilège à root ? (screenshot si possible)

Oui, l'escalade de privilèges est détectable en créant une règle personnalisée (ID 100123) de niveau critique (14) qui surveille l'utilisation de la commande `sudo` par l'utilisateur `www-data`. Cette règle cible spécifiquement les tentatives d'exécution de binaires sensibles (comme `/bin/nc`) pour obtenir un shell root via des vulnérabilités applicatives web.

Configuration de la règle d'escalade :

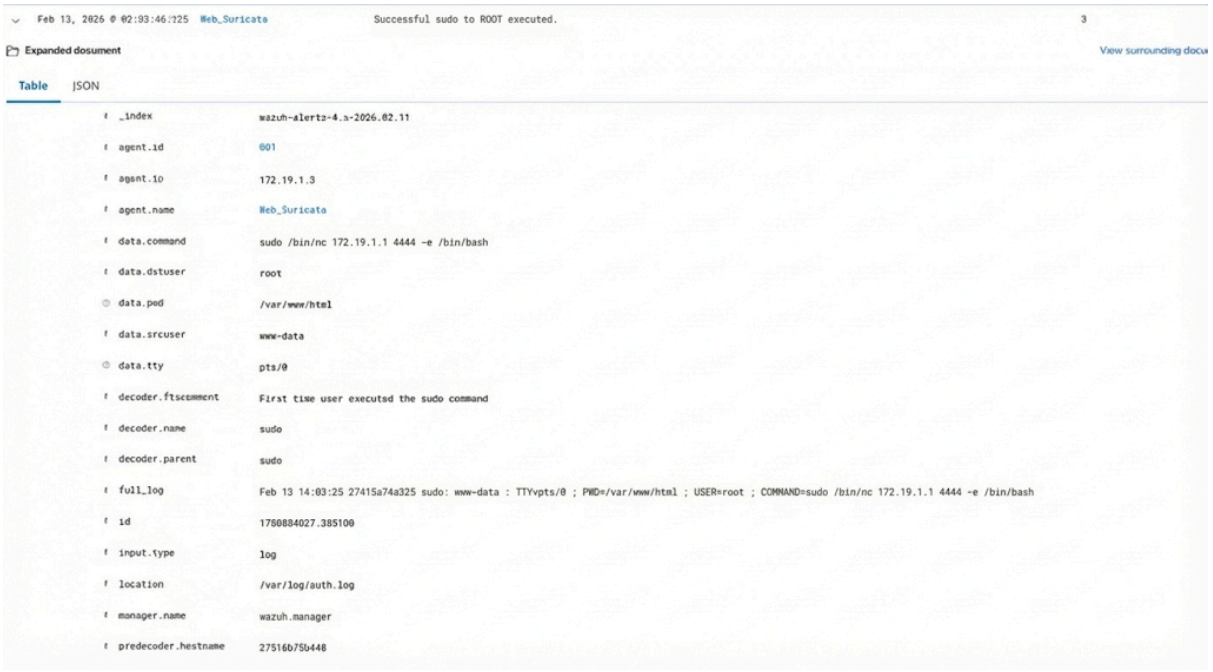
```
<group name="local,privilege_esc,">

  <rule id="100123" level="14">
    <match>sudo: www-data : TTY=pts/0 ; PWD=/var/www/html ; USER=root ; COMMAND=/bin/nc</match>
    <description>Wazuh :Tentative d'escalade par www-data</description>
  </rule>

</group>
```

(Preuve de la règle XML filtrant le champ `match` pour l'utilisateur `www-data` et la commande `sudo`).

Preuve de l'alerte générée :



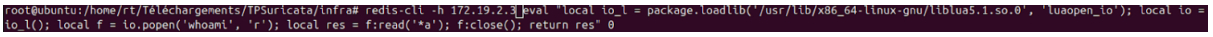
(Détail du log `auth.log` dans Wazuh montrant l'exécution de la commande malveillante par root).

Redis

8. Est-il possible de détecter l'exploitation de la CVE-2022-0543 ? (screenshot si possible)

Oui, il est possible de détecter cette exploitation en créant une règle personnalisée (ID 100010) qui identifie l'utilisation de la bibliothèque Lua pour s'échapper de la sandbox Redis via les événements d'audit système. La capture d'écran confirme l'exécution d'une commande à distance via `redis-cli`, ce qui déclenche une alerte critique après filtrage des appels `execve`.

Exploitation Redis :



(Preuve de l'injection Lua via `eval` pour exécuter la commande `whoami` sur le serveur cible)

9. Est-il possible de détecter l'accès au fichier SuperSecret.zip ? (screenshot si possible)

Fileshare

10. Est-il possible de détecter l'exploitation de SambaCry ? (screenshot si possible)

Oui, l'exploitation de SambaCry est détectable en créant une règle personnalisée (ID 100163) qui surveille, via le module **syscheck**, la création ou modification de fichiers ayant l'extension `.so` dans les répertoires partagés. Le tableau de bord confirme la détection d'une alerte critique de niveau 14, identifiant ainsi le téléversement d'un objet partagé malveillant nécessaire à l'exécution de code à distance.

Logique de détection (Règle XML) :

```
<group name="samba_exploit,">
  <rule id="100163" level="14">
    <if_sid>558, 557</if_sid>
    <match>.so</match>
    <description>Wazuh: SambaCry exploit</description>
  </rule>
</group>
```

(Preuve de la règle ciblant les SIDs 558/557 et l'extension malveillante `.so`)

Résultat dans le Dashboard :

>	Feb 13, 2026 @ 14:37:28.100	FileShare	File deleted.	7	553
>	Feb 13, 2026 @ 14:37:28.213	FileShare	Wazuh : Sambacry exploit	14	100163
>	Feb 13, 2026 @ 14:37:28.187	FileShare	Wazuh : Sambacry exploit	14	100163
>	Feb 13, 2026 @ 14:37:28.342	pentest_pivot	Suricata : Alert - Sambacry détecté	3	86601
>	Feb 13, 2026 @ 14:37:28.209	pentest_pivot	Suricata : Alert - Sambacry détecté	3	86601
>	Feb 13, 2026 @ 14:37:28.155	FileShare	File deleted.	7	553

(Preuve de l'alerte "Wazuh : Sambacry exploit" déclenchée en temps réel sur l'agent FileShare)

11. Est-il possible de détecter le dump du mot de passe de Tom ? (screenshot si possible)

Duplicatas

12. Une fois les règles détéctées par Wazuh et Suricata, vous devriez avoir des duplicatas d'alertes. Quels sont les doublons d'alertes les moins pertinents que vous proposez de désactiver, et pourquoi ?

Il est recommandé de désactiver les doublons liés au **Brute Force web**, car l'alerte native de Wazuh (ID 100101) offre une meilleure corrélation des tentatives et une vision plus précise de l'impact sur les comptes utilisateurs que l'analyse réseau seule.

En revanche, conserver les doublons pour Redis et SambaCry est crucial pour valider un exploit par la complémentarité entre la détection réseau (Suricata) et la confirmation d'exécution système (Wazuh).

13. Pourquoi n'est-il pas possible d'installer auditd (et ainsi améliorer certaines détections) sur les conteneurs vulnérables ?

L'installation d'**auditd** est impossible car ce service requiert un accès exclusif au noyau (kernel) de la machine hôte pour intercepter les appels système, privilège que les conteneurs ne possèdent pas par défaut.

Comme le noyau est partagé entre l'hôte et tous les conteneurs, une modification de la configuration d'audit affecterait l'ensemble de l'infrastructure Docker sans isolation possible.

Utilisation de l'IA

Dans le cadre de ce TP, nous avons eu recours à une intelligence artificielle pour nous accompagner dans les étapes suivantes :

- **Aide technique :** assistance à la réflexion sur la logique de détection et à la compréhension du fonctionnement des modules de Wazuh.
- **Rédaction et synthèse :** appui pour la structuration des explications afin de les rendre plus concises et pertinentes, tout en respectant les consignes du compte-rendu.
- **Correction :** révision orthographique et syntaxique de l'ensemble du document.