RÉPUBLIQUE DU BÉNIN
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ D'ABOMEY-CALAVI

INSTITUT DE FORMATION ET DE RECHERCHE EN INFORMATIQUE

BP 526 Cotonou Tel : +229 21 14 19 88
http://www.ifri-uac.net Courriel : contact@ifri.uac.bj

MÉMOIRE

pour l'obtention du

Diplôme de Licence en Informatique

Option : Sécurité Informatique

Présenté par :
Mafouz Kolawolé Aladé **LASSISSI**

Mise en place d'un système sécurisé utilisant l'IDS Suricata et la plateforme Wazuh pour la gestion des intrusions : cas de FUN HIGH TECH

Sous la supervision :
Mme Honorine **ADJIBI DEDJINOU**
Consultante en Audit et Gestion des Systèmes d'Information

Membres du jury :

Mr Sanda TIDJANI	Docteur	IFRI-UAC	Président
Mme Miranda GNONLONFOUN	Ingénieure	IFRI-UAC	Examinateur
Mme Honorine ADJIBI DEDJINOU	Ingénieure	IFRI-UAC	Rapporteur

Année Académique : **2025-2026**

Sommaire

Dédicace ii

Remerciements iii

Résumé iv

Abstract v

Liste des figures vi

Liste des tableaux viii

Liste des acronymes ix

Introduction 1

1 Revue littéraire 3

2 Méthodologie utilisée 11

3 Solution déployée 16

Conclusion 33

Bibliographie 34

Webographie 35

Table des matières 36

Dédicace

Je dédie ce modeste travail :
À mes parents
À mes frères,
À mes amis,
Mes guides au cours du stage,
Pour vos sacrifices et votre soutien tout au long de mon cursus.

Remerciements

Au terme de ce projet, nous aimerions attribuer nos sincères remerciements à Dieu le tout puissant pour nous avoir prêté vie, motivation et assistance dans la réalisation de ce travail. Nous exprimons notre vive gratitude aux personnes physiques et institutions qui ont contribué à rendre meilleur notre travail, notamment :

- au Professeur Eugène C. EZIN, Directeur de l'Institut de Formation et de Recherche en Informatique (IFRI) pour sa disponibilité et ses nombreux conseils ;
- à Monsieur Gaston EDAH, Directeur Adjoint pour ses nombreux conseils et sa présence durant notre formation ;
- à Madame Honorine ADJIBI DEDJINOU, maître mémoire qui nous a supervisés tout au long de notre recherche ;
- à Monsieur Charles GAVOEDO, Directeur Général de FUN HIGH TECH, pour nous avoir accordé un cadre adéquat pour le stage de fin de formation ;
- à Monsieur HESSOU Gael, Directeur Administratif de FUN HIGH TECH ;
- à tout le corps professoral de l'Institut de Formation et de Recherche en Informatique (IFRI), pour nous avoir transmis toutes les connaissances possibles que nous devrions avoir ;
- Pour finir, nous remercions les membres du jury qui ont accepté d'évaluer notre projet.

Nous leur présentons toute notre gratitude et nos profonds respects.

Résumé

Les menaces informatiques augmentent au fur et à mesure que l'humanité œuvre pour les progrès technologiques. C'est ainsi que les intrusions qui peuvent s'effectuer de plusieurs manières, ont vu le jour. Pour y remédier, nous disposons aujourd'hui de quelques technologies dont les systemes de surveillance et d'alerte de sécurite et les IDS, qui peuvent s'avérer très efficaces. SURICATA est un IDS qui repère les anomalies sur un réseau . Wazuh quant à elle, est une plateforme de collecte, d'analyse et de visualisation de données de journaux (logs) en temps réel. L'intégration de ces deux technologies nous permettra de mettre en place un système de sécurité autonome contre les intrusions et autres activités suspectes. Nous avons donc installé et configuré Wazuh et SURICATA dans un réseau virtuel et nous avons effectué des tests qui attestent de l'efficacité du système constitué par ces deux technologies.

Mots clés : IDS, SURICATA, Wazuh, intrusions, activités suspectes

Abstract

Cyber threats are increasing as humanity strives for technological advancement. This is how intrusions, which can be carried out in several ways, came into being. To remedy this, we now have some technologies, including security monitoring and alert systems and IDS, which can be very effective. SURICATA is an IDS that identifies anomalies on a network or host machine. Wazuh is a platform for collecting, analyzing and visualizing log data in real time. The integration of these two technologies will allow us to implement an autonomous security system against intrusions and other suspicious activities. We therefore installed and configured Wazuh and SURICATA in a virtual network and we carried out tests which attest to the effectiveness of the system made up of these two technologies.

Key words: IDS, SURICATA, Wazuh, intrusions, suspicious activities

Liste des figures

2.1 Oracle VM VirtualBox . . . 12
2.2 Logo suricata.[4] . . . 12
2.3 Logo Wazuh.[5] . . . 12
2.4 Architecture wazuh.[6] . . . 13
2.5 Architecture du réseau.[7] . . . 13
2.6 Architecture après déploiement. [8] . . . 14

3.1 Les VM . . . 17
3.2 suricata 7.0.7 . . . 18
3.3 demarrage service suricata . . . 19
3.4 Installation serveur Wazuh et ses composants . . . 19
3.5 Installation Wazuh . . . 19
3.6 Démarrage de Wazuh-manager . . . 20
3.7 Démarrage de Wazuh-indexer . . . 20
3.8 Démarrage de Wazuh-dashboard . . . 20
3.9 Interface de connexion . . . 21
3.10 installation et configuration de l'agent wazuh . . . 21
3.11 Démarrage de l'agent Wazuh . . . 22
3.12 installation de l'agent wazuh . . . 22
3.13 Démarrage de l'agent wazuh . . . 22
3.14 image montrant les agents wazuh deployés . . . 23
3.15 configuration de l'agent Wazuh . . . 23
3.16 Scan de port vers la machine cible . . . 24
3.17 Alertes générées pour le scan de port . . . 24
3.18 image montrant les logs générées par suricata . . . 25
3.19 informations sur l'alerte généré par suricata . . . 25
3.20 image montrant le serveur Apache . . . 26
3.21 Démarrage du service web apache . . . 26
3.22 configuration de l'agent Wazuh . . . 27
3.23 requête simulant une attaque par injection SQL . . . 27
3.24 Alertes générées pour l'attaque par injection SQL . . . 27
3.25 informations sur l'alerte généré par l'injection SQL . . . 28
3.26 image montrant l'installation de notre serveur ssh . . . 29
3.27 Démarrage du service ssh . . . 29
3.28 Connexion ssh par force brute . . . 29
3.29 Alertes générées lors de l'inondation du trafic sshd de la cible . . . 30
3.30 Image montrant la configuration du bloquage . . . 31

3.31 Alertes générées pour le bloquage 31
3.32 Test de connectivité vers la machine cible 32

Liste des tableaux

1.1 Grille comparative de quelques systèmes de détection d'intrusions 7
1.2 Grille comparative de quelques solutions SIEM 9

Liste des acronymes

DMZ :

Demilitarized Zone ix,

FIM :

File Integrity Monitoring ix,

HIDS :

Host Intrusion Detection System ix,

IDS :

Intrusion Detection System ix,

IP :

Internet Protocol ix,

NIDS :

Network Intrusion Detection System ix,

SIEM :

Security Information and Event Management ix,

Introduction Générale

La sécurité informatique est devenue un enjeu majeur pour les organisations face à la multiplication des cybermenaces. Les systèmes d'information sont constamment exposés à des attaques sophistiquées, rendant essentielle la mise en place de mécanismes de détection et de prévention efficaces. Dans ce contexte, les solutions comme les systèmes de détection d'intrusions (IDS) et les plateformes de gestion de la sécurité jouent un rôle crucial. Ce mémoire explore l'intégration de deux outils open source puissants, Suricata (un IDS performant) et Wazuh (une plateforme de sécurité unifiée), pour créer un système sécurisé capable de détecter et de gérer les intrusions de manière proactive.

1- Problématique

Face à l'évolution des cybermenaces, les organisations doivent s'assurer que leurs systèmes d'information sont protégés contre les intrusions malveillantes. Cependant, la complexité des attaques modernes rend souvent les solutions traditionnelles insuffisantes. Comment mettre en place un système sécurisé, capable de détecter, analyser et répondre efficacement aux intrusions tout en restant accessible et évolutif ? Ce mémoire propose d'étudier l'utilisation combinée de Suricata et Wazuh pour répondre à cette problématique.

2- Contexte et justification

FUN HIGH TECH, notre lieu de stage, est une société spécialisée dans le développement de solutions informatiques et domotiques. Elle manipule des données sensibles et conçoit des systèmes intelligents pour ses clients. Ces activités impliquent une forte dépendance à une infrastructure informatique fiable et sécurisée. Notre analyse de l'existant révèle que l'entreprise utilise des outils de supervision classiques pour sécuriser son système informatique. Cependant, ces solutions peuvent s'avérer insuffisantes face à l'évolution croissante des cybermenaces, qui pourraient exposer les données des clients et nuire à la réputation de l'entreprise. Face à ce constat, nous proposons une approche intégrée combinant Suricata (pour la détection des intrusions réseau) et Wazuh (pour l'analyse des logs des systèmes et la réponse aux incidents). Cette solution permettra d'améliorer la sécurité du réseau de FUN HIGH TECH en passant d'une supervision passive à une surveillance proactive, garantissant ainsi une détection en temps réel des menaces et une réaction adaptée aux incidents de sécurité.

3- Objectifs

L'objectif global de ce projet est de mettre en place un système de détection et de réponse aux intrusions en intégrant Suricata et Wazuh pour surveiller, analyser et réagir éfficacement aux menaces sur le réseau .De manière spécifique,nous aurons à :

- Déployer et configurer Suricata et Wazuh ;
- Centraliser et analyser les alertes de sécurité ;
- Mettre en œuvre un mécanisme de réponse aux intrusions ;

4- Organisation du mémoire

Ce document est réparti en trois chapitres. Le premier chapitre est relatif à la revue de littérature. Le second chapitre présente le système constitué de Wazuh et Suricata et expose les résultats espérés. Pour finir, le troisième chapitre présente le déploiement de la solution de sécurité et les tests de fonctionnalités de ce système.

Chapitre 1

Revue littéraire

Introduction

Dans un environnement numérique de plus en plus exposé aux cybermenaces, la détection et la gestion des intrusions sont devenues essentielles pour assurer la sécurité des systèmes d'information. Les outils comme Suricata, un système de détection d'intrusions (IDS) performant, et Wazuh, une plateforme de sécurité unifiée, offrent des solutions robustes pour surveiller et contrer les attaques.

1.1 La sécurité informatique [1]

1.1.1 Définition de la sécurité informatique

La *Sécurité Informatique* consiste à protéger des systèmes, des réseaux et des données contre les menaces, les attaques et les vulnérabilités. Elle implique la mise en place de mesures de sécurité techniques, administratives et physiques pour garantir la confidentialité, l'intégrité et la disponibilité des données et des systèmes.

1.1.2 Objectifs de la sécurité informatique

La sécurité informatique vise généralement cinq principaux objectifs :

- **la confidentialité** : consistant à assurer que seules les personnes autorisées aient accès aux ressources échangées ;
- **l'authentification** : consistant à assurer que seules les personnes autorisées aient accès aux ressources ;
- **l'intégrité** : c'est-à-dire les données ne doivent pas être modifiables de façon accidentelle ou par malveillance ;
- **la non-répudiation** : permettant de garantir qu'aucune action ne peut être niée par l'auteur ;
- **la disponibilité** : permettant de maintenir le bon fonctionnement du système d'information.

1.1.3 Outils nécessaires pour sécuriser un réseau informatique

La seule façon de garantir la sécurité d'un réseau est d'utiliser les derniers logiciels de protection informatique et de les mettre à jour constamment. Parmi les meilleurs outils de protection de données pour le réseau informatique d'une entreprise, on a : les pare-feu de nouvelle génération, les procurations serveurs et les antivirus.

- **Les systèmes de détection d'intrusion (IDS) :** Ce sont des outils de sécurité réseau conçus pour détecter les activités suspectes et les violations de politique de sécurité sur un réseau informatique. Les IDS fonctionnent en surveillant le trafic réseau et en analysant les données pour détecter les signes d'une activité malveillante.
- **Un SIEM (Security Information and Event Management) :** C'est une solution de gestion de la sécurité qui permet de collecter, analyser et corréler les données de sécurité en temps réel pour identifier et répondre aux menaces potentielles.C'est aussi un outil essentiel pour les équipes de sécurité qui cherchent à renforcer leur posture de sécurité, à améliorer leur capacité de détection des menaces et à assurer la conformité réglementaire.
- **Les VPN (Virtual Private Network) :** Ce sont des technologies de réseau qui permettent de créer des connexions sécurisées et privées sur des réseaux publics tels qu'Internet. Les VPN sont utilisés pour protéger la confidentialité et l'intégrité des données transmises sur le réseau en chiffrant le trafic réseau et en masquant l'adresse IP de l'utilisateur.

1.2 Notion sur les Systèmes de Détection d'Intrusion [2]

1.2.1 Définition des systèmes de détection d'intrusion

Un IDS est un ensemble de composants logiciels et/ou matériels dont la fonction principale est de détecter et analyser des activités anormales ou suspectes sur la cible qu'il est sensé surveiller. La cible peut être représentée par un réseau ou un hôte (un ordinateur simple). Il permet donc aux responsables de la sécurité du système d'information d'avoir connaissance des intrusions et autres attaques effectuées (réussies ou échouées). Voici quelques termes que nous retrouvons souvent quand on parle de détection d'intrusion :

- **IPS** : un outil similaire aux IDS mais permettant de prendre des mesures afin de diminuer les impacts d'une attaque.
- **Vrai Positif** : le test a correctement identifié un cas positif.
- **Vrai Négatif** : le test a correctement identifié un cas négatif.
- **Faux Positif** : le test a identifié un cas positif, mais le cas est en réalité négatif.
- **Faux Négatif** : le test a identifié un cas négatif, mais le cas est en réalité positif.

1.2.2 Caractéristiques d'un système de détection d'intrusion

Un IDS peut être basé sur un réseau ou sur un hôte. Il peut utiliser des signatures d'attaques connues ou des techniques d'apprentissage automatique pour détecter les attaques. Il peut générer des alertes

à la détection d'activités suspectes et peut être configuré pour différents niveaux de sensibilité. Lorsqu'il est attentivement surveillé, cela peut permettre de réduire le nombre de faux positifs. Voici une liste non-exhaustive des principales caractéristiques d'un IDS :

- **Surveillance Continue** : Un IDS surveille en permanence le trafic réseau, les systèmes et les applications à la recherche d'activités anormales ou d'intrusions potentielles.
- **Analyse des Comportements** : L'IDS analyse les modèles de comportement normaux et peut détecter les déviations significatives qui pourraient indiquer une activité malveillante.
- **Détection d'Anomalies** : L'IDS identifie les anomalies par rapport aux modèles de trafic ou de comportements habituels, ce qui permet de repérer les activités anormales ou inattendues.
- **Analyse des protocoles** : Il peut analyser les en-têtes de paquets et les protocoles réseau pour repérer des incohérences ou des manipulations suspectes.
- **Alertes et Notifications** : Lorsqu'une activité suspecte est détectée, l'IDS a généré des alertes et des notifications pour informer les administrateurs ou les équipes de sécurité.
- **Collecte des Données** : Les IDS collectent des données sur les activités réseau, les connexions, les tentatives d'accès et d'autres informations pour analyses

Les règles et les paramètres de l'IDS peuvent être personnalisés en fonction des besoins spécifiques de l'entreprise. Pour finir, et ceci représente la caractéristique que nous exploiterons le plus dans notre production, les IDS peuvent être intégrés avec d'autres systèmes de sécurité, tels que les pare-feu et les Systèmes de Prévention d'Intrusion (IPS).

1.2.3 Fonctionnement d'un système de détection d'intrusion

Il faut distinguer deux aspects dans le fonctionnement d'un IDS : le mode de détection utilisé et la réponse apportée par l'IDS lors de la détection d'une intrusion. Il existe deux modes de détection, la détection d'anomalies et la reconnaissance de signatures. De même, deux types de réponses existent, la réponse passive et la réponse active.

- **Les méthodes de détection**
  La détection d'anomalies : La détection d'anomalie consiste à détecter des anomalies par rapport à un trafic habituel. La mise en place comprend toujours une phase d'apprentissage au cours de laquelle les systèmes de détection d'intrusion vont découvrir le fonctionnement normal des éléments surveillés. Ils sont en mesure de signaler les divergences par rapport au fonctionnement normal. De plus, des ajustements sont nécessaires pour faire évoluer le modèle de référence de sorte qu'il reflète l'activité normale des utilisateurs et réduire le nombre de fausses alertes générées. Dans le cas d'un HIDS, ce type de détection peut être basé sur des informations telles que l'activité sur le disque, les horaires de connexion ou d'utilisation de certains fichiers.
  La reconnaissance de signature : Il faut noter que la reconnaissance de signature est le mode de fonctionnement le plus implémenté par les IDS du marché. Elle consiste à rechercher dans l'activité de l'élément surveillé les empreintes d'attaques connues. L'IDS par nature est réactif, il ne peut détecter que les attaques dont il possède la signature. Pour cela, il nécessite des mises à jour fréquentes. L'efficacité de ce système de détection réside dans la précision de sa

base de signature. C'est la raison pour laquelle ces systèmes sont contournés par les pirates. Il est possible d'établir des signatures génériques, qui permettent de détecter les variantes d'une attaque. Cela demande une bonne connaissance des attaques et du réseau, de façon à stopper les variantes d'une attaque, au niveau des paquets ou au niveau Protocol.

- **Les réponses "Active" et "Passive"**
  Il existe deux types de réponses, suivant les IDS utilisés. La réponse passive est disponible pour tous les IDS alors que la réponse active est plus ou moins implémentée.
  Réponse active : La réponse active a pour but de stopper une attaque au moment de sa détection. Elle implique des actions automatisées prises par un IDS qui permet de couper rapidement une connexion suspecte quand le système détecte une intrusion. Pour cela, on dispose de deux techniques : la reconfiguration du pare-feu et l'interruption de la connexion TCP.
  Réponse passive : La réponse passive d'un IDS consiste à enregistrer les intrusions détectées dans un fichier de log qui sera analysé par le responsable de sécurité. Certains IDS permettent de logger l'ensemble d'une connexion identifiée comme malveillante. Ceci permet de remédier aux failles de sécurité pour empêcher les attaques enregistrées de se reproduire, mais elle n'empêche pas directement une attaque de se produire.

1.2.4 Comparaison entre les différents types de Systèmes de Détection d'Intrusion

1.2.4.1 Les différents types de système de détection d'intrusion

- **Les Systèmes de Détection d'Intrusions Réseau (NIDS)**
  Les IDS réseaux analysent en temps réel le trafic qu'ils aspirent à l'aide d'une sonde carte réseau en mode promiscuous1. Ensuite, les paquets sont décortiqués puis analysés. Il est fréquent de trouver plusieurs IDS sur les différentes parties du réseau. On trouve souvent une architecture composée d'une sonde placée à l'extérieur du réseau afin d'étudier les tentatives d'attaques et d'une sonde en interne pour analyser les requêtes ayant traversé le pare-feu.

- **Les Systèmes de Détection d'Intrusions Hôte (HIDS)**
  Les IDS hôte permettent d'analyser, non seulement, le trafic réseau, mais aussi l'activité se passant sur la machine. Le but de ce type d'IDS est d'assurer l'intégrité des données d'un système et d'analyser le flux relatif à une machine ainsi que ces journaux. Il existe plusieurs solutions qui proposent cette fonctionnalité, par exemple le HIDS Tripwire.

- **Les Systèmes de Détection d'Intrusions Hybrides**
  Les IDS hybrides rassemblent les caractéristiques des NIDS et HIDS. Ils permettent, en un seul outil, de surveiller le réseau et les terminaux. Les sondes sont placées en des points stratégiques, et agissent comme NIDS et/ou HIDS suivant leurs emplacements. Toutes ces sondes remontent alors les alertes à une machine qui va centraliser le tout, et lier les informations d'origines multiples. Ainsi, nous comprenons que les IDS hybrides sont basés sur une architecture distribuée, où chaque composant unifie son format d'envoi. Cela permet de communiquer et d'extraire des alertes plus pertinentes.

TABLE 1.1 : Grille comparative de quelques systèmes de détection d'intrusions

Critères	Suricata	Snort	OSSEC	Zeek	Security Onion
Détection d'attaque réseau	Oui	Oui	Non	Oui mais pas en mode signature	Oui(via suricata et zeek)
Détection comportementale avancée	Oui (mode IDS et IPS)	Non	Oui mais limité	Oui (analyse de flux)	Oui (via zeek)
Débit et performance	Multi-threadé	Mono-threadé (moins performant)	NIDS non supporté	Très performant en analyse	Performant mais lourd
Facilité d'intégration	Oui	Possible mais nécessite adaptation	Oui	Possible mais pas direct	Oui

1.3 Notion de gestion des évènements et informations de sécurité (SIEM)[3]

1.3.1 Définition de SIEM

La *SIEM* est une solution de sécurité qui combine deux fonctions principales :

- **SIM (Security Information Management)** : collecte, analyse et génère des rapports sur les données de sécurité.
- **SEM (Security Event Management)** : Surveille et analyse les événements de sécurité en temps réel.

1.3.2 Objectifs de SIEM

- **Centralisation des données** : Agréger les logs et événements de sécurité provenant de diverses sources (pare-feu, IDS/IPS, serveurs, etc.) pour une vue unifiée ;
- **Détection des menaces** : Identifier les activités suspectes ou malveillantes en temps réel grâce à des règles et analyses comportementales ;
- **Corrélation des événements** : Relier les événements de sécurité pour détecter des schémas d'attaque complexes ;
- **Conformité réglementaire** : Aider à respecter les normes de sécurité (RGPD, PCI DSS, etc.) en générant des rapports d'audit ;
- **Réponse aux incidents** : Fournir des alertes précoces et des informations pour une réponse rapide aux incidents de sécurité.

1.3.3 Fonctionnement de SIEM

Le SIEM (Security Information and Event Management) fonctionne en suivant un processus structuré pour collecter, analyser et répondre aux événements de sécurité. Voici les étapes clés de son fonctionnement :

- **Collecte des données (Data Collection)** : Le SIEM collecte des logs et des événements de sécurité à partir de multiples sources, telles que :Pare feu,Systèmes de détection et prévention d'intrusions (IDS/IPS),Serveurs, routeurs et commutateurs,Applications et bases de données,Endpoints (postes de travail, appareils mobiles).Ces données sont centralisées dans une plateforme unique pour une analyse approfondie.
- **Normalisation des données (Data Normalization)** : Les logs provenant de différentes sources ont des formats variés. Le SIEM normalise ces données pour les rendre compatibles et faciliter leur analyse.
- **Corrélation des événements (Event Correlation)** : Le SIEM utilise des règles prédéfinies et des algorithmes pour relier les événements entre eux.Cela permet de détecter des schémas d'attaque complexes qui ne seraient pas visibles en examinant les événements individuellement.
- **Réponse aux incidents (Incident Response)** : Le SIEM fournit des informations détaillées pour aider les équipes à investiguer et à répondre aux incidents. Certains SIEM modernes intègrent des fonctionnalités de SOAR (Security Orchestration, Automation, and Response) pour automatiser les réponses.

1.3.4 Quelques catégories de SIEM

Les solutions SIEM (Security Information and Event Management) peuvent être classées en différentes catégories en fonction de leur architecture, de leur mode de déploiement, de leurs fonctionnalités et de leur public cible. Voici les principales catégories de SIEM :

- **SIEM traditionnels (On-Premise)** : Ces SIEM sont déployés localement, dans l'infrastructure de l'organisation. Ils Nécessitent des ressources matérielles et logicielles dédiées,Offrent un contrôle total sur les données et la configuration et conviennent aux organisations ayant des exigences strictes en matière de confidentialité des données. Comme exemple, nous avons IBM QRadar, Splunk Enterprise, McAfee ESM.
- **SIEM cloud (Cloud-Native ou SaaS)** : Ces SIEM sont hébergés dans le cloud et accessibles via un modèle SaaS (Software as a Service). Ils possèdent des caractéristiques (Aucune infrastructure locale requise, Mises à jour automatiques et gestion simplifiée). Comme exemple, nous avons Microsoft Sentinel (anciennement Azure Sentinel), Splunk Cloud, Sumo Logic.
- **SIEM open source** : Ces SIEM sont basés sur des logiciels open source, offrant une alternative économique aux solutions propriétaires. Ils nécessitent une expertise technique pour la configuration et la maintenance et une personnalisation poussée grâce à l'accès au code source. Comme exemple, nous avons OSSIM (Open Source SIEM), ELK Stack (Elasticsearch, Logstash, Kibana), Wazuh.
- **SIEM orientés conformité** : Ces SIEM sont conçus pour aider les organisations à se conformer aux réglementations et normes de sécurité, incluent des modèles de rapports prédéfinis pour des réglementations spécifiques (RGPD, PCI DSS, HIPAA, etc.) et simplifient les audits et les contrôles de conformité. Comme exemple, nous avons LogRhythm, AlienVault , ManageEngine EventLog Analyzer.

TABLE 1.2 : Grille comparative de quelques solutions SIEM

Critères	Wazuh	Splunk	ELK	Graylog	OSSIM
Détection d'intrusions sur les hôtes (HIDS)	Oui (FIM, détection comportementale)	Non	Non	Non	Oui (OSSEC, mais limité)
Réponse aux incidents	Oui (XDR, blocage d'IP)	Possible mais via un add-on	Possible mais complexe	Possible avec pipelines	Oui (via OSSEC)
Coût	Gratuit et open-source	Payant (cher en production)	Gratuit mais demande des ressources	Gratuit	Gratuit
Facilité d'intégration	Facile	Via add-on	Nécessite Logstash	Possible mais limité	Complexe

1.4 Justification des choix techniques

Afin de mettre en place une solution de sécurité efficace et adaptée aux besoins de l'entreprise **FUN HIGH TECH**, il a d'abord été essentiel de bien distinguer les deux couches principales à protéger dans une infrastructure informatique : le réseau et le système.

- **Le réseau** correspond à l'ensemble des équipements (commutateurs, routeurs, pare-feux, interfaces, etc.) et des protocoles de communication entre les machines. C'est à ce niveau que transitent les flux de données, ce qui en fait une cible privilégiée pour les attaques telles que les scans de ports, les attaques par déni de service (DoS), le spoofing ou les tentatives d'intrusion via le trafic.
- **Le système** quant à lui, désigne les serveurs, postes de travail et dispositifs exécutant un système d'exploitation (Linux, Windows, etc.). Les menaces à ce niveau concernent les accès non autorisés, les modifications de fichiers critiques, les élévations de privilèges, ou encore les attaques internes et post-intrusion.

1.4.1 Suricata (IDS -Sécurité réseau)

FUN HIGH TECH est une entreprise connectée à Internet, exposée à des risques réseau tels que les scans de ports, attaques DoS, ou tentatives d'injection. Un IDS (Intrusion Detection System) est essentiel pour analyser le trafic et détecter ces attaques en temps réel.Suricata a été retenu pour sa capacité à analyser le trafic à haut débit grâce au support du multithreading, ce qui le rend plus performant que Snort.Il permet de surveiller efficacement le trafic réseau entre les serveurs internes de **FUN HIGH TECH** (web, SSH, domotique) et Internet. Il assure une détection réactive des menaces réseau.

1.4.2 Wazuh (HIDS + SIEM - Sécurité système)

Les attaques ne se limitent pas au réseau ; elles peuvent aussi viser les systèmes eux-mêmes. Un SIEM (Security Information and Event Management) permet de collecter, centraliser, analyser et corréler les logs système et applicatifs. Le HIDS (Host-based Intrusion Detection System) assure la surveillance

interne des systèmes.Wazuh combine les fonctionnalités de HIDS, de SIEM et de monitoring. Contrairement à OSSIM (plus complexe et lourd), Wazuh est léger, modulaire et facilement déployable. Il intègre des règles de détection avancées, une interface Kibana, et une compatibilité native avec Suricata. Elle permet de surveiller en temps réel les événements de sécurité sur les serveurs internes de **FUN HIGH TECH** et sur l'IDS. Les agents Wazuh installés sur chaque système remontent les logs vers le serveur central pour analyse et réaction immédiate.

1.4.3 Architecture distribuée (agents + serveur Wazuh)

L'installation d'agents sur chaque système surveillé (serveurs web, SSH, IDS) permet une collecte granulaire des événements. Le serveur central Wazuh traite, corrèle et analyse toutes ces données.Les agents offrent une visibilité plus fine (analyse de journaux, intégrité des fichiers, processus, authentifications...). Cette approche est plus pertinente pour une structure comme **FUN HIGH TECH** qui souhaite assurer la traçabilité des activités critiques. Cela permet une surveillance centralisée mais déployée de manière distribuée, assurant scalabilité, résilience et efficacité dans l'analyse et la réaction face aux incidents.

Conclusion

Dans ce chapitre, nous avons présenté plusieurs technologies qui seront utilisées ensemble pour l'atteinte de nos objectifs. Il nous a permis de cerner les concepts des réseaux informatiques ainsi que celui de la Sécurité Informatique et ses principaux objectifs afin de remédier aux menaces constantes que subit les systèmes informatiques . Nous avons évoqué quelques solutions de sécurité existantes et pour finir, nous avons évoqué les raisons du choix de notre solution.

Chapitre 2

Méthodologie utilisée

Introduction

Dans un contexte où les menaces cybernétiques évoluent constamment, les méthodes et outils pour les contrer doivent également s'adapter et se perfectionner. La gestion des intrusions, en particulier, nécessite une approche méthodologique rigoureuse pour garantir une protection efficace des systèmes et des données. Dans ce chapitre, nous allons présenter les outils qui vont nous permettre à mener à bien notre solution ainsi que le fonctionnement de notre système à mettre en place.

2.1 Présentation des outils utilisés

2.1.1 Oracle VM Virtualbox [9]

Oracle VM VirtualBox est un hyperviseur open-source qui permet la création d'une ou plusieurs machines virtuelles au sein d'un même système d'exploitation.Celles-ci peuvent être reliées au réseau local avec des adresses IP différentes, tout en étant sur la même machine physique. Il est possible de faire fonctionner plusieurs machines virtuelles en même temps selon la limite correspondant aux performances de l'ordinateur hôte.



FIGURE 2.1 : Oracle VM VirtualBox

2.1.2 Suricata [10]

Suricata est un système de détection d'intrusion de réseau open source (NIDS) créé par une organisation appelée "The Open Information Security Foundation" (OISF). Suricata est un sniffer de paquets qui surveille le trafic réseau en temps réel, scrutant chaque paquet minutieusement pour détecter une charge utile dangereuse ou des anomalies suspectes. Il est basé sur libpcap qui est très utilisé dans les sniffers et les analyseurs de trafic TCP / IP. Grâce à l'analyse du protocole et à la recherche de contenu, Suricata détecte les méthodes d'attaque, y compris le déni de service, le dépassement de tampon, les attaques CGI, les balayages de port furtif et les sondes SMB.



FIGURE 2.2 : Logo suricata.[4]

2.1.3 Wazuh [11]



FIGURE 2.3 : Logo Wazuh.[5]

Wazuh est une plateforme open source utilisée pour la prévention, la détection et la réponse aux menaces. Elle sécurise les environnements de travail sur site, virtualisés, conteneurisés et en cloud. Wazuh est largement utilisé par des milliers d'organisations à travers le monde, de la petite entreprise à la grande entreprise. Pour finir, Wazuh intègre diverses fonctionnalités de sécurité, notamment la

détection d'intrusion, la détection de vulnérabilité, la surveillance de l'intégrité des fichiers, l'analyse des journaux et la gestion de la conformité.

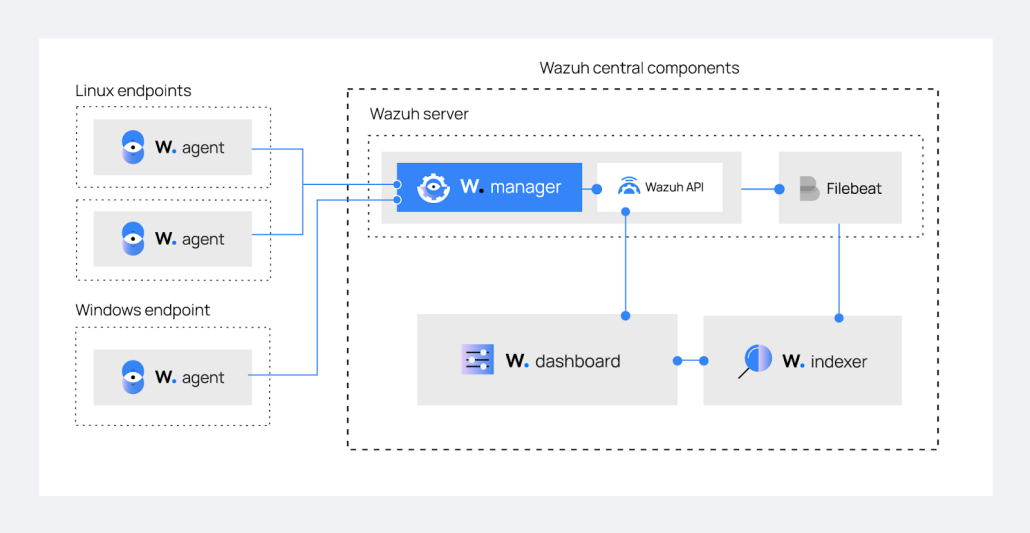


FIGURE 2.4 : Architecture wazuh.[6]

2.2 Architecture de la solution mis en place

2.2.1 Présentation de l'architecture existante

Avant toute modification de l'infrastructure réseau, il est essentiel de se remémorer sa forme actuelle, ce qui permettra de mieux évaluer les effets de la solution de surveillance après sa mise en oeuvre.

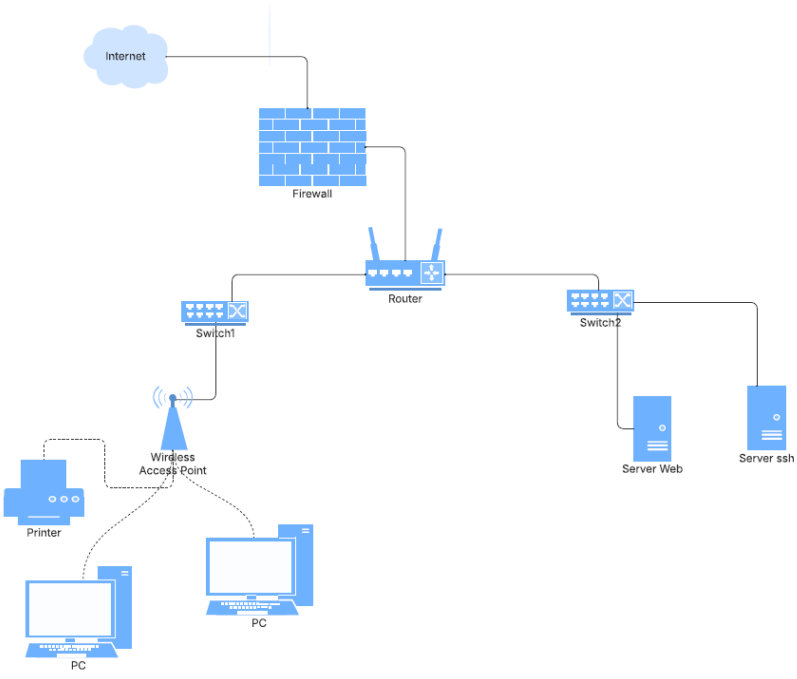


FIGURE 2.5 : Architecture du réseau.[7]

Le réseau interne de l'entreprise FUN HIGH TECH est constitué des équipements d'intercon-

nexion, de serveur, d'un ensemble de postes employés reliés par des câbles Ethernet ou via réseau WiFi. Elle est structurée pour assurer une connectivité sécurisée et contrôlée. L'accès à Internet est filtré par un pare-feu relié à un routeur central, qui distribue le trafic vers deux switchs. Le Switch 1 connecte les postes de travail de l'entreprise, une imprimante et un point d'accès Wi-Fi interne, tandis que le Switch 2 est dédié aux serveurs, incluant un serveur web et un serveur SSH. Seuls les postes de l'entreprise peuvent accéder au réseau interne et aux services, garantissant ainsi un environnement sécurisé et maîtrisé.

2.2.2 Présentation de l'architecture implémentée

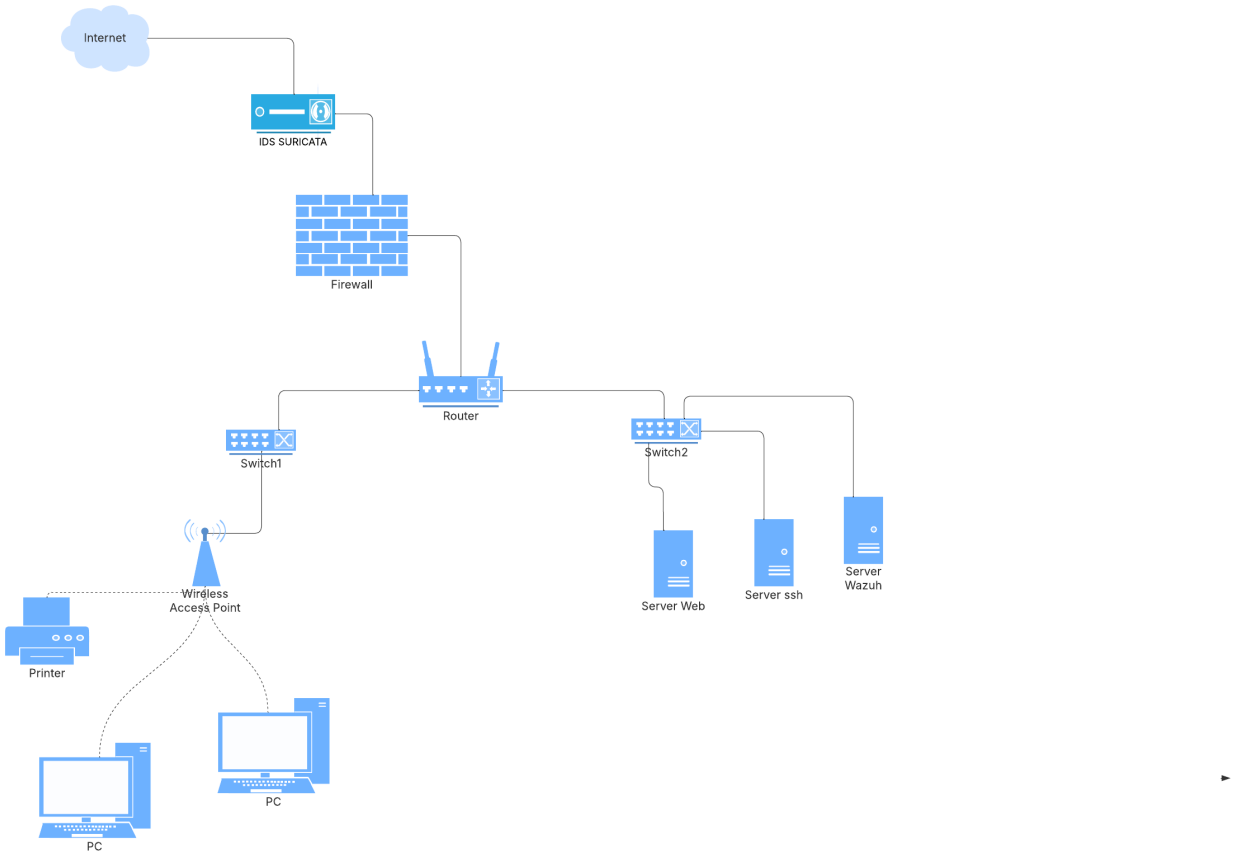


FIGURE 2.6 : Architecture après déploiement. [8]

L'architecture réseau de FUN HIGH TECH, après l'implémentation de la solution de surveillance, intègre désormais Suricata comme système de détection d'intrusions (IDS) pour analyser le trafic réseau et identifier les menaces. Wazuh, installé sur un serveur dédié, assure la surveillance des systèmes en collectant et analysant les événements provenant des serveurs et de l'IDS via ses agents déployés. Le pare-feu filtre le trafic entrant et sortant, tandis que le réseau interne reste accessible uniquement aux postes de l'entreprise. Cette solution permet une sécurité renforcée avec Suricata pour la protection réseau et Wazuh pour l'analyse des événements et la réponse aux incidents sur les systèmes.

2.3 Fonctionnement de la solution

Le système fonctionne en trois étapes clés qui sont : la **détection**, la **collecte et l'analyse** et enfin la **réponse aux intrusions**.

2.3.1 Détection

La détection repose sur Suricata et Wazuh, qui surveillent respectivement en temps réel le trafic réseau et les hôtes en analysant les paquets pour identifier des signatures d'attaques connues (scans de ports, attaques DDoS, injections SQL, tentatives de brute force, etc.). L'IDS est positionné de manière stratégique pour capturer tout le trafic passant à travers le pare-feu. Lorsqu'une activité suspecte est détectée, Suricata génère des logs d'alerte au format JSON qui sont ensuite envoyés aux systèmes de supervision pour analyse.

2.3.2 Collecte et l'analyse

Les événements issus de Suricata et des systèmes à surveiller sont collectés par les agents Wazuh, installés sur les serveurs critiques (serveur web, serveur SSH, IDS). Ces agents collectent différents types de logs : tentatives de connexion suspectes, modifications de fichiers sensibles, exécution de commandes suspectes, etc. Les logs sont ensuite envoyés au serveur Wazuh, qui applique des règles de corrélation et des analyses avancées pour identifier des attaques complexes et repérer d'éventuelles compromissions de systèmes.

2.3.3 Réponse aux intrusions

Lorsqu'une menace est confirmée, Wazuh peut déclencher des actions automatiques ou notifier les administrateurs pour une intervention rapide. Par exemple, en cas de tentative d'intrusion répétée, Wazuh peut bloquer l'adresse IP source, générer une alerte critique ou exécuter un script pour limiter les dégâts. L'interface Wazuh permet aux administrateurs de visualiser en détail les attaques et de prendre des décisions adaptées pour renforcer la sécurité du réseau et des systèmes.

Grâce à cette approche intégrée, Suricata assure une surveillance efficace du réseau, tandis que Wazuh gère la protection des systèmes et la réponse aux incidents, garantissant ainsi une défense proactive de FUN HIGH TECH contre les cyberattaques.

Conclusion

Suricata est un logiciel open source de détection d'intrusion (IDS), permettant l'inspection des paquets en profondeur. Suricata est un outil très intéressant dans la mise en place d'une sécurité réseau. Quant à Wazuh, il s'agit d'un outil qui combine des fonctionnalités de surveillance de sécurité,de détection d'inrusions,de gestion des logs et de réponse aux incidents. La complémentarité de ces deux outils représente un facteur déterminant dans la mise en place d'un système de sécurité robuste. C'est d'ailleurs une des principales raisons pour lesquelles nous avons opté pour ce système.

Chapitre 3

Solution déployée

Introduction

Dans ce dernier chapitre, nous allons installer Wazuh et effectuer les configurations adéquates. Nous allons ensuite installer Suricata et effectuer également toutes les configurations nécessaires à son fonctionnement. Pour finir, nous allons tester notre configuration en simulant quelques attaques et essayer de les détecter avec le système mis en place.

3.0.1 Préparation de l'environnement de travail

Pour effectuer les test, nous allons utiliser quatres machines virtuelles :

- **Une Virtual machine kali linux (attaquant)** : qui va permettre de réaliser les attaques.
- **Une Virtual machine Ubuntu (Serveur Wazuh)** : Le Serveur Wazuh va nous permettre de visualiser les alertes générées par Suricata ainsi que celles générées par les hôtes.Il va nous permettre de centraliser et visualiser toutes nos données.
- **Une Virtual machine Ubuntu (victime)** : qui va nous permettre de configurer suricata et qui servira egalement de cible durant l'attaque qui l'attaque par scan de port.
- **Une Virtual machine Ubuntu (victime)** : qui va nous permettre de configurer les serveurs Apache et SSH. Elle servira egalement de cible durant les attaques par injection sql et par brute force. La VM (Virtual machine) porte le nom de **DMZ (Demilitarized Zone)**.

L'image ci-dessous l'illustre :



FIGURE 3.1 : Les VM

3.1 Configuration des systèmes et test effectués

3.1.1 Installation et configuration de suricata

Suricata est un système de détection d'intrusion (IDS) et de prévention d'intrusion (IPS) open-source. Il est conçu pour être performant et polyvalent, capable de surveiller le trafic réseau en temps réel et de détecter des activités malveillantes ou suspectes. Suricata utilise des signatures et des règles pour identifier les menaces, et il peut être déployé sur divers types de réseaux.

```
mafouz@mafouz-VirtualBox:~$ sudo apt install -y suricata
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 :
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhtp2 libhyperscan5 libluajit-5.1-2 libluajit-5.1-common
  liblzma-dev libnet1 libnetfilter-queue1
Paquets suggérés :
  liblzma-doc
Les NOUVEAUX paquets suivants seront installés :
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhtp2 libhyperscan5 libluajit-5.1-2 libluajit-5.1-common
  liblzma-dev libnet1 libnetfilter-queue1 suricata
0 mis à jour, 11 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 6 363 ko dans les archives.
Après cette opération, 28,8 Mo d'espace disque supplémentaires seront utilisés.
Réception de :1 http://bj.archive.ubuntu.com/ubuntu jammy/universe amd64 libhyperscan5 amd64 5.4.0-2 [2 485 kB]
Réception de :2 https://ppa.launchpadcontent.net/oisf/suricata-stable/ubuntu jammy/main amd64 libhtp2 amd64 1:0.5.48-0ubuntu2 [75,7
kB]
Réception de :3 http://bj.archive.ubuntu.com/ubuntu jammy/main amd64 libevent-core-2.1-7 amd64 2.1.12-stable-1build3 [93,9 kB]
Réception de :4 http://bj.archive.ubuntu.com/ubuntu jammy/main amd64 libevent-pthreads-2.1-7 amd64 2.1.12-stable-1build3 [7 642 B]
Réception de :5 http://bj.archive.ubuntu.com/ubuntu jammy/universe amd64 libhiredis0.14 amd64 0.14.1-2 [32,8 kB]
Réception de :6 http://bj.archive.ubuntu.com/ubuntu jammy/universe amd64 libluajit-5.1-common all 2.1.0~beta3+dfsg-6 [44,3 kB]
Réception de :7 http://bj.archive.ubuntu.com/ubuntu jammy/universe amd64 libluajit-5.1-2 amd64 2.1.0~beta3+dfsg-6 [238 kB]
Réception de :8 http://bj.archive.ubuntu.com/ubuntu jammy/main amd64 libnet1 amd64 1.1.6+dfsg-3.1build3 [46,9 kB]
Réception de :9 http://bj.archive.ubuntu.com/ubuntu jammy/universe amd64 libnetfilter-queue1 amd64 1.0.5-2 [14,4 kB]
Réception de :10 http://bj.archive.ubuntu.com/ubuntu jammy/main amd64 liblzma-dev amd64 5.2.5-2ubuntu1 [159 kB]
Réception de :11 https://ppa.launchpadcontent.net/oisf/suricata-stable/ubuntu jammy/main amd64 suricata amd64 1:7.0.6-0ubuntu2 [3 16
6 kB]
6 363 ko réceptionnés en 15s (427 ko/s)
Préconfiguration des paquets...
Sélection du paquet libhyperscan5 précédemment désélectionné.
(Lecture de la base de données... 206549 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../00-libhyperscan5_5.4.0-2_amd64.deb ...
Dépaquetage de libhyperscan5 (5.4.0-2) ...
Sélection du paquet libevent-core-2.1-7:amd64 précédemment désélectionné.
```

FIGURE 3.2 : suricata 7.0.7

Il faut savoir que suricata est doté des plugins par défaut qui y sont constitués des fichiers ou sont écris de nombreuses règles de détections prédéfinies qu'on appelle les règles emerging.threats. Nous allons télécharger et activer ses règles, l'image ci dessous l'illustre bien :

```
root@mafouz-VirtualBox:/tmp# sudo tar -xvzf emerging.rules.tar.gz && sudo mkdir /etc/suricata/rules && sudo mv rules/*.rules /etc/su
ricata/rules
```

```
 100% - 4486304/4486304
26/9/2024 -- 20:51:57 - <Info> -- Done.
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/app-layer-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/decoder-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dhcp-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dnp3-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/dns-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/files.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/http2-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/http-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ipsec-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/kerberos-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/modbus-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/mqtt-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/nfs-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ntp-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/quic-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/rfb-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smb-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/smtp-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/ssh-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/stream-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Loading distribution rule file /usr/share/suricata/rules/tls-events.rules
26/9/2024 -- 20:51:57 - <Info> -- Ignoring file e972a1a2078ddbc4fb8ff12f8ce8253c/rules/emerging-deleted.rules
26/9/2024 -- 20:51:59 - <Info> -- Loaded 52920 rules.
26/9/2024 -- 20:51:59 - <Info> -- Disabled 14 rules.
26/9/2024 -- 20:51:59 - <Info> -- Enabled 0 rules.
26/9/2024 -- 20:51:59 - <Info> -- Modified 0 rules.
26/9/2024 -- 20:51:59 - <Info> -- Dropped 0 rules.
26/9/2024 -- 20:51:59 - <Info> -- Enabled 136 rules for flowbit dependencies.
26/9/2024 -- 20:51:59 - <Info> -- Creating directory /var/lib/suricata/rules.
26/9/2024 -- 20:51:59 - <Info> -- Backing up current rules.
26/9/2024 -- 20:51:59 - <Info> -- Writing rules to /var/lib/suricata/rules/suricata.rules: total: 52920; enabled: 39926; added: 5292
0; removed 0; modified: 0
26/9/2024 -- 20:51:59 - <Info> -- Writing /var/lib/suricata/rules/classification.config
26/9/2024 -- 20:52:00 - <Info> -- Testing with suricata -T.
26/9/2024 -- 20:52:19 - <Info> -- Done.
mafouz@mafouz-VirtualBox:~$
```

Bien que suricata soit installé et configuré,la prochaine étape consiste à lancer le service suricata

afin qui'il puisse être operationnel.

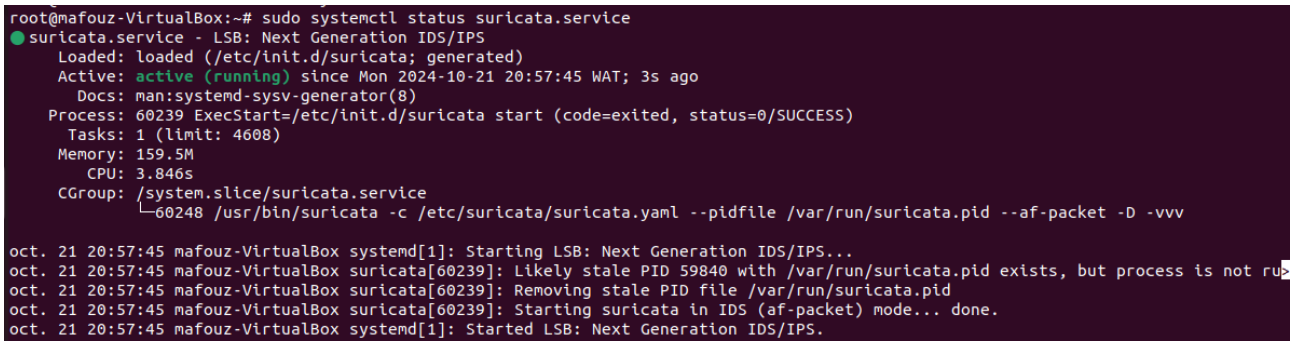


FIGURE 3.3 : demarrage service suricata

3.1.2 Installation et configuration de Wazuh et de ses composants

L'installation de Wazuh et de ses composants prend plusieurs minutes, sous prétextes de nombreuses charges de fichiers qu'il contient.

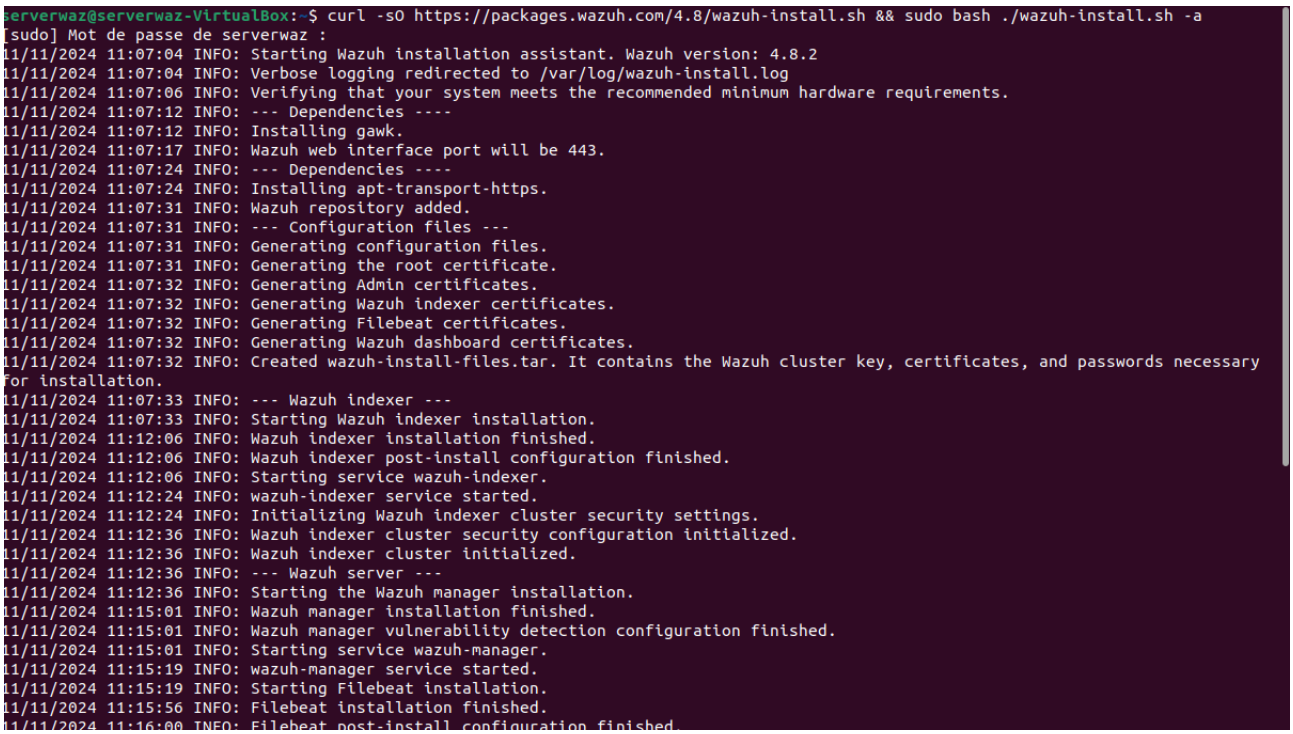


FIGURE 3.4 : Installation serveur Wazuh et ses composants

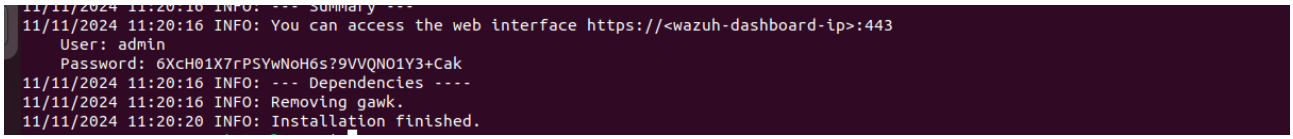


FIGURE 3.5 : Installation Wazuh

Après installation de Wazuh ainsi que de ses composants, nous allons maintenant procéder au démarrage des composants afin qu'on puisse accéder à l'interface de Wazuh.

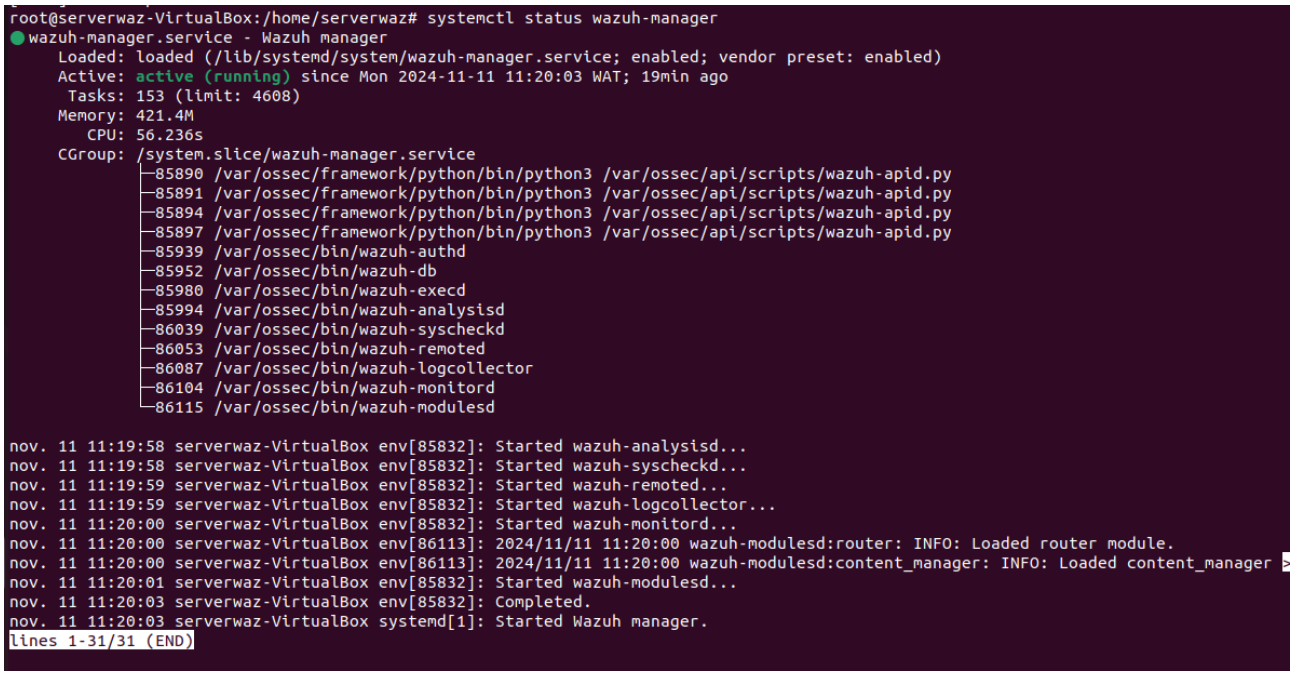


FIGURE 3.6 : Démarrage de Wazuh-manager

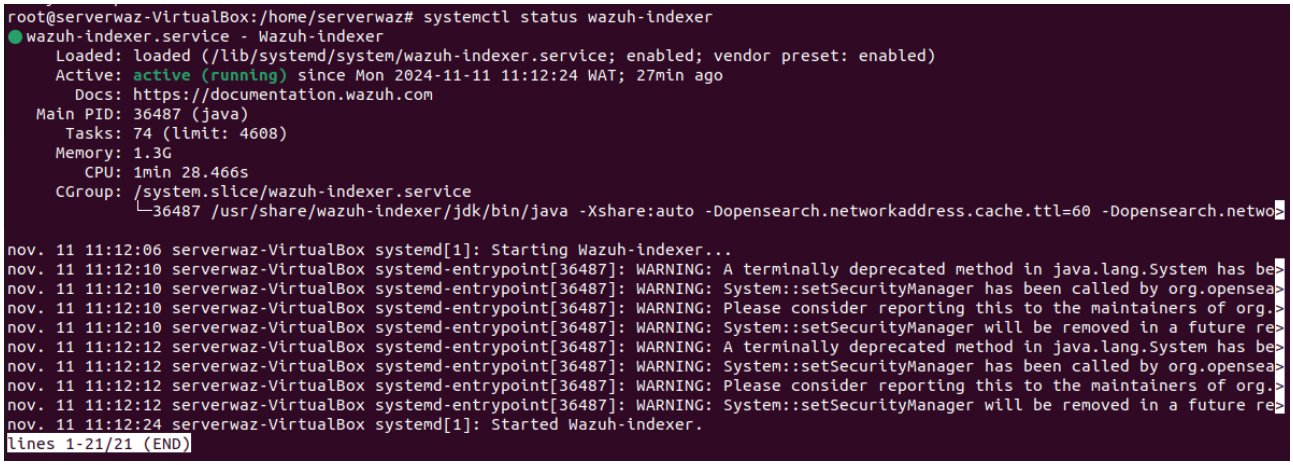


FIGURE 3.7 : Démarrage de Wazuh-indexer

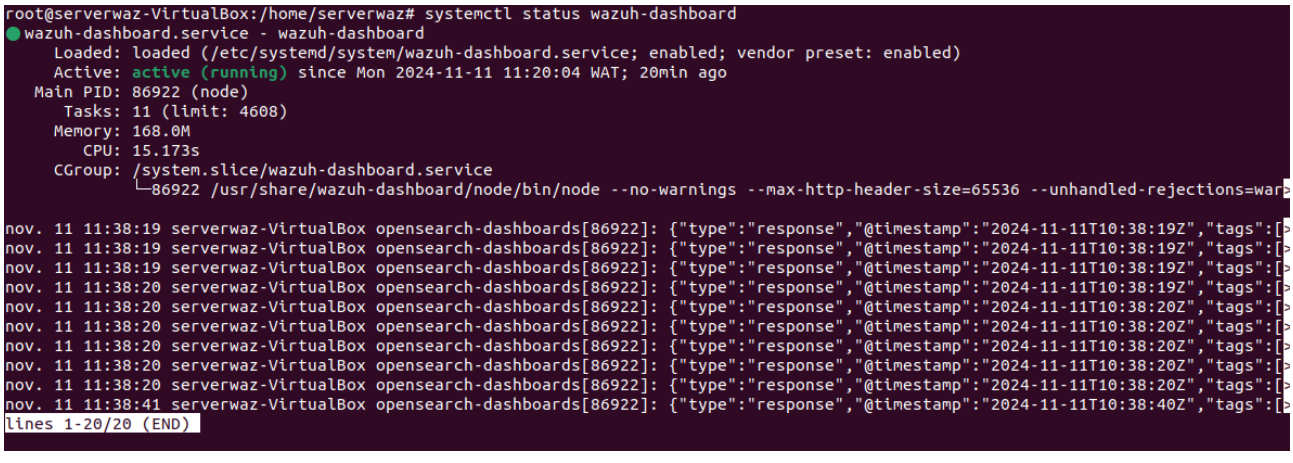


FIGURE 3.8 : Démarrage de Wazuh-dashboard

comme annoncé précédemment, nous allons accéder à l'interface de wazuh grâce à l'adresse

192.168.100.159. comme l'illustre le schéma suivant :

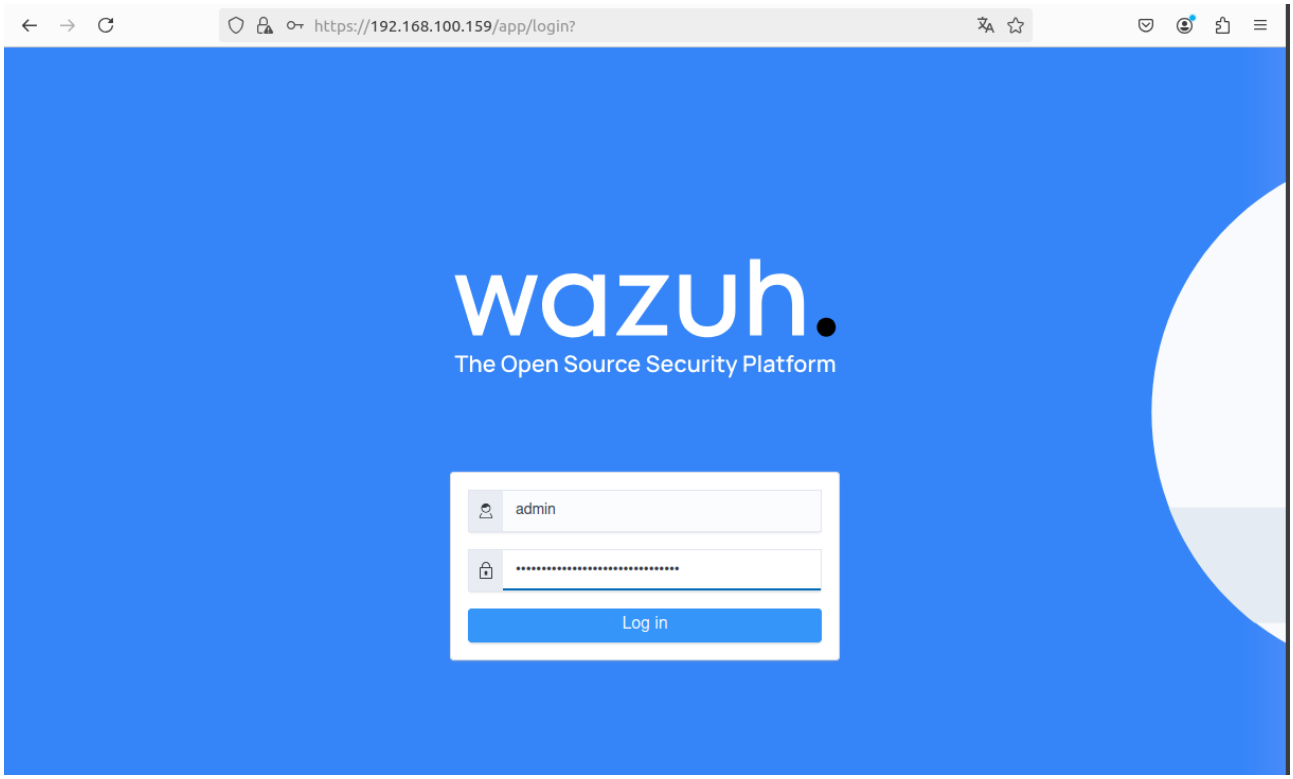


FIGURE 3.9 : Interface de connexion

- **Installation et configuration de l'agent Wazuh :** Sous notre machine ubuntu où est installé l'IDS suricata, nous allons deployer l'agent wazuh sur ce dernier afin que Wazuh puisse afficher sur son interface, le système à surveiller.



FIGURE 3.10 : installation et configuration de l'agent wazuh

Nous allons procéder ensuite au démarrage de service de l'agent wazuh afin qu'il puisse être opérationnel.

```
suricata@suricata-VirtualBox:~$ sudo systemctl status wazuh-agent
● wazuh-agent.service - Wazuh agent
     Loaded: loaded (/lib/systemd/system/wazuh-agent.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2025-01-30 14:53:14 WAT; 7s ago
    Process: 39741 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 28 (limit: 2605)
     Memory: 156.1M
        CPU: 1.978s
     CGroup: /system.slice/wazuh-agent.service
             ├─39763 /var/ossec/bin/wazuh-execd
             ├─39774 /var/ossec/bin/wazuh-agentd
             ├─39788 /var/ossec/bin/wazuh-syscheckd
             ├─39800 /var/ossec/bin/wazuh-logcollector
             └─39820 /var/ossec/bin/wazuh-modulesd

janv. 30 14:53:07 suricata-VirtualBox systemd[1]: Starting Wazuh agent...
janv. 30 14:53:07 suricata-VirtualBox env[39741]: Starting Wazuh v4.8.2...
janv. 30 14:53:08 suricata-VirtualBox env[39741]: Started wazuh-execd...
janv. 30 14:53:09 suricata-VirtualBox env[39741]: Started wazuh-agentd...
janv. 30 14:53:10 suricata-VirtualBox env[39741]: Started wazuh-syscheckd...
janv. 30 14:53:11 suricata-VirtualBox env[39741]: Started wazuh-logcollector...
janv. 30 14:53:12 suricata-VirtualBox env[39741]: Started wazuh-modulesd...
janv. 30 14:53:14 suricata-VirtualBox env[39741]: Completed.
janv. 30 14:53:14 suricata-VirtualBox systemd[1]: Started Wazuh agent.
suricata@suricata-VirtualBox:~$
```

FIGURE 3.11 : Démarrage de l'agent Wazuh

- **Installation et configuration de l'agent Wazuh :** Sous notre machine ubuntu où sont installés les serveurs Apache et SSH, nous allons deployer et lancer l'agent wazuh sur ce dernier.

```
dmz@dmz-VirtualBox:~$ wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.8.2-1_amd64.deb &&
sudo WAZUH_MANAGER='192.168.100.159' WAZUH_AGENT_NAME='Agentserv' dpkg -i ./wazuh-agent_4.8.2-1_amd64.deb
--2025-01-30 15:06:05--  https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.8.2-1_amd64.deb
Résolution de packages.wazuh.com (packages.wazuh.com)… 3.165.136.42, 3.165.136.126, 3.165.136.4, ...
Connexion à packages.wazuh.com (packages.wazuh.com)|3.165.136.42|:443… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 10260268 (9,8M) [application/vnd.debian.binary-package]
Enregistre : ‘wazuh-agent_4.8.2-1_amd64.deb’

wazuh-agent_4.8.2-1_amd64.deb    100%[=========================================================>]   9,78M   568KB/s    ds 26s

2025-01-30 15:06:41 (387 KB/s) - ‘wazuh-agent_4.8.2-1_amd64.deb’ enregistré [10260268/10260268]

[sudo] Mot de passe de dmz :
Sélection du paquet wazuh-agent précédemment désélectionné.
(Lecture de la base de données... 207375 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../wazuh-agent_4.8.2-1_amd64.deb ...
Dépaquetage de wazuh-agent (4.8.2-1) ...
Paramétrage de wazuh-agent (4.8.2-1) ...
```

FIGURE 3.12 : installation de l'agent wazuh

```
dmz@dmz-VirtualBox:~$ sudo systemctl status wazuh-agent
● wazuh-agent.service - Wazuh agent
     Loaded: loaded (/lib/systemd/system/wazuh-agent.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2025-01-30 15:08:10 WAT; 6s ago
    Process: 40963 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exited, status=0/SUCCESS)
      Tasks: 28 (limit: 2271)
     Memory: 146.6M
        CPU: 1.753s
     CGroup: /system.slice/wazuh-agent.service
             ├─40986 /var/ossec/bin/wazuh-execd
             ├─40999 /var/ossec/bin/wazuh-agentd
             ├─41013 /var/ossec/bin/wazuh-syscheckd
             ├─41028 /var/ossec/bin/wazuh-logcollector
             └─41044 /var/ossec/bin/wazuh-modulesd

janv. 30 15:08:03 dmz-VirtualBox systemd[1]: Starting Wazuh agent...
janv. 30 15:08:03 dmz-VirtualBox env[40963]: Starting Wazuh v4.8.2...
janv. 30 15:08:04 dmz-VirtualBox env[40963]: Started wazuh-execd...
janv. 30 15:08:05 dmz-VirtualBox env[40963]: Started wazuh-agentd...
janv. 30 15:08:06 dmz-VirtualBox env[40963]: Started wazuh-syscheckd...
janv. 30 15:08:07 dmz-VirtualBox env[40963]: Started wazuh-logcollector...
janv. 30 15:08:08 dmz-VirtualBox env[40963]: Started wazuh-modulesd...
janv. 30 15:08:10 dmz-VirtualBox env[40963]: Completed.
janv. 30 15:08:10 dmz-VirtualBox systemd[1]: Started Wazuh agent.
dmz@dmz-VirtualBox:~$
```

FIGURE 3.13 : Démarrage de l'agent wazuh

Nous allons maintenant accéder au dashboard de wazuh et voir tous les agents qui ont été déployer comme le montre l'image ci-dessous :

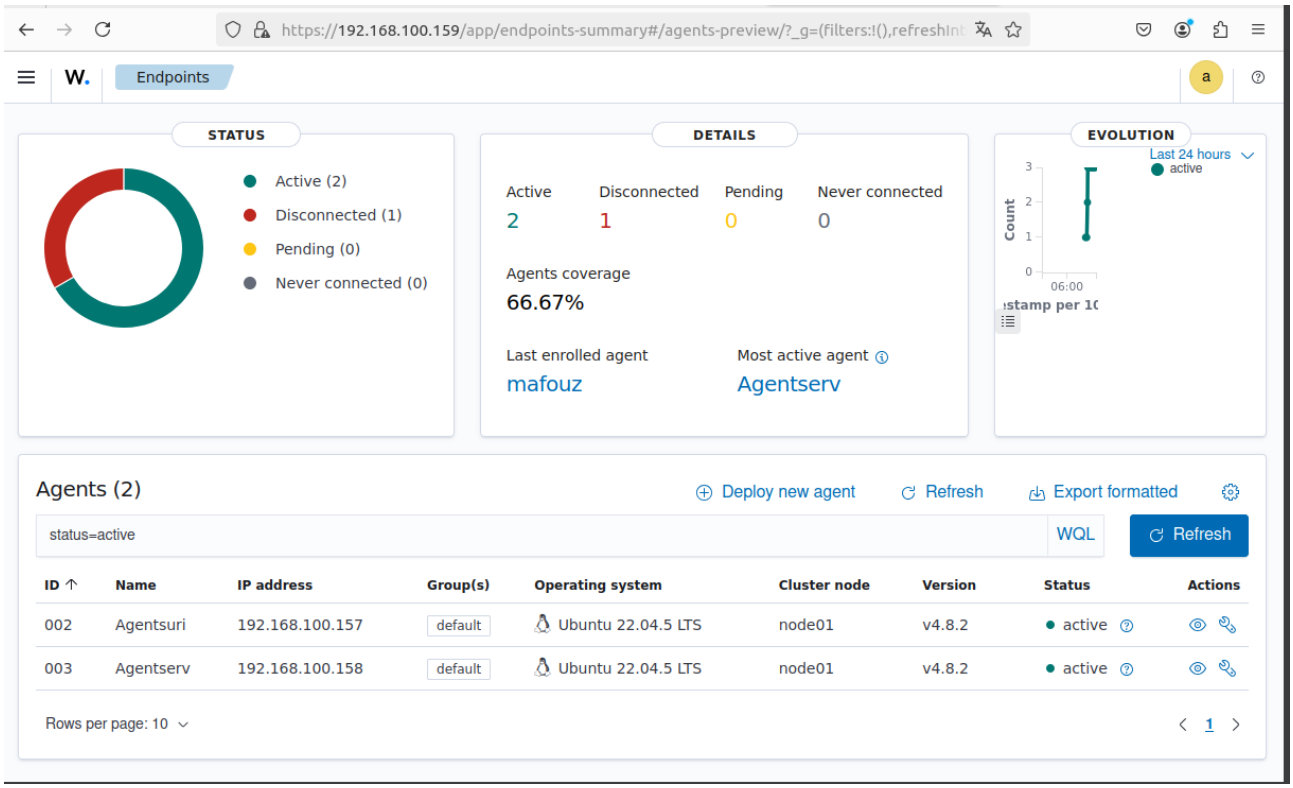


FIGURE 3.14 : image montrant les agents wazuh deployés

Une fois installé, nous allons configurer l'agent wazuh afin qu'il puisse envoyer les logs suricata afin que le serveur Wazuh puisse les collecter et les visualiser dans le dashboard. La configuration s'effectue dans le fichier **/var/ossec/etc/ossec.conf**. L'image ci-dessous l'illustre.

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

</ossec_config>
```

FIGURE 3.15 : configuration de l'agent Wazuh

Nous allons redémarrer le serveur Wazuh afin que les configurations soient prises en compte en entrant la commande suivante dans le terminal : **systemctl restart wazuh-manager**.

3.2 Tests et Résultats

Dans cette section, nous allons réaliser des tests qui vont nous permettre de tester l'efficacité du système que nous avons mis en place. Pour ce faire, nous allons réaliser quelques attaques, notamment :

- **Attaque par scan de port avec nmap**
- **Attaque par injection sql**
- **Attaque par force brute avec hydra**

3.2.1 scan de port avec nmap

NMAP ou **Network Mapper** est un utilitaire libre et open source, largement utilisé pour explorer les réseaux et effectuer des audits de sécurité. De nombreux administrateurs système et réseau le trouvent précieux pour diverses tâches telles que la cartographie du réseau, la gestion des mises à jour des services, et la surveillance de la disponibilité des hôtes et des services.Ce logiciel innovant utilise des paquets IP bruts pour déterminer quels hôtes sont actifs sur le réseau, identifier les services offerts (y compris le nom et la version de l'application), détecter les systèmes d'exploitation (y compris les versions), vérifier l'utilisation de filtres de paquets et de pare-feu, ainsi que d'autres fonctionnalités. Il a été spécialement conçu pour analyser rapidement de vastes réseaux, mais reste efficace même pour des hôtes individuels.Nous allons éffectuer un scan de port sur la cible(machine virtuelle où est installé suricata).

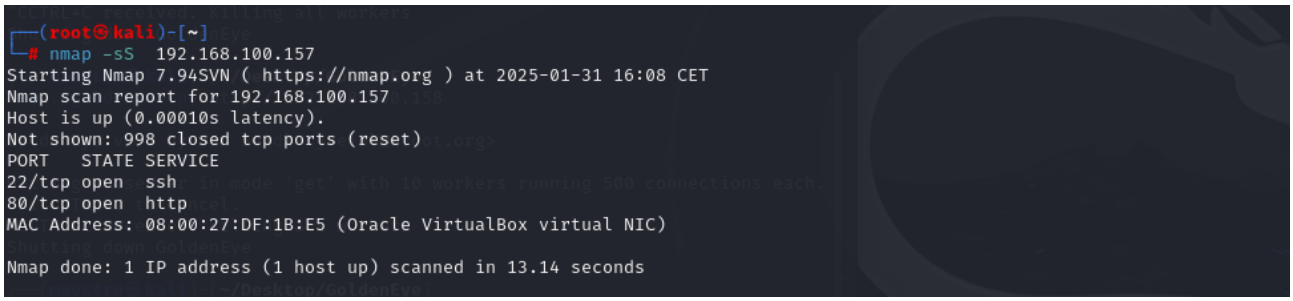


FIGURE 3.16 : Scan de port vers la machine cible

Nous pouvons constater que l'attaque a abouti et que les logs de Suricata ont bien été envoyés au serveur Wazuh. Voici les alertes générées par le système.

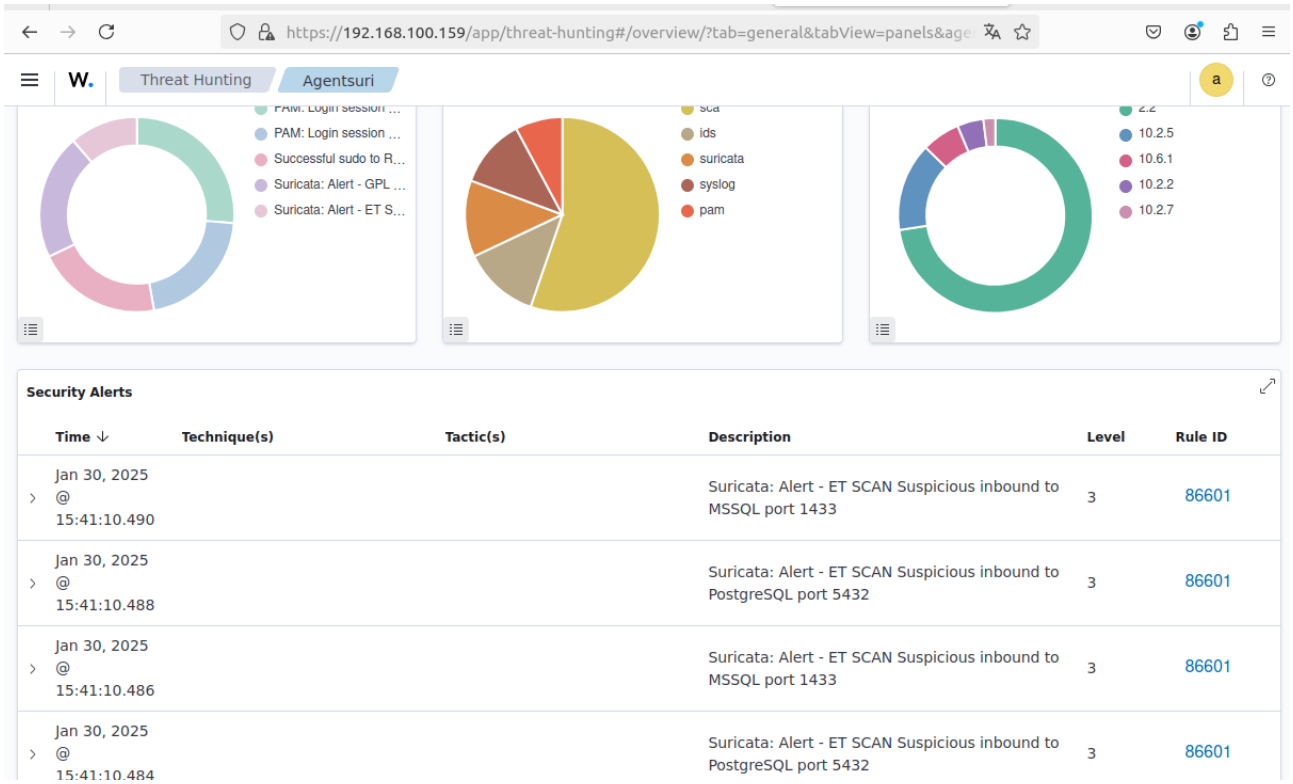


FIGURE 3.17 : Alertes générées pour le scan de port

```
01/30/2025-15:43:10.416892  [**] [1:2002910:6] ET SCAN Potential VNC Scan 5800-5820 [**] [Classification: Attempted Information Leak
] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:5811
01/30/2025-15:43:10.416892  [**] [1:2002910:6] ET SCAN Potential VNC Scan 5800-5820 [**] [Classification: Attempted Information Leak
] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:5811
01/30/2025-15:43:10.425386  [**] [1:2010939:3] ET SCAN Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially
Bad Traffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:5432
01/30/2025-15:43:10.429505  [**] [1:2010935:3] ET SCAN Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad T
raffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:1433
01/30/2025-15:43:10.425386  [**] [1:2010939:3] ET SCAN Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially
Bad Traffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:5432
01/30/2025-15:43:10.429505  [**] [1:2010935:3] ET SCAN Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad T
raffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:1433
01/30/2025-15:43:10.425386  [**] [1:2010939:3] ET SCAN Suspicious inbound to PostgreSQL port 5432 [**] [Classification: Potentially
Bad Traffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:5432
01/30/2025-15:43:10.429505  [**] [1:2010935:3] ET SCAN Suspicious inbound to MSSQL port 1433 [**] [Classification: Potentially Bad T
raffic] [Priority: 2] {TCP} 192.168.100.121:64937 -> 192.168.100.157:1433
```

FIGURE 3.18 : image montrant les logs générées par suricata

Nous pouvons aussi constater que lorsqu'on clique sur une alerte, on voit les détails sur les informations concernants les logs, cela montre que suricata a bien envoyé ses informations au serveur wazuh. L'image ci-dessous le montre :

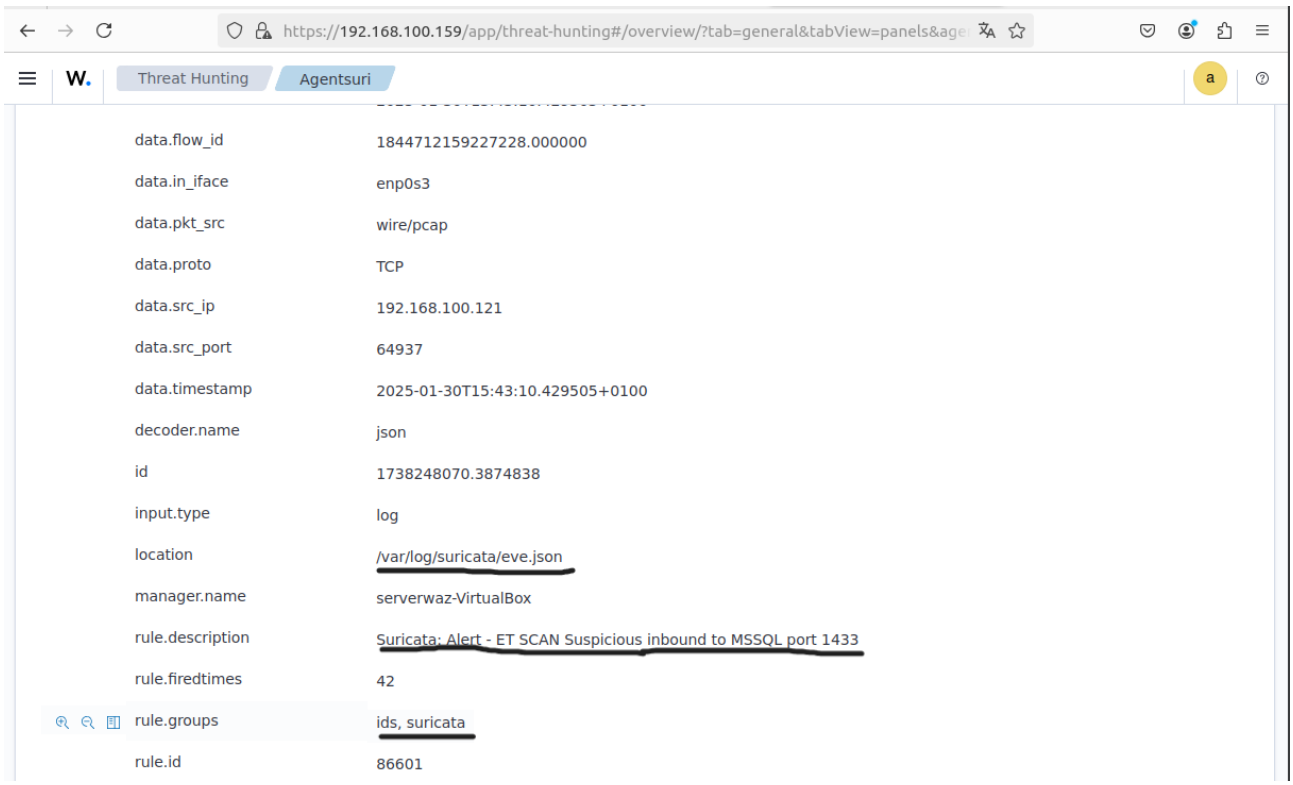


FIGURE 3.19 : informations sur l'alerte généré par suricata

3.2.2 Attaque par injection sql

Une attaque par injection SQL (SQL Injection ou SQLi) est une technique utilisée par les attaquants pour exploiter une vulnérabilité dans une application web qui interagit avec une base de données. L'objectif est d'injecter des requêtes SQL malveillantes dans un champ de saisie (comme un formulaire de connexion, une barre de recherche, etc.) pour manipuler ou voler des données.Dans notre cas ici, nous allons simuler une attaque par injection SQL sur notre serveur web Apache en envoyant une requête malveillante depuis notre machine Kali.

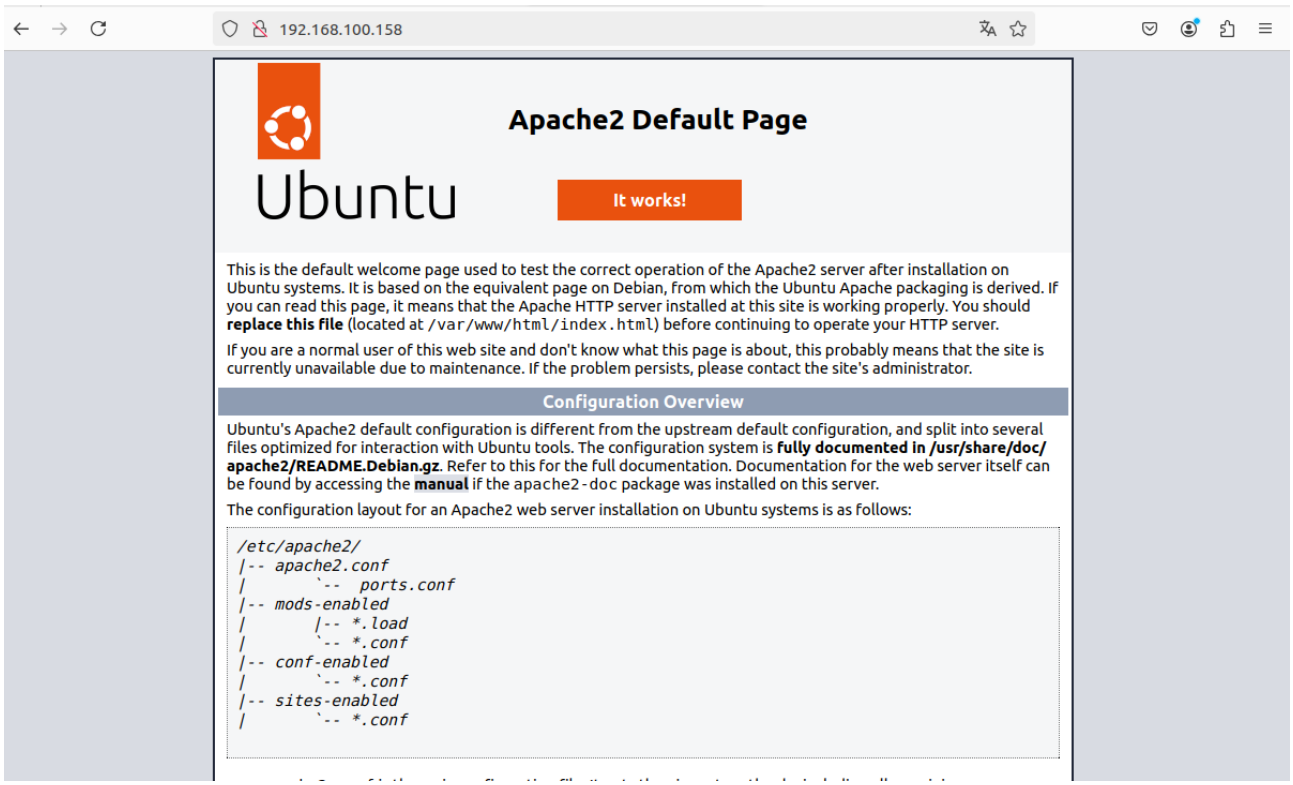


FIGURE 3.20 : image montrant le serveur Apache

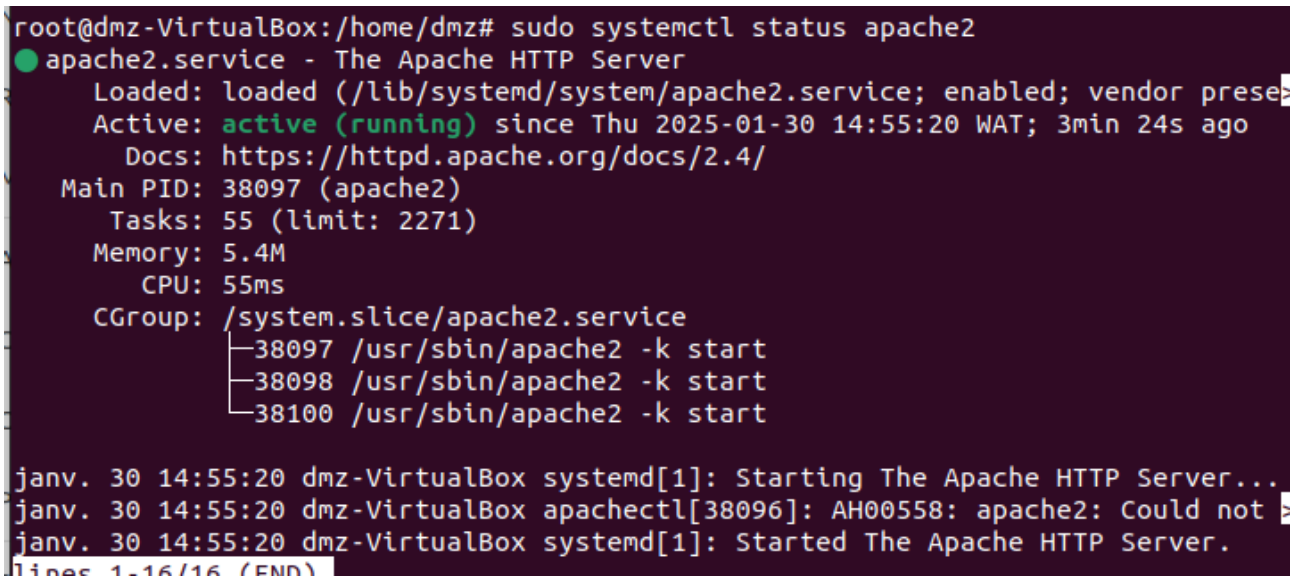


FIGURE 3.21 : Démarrage du service web apache

Une fois installé, nous allons configurer l'agent wazuh de notre machine où est installé notre serveur Apache afin qu'il puisse envoyer les logs apache au serveur Wazuh pour qu'il puisse les collecter et les visualiser dans le dashboard. La configuration s'effectue dans le fichier **/var/ossec/etc/ossec.conf**. L'image ci-dessous l'illustre.

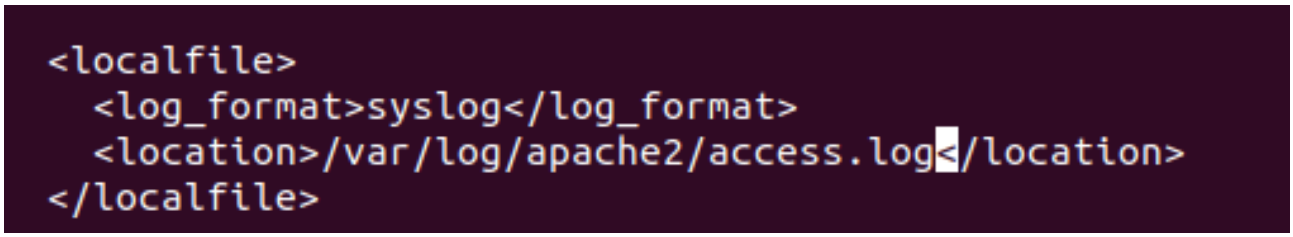


FIGURE 3.22 : configuration de l'agent Wazuh



FIGURE 3.23 : requête simulant une attaque par injection SQL

Nous pouvons constater que l'attaque a abouti et que les logs de notre serveur apache ont bien été envoyés au serveur Wazuh. Voici les alertes générées par le système.

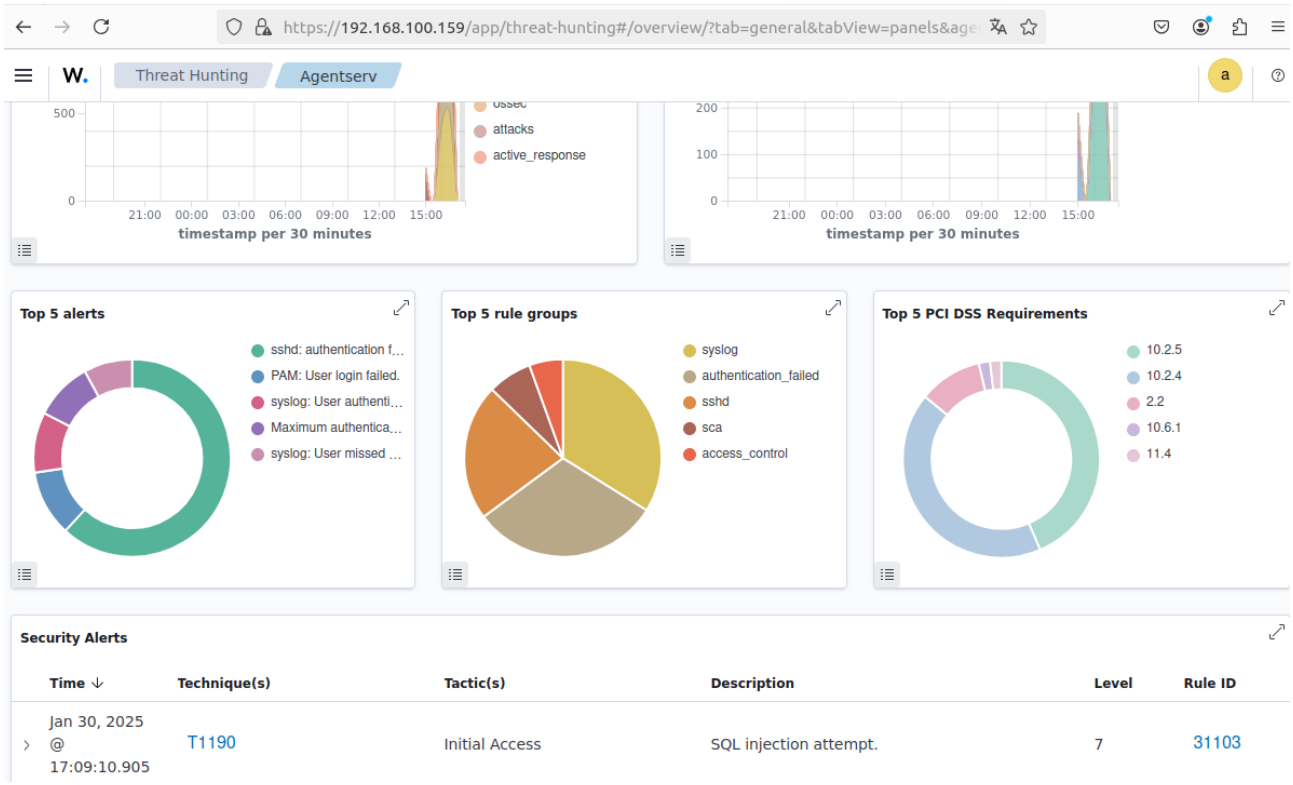


FIGURE 3.24 : Alertes générées pour l'attaque par injection SQL

Nous pouvons aussi constater que lorsqu'on clique sur une alerte, on voit les détails sur les informations concernants les logs, cela montre que les logs apache ont bien été envoyé au serveur wazuh. L'image ci-dessous nous le montre :

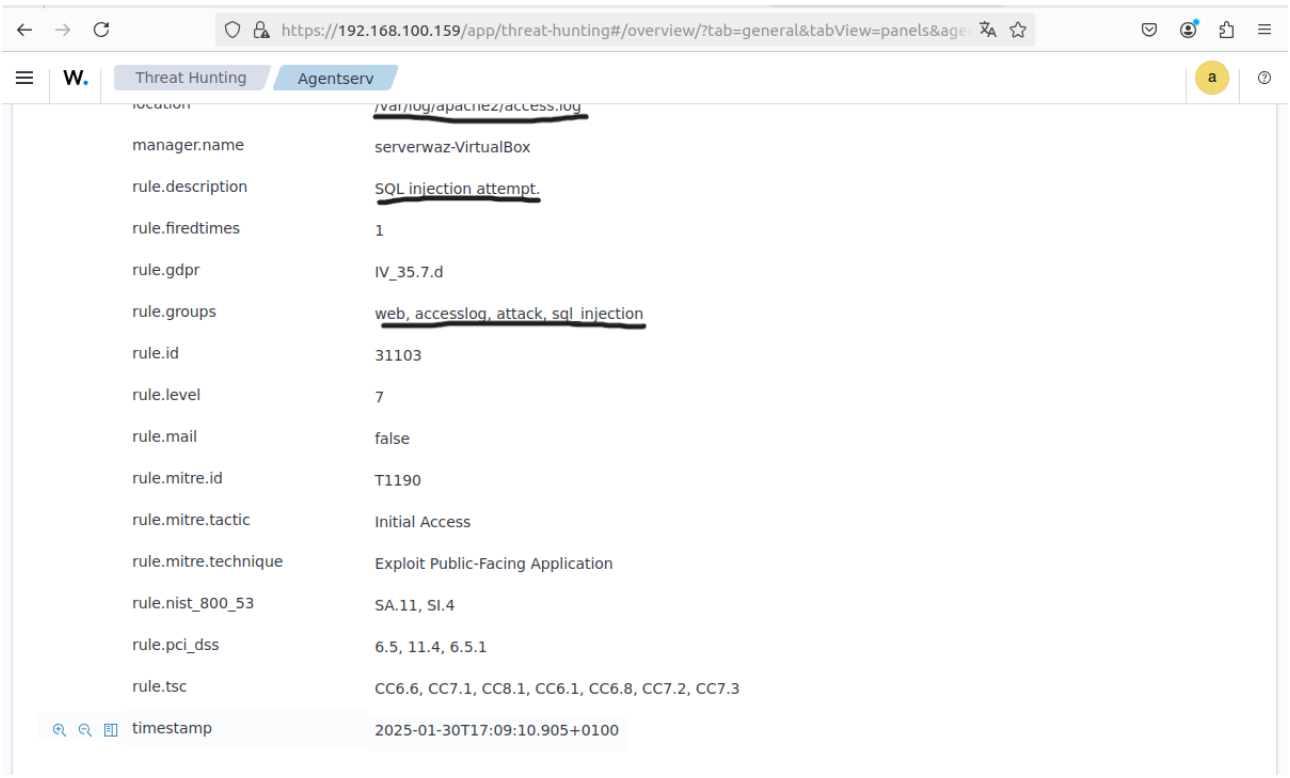


FIGURE 3.25 : informations sur l'alerte généré par l'injection SQL

3.2.3 Attaque par force brute avec hydra

hydra est un outil de test de pénétration largement utilisé en sécurité informatique pour effectuer des attaques par force brute sur des services protégés par mot de passe. Hydra est spécifiquement conçu pour tester la résistance des systèmes en tentant de deviner les mots de passe à l'aide de diverses techniques. L'utilisation de **hydra** nécessite :

- **Le choix de la cible**
- **Sélection du dictionnaire**
- **Exécution de la commande**

Nous allons simuler une attaque par force brute sur notre serveur ssh.

```
root@dmz-VirtualBox:/home/dmz# sudo apt install openssh-server -y
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 :
  ncurses-term openssh-sftp-server ssh-import-id
Paquets suggérés :
  molly-guard monkeysphere ssh-askpass
Les NOUVEAUX paquets suivants seront installés :
  ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 mis à jour, 4 nouvellement installés, 0 à enlever et 0 non mis à jour.
Il est nécessaire de prendre 751 ko dans les archives.
Après cette opération, 6 046 ko d'espace disque supplémentaires seront utilisés.
Réception de :1 http://bj.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openssh-sftp-server amd64 1:8.9p1-3ubuntu0.10 [38,9 kB]
Réception de :2 http://bj.archive.ubuntu.com/ubuntu jammy-updates/main amd64 openssh-server amd64 1:8.9p1-3ubuntu0.10 [435 kB]
Réception de :3 http://bj.archive.ubuntu.com/ubuntu jammy-updates/main amd64 ncurses-term all 6.3-2ubuntu0.1 [267 kB]
Réception de :4 http://bj.archive.ubuntu.com/ubuntu jammy/main amd64 ssh-import-id all 5.11-0ubuntu1 [10,1 kB]
751 ko réceptionnés en 2s (390 ko/s)
Préconfiguration des paquets...
Sélection du paquet openssh-sftp-server précédemment désélectionné.
(Lecture de la base de données... 207780 fichiers et répertoires déjà installés.)
Préparation du dépaquetage de .../openssh-sftp-server_1%3a8.9p1-3ubuntu0.10_amd64.deb ...
Dépaquetage de openssh-sftp-server (1:8.9p1-3ubuntu0.10) ...
Sélection du paquet openssh-server précédemment désélectionné.
Préparation du dépaquetage de .../openssh-server_1%3a8.9p1-3ubuntu0.10_amd64.deb ...
Dépaquetage de openssh-server (1:8.9p1-3ubuntu0.10) ...
Sélection du paquet ncurses-term précédemment désélectionné.
Préparation du dépaquetage de .../ncurses-term_6.3-2ubuntu0.1_all.deb ...
Dépaquetage de ncurses-term (6.3-2ubuntu0.1) ...
Sélection du paquet ssh-import-id précédemment désélectionné.
Préparation du dépaquetage de .../ssh-import-id_5.11-0ubuntu1_all.deb ...
Dépaquetage de ssh-import-id (5.11-0ubuntu1) ...
```

FIGURE 3.26 : image montrant l'installation de notre serveur ssh

```
root@dmz-VirtualBox:/home/dmz# sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2025-01-30 16:24:01 WAT; 35s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 44073 (sshd)
      Tasks: 1 (limit: 2271)
     Memory: 1.8M
        CPU: 16ms
     CGroup: /system.slice/ssh.service
             └─44073 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

janv. 30 16:24:00 dmz-VirtualBox systemd[1]: Starting OpenBSD Secure Shell server...
janv. 30 16:24:01 dmz-VirtualBox sshd[44073]: Server listening on 0.0.0.0 port 22.
janv. 30 16:24:01 dmz-VirtualBox sshd[44073]: Server listening on :: port 22.
janv. 30 16:24:01 dmz-VirtualBox systemd[1]: Started OpenBSD Secure Shell server.
```

FIGURE 3.27 : Démarrage du service ssh

Nous allons provoquer une connexion ssh avec la machine Kali Linux par force brute en utilisant la commande : **hydra -l root -P /usr/share/wordlists/wifite.txt ssh ://192.168.100.158**

```
┌──(root㉿kali)-[/home/maestro]
└─# hydra -l root -P /usr/share/wordlists/wifite.txt ssh://192.168.100.158
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
or for illegal purposes (this is non-binding, these ***** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-01-30 16:28:42
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, t
o prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 203808 login tries (l:1/p:203808), ~12738 tries per task
[DATA] attacking ssh://192.168.100.158:22/
[STATUS] 128.00 tries/min, 128 tries in 00:01h, 203682 to do in 26:32h, 14 active
^CThe session file ./hydra.restore was written. Type "hydra -R" to resume session.
```

FIGURE 3.28 : Connexion ssh par force brute

Les alertes générées par le système se présentent comme suit :

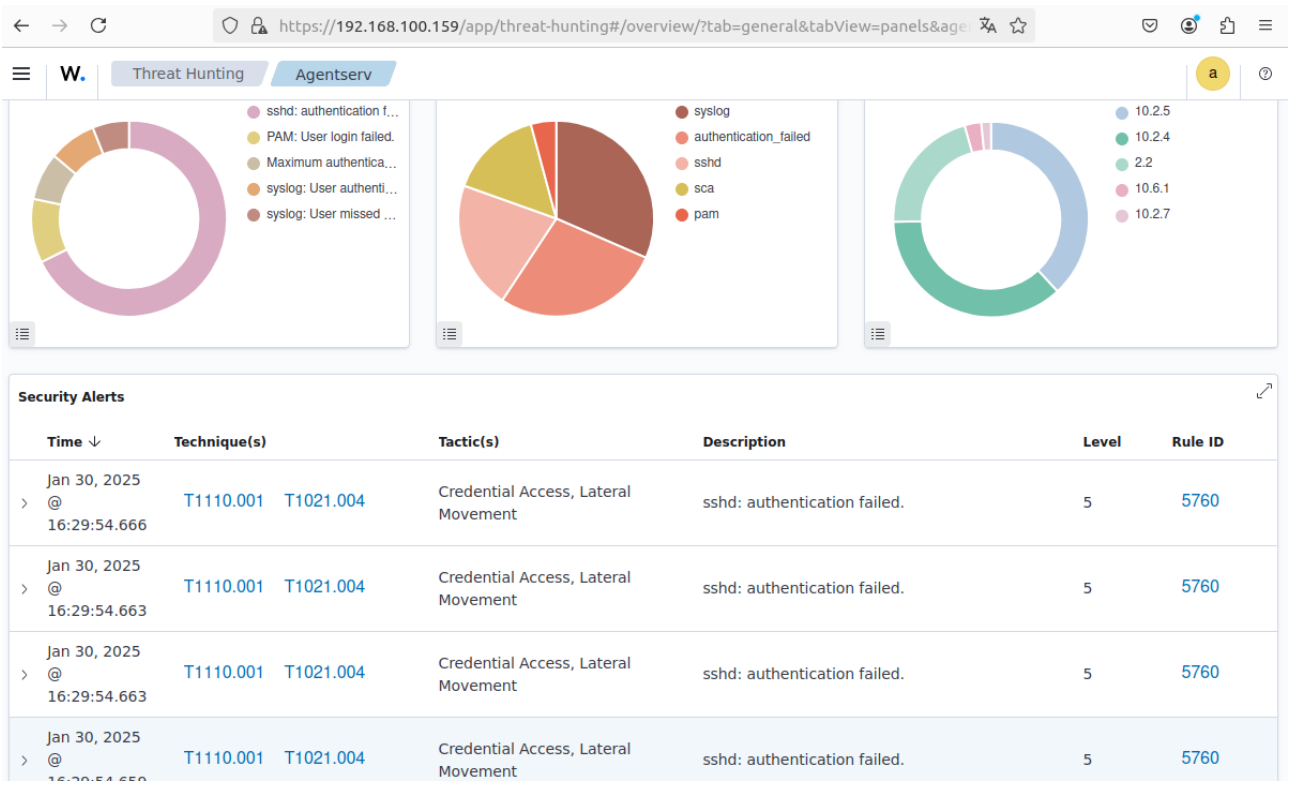


FIGURE 3.29 : Alertes générées lors de l'inondation du trafic sshd de la cible

- **Réponse aux incidents** : Wazuh offre plusieurs options de réponse aux incidents, qui permettent d'automatiser ou de déclencher des actions pour atténuer les menaces détectées. Voici les principales possibilités : **bloquage des addresses ip**, **isolation d'un hôte compromis**, **désactivation d'un compte utilisateur compromis**, **automatisation des scripts personalisés**, etc.. Pour ce rubrique, nous allons opter pour le bloquage de l'addresse ip de l'attaquant, grâce au module de **active response** que possède Wazuh. La configuration de ce bloquage va se faire dans le fichier **/var/ossec/etc/ossec.conf** de notre serveur Wazuh. D'apres la figure 3.30, nous allons nous servir des informations de l'alerte généré pour pouvoir procéder au bloquage de l'addresse ip de kali, lorsqu'il va essayer une autre tentative d'attaque au système. L'image ci-dessous nous le montre :

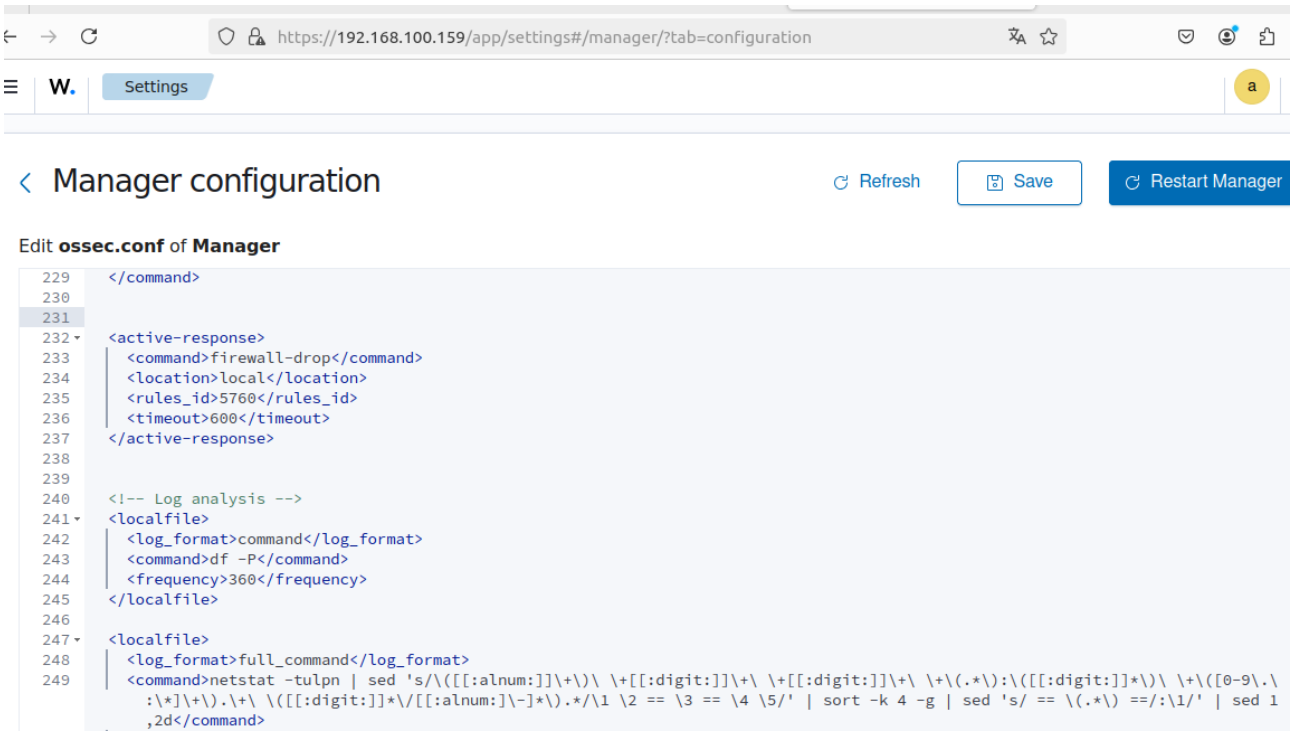


FIGURE 3.30 : Image montrant la configuration du bloquage

Nous allons redémarrer le serveur wazuh avec la commande : **systemctl restart wazuh-manager** afin que les configurations soient prises en compte.Voici l'alerte généré pour le bloquage de l'ip malveillant.

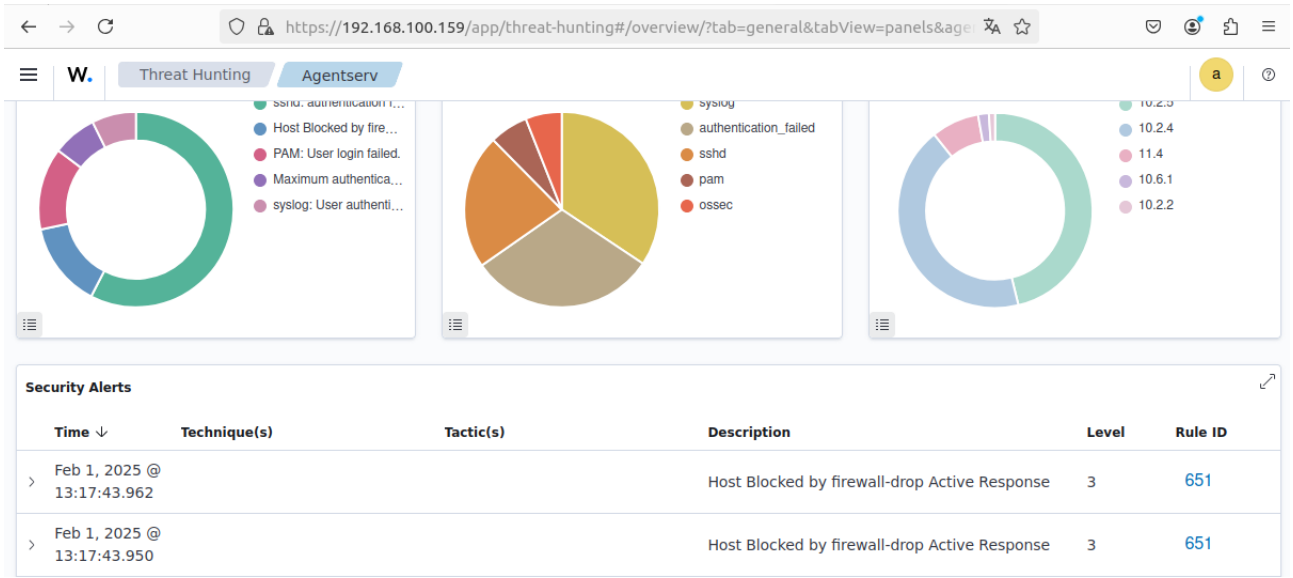


FIGURE 3.31 : Alertes générées pour le bloquage

Suite à ce blocage, nous allons essayer de communiquer avec la machine cible.Nous constatons que la communication n'est donc plus possible, comme le montre l'image ci-dessous :

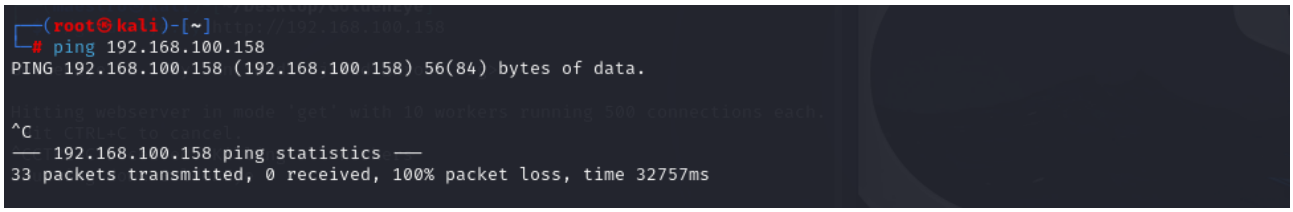


FIGURE 3.32 : Test de connectivité vers la machine cible

3.3 Performances de la solution

L'intégration de Suricata et Wazuh a permis d'obtenir une solution performante et efficace pour la détection et la gestion des intrusions. Suricata s'est révélé performant pour l'analyse du trafic réseau en identifiant les menaces en temps réel, tandis que Wazuh a complété cette surveillance en détectant les attaques sur les hôtes et en assurant la centralisation et l'analyse des logs. Grâce à l'interface de Wazuh, il a été possible de visualiser toutes les alertes et de répondre aux intrusions en appliquant des mesures de remédiation, comme le blocage automatique des adresses malveillantes. La solution mise en place s'est donc montrée réactive, offrant une bonne visibilité sur les menaces et permettant une réponse rapide aux attaques, ce qui améliore significativement la sécurité du réseau.

3.4 Faiblesses de la solution

Bien que l'intégration de Suricata et Wazuh ait apporté une amélioration significative en matière de détection et de gestion des intrusions, certaines limitations subsistent. La personnalisation des règles de Suricata s'est révélée complexe, notamment en raison du système de priorité d'exécution des règles, qui peut empêcher certaines règles personnalisées d'être prises en compte. De plus, pour garantir une détection optimale des menaces, une mise à jour régulière des règles est nécessaire, ce qui ajoute une contrainte de maintenance.

Conclusion

L'utilisation combinée de Wazuh et Suricata permettent d'obtenir un système de sécurité efficace, tant avec sa capacité de détection qu'avec sa puissance de réactivité. De plus, la facilité de passage à une échelle plus grande autorisent son déploiement dans divers environnements. Dans cette partie, nous avons illustré de manière succinte l'installation et le mécanisme de fonctionnement de WAZUH et de SURICATA. Nous avons effectué divers tests d'intrusion afin de montrer comment le système peut détecter efficacement une attaque de congestion de réseau à travers un scan, une attaque de type injection sql et une attaque par force brute et y répondre à ces derniers.

Conclusion Générale

À travers ce mémoire, nous avons exploré la mise en place d'un système de sécurité combinant Suricata et Wazuh afin de renforcer la détection et la réponse aux menaces. Dans un premier temps, nous avons présenté les concepts fondamentaux de la sécurité informatique et justifié le choix de cette solution en mettant en avant ses atouts en matière de surveillance et de gestion des incidents.

L'intégration de Suricata et Wazuh s'est révélée être une approche pertinente, combinant l'analyse approfondie du trafic réseau avec la surveillance des hôtes et la corrélation des événements de sécurité. Cette complémentarité permet de disposer d'un système de défense robuste et réactif, capable d'identifier et de signaler efficacement les activités suspectes. Grâce à sa modularité et son évolutivité, cette solution peut être adaptée à divers environnements, qu'il s'agisse de réseaux d'entreprise, d'infrastructures critiques ou d'autres contextes nécessitant un haut niveau de sécurité.

Les tests menés tout au long de cette étude ont permis de valider l'efficacité du système en détectant et en répondant à plusieurs scénarios d'attaques, confirmant ainsi sa pertinence dans une démarche de cybersécurité proactive. En combinant prévention, détection et réaction, cette approche constitue une solution fiable et évolutive face aux menaces de plus en plus sophistiquées qui pèsent sur les systèmes informatiques modernes.

Bibliographie

[1] computer security : IT security environment and contribution to the IT world, S.Sachdeva,2022

[2] Martin abadi : "sécurité réseaux en IT, une priorité" 2017

[3] SUPERVISION IT SERVERS, RESEAUX, APPLICATIONS, INCIDENTS INFORMATIQUES, GUIDE COMPLET,2023

Webographie

[4] https://www.google.com/search?sca_esv=248c1a1232d6ce85&sxsrf=AHTn8zrrDvF5n1RfsLyfIYqYcwb_6DP0hQ:1741518892001&q=logo+suricata&udm=2&fbs=ABzOT_BnMA=mosaic consulté le 09/03/2025.

[5] https://www.google.com/search?sca_esv=248c1a1232d6ce85&sxsrf=AHTn8zoizzAriG51hXAivDSZQ3s6Bbt21A:1741519211277&q=logo+wazuh&udm=2&fbs=ABzOT_BnMAgCW=mosaic consulté le 09/03/2025.

[6] https://www.google.com/search?sca_esv=248c1a1232d6ce85&sxsrf=AHTn8zp9N4fJ6jMpeWuYdtgFrS7ISsv=mosaic consulté le 09/03/2025

[7] https://lucid.app/lucidchart/5ccc30f3-e5a6-4f13-9898-be1a7e97a55b/edit?invitationId=inv_c210c1ad-9ef9-4b4b consulté le 12/03/2025

[8] https://lucid.app/lucidchart/5ccc30f3-e5a6-4f13-9898-be1a7e97a55b/edit?invitationId=inv_c210c1ad-9ef9-4b4b-8fe4-94f64f6bca6a&page=J4i0WSu9K63S# consulté le 12/03/2025

[9] Documentation oracle virtualbox, https://www.virtualbox.org/ consulté le 07/03/2025

[10] Documentation Suricata, https://suricata.io/documentation/ consulté le 07/03/2025

[11] Documentation Wazuh, https://documentation.wazuh.com/current/index.html consulté le 07/03/2025

Table des matières

Dédicace ii

Remerciements iii

Résumé iv

. iv

Abstract v

. v

Liste des figures vi

Liste des tableaux viii

Liste des acronymes ix

Introduction 1

1 Revue littéraire 3

Introduction 3

1.1 La sécurité informatique [1] 3

1.1.1 Définition de la sécurité informatique 3

1.1.2 Objectifs de la sécurité informatique 3

1.1.3 Outils nécessaires pour sécuriser un réseau informatique 4

1.2 Notion sur les Systèmes de Détection d'Intrusion [2] 4

1.2.1 Définition des systèmes de détection d'intrusion 4

1.2.2 Caractéristiques d'un système de détection d'intrusion 4

1.2.3 Fonctionnement d'un système de détection d'intrusion 5

1.2.4 Comparaison entre les différents types de Systèmes de Détection d'Intrusion . 6

1.2.4.1 Les différents types de système de détection d'intrusion 6

1.3 Notion de gestion des évènements et informations de sécurité (SIEM)[3] 7

1.3.1 Définition de SIEM 7

1.3.2 Objectifs de SIEM 7

1.3.3 Fonctionnement de SIEM 7

1.3.4 Quelques catégories de SIEM 8

1.4 Justification des choix techniques 9

1.4.1 Suricata (IDS -Sécurité réseau) 9

1.4.2 Wazuh (HIDS + SIEM - Sécurité système) 9

1.4.3 Architecture distribuée (agents + serveur Wazuh) 10

Conclusion . . . 10

2 Méthodologie utilisée **11**
Introduction . . . 11
2.1 Présentation des outils utilisés . . . 11
2.1.1 Oracle VM Virtualbox [9] . . . 11
2.1.2 Suricata [10] . . . 12
2.1.3 Wazuh [11] . . . 12
2.2 Architecture de la solution mis en place . . . 13
2.2.1 Présentation de l'architecture existante . . . 13
2.2.2 Présentation de l'architecture implémentée . . . 14
2.3 Fonctionnement de la solution . . . 15
2.3.1 Détection . . . 15
2.3.2 Collecte et l'analyse . . . 15
2.3.3 Réponse aux intrusions . . . 15
Conclusion . . . 15

3 Solution déployée **16**
Introduction . . . 16
3.0.1 Préparation de l'environnement de travail . . . 16
3.1 Configuration des systèmes et test effectués . . . 17
3.1.1 Installation et configuration de suricata . . . 17
3.1.2 Installation et configuration de Wazuh et de ses composants . . . 19
3.2 Tests et Résultats . . . 23
3.2.1 scan de port avec nmap . . . 24
3.2.2 Attaque par injection sql . . . 25
3.2.3 Attaque par force brute avec hydra . . . 28
3.3 Performances de la solution . . . 32
3.4 Faiblesses de la solution . . . 32
Conclusion . . . 32

Conclusion **33**

Bibliographie **34**

Webographie **35**

Table des matières **36**