Міністерство освіти і науки України
Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра

на тему: «Проєкт SIEM та XDR підсистеми для SOC на базі рішення Wazuh»

Виконав: студент 4-го курсу групи КБ-21-2
спеціальності 125 «Кібербезпека»
_____ Денис АТАМАНЧУК
Керівник: Старший викладач кафедри КІ та КБ
_____ Єлизавета ТРОКОЗ
Рецензент: Завідувач кафедри КІ та КБ, к.т.н., доцент
_____ Андрій ЄФІМЕНКО

Житомир – 2025

Державний університет «Житомирська політехніка»
Факультет інформаційно-комп'ютерних технологій
Кафедра комп'ютерної інженерії та кібербезпеки
Спеціальність 125 «Кібербезпека»
Освітня програма «Кібербезпека»

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
Андрій Єфіменко
3 березня 2025 р.

ЗАВДАННЯ
на випускну кваліфікаційну роботу бакалавра

Студент: Атаманчук Денис Олегович

Тема роботи: «Проєкт SIEM та XDR підсистеми для SOC на базі рішення Wazuh»

затверджена Наказом університету від 3 березня 2025 р. № 80/с

Термін здачі студентом закінченої роботи 11 червня 2025 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):

Об'єктом дослідження є процес проєктування, розгортання та налаштування підсистеми моніторингу подій безпеки SOC на базі рішення Wazuh SIEM та XDR, зокрема методи моніторингу, виявлення та реагування на кіберзагрози.

Предметом дослідження є архітектура, функціональність, процес проєктування та практична реалізації підсистеми для SOC на базі рішення Wazuh.

Обладнання та програмне забезпечення:

1. Платформа інформаційної безпеки SIEM та XDR – Wazuh
2. Системи журналювання та аудиту подій Syslog, Sysmon, AuditD
3. Мови програмування Python та Bash
4. Інтернет браузер Microsoft Edge
5. Програмний емулятор терміналу PuTTY
6. Середовище емуляції кінцевих вузлів та пристроїв Virtual Box
7. ОС Windows 11, Linux Ubuntu Server, Linux Ubuntu, macOS Big Sur

Консультанти з випускної кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Єфіменко А.А.	03.03.2025	03.03.2025
2	Трокоз Є.М.	16.04.2025	16.04.2025
3	Трокоз Є.М.	20.05.2025	20.05.2025

Студент _____ Денис АТАМАНЧУК

Керівник _____ Єлизавета ТРОКОЗ

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Видача завдання	03.03.2025	Виконано
2	Розборка плану роботи	10.04.2025	Виконано
3	Опрацювання теоретичного матеріалу	15.04.2025	Виконано
4	Проєктування підсистеми та всіх механізмів	25.04.2025	Виконано
5	Практична реалізація спроєктованої підсистеми	09.05.2025	Виконано
6	Тестування налаштованої підсистеми захисту	01.06.2025	Виконано
7	Подача чернетки керівнику на перевірку	11.06.2025	Виконано
8	Отримання відгуку керівника		
9	Допуск завідувача кафедри		
10	Подання роботи на рецензування		
11	Захист		

Студент _____ Денис АТАМАНЧУК

Керівник _____ Єлизавета ТРОКОЗ

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи складається зі вступу, чотирьох розділів, висновку, та списку використаної літератури. Текстова частина викладена на 139 сторінках друкованого тексту. В роботі наведено 155 рисунків та 3 таблиці. Список використаних джерел містить 38 найменування і займає 6 сторінки. Загальний обсяг роботи – 155 сторінок.

Метою кваліфікаційної роботи є дослідження архітектури, функціональних можливостей Wazuh як SIEM та XDR підсистем для SOC, зокрема методи моніторингу, виявлення та реагування на кіберзагрози, а також проєктування та реалізація дослідженої підсистеми для SOC на базі Wazuh.

У межах кваліфікаційного проєкту було виконано повне розгортання SIEM та XDR підсистеми на базі платформи Wazuh для потреб SOC. Реалізовано компоненти для збору логів, виявлення шкідливої активності, аномальної поведінки, спроб експлуатації вразливостей, а також автоматизованого реагування на інциденти. Особливу увагу приділено налаштуванню модулів інтеграції з VirusTotal, AbuseIPDB, засобів моніторингу PowerShell активності, контролю цілісності файлів та виявлення Rootkit загроз. Додатково налаштовано модулі інвентаризації систем, аналізу конфігурацій та управління вразливостями.

У результаті тестування підтверджено ефективність реалізованої SIEM та XDR підсистеми для виявлення поширених типових кіберзагроз та забезпечення автоматизованого реагування. Створена система може бути використана як основа для побудови повноцінного SOC у невеликих організаціях.

Ключові слова: WAZUH, SIEM, XDR,THREAT DETECTION, THREAT RESPONSE, PROTECTION, VULNERABILITY, SECURITY INCIDENT, INFORMATION SECURITY.

ANNOTATION

The explanatory note for the qualification project consists of an introduction, four chapters, a conclusion, and a list of references. The textual part comprises 139 pages of printed material and includes 155 illustrations and 3 tables. The list of references contains 38 sources and spans 6 pages. The total volume of the work is 155 pages.

The objective of this qualification project is to explore the architecture and functional capabilities of Wazuh as a SIEM and XDR subsystem for a Security Operations Center, with a particular focus on methods for monitoring, detecting, and responding to cybersecurity threats, as well as the design and implementation of the investigated subsystem for SOC based on the Wazuh platform.

As part of the project, a complete deployment of a SIEM and XDR subsystem based on the Wazuh platform was carried out to meet SOC requirements. Functional components were implemented to enable log collection, detection of malicious activity, anomalous behavior, and exploitation attempts, as well as automated incident response. Special attention was paid to configuring integrations with VirusTotal and AbuseIPDB, PowerShell activity monitoring, file integrity control, and rootkit detection. Additionally, modules for system inventory, configuration analysis, and vulnerability management were configured.

The testing results confirmed the effectiveness of the implemented SIEM and XDR subsystem in detecting common types of cyber threats and enabling automated response. The developed system can serve as a foundation for creating a fully functional SOC in small sized organizations.

Keywords**:** WAZUH, SIEM, XDR, THREAT DETECTION, THREAT RESPONSE, PROTECTION, VULNERABILITY, SECURITY INCIDENT, INFORMATION SECURITY.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ 8
ВСТУП 9
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ПОБУДОВИ SIEM ТА XDR ПІДСИСТЕМ ДЛЯ SOC 11
1.1. Огляд Security Operations Center 11
1.1.1. Основні функції та завдання SOC 11
1.1.2. Архітектура та компоненти SOC 13
1.1.3. Ролі та обов’язки спеціалістів SOC 14
1.1.4. Проблеми сучасного SOC 16
1.2. Огляд технологій SIEM та XDR 19
1.2.1. Концепція SIEM 19
1.2.2. Концепція XDR 22
1.2.3. Переваги інтеграції та порівняння SIEM з XDR 23
1.3. Огляд та порівняння Wazuh з іншими платформами для моніторингу інформаційної безпеки 25
1.3.1. Огляд рішення Wazuh та його архітектура 25
1.3.2. Основні функції та можливості Wazuh 32
1.3.3. Порівняльний аналіз Wazuh з конкурентами 35
1.3.4. Ключові переваги Wazuh у контексті SOC 39
РОЗДІЛ 2. ПРОЄКТУВАННЯ SIEM ТА XDR ПІДСИСТЕМИ ДЛЯ SOC НА БАЗІ WAZUH 42
2.1. Аналіз типових ризиків і визначення пріоритетних завдань Wazuh в контексті SOC 42
2.2. Огляд стандартних політик та правил виявлення загроз у Wazuh 44
2.3. Проєктування механізмів виявлення загроз для рішення Wazuh 47
2.4. Проєктування механізмів реагування на загрози для рішення Wazuh 51
2.5. Огляд та проєктування додаткових модулів у Wazuh для SOC 53
2.6. Проєктування інформаційних панелей для моніторингу показників 55
2.7. Проєктування загальної інфраструктури для системи Wazuh 57
РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ СПРОЄКТОВАНОЇ ПІДСИСТЕМИ 61
3.1. Налаштування інфраструктури та розгортання Wazuh 61

3.2. Налаштування механізмів збору логів та виявлення загроз 74
3.2.1. Налаштування модуля File Integrity Monitoring..... 74
3.2.2. Конфігурування інтеграції з сервісом VirusTotal 76
3.2.3. Налаштування розширеного збору логів з Microsoft Defender 78
3.2.4. Налаштування виявлення виконання шкідливих команд на ОС Windows та Linux з використанням Sysmon та AuditD 80
3.2.5. Налаштування виявлення аномальних мережевих з'єднань..... 85
3.2.6. Конфігурування інтеграції з сервісом AbuseIPDB 91
3.2.7. Налаштування виявлення технік експлуатації Windows PowerShell..... 95
3.2.8. Налаштування виявлення шкідливого ПЗ типу Rootkit..... 96
3.3. Налаштування механізмів реагування на загрози 98
3.3.1. Налаштування модуля Active Response для видалення шкідливих файлів виявлених сервісом VirusTotal 99
3.3.2. Налаштування реагування на SSH атаки типу грубої сили..... 103
3.3.3. Налаштування блокування шкідливих IP адрес виявлених сервісом AbuseIPDB та внаслідок аномальної мережевої активності 104
3.4. Налаштування додаткових модулів Wazuh 107
3.5. Конфігурування інформаційних дошок..... 111
РОЗДІЛ 4. ТЕСТУВАННЯ НАЛАШТОВАНОЇ ПІДСИСТЕМИ ТА ПРАКТИЧНЕ ВИКОРИСТАННЯ WAZUH SIEM ТА XDR ПІДСИСТЕМИ 116
4.1. Тестування реалізованих у Wazuh механізмів виявлення та реагування на загрози 116
4.2. Практичне використання додаткових модулів Wazuh в повсякденних завданнях SOC..... 128
ВИСНОВКИ..... 132
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 133
ДОДАТКИ..... 139

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ОЦБ – Операційний Центр Безпеки

ОС – Операційна Система

OS – Operating System

SOC – Security Operations Center

SIEM – Security information and event management

EDR – Endpoint detection and response

XDR – Extended detection and response

HIDS – Host-Based Intrusion Detection System

IDS – Intrusion detection system

IPS – Intrusion prevention system

PC – Personal Computer

MAC – Media Access Control

IP – Internet Protocol

SSH – Secure Shell

SSL – Secure Socket Layer

FIM – File intergrity monitor

API – Application Programming Interface

JSON – JavaScript Object Notation

XML – Extensible Markup Language

ВСТУП

Актуальність теми. Зі зростанням кількості та складності кіберзагроз організації всіх рівнів стикаються з необхідністю впровадження ефективних інструментів для захисту своїх інформаційних систем. У сучасному світі, де інформація є одним з найцінніших ресурсів, питання забезпечення її безпеки набуває критичного значення. Одними з основних засобів захисту є системи для моніторингу інформаційної безпеки та виявлення загроз, такі як SIEM та XDR. Використання відкритих платформ для цих цілей, зокрема Wazuh, дає змогу створювати масштабовані та ефективні рішення, що поєднують в собі високу гнучкість, економічність і можливість інтеграції з іншими інструментами кібербезпеки. Платформа Wazuh, як потужне рішення для моніторингу, аналізу та реагування на інциденти безпеки, дозволяє значно підвищити рівень захисту інформаційних ресурсів організацій.

Метою кваліфікаційної роботи є дослідження архітектури, функціональних можливостей Wazuh як SIEM та XDR підсистем для SOC, зокрема методи моніторингу, виявлення та реагування на кіберзагрози, а також проєктування та реалізація дослідженої підсистеми для SOC на базі Wazuh.

Визначена мета обумовлює наступні завдання:

- Проаналізувати теоретичні основи побудови SIEM та XDR підсистем у рамках структурного підрозділу Security Operations Center;
- Дослідити архітектуру, функціональні можливості та основні компоненти відкритої платформи Wazuh як рішення для SIEM та XDR;
- Провести порівняльний аналіз Wazuh з іншими відомими платформами для моніторингу інформаційної безпеки, виявити його переваги та недоліки;
- Спроєктувати механізми виявлення кіберзагроз на базі Wazuh, включаючи інтеграції зі сторонніми сервісами;
- Спроєктувати механізми автоматизованого реагування на інциденти, використовуючи модуль Active Response з налаштуванням блокування шкідливих файлів, IP адрес та виконання шкідливих команд;

- Розробити загальну інфраструктуру для ефективного розгортання та конфігурування підсистеми Wazuh для SOC;
- Спроєктувати додаткові конфігурування вбудованих модулів включно з налаштуванням інвентаризації та побудовою інформаційних дошок;
- Провести практичну реалізацію та тестування спроєктованої SIEM та XDR підсистеми на базі платформи Wazuh, а саме перевірити ефективність виявлення загроз, коректність реагування та придатність до використання в повсякденних завданнях SOC.

Об'єктом дослідження є процес проєктування, розгортання та налаштування підсистеми моніторингу подій безпеки SOC на базі рішення Wazuh SIEM та XDR, зокрема методи моніторингу, виявлення та реагування на поширені кіберзагрози.

Предметом дослідження є методи та технології розгортання, конфігурування та тестування платформи Wazuh як інструмента для виявлення загроз та реагування на поширені кіберзагрози.

У процесі виконання кваліфікаційної роботи було здійснено дослідження, проєктування, реалізацію та тестування SIEM та XDR підсистеми на базі Wazuh.

Результати тестування доводять ефективність спроєктованої та розробленої SIEM та XDR підсистеми для SOC на базі Wazuh. Створена підсистема може бути використана як основа для побудови SOC у невеликих та середніх організаціях.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ПОБУДОВИ SIEM ТА XDR ПІДСИСТЕМ ДЛЯ SOC

1.1. Огляд Security Operations Center

1.1.1. Основні функції та завдання SOC

Операційний центр безпеки (Security Operations Center, SOC) – це централізований підрозділ в організації, який відповідає за постійний моніторинг, виявлення, аналіз та реагування на кіберзагрози.

SOC виконує низку важливих функцій, спрямованих на забезпечення безпеки інформаційних систем [8]:

- *Моніторинг безпеки.* SOC здійснює цілодобовий нагляд за всіма аспектами інформаційної безпеки організації. Це включає відстеження мережевого трафіку, аналіз журналів подій та виявлення аномальної активності, яка може свідчити про потенційні загрози;
- *Виявлення та аналіз загроз.* За допомогою сучасних інструментів та аналітики, SOC виявляє потенційні кіберзагрози, такі як вторгнення, шкідливе програмне забезпечення або інші аномалії, які можуть вплинути на безпеку організації;
- *Реагування на інциденти.* Після виявлення загрози SOC розробляє та впроваджує стратегії реагування, які включають ізоляцію постраждалих систем, відновлення нормальної роботи та запобігання повторенню подібних інцидентів у майбутньому;
- *Покращення безпеки.* На основі аналізу інцидентів та виявлених вразливостей, SOC розробляє та впроваджує заходи для покращення загальної безпеки організації. Це включає оновлення політик безпеки, навчання персоналу та впровадження нових технологічних рішень.

Основними завданнями SOC наразі є розробка та впровадження політик безпеки, оцінка та управління ризиками, забезпечення відповідності стандартам та підвищення обізнаності персоналу щодо можливих кіберзагроз (Рис 1.1).

Розробка та впровадження політик безпеки відіграє критичну роль у забезпеченні інформаційної безпеки організації. Ці політики визначають правила, процедури та стандарти, спрямовані на захист інформаційних ресурсів від різноманітних загроз. Вони формують основу для управління ризиками та забезпечення відповідності нормативним вимогам [24].

В свою чергу оцінка та управління ризиками передбачає проведення регулярних тестування на наявність загроз та їх виправлення. Процес оцінки ризиків передбачає визначення важливості ризику, чи необхідно його зменшувати.

Забезпечення відповідності стандартам включає в себе дотриманням внутрішніх та зовнішніх стандартів безпеки, включаючи нормативно-правові вимоги. Це допомагає SOC виявляти можливі недоліки та підвищувати рівень координації, оскільки регулярні аудити, необхідні для підтвердження відповідності стандартам, виявляють потенційні слабкі місця в системах безпеки.

Підвищення обізнаності працівників та персоналу є одним з найважливіших ступенів захисту від потенційних загроз націлених на викрадення персональних та корпоративних даних, адже не завжди автоматизованим системам вдається запобігти проникненню загроз. Саме тому важливо, щоб працівники були добре обізнані про можливі ризики та методи їх уникнення .



Рисунок 1.1 – Основні завдання Security Operations Center

Сфери діяльності SOC поділяються на безпеку мережі, безпеку кінцевих точок, хмару безпеку та фізичну безпеку.

Безпека мережі передбачає захист мережевої інфраструктури від несанкціонованого доступу та атак, це є однією з головних задач SOC.

В свою чергу під безпекою кінцевих точок мається на увазі захист комп'ютерів, мобільних пристроїв та серверів від шкідливого програмного забезпечення та інших потенційних загроз [8].

Хмарна безпека передбачає забезпечення захисту даних та додатків, розміщених у хмарі.

При наявності фізичної безпеки спеціалісти займаються захистом фізичних ресурсів, таких як сервери, мережеве обладнання та інші критично важливі компоненти інфраструктури, від різноманітних загроз, включаючи несанкціонований доступ, крадіжки, саботаж та природні катастрофи.

1.1.2. Архітектура та компоненти SOC

Архітектура Операційного Центру Безпеки є комплексною системою, що поєднує різні компоненти та технології для забезпечення ефективного моніторингу, виявлення та реагування на кіберзагрози [17]. Основні елементи архітектури SOC включають:

- *Людські ресурси.* Провідним компонентом SOC є команда фахівців, яка включає аналітиків різних рівнів, інженерів з безпеки та керівників. Розвиток персоналу, утримання співробітників і залучення талантів є критичними для успішної роботи SOC.
- *Процеси та SOP (Standard Operation Procedure).* SOC впроваджує стандартизовані процеси та процедури для ефективного реагування на інциденти, включаючи виявлення, аналіз, ескалацію та відновлення. Ці процедури допомагають забезпечити швидке та скоординоване реагування на загрози.
- *Технології.* SOC використовує різноманітні інструменти та технології для моніторингу та аналізу даних. До них відносяться системи управління

інформацією та подіями безпеки (SIEM), інструменти для аналізу мережевого трафіку, системи виявлення вторгнень (IDS), а також платформи для автоматизації реагування на інциденти.

- *Інтеграція з іншими системами.* Ефективна архітектура SOC передбачає інтеграцію з іншими системами безпеки, такими як фаєрволи, антивірусні програми та системи контролю доступу. Це забезпечує комплексний підхід до захисту інформаційних ресурсів організації.

Завдяки цим компонентам, SOC здатний забезпечити цілісність, конфіденційність та доступність інформаційних систем, оперативно реагуючи на постійно змінювані кіберзагрози [8] (Рис. 1.2).



Рисунок 1.2 – Компоненти Security Operations Center

1.1.3. Ролі та обов’язки спеціалістів SOC

SOC функціонує як централізована команда спеціалістів, які займаються моніторингом загроз, їхнім розслідуванням та реагуванням. У великих SOC зазвичай використовується трирівнева структура аналітиків, які обробляють сповіщення безпеки, що генеруються системами моніторингу та SIEM.

Аналітики першого рівня відповідають за моніторинг безпекових сповіщень у реальному часі, проводять їхню первинну обробку і приймають рішення щодо

необхідності ескалації на наступний рівень. Вони налаштовують правила фільтрації хибнопозитивних подій та визначають, які загрози потребують детальнішого розслідування аналітиками другого рівня [8].

Аналітики другого рівня здійснюють глибший аналіз сповіщень, які надходять від аналітиків першого рівня, та корелюють ці дані з іншою інформацією для виявлення потенційних інцидентів безпеки. Вони оцінюють можливий вплив інциденту на активи підприємства та розробляють відповідні заходи реагування [24].

Аналітики третього рівня є досвідченими експертами з кібербезпеки, які займаються проактивним пошуком загроз у мережі замість очікування на автоматично згенеровані сповіщення. Вони також підтримують аналітиків першого та другого рівнів у випадках критичних інцидентів і допомагають у розслідуванні складних атак [11].

Зображена на рисунку 1.3 структура дозволяє ефективно розподіляти навантаження між фахівцями, забезпечуючи швидке та професійне реагування на кіберзагрози.

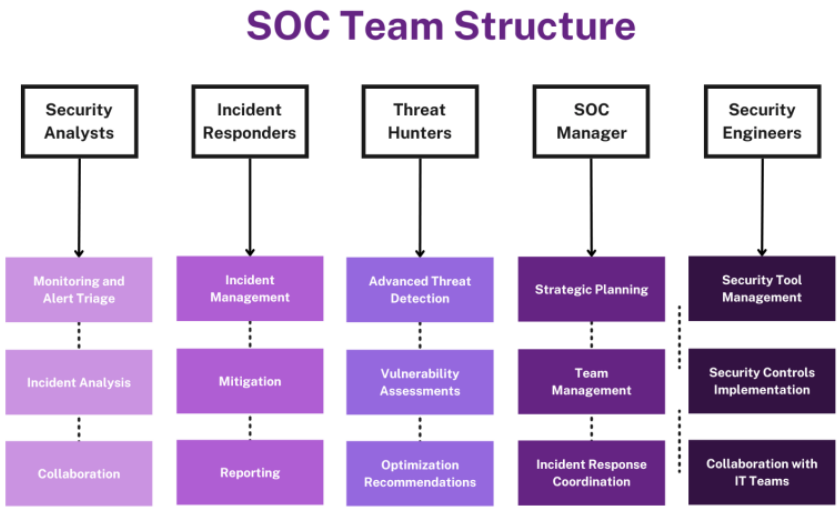


Рисунок 1.3 – Структура команди SOC

Окрім аналітиків безпеки, в SOC також працюють інші критично важливі фахівці, кожен із яких виконує важливі функції для забезпечення ефективної кібербезпеки організації.

Інженери безпеки відповідають за розгортання, налаштування та підтримку інструментів безпеки. Вони впроваджують заходи контролю безпеки та співпрацюють з IT командами для інтеграції систем.

Фахівці з реагування на інциденти зазвичай займаються управлінням інцидентами кібербезпеки. Їхня роль включає пом’якшення наслідків атак, створення звітності та впровадження стратегій відновлення після інцидентів.

Мисливці за загрозами – це спеціалісти, які займаються активним пошуком прихованих загроз у мережі, проводять оцінку вразливостей і розробляють рекомендації для оптимізації захисту [24].

Менеджер SOC керує командою SOC, займається стратегічним плануванням, управлінням персоналом і координацією реагування на інциденти.

У реальних умовах структура команди Операційного Центру Безпеки може змінюватися залежно від розміру організації, наявних ресурсів та специфіки загроз. У менших SOC або за умов обмеженого персоналу, аналітики різних рівнів часто виконують обов’язки, що виходять за межі їхніх стандартних ролей. Наприклад, аналітик третього рівня може не лише займатися проактивним пошуком загроз, але й брати на себе керівництво командою у разі потреби. Такий підхід дозволяє забезпечити гнучкість та ефективність роботи SOC, адаптуючи функції команди до поточних викликів і ресурсів організації.

1.1.4. Проблеми сучасного SOC

Сучасні центри операційної безпеки стикаються з численними проблемами, що ускладнюють їх ефективну роботу. Ці виклики можна розділити на три основні категорії: людський фактор, технологічні обмеження та проблеми процесів [11].

Проблеми людського фактора. Однією з головних проблем SOC є нестача кваліфікованих спеціалістів. Попит на досвідчених аналітиків зростає швидше, ніж ринок може їх підготувати, що призводить до перевантаження існуючого персоналу. Крім того, часто спостерігається недостатня підтримка з боку

керівництва, що ускладнює отримання необхідного фінансування та ресурсів. У багатьох організаціях SOC змушений конкурувати за бюджет із іншими IT-підрозділами, що негативно позначається на його розвитку.

Технологічні обмеження. Сучасний SOC використовує безліч різних інструментів для моніторингу, аналізу та реагування на загрози. Проте надмірна кількість розрізнених технологій без належної інтеграції створює так звані "прогалини у видимості". Це означає, що різні інструменти можуть неефективно обмінюватися інформацією, залишаючи "сліпі зони", які можуть бути використані зловмисниками. Ще однією проблемою є надмірне покладання на автоматизацію без достатньої кількості інтелектуального аналізу, що може призводити до великої кількості помилкових спрацьовувань і марної витрати ресурсів.

Проблеми процесів. Ефективна робота SOC залежить від добре налагоджених процесів. Однак, у багатьох організаціях ці процеси є неструктурованими або недостатньо розвиненими. Перевантаженість SOC великою кількістю тривог, відсутність ефективної класифікації та пріоритезації подій призводять до того, що критично важливі загрози можуть залишатися непоміченими. Також часто зустрічається проблема недостатньої автоматизації, через що час реагування на інциденти (MTTR) залишається занадто високим.

Типовими помилками при створенні SOC є недостатня підтримка керівництва та відсутність чітко визначених цілей, що ускладнює ефективне впровадження та подальшу роботу центру. Часто організації роблять надмірний акцент на впровадженні технологічних рішень, нехтуючи необхідністю підготовки персоналу, що призводить до низької ефективності розслідувань та реагування на інциденти. Додатковою проблемою є порушення логіки побудови SOC, коли намагаються одразу реалізувати складні механізми без належного опрацювання базових принципів безпеки, що ускладнює керування інцидентами та обробку загроз (Рис. 1.4).

Окрім цього, часто спостерігається відсутність фокусу на критичних завданнях безпеки, коли фахівці SOC змушені виконувати другорядні або

невластиві їм функції, що негативно впливає на їхню продуктивність. Деякі організації впроваджують SOC виключно для формального дотримання вимог регуляторів, не зосереджуючись на реальному підвищенні рівня кібербезпеки, що робить такий підхід неефективним. Завершальним фактором, що призводить до невдач, є недостатнє фінансування повсякденної роботи SOC після його запуску, через що він не отримує необхідних ресурсів для розвитку та підтримки високої ефективності [11].

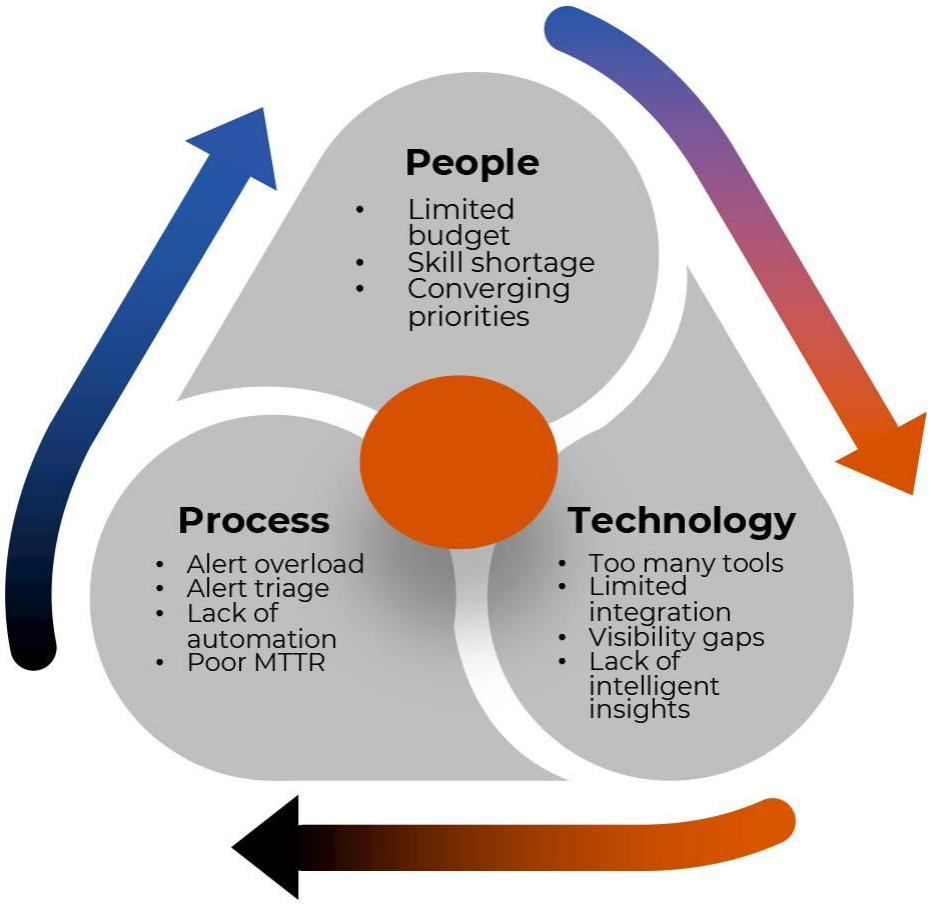


Рисунок 1.4 – Основні проблеми команди SOC

Незважаючи на всі ці виклики, ефективний SOC залишається головним елементом сучасної кібербезпеки.

Організації, які приділяють належну увагу процесам, людському фактору та технологічним аспектам, мають значно вищі шанси на успішне створення та підтримку SOC, здатного ефективно виявляти, аналізувати та нейтралізувати кіберзагрози.

1.2. Огляд технологій SIEM та XDR

1.2.1. Концепція SIEM

Security information and event management (SIEM) – це система управління інформацією та подіями безпеки, яка збирає, зберігає та аналізує журнали подій з різних джерел у мережі організації. Основна мета SIEM полягає у централізованому моніторингу, виявленні аномалій та кореляції подій для ідентифікації потенційних загроз. Вона забезпечує збереження логів для відповідності нормативним вимогам, аналізу інцидентів та підтримки розслідувань [13].

Сучасні системи SIEM працюють за принципом централізованого збору та аналізу даних із різних джерел: кінцевих пристроїв, серверів, мережевого обладнання, систем авторизації та аутентифікації, міжмережних екранів, антивірусів тощо. Для цього використовуються спеціальні агенти, які збирають, аналізують та передають інформацію в SIEM. Окрім традиційних джерел, деякі SIEM рішення здатні отримувати дані з хмарних сервісів та інших нестандартних платформ (Рис. 1.5).

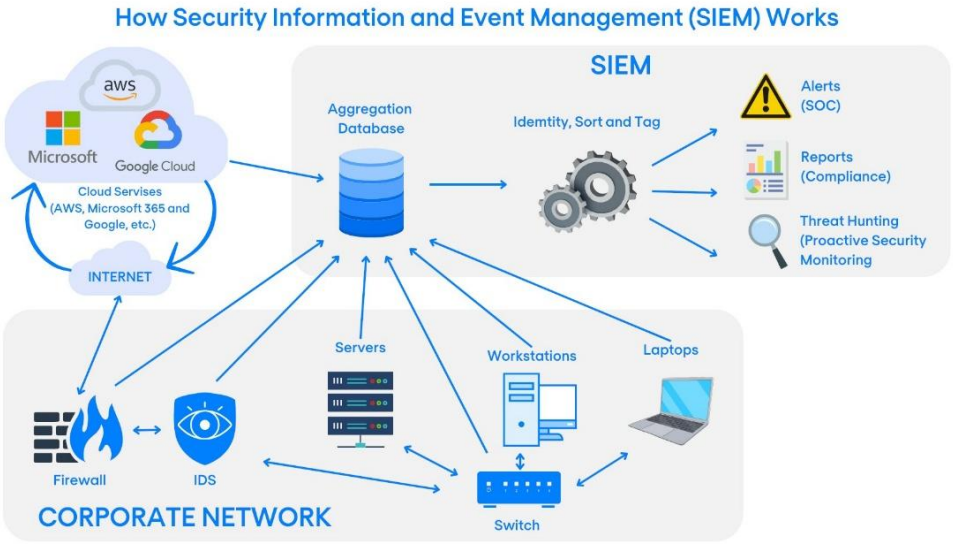


Рисунок 1.5 – Схема роботи SIEM

Однією з головних особливостей SIEM є здатність обробляти величезні обсяги інформації у режимі реального часу [12]. Це дозволяє швидко аналізувати

події безпеки, виявляти потенційні загрози та повідомляти аналітиків про підозрілі активності. Окремі події можуть здаватися незначними, але в сукупності вони можуть вказувати на загрозу. Саме тому SIEM використовує методи кореляції подій, а також технології штучного інтелекту та машинного навчання для виявлення складних атак [9].

Гнучкість SIEM систем дозволяє налаштовувати їх відповідно до специфіки кожної організації. Адміністратори можуть задавати правила виявлення загроз, встановлювати рівні критичності подій та визначати параметри реагування. Завдяки цьому SIEM глибоко інтегрується в загальну систему кібербезпеки підприємства, забезпечуючи ефективний моніторинг та координацію роботи інших захисних механізмів.

Однак розгортання SIEM вимагає значних ресурсів. Великий обсяг зібраних даних створює значне навантаження на мережеву інфраструктуру, а обробка логів потребує потужних обчислювальних ресурсів. Тому перед впровадженням SIEM важливо оцінити готовність IT-інфраструктури, щоб система працювала ефективно без критичних уповільнень та перебоїв.

Функціональні можливості SIEM охоплюють широкий спектр завдань, спрямованих на ефективний моніторинг, аналіз та реагування на інциденти інформаційної безпеки.

Одним із основних компонентів є *агрегація даних* – система збирає журнали подій із різноманітних джерел, включаючи мережеві пристрої, сервери, бази даних, додатки, системи безпеки та датчики. Зібрані дані консолідуються, що дозволяє виявляти критичні події та аналізувати загальну картину безпеки в організації [23].

Кореляція подій відіграє важливу роль у виявленні загроз, адже вона дозволяє знаходити взаємозв’язки між подіями, які на перший погляд можуть здаватися розрізненими. SIEM аналізує отриману інформацію, застосовуючи різні методи для об’єднання подій у значущі інциденти. Це дає можливість своєчасно виявляти складні атаки та швидше на них реагувати.

Завдяки *функції сповіщень*, система автоматично аналізує корельовані події та генерує тривожні повідомлення. Сповіщення можуть відображатися в інтерфейсі SIEM, надсилатися на електронну пошту або інтегруватися з іншими каналами комунікації, такими як SMS або месенджери.

Ще однією важливою особливістю є *візуалізація даних*, що дозволяє операторам швидко ідентифікувати відхилення від нормальної поведінки. Інформаційні панелі та інтерактивні графіки допомагають зрозуміти загальну картину подій у системі безпеки та приймати оперативні рішення (Рис. 1.6).

Система також має *високу сумісність*, що дозволяє інтегрувати її в наявні процеси управління інформаційною безпекою та аудиту. Використання спеціальних модулів автоматизації спрощує збір даних, формування звітності та адаптацію SIEM до вимог конкретного підприємства [23].

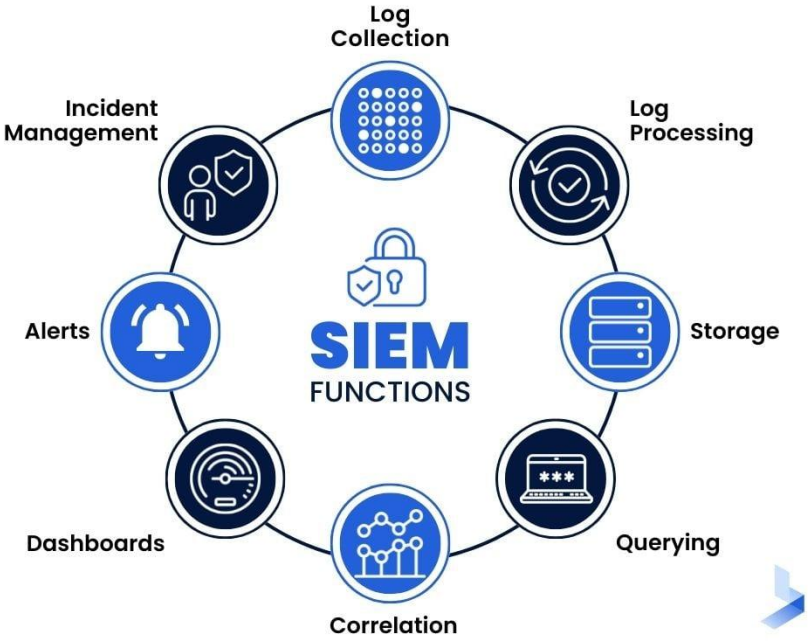


Рисунок 1.6 – Основні функції SIEM

Для довготривалого аналізу подій передбачено *зберігання даних* – всі журнали та записи подій архівуються, що дає змогу відстежувати історію подій та виконувати ретроспективний аналіз. Це особливо важливо для розслідувань інцидентів кібербезпеки, оскільки атаки можуть виявлятися лише через певний час після їхнього фактичного здійснення.

Крім автоматизованих механізмів, SIEM надає можливість *експертного аналізу*. Фахівці можуть виконувати пошук та кореляцію подій у великих масивах даних, що є важливим інструментом для глибокого розслідування інцидентів, проведення криміналістичних експертиз та виявлення прихованих загроз у мережі.

Загалом SIEM є невід'ємною частиною сучасного SOC, забезпечуючи централізований збір, аналіз та кореляцію подій безпеки. Завдяки своїм можливостям вона допомагає оперативно виявляти загрози, реагувати на інциденти та підвищувати загальний рівень кібербезпеки організації. Широке впровадження SIEM підтверджує її критичну роль у забезпеченні ефективного моніторингу та управління подіями безпеки.

1.2.2. Концепція XDR

Extended Detection and Response (XDR) – це сучасна технологія кібербезпеки, яка об'єднує та аналізує дані з різних рівнів захисту, таких як електронна пошта, кінцеві точки, сервери, хмарні робочі навантаження та мережі. Завдяки автоматизованій кореляції цих даних XDR забезпечує швидше виявлення загроз та покращує процеси розслідування й реагування на інциденти [10].

Однією з головних переваг використання XDR є точність виявлення загроз, в той час як SIEM не завжди здатен це зробити та іноді генерує хибно позитивні тривоги [10].

XDR розвивається на основі технології виявлення та реагування на кінцевих точках EDR, розширюючи її можливості шляхом інтеграції даних з інших джерел, таких як аналіз мережевого трафіку, системи захисту електронної пошти та хмарних сервісів [16]. Це дозволяє отримати більш повну картину потенційних атак і ефективніше на них реагувати.

Завдяки використанню штучного інтелекту та автоматизації, XDR оптимізує робочі процеси аналітиків безпеки, зменшуючи кількість хибно позитивних спрацьовувань та прискорюючи реагування на реальні загрози. Це сприяє

підвищенню ефективності роботи SOC та забезпечує більш надійний захист організації від складних кібератак.

Однією з вирішальних функцій XDR є автоматизація процесів аналізу та реагування на інциденти. Система здатна швидко обробляти великі обсяги даних, виявляти аномалії та корелювати події між різними доменами безпеки. Це дозволяє значно скоротити час на виявлення та реагування на загрози, зменшуючи навантаження на команди безпеки та мінімізуючи людський фактор [5][6].

XDR також інтегрується з існуючими інструментами та технологіями безпеки, такими як SIEM, включаючи рішення сторонніх виробників. Це розширює можливості системи, дозволяючи організаціям використовувати вже наявні інвестиції в безпеку та підвищувати ефективність операцій. Завдяки такій інтеграції XDR створює єдину екосистему, яка забезпечує централізований моніторинг та управління безпекою [14] (Рис. 1.7).

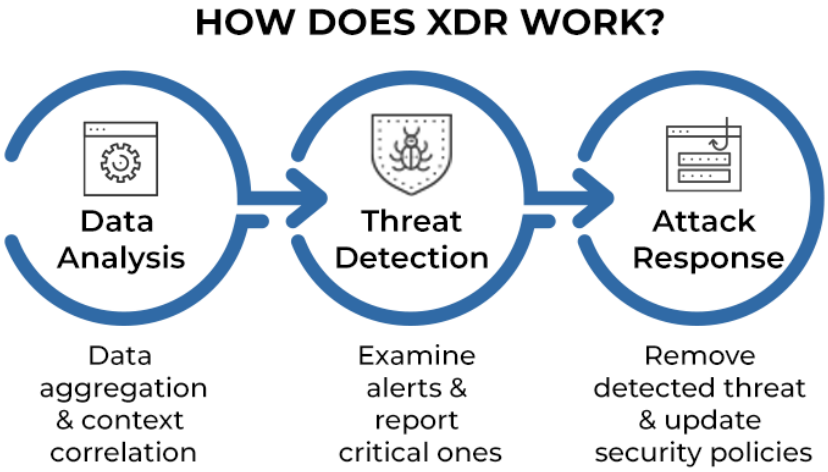


Рисунок 1.7 – Етапи роботи XDR системи

Загалом, XDR надає організаціям потужний інструмент для проактивного виявлення та нейтралізації загроз, розширюючи вже наявний арсенал засобів протидії загрозам та забезпечуючи надійний захист від сучасних атак.

1.2.3. Переваги інтеграції та порівняння SIEM з XDR

Інтеграція систем SIEM та XDR може створити потужний інструментарій для забезпечення кібербезпеки організації. SIEM збирає та аналізує журнали подій

з різних джерел, надаючи широке уявлення про активність у мережі та допомагаючи виявляти потенційні загрози. XDR, своєю чергою, об'єднує дані з кінцевих точок, мережевого трафіку та хмарних середовищ, забезпечуючи глибший аналіз та автоматизоване реагування на інциденти [3].

Одна з головних відмінностей між XDR і SIEM полягає в їхньому підході до загроз: XDR працює реактивно, зупиняючи атаку в момент її виконання, тоді як SIEM діє проактивно, акумулюючи величезні обсяги подій для подальшого аналізу та покращення архітектури безпеки. XDR зосереджений на фіксації подій, а SIEM на кореляції даних, оповіщеннях і глибокому аналізі. При цьому SIEM охоплює всю IT-інфраструктуру організації, а XDR переважно кінцеві точки, такі як комп'ютери та сервери [7].

XDR створений для швидкого виявлення та усунення загроз, тоді як SIEM, окрім цих функцій, забезпечує моніторинг відповідності, довготривале збереження логів і звітність. На відміну від SIEM, XDR не націлений на дотримання регуляторних вимог у сфері безпеки, а його головна цінність полягає у виявленні загроз та оперативному реагуванні. Саме тому організаціям, які потребують ширшої інтеграції логів або мають зобов'язання щодо їхнього зберігання, не обійтися без SIEM [2].

Попри те, що розробники XDR позиціонують його як наступний етап розвитку SIEM, насправді обидва підходи – це сучасні досягнення IT спільноти, які не конкурують між собою, а ефективно доповнюють один одного (Рис. 1.8).

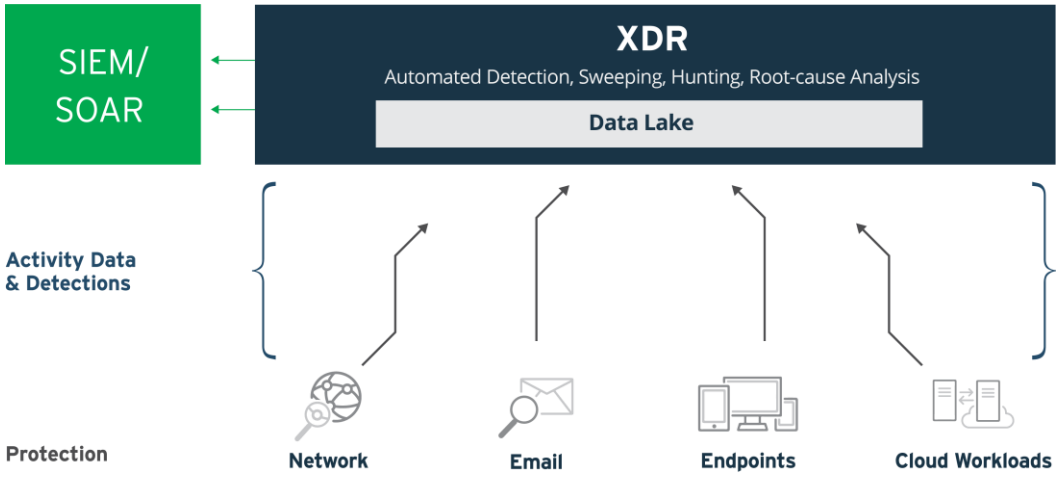


Рисунок 1.8 – Інтеграція SIEM та XDR

Традиційні методи виявлення загроз часто обмежені аналізом лише певного аспекту, наприклад, файлового рівня чи мережевого трафіку. XDR, навпаки, охоплює всі рівні одночасно, працюючи з даними всієї системи, включно з модулями та процесами, що в ній запущені. Завдяки цьому він забезпечує один із найвищих рівнів виявлення загроз і є незамінним інструментом для негайного реагування [14].

SIEM же виконує роль аналітичного центру, акумулюючи та систематизуючи дані про безпеку всієї інфраструктури. Його цінність полягає не в миттєвій реакції, а в довготривалому аналізі подій у часовому контексті. У результаті ці два підходи створюють потужну синергію, що забезпечує повноцінний контроль над кібербезпекою організації.

1.3. Огляд та порівняння Wazuh з іншими платформами для моніторингу інформаційної безпеки

1.3.1. Огляд рішення Wazuh та його архітектура

Wazuh – це платформа безпеки, яка поєднує функціональність XDR та SIEM, забезпечуючи комплексний захист як кінцевих пристроїв, так і хмарних робочих навантажень. Вона складається з універсального агента і трьох основних складових: сервера, індексатора та вебінтерфейсу для моніторингу та аналізу [22].

Це рішення є безкоштовним з відкритим вихідним кодом. Компоненти розповсюджуються під ліцензіями GNU GPLv2 та Apache License 2.0, що робить платформу доступною для вільного використання, модифікації та розповсюдження [21].

Як XDR-рішення, платформа дає змогу ефективно виявляти, аналізувати та реагувати на загрози на різних рівнях IT-інфраструктури. Вона збирає телеметричні дані з комп’ютерів, мережевого обладнання, хмарних сервісів, сторонніх API та інших джерел, об’єднуючи все в єдину систему спостереження.

Стосовно SIEM функціональності Wazuh є потужним інструментом для централізованого збору та обробки подій у реальному часі. Завдяки цьому можна

оперативно виявляти потенційні загрози та забезпечувати відповідність нормативним стандартам, охоплюючи дані з найрізноманітніших джерел: від пристроїв і додатків до хмарних середовищ.

Платформу активно використовують як компанії, так і окремі спеціалісти з кібербезпеки для захисту даних. Вона дозволяє не лише реагувати на загрози, а й запобігати їм. Крім того, її застосовують для дотримання міжнародних стандартів і рекомендацій — таких як PCI DSS, HIPAA або CIS Benchmarks.

Окремо варто згадати підтримку контейнеризованих середовищ. Платформа пропонує захист на етапі виконання в середовищах Docker та Kubernetes, інтегруючись із відповідними API. Агент може працювати прямо на хості Docker, забезпечуючи повний набір засобів для виявлення і реагування на загрози в контейнерах.

Отже, говорячи про архітектуру рішення, Wazuh базується на агенті Wazuh, який розгортається на кінцевих точках, що підлягають моніторингу, та на трьох центральних компонентах: сервері Wazuh, індексаторі Wazuh та інформаційній панелі Wazuh [21]. Нижче, на рисунку 1.9 зображено узагальнену діаграму компонентів та потоків даних Wazuh.

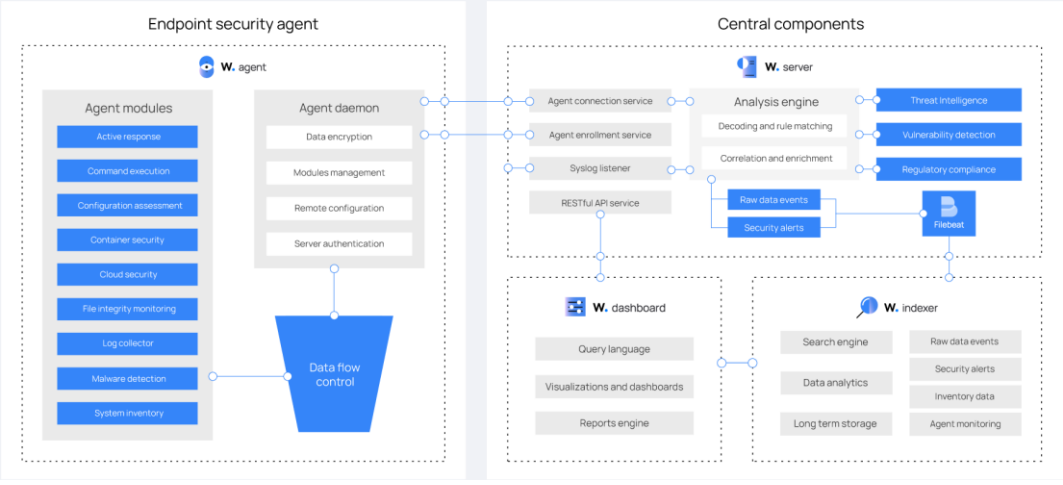


Рисунок 1.9 – Компоненти та потоки даних Wazuh

Wazuh Indexer – це високопродуктивна та масштабована повнотекстова пошуково-аналітична система. Цей центральний компонент Wazuh індексує та зберігає сповіщення, згенеровані сервером Wazuh, і надає можливості пошуку та

аналізу даних у режимі, близькому до реального часу. Індексатор Wazuh може бути сконфігурований як одновузловий або багатовузловий кластер, що забезпечує масштабованість і високу доступність.

Індексатор Wazuh зберігає дані у вигляді JSON-документів. Кожен документ співвідносить набір ключів, імен полів або властивостей з відповідними значеннями, які можуть бути рядками, числами, булевими виразами, датами, масивами значень, геолокацією або іншими типами даних.

Індекс – це колекція документів, які пов'язані між собою. Документи, що зберігаються в індексаторі Wazuh, розподілені по різних контейнерах, відомих як шарди. Розподіляючи документи між кількома шардами, а шарди між кількома вузлами, індексатор Wazuh може забезпечити надмірність. Це захищає вашу систему від апаратних збоїв і збільшує пропускну здатність запитів по мірі додавання вузлів до кластера. На рисунку 1.10 зображено діаграму будови кластера Wazuh індексатора, який було розгорнуто як багатовузловий.

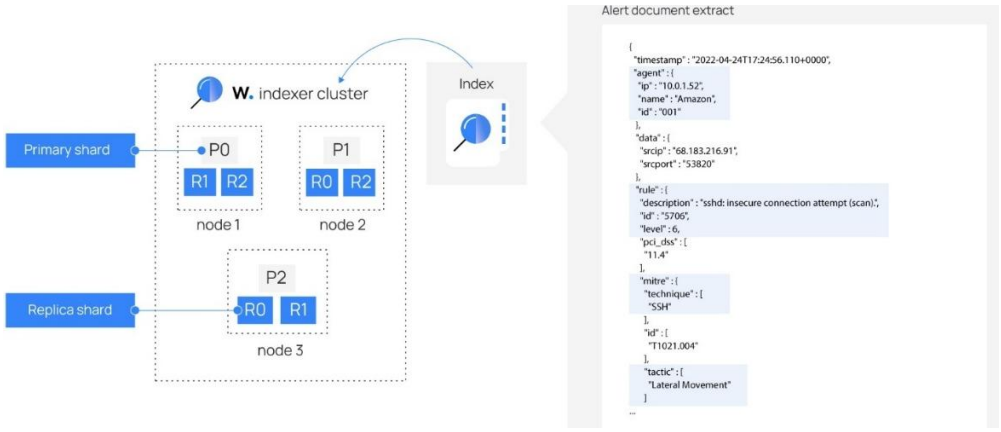


Рисунок 1.10 – Будова Wazuh Indexer, розгорнутого як багатовузловий кластер

Wazuh індексер підтримує взаємодію з кластером індексаторів за допомогою REST API Wazuh indexer, який пропонує велику гнучкість. Або наявний більш звичний варіант комунікації використовуючи Wazuh Dashboard, який буде розглянуто далі.

Wazuh Server аналізує дані, отримані від агентів, запускаючи сповіщення при виявленні загроз або аномалій. Він також використовується для віддаленого керування конфігурацією агентів і моніторингу їхнього стану.

На сервері Wazuh працює механізм аналізу, RESTful API Wazuh, служба реєстрації агентів, служба підключення агентів, демон кластера Wazuh і Filebeat. Сервер встановлений на операційній системі Linux і зазвичай працює на окремій фізичній машині, віртуальній машині, докер-контейнері або хмарному екземплярі.

Додатково, Wazuh Server підтримує інтеграцію з іншими корисними програмними застосунками, такими як ServiceNow, Jira, PagerDuty, а також платформи обміну повідомленнями, такі як Slack. Ці інтеграції зручні для оптимізації роботи SOC [21].

Нижче, на рисунку 1.11 показано архітектуру та компоненти Wazuh Server.

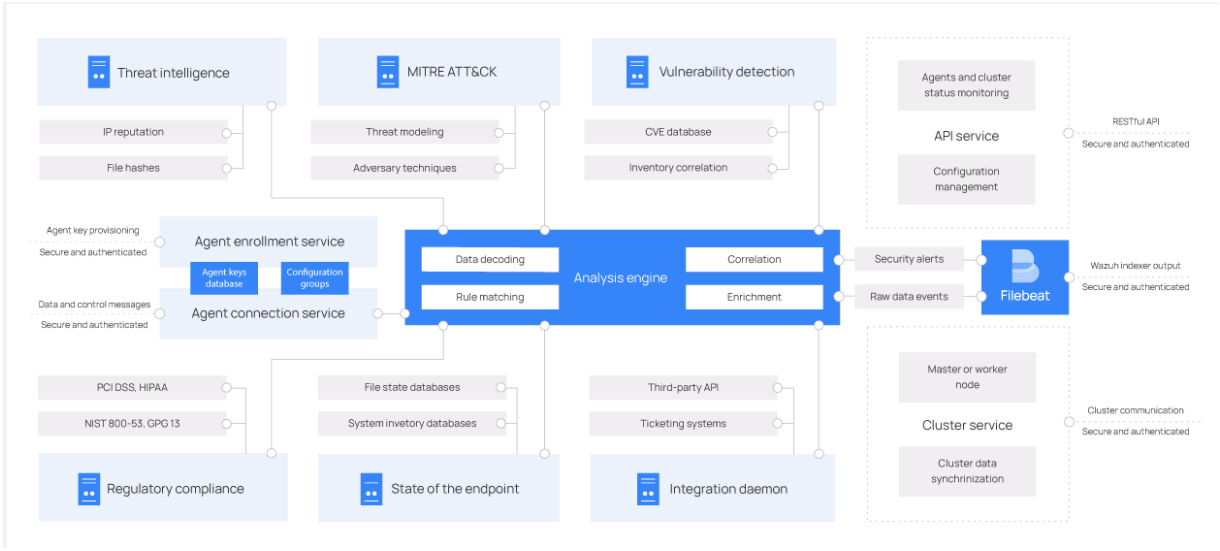


Рисунок 1.11 – Архітектура та компоненти Wazuh Server

Сервер Wazuh складається з декількох компонентів, які виконують різні функції, такі як реєстрація нових агентів, перевірка ідентичності кожного агента та шифрування зв’язку між агентом Wazuh та сервером Wazuh.

Agent enrollment service. Використовується для реєстрації нових агентів. Ця служба надає та розподіляє унікальні ключі автентифікації для кожного агента. Процес працює як мережева служба і підтримує автентифікацію за допомогою сертифікатів TLS/SSL або за допомогою фіксованого пароля.

Agent connection service. Ця служба отримує дані від агентів. Він використовує ключі, спільні для служби реєстрації, для перевірки ідентичності кожного агента і шифрування зв’язку між агентом Wazuh і сервером Wazuh. Крім

того, цей сервіс забезпечує централізоване управління конфігурацією, що дозволяє віддалено змінювати налаштування агентів [21].

Analysis engine. Це серверний компонент, який виконує аналіз даних. Він використовує декодери для визначення типу оброблюваної інформації (події Windows, журнали SSH, журнали веб-сервера тощо). Ці декодери також витягують відповідні елементи даних з повідомлень журналів, такі як IP адреса джерела, ідентифікатор події або ім'я користувача. Потім, використовуючи правила, analysis engine визначає конкретні шаблони в декодованих подіях, які можуть викликати сповіщення.

Wazuh RESTful API. Цей сервіс надає інтерфейс для взаємодії з інфраструктурою Wazuh. Він використовується для керування налаштуваннями конфігурації агентів і серверів, моніторингу стану інфраструктури та загального стану, керування та редагування декодерів і правил Wazuh, а також для запитів про стан кінцевих точок, за якими ведеться спостереження. Wazuh Dashboard також використовує його.

Wazuh cluster daemon. Цей сервіс використовується для горизонтального масштабування серверів Wazuh, розгортаючи їх як кластер. Така конфігурація в поєднанні з мережевим балансувальником навантаження забезпечує високу доступність і балансування навантаження. Демон кластера Wazuh – це те, що сервери Wazuh використовують для зв'язку один з одним та синхронізації.

Filebeat. використовується для надсилання подій та сповіщень до індексатора Wazuh. Він зчитує вихідні дані аналітичного механізму Wazuh і надсилає події в режимі реального часу. Він також забезпечує балансування навантаження при підключенні до багатовузлового кластера індексаторів Wazuh.

Wazuh Dashboard – це гнучкий та інтуїтивно зрозумілий вебінтерфейс для обробки, аналізу та візуалізації даних про події безпеки та сповіщення. Вона також використовується для управління та моніторингу самої платформи. Крім того, інтерфейс підтримує функціонал керування доступом на основі ролей (RBAC) і єдиного входу (SSO).

Wazuh Dashboard полегшує користувачам навігацію серед різноманітних типів даних, які збирає агент Wazuh, а також сповіщень, що генеруються сервером. Користувачі мають можливість генерувати звіти та налаштовувати власні інформаційні панелі з різноманітними графіками та даними [21].

Зокрема, платформа пропонує готові дашборди для контролю відповідності вимогам нормативів, таких як PCI DSS, GDPR, HIPAA та NIST 800-53. Крім цього, доступний зручний інтерфейс для навігації по фреймворку MITRE ATT&CK та пов'язаних з ним сповіщеннях.

Wazuh Agent в свою чергу підтримує роботу на операційних системах Linux, Windows, macOS, Solaris та AIX. Його можна розгортати на ноутбуках, настільних комп'ютерах, серверах, хмарних сервісах, у контейнерах або на віртуальних машинах. Агент збирає різноманітні дані про стан системи та роботу застосунків, передаючи їх на сервер Wazuh через зашифрований та автентифікований канал.

Архітектура агента побудована за модульним принципом. Кожен компонент відповідає за виконання окремих завдань – таких як моніторинг файлової системи, зчитування журналів, збір інвентаризаційної інформації, перевірка конфігурацій системи та пошук шкідливого програмного забезпечення. Користувачі можуть керувати модулями агента за допомогою конфігураційних параметрів, адаптуючи його під власні потреби (Рис. 1.12).

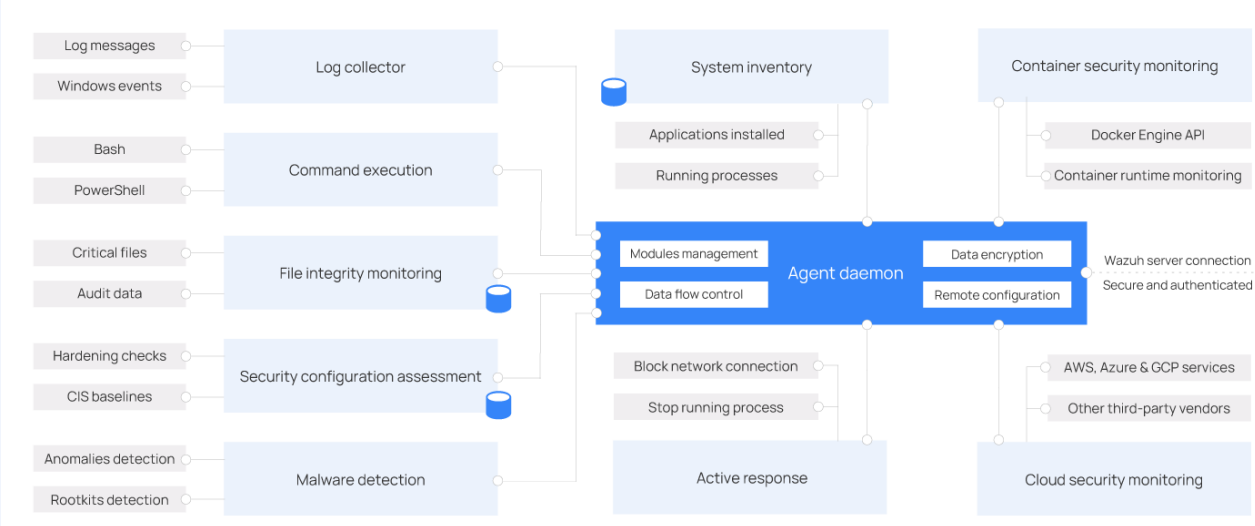


Рисунок 1.12 - Архітектура та компоненти Wazuh Agent

Wazuh Agent обмінюється даними з сервером для передачі зібраної інформації та подій, пов'язаних із безпекою. Окрім цього, агент надсилає службові дані, зокрема інформацію про свою конфігурацію та поточний стан. Після встановлення з'єднання його можна віддалено оновлювати, моніторити й налаштовувати через Wazuh Server використовуючи Dashboard.

Комунікація між агентом і сервером відбувається через захищений канал TCP або UDP із підтримкою шифрування та стиснення даних у реальному часі. Також реалізовано механізми контролю потоку, які запобігають перевантаженню каналу: події ставляться в чергу за потреби, що дозволяє зберігати пропускну здатність мережі [21].

Перед першим підключенням до сервера агент необхідно зареєструвати. Під час цього процесу йому надається унікальний ключ, який використовується для автентифікації та шифрування даних.

Як вже було згадано раніше, Wazuh має можливість розгортання Wazuh Server та Wazuh Indexer кластерами для забезпечення балансування навантаження та високої доступності. Нижче, на рисунку 1.13 продемонстровано кластерну архітектуру розгортання Wazuh.

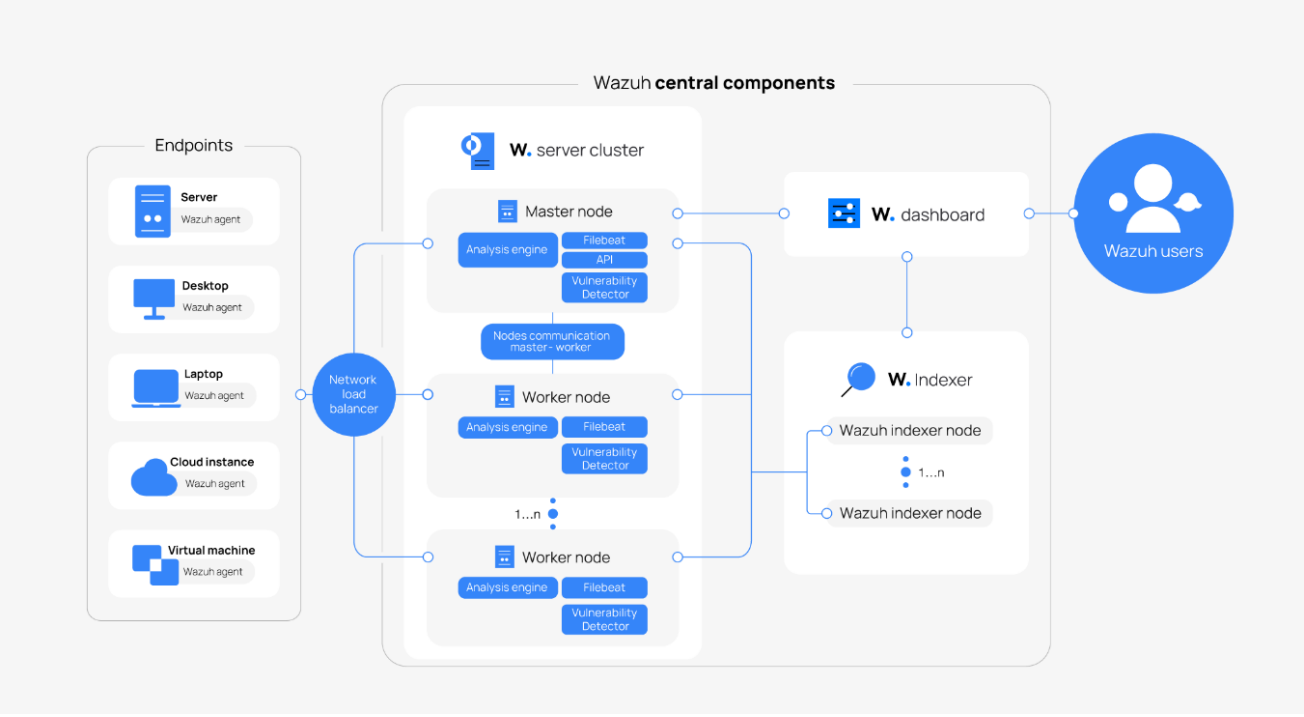


Рисунок 1.13 – Кластерна архітектура Wazuh

1.3.2. Основні функції та можливості Wazuh

Після ознайомлення з архітектурою та принципами роботи Wazuh, доцільно розглянути її функціональні можливості більш детально. Платформа призначена для забезпечення комплексного підходу до інформаційної безпеки, поєднуючи в собі інструменти розширеного виявлення загроз XDR та управління подіями безпеки SIEM [22]. Wazuh реалізує широкий спектр функцій, які дозволяють виявляти, аналізувати та оперативно реагувати на інциденти безпеки в різних середовищах – від фізичних серверів до хмарних інфраструктур (Рис. 1.14). До основних функцій та можливостей Wazuh належать:

Endpoint security	Threat intelligence	Security operations	Cloud security
Configuration assessment	Threat hunting	Incident response	Container security
Malware detection	Log data analysis	Regulatory compliance	Posture management
File integrity monitoring	Vulnerability detection	IT hygiene	Workload protection

Рисунок 1.14 – Основні функції та можливості Wazuh

Оцінка конфігурації. Wazuh відстежує налаштування конфігурації системи та додатків, щоб переконатися, що вони відповідають політикам безпеки, та стандартам. Агенти Wazuh виконують періодичне сканування, щоб виявити неправильні конфігурації або прогалини в безпеці кінцевих точок, які можуть бути використані зловмисниками. Крім того, Wazuh підтримує налаштування цих перевірок конфігурації таким чином, щоб вони відповідали потребам організації. Сповіщення про безпеку містять рекомендації щодо покращення конфігурації, посилання та зіставлення з нормативними вимогами [21].

Виявлення шкідливого програмного забезпечення. Wazuh може виявляти зловмисні дії та індикатори компрометації, які виникають на кінцевих точках в

результаті зараження шкідливим програмним забезпеченням або кібератаки. Стандартний набір правил Wazuh і такі функції, як оцінка конфігурації безпеки SCA, Rootcheck і моніторинг цілісності файлів FIM, допомагають виявляти зловмисні дії та аномалії.

Моніторинг цілісності файлів. Wazuh має можливість стежить за файловою системою, виявляючи зміни у вмісті, дозволах, правах власності та атрибутах файлів, які потрібно відстежувати. Крім того, він ідентифікує користувачів і програми, які використовуються для створення або зміни файлів. Також, функція моніторингу цілісності файлів Wazuh може бути використана у поєднанні з аналітикою загроз для виявлення загроз або скомпрометованих кінцевих точок. Крім того, FIM допомагає відповідати декільком регуляторним стандартам, таким як PCI DSS, NIST та іншим.

Полювання на загрози. Wazuh пропонує комплексну видимість кінцевих точок та інфраструктури, що контролюються. Він забезпечує збереження журналів, індексування та можливості запитів, які допомагають розслідувати загрози, що могли обійти початкові засоби контролю безпеки. Правила виявлення загроз зіставляються з фреймворком MITRE ATT&CK, щоб допомогти в розслідуванні і пошуку тактик, методів і процедур, які зазвичай використовуються зловмисниками. Wazuh також інтегрується зі сторонніми каналами розвідки загроз і платформами для покращеного полювання на загрози.

Аналіз даних журналів. Агенти Wazuh збирають журнали операційної системи та додатків та пересилають їх на сервер Wazuh для аналізу та зберігання на основі правил. Правила Wazuh виявляють помилки додатків або системи, неправильні конфігурації, зловмисні дії, порушення політик та інші проблеми з безпекою та роботою.

Виявлення вразливостей. Агенти Wazuh збирають дані інвентаризації програмного забезпечення та надсилають цю інформацію на сервер Wazuh. Зібрані дані інвентаризації потім співставляються з постійно оновлюваними базами даних CVE, щоб виявити відоме вразливе програмне забезпечення.

Автоматизоване виявлення вразливостей допомагає знайти недоліки у критично важливих активах і вжити заходів до того, як зловмисники використають їх у зловмисних цілях.

Реагування на інциденти. Wazuh надає нестандартні активні реакції для виконання різних контрзаходів проти поточних загроз. Ці реакції запускаються при досягненні певних критеріїв, вони включають такі дії, як блокування мережевого доступу до кінцевої точки з боку джерела загрози та інші. Крім того, Wazuh можна використовувати для віддаленого запуску команд або системних запитів, виявлення індикаторів компрометації та виконання завдань з реагування на інциденти.

Відповідність нормативним вимогам. Wazuh надає деякі з необхідних засобів контролю безпеки, щоб відповідати галузевим стандартам і нормам. Деякі з цих засобів контролю безпеки включають моніторинг цілісності файлів, оцінку конфігурації безпеки, виявлення вразливостей, інвентаризацію системи та інші. Ці можливості в поєднанні з масштабованістю та багатоплатформною підтримкою допомагають організаціям відповідати технічним вимогам. Wazuh надає звіти та інформаційні панелі для таких нормативних документів, як PCI DSS, NIST, TSC та HIPAA.

IT-гігієна. Wazuh створює актуальну системну інвентаризацію всіх кінцевих точок, за якими здійснюється моніторинг. Ця інвентаризація містить такі дані, як встановлені програми, запущені процеси, відкриті порти, інформацію про апаратне забезпечення та операційну систему тощо. Збір цієї інформації допомагає організаціям оптимізувати видимість активів і підтримувати належну IT-гігієну. Кілька інших можливостей Wazuh, таких як виявлення вразливостей, оцінка конфігурації безпеки та виявлення шкідливого програмного забезпечення, допомагають захистити кінцеві точки, що контролюються, та поліпшити IT-гігієну.

Безпека контейнерів. Wazuh забезпечує безпеку хостів і контейнерів Docker, відстежуючи їхню поведінку і виявляючи загрози, вразливості та аномалії.

Агент Wazuh має вбудовану інтеграцію з ядром Docker, що дозволяє користувачам відстежувати образи, томи, мережеві налаштування та запущені контейнери. Wazuh безперервно збирає та аналізує детальну інформацію про виконання. Наприклад, сповіщення про контейнери, що працюють у привілейованому режимі, вразливі програми, оболонку, запущену в контейнері, зміни в постійних томах або образах та інші можливі загрози.

Таким чином, Wazuh є потужним інструментом з відкритим вихідним кодом, який поєднує в собі функціональність рішень XDR та SIEM. Його вбудовані можливості забезпечують широкий спектр функцій – від збору та аналізу журналів до виявлення вразливостей і моніторингу відповідності нормативним вимогам.

Завдяки гнучкій архітектурі та модульному підходу, Wazuh ефективно покриває більшість базових потреб SOC та може використовуватися як основа для побудови масштабованої системи моніторингу кіберзагроз у різних типах IT-інфраструктур.

1.3.3. Порівняльний аналіз Wazuh з конкурентами

У сучасних умовах швидкого розвитку кіберзагроз організації все частіше звертаються до рішень класу SIEM та XDR, які забезпечують централізований моніторинг подій безпеки, виявлення інцидентів, відповідність нормативним вимогам і автоматизацію реагування. На ринку представлено доволі широкий спектр таких рішень, як комерційних, так і з відкритим вихідним кодом. У зв’язку з цим виникає потреба у порівняльному аналізі найбільш популярних платформ, що дозволить обґрунтовано обрати інструмент, який найкраще відповідатиме потребам конкретної IT-інфраструктури.

У попередніх розділах було розглянуто функціонал платформи Wazuh, яка позиціонується як повноцінне open-source рішення для розгортання SIEM/XDR-системи. Вона поєднує в собі можливості збору, обробки та аналізу телеметричних даних, виявлення загроз, моніторингу відповідності стандартам

безпеки, а також управління агентами у різноманітних середовищах, від фізичних серверів до хмарних платформ і контейнеризованих додатків.

Завдяки відкритій архітектурі, високій гнучкості конфігурації та можливості масштабування, Wazuh знаходить широке застосування як у малих компаніях, так і у великих організаціях з високими вимогами до безпеки. Проте для об'єктивного розуміння його місця серед інших рішень, доцільно провести зіставлення основних характеристик Wazuh із функціоналом конкурентів. Це дозволить краще оцінити її сильні та слабкі сторони, а також виявити потенційні сценарії використання, в яких вона буде найбільш ефективною.

На ринку рішень інформаційної безпеки, орієнтованих на функціонал SIEM та XDR, можна виокремити кілька популярних інструментів, які найчастіше розглядаються як альтернативи Wazuh. Серед них:

Splunk. Це одна з найпоширеніших та найпотужніших комерційних платформ для збору, обробки та аналітики даних. Ця платформа має велику кількість інструментів для збору та кореляції журналів, виявлення загроз і їх розслідування, а також для створення аналітичних дашбордів. Splunk пропонує потужні функції машинного навчання для автоматичного виявлення аномалій і прогнозування загроз. Хоча Splunk є комерційним продуктом, який вимагає значних фінансових витрат, він має широкий спектр можливостей для великих організацій, що потребують високої продуктивності та масштабованості. Однак висока вартість і складність налаштування часто стають перешкодою для малих та середніх підприємств.

Elastic Security, який є частиною Elastic Stack, є потужним інструментом для збору та аналізу даних, орієнтованим на інтеграцію з іншими компонентами екосистеми Elastic. Це рішення підтримує функції SIEM, забезпечує виявлення загроз та моніторинг інцидентів в реальному часі. Elastic Security надає вбудовані дашборди для аналізу подій, а також інструменти для виконання пошуку по великих обсягах даних, що робить його корисним для середніх і великих організацій. Однак, розширені функції для XDR доступні лише в платній версії,

що обмежує доступ до всього спектра можливостей при використанні в безкоштовній версії. Elastic Security – це рішення з відкритим кодом, однак для повноцінного використання необхідно підписуватись на платну версію [1].

AlienVault OSSIM – це open-source платформа для управління інформацією та подіями безпеки, що також включає функції виявлення загроз і керування інцидентами. OSSIM є частиною більш широкого рішення від AT&T Cybersecurity та пропонує інтеграцію з численними відкритими джерелами інформації про загрози. Основною перевагою OSSIM є його здатність інтегрувати різноманітні відкриті інструменти безпеки в єдину платформу, що дозволяє знижувати витрати на моніторинг та виявлення загроз. Однак у порівнянні з іншими рішеннями, OSSIM має обмежену масштабованість та не так добре підтримує новітні методи виявлення загроз, такі як аналіз поведінки та машинне навчання.

OSSEC є старим, але все ще популярним open-source рішенням для виявлення вторгнень, яке є основою для Wazuh. Воно спеціалізується на моніторингу цілісності файлів, зборі журналів та виявленні аномалій на рівні хосту. Однак OSSEC має обмежену функціональність в порівнянні з більш сучасними рішеннями SIEM/XDR, оскільки не підтримує масштабованість на рівні кластеризації та має менш зручний веб-інтерфейс. Незважаючи на це, OSSEC може бути корисним для малих підприємств, де необхідна базова система для моніторингу безпеки без великих витрат на інфраструктуру. Wazuh розвивається на основі OSSEC, додаючи підтримку більш складних функцій та інтеграцій.

Graylog – це платформа для збору, зберігання та аналізу логів, яка підтримує функціонал SIEM. Вона пропонує зручний інтерфейс для управління великими обсягами даних і підтримує кореляцію подій, що дозволяє виявляти та аналізувати кіберзагрози. Graylog підтримує велику кількість вхідних джерел даних і забезпечує хорошу інтеграцію з іншими інструментами для моніторингу безпеки. Однак вона не має такої ж глибокої підтримки для XDR, як більш спеціалізовані платформи, і її функціональність у порівнянні з Wazuh є

обмеженою в плані виявлення загроз та моніторингу в реальному часі. Graylog може бути хорошим варіантом для організацій, що потребують просту, масштабовану систему для збору та аналізу логів, але з обмеженими вимогами до виявлення загроз.

Нижче представлена таблиця 1.1 для чіткого уявлення про головні відмінності між Wazuh та його головними конкурентами в контексті функціональних можливостей, ліцензування, масштабованості та вартості.

Таблиця 1.1

Порівняння Wazuh з головними конкурентами

Критерій	Wazuh	Splunk	Elastic Security	OSSEC
Тип ліцензії	Безкоштовна	Платна	Частково безкоштовна	Безкоштовна
Функціональність SIEM	Вбудована	Вбудована	Вбудована	Обмежена
Функціональність XDR	Вбудована	Доступна через додатки	Доступна через інтеграції	Не підтримується
Масштабованість	Висока, підтримує кластеризацію	Висока, підтримує кластеризацію	Висока, підтримує кластеризацію	Обмежена; підходить для малих систем
Моніторинг агентів	Повний; безкоштовний	Повний; платний	Повний; безкоштовний	Повний; безкоштовний
Безагентний моніторинг	Повний; безкоштовний	Повний; платний	Повний; безкоштовний	Не підтримується
Інтеграція з MITRE ATT&CK	Повна; безкоштовна	Повна; платна	Повна; безкоштовна	Обмежена; базова інтеграція
Можливості для машинного навчання	Через інтеграції	Вбудовані; платні	Через інтеграції	Не підтримується
Інтерфейс користувача	Web-інтерфейс; безкоштовний	Web-інтерфейс; платний	Web-інтерфейс; безкоштовний	CLI; базовий інтерфейс
Оновлення агентів віддалено	Повне; безкоштовне	Часткове; платне	Часткове; безкоштовне	Не підтримується
Легкість налаштування	Середня; вимагає технічних знань	Висока; потребує навчання	Середня; вимагає технічних знань	Низька; просте налаштування

Порівняння Wazuh, Splunk, Elastic Security та OSSEC показує, що кожен з цих інструментів має свої переваги та обмеження. Wazuh виділяється як потужна, безкоштовна платформа з відкритим кодом, яка забезпечує всі основні функції SIEM та XDR, а також має хорошу масштабованість і підтримку безагентного моніторингу.

Splunk є комерційним продуктом з дуже високою масштабованістю та потужними можливостями аналізу даних, але його вартість може бути значною, а деякі функції доступні лише в платних версіях.

Elastic Security, як частина Elastic Stack, пропонує чудові можливості для виявлення загроз і аналізу, а також підтримує відкритий код, що робить його привабливим для користувачів, які хочуть мати контроль над своїми даними.

OSSEC, з іншого боку, є простим і ефективним рішенням для малих середовищ, де важлива базова безпека хостів, але він має обмежену масштабованість і функціональність порівняно з іншими інструментами.

Загалом, вибір між цими платформами залежить від конкретних потреб організації, якщо потрібна безкоштовна та масштабована система з розвиненими можливостями, Wazuh буде хорошим вибором. Для великих підприємств з вимогами до високої масштабованості та глибокого аналізу даних більше підходить Splunk, хоча й Wazuh теж добре підходить для цього. Для тих, хто хоче інтеграцію з Elastic Stack, Elastic Security пропонує хорошу альтернативу. В той час, OSSEC залишається простим і ефективним для невеликих середовищ з обмеженими ресурсами.

1.3.4. Ключові переваги Wazuh у контексті SOC

Wazuh є підходящим інструментом для операційних центрів безпеки, пропонуючи низку ключових переваг, які сприяють підвищенню ефективності та оперативності в управлінні безпекою. Як платформа з відкритим кодом, Wazuh є економічно ефективним рішенням, що дозволяє організаціям знижувати витрати на ліцензії та підтримку.

Рішення Wazuh забезпечує інтеграцію з різними платформами аналізу загроз, що покращує можливості виявлення за допомогою інформації про відомі загрози, такі як індикатори компрометації, шкідливі IP адреси, домени та URL адреси. Ця інформація допомагає командам SOC оцінювати характер та серйозність попереджень, що підвищує ефективність реагування на потенційні загрози.

Модуль Active Response автоматизує реагування на виявлені інциденти, зводячи шкоду до мінімуму. Незалежно від того, чи використовуються вбудовані дії реагування, чи налаштовуються власні, Wazuh забезпечує швидке стримування загроз відповідно до плану реагування на інциденти, що підвищує оперативність SOC.

Wazuh спрощує процес дотримання нормативних вимог, таких як GDPR, HIPAA, PCI DSS та інших, автоматизуючи перевірки на відповідність нормам, створюючи детальні звіти та постійно контролюючи інфраструктуру на відповідність регуляторним стандартам. Це допомагає SOC забезпечувати відповідність стандартам безпеки та конфіденційності, знижуючи ризики юридичних та фінансових наслідків.

Крім того, Wazuh надає операційним центрам безпеки потужні можливості для інвентаризації систем, що сприяють покращенню управління безпекою. Завдяки модулю Syscollector, Wazuh збирає детальну інформацію про апаратні та програмні ресурси кінцевих точок, включаючи дані про апаратне забезпечення, операційні системи, встановлені пакети, мережеві інтерфейси, порти та запущені процеси. Ця централізована інвентаризація дозволяє SOC мати повне уявлення про інфраструктуру, що полегшує виявлення незвичних або потенційно небезпечних змін, а також сприяє дотриманню нормативних вимог.

Завдяки цим перевагам, Wazuh є ефективним інструментом для SOC, забезпечуючи комплексний захист та ефективне управління безпекою в організації, та надаючи SOC команді можливість детально налаштувати Wazuh відповідно до своїх вимог через масштабованість та гнучкість рішення.

Висновки до першого розділу

У першому розділі було здійснено теоретичний огляд компонентів необхідних для побудови SIEM та XDR підсистеми в межах функціонування Security Operations Center. Було розглянуто основні завдання та функції SOC, його архітектуру, а також ролі фахівців, які забезпечують безперервний моніторинг і реагування на кіберзагрози.

Особливу увагу приділено аналізу технологій SIEM та XDR, їх концептуальним засадам і взаємодії, що дозволяє підвищити ефективність виявлення інцидентів безпеки. Проведено порівняння системи Wazuh з іншими платформами моніторингу, в результаті якого було визначено її ключові переваги, зокрема відкриту архітектуру, розширюваність, широку функціональність та гнучкість у налаштуванні.

Таким чином, отримані результати засвідчили доцільність вибору платформи Wazuh як основи для впровадження інтегрованої SIEM та XDR підсистеми в контексті сучасного SOC.

РОЗДІЛ 2. ПРОЄКТУВАННЯ SIEM ТА XDR ПІДСИСТЕМИ ДЛЯ SOC НА БАЗІ WAZUH

2.1. Аналіз типових ризиків і визначення пріоритетних завдань Wazuh в контексті SOC

Операційні Центри безпеки регулярно стикаються з новими ризиками та необхідністю реагувати на них. Однак для ефективної роботи SOC важливо правильно визначити пріоритети серед численних потенційних ризиків та завдань. У зв'язку з цим, SOC повинні зосереджуватися на тих аспектах, які можуть мати найбільший вплив на безпеку організації та її інформаційні системи.

У цьому підрозділі буде зосереджено увагу на аналізі типових ризиків та завдань, з якими стикаються SOC. Окрему увагу буде приділено визначенню найбільш важливих та актуальних з них, які можна ефективно вирішити за допомогою SIEM та XDR рішення Wazuh. Метою є виявлення таких пріоритетних задач, де використання Wazuh може забезпечити значне покращення в моніторингу, виявленні загроз і управлінні інцидентами безпеки. Це дозволить не лише знижувати ризики, але й підвищувати загальний рівень ефективності роботи SOC.

Одним з найпоширеніших та типових ризиків є *зараження шкідливим програмним забезпеченням*. Це може проявлятися у вигляді несанкціонованих змін у важливих системних директоріях, таких як папки користувача, також у вигляді виконання підозрілих або шкідливих файлів та команд на кінцевих точках, що призводить до компрометації системи. Крім того, зміни у реєстрі також можуть вказувати на можливе зараження або втручання в роботу системи.

Аномальна командна активність, зокрема застосування так званих методів "Living off the Land", також є серйозною загрозою. Це включає використання легітимних утиліт для шкідливих цілей, наприклад, через інструменти такі як netcat, powershell або certutil. Запуск нетипових скриптів або команд надає зловмисникам можливість здійснювати атаки, не залучаючи підозри.

Ще одним пріоритетним ризиком є *аномальна мережева активність*, що може сигналізувати про спроби встановлення прихованих з'єднань, виходи на відомі шкідливі IP адреси або використання нетипових портів.

Не менш актуальною є загроза *несанкціонованого доступу*, зокрема спроби брутфорс-атак на сервіси аутентифікації, такі як SSH. Вчасне виявлення таких атак і автоматичне блокування джерел дозволить значно знизити ймовірність успішного проникнення в систему.

Ще одним ризиком є *експлуатація відомих вразливостей програмного забезпечення*, що базується на експлуатації незакритих CVE. Це загроза, коли зловмисники використовують відомі та загальнодоступні уразливості в системах для отримання доступу до конфіденційної інформації або для запуску атак. Причиною відповідних атак є несвоєчасне оновлення програмного забезпечення.

Недоліки конфігурації систем безпеки також можуть створювати серйозні уразливості. Наприклад, відсутність антивірусного захисту або використання застарілих сигнатур може дозволити зловмисникам проходити через систему без виявлення. Відключене логування важливих подій робить важким або навіть неможливим відстеження порушень безпеки. Регулярне сканування та оцінка безпекової конфігурації є невід'ємною частиною проактивного захисту.

Для чіткого розуміння ризиків та можливостей їх уникнення використовуючи рішення Wazuh, нижче представлена таблиця 2.1.

Таблиця 2.1

Механізми Wazuh для вирішення типових ризиків і завдань SOC

Тип ризику/завдання	Механізм Wazuh для вирішення/пом'якшення
Зараження шкідливим ПЗ – Malware	Виявлення додавання або зміни файлів за допомогою FIM модуля. Далі перевірка файлу на шкідливість за допомогою інтеграції з антивірусним рішенням VirusTotal. Та видалення відповідного шкідливого файлу за допомогою налаштованого модуля Active Response.
Зміни у важливих системних директоріях та директоріях користувача	Моніторинг змін у відповідних директоріях за допомогою відповідно налаштованого FIM модуля.

Продовження таблиці 2.1

Механізми Wazuh для вирішення типових ризиків і завдань SOC

Тип ризику/завдання	Механізм Wazuh для вирішення/пом'якшення
Використання потенційно шкідливих команд CLI	Налаштування виявлення потенційно шкідливих команд та скриптів через аналіз журналів подій для powershell, cmd, bash за допомогою log analysis.
Аномальна мережева активність	Загальне виявлення мережевих з'єднань на кінцевому пристрої, після чого аналіз на предмет з'єднання по потенційно небезпечним портами. Також додатково може проводитися аналіз на шкідливість IP адреси за допомогою сторонніх інтеграцій. На завершення коректно налаштований модуль Active Response для блокування шкідливого мережевого з'єднання.
Використання відомих вразливостей ПЗ	Регулярне проведення тестувань на вразливості за допомогою модуля Vulnerability Management, який надає можливість аналізу вразливостей з використанням БД CVE.
Недоліки конфігурації систем безпеки	Регулярне проведення тестувань на наявність вразливостей у конфігурації системи, за допомогою модуля Configuration Assessment.

Wazuh також забезпечує функціональність інвентаризації, яка дозволяє SOC здійснювати точний моніторинг усіх активів на пристрої. Завдяки цьому забезпечується повний огляд компонентів і служб, що дає змогу вчасно виявляти незареєстровані, незахищені або потенційно вразливі процеси. Така функціональність є надзвичайно важливою в контексті полювання на загрози.

Таким чином, за допомогою Wazuh SOC може не лише реагувати на вже виявлені події безпеки, а й здійснювати проактивний моніторинг, що значно підвищує ефективність роботи центру безпеки.

2.2. Огляд стандартних політик та правил виявлення загроз у Wazuh

Однією з переваг Wazuh є наявність вбудованих правил, які забезпечують широкий спектр можливостей для виявлення загроз та аномалій у системі. Стандартні правила входять до складу встановлення Wazuh і розташовуються в

каталозі /var/ossec/ruleset/rules/ на Wazuh Server. Вони охоплюють різноманітні події безпеки та джерела логів, забезпечуючи базову лінію для виявлення базових кіберзагроз [21].

Ці правила час від часу оновлюються командою Wazuh для врахування нових загроз і вдосконалення можливостей системи. Стандартні правила здатні виявляти певні атаки, уразливості або підозрілу активність.

Всі правила в Wazuh класифікуються за рівнями від 0 до 16. Ця класифікація дозволяє визначати ступінь серйозності кожного інциденту безпеки, що допомагає аналітикам SOC ефективно реагувати на події. Нижче, в таблиці 2.2 представлений список рівнів та їх опис.

Таблиця 2.2

Класифікацій правил Wazuh за рівнями

Рівень	Назва	Опис
0	Ігнорується	Події без значущості для безпеки, використовуються для уникнення хибнопозитивних спрацювань і не відображаються в дашборді подій безпеки.
2	Системне повідомлення низького пріоритету	Системні сповіщення або статусні повідомлення, які не мають відношення до безпеки та не відображаються в дашборді подій безпеки.
3	Успішні/Авторизовані дії	Включає успішні спроби входу, дозволені дії файрволів.
4	Системна помилка низького пріоритету	Помилки, пов’язані з некоректними конфігураціями або використанням непотрібних пристроїв/додатків, зазвичай не мають відношення до безпеки.
5	Помилки, спричинені користувачем	Випадки, як-от невірний пароль або відхилені дії, які самі по собі не мають значущості для безпеки.
6	Атака низької важливості	Вказує на хробака або вірус, який не впливає на систему (наприклад, Code Red для серверів Apache), а також часті події IDS та часті помилки.
7	Співпадіння з "поганими" словами	Містить слова на кшталт "bad", "error" тощо. Такі події здебільшого некласифіковані та можуть мати певне значення для безпеки.
8	Вперше зафіксований випадок	Включає події, що відбулися вперше, як-от перше спрацювання IDS або перший вхід користувача. Також включає дії, що мають значення для безпеки.
9	Помилка від недійсного джерела	Включає спроби входу як невідомий користувач або з недійсного джерела. Може мати значення для безпеки, особливо якщо повторюється. Також включає помилки, пов’язані з обліковим записом "admin" (root).

Продовження таблиці 2.2

Класифікацій правил Wazuh за рівнями

Рівень	Назва	Опис
10	Кілька помилок, спричинених користувачем	Включає декілька невірних паролів, кілька невдалих спроб входу тощо. Це може вказувати на атаку або просто сигналізувати, що користувач забув свої облікові дані.
11	Попередження про цілісність	Повідомлення щодо модифікації бінарних файлів або наявності руткітів (визначених Rootcheck). Це може вказувати на успішну атаку. Також включає події IDS, які ігноруються через високу кількість повторень.
12	Подія високої важливості	Включає повідомлення про помилки або попередження від системи, ядра тощо. Це може вказувати на атаку проти конкретного додатка.
13	Незвичайна помилка (висока важливість)	Збігається з поширеним шаблоном атаки в більшості випадків.
14	Високоважлива подія безпеки	Зазвичай спрацьовує при кореляції подій та вказує на атаку.
15	Серйозна атака	Немає вірогідності на хибнопозитивні спрацювання. Потрібна негайна увага.

Вбудовані правила та політики в Wazuh забезпечують базову функціональність для організацій, що впроваджують систему моніторингу безпеки.

Стандартний набір правил Wazuh включає понад 3,000 попередньо налаштованих правил та декодерів, що охоплюють широкий спектр подій безпеки та джерел журналів. Вони можуть бути використані "як є" для швидкого старту або адаптовані відповідно до специфічних потреб організації. Завдяки високій гнучкості та постійному оновленню стандартні політики Wazuh дозволяють виявляти та реагувати на базові кіберзагрози.

Окрім вбудованих правил, Wazuh також надає широкі можливості для створення та налаштування власних, кастомних правил. Це є одним із ключових аспектів гнучкої конфігурації системи, оскільки саме завдяки кастомним правилам можливо адаптувати Wazuh під специфічні потреби конкретного середовища. Такий підхід дозволяє ефективно реагувати на інциденти,

враховуючи особливості інфраструктури, бізнес-логіки та пріоритетів безпеки організації.

2.3. Проєктування механізмів виявлення загроз для рішення Wazuh

У цьому підрозділі буде розглянуто процес проєктування механізмів виявлення загроз для системи безпеки на базі Wazuh. Основна увага буде зосереджена на створенні ефективних правил та політик для виявлення типових загроз, які були визначені в підрозділі 2.1.

File Integrity Monitoring (FIM) є одним із провідних механізмів виявлення загроз в контексті рішення Wazuh, він дозволяє здійснювати контроль за змінами в критичних файлах та папках. FIM активно використовується для виявлення несанкціонованих змін в системах, що можуть свідчити про наявність шкідливих дій або спроби зловмисників змінити важливі системні файли. Також важливо налаштувати моніторинг не лише для базових системних файлів, але й для користувацьких папок, що містять чутливі або критичні дані [19]. Схема роботи модуля моніторингу цілісності файлів зображена на рисунку 2.1.

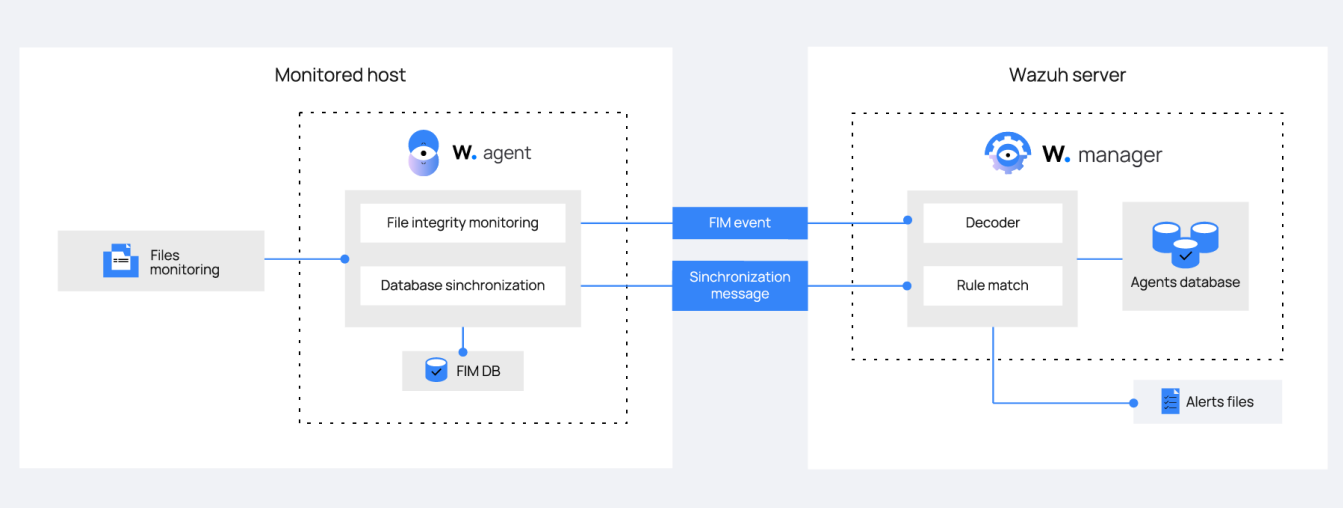


Рисунок 2.1 – Схема роботи FIM модуля

Для ефективного моніторингу у цьому контексті можна визначити кілька критичних директорій, які потрібно постійно контролювати на предмет змін. Має сенс проводити моніторинг наступних директорій:

Downloads – це директорія яка слугує для зберігання файлів, що були завантажені до системи з різноманітних додатків включаючи браузер. Вона є уразливою через можливе завантаження шкідливих програм чи інших небезпечних файлів, які можуть становити загрозу.

Desktop – директорія, яка часто використовується для зберігання робочих файлів користувачів. Вона може містити конфіденційну інформацію, що робить її цільовою для атак.

Documents – директорія для зберігання документів користувача. Оскільки вона містить важливі файли, її також потрібно ретельно моніторити.

Після визначення важливих директорій для моніторингу, наступним кроком буде налаштування правил для відстеження змін у цих директоріях. У Wazuh для цього використовується функціональність FIM, яка дозволяє стежити за певними подіями, пов'язаними з файлами: створення, видалення, зміна вмісту або метаданих. При цьому дуже важливо налаштувати саме ті параметри, які є критичними для безпеки, тобто моніторити будь-які зміни, які можуть свідчити про несанкціоновану діяльність.

У разі виявлення несанкціонованих змін, Wazuh згенерує нам відповідні алерти, які дозволять швидко реагувати на загрози. У разі виявлення несанкціонованих змін у зазначених папках, алерти будуть містити інформацію про тип зміни, дату та час події, а також відомості про користувача, який здійснив зміну [19].

Інтеграція з VirusTotal в рамках системи Wazuh є важливим елементом для підвищення рівня виявлення шкідливих програм і загроз на кінцевих точках. VirusTotal – це платформа, яка дозволяє перевіряти файли та URL-адреси на наявність шкідливого вмісту, використовуючи більше ніж 70 антивірусних двигунів і системи для виявлення загроз.

Інтеграція з цією платформою дозволить автоматично перевіряти хеші файлів, виявлених на кінцевих точках, проти бази даних VirusTotal, що дозволяє швидко ідентифікувати відомі загрози [20].

У рамках проєкту Wazuh інтеграція з VirusTotal буде реалізована через API цієї платформи. Після того, як Wazuh виявить файл на системі або отримає повідомлення про зміни в певних директоріях, наприклад, у директорії Downloads або Documents, він автоматично передасть хеш цього файлу до VirusTotal для перевірки. У разі якщо VirusTotal ідентифікує файл як шкідливий, Wazuh згенерує відповідне сповіщення, що дасть змогу швидко вжити необхідних заходів. Також, в підрозділі 2.4 буде розглянуто та спроєктовано механізм автоматичної відповіді на загрози виявлені модулем FIM, який інтегрований з VirusTotal.

Розширений збір логів з Microsoft Defender є важливою складовою для посилення можливостей Wazuh щодо виявлення шкідливих програм та небезпечних подій на кінцевих точках. Проєктуючи цю інтеграцію, потрібно врахувати кілька критичних аспектів, які дозволять ефективно використовувати можливості Defender у рамках системи Wazuh для моніторингу та реагування на загрози [25].

В першу чергу, необхідно налаштувати зв’язок між Wazuh та Microsoft Defender, що дозволить отримувати та обробляти дані з Defender, встановленого на кінцевих точках з ОС Windows. Важливо правильно налаштувати отримання логів та їх декодування, для цього буде використовуватися офіційна документація Wazuh. Дані повинні надходити в реальному часі, щоб забезпечити швидку реакцію на потенційно небезпечні події.

Далі, необхідно створити кастомні правила у Wazuh для обробки даних, що надходять з Defender. Це включає налаштування правил для виявлення подій, пов’язаних з виявленням шкідливого програмного забезпечення, таких як завантаження підозрілих файлів або спроби їх виконання.

Проєктування *виявлення шкідливих команд* на кінцевих пристроях, де встановлений Wazuh Agent передбачає реалізацію виявлення виконання підозрілих команд або потенційно небезчинних. Це такі команди як netcat, curl, certutil, whoami та інші, які часто використовуються в атаках типу lateral movement або для створення бекдорів у системах [4].

Головним кроком є налаштування збору логів від усіх кінцевих точок. Для Windows, це буде журнал подій Windows, зокрема модуль Sysmon, який забезпечує детальну інформацію про запущені процеси, включаючи команди командного рядка. Для Linux-систем, доцільно налаштувати збір логів через Auditd, що дозволить отримувати точні дані про виконувані системні команди та створювати їх хронологічну таблицю. В свою чергу для MacOS систем Wazuh надає можливість отримувати логи з системної утиліти log.

Особлива увага має бути приділена налаштуванню порогових значень для активування алертів, оскільки деякі з цих команд можуть використовуватись легітимно для адміністрування системи, і важливо уникати помилкових спрацьовувань. Це можна досягнути за допомогою фільтрації команд за контекстом, таких як частота виконання.

Проєктуючи *механізм виявлення аномальних мережевих з'єднань* важливо врахувати кілька головних аспектів, що дозволяють виявити потенційно небезпечні чи несанкціоновані мережеві активності.

Основним завданням є виявлення з'єднань, які не відповідають нормальному профілю трафіку або характеру роботи мережі, що може вказувати на шкідливу активність.

Для реалізації такого механізму доцільно розпочати з визначення переліку "не підозрілих" портів. До них належать ті порти, які зазвичай використовуються під час повсякденної експлуатації системи та не асоціюються з відомими загрозами. Це, зокрема, стандартні порти для типових сервісів – наприклад, 80 (HTTP), 443 (HTTPS), 22 (SSH), 53 (DNS), 25 (SMTP) та інші.

Формування такого переліку дозволяє відфільтрувати нормальну мережеву активність і зосередитися саме на виявленні нетипових або потенційно шкідливих з'єднань.

Після створення відповідного списку портів, необхідно створити відповідні правила які будуть надсилати повідомлення у випадку підключення до одного з портів якого немає в списку "не підозрілих" портів. У випадку спроби

підключення через підозрілий порт, система Wazuh повинна генерувати алерт, який повідомлятиме аналітиків SOC про потенційно небезпечну активність [15].

Цей механізм виявлення дозволить забезпечити базове виявлення на основі мережевої активності, що є важливим для виявлення спроб атак, які використовують порти для несанкціонованого доступу до систем.

2.4. Проєктування механізмів реагування на загрози для рішення Wazuh

У цьому підрозділі розглядається процес проєктування механізмів реагування на загрози для системи Wazuh. Враховуючи функціональність Wazuh, у цьому підрозділі також буде описано, як створити та налаштувати правила для реагування на виявлені загрози, включаючи автоматичні дії, аналіз подій і інтеграцію з іншими інструментами безпеки.

У попередньому підрозділі було спроєктовано виявлення загроз через інтеграцію з антивірусними рішеннями, такими як Microsoft Defender та VirusTotal, а також через моніторинг змін у критичних каталогах за допомогою File Integrity Monitoring модуля. Це дозволяє успішно почати проєктування механізмів реагування на вже потенційно виявлені загрози.

Наступним кроком у проєктуванні є налаштування *Active Response* модуля для автоматичного видалення цих файлів (Рис.2.2).

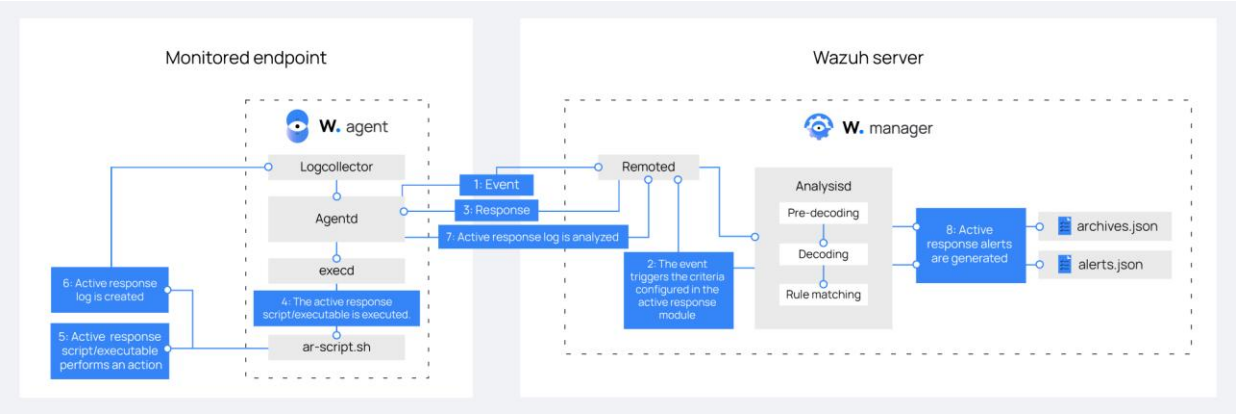


Рисунок 2.2 – Схема роботи модуля Active Response

Процес налаштування цього механізму вимагає створення спеціалізованих скриптів, які будуть виконуватись на кінцевих точках після того, як Wazuh

виявить загрозу. Скрипти повинні бути спроєктовані таким чином, щоб безпечно видаляти шкідливі файли без порушення цілісності інших важливих системних даних або додатків. Для цього необхідно, щоб скрипти здійснювали перевірку шляхів до файлів, які підлягають видаленню, а також мали функцію підтвердження, щоб уникнути випадкових помилок при автоматичному реагуванні [18].

Для автоматичного видалення файлів важливо створити правило в Wazuh, яке буде активувати цей скрипт тільки за певних умов. Видалення повинне відбуватися лише у разі виявлення файлів, що відносяться до категорії шкідливих, та після того, як файл пройшов перевірку через VirusTotal і отримав негативний результат. Важливо, щоб такі скрипти працювали не лише для індивідуальних файлів, а й для всіх компонентів потенційної загрози, включаючи скрипти, реєстраційні ключі та інші елементи, пов'язані з шкідливим ПЗ.

Також має бути додане журналювання усіх спроб видалення файлів, що дозволить аналітикам SOC проводити детальніший подальший аналіз [32].

Проєктування механізмів *реагування на мережеві загрози*, зокрема блокування IP адрес під час виявлення шкідливих мережевих з'єднань, є важливим етапом захисту кінцевих пристроїв від потенційної взаємодії з небезпечними IP адресами та іншими інтернет ресурсами. Це дозволить своєчасно розривати зв'язки з зовнішніми джерелами загроз і запобігати подальшому розповсюдженню шкідливої активності.

Проєктування механізму *активного реагування на мережеві загрози* передбачає розробку скриптів для автоматичного блокування явно шкідливих мережевих підключень, зокрема IP адрес. Основою для такого реагування є чітко визначені умови активації блокування. Доцільно реалізувати інтеграцію з зовнішніми репутаційними сервісами, такими як AbuseIPDB. Цей сервіс дозволяє здійснювати перевірку IP адреси та отримувати рівень її шкідливості на основі глобальної статистики зловмисної активності. Значення цього рівня буде використане як головний критерій для активації механізму Active Response у

Wazuh. Правила будуть налаштовані таким чином що би визначати 4 рівні загрози відповідно до отриманої інформації про IP адресу з сервісу AbuseIPDB. Блокування IP адреси буде відбуватися тільки при ідентифікації адреси як явно небезпечної та шкідливої.

Для коректної роботи механізму активного реагування на кінцевих вузлах необхідно мати окремі скрипти, адаптовані до типу операційної системи. Кожен скрипт буде налаштований таким чином, щоб здійснювати блокування IP адреси за допомогою відповідного фаєрвола – наприклад, iptables для Linux, pf для macOS або netsh advfirewall для Windows. Важливо враховувати, що на деяких системах фаєрвол за замовчуванням є вимкненим.

Active Response скрипти для блокування IP адрес також будуть використані для блокування IP адрес при виявленні *брутфорс-атак на SSH*. Wazuh відстежує кількість невдалих спроб входу та за допомогою скрипта зможе автоматично додати підозрілу IP до фаєрволу для негайного блокування.

2.5. Огляд та проєктування додаткових модулів у Wazuh для SOC

У рамках побудови ефективної SIEM та XDR підсистеми для SOC важливо не лише виявляти та реагувати на загрози, але й інтегрувати додаткові механізми для проактивного моніторингу, оцінки безпеки та управління вразливостями. З цією метою Wazuh надає ряд потужних інструментів, які можуть значно підвищити ефективність операцій SOC.

У цьому підрозділі буде розглянуто три ключові механізми, які будуть оглянуті та спроєктовані для подальшого використання SOC, а саме Wazuh System Inventory, Security Configuration Assessment та Vulnerability Management [21].

Для автоматичної інвентаризації системи на якій встановлено Wazuh Agent, використовується механізм *Wazuh System Inventory*, який дозволяє збирати детальну інформацію про кінцеву точку. Завдяки цьому механізму Операційний Центр Безпеки матиме можливість отримати повну картину про апаратні, програмні та мережеві активи кінцевого пристрою.

Проєктування цього механізму передбачає інтеграцію Wazuh з усіма кінцевими точками. Потрібно налаштувати модуль System Inventory для збору інформації про операційну систему, встановлені програми, активні служби, а також про поточні процеси з певними інтервалами.

Збір такої інформації дозволяє не лише підтримувати актуальну картину інфраструктури, а й дає можливість SOC виявляти заборонені в організації програмні засоби та застосунки.

Проєктування механізму *Security Configuration Assessment* в свою чергу передбачає впровадження перевірок відповідності стандартам безпеки на всіх кінцевих пристроях, на яких встановлений Wazuh Agent. Задача цього механізму полягає у проведенні автоматичних аудитів безпеки конфігурацій кінцевих точок, що дозволяє виявити порушення політик безпеки на рівні налаштувань операційних систем та програмного забезпечення.

Wazuh вже включає в себе набір стандартних конфігураційних перевірок, таких як CIS Benchmarks, які визначають налаштування системи для забезпечення її безпеки.

Один з найважливіших елементів у забезпеченні інформаційної безпеки є управління вразливостями. *Vulnerability Management* у Wazuh дозволяє здійснювати автоматичне виявлення вразливостей у програмному забезпеченні на основі даних з різних джерел, таких як CVE. Для цього Wazuh використовує модуль Vulnerability Detection, який здатний сканувати системи на наявність відомих вразливостей, що базуються на версіях ПЗ, встановлених на кінцевих точках.

Слід визначити, які джерела інформації про вразливості будуть використовуватися для сканування (CVE, NVD тощо), як часто відбуватиметься оновлення бази вразливостей, а також налаштувати правила для аналізу результатів сканування.

Для ефективного управління вразливостями в рамках SOC також потрібно забезпечити отримання зручних звітів для аналітиків SOC.

2.6. Проєктування інформаційних панелей для моніторингу показників

Інформаційні панелі допомагають спеціалістам операційного центру безпеки ефективно аналізувати дані, які надходять до SIEM системи. Вчасне виявлення підозрілих подій, аномалій у поведінці користувачів або систем, а також активності потенційних загроз великою мірою залежить від зручної та інформативності інформаційних панелей. Панелі дозволяють швидко отримати загальну картину стану безпеки, а також перейти до деталізованого аналізу інцидентів для ухвалення рішень щодо подальших дій реагування.

Загальна інформаційна панель є найважливішою, її основне призначення полягає у забезпеченні миттєвого доступу до загальної картини подій безпеки. Ця панель виконує роль своєрідного дашборду SOC аналітика, на якому він бачить найважливіші метрики в реальному часі. В свою чергу після виявлення аномалій або відхилення в метриках на загальному дашборді спеціаліст може детальніше провести аналіз та розслідування за допомогою додаткових модулів таких як Discovery, де зберігаються всі логи.

Передбачається, що ця панель міститиме віджети для відображення статусу підключених агентів, що дозволяє оцінити охоплення моніторингу та виявити потенційно неактивні вузли.

Також на панелі буде реалізовано підрахунок усіх подій за обраний період разом із фільтрацією критичних та високорівневих, що дає змогу оперативно відреагувати на потенційно небезпечні інциденти.

Тобто передбачається вивід списку високих та критичних повідомлень, які будуть включати тривоги Active Response, VirusTotal, AbuseIPDB та інші. Цей список відображатиме останні критичні події з описами, датами, рівнями серйозності та ідентифікаторами правил. Він фактично є точкою старту для розслідування інцидентів, дозволяючи одразу перейти до аналізу найбільш значущих подій.

Крім цього, важливим є виведення окремих лічильників для подій, пов’язаних із механізмами активної відповіді, шкідливими IP адресами та

позитивними виявленнями з VirusTotal. Це дозволить візуально і швидко оцінити кількість зовнішніх загроз та результативність захисних механізмів.

Загальна інформаційна панель повинна стати зручною точкою входу для всіх учасників процесу моніторингу, як для аналітиків першої лінії, так і для інженерів, що займатимуться глибшим аналізом.

Також проєктується *Інформаційна панель подій SSH* з метою окремого винесення подій, пов'язаних з аутентифікацією по протоколу SSH, для оперативного виявлення зловмисної активності, а також аналізу брутфорс атак і спроб несанкціонованого доступу до систем. Враховуючи, що SSH є одним з основних векторів атак при доступі до серверів, необхідність у створенні спеціалізованої панелі є виправданою з погляду як безпеки, так і ефективного реагування [32].

Основою панелі виступатиме таблиця з подіями аутентифікації, де детально буде відображено IP адреси джерел, імена агентів, час спроби входу, опис спрацювання правила та рівень його критичності. Це дозволить швидко локалізувати джерело потенційної загрози і виявити патерни зловмисної поведінки, наприклад, повторювані спроби входу під різними обліковими записами.

Панель також міститиме блоки з кількісною статистикою: загальна кількість подій SSH, кількість успішних аутентифікацій та кількість невдалих. Це дасть змогу наочно відстежити баланс між легітимною та підозрілою активністю. Для візуального представлення буде використано графіки, що показуватимуть розподіл IP адрес з яких було виконано успішну та неуспішну аутентифікацію.

Ця інформаційна панель дозволить аналітикам SOC ефективно ідентифікувати аномалії в SSH та виявляти спроби брутфорсу в реальному часі. У перспективі цю панель можна інтегрувати з автоматизованими механізмами активної відповіді, що дозволить на основі певного порогу невдалих входів автоматично додавати IP адресу до чорного списку або генерувати сповіщення з високим пріоритетом.

Таким чином, *інформаційна панель SSH* забезпечить хоч і вузьконаправлений, але критично важливий контроль одного з найбільш атакованих векторів в сучасних інфраструктурах.

Також необхідно додати, що створення інформаційних панелей у Wazuh потребує розробки великої кількості окремих візуалізацій, які потім об'єднуються в єдині інформаційні панелі. Кількість і якість цих візуалізацій безпосередньо залежить від наявних правил кореляції та обсягу зібраних логів. Тому максимально широкий збір логів є критично важливим для ефективного формування інформаційних панелей.

2.7. Проєктування загальної інфраструктури для системи Wazuh

При проєктуванні інфраструктури для системи моніторингу безпеки Wazuh не менш важливим є вибір середовища, в якому ця система буде розгортатися, налаштовуватися та тестуватися. Існують різні варіанти: від ізольованих лабораторних мереж, побудованих на емуляторах типу GNS3 або віртуальних середовищах, до реальних локальних мереж і навіть хмарних платформ.

У рамках цього проєкту було прийнято рішення використати домашню мережу замість ізольованої лабораторної інфраструктури. Це пов'язано з тим, що сучасні системи класу SIEM та XDR доволі часто розгортаються в хмарних середовищах з використанням моделей IaaS (Infrastructure as a Service) або PaaS (Platform as a Service). Використання IaaS дозволяє організаціям орендувати віртуальні ресурси такі як сервери, мережеве обладнання та сховище без необхідності вкладень у фізичну інфраструктуру, що значно знижує початкові витрати. PaaS, у свою чергу, забезпечує готову платформу для розгортання та управління системами безпеки, спрощуючи процес впровадження та масштабування за рахунок автоматизації управління операційними системами, середовищами розробки і базами даних. Крім того, хмарні рішення зменшують потребу у власних IT ресурсах, зокрема відсутність необхідності у створенні та підтримці локальної мережі, що значно знижує складність і витрати на

інфраструктуру. Це дозволяє організаціям зосередитися на аналізі та реагуванні на інциденти, а не на адмініструванні мережевого обладнання.

Домашня мережа відображає типове середовище, у якому Wazuh часто експлуатується, тому її використання забезпечує більш релевантні результати. Також важливо зазначити, що мета роботи полягає у практичному розгортанні, налаштуванні та дослідженні системи у реальних умовах, а не у створенні ізольованої лабораторної мережі. Таким чином, вибір домашньої мережі є оптимальним для досягнення поставлених цілей дипломної роботи.

Наступним є *розподіл IP адрес* між сервером Wazuh та кінцевими точками мережі. Сервер Wazuh, що буде розгорнутий на Linux Ubuntu Server матиме статичну IP адресу, що гарантуватиме його стабільну доступність в мережі, та спростить налаштування міжмережевого екрану, а також дозволить безперервно підтримувати підключення агентів та інших сервісів без ризику втрати зв’язку через зміну адреси.

Водночас для інших пристроїв у мережі, зокрема кінцевих точок буде використано динамічне призначення IP адрес через DHCP. Це дозволить оптимізувати керування адресним простором, спростить адміністрування мережі, а також забезпечить автоматичне підключення нових пристроїв без необхідності ручного налаштування IP.

Використання такої архітектури є доцільним для домашньої мережі, оскільки забезпечує баланс між зручністю адміністрування та стабільністю роботи системи безпеки.

Статична IP для сервера Wazuh є необхідною умовою для налаштування надійного моніторингу та збирання логів з усіх кінцевих точок, що працюють із динамічними адресами.

Як результат, спроєктована мережа складатиметься з одного сервера з фіксованою IP адресою та декількох кінцевих пристроїв з динамічними IP адресами, що забезпечить гнучкість, надійність і зручність у подальшому масштабуванні та адмініструванні Wazuh.

Стосовно *кінцевих пристроїв* на яких буде встановлено Wazuh Agent. Проєктується підключення трьох кінцевих пристроїв з операційними системами Windows 11, Linux Ubuntu та macOS Big Sur.

Для організації моніторингу та управління цими пристроями у Wazuh буде створено окремі групи для кожного типу операційної системи, а саме групи windows, linux та macos. Розподіл на групи дозволить застосовувати унікальні налаштування, що відповідатимуть специфіці кожної ОС.

Для адміністрування серверів та кінцевих пристроїв у проєкті в першу чергу буде використано *протокол SSH*, який забезпечить безпечне шифроване з'єднання для віддаленого доступу, однак GUI операційних систем теж буде використано. Для роботи з SSH з'єднаннями буде використовуватися клієнт PuTTY.

PuTTY дозволить швидко встановити з'єднання з сервером, а також з іншими кінцевими пристроями, що підтримують SSH. Важливою особливістю використання PuTTY є можливість збереження конфігурацій підключень у вигляді сесій, що значно спростить повторне підключення та управління кількома серверами одночасно.

Збереження та управління сесіями SSH в PuTTY дозволяє централізовано організувати доступ до серверів, уникнути повторного введення параметрів підключення та підвищити ефективність роботи адміністратора. Це є доволі зручно при регулярному моніторингу, налаштуванні та усуненні різних несправностей.

Для *розгортання сервера Wazuh обрана операційна система Ubuntu* версії Ubuntu Server, яка є стабільною, безпечною та широко підтримуваною платформою для серверних застосунків. Наступним кроком буде налаштування мережевих параметрів, а саме призначення статичної IP адреси.

Для підвищення рівня безпеки на сервері буде налаштований міжмережевий екран за допомогою утиліти ufw (Uncomplicated Firewall). Ufw дозволить просто і ефективно створити правила доступу, які обмежать небажані з'єднання та відкриють лише необхідні порти для роботи Wazuh і SSH. Також матиме сенс

дозволити доступ підключення до сервера тільки зі списку IP адресів домашньої мережі.

Для забезпечення безпечного доступу до сервера буде використовуватися виключно протокол *SSH з автентифікацією на основі сертифікатів*. Протокол Telnet, який не забезпечує шифрування, буде вимкнено, щоб виключити можливі ризики безпеки.

Також під час встановлення потрібно обрати не менше 50ГБ дискового простору, що дозволить уникнути проблем, пов’язаних із переповненням диску.

Висновки до другого розділу

У другому розділі було здійснено проєктування SIEM та XDR підсистеми для SOC на базі платформи Wazuh. Проведено аналіз типових кіберзагроз, з якими стикаються сучасні SOC, та визначено пріоритетні завдання, які Wazuh здатен ефективно вирішувати в контексті SOC.

Проєктування охопило ключові аспекти функціоналу системи. Розроблено механізми виявлення аномалій та шкідливої активності, що включає моніторинг цілісності файлів, аналіз командної активності, виявлення аномальної мережевої поведінки, а також інтеграції із зовнішніми репутаційними сервісами, такими як VirusTotal та AbuseIPDB. Окрім виявлення, спроєктовано автоматизоване реагування, зокрема налаштування модуля Active Response для видалення шкідливих файлів та блокування небезпечних мережевих з’єднань. Додатково оглянуто та спроєктовано використання таких модулів, як інвентаризація систем, оцінка конфігурації безпеки та управління вразливостями. Особливу увагу приділено проєктуванню зручних та інформативних візуалізацій.

Також, визначено архітектуру розгортання системи, яка передбачає використання домашньої мережі. Опрацьовано питання розподілу IP адрес для забезпечення стабільності та безпеки функціонування Wazuh.

Таким чином, у другому розділі було спроєктовано SIEM та XDR підсистему на базі Wazuh.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ СПРОЄКТОВАНОЇ ПІДСИСТЕМИ

3.1. Налаштування інфраструктури та розгортання Wazuh

Перед початок налаштування та конфігурування системи управління інформаційною безпекою та подіями безпеки Wazuh необхідно підготувати інфраструктуру яка була спроєктована в другому розділі.

А саме за допомогою середовища емуляції кінцевих вузлів Oracle VirtualBox необхідно встановити ОС Ubuntu Server та провести попередньо спроєктовані налаштування для імплементації захисту. Після чого вже можна приступити до базового розгортання платформи Wazuh SIEM та XDR.

Крім того, необхідно за допомогою VirtualBox також встановити ОС Windows 11, Linux Ubuntu та MacOS Big Sur, які будуть використовуватися як кінцеві пристрої для встановлення Wazuh Agent.

Приступаючи до налаштування інфраструктури та розгортання Wazuh необхідно завантажити та встановити Oracle VirtualBox. Це програмне забезпечення вже було попередньо встановлено на основній системі.

Далі приступаємо до завантаження та встановлення Ubuntu Server, на якому безпосередньо вже буде встановлена платформа Wazuh SIEM та XDR. Було завантажено образ з офіційного веб сайту (Рис. 3.1).

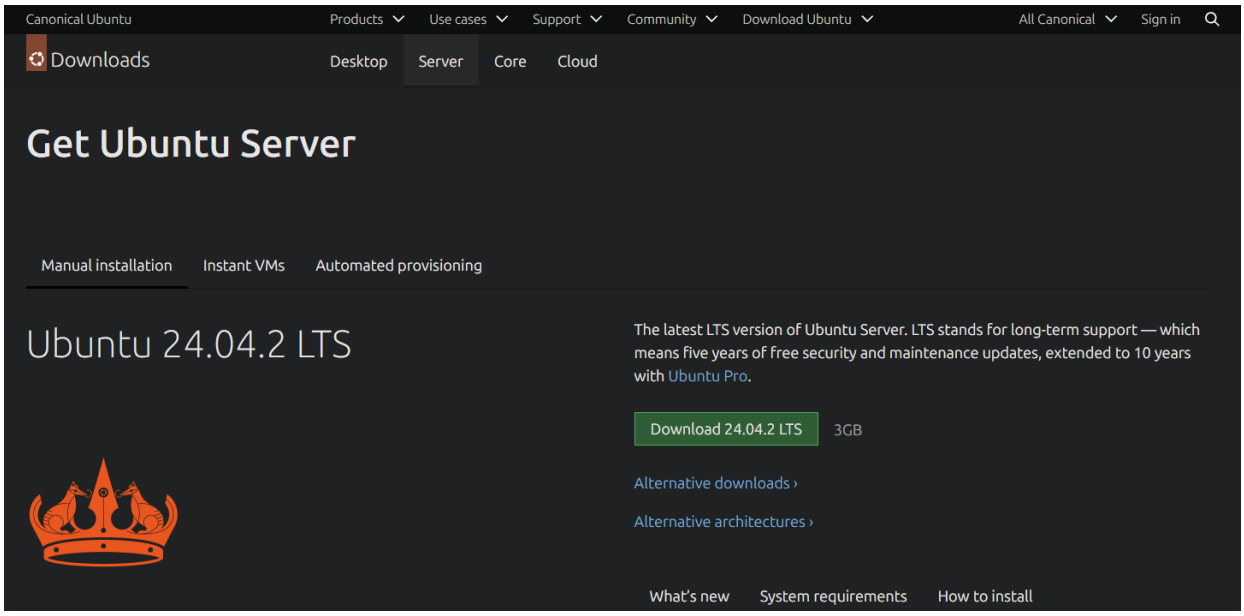


Рисунок 3.1 – Завантаження Ubuntu Server з офіційного веб сайту

Після чого було створено нову віртуальну машину у VirtualBox з використанням попередньо завантаженого образу (рис. 3.2).

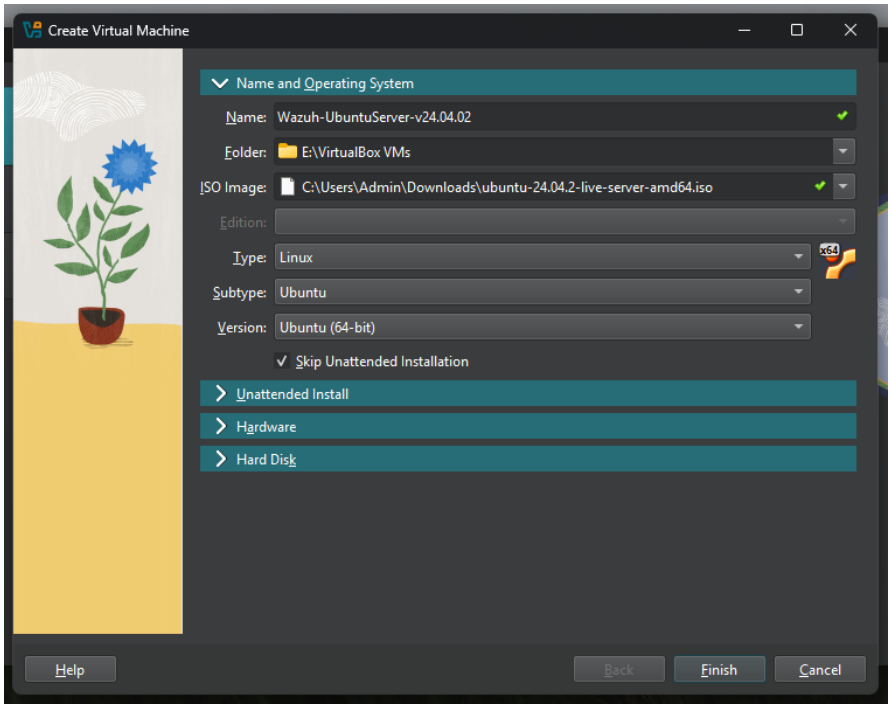


Рисунок 3.2 – Імпортування Ubuntu Server до VirtualBox

Під час імпортування було налаштовано розмір ОЗУ на рівні 8ГБ, кількість ядер процесора на рівні 4 та розмір жорсткого диску в 50 ГБ.

Далі було проведено встановлення Ubuntu Server та його базове конфігурування під час встановлення [34] (рис. 3.3).

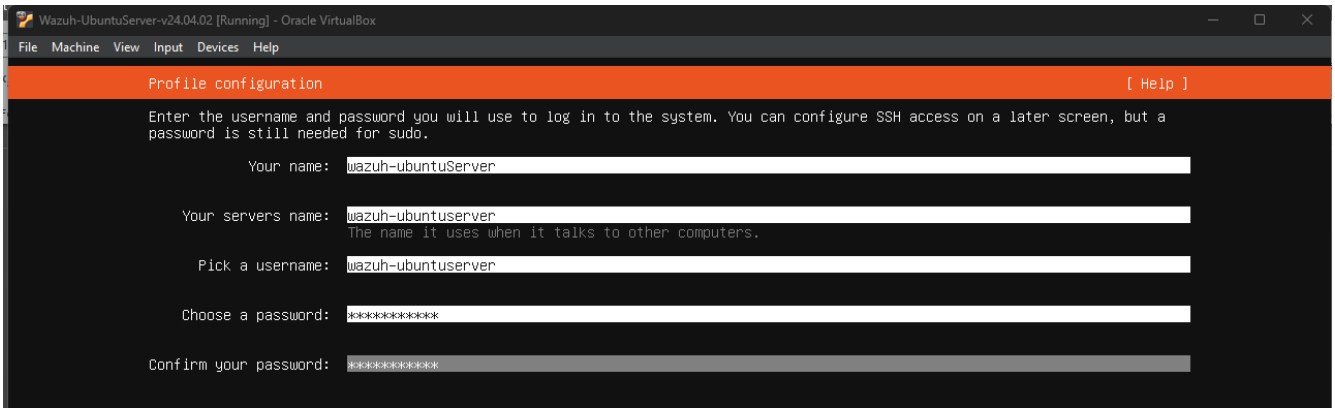


Рисунок 3.3 – Налаштування профілю

Також під час встановлення Ubuntu Server було одразу встановлено OpenSSH сервер для можливості підключення по SSH (рис. 3.4).

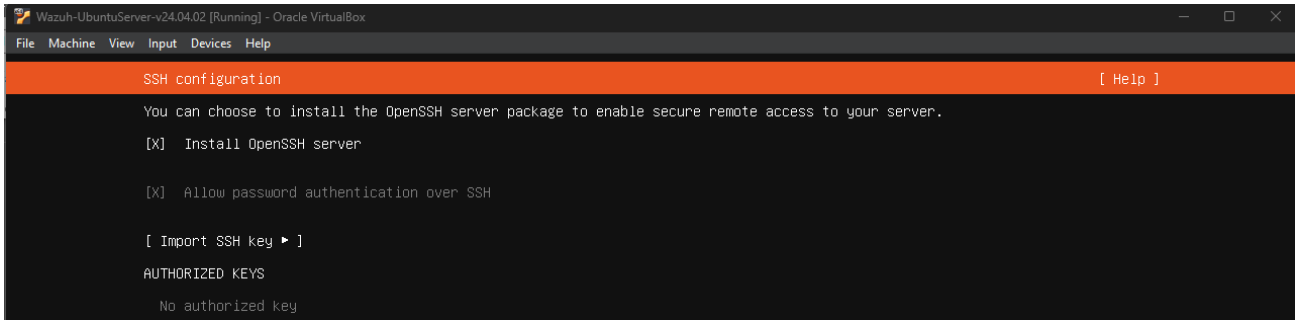


Рисунок 3.4 – Встановлення OpenSSH серверу на Ubuntu Server

Після успішного встановлення Ubuntu Server та входу в систему як wazuh-ubuntuserver було оновлено всі встановленні пакети для уникнення помилок та використання неактуального або вразливого ПЗ (рис. 3.5).

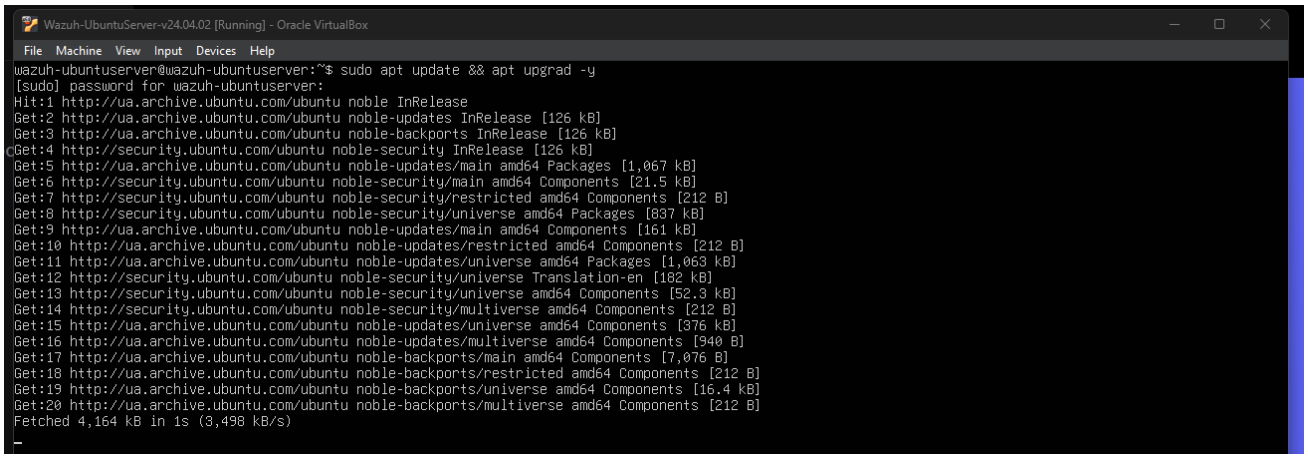


Рисунок 3.5 – Оновлення встановлених пакетів

Далі було налаштовано статичну IP адресу для Ubuntu Server. Було відредаговано конфігураційний файл /etc/netplan/50-cloud-init.yaml (рис. 3.6). Після внесення змін до конфігураційного файлу було також використано команду sudo netplan apply для застосування нових налаштувань.

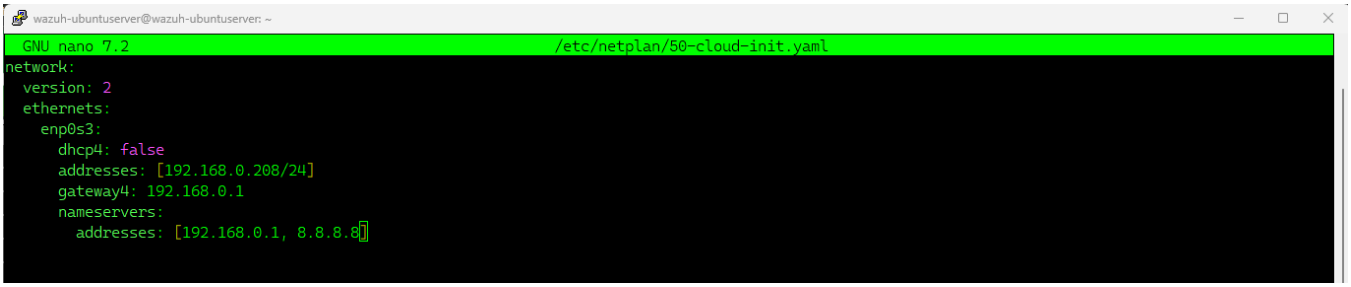


Рисунок 3.6 – Призначення статичної IP адреси для Ubuntu Server

Також на домашньому маршрутизатор було виключено IP адресу 192.168.0.208 зі списку IP адрес для DHCP. Також було зв’язано IP адресу до MAC адреси Ubuntu Server для підвищення рівня захисту (рис. 3.7).

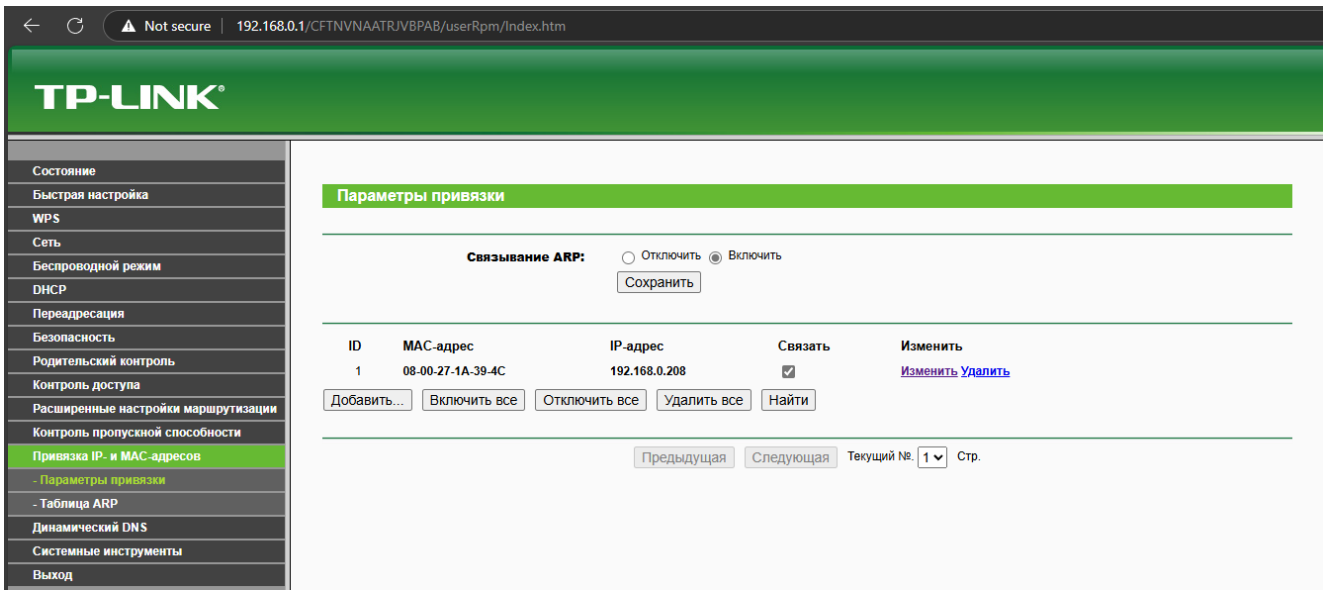


Рисунок 3.7 – Зв’язування IP адреси з MAC адресою Ubuntu Server

Після налаштування статичної адресації на сервері було увімкнено та налаштовано фаєрвол UFW для можливості підключення тільки до необхідний портів. Наприклад, по SSH можуть підключитися тільки пристрої які знаходяться в домашній мережі (рис. 3.8).

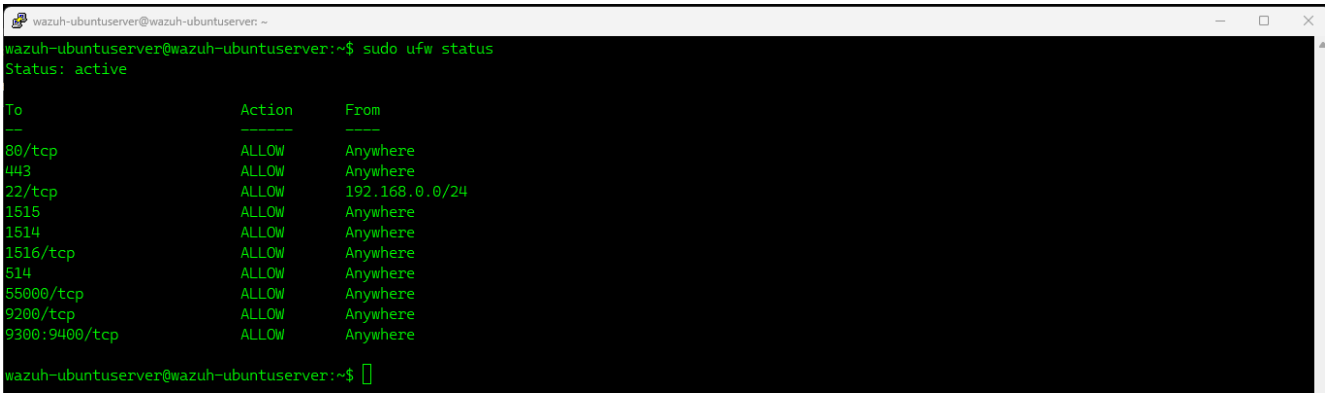


Рисунок 3.8 – Статус фаєрволу UFW після налаштувань на Ubuntu Server

В свою чергу OpenSSH сервер на Ubuntu Server був налаштований на максимальну кількість спроб в 6 разів, максимум 2 сесії одночасно, та можливість аутентифікації виключно використовуючи публічний ключ (рис. 3.9).

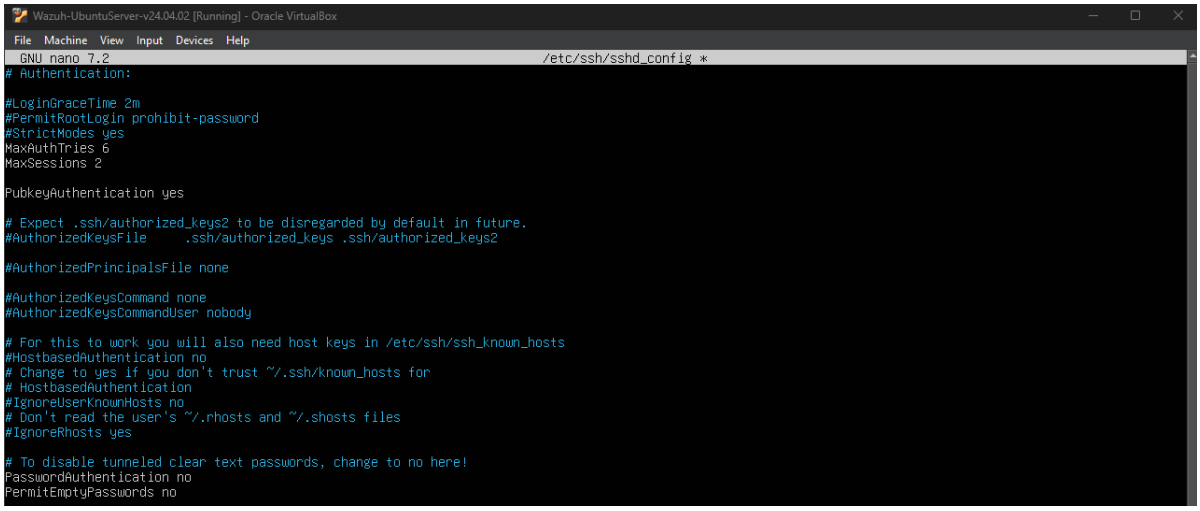


Рисунок 3.9 – Налаштування OpenSSH серверу на Ubuntu Server

Надалі на основній системі було згенеровано пару публічного та приватного ключа типу ed25519 з довжиною 4096 біт, що забезпечує високий рівень безпеки. Публічний ключ було надіслано на Ubuntu Server (рис. 3.10 – рис. 3.11).

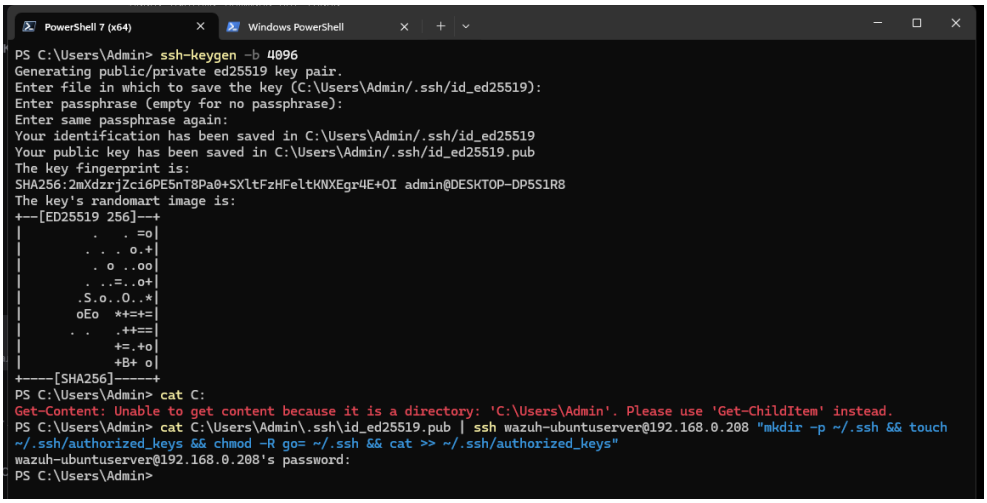


Рисунок 3.10 – Генерація пари приватного та публічного ключа



Рисунок 3.11 – Успішна аутентифікація з використанням публічного ключа

Після того як було успішно встановлено та налаштовано Ubuntu Server можна приступити до розгортання та базового налаштування SIEM та XDR платформи Wazuh. Було завантажено скрипт для автоматичного встановлення

Wazuh з офіційного веб сайту. Скрипт передбачає автоматичне завантаження та встановлення компонентів Wazuh Server, Dashboard та Indexer на єдиному Linux сервері, в нашому випадку на Linux Ubuntu Server (рис. 3.12).

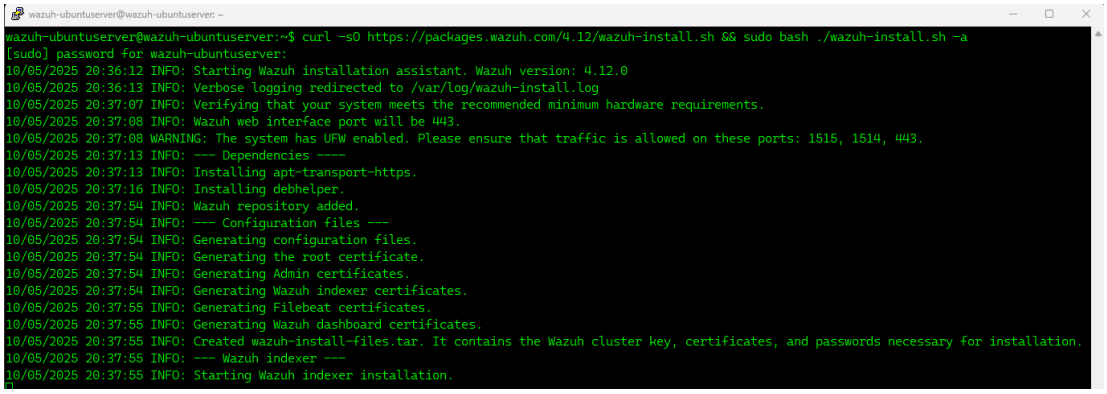


Рисунок 3.12 – Завантаження скрипта та встановлення компонентів Wazuh

На завершенні інсталяційного процесу скрипт надає облікові дані для входу до Wazuh Dashboard як користувач з правами admin (рис. 3.13).

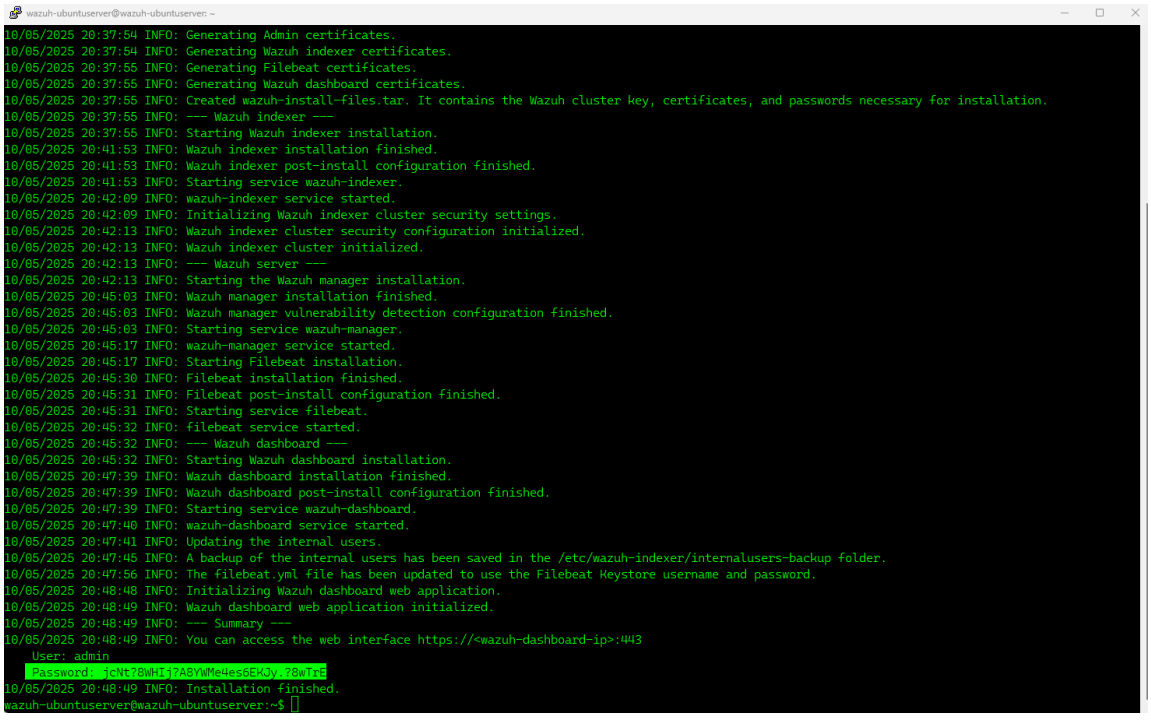


Рисунок 3.13 – Завершення встановлення Wazuh та отримання облікових даних

Крім того, додатково, після встановлення Wazuh компонентів було видалено посилання зі списку джерел apt для унеможливлення незапланованого оновлення одного з компонентів, що може пошкодити роботу системи.

На кінець встановлення Wazuh було виконано вхід до Wazuh Dashboard для перевірки працездатності встановленої системи (рис. 3.14).

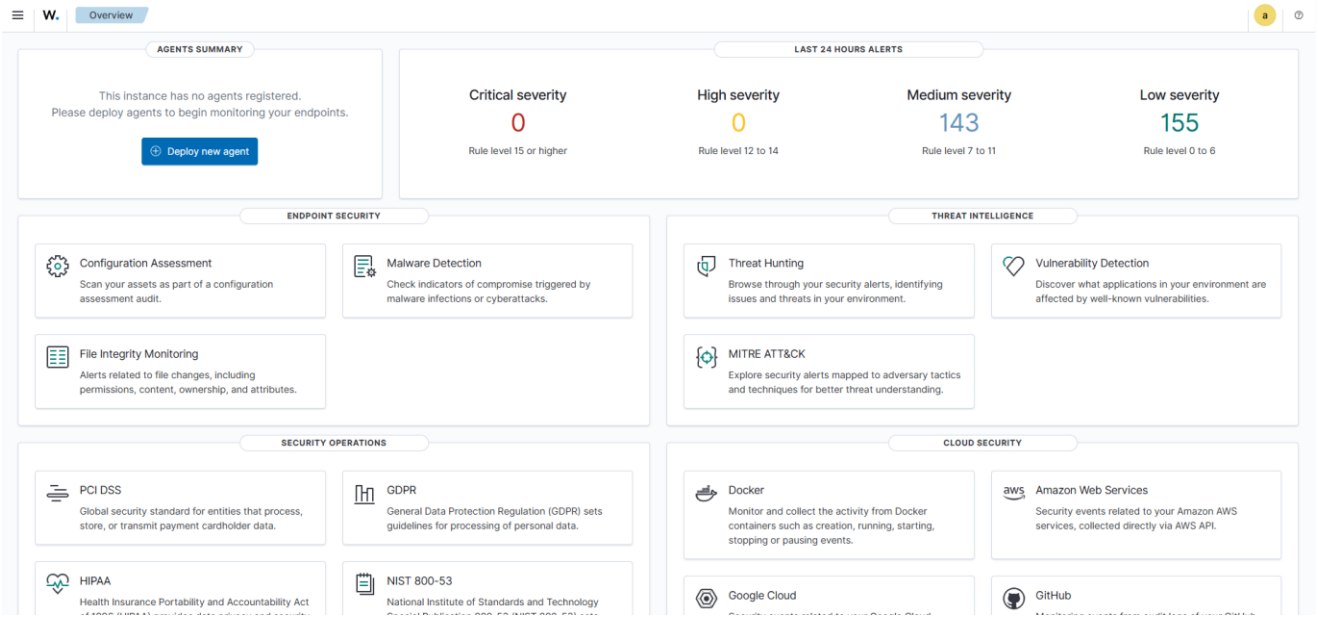


Рисунок 3.14 – Успішний вхід до Wazuh Dashboard

Надалі було завантажено образи операційних систем Windows 11, Linux Ubuntu та MacOS Big Sur та імпортовано до Oracle Virtual Box (рис. 3.15 – рис. 3.17).

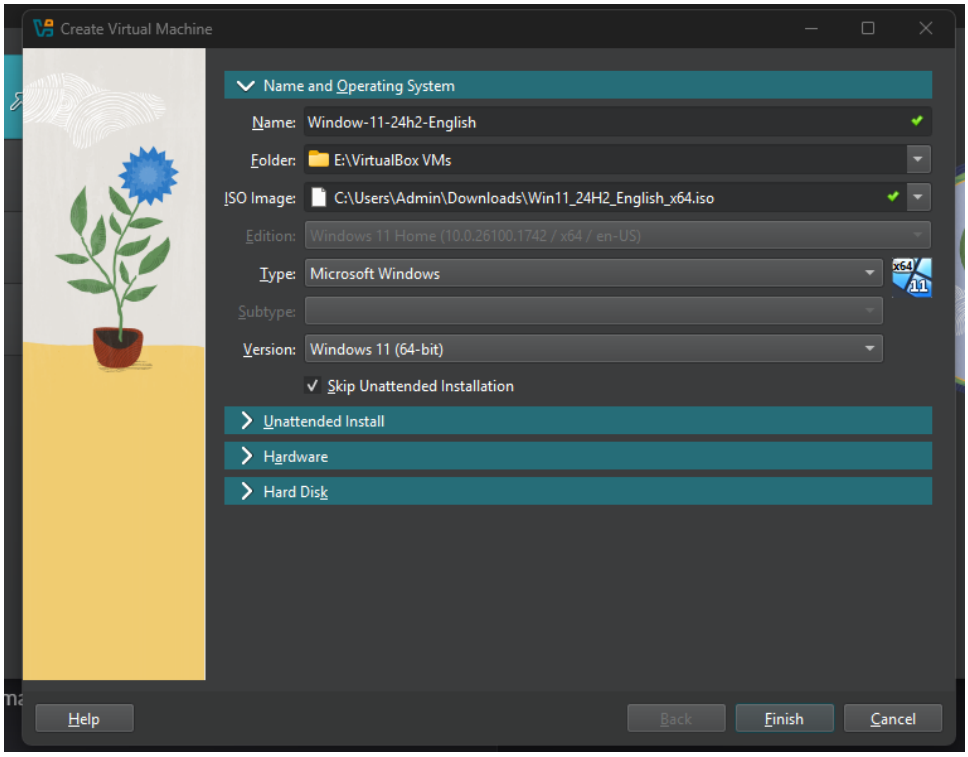


Рисунок 3.15 – Створення віртуальної машини з ОС Windows 11

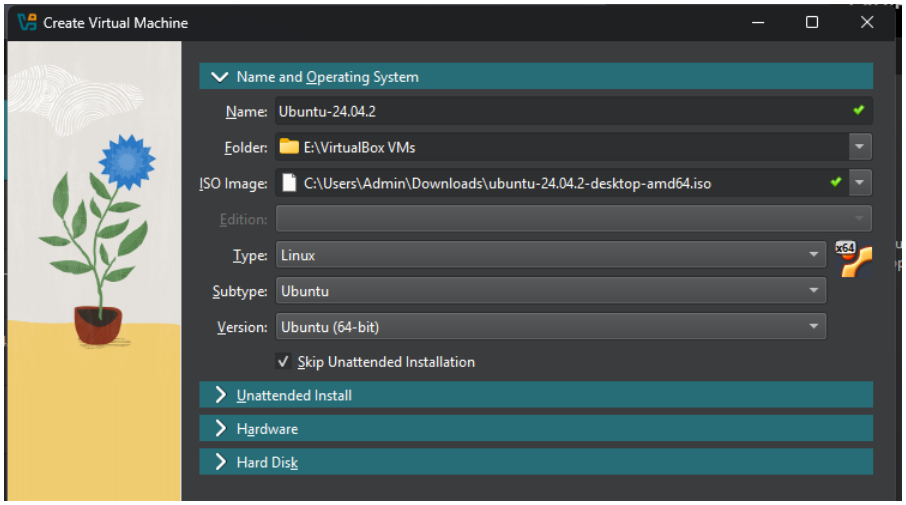


Рисунок 3.16 – Створення віртуальної машини з ОС Linux Ubuntu

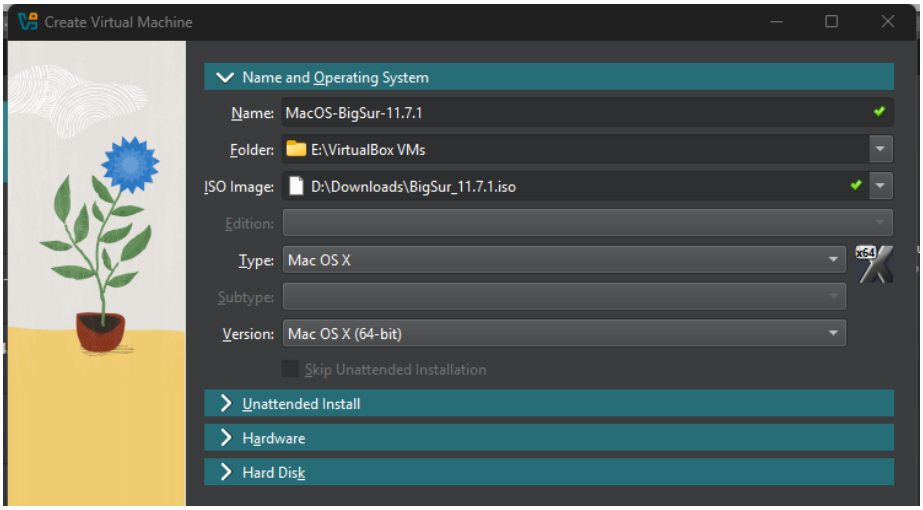


Рисунок 3.17 – Створення віртуальної машини з ОС MacOS BigSur

Після успішного створення віртуальних машин з використання завантажених iso файлів було проведено встановлення кожної операційної системи на відповідних віртуальних машинах (рис. 3.18 – рис. 3.19).

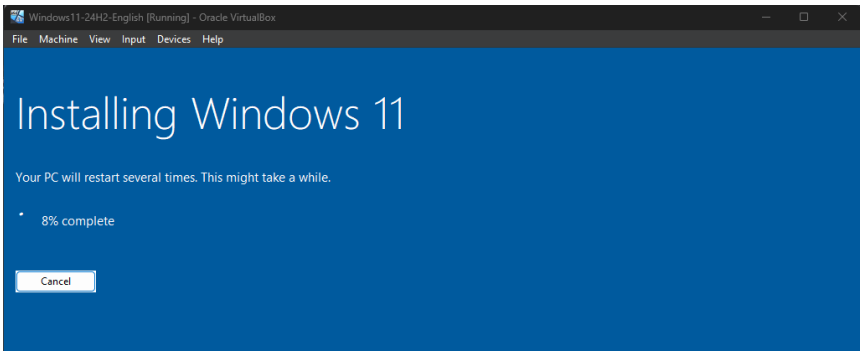


Рисунок 3.18 – Встановлення ОС Windows 11 на віртуальну машину

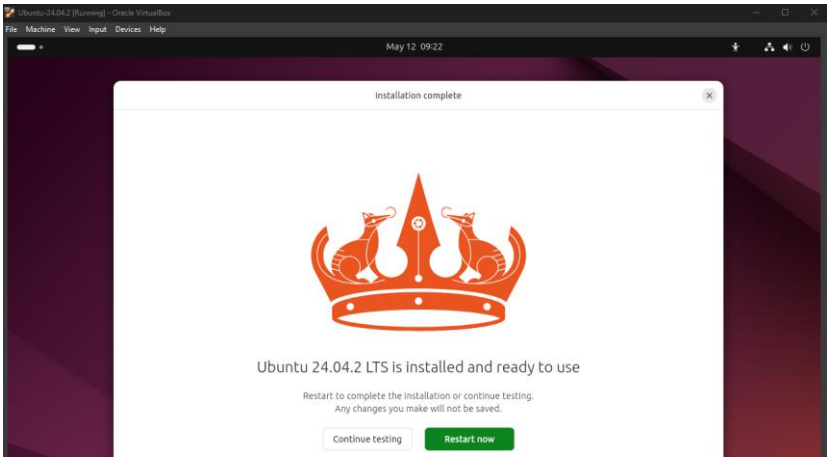


Рисунок 3.19 – Встановлення ОС Linux Ubuntu на віртуальну машину

Однак для встановлення MacOS Big Sur необхідно було провести додаткові операцій, адже вона не підтримує центральні процесори AMD та інші додаткові компоненти (рис. 3.20). Після проведення операцій зі зміни значень компонентів, було успішно встановлено MacOS Big Sur (рис. 3.21).

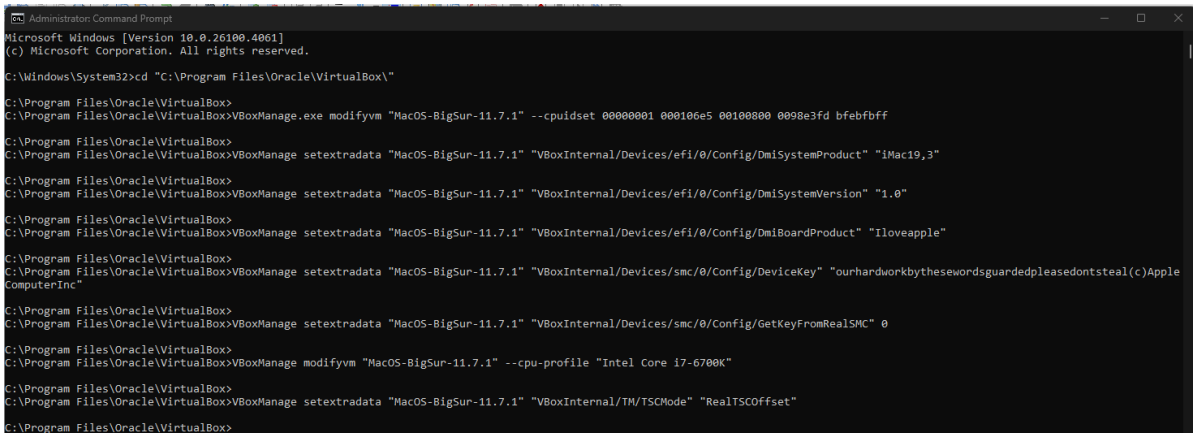


Рисунок 3.20 – Операції зі зміни значень компонентів віртуальної машини MacOS

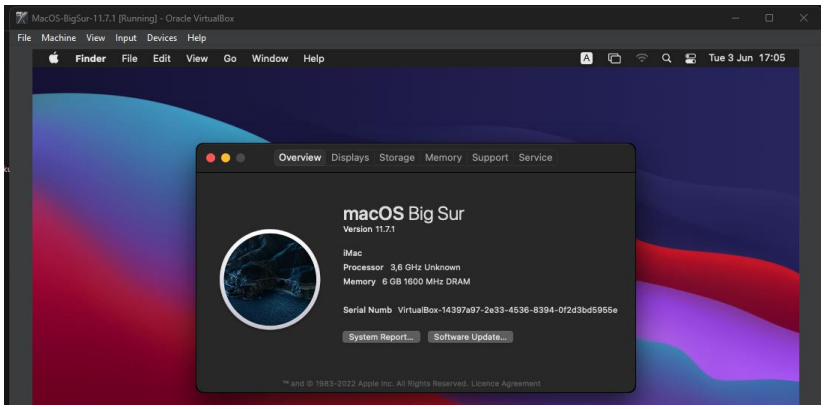


Рисунок 3.21 – Успішно встановлена ОС MacOS Big Sur на віртуальну машину

Попередньо перед проведенням налаштувань системи Wazuh необхідно також встановити Wazuh Agent на вже готові віртуальній машини з різними операційними системами.

Процес встановлення Wazuh Agent складається зі створення унікальної конфігурації для кожного агента за допомогою відповідного функціоналу у Wazuh Dashboard, після чого генерується спеціальне посилання для завантаження Wazuh Agent з налаштованою конфігурацією (рис. 3.22 – рис. 3.23).

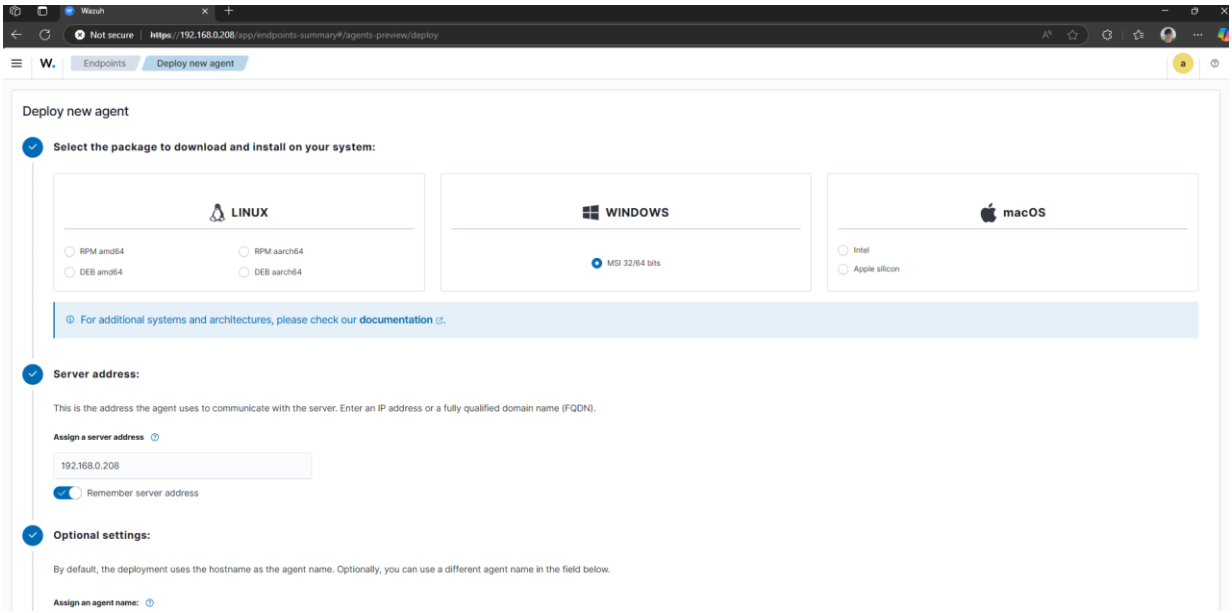


Рисунок 3.22 – Створення конфігурації для розгортання нового Wazuh Agent

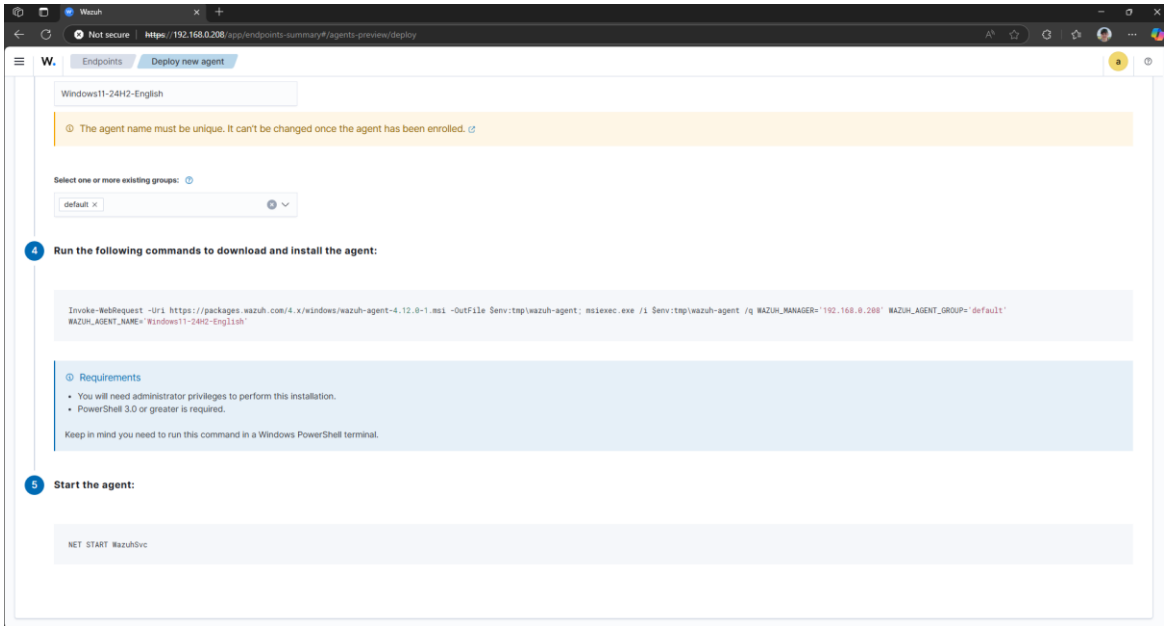


Рисунок 3.23 – Створення конфігурації для розгортання нового Wazuh Agent

Команду разом з посиланням, яка була вже згенерована за допомогою Wazuh Dashboard необхідно виконати від користувача адміністратора або від root в залежності від ОС.

Таким чином Wazuh Agent було успішно встановлено на трьох віртуальних машинах з операційними системами Windows, Linux та MacOS (рис. 3.24, рис. 3.25 – рис. 3.26).

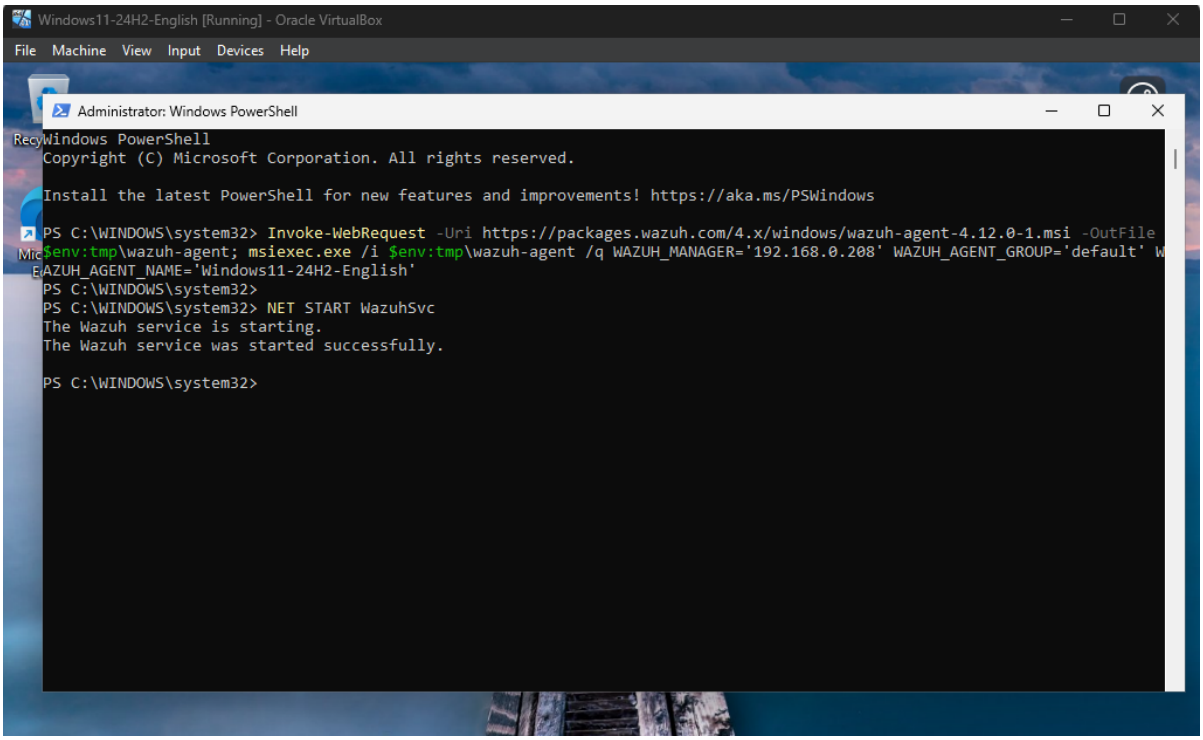


Рисунок 3.24 – Успішно встановлений та запущений Wazuh Agent на Windows 11

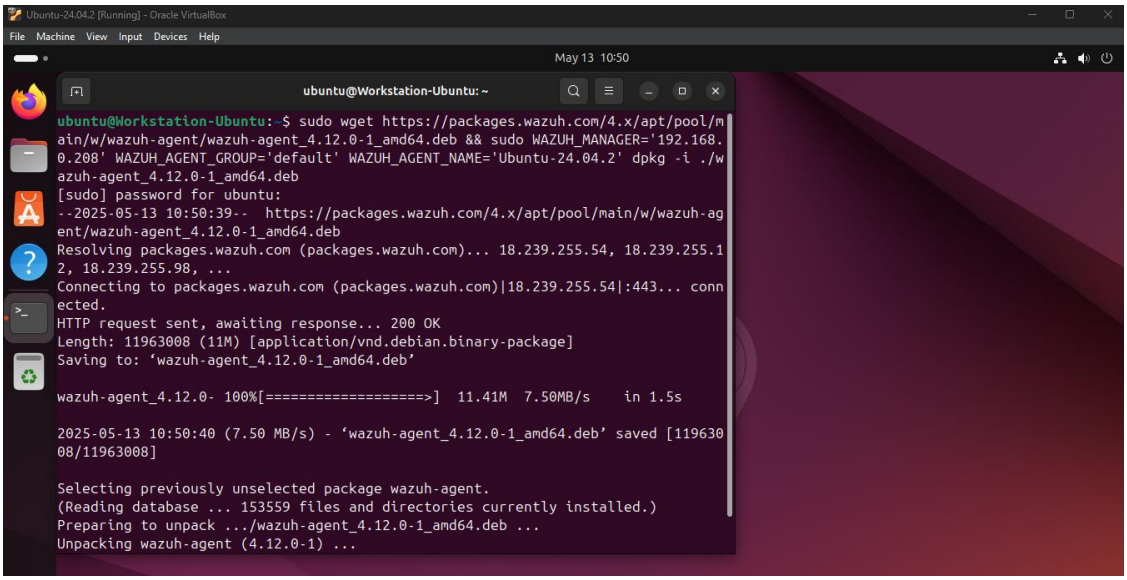


Рисунок 3.25 – Успішно встановлений та запущений Wazuh Agent на Linux Ubuntu

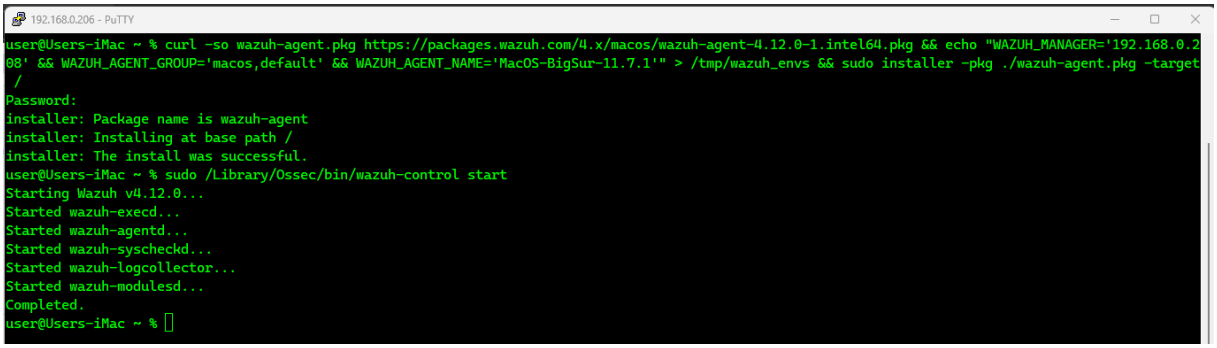


Рисунок 3.26 – Успішно встановлений та запущений Wazuh Agent на MacOS

Після встановлення Wazuh Agent на віртуальні машини, було створено унікальні групи у Wazuh для Windows, Linux та MacOS (рис. 3.27 – рис. 3.29). Створення окремих груп надає можливість гнучко налаштовувати конфігурацію для кінцевих пристроїв з різними ОС. До прикладу надалі буде налаштовано FIM модуль, і відповідно в кожній ОС шляхи директорій різні, в цьому випадку створення груп допомагає визнати шляхи директорій для кожної ОС окремо.

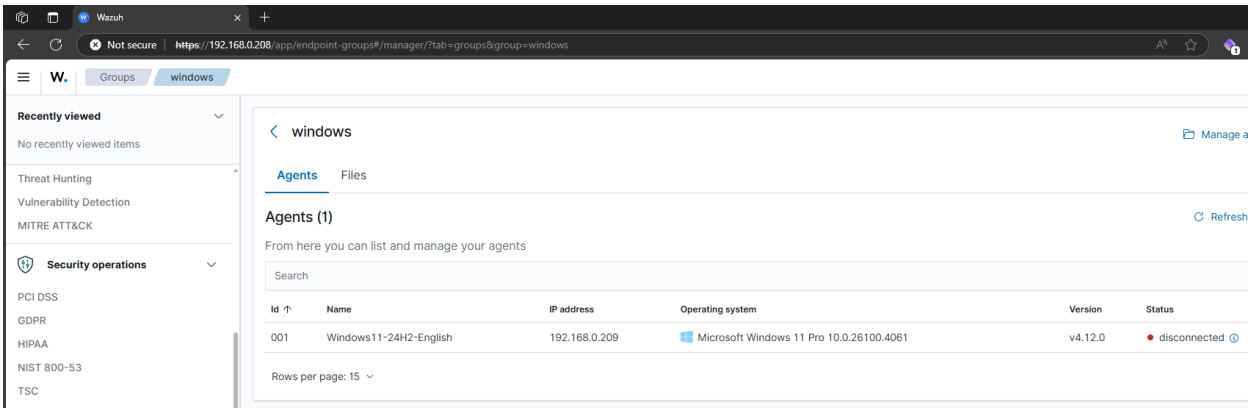


Рисунок 3.27 – Створена група для кінцевих пристроїв з ОС Windows

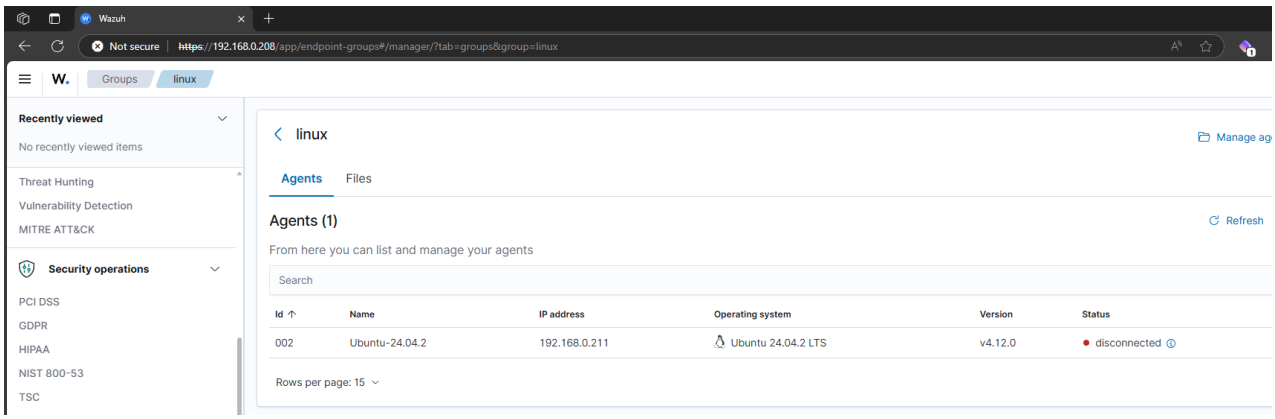


Рисунок 3.28 – Створена група для кінцевих пристроїв з ОС Linux

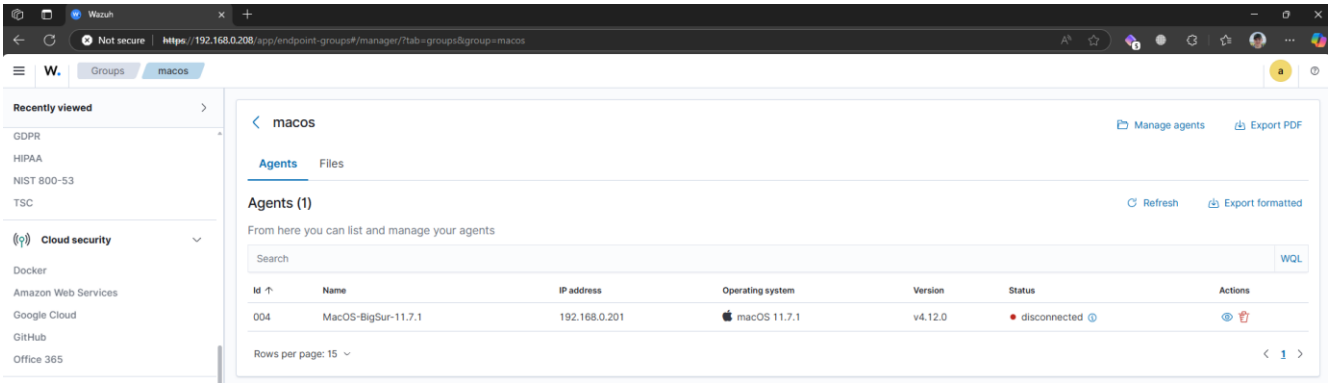


Рисунок 3.29 – Створена група для кінцевих пристроїв з ОС MacOS

Крім того, Wazuh надає можливість створювати різноманітні групи в залежності від потреб, що дає змогу групувати та додавати кінцеві пристрої відповідно до потреб та поставлених задач.

Для коректної роботи віддаленої конфігурації необхідно дозволити виконання віддалених команд на кінцевих пристроях. Налаштування відбувається в конфігураційному файлі local_internal_options.conf в директорії Wazuh Agent на кінцевому пристрої. Тож, були проведені налаштування та увімкнено можливість виконання віддалених команд для Windows, Linux та MacOS (рис. 3.30 – рис. 3.32).

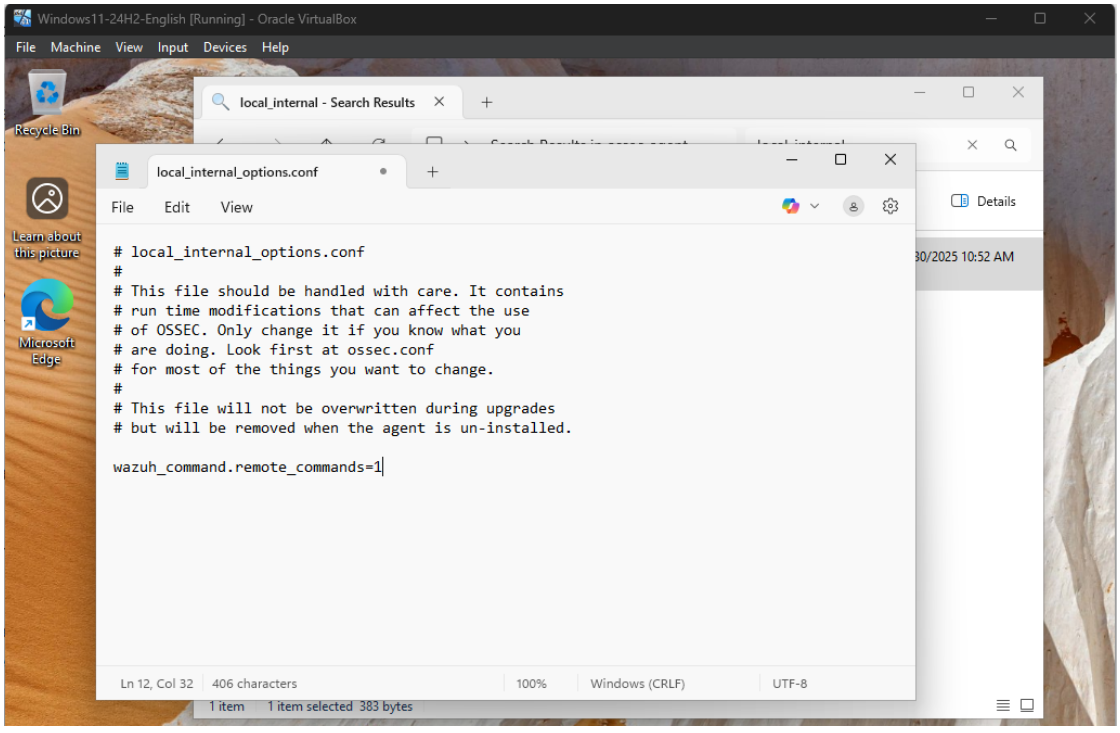


Рисунок 3.30 – Налаштування дозволу виконання віддалених команд на Windows

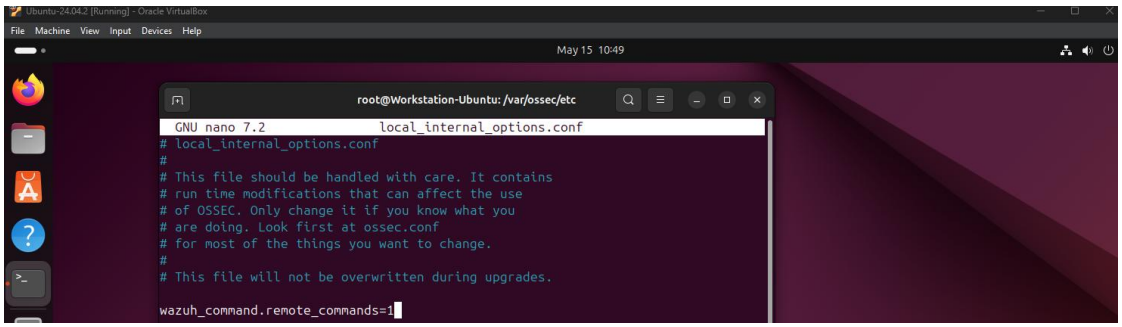


Рисунок 3.31 – Налаштування дозволу виконання віддалених команд на Linux

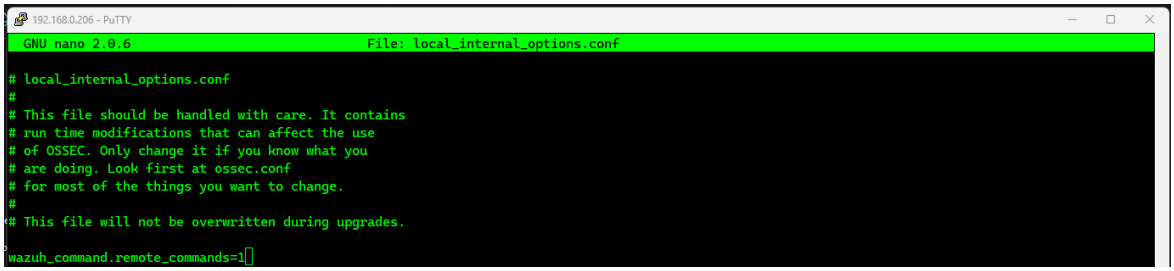


Рисунок 3.32 – Налаштування дозволу виконання віддалених команд на MacOS

В разі відсутності дозволу на виконання віддалених команд унікальна конфігурація для створених груп не буде надсилатися на кінцеві пристрої. Зазвичай цей дозвіл відсутній задля підвищення рівня безпеки та щоб унеможливити виконання шкідливих команд в разі отримання зловмисником доступу до платформи Wazuh SIEM та XDR.

3.2. Налаштування механізмів збору логів та виявлення загроз

3.2.1. Налаштування модуля File Integrity Monitoring

Модуль File Integrity Monitoring є одним з найважливіших у Wazuh, оскільки він є основною для коректної роботи певних інших модулів. FIM модуль має можливість проводити моніторинг в режимі реального часу або за розкладом, фіксуючи такі дії, як створення, зміна, видалення або зміна прав доступу до файлів і директорій. FIM є головним інструментом для забезпечення відповідності безпековим стандартам таким як HIPPA.

Для групи кінцевих пристроїв з ОС Windows було налаштовано моніторинг користувацьких директорій Downloads, Desktop та Documents. Крім того, було

також увімкнуто щогодинні сканування системних директорій System32 та Startup, а також деяких системних регістрів (рис. 3.33).

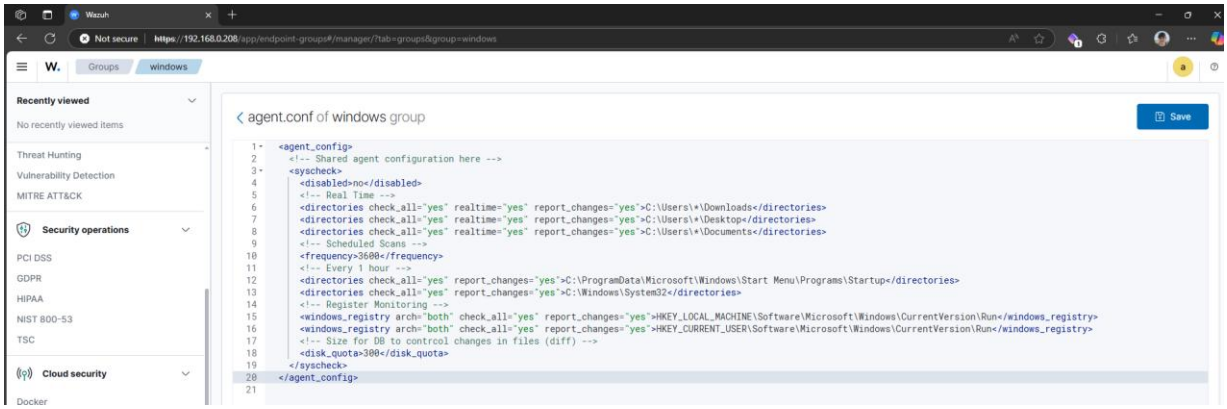


Рисунок 3.33 – Налаштування модуля FIM для пристроїв з ОС Windows

Необхідно також зазначити, що модуль FIM має різноманітні налаштування, в нашому випадку для всіх ОС було налаштовано захоплення та перевірку всіх можливих атрибутів таких як власник, розмір, хеш SHA256 та інших. Також для користувацьких директорій було налаштовано перевірку в реальному часі без затримок, в інших випадках перевірки відбуваються щогодини. Додатково було увімкнено звітування змін, тобто змінений файл порівнюється з тим що був без змін. Цей механізм реалізовано за допомогою збереження та порівняння проміжних змін файлів, що знаходяться у спеціальній БД на кінцевому пристрої.

Для групи кінцевих пристроїв з ОС Linux було теж налаштовано моніторинг користувацьких директорій Downloads, Desktop та Documents. Крім того, було увімкнуто щогодинні сканування для системних директорій /usr/bin/ та /etc/init.d, в яких зберігаються системні виконувані файли та скрипти запуску сервісів під час старту або зупинки системи відповідно (рис. 3.34).

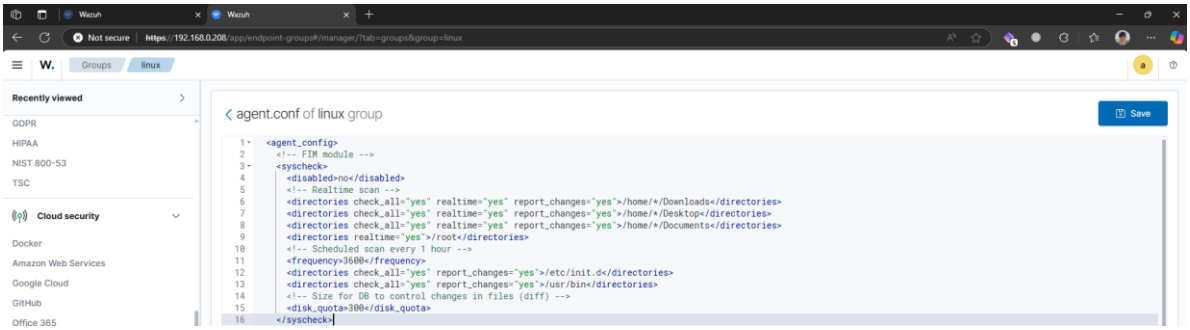


Рисунок 3.34 – Налаштування модуля FIM для пристроїв з ОС Linux

Для групи кінцевих пристроїв з ОС MacOS було теж налаштовано моніторинг користувацьких директорій Downloads, Desktop та Documents (рис. 3.35). Однак варто зауважити, що нажаль наразі MacOS не підтримує моніторинг змін в реальному часі, тому було вирішено налаштувати таким чином що би загальні сканування відбувалися щохвилини. Це надає нам змогу з певною періодичністю відслідковувати зміни в необхідних нам директоріях. Через це було вимкнено моніторинг двох системних директорій, адже в разі увімкнення буде генеруватися неймовірно велика кількість повідомлень через вміст великої кількості файлів.

В той час як моніторинг в реальному часі здатен відслідковувати тільки зміни, то повні сканування такого функціоналу не надають, та сканують в свою чергу все що знаходиться в директорії.

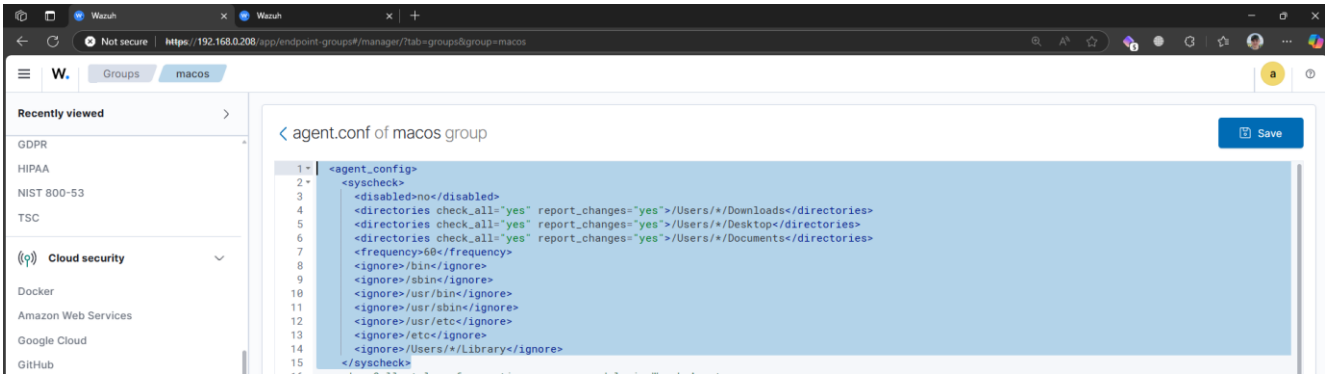


Рисунок 3.35 – Налаштування модуля FIM для пристроїв з ОС MacOS

В залежності від потреби є можливість змінювати інтервали, директорії для сканування або відслідковування, та інші важливі налаштування модуля FIM що реалізується за допомогою механізму syscheck.

3.2.2. Конфігурування інтеграції з сервісом VirusTotal

Модуль File Integrity Monitoring є безумовно корисним, однак він стає ще більше інформативним якщо його інтегрувати разом з сервісом для перевірки файлів та посилань на шкідливість VirusTotal. Цей сервіс зберігає всі аналізи, які він виконує, дозволяючи користувачам шукати хеші файлів.

Інтеграція FIM модуля та сервісу VirusTotal працює шляхом передачі хешів файлів, які обробив модуль FIM до сервісу VirusTotal за допомогою HTTP POST запиту до бази даних через VirusTotal API.

Інтеграція отримує відповідь у форматі JSON, яка є результатом запиту. Відповідь викликає одне з наступних сповіщень Wazuh:

- Помилка: перевірте облікові дані.
- Помилка: досягнуто ліміту швидкості запитів публічного API.
- Сповіщення: Немає записів у базі даних VirusTotal.
- Сповіщення: позитивних моментів не виявлено.
- Сповіщення: X двигунів виявили цей файл.

Для початку інтеграції необхідно отримати ключ API для можливості надсилання запитів до БД VirusTotal. З цією метою було створено обліковий запис VirusTotal та отримано необхідний API ключ (рис. 3.36).

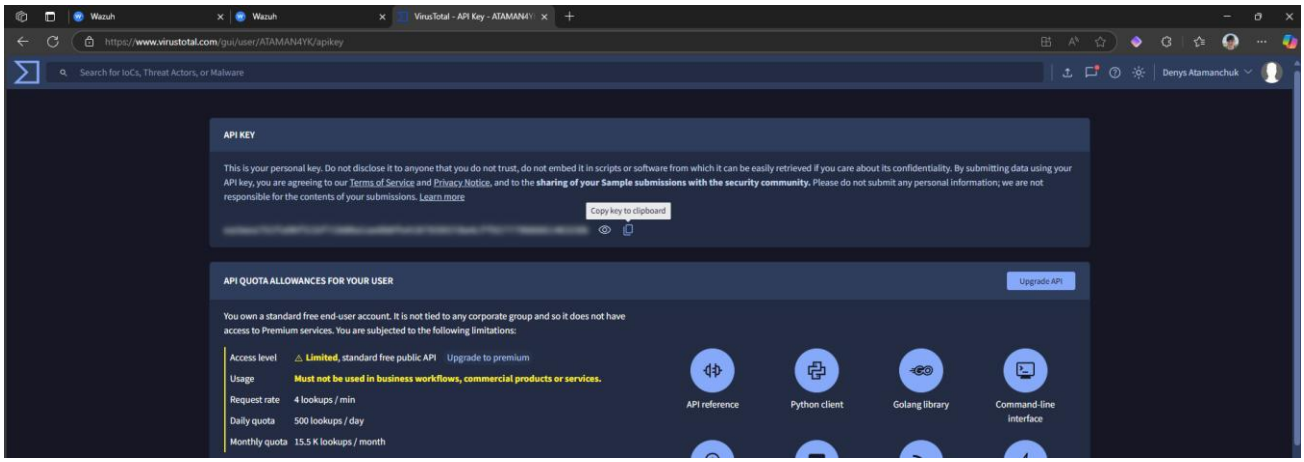


Рисунок 3.36 – Власний API ключ в обліковому записі VirusTotal

Важливо зазначити, що загалом будь які інтеграції Wazuh передбачають наявність відповідних кастомних або офіційно наданих скриптів які опрацьовують запити та відповіді. У випадку інтеграції з VirusTotal – Wazuh офіційно надає готовий скрипт для проведення інтеграції, тому необхідності писати власноруч немає.

Після отримання API ключа надалі було додано налаштування інтеграції до конфігураційного файлу Wazuh, а саме /var/ossec/etc/ossec.conf (рис. 3.37).

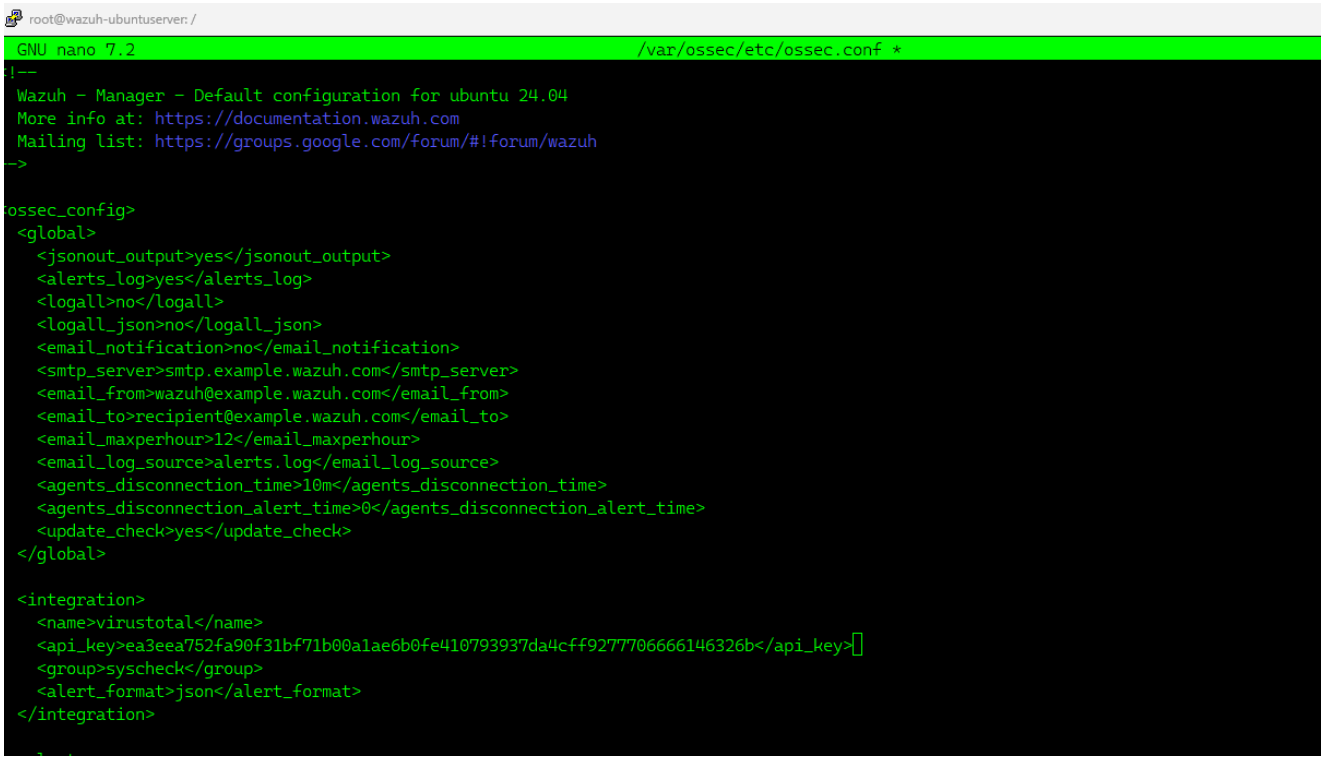


Рисунок 3.37 – Додана інтеграція VirusTotal до конфігураційного файлу Wazuh

З проведених налаштувань можемо побачити що інтеграція спрацьовує на будь які повідомлення правил що перебувають в групі syscheck. Також кожна інтеграція має свою унікальну назву та може мати додаткові поля в залежності від функцій які інтеграція виконує.

3.2.3. Налаштування розширеного збору логів з Microsoft Defender

Microsoft Defender є компонентом захисту від шкідливого програмного забезпечення операційної системи Microsoft Windows. Додаткових збір логів з журналів MS Defender надасть інформацію про загальний стан служби MS Defender, інформацію про результати сканувань, та інформацію про різноманітні спрацювання MS Defender на шкідливе програмне забезпечення на кінцевих точках з ОС Windows.

Для збору журналів MS Defender необхідно налаштувати конфігураційний файл попередньо створеної групи для кінцевих пристроїв з ОС Windows. Було додано новий блок для збору журналів до конфігураційного файлу (рис. 3.38).

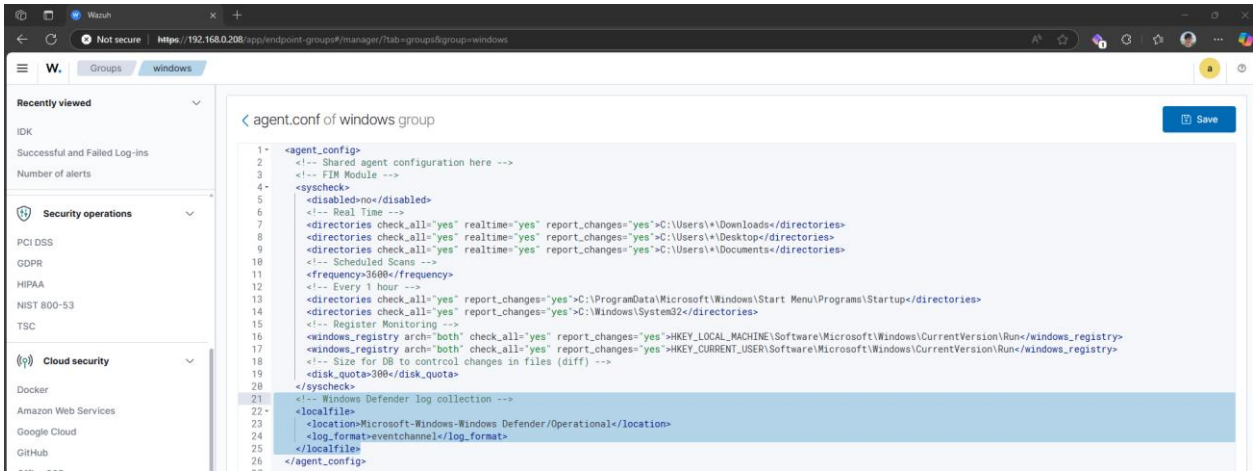


Рисунок 3.38 – Додавання блоку конфігурації для збору журналів з MS Defender

За допомогою тега location вказується розташування журналу або логів які будуть зібрані. В свою чергу тег log_format вказує форматування зібраних логів, у випадку системних логів Windows використовується формат eventchannel, що означає, що дані будуть зчитуватися безпосередньо через API Windows Event Log, а не як текстовий лог файл.

Після збирання логів відповідного формату вони мають оброблятися спеціальними декодерами, які дістають необхідні поля. Створення правил залежить від полів які були зібрані декодерами логів.

У випадку збирання журналів MS Defender, система Wazuh має вже попередньо створені декодери та правила для обробки відповідних подій безпеки. (рис. 3.39 – рис. 3.40). Це значно спрощує інтеграцію, оскільки не потрібно створювати власні декодери та правила вручну.

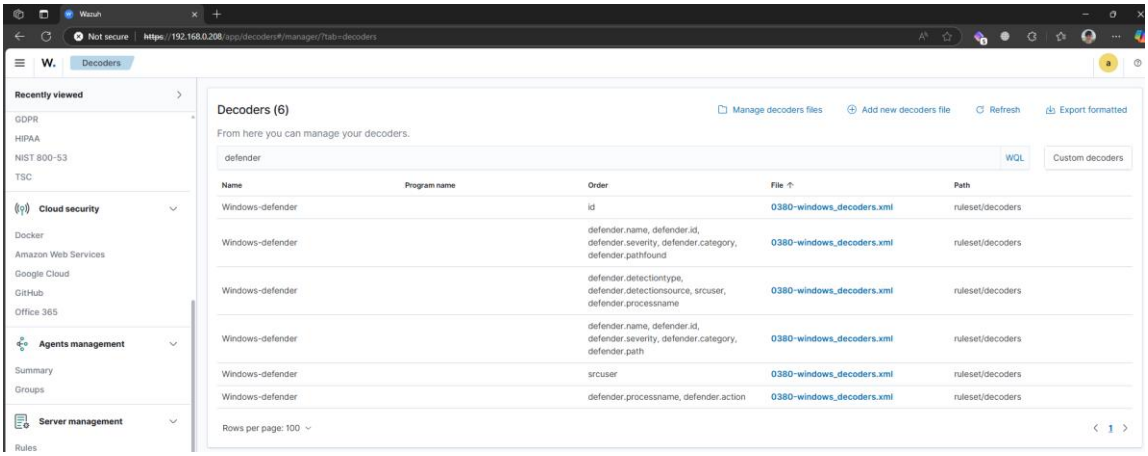


Рисунок 3.39 – Вбудовані файли з декодерами для журналів MS Defender

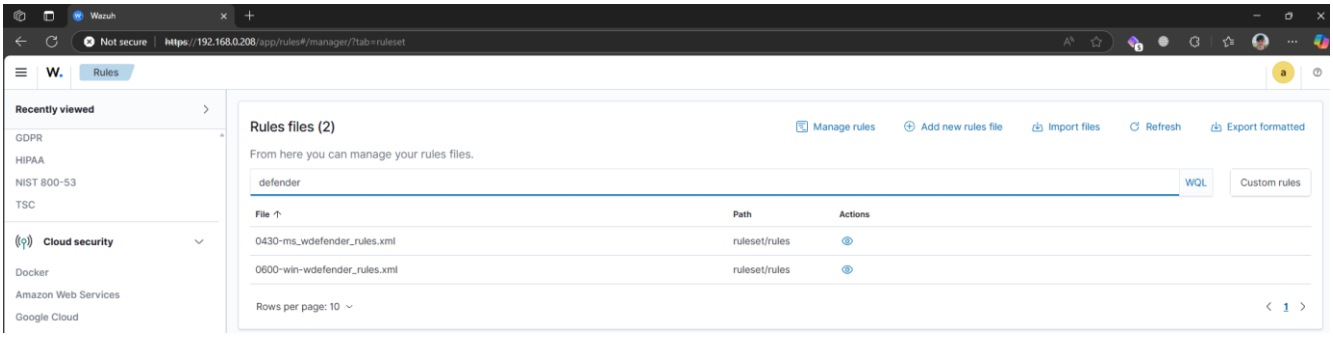


Рисунок 3.40 – Вбудовані файли з правилами для журналів MS Defender

3.2.4. Налаштування виявлення виконання шкідливих команд на ОС Windows та Linux з використанням Sysmon та AuditD

Виявлення виконання передбачає в першу чергу збір логів або журналів з відповідною інформацією з необхідний систем. У випадку операційної системи Windows таким інструментом є Sysmon, який є частиною набору Sysinternals і дозволяє отримувати детальні журнали про створення процесів, зміну файлів, підключення до мережі тощо. Для операційних систем Linux аналогічну функціональність забезпечує AuditD демон аудиту, який інтегрується з ядром Linux і дозволяє відстежувати системні виклики, зміни прав доступу та запуск команд.

Аналогічно до налаштування інтеграції з MS Defender, тут теж необхідно правильно обробити вхідні дані. Для цього вже існую створені декодери та базові правила, які спрощують роботу.

Після обробки отриманих даних нам необхідно буде створити спеціальний список шкідливих команд який буде знаходитися в конфігурації Wazuh та надалі створити кастомні правила для надсилання попереджень по виконання потенційно шкідливих команд.

Для початку було встановлено AuditD на Linux та Sysmon на Windows (рис. 3.41 – рис. 3.42). Крім того було проведено базове налаштування AuditD для збору відповідних логів. В свою чергу Sysmon вже вимагав повноцінної окремої конфігурації для збору необхідних логів, відповідну конфігурацію було імпортовано з документації.

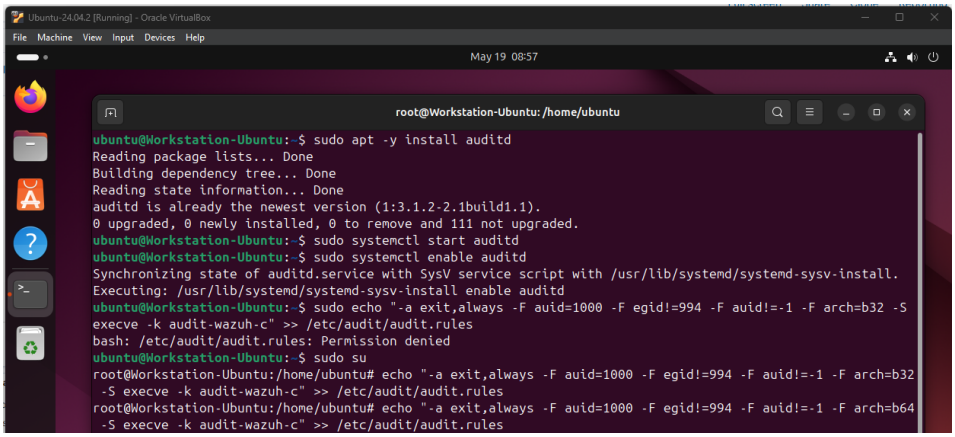


Рисунок 3.41 – Встановлення та базове налаштування AudiD на Linux

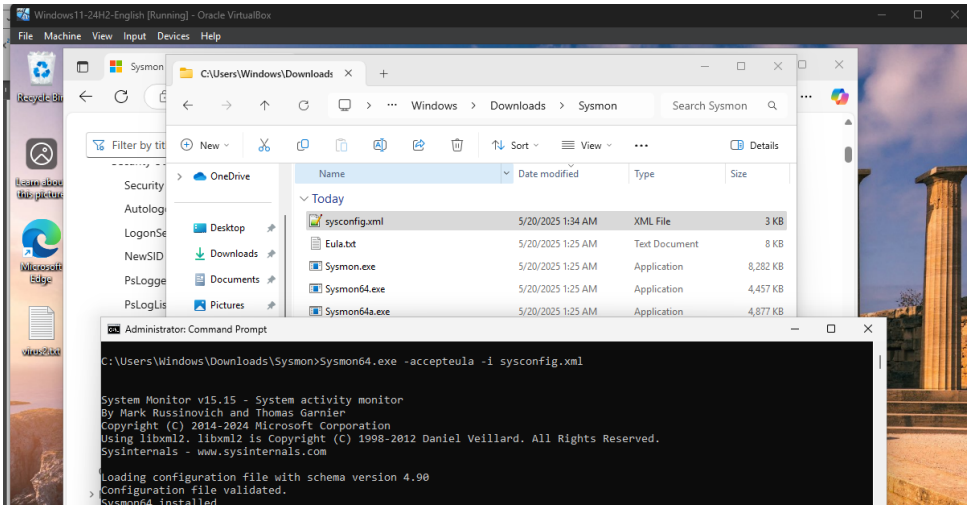


Рисунок 3.42 – Встановлення Sysmon з використанням sysconfig.conf на Windows

Надалі було додано відповідні блоки для збору логів з AuditD та Sysmon до конфігураційних файлів відповідних груп для операційних систем (рис. 3.43 – рис. 3.44).

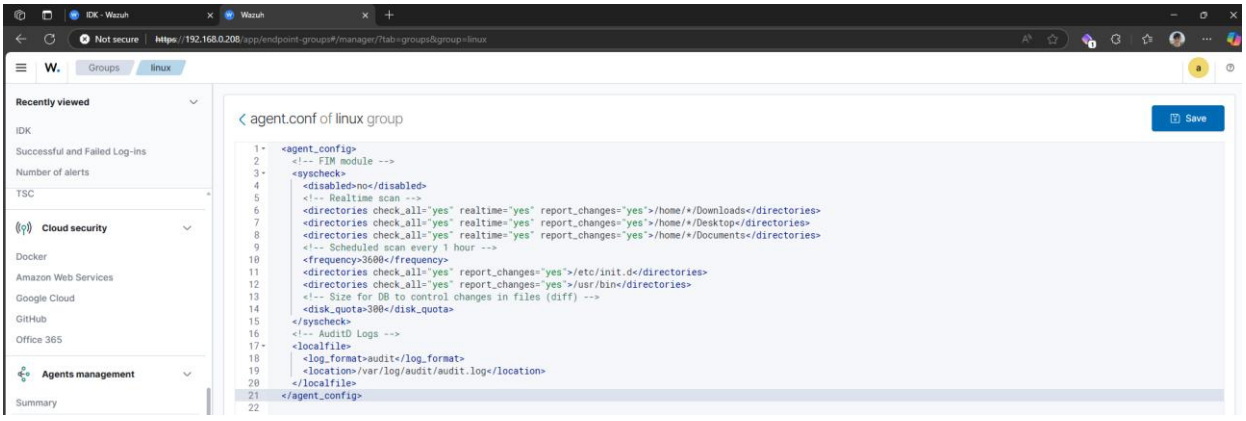


Рисунок 3.43 – Додавання блоку конфігурації для збору журналів з AuditD

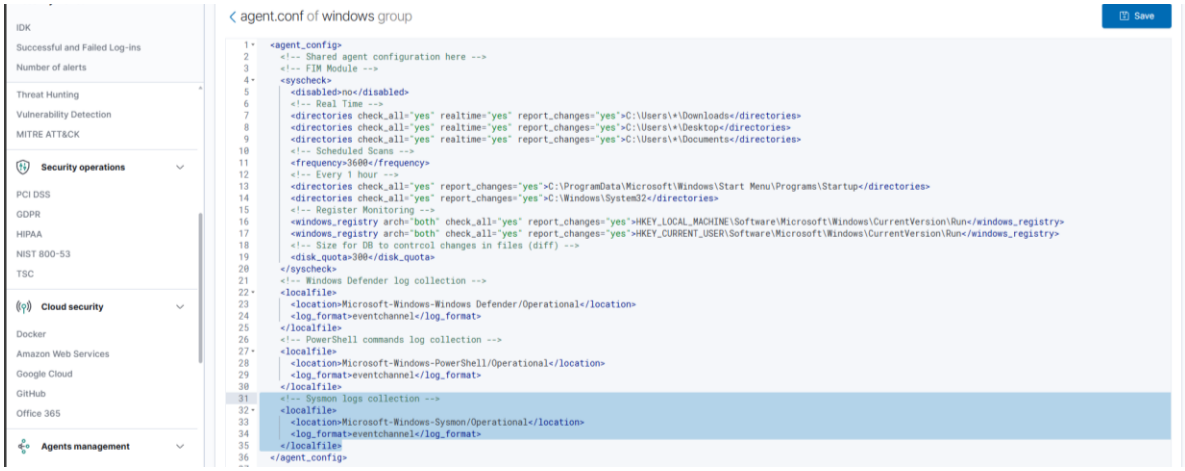


Рисунок 3.44 – Додавання блоку конфігурації для збору журналів з Sysmon

Декодування зібраної інформації за допомогою AuditD відбувається попередньо створеним декодером 0040-auditd_decoders.xml. Також деяка кількість вже створених правил знаходиться в 0365-auditd_rules.xml. Попередньо створені базові правила надають можливість одразу створювати власні більш складні без створення базових для виявлення загального вводу якихось команд до кінцевому пристрої.

Декодуванням інформації зібраної Sysmon для Windows займається декодер 0380-windows_decoders.xml, в свою чергу генерацію повідомлень виконує блок правил 0575-win-base_rules.xml та 0595-win-sysmon_rules.xml.

Крім цього, декодер 0595-win-sysmon_rules.xml має певну кількість вже створених правил які можуть виявляти загрози, це також безумовно підвищить рівень безпеки (рис. 3.45).

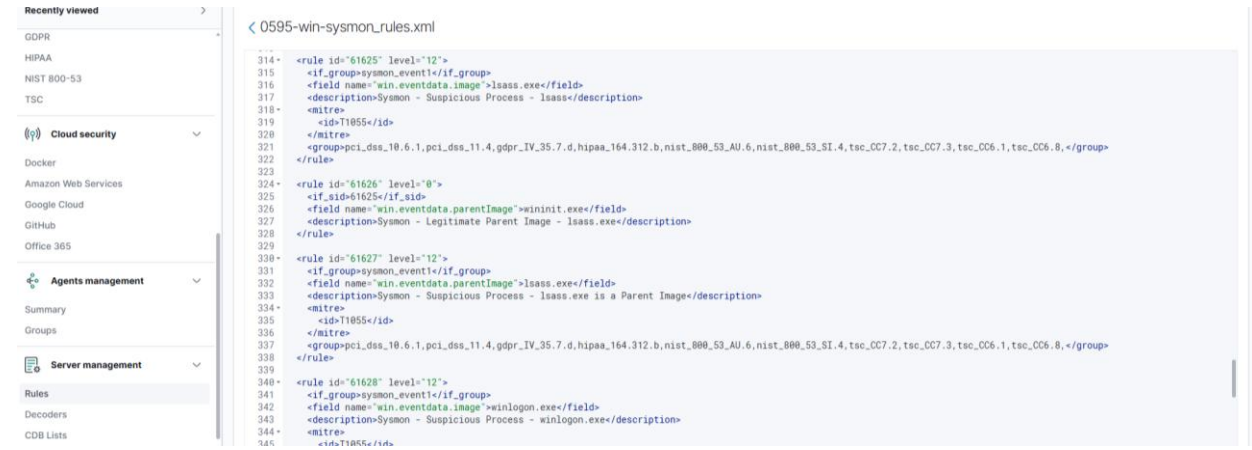


Рисунок 3.45 – Правила в 0595-win-sysmon_rules.xml які здатні виявляти загрози

Після успішно налаштування збору необхідної інформації та декодування необхідно створити кастомні правила які будуть спрацьовувати на потенційно небезпечні або підозрілі команди.

Було створено унікальні списки в директорії /var/ossec/etc/lists/ з командами які можуть використовувати зловмисниками для отримання певної інформації, доступу до системи або для виконання інших зловмисних дій (рис. 3.46 – рис. 3.47).

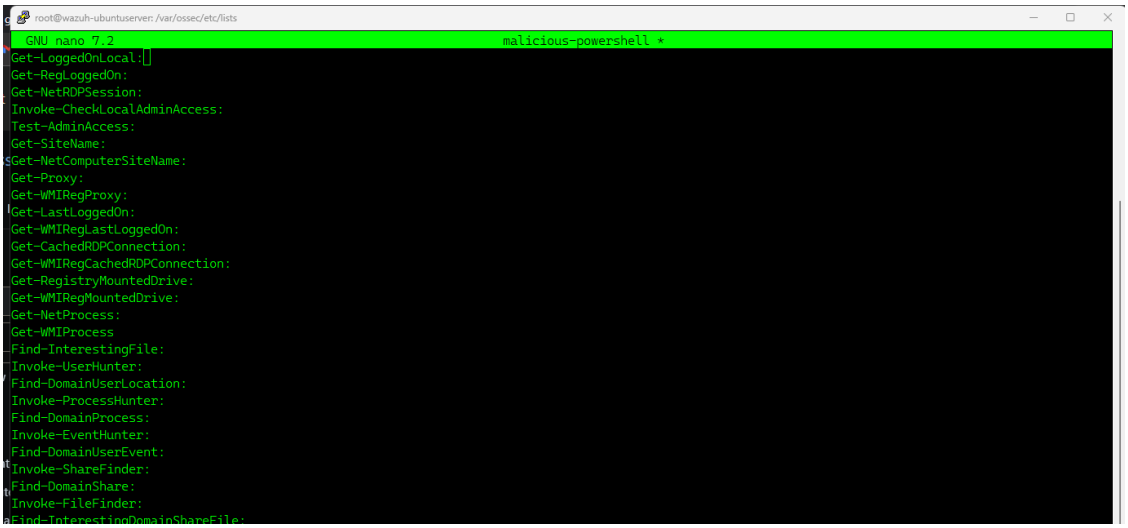


Рисунок 3.46 – Невелика частина створеного списку шкідливих команд powershell

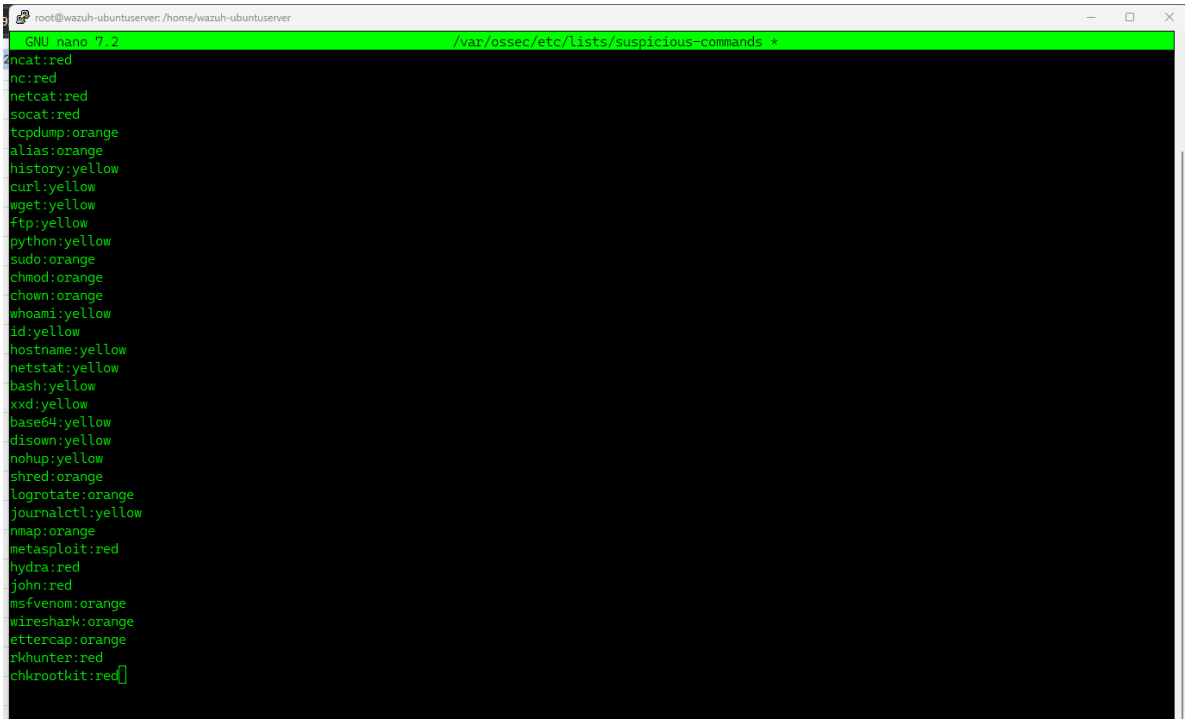


Рисунок 3.47 – Списку шкідливих команд для ОС Linux

Варто зауважити, що обидва списки створені у форматі “ключ:значення”, це надає змогу вказувати власні додаткові значення, за допомогою яких можна ідентифікувати рівень шкідливості використаної команди. У випадку списку команд для powershell всі поля значень пусті, адже всі ці команди являються небезпечними однаково.

Надалі було додано обидва створених списки до загального конфігураційного файлу Wazuh /var/ossec/etc/ossec.conf в спеціальний блок для правил ruleset, це надає змогу використовувати створені списки для кастомних правил (додаток А).

Кінцевим етапом було створення відповідних кастомних правил в local_rules.xml для надсилання повідомлень різних рівнів про використання потенційно шкідливих або шкідливих команд. Для команд які були введенні на ОС Linux було визначено три рівні: підозріла команда, доволі підозріла та високо підозріла команда або небезпечна команда. В свою чергу для Windows було визначено всі команди як шкідливі (рис. 3.48 – рис. 3.49).



Рисунок 3.48 – Власні правила для спрацювання на підозрілі команди на Linux



Рисунок 3.49 – Власне правило для спрацювання на шкідливі команди Powershell

Також після створення списків, додавання правил та внесення інших змін щоразу було проведено перезавантаження Wazuh Manager (Server), це необхідно робити щоразу для застосування внесених змін.

3.2.5. Налаштування виявлення аномальних мережевих з’єднань

Налаштування виявлення аномальних мережевих з’єднань передбачає знову ж таки збір відповідних логів, їх обробка та спрацювання за допомогою створених власних правил [27].

Функціонал Sysmon надає можливість збирати інформацію про всі мережеві з’єднання включно порти які використовуються на кінцевому пристрої. Крім того, важливим моментом є те що Sysmon може бути встановлений на Linux, що і буде зроблено в подальшому [28].

У випадку ОС MacOS, надійного механізму збору інформації про мережеві підключення немає. Однак будуть додані власні налаштування з виконанням команди для отримання списку мережевих з’єднань з певною періодичність, що надасть змогу захистити кінцевих пристрій від аномальних з’єднань на ОС MacOS.

Для початку було завантажено та встановлено Sysmon для ОС Linux з використанням конфігураційного файлу з документації(рис. 3.50 та 3.51).

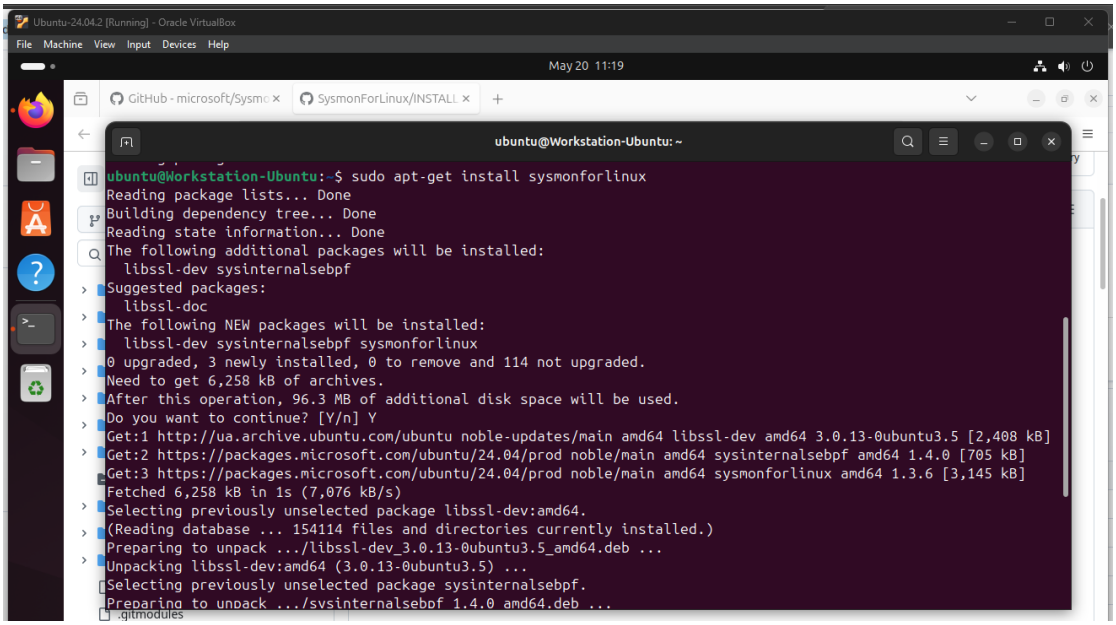


Рисунок 3.50 – Завантаження та встановлення Sysmon на Linux

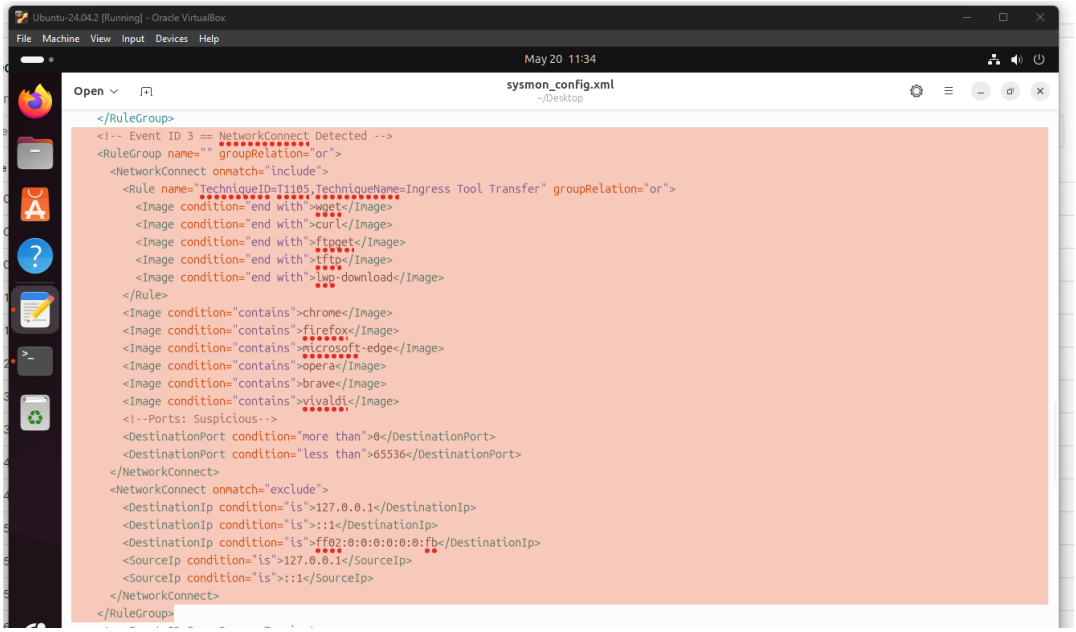


Рисунок 3.51 – Кастомний скрипт з налаштуваннями Sysmon на Linux

Далі було додано відповідний блок налаштувань до конфігураційного файлу групи ОС Linux для збору інформації з Sysmon для Linux (рис. 3.52).

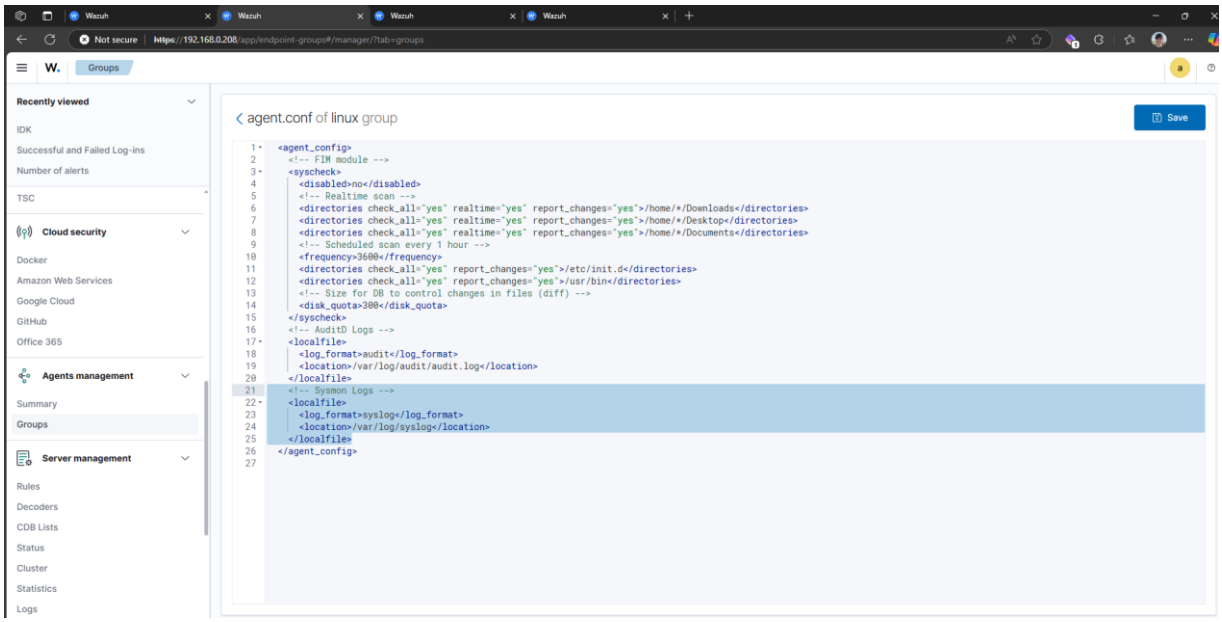


Рисунок 3.51 – Додавання блоку конфігурації для збору журналів з Sysmon

Для декодування та базового опрацювання правилами отриманих логів з журналів Sysmon на Linux були створені та додані власні декодери та правила 200150-sysmon_for_linux_rules.xml та sysmonforlinux-decoders.xml відповідно (рис. 3.53 – рис. 3.54)(додаток Б).

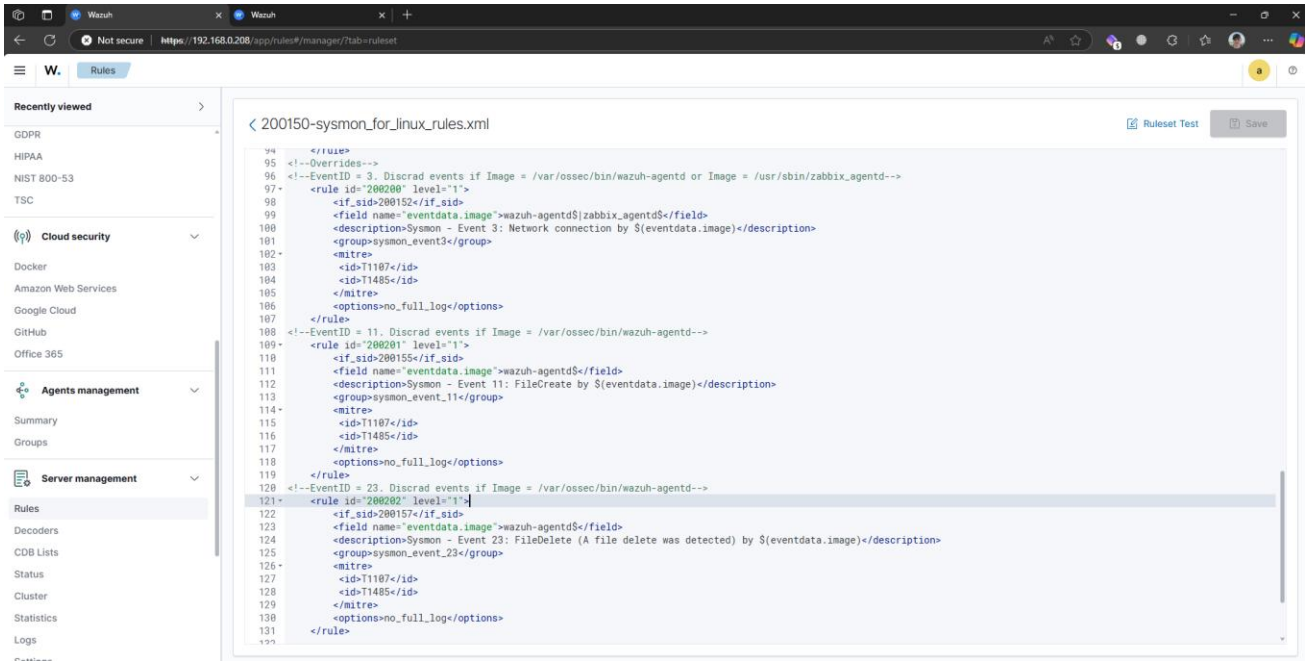


Рисунок 3.52 – Частина власних базових правил для Sysmon на Linux

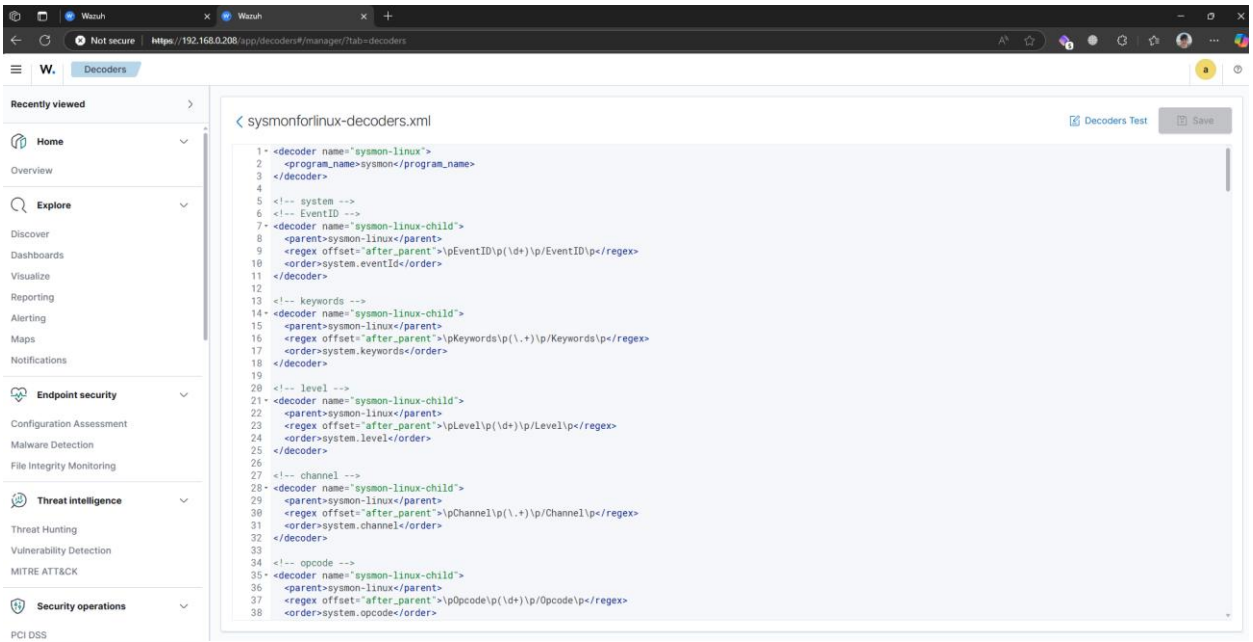


Рисунок 3.53 – Частина власних декодерів для Sysmon на Linux

Стосовно збору інформації про мережеві з’єднання на операційній системі MacOS, були додані власні налаштування для регулярного збору за допомогою команди netstat з фільтрацією TCP та UDP з’єднань. Також була використана команда awk для нормалізації виводу, що надасть змого краще отримувати та обробляти отриману інформацію (рис. 3.54).

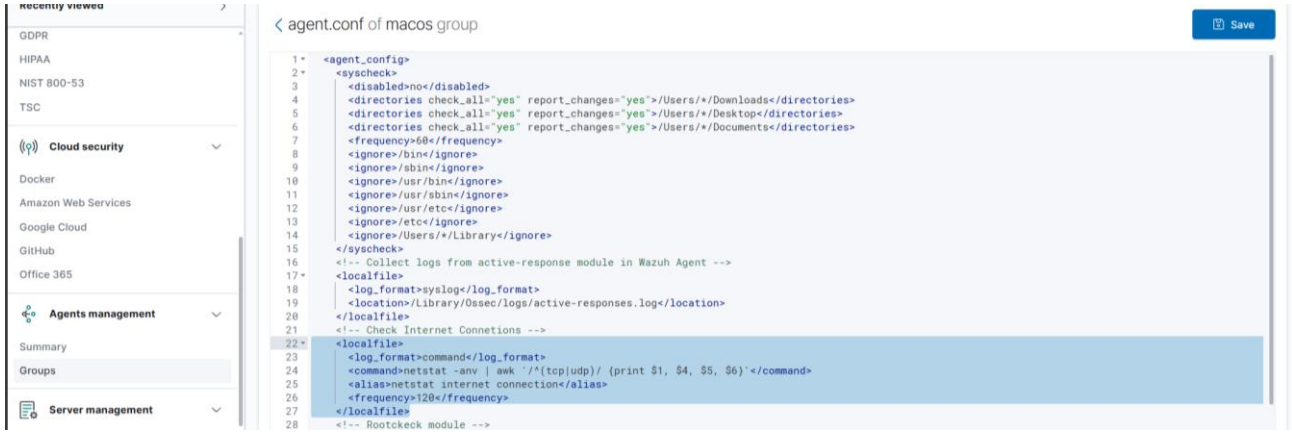


Рисунок 3.54 – Додавання блоку конфігурації для збору інформації про мережеві з’єднання на ОС MacOS

Для декодування отриманої інформації було створено та протестовано власний декодер (рис. 3.55 – рис. 3.56).



Рисунок 3.55 – Декодер для обробки отриманої інформації з netstat на MacOS

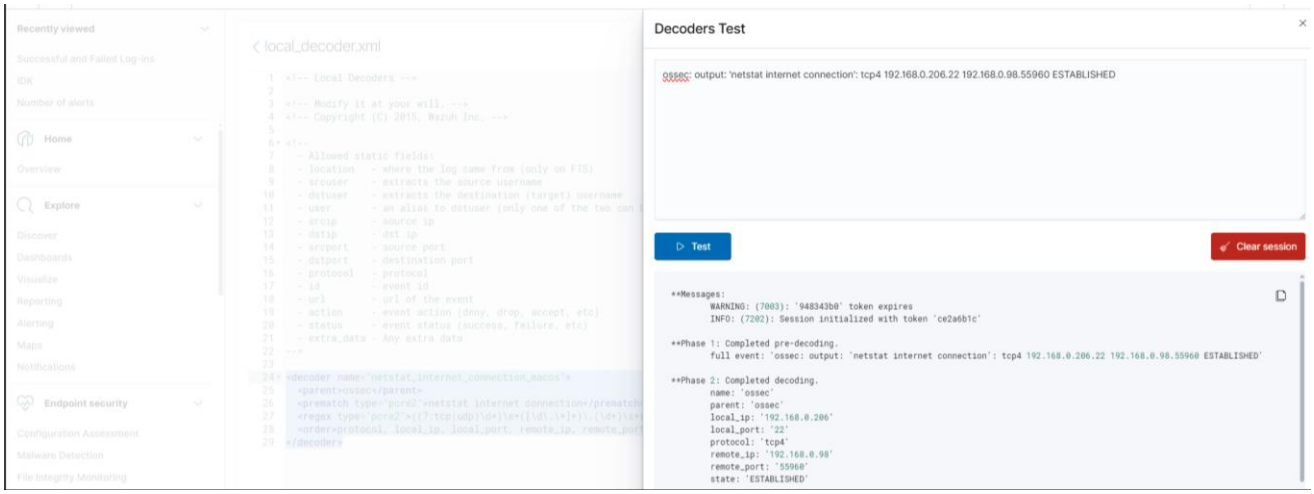


Рисунок 3.56 – Успішне тестування створеного декодеру

Власне етап підготовки був завершений, інформації про мережеві підключення успішно збирається та обробляється з усіх кінцевих пристроїв.

Надалі було створено список в /var/ossec/etc/lists під назвою common-ports, в якому було перелічено список портів які використовуються при повсякденному використані пристроїв, так які як правило є безпечними. Логіка полягає у фільтрації та надсиланні повідомлень які не збігаються з відповідними загальними портами (рис. 3.57).

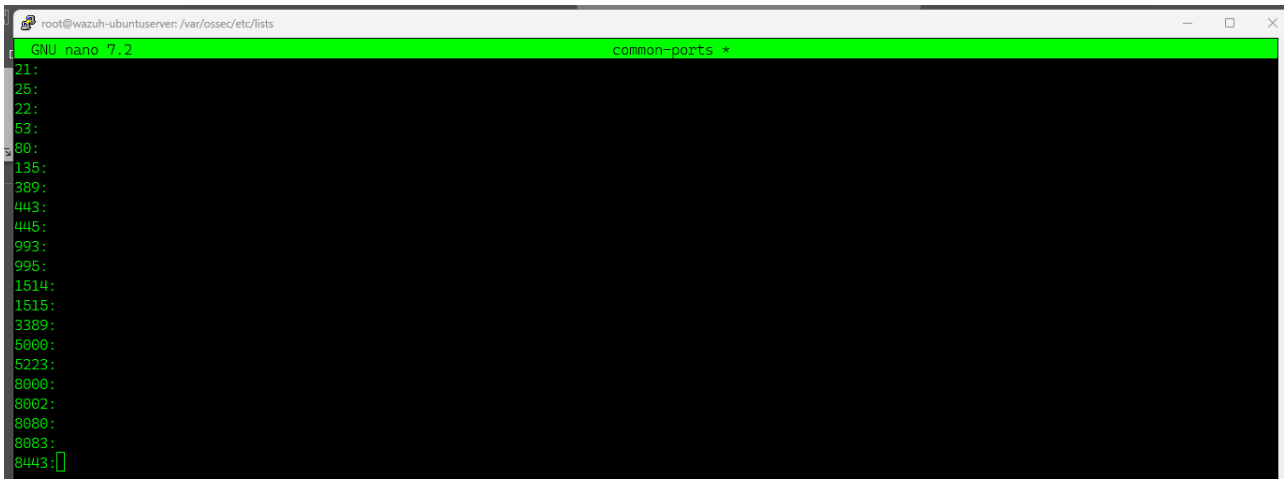


Рисунок 3.57 – Частина створеного списку звичайних портів common-ports

Також було додано створений список до загального конфігураційного файлу Wazuh /var/ossec/etc/ossec.conf в спеціальний блок для правил ruleset, це надасть змогу використовувати створений список для кастомних правил.

На кінець були створені власні правила для відслідковування та надсилання повідомлень про підключення до нестандартних портів (рис. 3.58 – рис. 3.59).

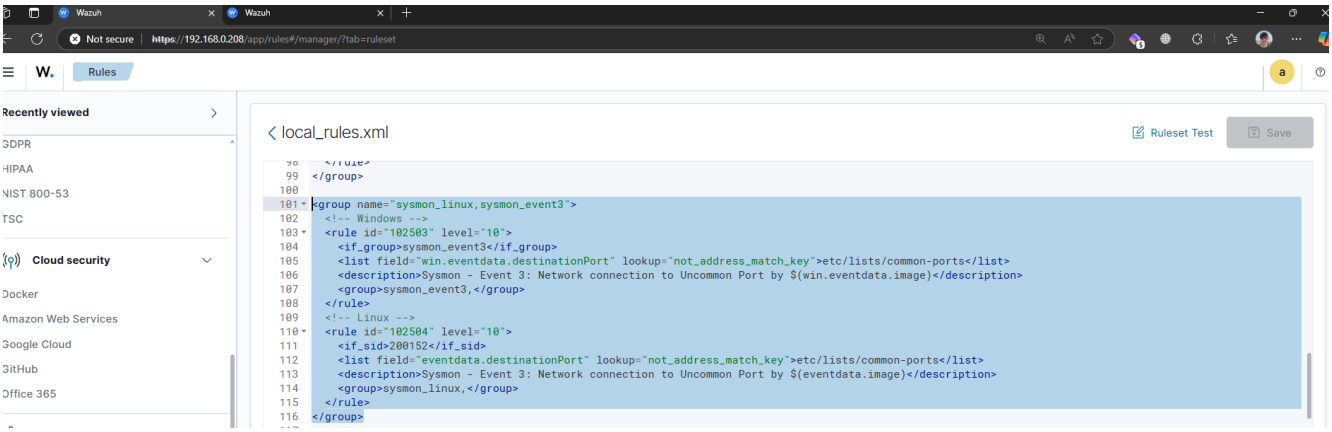


Рисунок 3.58 – Група правил виявлення підключення до нестандартних портів



Рисунок 3.59 – Група правил виявлення підключення до нестандартних портів

Також варто зауважити, що додатково було створено власні правила для відслідковування мережевих з'єднань через браузер та командний рядок (див. рис. 3.59 – рис. 3.60). Ці правила значно підвищать рівень виявлення аномальних мережевих з'єднань, адже зазвичай користувачі не використовують командний рядок для встановлення мережевих з'єднань.



Рисунок 3.60 – Група правил виявлення мережевих з'єднань встановлених за допомогою веб браузера або командного рядка

Відслідковування мережевих з'єднань через веб браузер та командний рядок надасть змогу розширити виявлення підключень до шкідливих веб ресурсів та шкідливих IP адрес за допомогою інтеграції з AbuseIPDB. Це безсумнівно значно посилює безпеку кінцевих пристроїв, адже після виявлення шкідливих підключень можливе реагування у вигляді блокування цих підключень.

Також додатково для всіх кінцевих пристроїв було створено правило для інформування про підключення нових мережевих інтерфейсів (рис. 3.61).



Рисунок 3.61 – Правила для відслідковування мережевих інтерфейсів

3.2.6. Конфігурування інтеграції з сервісом AbuseIPDB

Сервіс AbuseIPDB допомагає системним адміністраторам та аналітикам безпеки перевіряти та повідомляти про IP адреси, які були задіяні в різних категоріях шкідливих атак. AbuseIPDB надає API для перевірки та звітування про IP адресу на наявність шкідливої або зловмисної активності [29].

Для початку було створено особистий обліковий запис на сервісі AbuseIPDB та згенеровано API ключ для інтеграції з Wazuh (рис. 3.62).

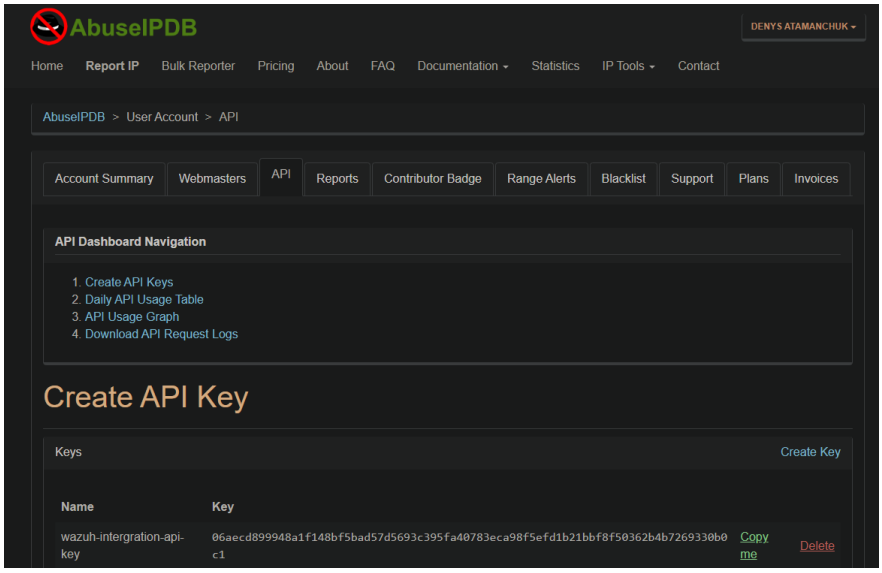


Рисунок 3.62 – Отримання API ключа в особистому обліковому записі AbuseIPDB

Попередньо вже було організовано збір та обробку інформації про мережеві підключення на всіх операційний системах. Передбачається розширений аналіз IP адрес за допомогою сервісу AbuseIPDB отриманих внаслідок попереджень пов’язаних з встановлення мережевих з’єднань через веб браузер або командний рядок.

Після отримання API ключа надалі було додано налаштування інтеграції до конфігураційного файлу Wazuh, а саме /var/ossec/etc/ossec.conf (рис. 3.63).

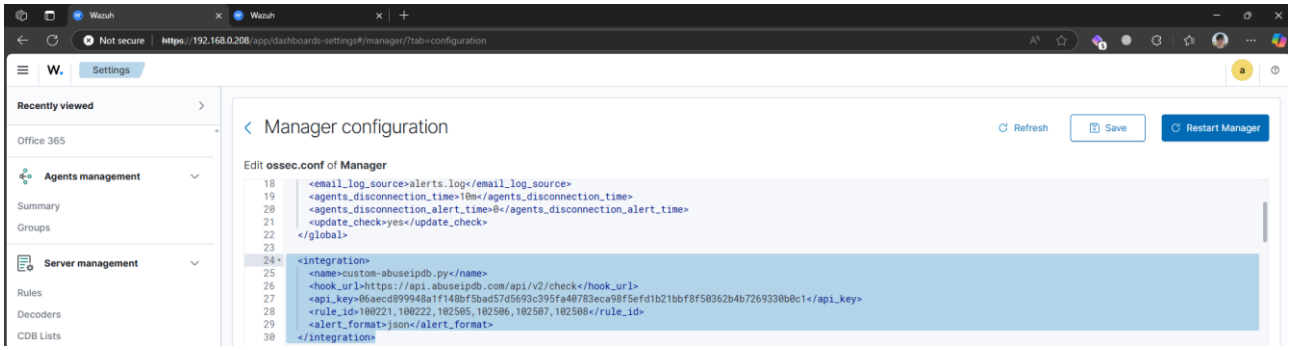


Рисунок 3.63 – Додана інтеграція AbuseIPDB до конфігураційного файлу Wazuh

З проведених налаштувань можемо побачити що інтеграція спрацьовує на повідомлення правил з номерами 100221, 100222, 102505, 102506, 102507 та 102508. Також можемо побачити що інтеграція має поле hook_url в якому вказується посилання на яке будуть надсилатися HTTP запити.

Для інтеграції Wazuh та AbuseIPDB було створено скрипт custom-abuseipdb.py, який отримує дані що надходять з правила спрацювання, обробляє відповідні дані отримуючи необхідну IP адресу. Після чого власний скрипт відправляє запит до сервісу AbuseIPDB для отримання відповіді з рівнем шкідливості певної IP адреси (рис. 3.64)(додаток В).

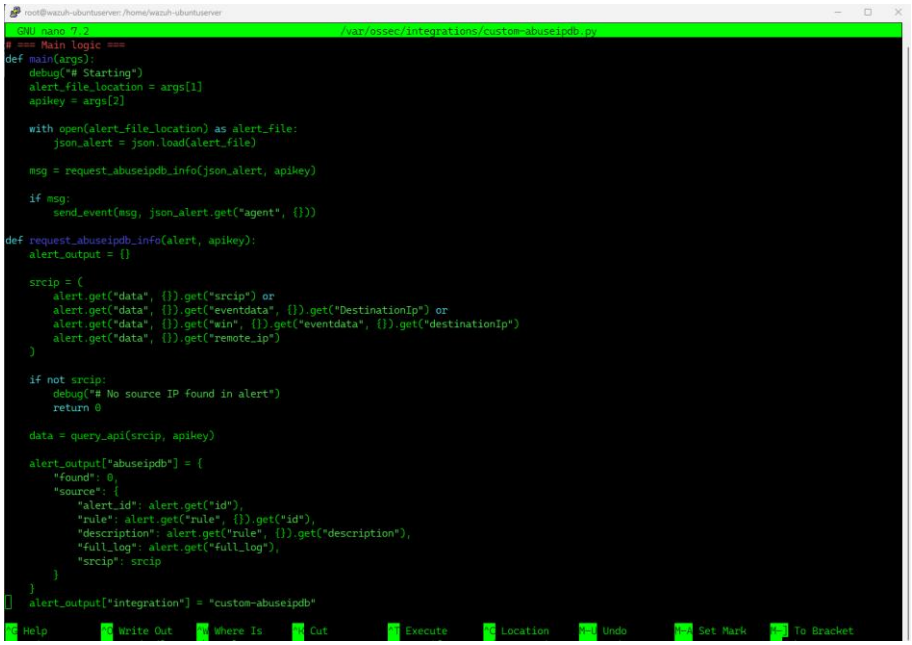


Рисунок 3.64 – Частина скрипта для інтеграції Wazuh з AbuseIPDB

Варто також зазначити, що була розроблена додаткова логіка кешування IP адрес задля уникнення великої кількості запитів до сервісу AbuseIPDB (рис. 3.64). Кешування працює чином зберігання IP адреси у SQLite базі даних протягом одного дня, логіка полягає в тому що рівень небезпеки IP адреси з малою ймовірність зміниться значно за цей період часу. Однак в скрипті також передбачено можливу зміну часу збереження IP адрес.

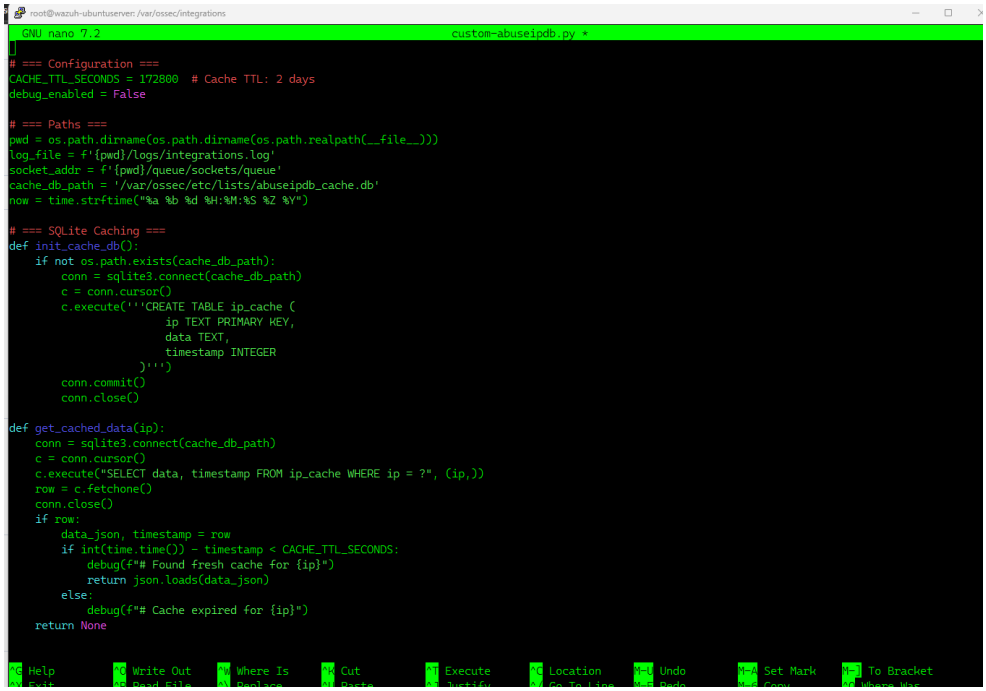


Рисунок 3.65 – Механізм кешування IP адрес в інтеграції з AbuseIPDB

Відповідно після спрацювання правила та отримання повідомлення, скрипт інтеграції в першу чергу перевіряє наявність IP адреси в базі даних SQLite, в разі відсутності надсилає запит, отримує дані та додає IP адресу до бази даних (рис. 3.65).

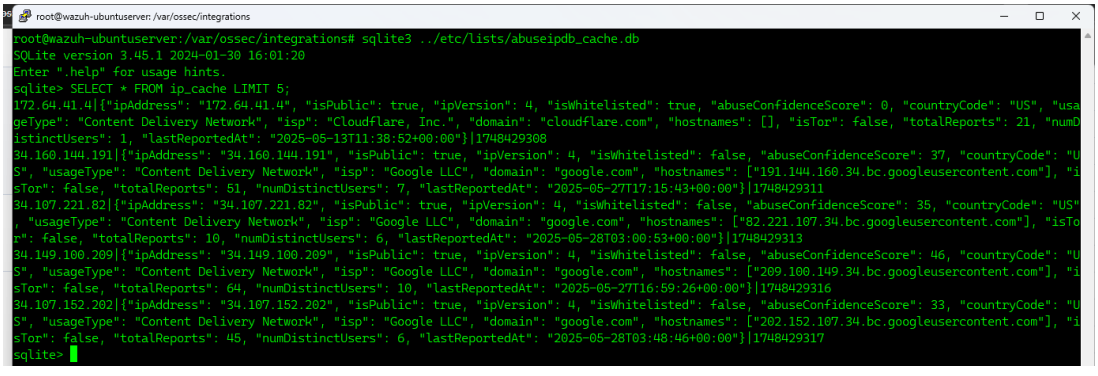


Рисунок 3.66 – Результат кешування IP адрес в інтеграції з AbuseIPDB

Таким чином відбувається значна економія кількості запитів, що дозволяє залишити певну кількість запитів для інших IP адрес, адже AbuseIPDB надає обмеження в 1000 перевірок на день.

Після отримання відповіді, або після отримання інформації про IP адресу з нашої бази даних скрипт створює та надсилає до Wazuh власні івенти з інформацією про результати за допомогою підключення до спеціальних сокетів.

Внаслідок надсилання івентів скриптом для інтеграції Wazuh отримує неопрацьовану але декодовану інформацію до свого індекса wazuh-alerts-*.

Надалі було створено групу власних правил для обробки надісланої інформації внаслідок інтеграції з AbuseIPDB (рис. 3.67)(додаток Г). Правила було розподілено відповідно до власного рівня класифікації небезпеки IP адреси, а саме розподілено на нульовий, низький, середній та високий рівень небезпеки. Таке розподілення надає змогу в подальшому створити активну відповідь у вигляді блокування саме IP адрес з високим рівнем небезпеки.



Рисунок 3.67 – Створена група правил для опрацювання результатів AbuseIPDB

Таким чином інтеграція з сервісом для перевірки IP адрес на шкідливість AbuseIPDB була успішно інтегрована до платформи Wazuh SIEM та XDR.

3.2.7. Налаштування виявлення технік експлуатації Windows PowerShell

Для початку виявлення зловживання PowerShell, було увімкнуто детальне ведення журналу в PowerShell, оскільки Windows за замовчуванням не збирає детальну інформацію про виконані команди в PowerShell через збільшення використання системних ресурсів та вимог до сховища [30] (рис. 3.68).

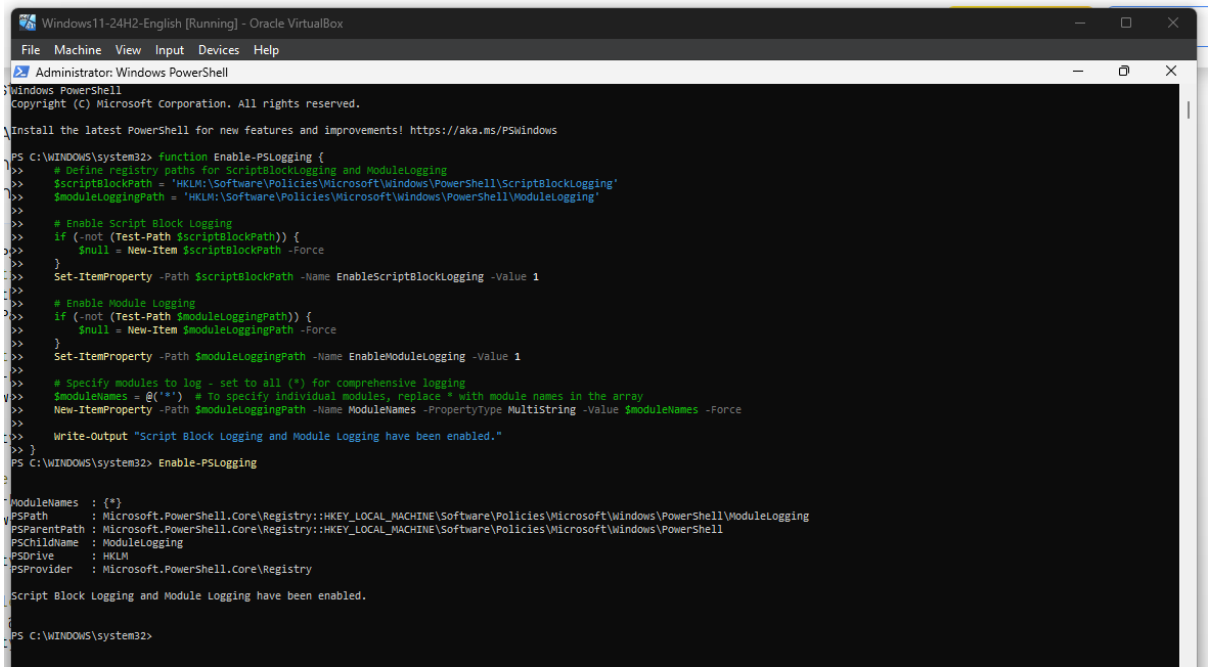


Рисунок 3.68 – Увімкнення детального логування подій Powershell

Було додано новий блок для збору журналів до конфігураційного файлу групи користувачів операційної системи Windows (рис. 3.69).

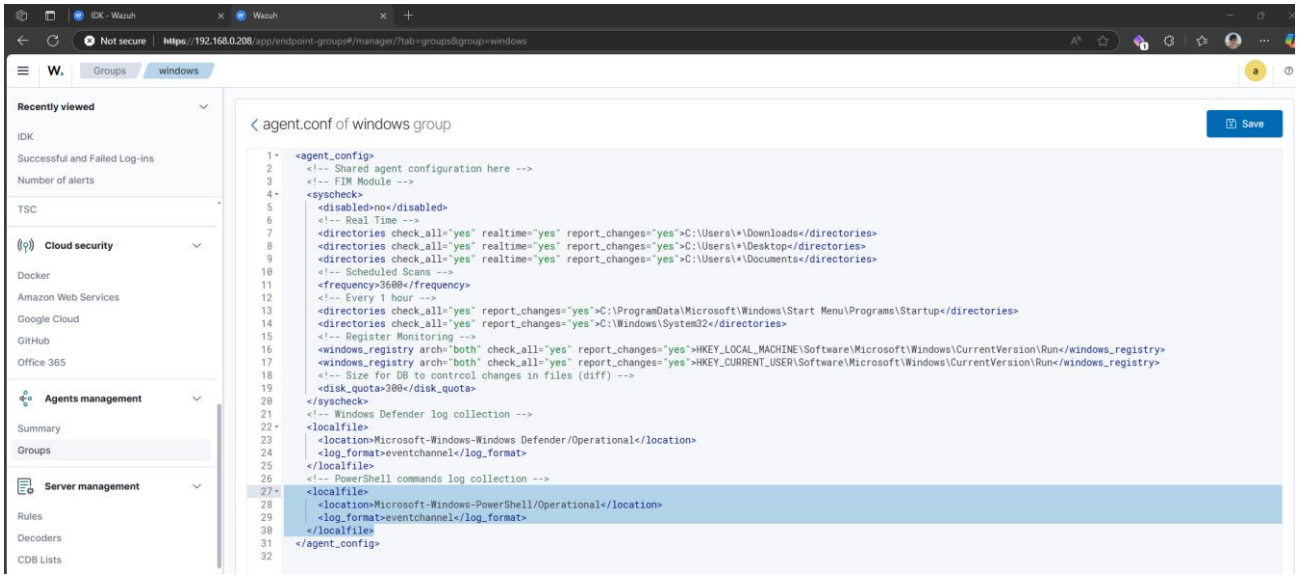


Рисунок 3.69 – Додавання блоку конфігурації для збору журналів з PowerShell

Після налаштування успішного збирання логів було створено групу правил в /var/ossec/etc/rules/local_rules.xml для виявлення потенційних технік експлуатації PowerShell (рис. 3.70).

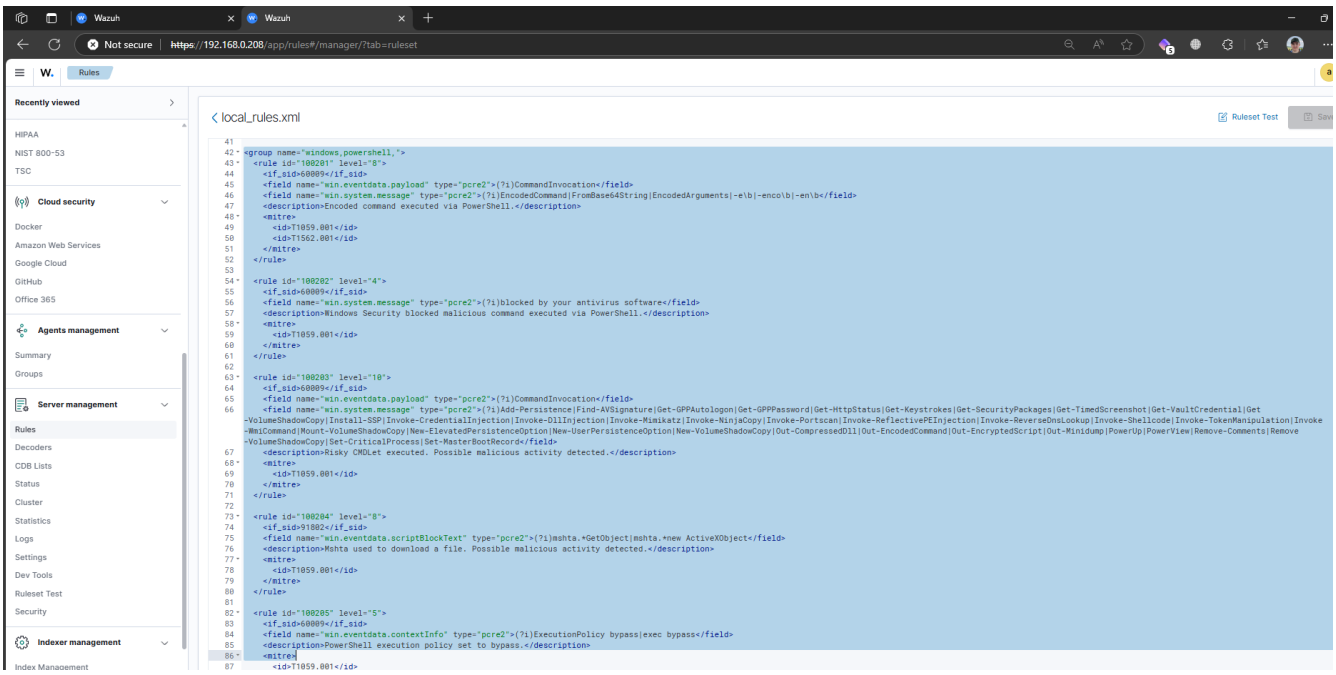


Рисунок 3.70 – Група правил для виявлення технік експлуатації PowerShell

Створені правила розраховані на різноманітні спрацювання, наприклад коли PowerShell команда була виконана у закодованому вигляді, що зазвичай роблять зловмисники для уникнення блокування шкідливого коду.

Також після додавання правил та внесення інших змін було проведено перезавантаження Wazuh Manager (Server), це необхідно робити щоразу для застосування внесених змін.

3.2.8. Налаштування виявлення шкідливого ПЗ типу Rootkit

Wazuh використовує модуль Rootcheck для виявлення аномалій та шкідливого програмного забезпечення типу Rootkit, що можуть свідчити про наявність шкідливого коду на кінцевому пристрої.

За замовчуванням сканування Rootcheck налаштовано на виконання кожні 12 годин, що є завеликим інтервалом для ефективного виявлення шкідливого програмного забезпечення [31].

Для початку було прямо переналаштовано модуль Rootkit в кожному конфігураційному файлі для відповідних груп операційних систем (рис. 3.71)(додаток Д, додаток И та додаток К).

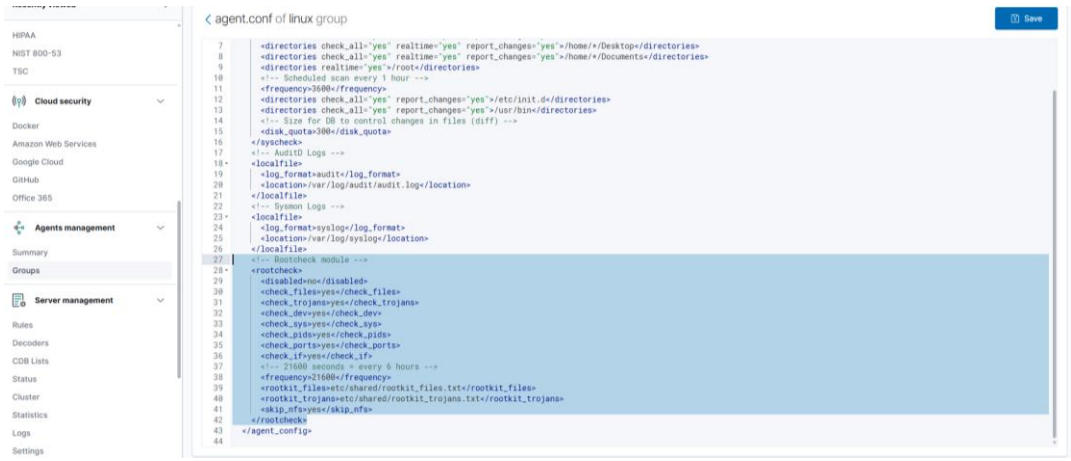


Рисунок 3.71 – Налаштування модуля Rootcheck для кінцевих пристроїв з Linux

Схожим чином було проведено налаштування модуля для всіх конфігураційних файлів відповідних груп операційних систем.

Також було увімкнено додаткові перевірки на наявність шкідливого програмного забезпечення типу Trojan. Крім того, проведені налаштування передбачають сканування всіх файлів в директоріях dev та sys, перевірку портів та ідентифікаторів на предмет експлуатації їх шкідливим ПЗ.

Варто лише зверну увагу на Windows, модуль Rootcheck для ОС Windows працює в доволі обмеженому режимі, тому певні налаштування були вимкнуті (рис. 3.72).

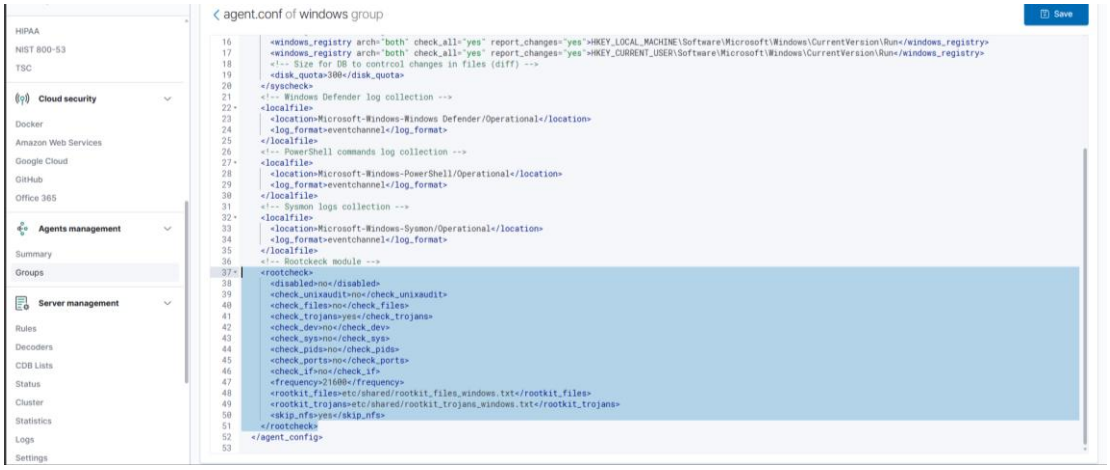


Рисунок 3.72 – Налаштування модуля Rootcheck для кінцевих пристроїв Windows

Важливим етам налаштування модулю Rootcheck було оновлення списку для виявлення ПЗ типу Rootkit. Було імпортовано вагомий список відомих шкідливих файлів які відносяться до ПЗ типу Rootkit (рис. 3.72).

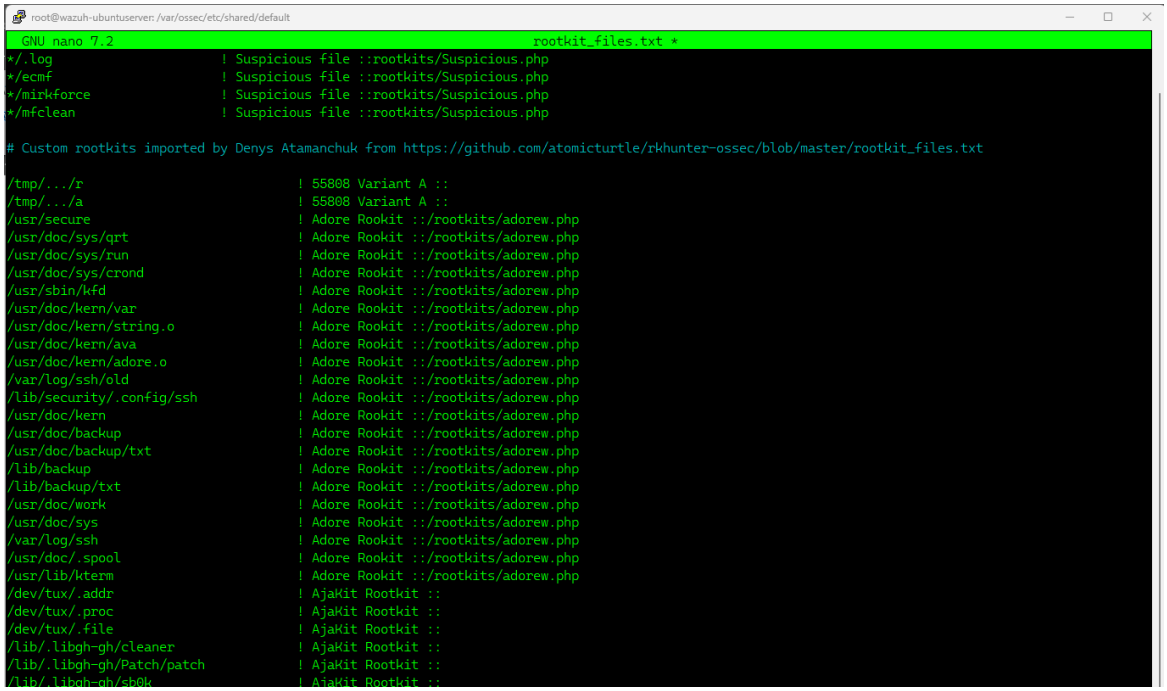


Рисунок 3.72 – Невелика частина оновленого файлу rootkit_files.txt

Таким чином було налаштовано модуль Rootcheck для всіх операційних систем кінцевих пристроїв нашого проєкту. Хоч модуль має обмежений функціонал в контексті ОС Windows, однак це безумовно підвищує рівень безпеки кінцевого пристрою.

3.3. Налаштування механізмів реагування на загрози

Платформа Wazuh SIEM та XDR реалізує свій потенціал розширеної системи виявлення та реагування на вторгнення за допомогою модуля Active Response. Модуль активного реагування допомагає командам безпеки автоматизувати дії реагування на основі певних тригерів, що дозволяє ефективно керувати інцидентами безпеки.

Модуль Active Response виконує скрипти на контрольованих кінцевих пристроях, коли спрацьовує сповіщення про певний ідентифікатор правила, рівень або групу правил.

Скрипти модуля Active Response можуть бути зі збереженням стану, або без збереження стану, це було враховано при написанні кастомних скриптів. Активні відповіді без збереження стану це одноразові дії без визначення події для їх скасування або зупинки. Відповіді зі збереженням стану скасовують або зупиняють свої дії через певний проміжок часу.

3.3.1. Налаштування модуля Active Response для видалення шкідливих файлів виявлених сервісом VirusTotal

Видалення попередньо виявлених шкідливих файлів є критично важливим завданням, адже чим довше шкідливі файли знаходяться на кінцевому пристрої тим більша вірогідність його зараження.

Для видалення шкідливих файлів в першу чергу було додано команди активних відповідей до глобального конфігураційного файлу ossec.conf (рис. 3.73).

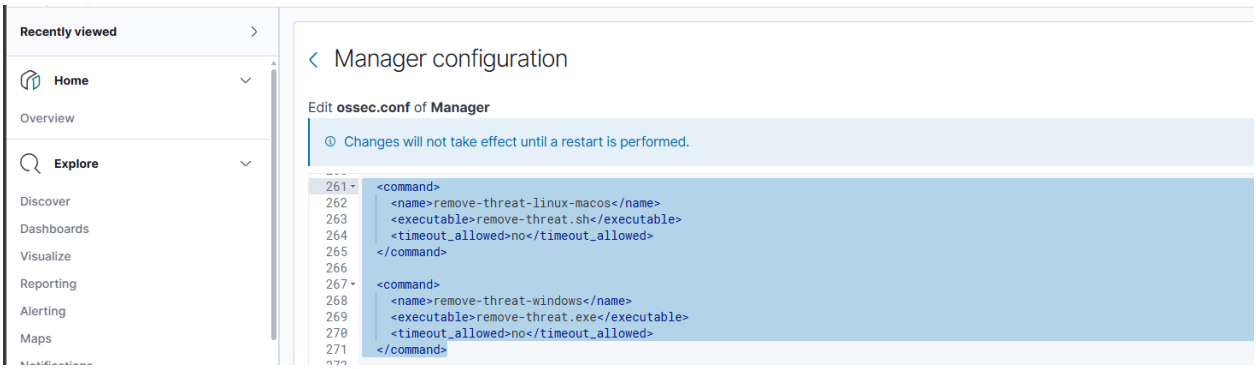


Рисунок 3.73 – Команди активної відповіді для видалення файлів

Створені команди визначають який скрипт буде виконуватися у випадку застосування цих команд, та визначають чи застосування команд передбачає збереження стану, тобто відновлення видалених файлів після видалення через деякий час [32].

Надалі було додано конкретні активні відповіді до загальної конфігурації, які будуть використовувати попередньо створені команди. Крім того при створені активних відповідей був вказаний ідентифікатор попередження 87105, тобто на це попередження буде спрацьовувати активна відповідь (рис. 3.74).

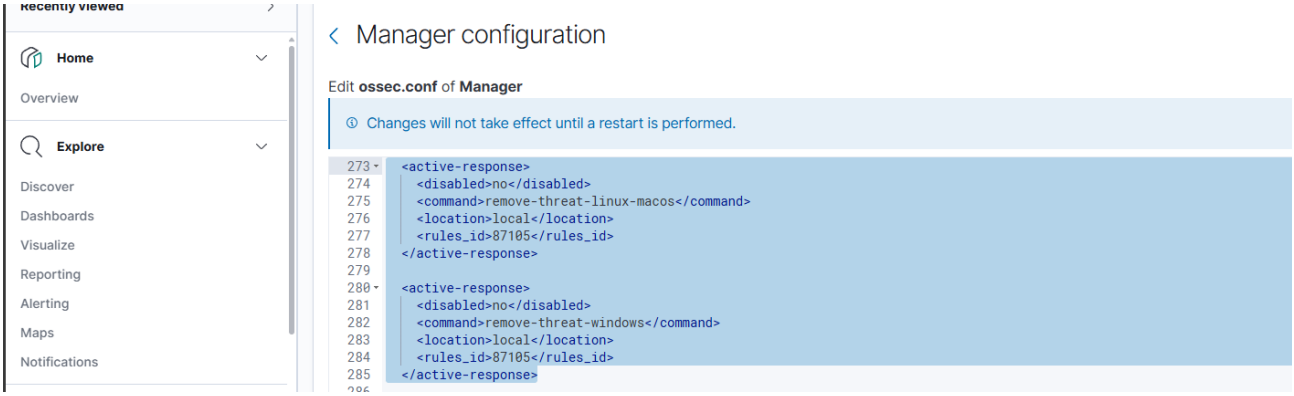


Рисунок 3.74 – Створення активних відповідей для видалення шкідливих файлів

Правило з ідентифікатором 87105 спрацьовує при успішному спрацюванні інтеграції з VirusTotal, та після успішної ідентифікації цим сервісом файлу як шкідливого.

Після створення команд та активних відповідей для видалення шкідливих файлів необхідно створити відповідні скрипти на операційних системах Windows, Linux та MacOS.

Кінцевий пристрій з ОС Linux мав вже попередньо готовий скрипт для видалення файлів, однак він працював не коректно при інтеграції з сервісом VirusTotal. Це зумовлено тим що скрипт для видалення файлів має отримати інформацію від Wazuh Server та знайти необхідні поля з розташуванням файлу, однак інтеграція з VirusTotal змінює розташування певних полів у JSON відповіді. Тому було повністю написано власний скрипт для видалення шкідливих файлів на ОС Linux та ОС MacOS, адже обидві операційні системи мають однакові встановлені скрипти (рис. 3.75)(додаток Л).

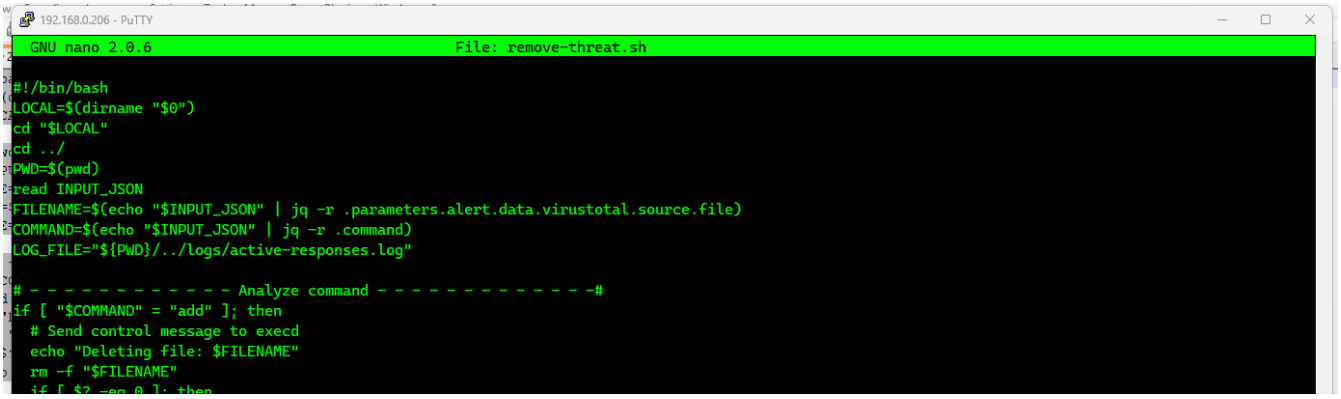


Рисунок 3.75 – Частина власного скрипта для видалення шкідливих файлів

Надалі було створено відповідний скрипт на ОС MacOS. Крім того, власноруч створені скрипти для активних відповідей повинні належати користувачу root та групі Wazuh. Тому для відповідних скриптів були змінені власники та змінені права доступу (рис. 3.76 – рис. 3.77). Також було перезавантажено агенти на кінцевих пристроях для застосування змін.

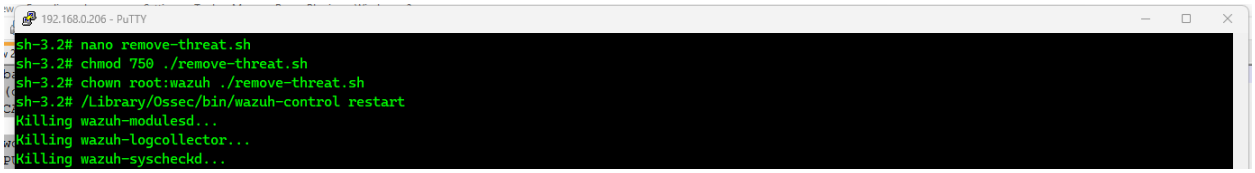


Рисунок 3.76 – Додавання, зміна власників та прав доступу для скрипта на MacOS

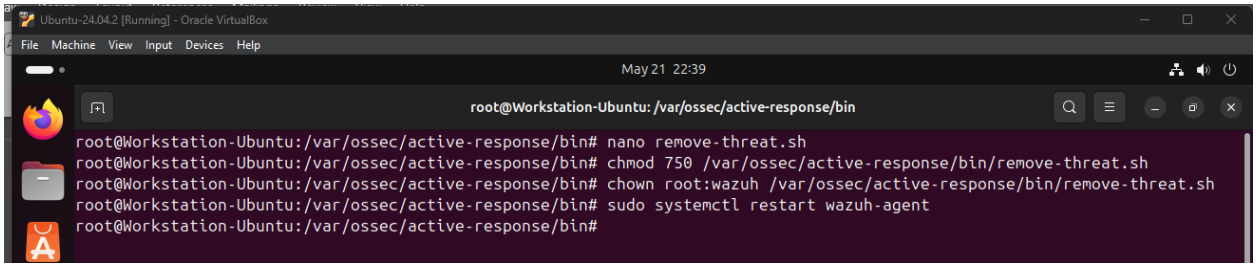


Рисунок 3.77 – Додавання, зміна власників та прав доступу для скрипта на Linux

Говорячи про операційну систему Windows, попередньо встановлених скриптів не надається, тому тут теж було написано власний скрипт для видалення шкідливих файлів на Python (рис. 3.78).

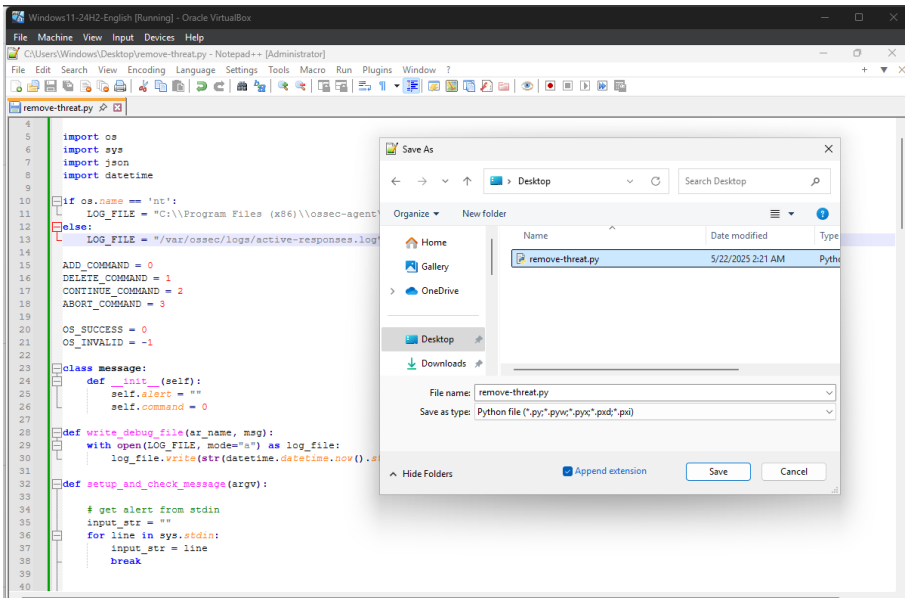


Рисунок 3.78 - Власний скрипт для видалення шкідливих файлів на ОС Windows

Однак потрібно зауважити, що платформа SIEM Wazuh та XDR не має можливості блокування використовуючи скрипти Python на ОС Windows. Внаслідок цього було встановлено та запущено утиліту pyintsaller для компіляції виконуваного файлу з розширенням .exe (рис. 3.79 – рис. 3.80).

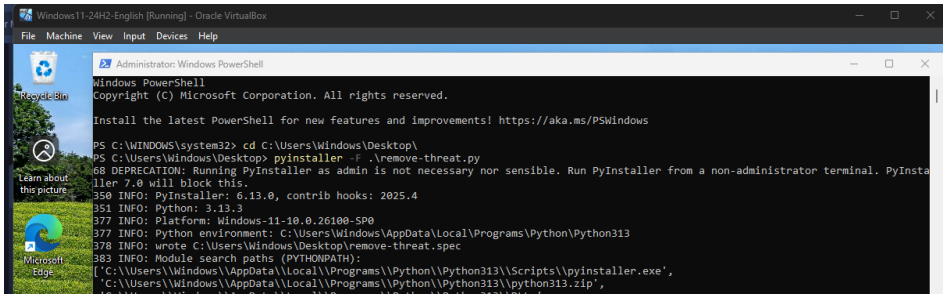


Рисунок 3.79 – Запуск компіляції скрипта для видалення шкідливих файлів

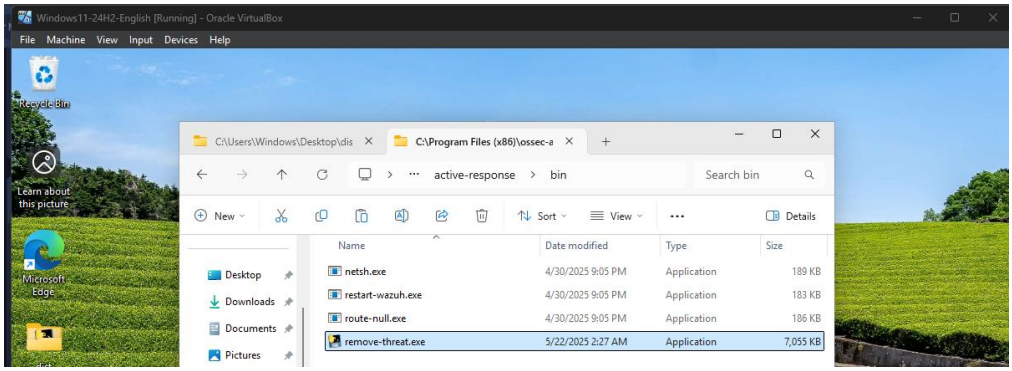


Рисунок 3.80 – Успішно скомпільований та перенесений виконуваний файл

На цьому етапі безпосереднє налаштування активної відповіді для видалення шкідливих файлів є завершеним та успішним. Було лише також додатково створені два правила для генерації інформації про успішний чи негативний результат видалення шкідливих файлів (рис. 3.81).



Рисунок 3.81 – Створені правила для надсилання повідомлень про результати

3.3.2. Налаштування реагування на SSH атаки типу грубої сили

Для початку налаштування реагування на SSH атаки було створено відповідну команду для блокування IP адреси за допомогою Fireawall в загальному конфігураційному файлі ossec.conf [33] (рис. 3.82).

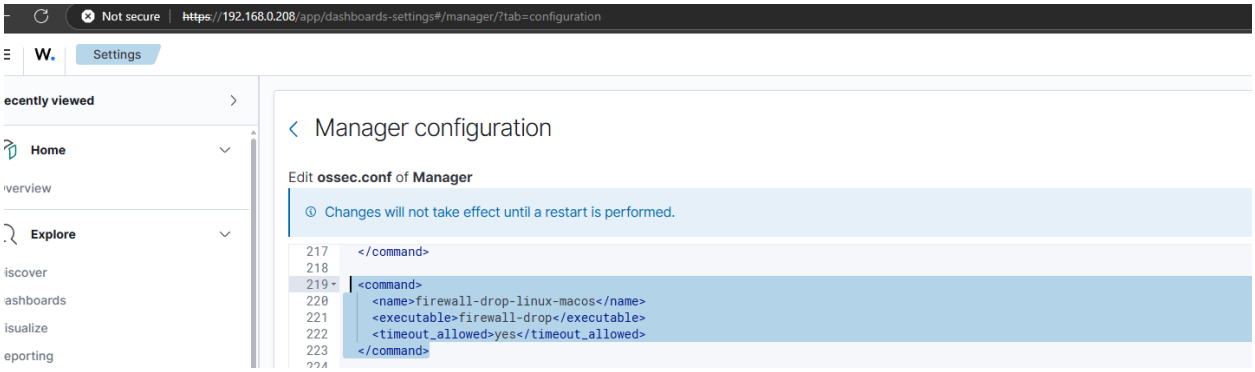


Рисунок 3.82 – Команда активної відповіді для блокування IP адреси

Надалі було додано конкретну активну відповідь до загальної конфігурації, яка буде використовувати попередньо створену команду. Крім того при створені активної відповіді було вказаний ідентифікатор попередження 5763 (рис. 3.83).

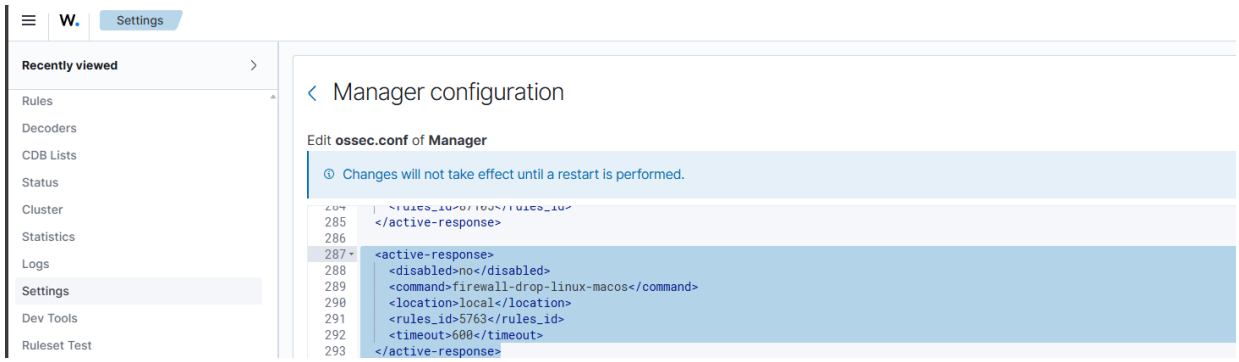


Рисунок 3.83 – Створена активна відповідь для блокування IP адрес

Правило з ідентифікатором 5763 спрацьовує при успішній ідентифікації великої кількості неуспішних спроб підключення до SSH серверу на кінцевому пристрої Linux або MacOS.

Також варто зазначити, що для ОС Windows необхідності створювати відповідну активну відповідь немає, адже за замовчуванням SSH Server не встановляється та не активується на ОС Windows, на відмінну від ОС Linux та MacOS.

Стосовно скрипта активної відповіді який буде блокувати відповідні шкідливі IP адреси. Платформа Wazuh SIEM та XDR надає готові виконувані файли для блокування IP адрес на операційних системах Linux та MacOS, тому в цьому випадку писати власні скрипти необхідності немає (рис. 3.84).

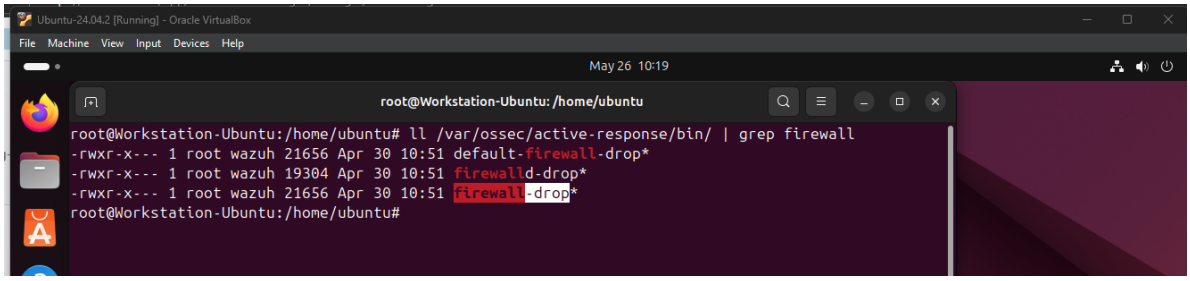


Рисунок 3.84 – Скрипт активної відповіді для блокування IP адрес

Не менш важливим моментом також є увімкнення фаєрволу для успішної роботи скрипта. На ОС Linux за замовчуванням було увімкнено Uncomplicated Firewall, що працює на основі iptables. На ОС MacOS за замовчуванням Firewall є вимкненим, тому його було увімкнуто (рис. 3.85).

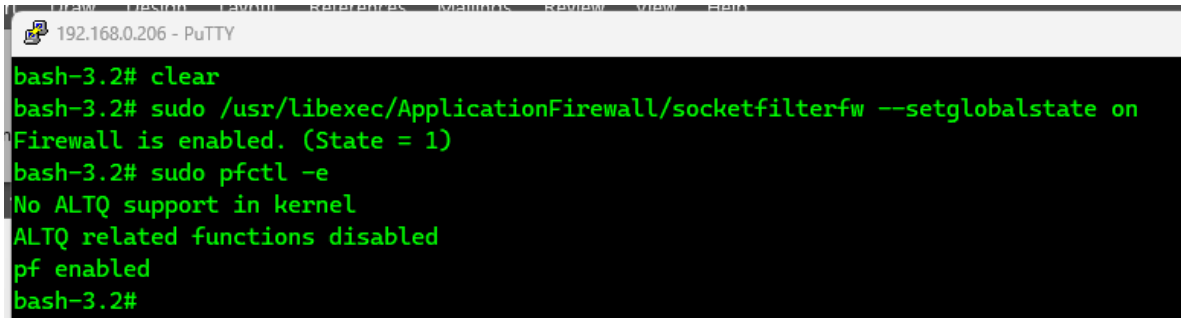


Рисунок 3.85 – Увімкнення Firewall на MacOS

Після проведення попередніх налаштувань активна відповідь на спроби грубої сили на кінцевих пристроях з операційною системою Linux та MacOS є успішно налаштованою.

3.3.3. Налаштування блокування шкідливих IP адрес виявлених сервісом AbuseIPDB та внаслідок аномальної мережевої активності

Блокування шкідливих IP адрес виявлених за допомогою сервісу AbuseIPDB має відбуватися за допомогою власних скриптів для блокування, адже надані

скрипти в ОС Linux та MacOS не налаштовані для блокування IP адрес, що надходять з сервісу AbuseIPDB.

Відповідь яку надсилає сервіс AbuseIPDB до скрипта активного реагування містить додаткові спеціальні поля які необхідно окремо опрацювати та додати до відповіді яку надішле скрипт після спрацювання.

Аналогічно до інших активних відповідей було створено команди активної відповіді для всіх операційних систем та було створено самі активні відповіді в загальному конфігураційному файлі ossec.conf (рис. 3.86 – рис. 3.87).



Рисунок 3.86 – Додавання команд активної відповіді для AbuseIPDB



Рисунок 3.87 – Додавання активних відповідей для AbuseIPDB

Активні відповіді були налаштовані таким чином що би спрацьовувати лише на ідентифіковані IP адреси сервісом AbuseIPDB як шкідливі високого рівня, тобто рівня шкідливості від 70 до 100 включно.

Після налаштування команд та активних відповідей необхідно створити кастомні скрипти для блокування IP адрес отриманих з сервісу AbuseIPDB. Для всіх операційних систем було створено власні унікальні скрипти (рис. 3.88, рис. 3.89 – рис. 3.90)(додаток М, додаток Н та додаток П).

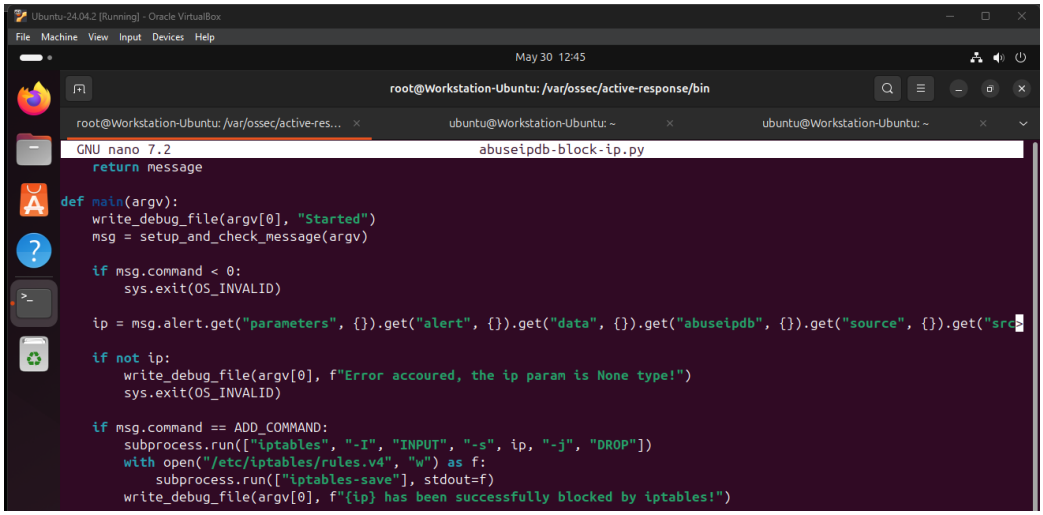


Рисунок 3.88 – Частина власного скрипта для Linux

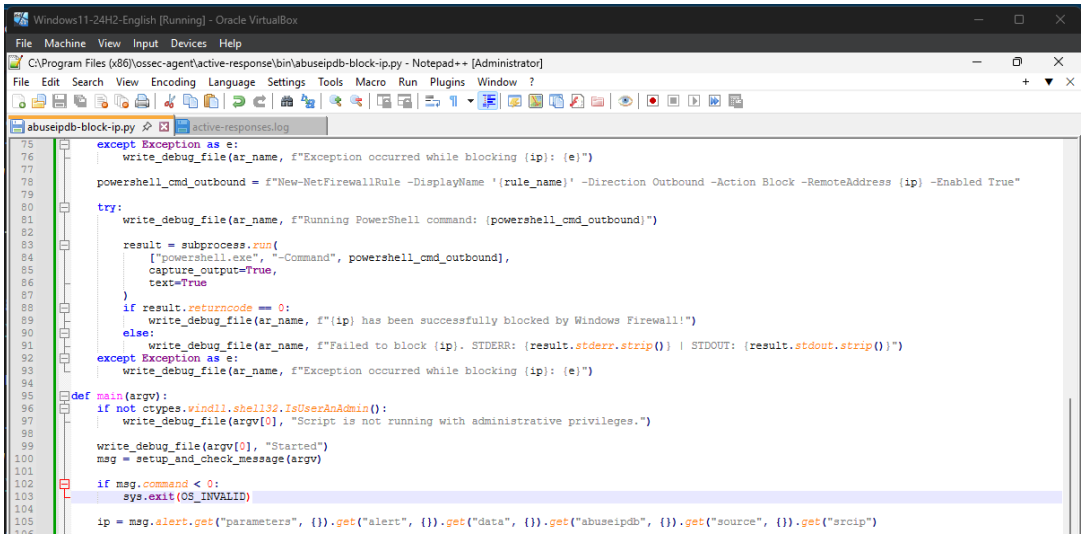


Рисунок 3.89 – Частина власного скрипта для Windows

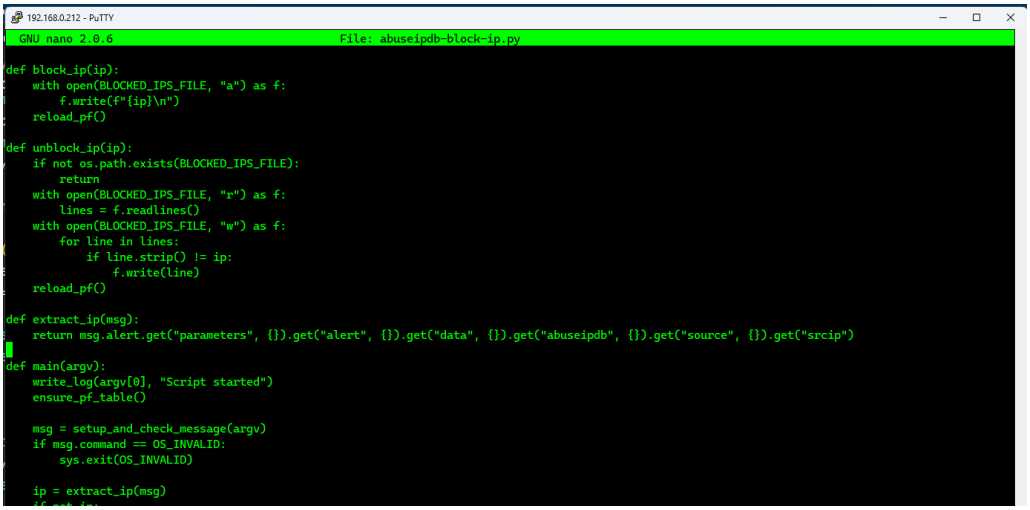


Рисунок 3.90 – Частина власного скрипта для MacOS

Також варто зауважити що скрипт на операційній системі Windows був скомпільований у виконуваний .exe файл, так як платформа Wazuh SIEM та XDR не підтримує виконання Python скриптів на ОС Windows (рис. 3.91).

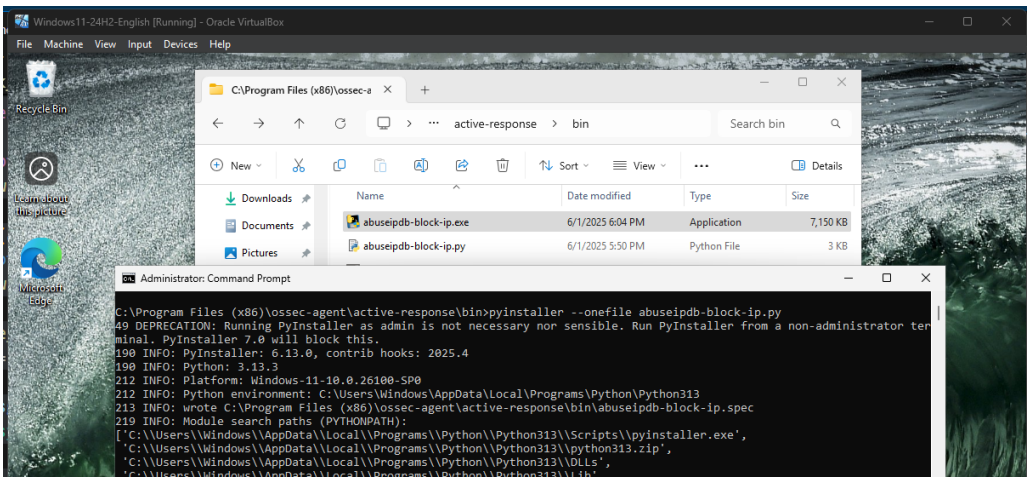


Рисунок 3.91 – Скомпільований скрипт активної відповіді на ОС Windows

Надалі були проведені додаткові налаштування для коректної роботи екранів безпеки після перезавантажень також проведені налаштування з додавання так званих білих списків IP адрес для деяких правил, для уникнення хибно позитивних спрацювань.

Було також налаштовано відповідних власника root та групу власників Wazuh, також було налаштовано права на виконання створених скриптів.

Після проведення основних налаштувань було також додатково створено правило для надсилання повідомлень про успішне блокування IP адрес (рис. 3.92).

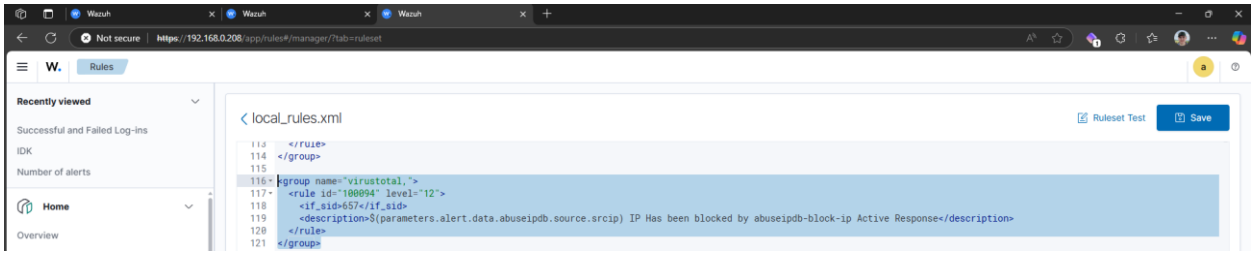


Рисунок 3.92 – Правило для надсилання повідомлення про результати блокування

3.4. Налаштування додаткових модулів Wazuh

Механізми System Inventory, Security Configuration Assesment та Vulnerability Detection за замовчування увімкнені у платформі Wazuh SIEM та

XDR. Однак для кращої роботи та отримання більшої кількості інформації для підвищення рівня безпеки необхідно внести деякі зміни до роботи перерахованих модулів [35].

В першу чергу було внесено зміни до роботи механізму Wazuh System Inventory, було переналаштовано конфігурацію у файлі ossec.conf, забезпечивши сканування інвентарю кожну хвилину замість стандартної години, саме це дозволить оперативно фіксувати зміни в стані системи, а саме установлені пакети, процеси, порти та апаратне забезпечення (рис. 3.93).

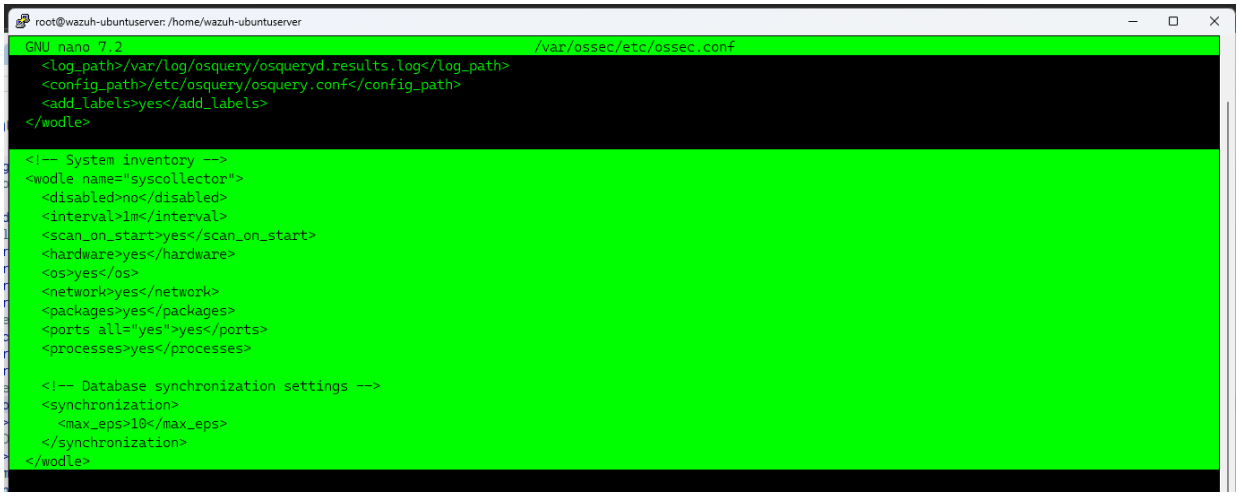


Рисунок 3.93 – Зміна налаштувань модуля System Inventory (Syscollector)

Розширення частоти сканувань до щохвилинної допоможе SOC аналітикам більш оперативно виявляти будь які зміни пов’язані з інвентарем системи. Крім того, параметр scan_on_start=“yes” гарантує, що перше повне сканування відбувається одразу після запуску агента.

Також було вже попередньо додано правило для моніторингу додавання нових мережевих інтерфейсів на всіх кінцевих пристроях, моніторинг мережевих інтерфейсів на пряму залежить від налаштувань System Inventory (див. рис. 3.61).

Надалі було внесено незначні зміни до модуля Security Configuration Assesment, налаштовано сканування кожні 6 годин, що забезпечує спеціалістів відносно актуальною інформацію та не перевантажує кінцеві пристрої. Також увімкнено опцію проведення Security Configuration Assesment при старті кінцевого пристрою (рис. 3.94).

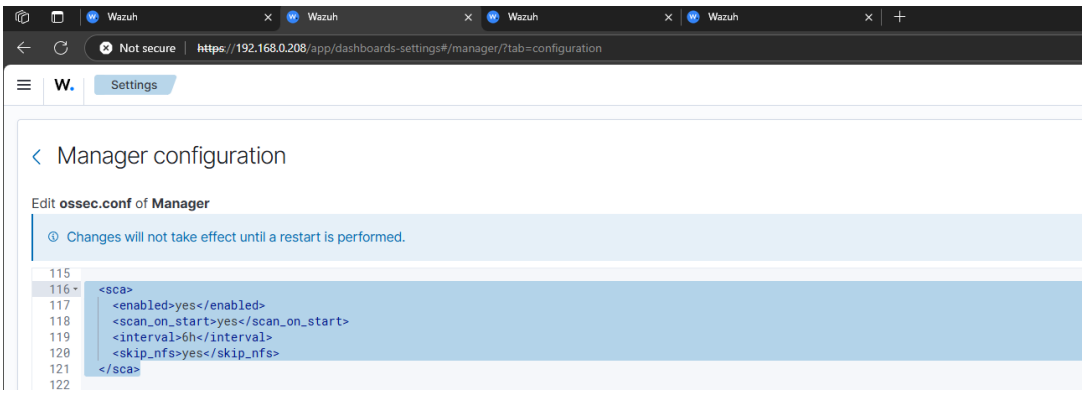


Рисунок 3.94 – Змінена конфігурація модуля Security Configuration Assesment

Vulnerability Detection модуль у платформі Wazuh SIEM та XDR відповідає за виявлення відомих вразливостей у встановленому програмному забезпеченні на основі CVE баз даних.

У конфігурацію цього модуля було внесено певні зміни, зокрема знижено частоту проведення сканувань до 60 хвилин. Окрім того, було активовано важливу опцію індексування, яка забезпечує коректне збереження та відображення результатів сканувань Vulnerability Detection у механізму пошуку логів Discover (рис. 3.95). Завдяки цьому підвищується ефективність аналізу вразливостей та покращується доступність даних для подальшого моніторингу.

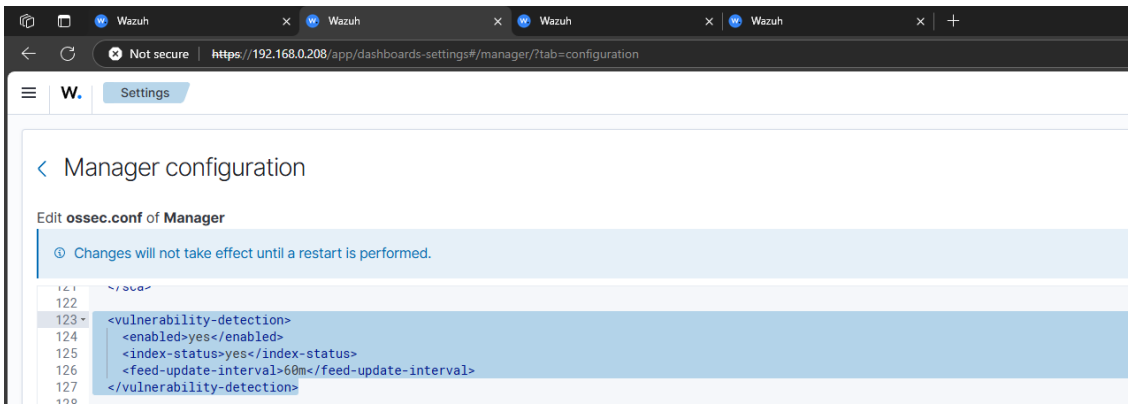


Рисунок 3.95 – Змінена конфігурація модуля Vulnerability Detection

На кінець було адаптовано попередньо створені правила надсилання попереджень під стандарт General Data Protection IV. Було визначено до яких статей та параграфів цього регламенту відносяться спрацювання правил та додано правила до відповідних груп (рис. 3.96, рис. 3.97 – рис. 3.98).



Рисунок 3.96 – Адаптація правил під відповідні статті та параграфи GDPR



Рисунок 3.97 – Адаптація правил під відповідні статті та параграфи GDPR

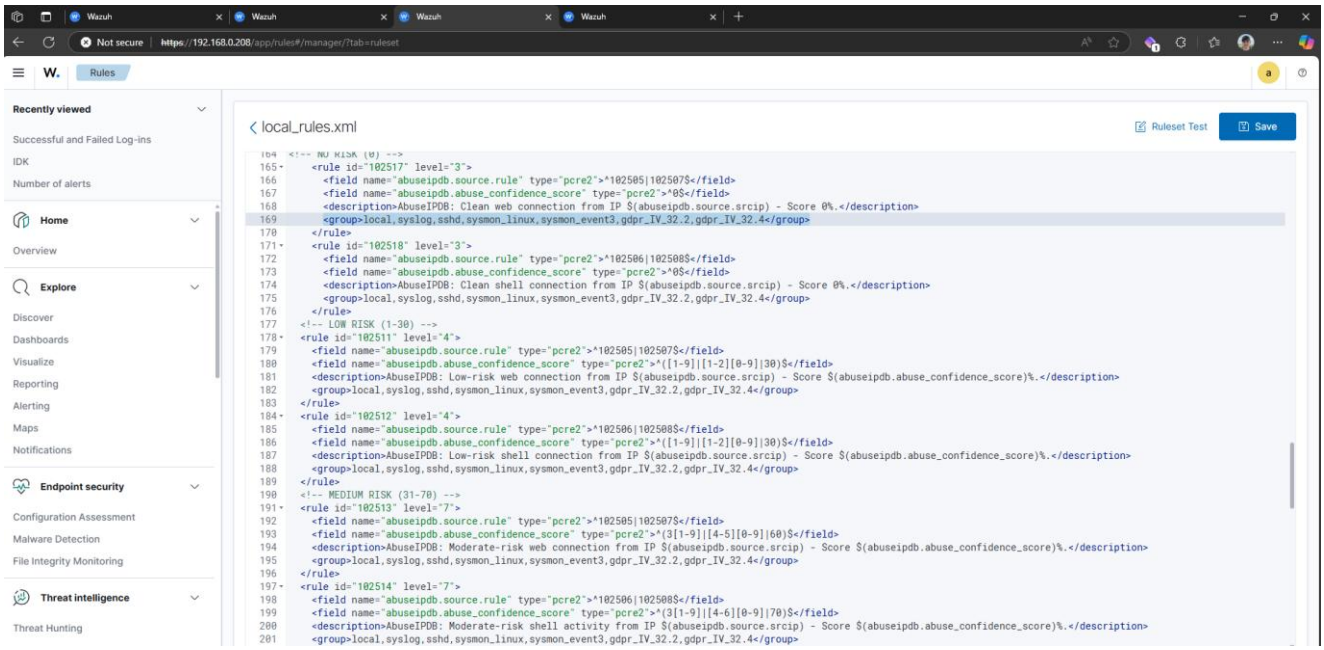


Рисунок 3.98 – Адаптація правил під відповідні статті та параграфи GDPR

Адаптація правил у Wazuh відповідно до вимог регламенту GDPR дозволяє відстежувати реальні показники відповідності нормативам захисту даних (рис. 3.99). У разі відсутності такої адаптації платформа Wazuh SIEM та XDR не зможе забезпечити належний рівень моніторингу та звітності, що може призвести до порушень або невідповідності нормативам захисту даних [38].

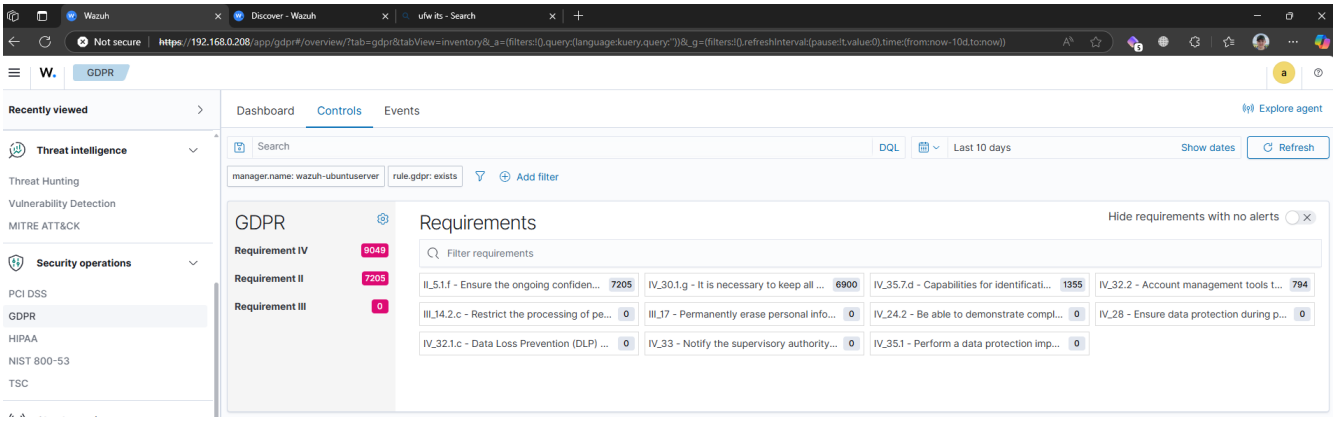


Рисунок 3.99 – Наявність показників GDPR у Wazuh Dashboard

Для адаптації було обрано саме відповідності регламенту GDPR, оскільки цей нормативний акт є найпоширенішим та найбільш значущим стандартом захисту персональних даних у Європейському Союзі. Хоча платформа Wazuh надає можливість адаптації та відслідковування відповідності іншим стандартам, однак GDPR має найширше практичне застосування у європейському регіоні і є обов'язковим для більшості організацій. Саме тому адаптація правил до GDPR у Wazuh було пріоритетним завданням.

3.5. Конфігурування інформаційних дошок

Створення зручних та інформативних інформаційній дошок у Wazuh дозволяє сформувати інтегровану систему моніторингу та аналізу інцидентів безпеки, суттєво підвищуючи ефективність роботи операційного центру. Завдяки такому підходу аналітики мають змогу одразу побачити стан агентів, виявлені загрози, результати перевірок конфігурацій та іншу критично важливу інформацію.

Інформаційні дошки забезпечують швидкий доступ до ключових показників, дозволяючи оперативно реагувати на інциденти, заглиблюватися в аналіз подій і приймати обґрунтовані рішення з мінімальною затримкою.

Для початку конфігурування інформаційних дошок було створено власні візуалізації, які відображають необхідний блок інформації (рис. 3.100). Візуалізації слугують своєрідними блоками з яких створюються всі інформаційні панелі.

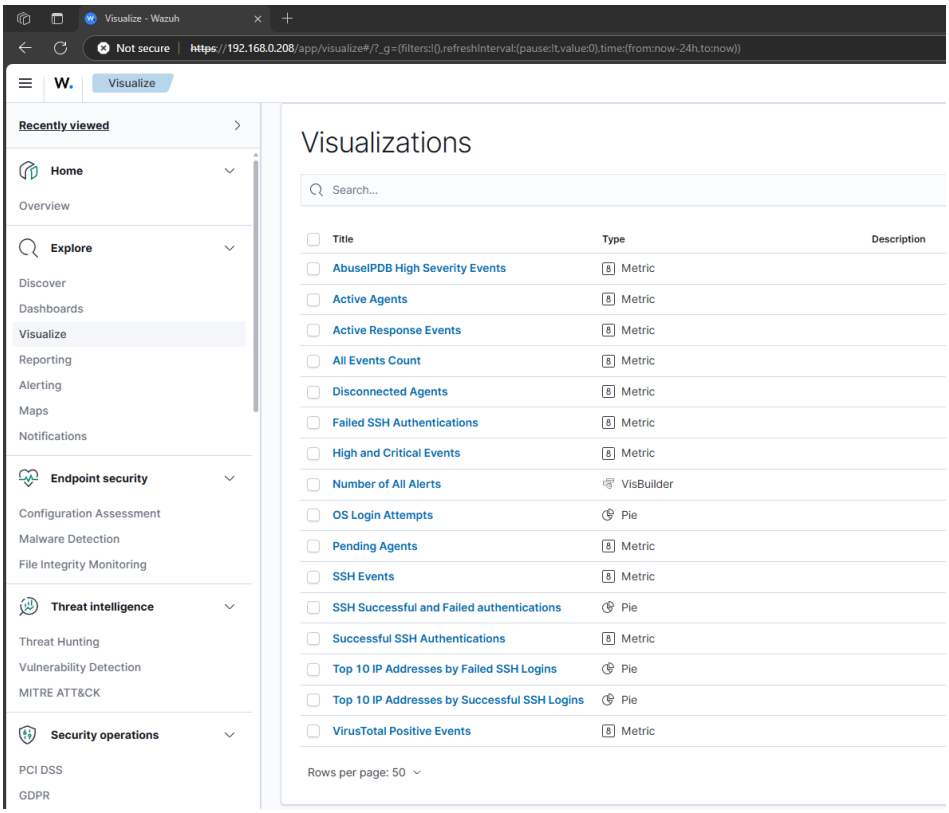


Рисунок 3.100 – Список створених візуалізацій для інформаційних панелей

Створення кожної візуалізації вимагало створення унікальних налаштувань для агрегації, фільтрування та сортування подій певними методами які як наслідок надавали необхідну інформацію про події з індексу.

До прикладу було налаштовано візуалізацію SSH Successful and Failed authentications (рис. 3.101). Ця візуалізація передбачала агрегацію логів за кількість, розділення кола на два показники, та для кожного показника налаштування відповідного фільтрування по успішним та невдалим аутентифікаціям.

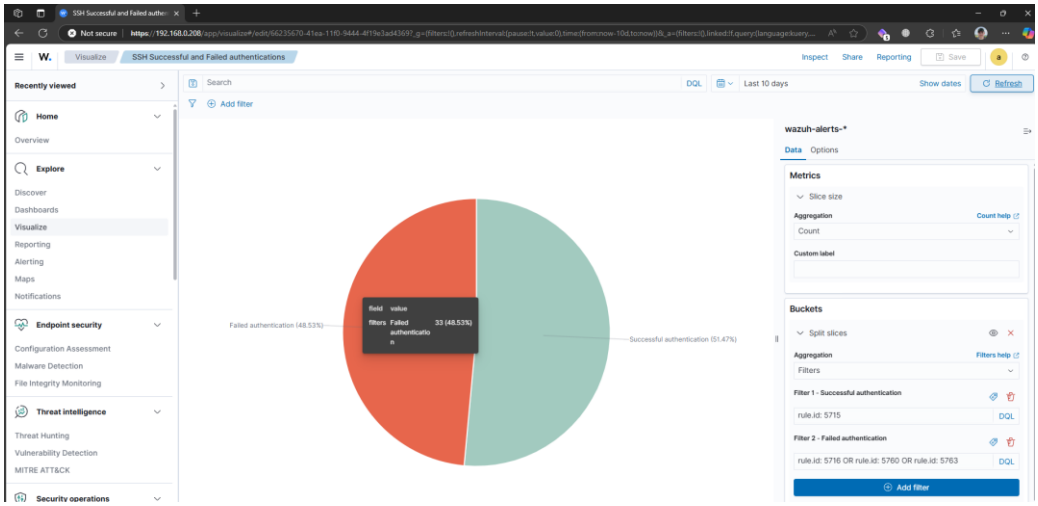


Рисунок 3.101 – Налаштування візуалізації успішних та негативних входів до SSH

Таким чином було створено 16 візуалізацій, більша частина з яких була використання для створення двох попередньо спроєктованих інформаційних панелей.

Загальна інформаційна панель General Dashboard була створена з метою моніторингу найважливіших показників безпеки в інформаційній системі, яку обслуговує операційний центр безпеки (рис. 3.102).

На панелі відображається кількість активних, відключених і агентів, що очікують на реєстрацію, це дозволяє оцінити загальний стан взаємозв’язку між агентами та сервером.

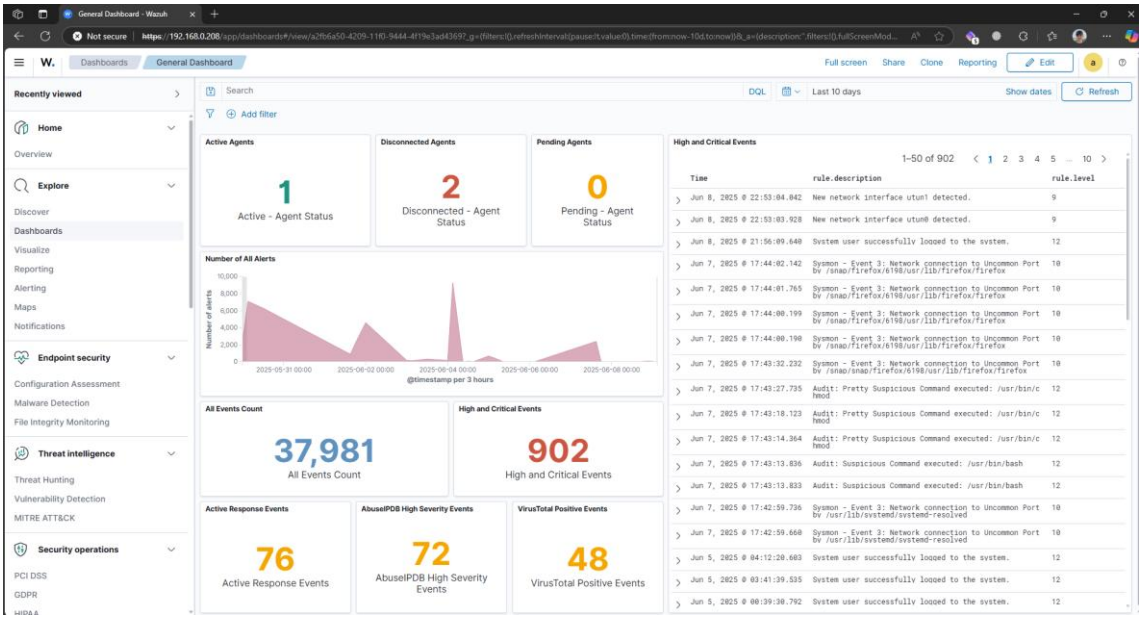


Рисунок 3.102 – Загальна інформаційна панель

У центрі панелі розміщено графік, який показує динаміку кількості сповіщень за певну кількість останніх днів з автоматичним розбиттям по трьохгодинних інтервалах.

Нижня частина панелі представляє загальну кількість згенерованих подій та окремо виділено події високої та критичної важливості. Також доступна аналітика по категоріях, зокрема події типу Active Response, події з високим рівнем загрози з AbuseIPDB та виявленні гарантовано шкідливі файли за допомогою системи VirusTotal.

Праворуч відображено таблицю подій високого та критичного рівня, що містить часові мітки, описи правил, які спрацювали, та рівень важливості події.

Створення інформаційної панелі подій SSH Events Dashboard було зумовлено моніторингом спроб підключення до систем через протокол SSH, що є важливим з точки зору контролю доступу та виявлення потенційних атак, зокрема перебору паролів (рис. 3.103).

У центральній частині панелі відображається журнал подій із зазначенням часу, опису дії, імен агентів, IP адрес джерел та рівнів важливості. Це дає змогу аналітику швидко ідентифікувати джерела активності та оцінити її критичність.

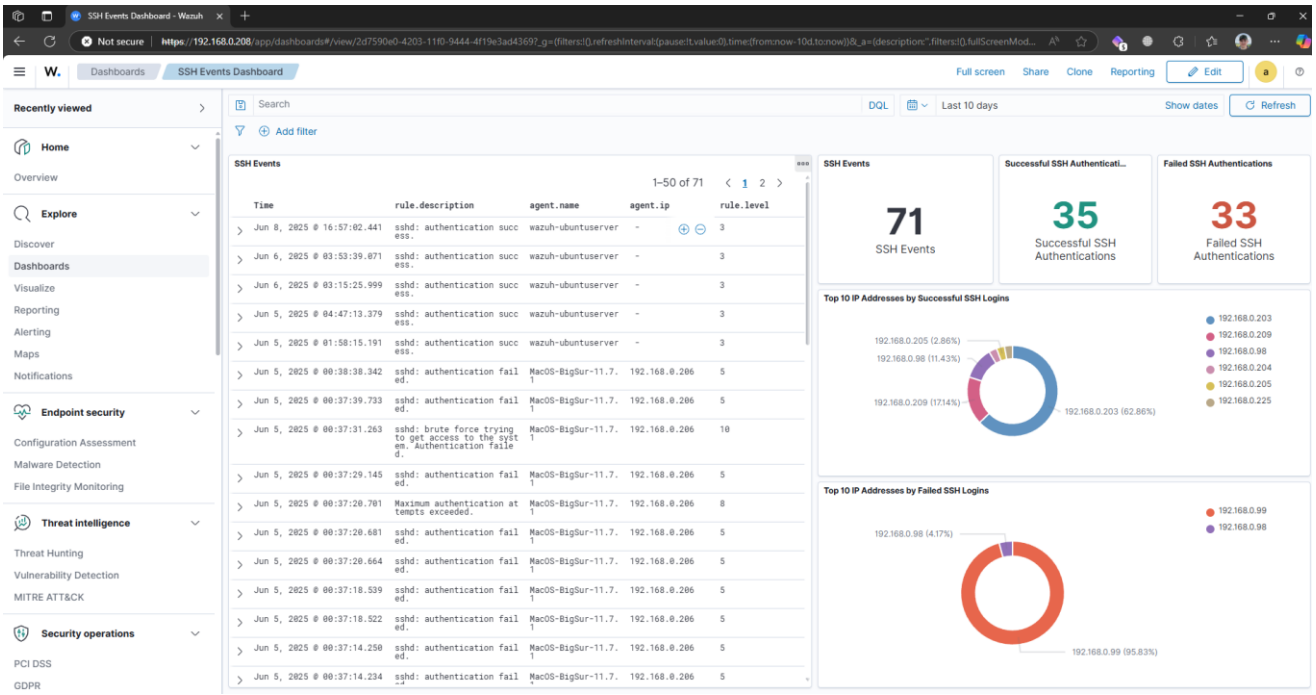


Рисунок 3.103 – Інформаційна панель SSH Events Dashboard

Права частина панелі містить візуалізації, що допомагають швидко зрозуміти ситуацію, зокрема загальну кількість подій, а також статистики за IP адресами, які найчастіше здійснювали успішні та невдалі входи.

Ця панель є важливим інструментом для виявлення підозрілої поведінки в мережі, підвищення прозорості процесів автентифікації та підтримки загальної інформаційної безпеки інфраструктури.

Висновки до третього розділу

У третьому розділі було здійснено практичну реалізацію спроєктованої SIEM та XDR підсистеми на базі Wazuh. Для початку було проведено налаштування інфраструктури та розгортання самої платформи Wazuh, що стало основою для подальшої роботи. В межах реалізації механізмів збору логів та виявлення загроз було налаштовано збір логів із різних джерел що створило комплексне виявлення загроз, яке охоплює аналіз подій з файлової системи, виявлення аномальної мережевої активності, шкідливих команд, технік експлуатації PowerShell та активності пов’язаної з Rootkit подібним програмним забезпеченням. Особливу увагу приділено інтеграції із зовнішніми репутаційними сервісами, зокрема VirusTotal та AbuseIPDB. Впроваджено також логування з Microsoft Defender, що розширює моніторинг кінцевих пристроїв на базі Windows.

На базі реалізованих механізмів виявлених подій налаштовано механізми активного реагування, які включають автоматичне видалення шкідливих файлів, блокування небезпечних IP адрес, а також протидію спробам несанкціонованого доступу через SSH.

Було налаштовано інвентаризацію систем за допомогою модуля Syscollector, перевірку відповідності конфігурації політикам безпеки через модуль SCA, а також механізми виявлення вразливостей. Окремо здійснено адаптацію розроблених правил під вимоги GDPR.

Таким чином, у третьому розділі було реалізовано Wazuh SIEM та XDR підсистему.

РОЗДІЛ 4. ТЕСТУВАННЯ НАЛАШТОВАНОЇ ПІДСИСТЕМИ ТА ПРАКТИЧНЕ ВИКОРИСТАННЯ WAZUH SIEM ТА XDR ПІДСИСТЕМИ

4.1. Тестування реалізованих у Wazuh механізмів виявлення та реагування на загрози

Процес тестування після проведення налаштувань є критично важливим етапом, оскільки він дозволяє виявити потенційні помилки та показати ефективність налаштованої системи.

Тестування буде проходити комплексним методом. Комплексне тестування передбачає використання декількох налаштованих механізмів або модулів одночасно з метою отримання комплексної інформації про роботу системи.

Для початку буде протестовано ефективність роботи налаштованого модуля FIM який інтегрований з сервісом VirusTotal. Крім того, в свою чергу скрипт активної відповіді повинен видалити знайдений шкідливий файл.

На робочий стіл кінцевого пристрою з операційною системою MacOS було завантажено спеціальний тестовий файл eicar який ідентифікується антивірусним ПЗ як шкідливий [36].

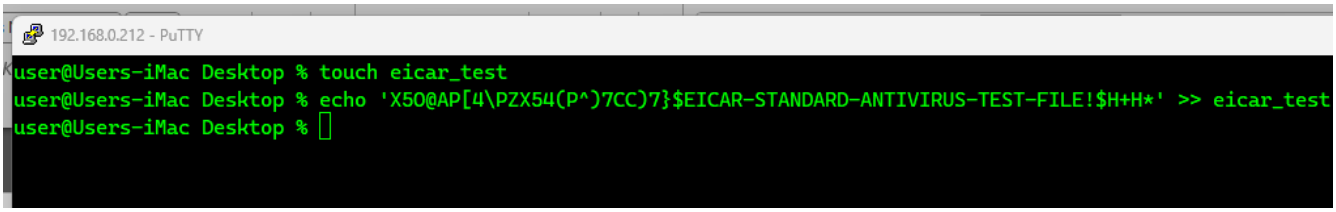
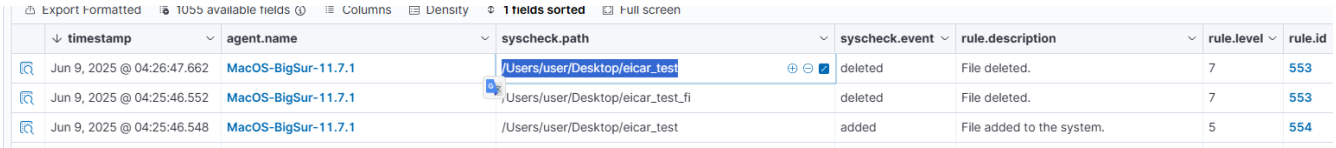


Рисунок 4.1 – Створений тестовий файл зі шкідливим вмістом

Після створення файлу на робочому столі, протягом хвилини на кінцевому пристрої відбулося сканування та було виявлено додавання нового файлу eicar_test (рис. 4.2).


Export Formatted | 1055 available fields | Columns | Density | 1 fields sorted | Full screen

timestamp	agent.name	syscheck.path	syscheck.event	rule.description	rule.level	rule.id
Jun 9, 2025 @ 04:26:47.662	MacOS-BigSur-11.7.1	/Users/user/Desktop/eicar_test	deleted	File deleted.	7	553
Jun 9, 2025 @ 04:25:46.552	MacOS-BigSur-11.7.1	/Users/user/Desktop/eicar_test_fi	deleted	File deleted.	7	553
Jun 9, 2025 @ 04:25:46.548	MacOS-BigSur-11.7.1	/Users/user/Desktop/eicar_test	added	File added to the system.	5	554

Рисунок 4.2 – Результат роботи механізму File Integrity Monitoring

Далі файл майже одразу зник, однак логи надійшли з затримкою, адже файл з логами активної відповіді на кінцевому пристрої MacOS читається з певною затримкою. Крім того, можемо побачити, що інтеграція з сервісом VirusTotal спрацювала успішно та 3 двигуни виявили вразливість у файлі. Шкідливий файл було успішно видалено власним скриптом активної відповіді (рис. 4.3).

timestamp	agent.name	rule.description	rule.level	rule.id
Jun 9, 2025 @ 04:25:51.699	MacOS-BigSur-11.7.1	VirusTotal: Alert - /Users/user/Desktop/eicar_test_fi - No positives found	3	87104
Jun 9, 2025 @ 04:25:51.498	MacOS-BigSur-11.7.1	active-response/bin/remove-threat.sh removed threat located at /Users/user/Desktop/eicar_test	12	100092
Jun 9, 2025 @ 04:25:49.931	MacOS-BigSur-11.7.1	VirusTotal: Alert - /Users/user/Desktop/eicar_test - 3 engines detected this file	12	87105

Рисунок 4.3 – Успішна ідентифікація файлу як шкідливого та його видалення

Якщо відкрити згенероване попередження після сканування файлу сервісом VirusTotal то можемо побачити низку важливих полів, які можуть бути використані для розслідування інциденту (рис. 4.4)

Document Details

View surrounding documents | View single document

_index	wazuh-alerts-4.x-2025.06.09
agent.id	004
agent.ip	192.168.0.212
agent.name	MacOS-BigSur-11.7.1
data.integration	virustotal
data.virustotal.found	1
data.virustotal.malicious	1
data.virustotal.permalink	https://www.virustotal.com/gui/file/3c53aa3a7d9ae5b66df8ab86b8a2fe579bde87c26be3d75072eef00832fb8b56/detection/f-3c53aa3a7d9ae5b66df8ab86b8a2fe579bde87c26be3d75072eef00832fb8b56-1745285338
data.virustotal.positives	3
data.virustotal.scan_date	2025-04-22 01:28:58
data.virustotal.sha1	05b20677878e50cecd47b56b042309442d10322f
data.virustotal.source.alert_id	1749432407.58035
data.virustotal.source.file	/Users/user/Desktop/eicar_test
data.virustotal.source.md5	87bf6020e70d538fc493b0636c2486e4
data.virustotal.source.sha1	05b20677878e50cecd47b56b042309442d10322f
data.virustotal.total	61
decoder.name	json
id	1749432410.58737
input.type	log
location	virustotal
manager.name	wazuh-ubuntuserver
rule.description	VirusTotal: Alert - /Users/user/Desktop/eicar_test - 3 engines detected this file
rule.firedtimes	6
rule.gdpr	IV_35.7.d
rule.groups	virustotal
rule.id	87105
rule.level	12
rule.mail	true
rule.mitre.id	T1203
rule.mitre.tactic	Execution
rule.mitre.technique	Exploitation for Client Execution
rule.pci_dss	10.6.1, 11.4
timestamp	Jun 9, 2025 @ 04:26:50.791

Рисунок 4.4 – Результат сканування файлу сервісом VirusTotal

Відкривши згенероване повідомлення активної відповіді ми можемо побачити дані про запущений скрипт, файл відносно якого було запущено скрипт, локацію логів з яких було отримано інформацію та згенеровано попередження, та інші важливі поля, які теж можуть допомогти в розслідуванні інциденту (рис. 4.5)

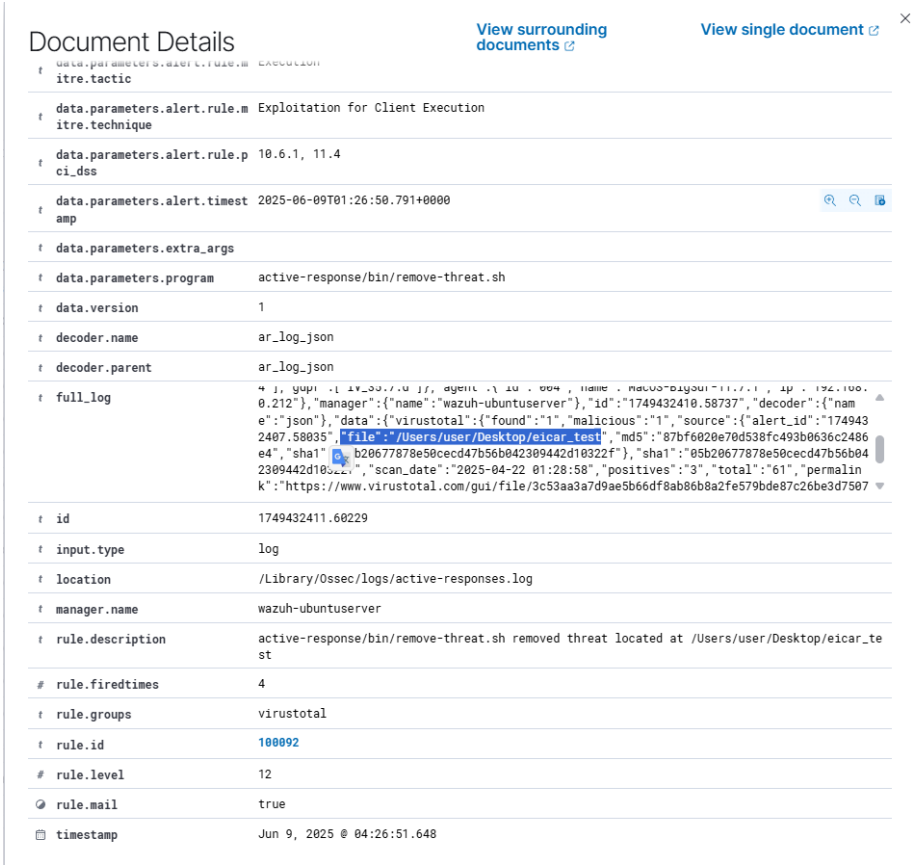


Рисунок 4.5 – Результат роботи модуля Active Response

Стосовно роботи модуля FIM, відкривши згенероване повідомлення можемо побачити файл на який прийшло попередження, права доступу, хеш значення, розмір файлу та іншу важливу інформацію (рис. 4.6).

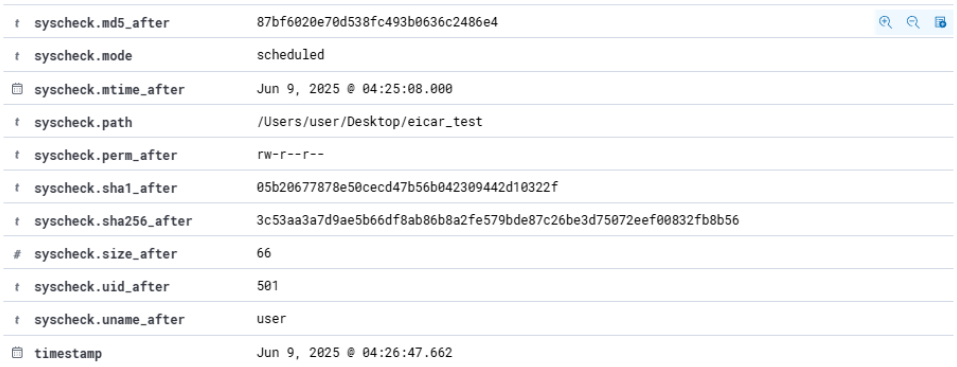


Рисунок 4.6 – Результат роботи модуля FIM

Надалі було проведено аналогічні тестування на кінцевих пристроях з операційними системами Windows та Linux. Варто зауважити, що на кінцевому пристрої з ОС Linux, всі механізми спрацювали майже миттєво, навіть файл не встиг з’явитися на робочому столі (рис. 4.7 – рис. 4.8).

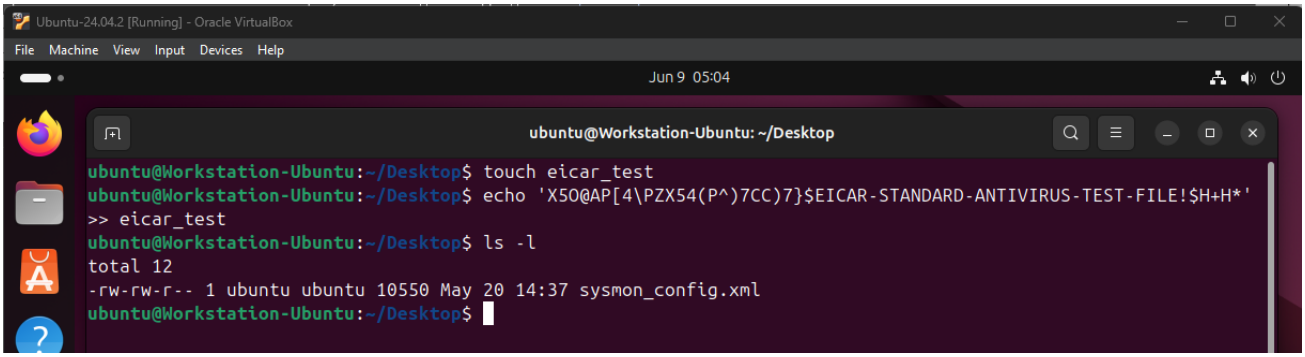


Рисунок 4.7 – Створення шкідливого файлу на ОС Linux

	Jun 9, 2025 @ 05:03:25.447	Ubuntu-24.04.2	active-response/bin/remove-threat.sh removed threat located at /home/ubuntu/Desktop/eicar_test	12	100092
	Jun 9, 2025 @ 05:03:25.081	Ubuntu-24.04.2	File deleted.	7	553
	Jun 9, 2025 @ 05:03:25.054	Ubuntu-24.04.2	VirusTotal: Alert - /home/ubuntu/Desktop/eicar_test - 3 engines detected this file	12	87105
	Jun 9, 2025 @ 05:03:23.600	Ubuntu-24.04.2	VirusTotal: Alert - /home/ubuntu/Desktop/eicar_test - No positives found	3	87104
	Jun 9, 2025 @ 05:03:23.445	Ubuntu-24.04.2	Sysmon - Event 1: Process creation /usr/bin/dash	3	200151
	Jun 9, 2025 @ 05:03:22.859	Ubuntu-24.04.2	Sysmon - Event 1: Process creation /usr/bin/dash	3	200151
	Jun 9, 2025 @ 05:03:22.128	Ubuntu-24.04.2	Integrity checksum changed.	7	550
	Jun 9, 2025 @ 05:03:20.858	Ubuntu-24.04.2	Audit: Command: /usr/bin/touch.	3	80792
	Jun 9, 2025 @ 05:03:20.517	Ubuntu-24.04.2	File added to the system.	5	554

Рисунок 4.8 – Результат роботи механізмів виявлення та реагування на загрози

На кінцевому пристрої з операційною системою Windows модуль FIM інтегрований з сервісом VirusTotal та модуль Active Response успішно виявили та видалити шкідливий файл створений на робочому столі (рис. 4.9 – рис. 4.10).

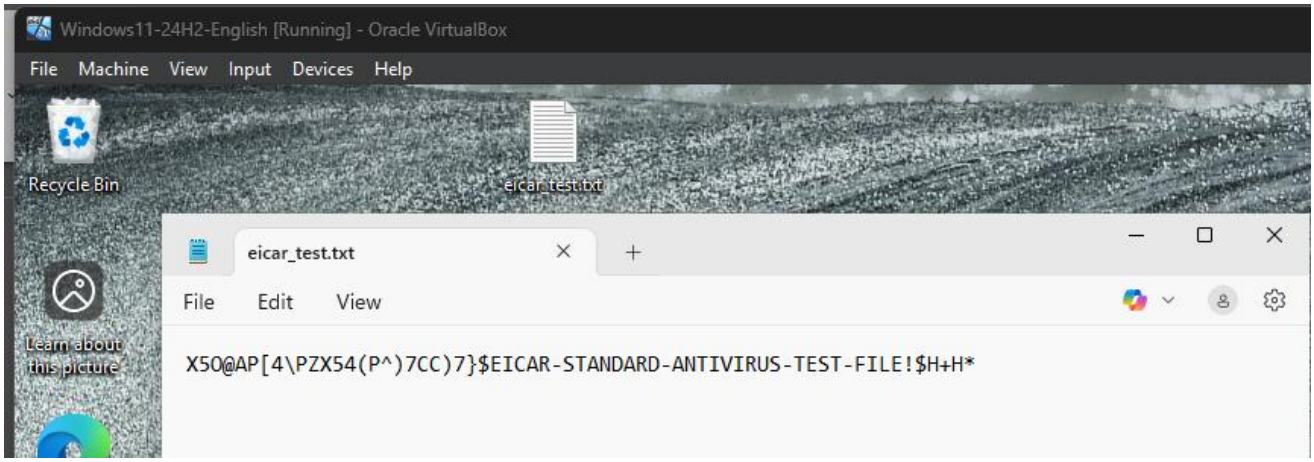


Рисунок 4.9 – Створення шкідливого файлу на робочому столі

	Jun 9, 2025 @ 05:29:59.914	Windows11-24H2-English	active-response/bin/remove-threat.exe removed threat located at c:\users\windows\desktop\eicar_test.txt	12	100092
	Jun 9, 2025 @ 05:29:58.917	Windows11-24H2-English	VirusTotal: Alert - c:\users\windows\desktop\eicar_test.txt - 66 engines detected this file	12	87105
	Jun 9, 2025 @ 05:29:57.863	Windows11-24H2-English	active-response/bin/remove-threat.exe removed threat located at c:\users\windows\desktop\eicar_test.txt	12	100092
	Jun 9, 2025 @ 05:29:57.806	Windows11-24H2-English	active-response/bin/remove-threat.exe removed threat located at c:\users\windows\desktop\eicar_test.txt	12	100092
	Jun 9, 2025 @ 05:29:57.420	Windows11-24H2-English	VirusTotal: Alert - c:\users\windows\desktop\eicar_test.txt - 66 engines detected this file	12	87105
	Jun 9, 2025 @ 05:29:56.043	Windows11-24H2-English	File deleted.	7	553
	Jun 9, 2025 @ 05:29:55.536	Windows11-24H2-English	VirusTotal: Alert - c:\users\windows\desktop\eicar_test.txt - 66 engines detected this file	12	87105
	Jun 9, 2025 @ 05:29:53.621	Windows11-24H2-English	Integrity checksum changed.	7	550
	Jun 9, 2025 @ 05:29:53.017	Windows11-24H2-English	File added to the system.	5	554
	Jun 9, 2025 @ 05:28:00.878	Windows11-24H2-English	Service startup type was changed	3	61104
	Jun 9, 2025 @ 05:27:42.005	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
	Jun 9, 2025 @ 05:27:12.774	Windows11-24H2-English	Software protection service scheduled successfully.	3	60642

Рисунок 4.10 – Результат роботи механізмів виявлення та реагування на загрози

Таким чином можемо зробити висновок, що модуль FIM в інтеграції з сервісом VirusTotal та модулем Active Response ефективно виявляють та реагують на загрози, що знаходяться на кінцевих пристроях усіх ОС.

Для тестування розширеного збору логів з MS Defender було створено шкідливий файл в директорія яка не налаштована на сканування FIM модулем (рис. 4.11 – рис. 4.12).

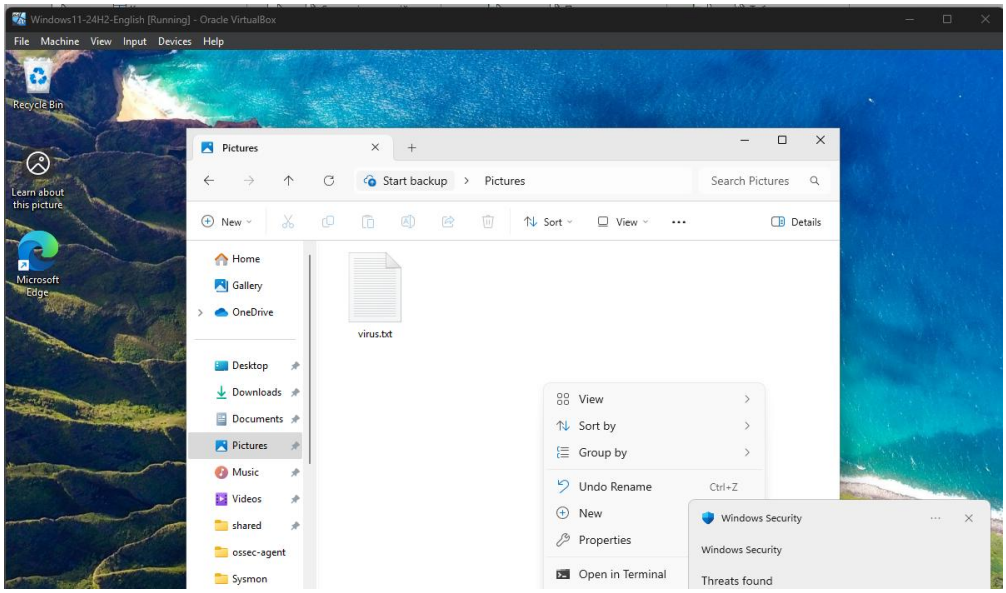


Рисунок 4.11 – Додавання шкідливого файлу до директорії Pictures

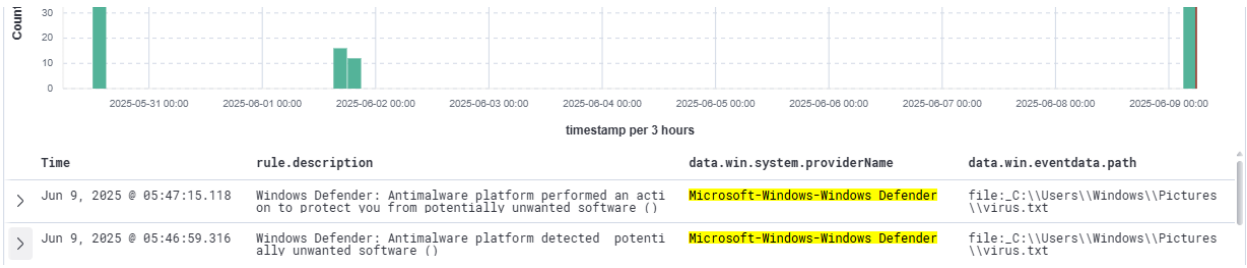


Рисунок 4.12 – Успішне отримання логів з MS Defender про видалення файлу

Отримані попередження за допомогою інтеграції з MS Defender містять детальну інформацію про спрацювання, зокрема інформацію про процес що створив файл, вид загрози виявлений MS Defender, який вид захисту видалив шкідливий файл [25] (рис. 4.13).

t data.win.eventdata.post Clean Status	0
t data.win.eventdata.pre Execution Status	0
t data.win.eventdata.process Name	C:\\Program Files\\WindowsApps\\Microsoft.WindowsNotepad_11.2503.16.0_x64__8wekyb3d8bbwe\\Notepad\\Notepad.exe
t data.win.eventdata.product Name	Microsoft Defender Antivirus
t data.win.eventdata.product Version	4.18.25040.2
t data.win.eventdata.security intelligence Version	AV: 1.429.423.0, AS: 1.429.423.0, NIS: 1.429.423.0
t data.win.eventdata.severity ID	5
t data.win.eventdata.severity Name	Severe
t data.win.eventdata.source ID	3
t data.win.eventdata.source Name	Real-Time Protection
t data.win.eventdata.state	1
t data.win.eventdata.status Code	1
t data.win.eventdata.threat ID	2147519003
t data.win.eventdata.threat Name	Virus:DOS/EICAR_Test_File
t data.win.eventdata.type ID	0

Рисунок 4.13 – Отримане попередження з MS Defender про видалення файлу

Таким чином можемо зробити висновок, що інтеграція з Microsoft Defender ефективно надсилає журнали, зокрема про виявлення та реагування на загрози, що знаходяться на кінцевих пристроях з ОС Windows.

Далі було проведено комплексне тестування аномальних мережевих з’єднань, було створено підключення до шкідливої IP адреси за допомогою веб браузера на кінцевому пристрої з ОС Windows (рис. 4.14).

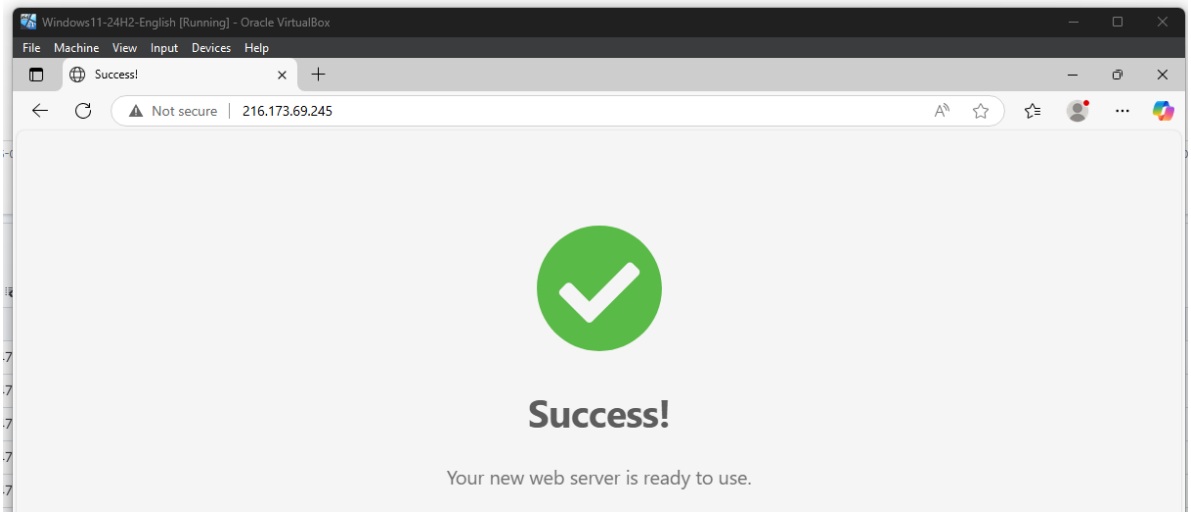
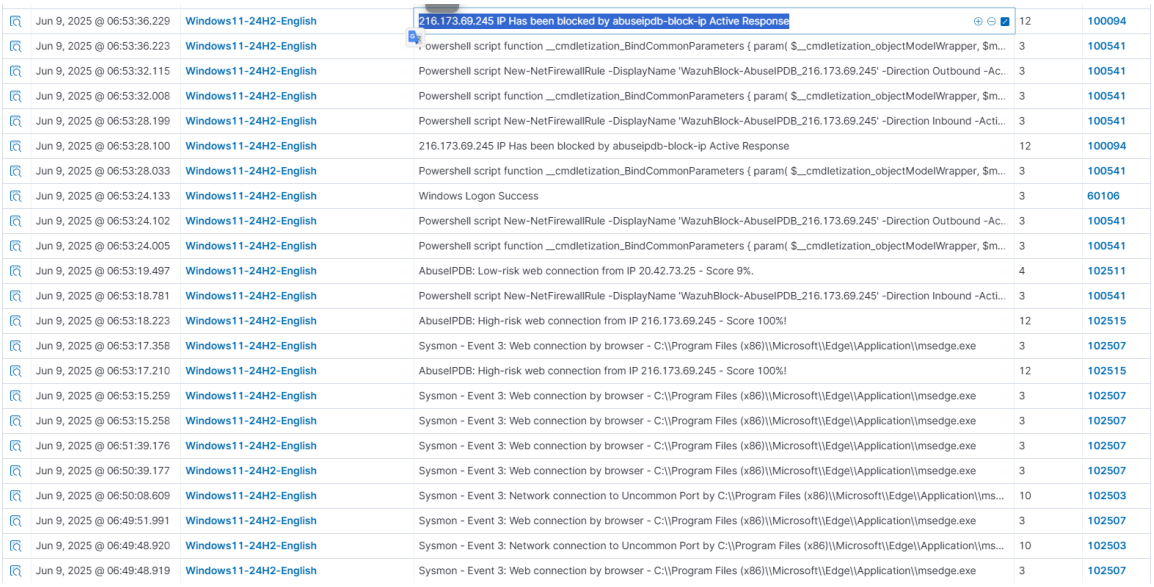


Рисунок 4.14 – Успішне підключення до шкідливої IP адреси

Після підключення до зловмисного веб сайту Sysmon успішно надсилає інформацію до Wazuh про декілька встановлених мережевих підключень до IP адреси через веб браузер. Надалі попередньо створені правила у файлі local_rules.xml автоматично генерують повідомлення про підключення та передають згенероване повідомлення до інтеграції з AbuseIPDB. Як наслідок сервіс AbuseIPDB успішно повертає інформацію про шкідливість підключення та кастомний скрипт додає IP адресу до Windows Defender Firewall, чим в свою чергу її блокує (рис. 4.15 – рис. 4.17).



Jun 9, 2025 @ 06:53:36.229	Windows11-24H2-English	216.173.69.245 IP Has been blocked by abuseipdb-block-ip Active Response	12	100094
Jun 9, 2025 @ 06:53:36.223	Windows11-24H2-English	owershell script function __cmdletization_BindCommonParameters { param($__cmdletization_objectModelWrapper, $m...	3	100541
Jun 9, 2025 @ 06:53:32.115	Windows11-24H2-English	Powershell script New-NetFirewallRule -DisplayName 'WazuhBlock-AbuseIPDB_216.173.69.245' -Direction Outbound -Ac...	3	100541
Jun 9, 2025 @ 06:53:32.008	Windows11-24H2-English	Powershell script function __cmdletization_BindCommonParameters { param($__cmdletization_objectModelWrapper, $m...	3	100541
Jun 9, 2025 @ 06:53:28.199	Windows11-24H2-English	Powershell script New-NetFirewallRule -DisplayName 'WazuhBlock-AbuseIPDB_216.173.69.245' -Direction Inbound -Acti...	3	100541
Jun 9, 2025 @ 06:53:28.100	Windows11-24H2-English	216.173.69.245 IP Has been blocked by abuseipdb-block-ip Active Response	12	100094
Jun 9, 2025 @ 06:53:28.033	Windows11-24H2-English	Powershell script function __cmdletization_BindCommonParameters { param($__cmdletization_objectModelWrapper, $m...	3	100541
Jun 9, 2025 @ 06:53:24.133	Windows11-24H2-English	Windows Logon Success	3	60106
Jun 9, 2025 @ 06:53:24.102	Windows11-24H2-English	Powershell script New-NetFirewallRule -DisplayName 'WazuhBlock-AbuseIPDB_216.173.69.245' -Direction Outbound -Ac...	3	100541
Jun 9, 2025 @ 06:53:24.005	Windows11-24H2-English	Powershell script function __cmdletization_BindCommonParameters { param($__cmdletization_objectModelWrapper, $m...	3	100541
Jun 9, 2025 @ 06:53:19.497	Windows11-24H2-English	AbuseIPDB: Low-risk web connection from IP 20.42.73.25 - Score 9%.	4	102511
Jun 9, 2025 @ 06:53:18.781	Windows11-24H2-English	Powershell script New-NetFirewallRule -DisplayName 'WazuhBlock-AbuseIPDB_216.173.69.245' -Direction Inbound -Acti...	3	100541
Jun 9, 2025 @ 06:53:18.223	Windows11-24H2-English	AbuseIPDB: High-risk web connection from IP 216.173.69.245 - Score 100%!	12	102515
Jun 9, 2025 @ 06:53:17.358	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:53:17.210	Windows11-24H2-English	AbuseIPDB: High-risk web connection from IP 216.173.69.245 - Score 100%!	12	102515
Jun 9, 2025 @ 06:53:15.259	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:53:15.258	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:51:39.176	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:50:39.177	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:50:08.609	Windows11-24H2-English	Sysmon - Event 3: Network connection to Uncommon Port by C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\ms...	10	102503
Jun 9, 2025 @ 06:49:51.991	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507
Jun 9, 2025 @ 06:49:48.920	Windows11-24H2-English	Sysmon - Event 3: Network connection to Uncommon Port by C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\ms...	10	102503
Jun 9, 2025 @ 06:49:48.919	Windows11-24H2-English	Sysmon - Event 3: Web connection by browser - C:\\Program Files (x86)\\Microsoft\\Edge\\Application\\msedge.exe	3	102507

Рисунок 4.15 – Результат роботи механізмів виявлення та реагування на загрози

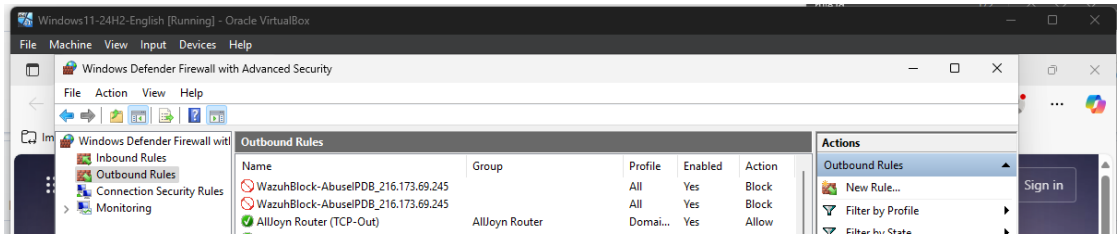


Рисунок 4.16 – Заблокована IP адреса на вихідне підключення в Defender Firewall

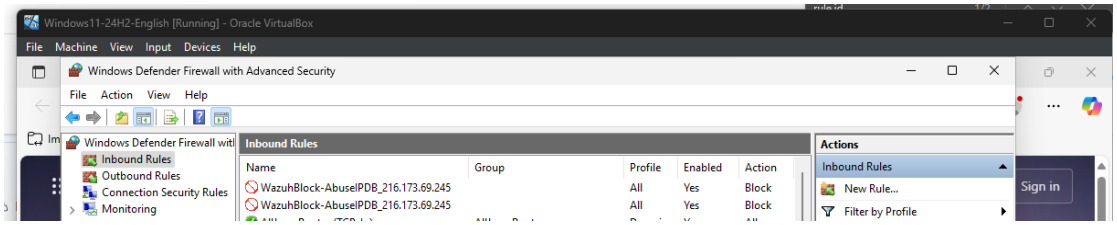


Рисунок 4.17 – Заблокована IP адреса на вхідне підключення в Defender Firewall

При спробі повторного підключення до зловмисного веб сайту не вдається встановити підключення внаслідок успішного блокування (рис. 4.18).

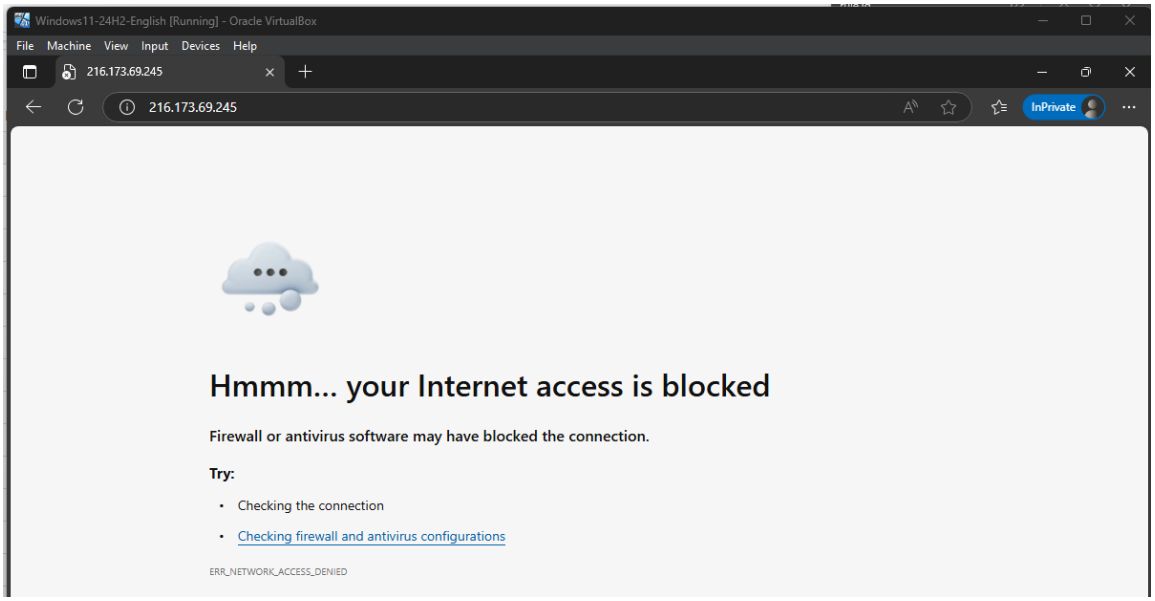


Рисунок 4.18 – Невдала спроба підключення до зловмисного веб сайту

Схоже підключення до зловмисної IP адреси було виконане на ОС Linux з використанням команди curl (рис. 4.19). Як наслідок налаштовані механізми виявлення інтегровані разом з Abuse IPDB та модулем Active Response успішно виявили та заблокували зловмисну IP адресу (рис. 4.20).

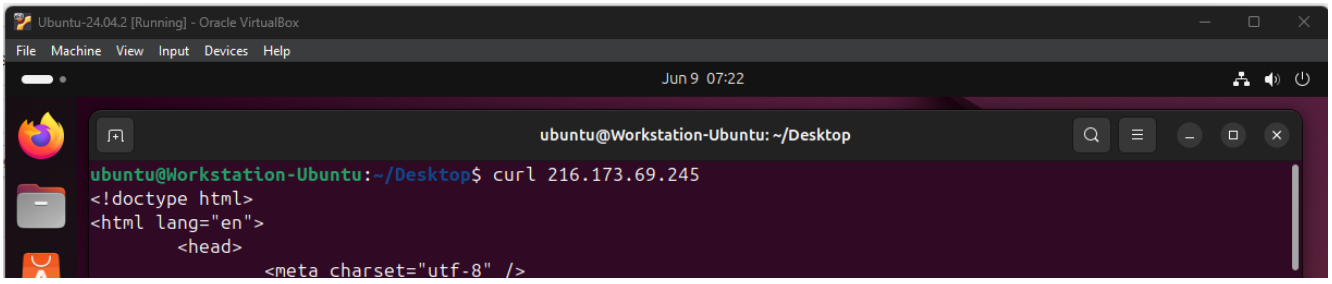


Рисунок 4.19 – Підключення до зловмисної IP адреси

Jun 9, 2025 @ 07:22:58.998	Ubuntu-24.04.2	216.173.69.245 IP Has been blocked by abuseipdb-block-ip Active Response	12	100094
Jun 9, 2025 @ 07:22:57.616	Ubuntu-24.04.2	AbuseIPDB: High-risk shell command attempt from IP 216.173.69.245 - Score 100%!	12	102516
Jun 9, 2025 @ 07:22:56.996	Ubuntu-24.04.2	216.173.69.245 IP Has been blocked by abuseipdb-block-ip Active Response	12	100094
Jun 9, 2025 @ 07:22:56.600	Ubuntu-24.04.2	AbuseIPDB: High-risk shell command attempt from IP 216.173.69.245 - Score 100%!	12	102516
Jun 9, 2025 @ 07:22:55.639	Ubuntu-24.04.2	Sysmon - Event 3: Network connection by Shell - /usr/bin/curl	3	102506
Jun 9, 2025 @ 07:22:55.639	Ubuntu-24.04.2	Sysmon - Event 1: Process creation /usr/bin/curl	3	200151
Jun 9, 2025 @ 07:22:54.992	Ubuntu-24.04.2	Sysmon - Event 3: Network connection by Shell - /usr/bin/curl	3	102506
Jun 9, 2025 @ 07:22:54.989	Ubuntu-24.04.2	Audit: Suspicious Command executed: /usr/bin/curl	12	100212
Jun 9, 2025 @ 07:22:54.989	Ubuntu-24.04.2	Sysmon - Event 1: Process creation /usr/bin/curl	3	200151

Рисунок 4.20 – Результат роботи механізмів виявлення та реагування на загрози

При спробі повторного підключення до зловмисної IP адреси не вдається встановити підключення внаслідок успішного блокування (рис. 4.21). Крім того важливо зауважити, що успішно спрацював механізм виявлення потенційно небезпечних та підозрілих команд, зокрема для команди curl (див. Рисунок 4.20).

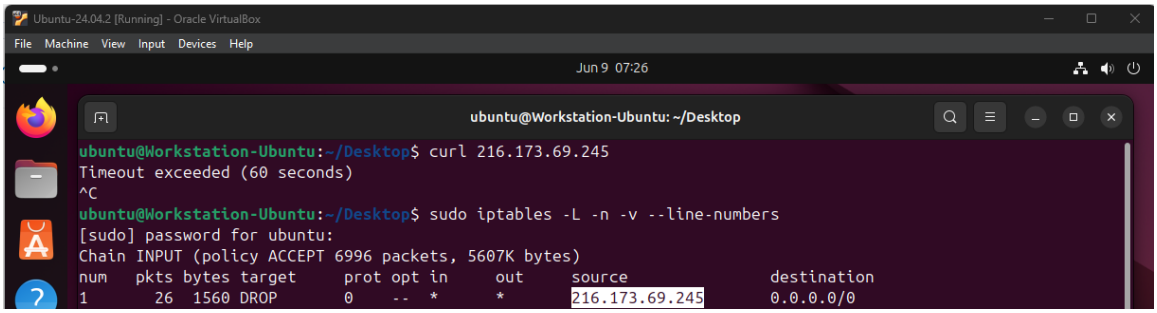


Рисунок 4.21 – Невдала спроба підключення до зловмисного веб сайту

Таким чином можемо зробити висновок комплексного тестування, що виявлення потенційно шкідливих команд, аномальних мережевих підключень інтегроване з сервісом AbuseIPDB та модулем Active Response працює ефективно, зокрема успішно виявляє, ідентифікує та блокує зловмисні IP адреси на ОС Windows та Linux.

Для проведення тестування технік експлуатації Windows PowerShell було виконано закодовану команду типу Base64 та шкідливу команду типу LOTL, яка виконує JavaScript одразу з URL запиту (рис. 4.22).

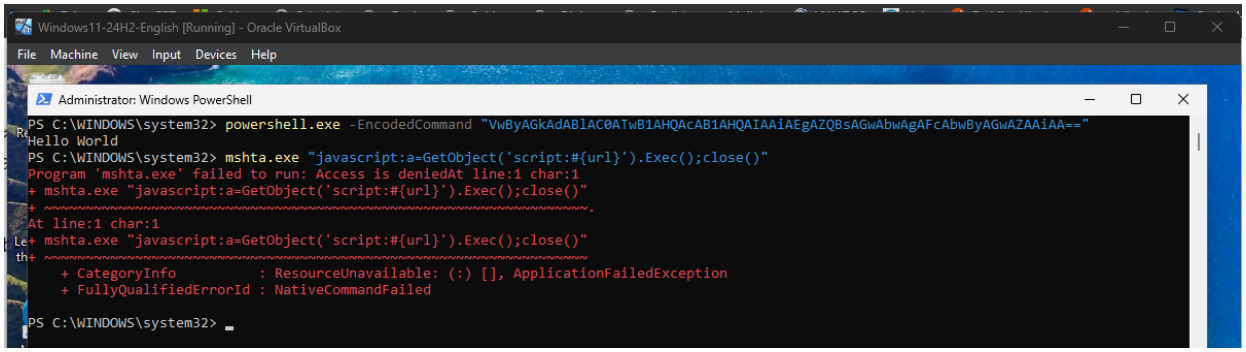


Рисунок 4.22 – Виконання шкідливих команд на Windows PowerShell

Як наслідок налаштовані механізми виявлення технік експлуатації Windows PowerShell успішно виявили та надіслали точні попередження про експлуатацію різними видами шкідливих команд [27]. Крім того, варто зауважити що Windows

Defender успішно заблокував одну з виконаних зловмисних команд, та завдяки інтеграції було надіслано повідомлення до Wazuh про її блокування (рис. 4.23).

	Jun 9, 2025 @ 07:47:32.677	Windows11-24H2-English	Windows Defender: Antimalware platform detected potentially unwanted software ()	12	62123
	Jun 9, 2025 @ 07:47:32.637	Windows11-24H2-English	Windows Defender: Antimalware platform detected potentially unwanted software ()	12	62123
	Jun 9, 2025 @ 07:47:32.284	Windows11-24H2-English	Mshta used to download a file. Possible malicious activity detected.	8	100204
	Jun 9, 2025 @ 07:47:24.272	Windows11-24H2-English	Executable file dropped in folder commonly used by malware	15	92213
	Jun 9, 2025 @ 07:47:24.256	Windows11-24H2-English	Powershell.exe spawned a powershell process which executed a base64 encoded command	12	92057
	Jun 9, 2025 @ 07:47:24.176	Windows11-24H2-English	Encoded command executed via PowerShell.	8	100201
	Jun 9, 2025 @ 07:47:24.160	Windows11-24H2-English	Encoded command executed via PowerShell.	8	100201
	Jun 9, 2025 @ 07:47:24.160	Windows11-24H2-English	Powershell script Write-Output \"Hello World\" Executed	3	100541
	Jun 9, 2025 @ 07:47:24.111	Windows11-24H2-English	Powershell script powershell.exe -EncodedCommand \"VwByAGkAdABIAC0ATwB1AHQAcAB1AHQAIAAiAEgAZQBsAGwA...	3	100541

Рисунок 4.23 – Результат роботи механізмів виявлення та реагування на загрози

Таким чином ми запевнились що виявлення технік експлуатації Windows PowerShell та інтеграція з Windows Defender функціонують коректно, ефективно виявляють та запобігають шкідливим командам.

Для перевірки роботи модуля Rootcheck та оновленого списку програмного забезпечення типу Rootkit було імітовано наявність потенційно шкідливих файлів пов’язаних з цим шкідливим типом програмного забезпечення (рис. 4.24).

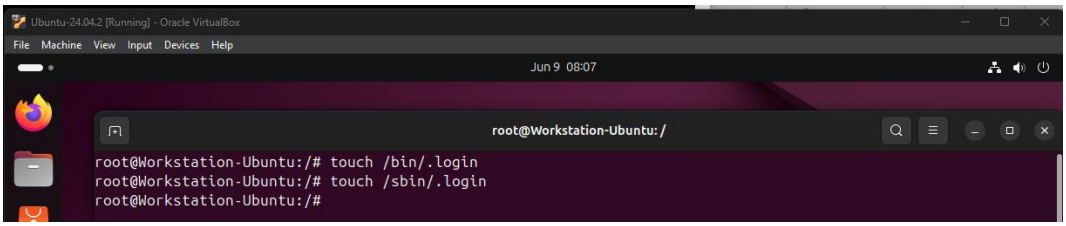


Рисунок 4.24 – Імітування потенційно шкідливих файлів у системних директоріях

Після перезапуску Wazuh Agent на кінцевому пристрої з ОС Linux відбулося заплановане сканування модуля Rootcheck, та було успішно виявлено імітовані файли пов’язані з ПЗ типу Rootkit (рис. 4.25).

	timestamp	agent.name	data.title	rule.description	rule.level	rule.id
	Jun 9, 2025 @ 08:08:38.013	Ubuntu-24.04.2	Rootkit 'Miscellaneous' detected by the presence of file '/sbin/.login'.	Host-based anomaly detection event (rootcheck).	7	510
	Jun 9, 2025 @ 08:08:38.010	Ubuntu-24.04.2	Rootkit 'Miscellaneous' detected by the presence of file '/bin/.login'.	Host-based anomaly detection event (rootcheck).	7	510
	Jun 9, 2025 @ 08:08:38.010	Ubuntu-24.04.2	Rootkit 'LDP' detected by the presence of file '/bin/.login'.	Host-based anomaly detection event (rootcheck).	7	510

Рисунок 4.25 – Результат роботи механізмів виявлення та реагування на загрози

Таким чином ми запевнились що модуль Rootcheck ефективно функціонує з оновленим списком шкідливого програмного забезпечення, зокрема було успішно виявлено імітовані шкідливі файли типу Rootkit на кінцевому пристрої з

операційною системою Linux. На інших кінцевих пристроях з іншими операційними системами модуль Rootcheck працює аналогічним чином.

Проведення тестування реагування на SSH атаки грубої сили відбувалося методом перевищення допустимої кількості спроб аутентифікації, що викликало спрацювання попередньо налаштованих правил виявлення та активної відповіді для блокування IP адреси джерела (рис. 4.26 – рис. 4.27).

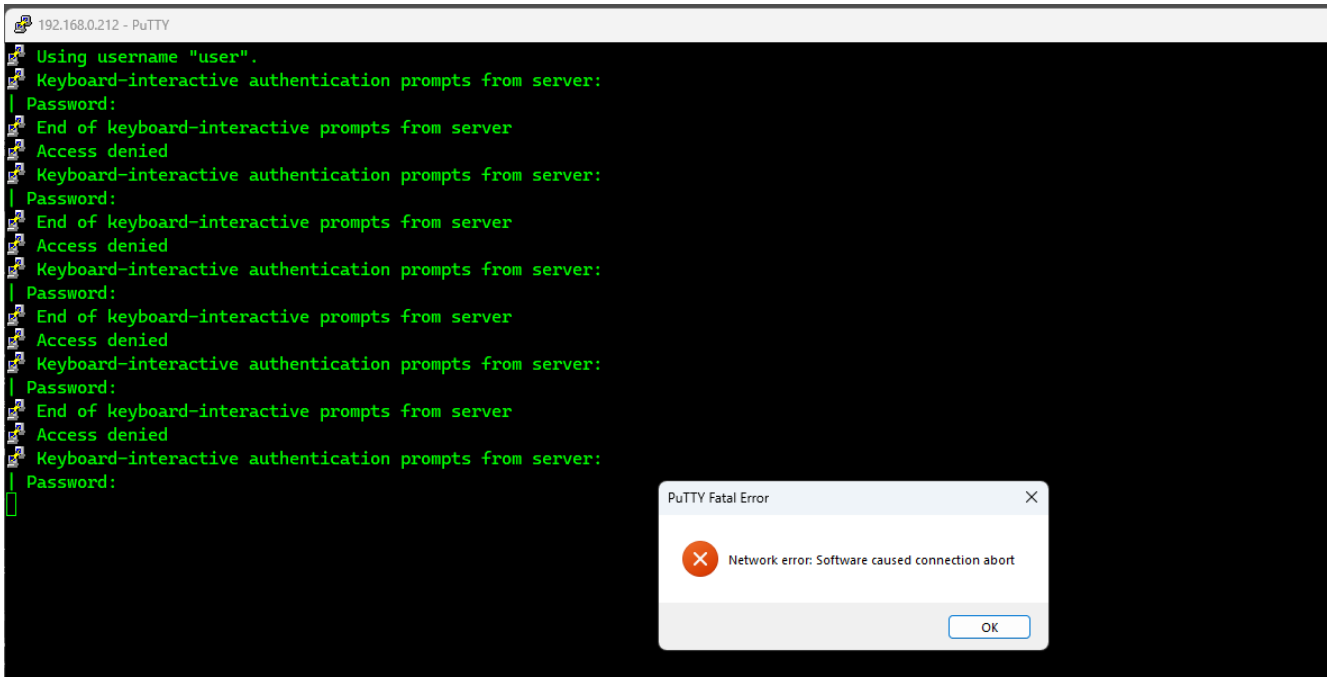
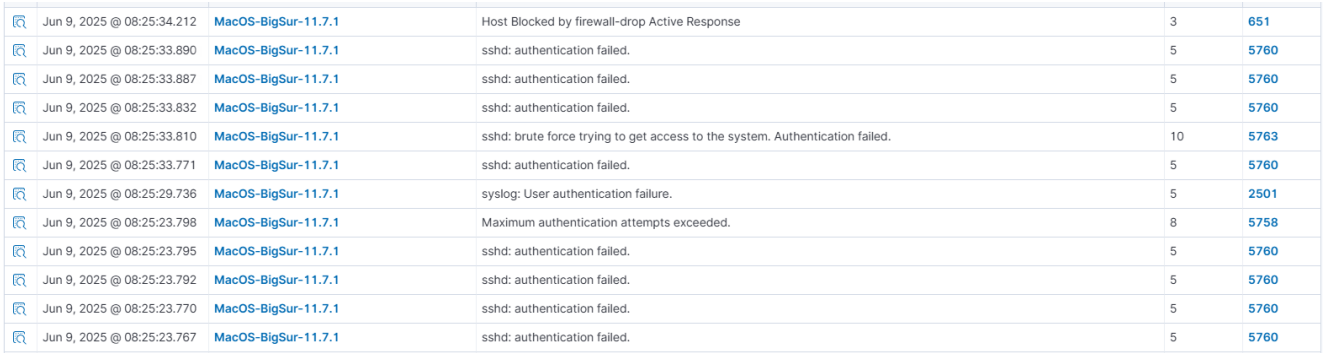


Рисунок 4.26 – Перевищення допустимої кількості спроб аутентифікації



	Jun 9, 2025 @ 08:25:34.212	MacOS-BigSur-11.7.1	Host Blocked by firewall-drop Active Response	3	651
	Jun 9, 2025 @ 08:25:33.890	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:33.887	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:33.832	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:33.810	MacOS-BigSur-11.7.1	sshd: brute force trying to get access to the system. Authentication failed.	10	5763
	Jun 9, 2025 @ 08:25:33.771	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:29.736	MacOS-BigSur-11.7.1	syslog: User authentication failure.	5	2501
	Jun 9, 2025 @ 08:25:23.798	MacOS-BigSur-11.7.1	Maximum authentication attempts exceeded.	8	5758
	Jun 9, 2025 @ 08:25:23.795	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:23.792	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:23.770	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760
	Jun 9, 2025 @ 08:25:23.767	MacOS-BigSur-11.7.1	sshd: authentication failed.	5	5760

Рисунок 4.27 – Результат роботи механізмів виявлення та реагування на загрози

Попередньо створений скрипт активної відповіді успішно заблокував IP адресу зловмисника, чим унеможливив подальші спроби підключення до кінцевого пристрою (рис. 4.28).

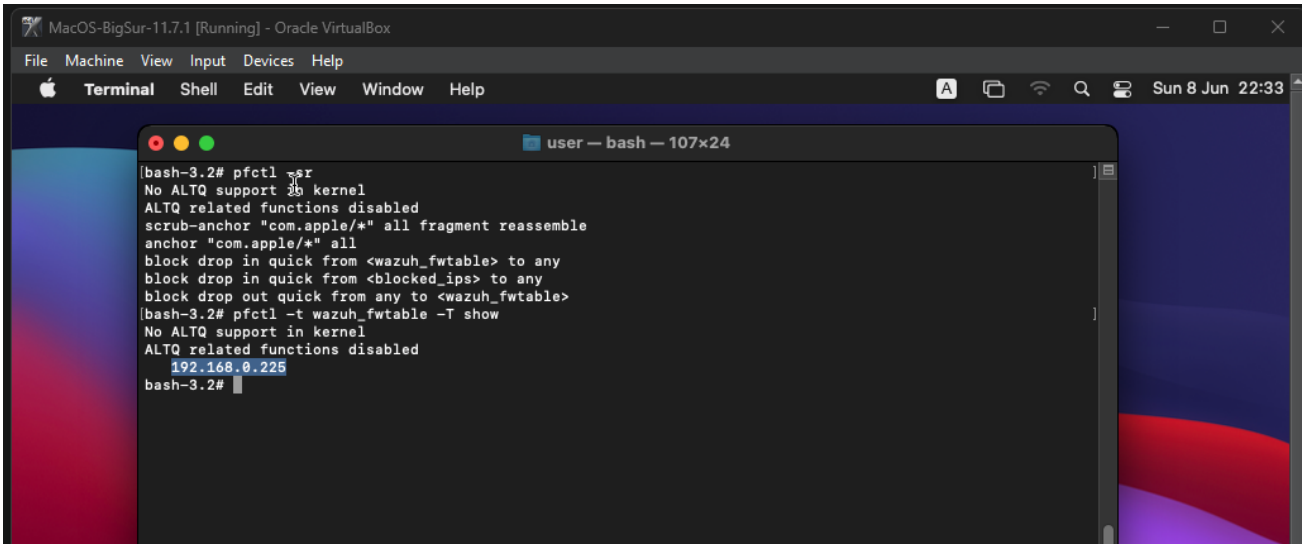


Рисунок 4.28 – Успішно заблокована IP адреса зловмисника

Таким чином ми запевнились що атаки на SSH типу грубої сили ефективно виявляються та зловмисні IP адреси блокуються. Також варто зазначити, що на кінцевому пристрої з операційною системою Linux виявлення та блокування працює аналогічним чином.

Варто також звернути увагу на попередньо створену інформаційну дошку SSH Events Dashboard, вона була успішно адаптована та оновлена інформацією отриманою внаслідок тестування атак грубої сили на SSH (рис. 4.29).

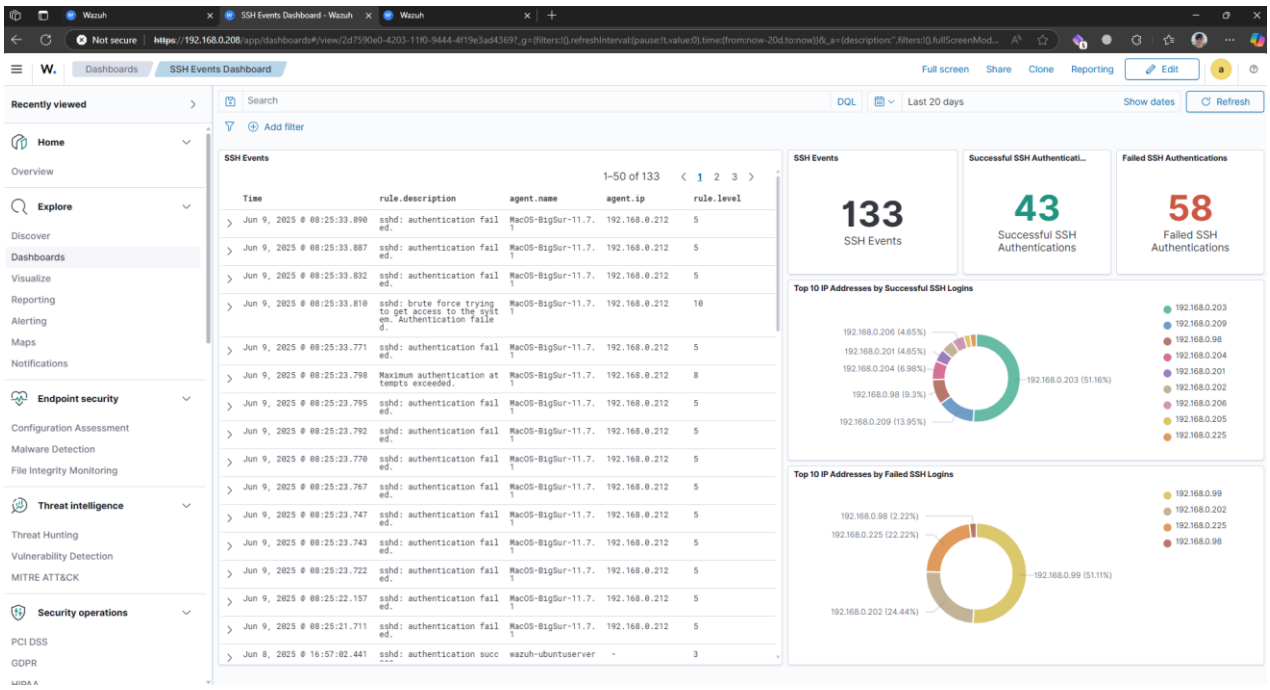


Рисунок 4.29 – Інформаційна панель SSH Events Dashboard

4.2. Практичне використання додаткових модулів Wazuh в повсякденних завданнях SOC

У межах повсякденної діяльності операційного центру безпеки значну роль відіграють також додаткові модулі системи Wazuh, які забезпечують розширене бачення стану інфраструктури, сприяють проактивному виявленню вразливостей та забезпечують відповідність системи встановленим політикам безпеки.

Зокрема для модуля System Inventory один з практичних сценаріїв використання є проведення процесу інвентаризації програмного забезпечення на всіх або окремих кінцевих пристроях організації.

Процес інвентаризації є критично важливим, адже він надає змогу проаналізувати встановлені програмні забезпечення та виявити потенційно небажані, або заборонені в організації.

До прикладу, відкривши сторінку інвентаризації певного кінцевого пристрою в нас є можливість згенерувати та завантажити звіт (рис. 4.30).

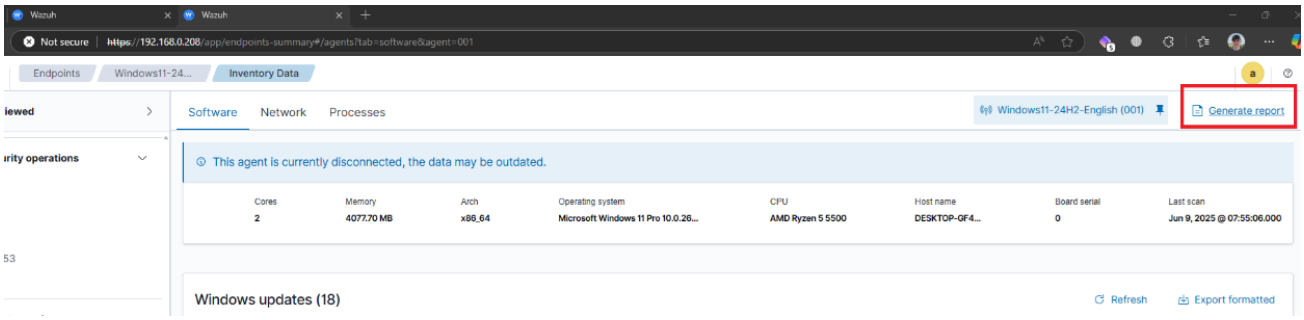


Рисунок 4.30 – Опція для генерування звіту інвентаризації

Згенерований звіт містить детальну інформацію про встановлені програмні застосунки, запущені процеси, відкриті мережеві порти, мережеві інтерфейси, список оновлень операційної системи та іншу важливу інформацію, яка може бути використана в інших цілях.

У випадку процесу інвентаризації може бути використана надана інформація про програмне забезпечення, в свою чергу може бути визначено таке програмне забезпечення яке має бути видаленим, відповідно до внутрішніх політик організації (рис. 4.31).

Operating system information

- x86_64
- Microsoft Windows 11 Pro 10.0.26100.4061

Packages

Name	Architecture	Version	Vendor
Oracle VirtualBox Guest Additions 7.1.4	x86_64	7.1.4.165100	Oracle and/or its affiliates
Wazuh Agent	i686	4.12.0	Wazuh, Inc.
Notepad++ (64-bit x64)	x86_64	8.8	Notepad++ Team
Python 3.13.3 Executables (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Development Libraries (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Documentation (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Core Interpreter (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Add to Path (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Tcl/Tk Support (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Standard Library (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 Test Suite (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python 3.13.3 pip Bootstrap (64-bit)	x86_64	3.13.3150.0	Python Software Foundation
Python Launcher	i686	3.13.3150.0	Python Software Foundation
Microsoft Edge WebView2 Runtime	i686	137.0.3296.68	Microsoft Corporation
Microsoft Edge	i686	137.0.3296.68	Microsoft Corporation

Copyright © Wazuh, Inc.

Рисунок 4.31 – Визначення небажаного ПЗ під час інвентаризації

У випадку використання модуля Security Configuration Assessment існують практичні сценарії виконання завдань по регулярному усуненню критичних недоліків конфігурацій важливих кінцевих пристроїв, так як до прикладу серверів.

Модуль Security Configuration Assessment надає змогу зручно переглянути та ознайомитися з можливими шляхами усунення наявних недоліків конфігурації (рис. 4.32).

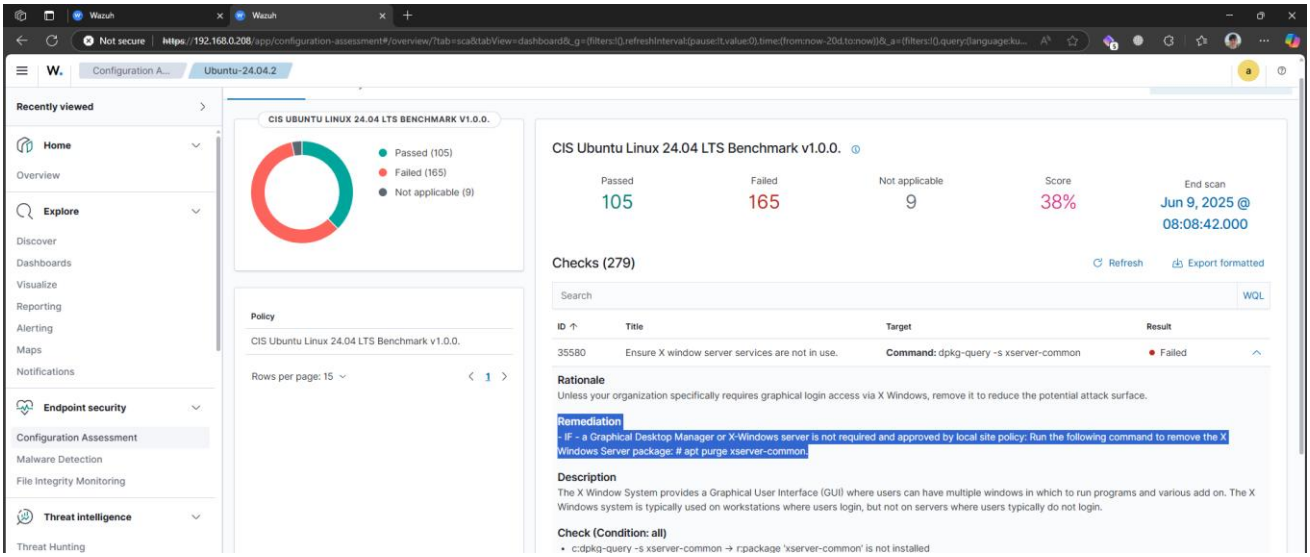


Рисунок 4.32 – Управління недоліками конфігурації пристроїв

Зокрема для модуля Vulnerability Detection одним із найбільш практичних сценаріїв використання є процес виявлення відомих вразливостей у встановленому програмному забезпеченні кінцевих пристроїв організації. Цей модуль аналізує зібрані дані інвентаризації та співставляє їх із загальновідомими базами вразливостей, зокрема базою даних NVD, що дозволяє своєчасно виявляти компоненти з відомими CVE записами [38].

Відкривши інтерфейс виявлених вразливостей, ми можемо переглянути перелік найбільш критичних вразливостей на усіх кінцевих пристроях, їх критичність та номер запису в базі даних CVE (рис. 4.33).

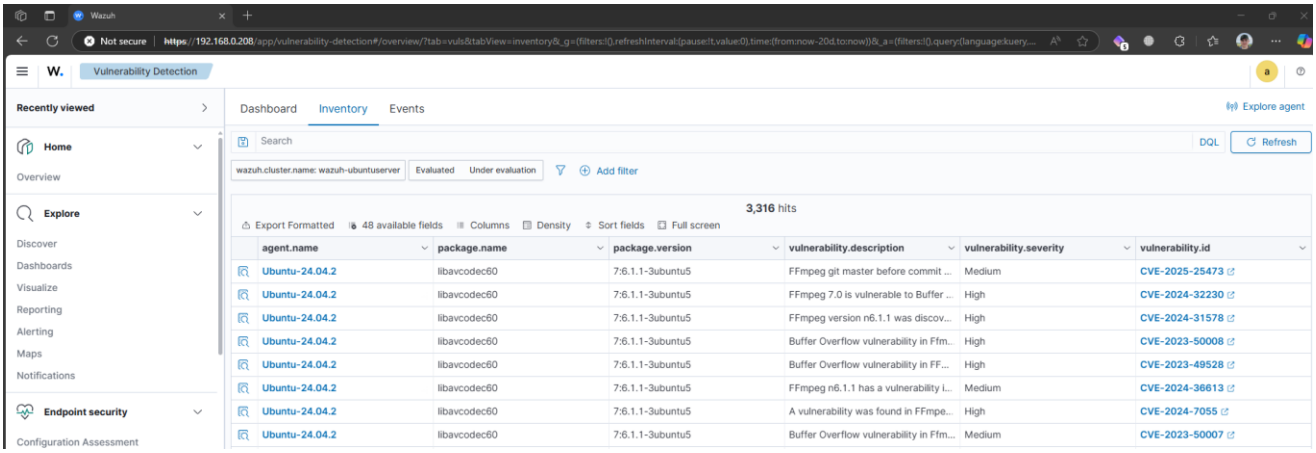


Рисунок 4.33 – Список найбільш критичних вразливостей

Крім того, важливо зауважити, що модуль Vulnerability Detection надає змогу отримувати детальну інформацію про вразливості за допомогою сервісу Wazuh Vulnerability Explorer (рис. 4.34).

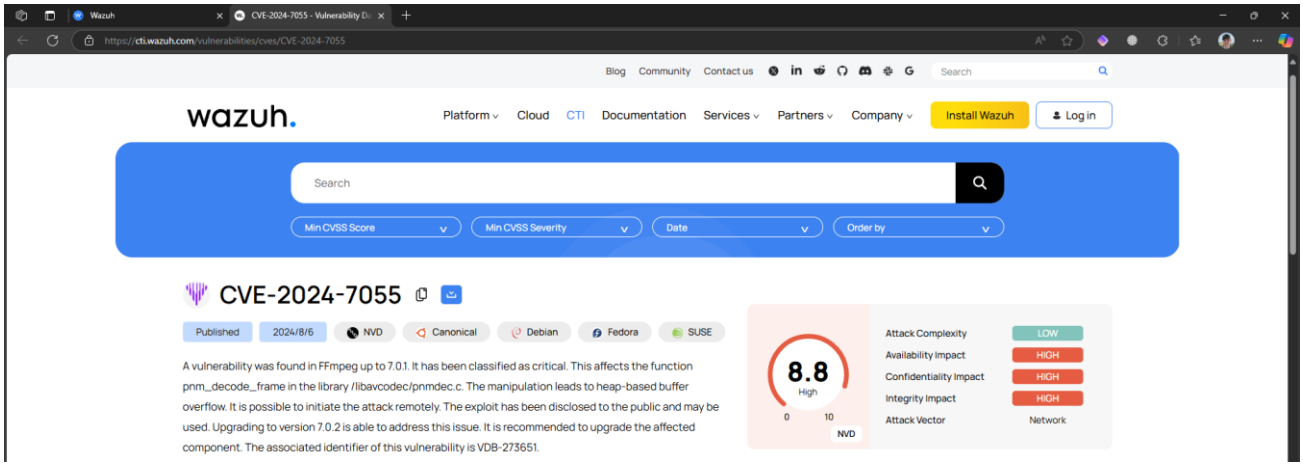


Рисунок 4.34 – Інформація про вразливість у Wazuh Vulnerability Explorer

Висновки до четвертого розділу

У четвертому розділі було проведено комплексні тестування практично реалізованої SIEM та XDR підсистеми на базі Wazuh, а саме тестування механізмів виявлення та реагування на загрози з використанням додаткових інтеграцій.

У ході тестування було здійснено перевірку функціонування механізмів розробленої системи, зокрема модуля контролю цілісності файлів, інтеграцій із зовнішніми репутаційними сервісами VirusTotal та AbuseIPDB, засобів моніторингу командної активності з використанням Sysmon та AuditD, а також механізмів виявлення аномальної мережевої активності.

Результати тестування продемонстрували здатність системи своєчасно виявляти широкий спектр типових кіберзагроз, зокрема виконання шкідливих команд, використання технік експлуатації PowerShell, активність характерну для програмного забезпечення типу Rootkit, спроби атак методом грубої сили, а також встановлення з’єднань із шкідливими IP адресами. Крім того, налаштування модуля Active Response підтвердило ефективність автоматичного реагування, зокрема блокуванням зловмисних IP адрес та видаленням шкідливих файлів.

Додаткову увагу приділено практичному застосуванню додаткових модулів Wazuh у повсякденній діяльності операційного центру безпеки. Модулі System Inventory, Security Configuration Assessment та Vulnerability Detection дозволили протестувати поверхневу інвентаризацію системи, перевірку відповідності конфігурацій безпеки, а також виявлення відомих вразливостей. Це надає можливість розширює функціональність системи від пасивного моніторингу до активного управління ризиками.

Таким чином, у четвертому розділі було проведено тестування Wazuh SIEM та XDR підсистеми, яке підтвердило її відповідність проєктним вимогам, а також продемонструвало практичну ефективність у виявленні та реагуванні на поширені кіберзагрози.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було опрацьовано теоретичні основи побудови SIEM та XDR підсистеми для SOC. Досліджено функції, завдання, архітектуру SOC, а також актуальні проблеми його функціонування. Розглянуто концепції технологій SIEM і XDR, їхні переваги, особливості та можливості інтеграції. Проведено огляд та порівняльний аналіз доступних платформ для моніторингу інформаційної безпеки, що дозволило обґрунтувати вибір Wazuh як основи для подальшого проєктування підсистеми.

Було спроєктовано проєкт SIEM та XDR підсистеми для SOC на базі Wazuh з урахуванням типових кіберзагроз. Реалізовано механізми виявлення шкідливого програмного забезпечення, аномальних мережевих з'єднань, атак типу грубої сили, шкідливих скриптів PowerShell, а також загроз, пов'язаних з Rootkit. Впроваджено засоби реагування на загрози, зокрема автоматичне видалення шкідливих файлів, блокування зловмисних IP адрес, реагування на SSH атаки. Додатково налаштовано модулі інвентаризації систем, оцінки конфігурацій безпеки та управління вразливостями, а також інформаційні панелі для оперативного моніторингу.

Практичне тестування підтвердило ефективність створеної SIEM та XDR підсистеми у виявленні типових загроз та забезпеченні автоматизованого реагування. Наведено приклади практичного використання додаткових модулів Wazuh в повсякденних завданнях SOC.

У результаті виконаної кваліфікаційної роботи були здобуті поглиблені знання про SIEM та XDR платформу Wazuh, її архітектуру та принципи роботи в контексті SOC. Була досконально вивчена структура архітектури SOC, включаючи основні компоненти, які взаємодіють у процесі моніторингу, виявлення та реагування на загрози. Отримані знання та практичні навички дозволили ефективно спроєктувати, реалізувати та протестувати SIEM та XDR підсистему на базі Wazuh. Створена підсистема може бути використана як основа для побудови SOC у невеликих та середніх організаціях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Is Elastic Stack (ELK) the Best SIEM Tool? | BitLyft Cybersecurity [Електронний ресурс] // Managed Security Services | MSSP | BitLyft Cybersecurity. – Режим доступу: https://www.bitlyft.com/resources/is-elastic-stack-elk-the-best-siem-tool (дата звернення: 01.05.2025). – Назва з екрана.
2. Використання розширеного виявлення та реагування (XDR) для готовності до аудиту CMMC [Електронний ресурс] // Enterprise Risk Management & GRC Solution Provider: Continuum GRC. – Режим доступу: https://continuumgrc.com/uk/leveraging-extended-detection-and-response-xdr-for-cmmc-audit-readiness/ (дата звернення: 03.05.2025). – Назва з екрана.
3. CoreWin. Benefits with SIEM, XDR, Wazuh. [Електронний ресурс] // CoreWin. – Режим доступу: https://corewin.ua/ceo-blog/benefits-with-siem-xdr-wazuh/ (дата звернення: 03.05.2025). – Назва з екрана.
4. Захист від програм-вимагачів на Windows за допомогою Wazuh [Електронний ресурс] // Corewin. – Режим доступу: https://corewin.ua/blog/ransomware-protection-on-windows-with-wazuh/ (дата звернення: 03.05.2025). – Назва з екрана.
5. What is XDR? Extended Detection & Response | CrowdStrike [Електронний ресурс] // CrowdStrike: We Stop Breaches with AI-native Cybersecurity. – Режим доступу: https://www.crowdstrike.com/en-us/cybersecurity-101/endpoint-security/extended-detection-and-response-xdr/ (дата звернення: 04.05.2025). – Назва з екрана.
6. What Is Extended Detection and Response (XDR)? XDR Security Guide [Електронний ресурс] // All-in-One Cybersecurity Platform - Cynet. – Режим доступу: https://www.cynet.com/xdr-security/understanding-xdr-security-concepts-features-and-use-cases/ (дата звернення: 04.05.2025). – Назва з екрана.
7. Розширене виявлення і реагування (Extended Detection and Response - XDR) | ESKA [Електронний ресурс] // ESKA. – Режим доступу:

https://eska.global/solutions/xdr (дата звернення: 05.05.2025). – Назва з екрана.

8. Центр безпеки (Security Operation Center, SOC) на варті кібербезпеки | Блог ESKA [Електронний ресурс] // ESKA. – Режим доступу: https://eska.global/blog/centr-bezpeki-security-operation-center-soc-na-varti-kiberbezpeki/ (дата звернення: 06.05.2025). – Назва з екрана.
9. Що таке управління інформаційною безпекою та подіями (SIEM) і чому це важливо? | ESKA Блог [Електронний ресурс] // ESKA. – Режим доступу: https://eska.global/blog/sho-take-upravlinnya-informacijnoyu-bezpekoyu-ta-podiyami-siem-i-chomu-ce-vazhlivo (дата звернення: 10.05.2025). – Назва з екрана.
10. What is XDR? Extended Detection and Response Security | Fortinet [Електронний ресурс] // Fortinet. – Режим доступу: https://www.fortinet.com/resources/cyberglossary/what-is-XDR (дата звернення: 10.05.2025). – Назва з екрана.
11. SOC: що це таке і навіщо воно потрібне? [Електронний ресурс] // Comprehensive service it company Kiev | Indevlab. – Режим доступу: https://indevlab.com/uk/blog-ua/soc-scho-tse-take-i-navischo-vono-potribne/ (дата звернення: 10.05.2025). – Назва з екрана.
12. Що таке SIEM і чому вона потрібна вашому бізнесу [Електронний ресурс] // Klik Ukraine Support Support - IT сервіс провайдер. – Режим доступу: https://www.kliksolutions.com.ua/great-info/shho-take-siem-i-chomu-vona-potribna-vashomu-biznesu/ (дата звернення: 11.05.2025). – Назва з екрана.
13. Що таке SIEM? | Захисний комплекс Microsoft [Електронний ресурс] // Your request has been blocked. This could be due to several reasons. – Режим доступу: https://www.microsoft.com/uk-ua/security/business/security-101/what-is-siem#SIEM-capabilities-and-use-cases (дата звернення: 13.05.2025). – Назва з екрана.

14. Що таке XDR? (Розширене виявлення й реагування) | Захисний комплекс Microsoft [Електронний ресурс] // Microsoft – AI, Cloud, Produktivität, Computing, Gaming und Apps. – Режим доступу: https://www.microsoft.com/uk-ua/security/business/security-101/what-is-xdr (дата звернення: 14.05.2025). – Назва з екрана.
15. SOCFortress. Detecting Abnormal Network Ports With Wazuh [Електронний ресурс] / SOCFortress // Medium. – Режим доступу: https://socfortress.medium.com/detecting-abnormal-network-ports-with-wazuh-6188ee8ed156 (дата звернення: 14.05.2025). – Назва з екрана.
16. EDR vs XDR vs SIEM: Який інструмент підійде саме вам? |SoftwareOne blog [Електронний ресурс] // SoftwareOne. – Режим доступу: https://www.softwareone.com/uk-ua/blog/articles/2024/07/23/edr-vs-exd-vs-siem-what-tool-is-right-for-you (дата звернення: 15.06.2025). – Назва з екрана.
17. tricksonflicks. SECURITY OPERATION CENTRE [Електронний ресурс] / tricksonflicks // Tricksonflicks. – Режим доступу: https://tricksonflicks.blogspot.com/2018/03/security-operation-centre.html (дата звернення: 17.05.2025). – Назва з екрана.
18. Active Response - Capabilities Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/active-response/index.html (дата звернення: 18.05.2025). – Назва з екрана.
19. File integrity monitoring - Capabilities Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/file-integrity/index.html (дата звернення: 18.05.2025). – Назва з екрана.
20. VirusTotal integration - Malware detection Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/malware-

detection/virus-total-integration.html (дата звернення: 20.05.2025). – Назва з екрана.

21. Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/ (дата звернення: 21.05.2025). – Назва з екрана.
22. Wazuh - Open Source XDR. Open Source SIEM. [Електронний ресурс] // Wazuh. – Режим доступу: https://wazuh.com/ (дата звернення: 21.05.2025). – Назва з екрана.
23. Учасники проектів Вікімедіа. Security information and event management (SIEM) – Вікіпедія [Електронний ресурс] / Учасники проектів Вікімедіа // Вікіпедія. – Режим доступу: https://uk.wikipedia.org/wiki/Security_information_and_event_management_(SIEM) (дата звернення: 21.05.2025). – Назва з екрана.
24. Dubetcky O. Як побудувати та запустити центр безпеки (SOC) [Електронний ресурс] / Oleh Dubetcky // Medium. – Режим доступу: https://oleg-dubetcky.medium.com/як-побудувати-та-запустити-центр-безпеки-soc-504266cefa53 (дата звернення: 22.05.2025). – Назва з екрана.
25. Windows Defender logs collection - Malware detection [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/malware-detection/win-defender-logs-collection.html (дата звернення: 23.05.2025). – Назва з екрана.
26. Monitoring execution of malicious commands - Proof of Concept guide [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/proof-of-concept-guide/audit-commands-run-by-user.html (дата звернення: 24.05.2025). – Назва з екрана.
27. SOCFortress. Detecting Abnormal Network Ports With Wazuh [Електронний ресурс] / SOCFortress // Medium. – Режим доступу:

https://socfortress.medium.com/detecting-abnormal-network-ports-with-wazuh-6188ee8ed156 (дата звернення: 26.05.2025). – Назва з екрана.

28. Taylor Walton. Detecting Abnormal Network Connections With Wazuh [Електронний ресурс], 2022 / Taylor Walton // YouTube. – Режим доступу: https://www.youtube.com/watch?v=M_OfZ7jiQEU (дата звернення: 26.05.2025). – Назва з екрана.
29. Detecting known bad actors with Wazuh and AbuseIPDB | Wazuh [Електронний ресурс] // Wazuh. – Режим доступу: https://wazuh.com/blog/detecting-known-bad-actors-with-wazuh-and-abuseipdb/ (дата звернення: 26.05.2025). – Назва з екрана.
30. Detecting PowerShell exploitation techniques in Windows using Wazuh | Wazuh [Електронний ресурс] // Wazuh. – Режим доступу: https://wazuh.com/blog/detecting-powershell-exploitation-techniques-in-windows-using-wazuh/ (дата звернення: 27.05.2025). – Назва з екрана.
31. Rootkits behavior detection - Malware detection Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/malware-detection/rootkits-behavior-detection.html (дата звернення: 28.05.2025). – Назва з екрана.
32. Bassett S. Security Monitoring with Wazuh: A Hands-On Guide to Effective Enterprise Security Using Real-life Use Cases in Wazuh / Santiago Bassett. – [Б. м.] : Packt Publishing, Limited, 2024.
33. Blocking SSH brute-force attack with Active Response - Use cases [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/active-response/ar-use-cases/blocking-ssh-brute-force.html (дата звернення: 02.06.2025). – Назва з екрана.
34. Petersen R. Ubuntu 20.04 LTS Server: Administration and Reference / Richard Petersen. – [Б. м.] : Surfing Turtle Press, 2020. – 627 с.

35. System inventory - Capabilities Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/system-inventory/index.html (дата звернення: 03.06.2025). – Назва з екрана.
36. Rastogi S. Exploring macOS: A Journey Through the Mac Ecosystem : навч. посіб. / Sagar Rastogi, Jasdeep Singh. – [Б. м.] : Springer, 2023. – 372 с.
37. Security Configuration Assessment - Capabilities Â· Wazuh documentation [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/user-manual/capabilities/sec-config-assessment/index.html (дата звернення: 02.07.2025). – Назва з екрана.
38. GDPR IV, Controller and processor [Електронний ресурс] // Wazuh documentation. – Режим доступу: https://documentation.wazuh.com/current/compliance/gdpr/gdpr-IV.html (дата звернення: 07.06.2025). – Назва з екрана.

ДОДАТКИ

Повний конфігураційний файл Wazuh ossec.conf

```
<!--
  Wazuh - Manager - Default configuration for ubuntu 24.04
  More info at: https://documentation.wazuh.com
  Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
  <global>
    <jsonout_output>yes</jsonout_output>
    <alerts_log>yes</alerts_log>
    <logall>no</logall>
    <logall_json>no</logall_json>

<email_notification>no</email_notification>

<smtp_server>smtp.example.wazuh.com</smtp_server>

<email_from>wazuh@example.wazuh.com</email_from>

<email_to>recipient@example.wazuh.com</email_to>

<email_maxperhour>12</email_maxperhour>

<email_log_source>alerts.log</email_log_source>

<agents_disconnection_time>10m</agents_disconnection_time>

<agents_disconnection_alert_time>0</agents_disconnection_alert_time>
    <update_check>yes</update_check>
  </global>

  <integration>
    <name>custom-abuseipdb.py</name>

<hook_url>https://api.abuseipdb.com/api/v2/check</hook_url>

<api_key>06aecd899948a1f148bf5bad57d5693c395fa40783eca98f5efd1b21bbf8f50362b4b7269330b0c1</api_key>

<rule_id>100221,100222,102505,102506,102507,102508</rule_id>
    <alert_format>json</alert_format>
  </integration>

  <integration>
    <name>virustotal</name>

<api_key>ea3eea752fa90f31bf71b00a1ae6b0fe410793937da4cff9277706666146326b</api_key>
    <group>syscheck</group>
    <alert_format>json</alert_format>
  </integration>

  <alerts>
    <log_alert_level>3</log_alert_level>

<email_alert_level>12</email_alert_level>
  </alerts>

  <!-- Choose between "plain", "json", or "plain,json" for the format of internal logs -->
  <logging>
    <log_format>plain</log_format>
  </logging>

  <remote>
    <connection>secure</connection>
    <port>1514</port>
    <protocol>tcp</protocol>
    <queue_size>131072</queue_size>
  </remote>

  <!-- Policy monitoring -->
  <rootcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>yes</check_dev>
    <check_sys>yes</check_sys>
    <check_pids>yes</check_pids>
    <check_ports>yes</check_ports>
    <check_if>yes</check_if>

    <!-- Frequency that rootcheck is executed - every 12 hours -->
    <frequency>43200</frequency>


<rootkit_files>etc/rootcheck/rootkit_files.txt</rootkit_files>

<rootkit_trojans>etc/rootcheck/rootkit_trojans.txt</rootkit_trojans>

    <skip_nfs>yes</skip_nfs>

    <ignore>/var/lib/containerd</ignore>

<ignore>/var/lib/docker/overlay2</ignore>
  </rootcheck>

  <wodle name="cis-cat">
    <disabled>yes</disabled>
    <timeout>1800</timeout>
    <interval>1d</interval>
    <scan-on-start>yes</scan-on-start>

    <java_path>wodles/java</java_path>

<ciscat_path>wodles/ciscat</ciscat_path>
  </wodle>

  <!-- Osquery integration -->
  <wodle name="osquery">
    <disabled>yes</disabled>
    <run_daemon>yes</run_daemon>

<log_path>/var/log/osquery/osqueryd.results.log</log_path>

<config_path>/etc/osquery/osquery.conf</config_path>
    <add_labels>yes</add_labels>
  </wodle>

  <!-- System inventory -->
  <wodle name="syscollector">
    <disabled>no</disabled>
    <interval>1m</interval>
    <scan_on_start>yes</scan_on_start>
    <hardware>yes</hardware>
    <os>yes</os>
    <network>yes</network>
    <packages>yes</packages>
    <ports all="yes">yes</ports>
    <processes>yes</processes>

    <!-- Database synchronization settings -->
    <synchronization>
      <max_eps>10</max_eps>
    </synchronization>
  </wodle>

  <sca>
    <enabled>yes</enabled>
    <scan_on_start>yes</scan_on_start>
    <interval>6h</interval>
    <skip_nfs>yes</skip_nfs>
  </sca>

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

  <indexer>
    <enabled>yes</enabled>
    <hosts>
      <host>https://127.0.0.1:9200</host>
    </hosts>
    <ssl>
      <certificate_authorities>
        <ca>/etc/filebeat/certs/root-ca.pem</ca>
      </certificate_authorities>
      <certificate>/etc/filebeat/certs/wazuh-server.pem</certificate>
      <key>/etc/filebeat/certs/wazuh-server-key.pem</key>
    </ssl>
  </indexer>

  <!-- File integrity monitoring -->
  <syscheck>
    <disabled>no</disabled>

    <!-- Frequency that syscheck is executed default every 12 hours -->
    <frequency>43200</frequency>

    <scan_on_start>yes</scan_on_start>

    <!-- Generate alert when new file detected -->
    <alert_new_files>yes</alert_new_files>

    <!-- Don’t ignore files that change more than ‘frequency’ times -->
    <auto_ignore frequency="10" timeframe="3600">no</auto_ignore>

    <!-- Directories to check  (perform all possible verifications) -->

<directories>/etc,/usr/bin,/usr/sbin</directories>
```

```
<directories>/bin,/sbin,/boot</directories>

    <!-- Files/directories to ignore -->
    <ignore>/etc/mtab</ignore>
    <ignore>/etc/hosts.deny</ignore>
    <ignore>/etc/mail/statistics</ignore>
    <ignore>/etc/random-seed</ignore>
    <ignore>/etc/random.seed</ignore>
    <ignore>/etc/adjtime</ignore>
    <ignore>/etc/httpd/logs</ignore>
    <ignore>/etc/utmpx</ignore>
    <ignore>/etc/wtmpx</ignore>
    <ignore>/etc/cups/certs</ignore>
    <ignore>/etc/dumpdates</ignore>
    <ignore>/etc/svc/volatile</ignore>

    <!-- File types to ignore -->
    <ignore
type="sregex">.log$|.swp$</ignore>

    <!-- Check the file, but never compute
the diff -->
    <nodiff>/etc/ssl/private.key</nodiff>

    <skip_nfs>yes</skip_nfs>
    <skip_dev>yes</skip_dev>
    <skip_proc>yes</skip_proc>
    <skip_sys>yes</skip_sys>

    <!-- Nice value for Syscheck process -->
    <process_priority>10</process_priority>

    <!-- Maximum output throughput -->
    <max_eps>50</max_eps>

    <!-- Database synchronization settings --
>
    <synchronization>
      <enabled>yes</enabled>
      <interval>5m</interval>
      <max_eps>10</max_eps>
    </synchronization>
  </syscheck>

  <!-- Active response -->
  <global>
    <white_list>127.0.0.1</white_list>

<white_list>^localhost.localdomain$</whit
e_list>
    <white_list>127.0.0.53</white_list>
  </global>

  <command>
    <name>disable-account</name>
    <executable>disable-
account</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>restart-wazuh</name>
    <executable>restart-wazuh</executable>
  </command>

  <command>
    <name>firewall-drop-linux-
macos</name>
    <executable>firewall-drop</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>abuseipdb-block-ip-linux-
macos</name>
    <executable>abuseipdb-block-
ip.py</executable>

<timeout_allowed>no</timeout_allowed>
  </command>

  <command>
    <name>abuseipdb-block-ip-
windows</name>
    <executable>abuseipdb-block-
ip.exe</executable>

<timeout_allowed>no</timeout_allowed>
  </command>

  <command>
    <name>host-deny</name>
    <executable>host-deny</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>route-null</name>
    <executable>route-null</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>win_route-null</name>
    <executable>route-null.exe</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>netsh</name>
    <executable>netsh.exe</executable>

<timeout_allowed>yes</timeout_allowed>
  </command>

  <command>
    <name>remove-threat-linux-
macos</name>
    <executable>remove-
threat.sh</executable>

<timeout_allowed>no</timeout_allowed>
  </command>

  <command>
    <name>remove-threat-windows</name>
    <executable>remove-
threat.exe</executable>

<timeout_allowed>no</timeout_allowed>
  </command>

  <active-response>
    <disabled>no</disabled>
    <command>remove-threat-linux-
macos</command>
    <location>local</location>
    <rules_id>87105</rules_id>
  </active-response>

  <active-response>
    <disabled>no</disabled>
    <command>remove-threat-
windows</command>
    <location>local</location>
    <rules_id>87105</rules_id>
  </active-response>

  <active-response>
    <disabled>no</disabled>
    <command>firewall-drop-linux-
macos</command>
    <location>local</location>
    <rules_id>5763</rules_id>
    <timeout>600</timeout>
  </active-response>

  <active-response>
    <disabled>no</disabled>
    <command>abuseipdb-block-ip-linux-
macos</command>
    <location>local</location>
    <rules_id>102515,102516</rules_id>
  </active-response>

  <active-response>
    <disabled>no</disabled>
    <command>abuseipdb-block-ip-
windows</command>
    <location>local</location>
    <rules_id>102515,102516</rules_id>
  </active-response>

  <!-- Log analysis -->
  <localfile>
    <log_format>command</log_format>
    <command>df -P</command>
    <frequency>360</frequency>
  </localfile>

  <localfile>

<log_format>full_command</log_format>
    <command>netstat -tulpn | sed
‘s/\([[:alnum:]]\+\)\ \+[[:digit:]]\+\
\+[[:digit:]]\+\ \+\(.*\):\([[:digit:]]*\)\ \+\([0-
9\.\:\*]\+\).\+\ \([[:digit:]]*\/[[:alnum:]\-
]*\).*/\1 \2 == \3 == \4 \5/’ | sort -k 4 -g |
sed ‘s/ == \(.*\) ==/:\1/’ | sed
1,2d</command>
    <alias>netstat listening ports</alias>
    <frequency>360</frequency>
  </localfile>

  <localfile>

<log_format>full_command</log_format>
    <command>last -n 20</command>
    <frequency>360</frequency>
  </localfile>

  <ruleset>
    <!-- Default ruleset -->

<decoder_dir>ruleset/decoders</decoder_di
r>
    <rule_dir>ruleset/rules</rule_dir>
    <rule_exclude>0215-
policy_rules.xml</rule_exclude>
    <list>etc/lists/audit-keys</list>
    <list>etc/lists/amazon/aws-
eventnames</list>
    <list>etc/lists/security-
eventchannel</list>

    <!-- Custom rulesets -->
    <list>etc/lists/suspicious-
commands</list>
    <list>etc/lists/common-ports</list>
    <list>etc/lists/malicious-
powershell</list>

    <!-- User-defined ruleset -->
```

```
<decoder_dir>etc/decoders</decoder_dir>
    <rule_dir>etc/rules</rule_dir>
  </ruleset>

  <rule_test>
    <enabled>yes</enabled>
    <threads>1</threads>
    <max_sessions>64</max_sessions>

<session_timeout>15m</session_timeout>
  </rule_test>

  <!-- Configuration for wazuh-authd -->
  <auth>
    <disabled>no</disabled>
    <port>1515</port>
    <use_source_ip>no</use_source_ip>
    <purge>yes</purge>
    <use_password>no</use_password>

<ciphers>HIGH:!ADH:!EXP:!MD5:!RC4:!
3DES:!CAMELLIA:@STRENGTH</ciphe
rs>
    <!-- <ssl_agent_ca></ssl_agent_ca> -->
    <ssl_verify_host>no</ssl_verify_host>

<ssl_manager_cert>etc/sslmanager.cert</ssl
_manager_cert>

<ssl_manager_key>etc/sslmanager.key</ssl
_manager_key>

<ssl_auto_negotiate>no</ssl_auto_negotiat
e>
  </auth>

  <cluster>
    <name>wazuh</name>
    <node_name>node01</node_name>
    <node_type>master</node_type>
    <key></key>
    <port>1516</port>
    <bind_addr>0.0.0.0</bind_addr>
    <nodes>
      <node>NODE_IP</node>
    </nodes>
    <hidden>no</hidden>
    <disabled>yes</disabled>
  </cluster>

</ossec_config>

<ossec_config>
  <localfile>
    <log_format>journald</log_format>
    <location>journald</location>
  </localfile>

  <localfile>
    <log_format>syslog</log_format>
    <location>/var/ossec/logs/active-
responses.log</location>
  </localfile>

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

</ossec_config>
```

200150-sysmon_for_linux_rules.xml

```
<group name="sysmon_linux,">
  <rule id="200150" level="3">
    <decoded_as>sysmon-
linux</decoded_as>
    <field
name="system.eventId">\.+</field>
    <description>Sysmon For Linux
Event</description>
    <mitre>
     <id>T1204</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 1-->
  <rule id="200151" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^1$</field>
    <description>Sysmon - Event 1:
Process creation
$(eventdata.image)</description>
    <group>sysmon_event1</group>
    <mitre>
     <id>T1204</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 3-->
  <rule id="200152" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^3$</field>
    <description>Sysmon - Event 3:
Network connection by
$(eventdata.image)</description>
    <group>sysmon_event3</group>
    <mitre>
     <id>T1043</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 5-->
  <rule id="200153" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^5$</field>
    <description>Sysmon - Event 5:
Process terminated
$(eventdata.image)</description>
    <group>sysmon_event5</group>
    <mitre>
     <id>T1204</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 9-->
  <rule id="200154" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^9$</field>
    <description>Sysmon - Event 9: Raw
Access Read by
$(eventdata.image)</description>
    <group>sysmon_event9</group>
    <mitre>
     <id>T1204</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 11-->
  <rule id="200155" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^11$</field>
    <description>Sysmon - Event 11:
FileCreate by
$(eventdata.image)</description>

<group>sysmon_event_11</group>
    <mitre>
     <id>T1044</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 16-->
  <rule id="200156" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^16$</field>
    <description>Sysmon - Event 16:
Sysmon config state changed
$(Event.EventData.Data.Configuration)</d
escription>
    <group>sysmon_event_16</group>
    <mitre>
     <id>T1562</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 23-->
  <rule id="200157" level="3">
    <if_sid>200150</if_sid>
    <field
name="system.eventId">^23$</field>
    <description>Sysmon - Event 23:
FileDelete (A file delete was detected) by
$(eventdata.image)</description>
    <group>sysmon_event_23</group>
    <mitre>
     <id>T1107</id>
     <id>T1485</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--Overrides-->
<!--EventID = 3. Discrad events if Image =
/var/ossec/bin/wazuh-agentd or Image =
/usr/sbin/zabbix_agentd-->
  <rule id="200200" level="1">
    <if_sid>200152</if_sid>
    <field
name="eventdata.image">wazuh-
agentd$|zabbix_agentd$</field>
    <description>Sysmon - Event 3:
Network connection by
$(eventdata.image)</description>
    <group>sysmon_event3</group>
    <mitre>
     <id>T1107</id>
     <id>T1485</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 11. Discrad events if Image
= /var/ossec/bin/wazuh-agentd-->
  <rule id="200201" level="1">
    <if_sid>200155</if_sid>
    <field
name="eventdata.image">wazuh-
agentd$</field>
    <description>Sysmon - Event 11:
FileCreate by
$(eventdata.image)</description>
    <group>sysmon_event_11</group>
    <mitre>
     <id>T1107</id>
     <id>T1485</id>
    </mitre>
    <options>no_full_log</options>
  </rule>
<!--EventID = 23. Discrad events if Image
= /var/ossec/bin/wazuh-agentd-->
  <rule id="200202" level="1">
    <if_sid>200157</if_sid>
    <field
name="eventdata.image">wazuh-
agentd$</field>
    <description>Sysmon - Event 23:
FileDelete (A file delete was detected) by
$(eventdata.image)</description>
    <group>sysmon_event_23</group>
    <mitre>
     <id>T1107</id>
     <id>T1485</id>
    </mitre>
    <options>no_full_log</options>
  </rule>

<!-- Frequency rule to capture 2 sysmon
event 3 alerts within of the same destination
ip within 5 seconds -->
<rule frequency="2" id="200203"
level="1" timeframe="60">

                <if_matched_sid>20015
2</if_matched_sid>

                <same_field>eventdata.
DestinationIp</same_field>

                <description>Multiple
Sysmon Level 3 alerts for same destination
IP.</description>

                <options>no_full_log</o
ptions>
                </rule>

</group>
```

custom-abuseipdb.py

```
#!/var/ossec/framework/python/bin/python3

import json
import sys
import time
import os
import sqlite3
from socket import socket, AF_UNIX, SOCK_DGRAM

try:
    import requests
except Exception:
    print("No module 'requests' found. Install: pip install
requests")
    sys.exit(1)

# === Configuration ===
CACHE_TTL_SECONDS = 172800  # Cache TTL: 2 days
debug_enabled = False

# === Paths ===
pwd =
os.path.dirname(os.path.dirname(os.path.realpath(__file__)))
log_file = f'{pwd}/logs/integrations.log'
socket_addr = f'{pwd}/queue/sockets/queue'
cache_db_path = '/var/ossec/etc/lists/abuseipdb_cache.db'
now = time.strftime("%a %b %d %H:%M:%S %Z %Y")

# === SQLite Caching ===
def init_cache_db():
    if not os.path.exists(cache_db_path):
        conn = sqlite3.connect(cache_db_path)
        c = conn.cursor()
        c.execute('''CREATE TABLE ip_cache (
                    ip TEXT PRIMARY KEY,
                    data TEXT,
                    timestamp INTEGER
                )''')
        conn.commit()
        conn.close()

def get_cached_data(ip):
    conn = sqlite3.connect(cache_db_path)
    c = conn.cursor()
    c.execute("SELECT data, timestamp FROM ip_cache
WHERE ip = ?", (ip,))
    row = c.fetchone()
    conn.close()
    if row:
        data_json, timestamp = row
        if int(time.time()) - timestamp <
CACHE_TTL_SECONDS:
            debug(f"# Found fresh cache for {ip}")
            return json.loads(data_json)
        else:
            debug(f"# Cache expired for {ip}")
    return None

def cache_ip_data(ip, data):
    conn = sqlite3.connect(cache_db_path)
    c = conn.cursor()
    c.execute("INSERT OR REPLACE INTO ip_cache (ip, data,
timestamp) VALUES (?, ?, ?)",
              (ip, json.dumps(data), int(time.time())))
    conn.commit()
    conn.close()

# === Main logic ===
def main(args):
    debug("# Starting")
    alert_file_location = args[1]
    apikey = args[2]

    with open(alert_file_location) as alert_file:
        json_alert = json.load(alert_file)

    msg = request_abuseipdb_info(json_alert, apikey)

    if msg:
        send_event(msg, json_alert.get("agent", {}))

def request_abuseipdb_info(alert, apikey):
    alert_output = {}

    srcip = (
        alert.get("data", {}).get("srcip") or
        alert.get("data", {}).get("eventdata",
{}).get("DestinationIp") or
        alert.get("data", {}).get("win", {}).get("eventdata",
{}).get("destinationIp") or
        alert.get("data", {}).get("remote_ip")
    )

    if not srcip:
        debug("# No source IP found in alert")
        return 0

    data = query_api(srcip, apikey)

    alert_output["abuseipdb"] = {
        "found": 0,
        "source": {
            "alert_id": alert.get("id"),
            "rule": alert.get("rule", {}).get("id"),
            "description": alert.get("rule", {}).get("description"),
            "full_log": alert.get("full_log"),
            "srcip": srcip
        }
    }
    alert_output["integration"] = "custom-abuseipdb"

    if in_database(data):
        alert_output["abuseipdb"]["found"] = 1
        abuse_confidence_score, country_code, usage_type, isp,
domain, total_reports, last_reported_at = collect(data)
        alert_output["abuseipdb"].update({
            "abuse_confidence_score": abuse_confidence_score,
            "country_code": country_code,
            "usage_type": usage_type,
            "isp": isp,
            "domain": domain,
            "total_reports": total_reports,
            "last_reported_at": last_reported_at
        })

    debug(alert_output)
    return alert_output
```

```
def query_api(srcip, apikey):
    cached = get_cached_data(srcip)
    if cached:
        return cached

    debug(f"# Querying AbuseIPDB for {srcip}")
    params = {‘maxAgeInDays’: ‘90’, ‘ipAddress’: srcip}
    headers = {
        "Accept-Encoding": "gzip, deflate",
        ‘Accept’: ‘application/json’,
        "Key": apikey
    }

    response =
requests.get(‘https://api.abuseipdb.com/api/v2/check’,
params=params, headers=headers)
    if response.status_code == 200:
        data = response.json()["data"]
        cache_ip_data(srcip, data)
        return data
    else:
        alert_output = {
            "abuseipdb": {
                "error": response.status_code,
                "description": response.json().get("errors",
[{}])[0].get("detail", "Unknown error")
            },
            "integration": "custom-abuseipdb"
        }
        debug("# Error querying AbuseIPDB")
        send_event(alert_output)
        exit(0)

def in_database(data):
    return data.get(‘totalReports’, 0) > 0

def collect(data):
    return (
        data[‘abuseConfidenceScore’],
        data[‘countryCode’],
        data[‘usageType’],
        data[‘isp’],
        data[‘domain’],
        data[‘totalReports’],
        data[‘lastReportedAt’]
    )

def send_event(msg, agent=None):
    if not agent or agent.get("id") == "000":
        string = f’1:abuseipdb:{json.dumps(msg)}’
    else:
        string = f’1:[{agent.get("id")}] ({agent.get("name")})
{agent.get("ip", "any")}->abuseipdb:{json.dumps(msg)}’

    debug(string)
    sock = socket(AF_UNIX, SOCK_DGRAM)
    sock.connect(socket_addr)
    sock.send(string.encode())
    sock.close()

def debug(msg):
    if debug_enabled:
        full_msg = f"{now}: {msg}\n"
        print(full_msg)
        with open(log_file, "a") as f:
            f.write(full_msg)

if __name__ == "__main__":
    try:
        if len(sys.argv) >= 3:
            debug_enabled = (len(sys.argv) > 3 and sys.argv[3] ==
‘debug’)
        else:
            debug("# Exiting: Bad arguments.")
            sys.exit(1)

        init_cache_db()
        main(sys.argv)

    except Exception as e:
        debug(str(e))
        raise
```

local_rules.xml

```
<!-- Local rules -->

<group name="netstat">
  <rule id="100221" level="2">
    <decoded_as>ossec</decoded_as>
    <field name="remote_port"
type="pcre2">^(80|443|8080|8443|8000|8888|8008)$</field>
    <description>Connection to web port detected</description>
    <group>ossec,netstat,</group>
  </rule>
  <rule id="100222" level="3">
    <decoded_as>ossec</decoded_as>
    <field name="remote_port"
type="pcre2">^(?!\*\.)\d+$</field>
    <list field="remote_port"
lookup="not_address_match_key">etc/lists/common-
ports</list>
    <description>Connection to uncommon port
detected</description>
    <group>ossec,netstat,</group>
  </rule>
</group>

<group name="audit">
  <rule id="100210" level="12">
    <if_sid>80792</if_sid>
    <list field="audit.command" lookup="match_key_value"
check_value="red">etc/lists/suspicious-commands</list>
    <description>Audit: Highly Suspicious Command executed:
$(audit.exe)</description>
    <group>audit_command,</group>
  </rule>

  <rule id="100211" level="12">
    <if_sid>80792</if_sid>
    <list field="audit.command" lookup="match_key_value"
check_value="orange">etc/lists/suspicious-commands</list>
    <description>Audit: Pretty Suspicious Command executed:
$(audit.exe)</description>
    <group>audit_command,</group>
  </rule>

  <rule id="100212" level="12">
    <if_sid>80792</if_sid>
    <list field="audit.command" lookup="match_key_value"
check_value="yellow">etc/lists/suspicious-commands</list>
    <description>Audit: Suspicious Command executed:
$(audit.exe)</description>
    <group>audit_command,</group>
  </rule>
</group>

<group name="windows,powershell,">
  <rule id="100201" level="8">
    <if_sid>60009</if_sid>
    <field name="win.eventdata.payload"
type="pcre2">(?i)CommandInvocation</field>
    <field name="win.system.message"
type="pcre2">(?i)EncodedCommand|FromBase64String|Encod
edArguments|-e\b|-enco\b|-en\b</field>
    <description>Encoded command executed via
PowerShell.</description>
    <mitre>
      <id>T1059.001</id>
      <id>T1562.001</id>
    </mitre>
  </rule>

  <rule id="100202" level="4">
    <if_sid>60009</if_sid>
    <field name="win.system.message"
type="pcre2">(?i)blocked by your antivirus software</field>
    <description>Windows Security blocked malicious
command executed via PowerShell.</description>
    <mitre>
      <id>T1059.001</id>
    </mitre>
  </rule>

  <rule id="100203" level="10">
    <if_sid>60009</if_sid>
    <field name="win.eventdata.payload"
type="pcre2">(?i)CommandInvocation</field>
    <field name="win.system.message" type="pcre2">(?i)Add-
Persistence|Find-AVSignature|Get-GPPAutologon|Get-
GPPPassword|Get-HttpStatus|Get-Keystrokes|Get-
SecurityPackages|Get-TimedScreenshot|Get-
VaultCredential|Get-VolumeShadowCopy|Install-SSP|Invoke-
CredentialInjection|Invoke-DllInjection|Invoke-
Mimikatz|Invoke-NinjaCopy|Invoke-Portscan|Invoke-
ReflectivePEInjection|Invoke-ReverseDnsLookup|Invoke-
Shellcode|Invoke-TokenManipulation|Invoke-
WmiCommand|Mount-VolumeShadowCopy|New-
ElevatedPersistenceOption|New-UserPersistenceOption|New-
VolumeShadowCopy|Out-CompressedDll|Out-
EncodedCommand|Out-EncryptedScript|Out-
Minidump|PowerUp|PowerView|Remove-Comments|Remove-
VolumeShadowCopy|Set-CriticalProcess|Set-
MasterBootRecord</field>
    <description>Risky CMDLet executed. Possible malicious
activity detected.</description>
    <mitre>
      <id>T1059.001</id>
    </mitre>
  </rule>

  <rule id="100204" level="8">
    <if_sid>91802</if_sid>
    <field name="win.eventdata.scriptBlockText"
type="pcre2">(?i)mshta.*GetObject|mshta.*new
ActiveXObject</field>
    <description>Mshta used to download a file. Possible
malicious activity detected.</description>
    <mitre>
      <id>T1059.001</id>
    </mitre>
  </rule>

  <rule id="100205" level="5">
    <if_sid>60009</if_sid>
    <field name="win.eventdata.contextInfo"
type="pcre2">(?i)ExecutionPolicy bypass|exec bypass</field>
    <description>PowerShell execution policy set to
bypass.</description>
    <mitre>
      <id>T1059.001</id>
    </mitre>
  </rule>
```

```
  <rule id="100206" level="5">
    <if_sid>60009</if_sid>
    <field name="win.eventdata.contextInfo"
type="pcre2">(?i)Invoke-WebRequest|IWR.*-url|IWR.*-
InFile</field>
    <description>Invoke Webrequest executed, possible
download cradle detected.</description>
    <mitre>
      <id>T1059.001</id>
    </mitre>
  </rule>
</group>

<group name="sysmon_linux,sysmon_event3">
  <!-- Windows -->
  <rule id="102503" level="10">
    <if_group>sysmon_event3</if_group>
    <list field="win.eventdata.destinationPort"
lookup="not_address_match_key">etc/lists/common-
ports</list>
    <description>Sysmon - Event 3: Network connection to
Uncommon Port by $(win.eventdata.image)</description>
    <group>sysmon_event3,</group>
  </rule>
  <!-- Linux -->
  <rule id="102504" level="10">
    <if_sid>200152</if_sid>
    <list field="eventdata.destinationPort"
lookup="not_address_match_key">etc/lists/common-
ports</list>
    <description>Sysmon - Event 3: Network connection to
Uncommon Port by $(eventdata.image)</description>
    <group>sysmon_linux,</group>
  </rule>
</group>

<group name="virustotal,">
  <rule id="100092" level="12">
    <if_sid>657</if_sid>
    <match>Successfully removed threat</match>
    <description>$(parameters.program) removed threat located
at $(parameters.alert.data.virustotal.source.file)</description>
  </rule>

  <rule id="100093" level="12">
    <if_sid>657</if_sid>
    <match>Error removing threat</match>
    <description>Error removing threat located at
$(parameters.alert.data.virustotal.source.file)</description>
  </rule>
</group>

<group name="virustotal,">
  <rule id="100094" level="12">
    <if_sid>657</if_sid>
    <description>$(parameters.alert.data.abuseipdb.source.srcip)
IP Has been blocked by abuseipdb-block-ip Active
Response</description>
  </rule>
</group>

<group name="sysmon_linux,sysmon_event3,">
  <!-- Linux -->
  <rule id="102505" level="3">
    <if_sid>200152</if_sid>
    <field name="eventdata.image"
type="pcre2">(?i)(edge|firefox|safari|chrome|opera|brave|chro
mium|tor|vivaldi|arc)</field>
    <description>Sysmon - Event 3: Web connection by browser
- $(eventdata.image)</description>
    <group>sysmon_linux,</group>
  </rule>
  <rule id="102506" level="3">
    <if_sid>200152</if_sid>
    <field name="eventdata.image"
type="pcre2">(?i)(terminal|openconsole|cmd|powershell|pwsh|
bash|zsh|sh|fish|dash|ksh|csh|tcsh|bin|sbin)</field>
    <description>Sysmon - Event 3: Network connection by
Shell - $(eventdata.image)</description>
    <group>sysmon_linux,</group>
  </rule>
  <!-- Windows -->
  <rule id="102507" level="3">
    <if_group>sysmon_event3</if_group>
    <field name="win.eventdata.image"
type="pcre2">(?i)(edge|firefox|safari|chrome|opera|brave|chro
mium|tor|vivaldi|arc)</field>
    <field name="win.eventdata.destinationIp" type="pcre2"
negate="yes">^(127\.|10\.|192\.168\.|172\.(1[6-9]|2[0-9]|3[0-
1])\.)</field>
    <description>Sysmon - Event 3: Web connection by browser
- $(win.eventdata.image)</description>
    <group>sysmon_event3,</group>
  </rule>
  <rule id="102508" level="3">
    <if_group>sysmon_event3</if_group>
    <field name="win.eventdata.image"
type="pcre2">(?i)(terminal|openconsole|cmd|powershell|pwsh|
bash|zsh|sh|fish|dash|ksh|csh|tcsh|bin|sbin)</field>
    <field name="win.eventdata.destinationIp" type="pcre2"
negate="yes">^(127\.|10\.|192\.168\.|172\.(1[6-9]|2[0-9]|3[0-
1])\.)</field>
    <description>Sysmon - Event 3: Network connection by
Shell - $(win.eventdata.image)</description>
    <group>sysmon_event3,</group>
  </rule>
</group>

<group name="local,syslog,sysmon_linux,sysmon_event3,">
<!-- NO RISK (0) -->
    <rule id="102517" level="3">
      <field name="abuseipdb.source.rule"
type="pcre2">^102505|102507$</field>
      <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^0$</field>
      <description>AbuseIPDB: Clean web connection from IP
$(abuseipdb.source.srcip) - Score 0%.</description>

<group>local,syslog,sysmon_linux,sysmon_event3,</group>
    </rule>
    <rule id="102518" level="3">
      <field name="abuseipdb.source.rule"
type="pcre2">^102506|102508$</field>
      <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^0$</field>
      <description>AbuseIPDB: Clean shell connection from IP
$(abuseipdb.source.srcip) - Score 0%.</description>

<group>local,syslog,sysmon_linux,sysmon_event3,</group>
    </rule>
  <!-- LOW RISK (1-30) -->
  <rule id="102511" level="4">
    <field name="abuseipdb.source.rule"
type="pcre2">^102505|102507$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^([1-9]|[1-2][0-9]|30)$</field>
```

```
    <description>AbuseIPDB: Low-risk web connection from IP
$(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%.</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
  <rule id="102512" level="4">
    <field name="abuseipdb.source.rule"
type="pcre2">^102506|102508$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^([1-9]|[1-2][0-9]|30)$</field>
    <description>AbuseIPDB: Low-risk shell connection from
IP $(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%.</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
  <!-- MEDIUM RISK (31-70) -->
  <rule id="102513" level="7">
    <field name="abuseipdb.source.rule"
type="pcre2">^102505|102507$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^(3[1-9]|[4-5][0-9]|60)$</field>
    <description>AbuseIPDB: Moderate-risk web connection
from IP $(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%.</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
  <rule id="102514" level="7">
    <field name="abuseipdb.source.rule"
type="pcre2">^102506|102508$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^(3[1-9]|[4-6][0-9]|70)$</field>
    <description>AbuseIPDB: Moderate-risk shell activity from
IP $(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%.</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
  <!-- HIGH RISK (71-100) -->
  <rule id="102515" level="12">
    <field name="abuseipdb.source.rule"
type="pcre2">^102505|102507$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^(7[1-9]|[8-9][0-9]|100)$</field>
    <description>AbuseIPDB: High-risk web connection from
IP $(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%!</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
  <rule id="102516" level="12">
    <field name="abuseipdb.source.rule"
type="pcre2">^102506|102508$</field>
    <field name="abuseipdb.abuse_confidence_score"
type="pcre2">^(7[1-9]|[8-9][0-9]|100)$</field>
    <description>AbuseIPDB: High-risk shell command attempt
from IP $(abuseipdb.source.srcip) - Score
$(abuseipdb.abuse_confidence_score)%!</description>
    <group>local,syslog,sysmon_linux,sysmon_event3,</group>
  </rule>
</group>

<group name="syscollector2,">
<!-- New network interface -->
  <rule id="100371" level="2">
    <if_sid>221</if_sid>
    <field name="type">dbsync_network_iface</field>
    <description>Syscollector network interface
event.</description>
  </rule>
  <rule id="100372" level="9">
    <if_sid>100371</if_sid>
    <field name="operation_type">INSERTED</field>
    <description>New network interface $(netinfo.iface.name)
detected.</description>
  </rule>
</group>

<!-- Modify it at your will. -->
<!-- Copyright (C) 2015, Wazuh Inc. -->

<!-- Example -->
<group name="local,syslog,sshd,">

  <!--
  Dec 10 01:02:02 host sshd[1234]: Failed none for root from
1.1.1.1 port 1066 ssh2
  -->
  <rule id="100001" level="5">
    <if_sid>5716</if_sid>
    <srcip>1.1.1.1</srcip>
    <description>sshd: authentication failed from IP
1.1.1.1.</description>

<group>authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,</
group>
  </rule>

</group>
```

agent.conf windows group

```
<agent_config>
  <!-- Shared agent configuration here -->
  <!-- FIM Module -->
  <syscheck>
    <disabled>no</disabled>
    <!-- Real Time -->
    <directories check_all="yes" realtime="yes"
report_changes="yes">C:\Users\*\Downloads</directori
es>
    <directories check_all="yes" realtime="yes"
report_changes="yes">C:\Users\*\Desktop</directories>
    <directories check_all="yes" realtime="yes"
report_changes="yes">C:\Users\*\Documents</directori
es>
    <!-- Scheduled Scans -->
    <frequency>3600</frequency>
    <!-- Every 1 hour -->
    <directories check_all="yes"
report_changes="yes">C:\ProgramData\Microsoft\Wind
ows\Start Menu\Programs\Startup</directories>
    <directories check_all="yes"
report_changes="yes">C:\Windows\System32</directori
es>
    <!-- Register Monitoring -->
    <windows_registry arch="both" check_all="yes"
report_changes="yes">HKEY_LOCAL_MACHINE\Sof
tware\Microsoft\Windows\CurrentVersion\Run</windo
ws_registry>
    <windows_registry arch="both" check_all="yes"
report_changes="yes">HKEY_CURRENT_USER\Soft
ware\Microsoft\Windows\CurrentVersion\Run</window
s_registry>
    <!-- Size for DB to contrcol changes in files (diff) -->
    <disk_quota>300</disk_quota>
  </syscheck>
  <!-- Windows Defender log collection -->
  <localfile>
    <location>Microsoft-Windows-Windows
Defender/Operational</location>
    <log_format>eventchannel</log_format>
  </localfile>
  <!-- PowerShell commands log collection -->
  <localfile>
    <location>Microsoft-Windows-
PowerShell/Operational</location>
    <log_format>eventchannel</log_format>
  </localfile>
  <!-- Sysmon logs collection -->
  <localfile>
    <location>Microsoft-Windows-
Sysmon/Operational</location>
    <log_format>eventchannel</log_format>
  </localfile>
  <!-- Rootckeck module -->
  <rootcheck>
    <disabled>no</disabled>
    <check_unixaudit>no</check_unixaudit>
    <check_files>no</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>no</check_dev>
    <check_sys>no</check_sys>
    <check_pids>no</check_pids>
    <check_ports>no</check_ports>
    <check_if>no</check_if>
    <frequency>21600</frequency>

<rootkit_files>etc/shared/rootkit_files_windows.txt</roo
tkit_files>

<rootkit_trojans>etc/shared/rootkit_trojans_windows.txt
</rootkit_trojans>
    <skip_nfs>yes</skip_nfs>
  </rootcheck>
</agent_config>
```

Додаток И

agent.conf linux group

```
<agent_config>
  <!-- FIM module -->
  <syscheck>
    <disabled>no</disabled>
    <!-- Realtime scan -->
    <directories check_all="yes" realtime="yes"
report_changes="yes">/home/*/Downloads</directories
>
    <directories check_all="yes" realtime="yes"
report_changes="yes">/home/*/Desktop</directories>
    <directories check_all="yes" realtime="yes"
report_changes="yes">/home/*/Documents</directories
>
    <directories realtime="yes">/root</directories>
    <!-- Scheduled scan every 1 hour -->
    <frequency>3600</frequency>
    <directories check_all="yes"
report_changes="yes">/etc/init.d</directories>
    <directories check_all="yes"
report_changes="yes">/usr/bin</directories>
    <!-- Size for DB to control changes in files (diff) -->
    <disk_quota>300</disk_quota>
  </syscheck>
  <!-- AuditD Logs -->
  <localfile>
    <log_format>audit</log_format>
    <location>/var/log/audit/audit.log</location>
  </localfile>
  <!-- Sysmon Logs -->
  <localfile>
    <log_format>syslog</log_format>
    <location>/var/log/syslog</location>
  </localfile>
  <!-- Rootcheck module -->
  <rootcheck>
    <disabled>no</disabled>
    <check_files>yes</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>yes</check_dev>
    <check_sys>yes</check_sys>
    <check_pids>yes</check_pids>
    <check_ports>yes</check_ports>
    <check_if>yes</check_if>
    <!-- 21600 seconds = every 6 hours -->
    <frequency>21600</frequency>

<rootkit_files>etc/shared/rootkit_files.txt</rootkit_files>

<rootkit_trojans>etc/shared/rootkit_trojans.txt</rootkit_t
rojans>
    <skip_nfs>yes</skip_nfs>
  </rootcheck>
</agent_config>
```

Додаток К

agent.conf macos group

```
<agent_config>
  <syscheck>
    <disabled>no</disabled>
    <directories check_all="yes"
report_changes="yes">/Users/*/Downloads</directories
>
    <directories check_all="yes"
report_changes="yes">/Users/*/Desktop</directories>
    <directories check_all="yes"
report_changes="yes">/Users/*/Documents</directories
>
    <frequency>60</frequency>
    <ignore>/bin</ignore>
    <ignore>/sbin</ignore>
    <ignore>/usr/bin</ignore>
    <ignore>/usr/sbin</ignore>
    <ignore>/usr/etc</ignore>
    <ignore>/etc</ignore>
    <ignore>/Users/*/Library</ignore>
  </syscheck>
  <!-- Collect logs from active-response module in
Wazuh Agent -->
  <localfile>
    <log_format>syslog</log_format>
    <location>/Library/Ossec/logs/active-
responses.log</location>
  </localfile>
  <!-- Check Internet Connetions -->
  <localfile>
    <log_format>command</log_format>
    <command>netstat -anv | awk '/^(tcp|udp)/ {print $1,
$4, $5, $6}'</command>
    <alias>netstat internet connection</alias>
    <frequency>120</frequency>
  </localfile>
  <!-- Rootckeck module -->
  <rootcheck>
    <disabled>no</disabled>
    <check_unixaudit>yes</check_unixaudit>
    <check_files>yes</check_files>
    <check_trojans>yes</check_trojans>
    <check_dev>yes</check_dev>
    <check_sys>yes</check_sys>
    <check_pids>yes</check_pids>
    <check_ports>yes</check_ports>
    <check_if>yes</check_if>
    <ignore type="sregex">^/etc/</ignore>
    <!-- Frequency that rootcheck is executed - every 6
hours -->
    <frequency>21600</frequency>

<rootkit_files>etc/shared/rootkit_files.txt</rootkit_files>

<rootkit_trojans>etc/shared/rootkit_trojans.txt</rootkit_t
rojans>
    <skip_nfs>yes</skip_nfs>
  </rootcheck>
</agent_config>
```

remove-threat.sh for Linux

```
#!/bin/bash

LOCAL=`dirname $0`;
cd $LOCAL
cd ../

PWD=`pwd`

read INPUT_JSON
FILENAME=$(echo $INPUT_JSON | jq -r .parameters.alert.data.virustotal.source.file)
COMMAND=$(echo $INPUT_JSON | jq -r .command)
LOG_FILE="${PWD}/../logs/active-responses.log"

#------------------------ Analyze command -------------------------#
if [ ${COMMAND} = "add" ]
then
 # Send control message to execd
 printf '{"version":1,"origin":{"name":"remove-threat","module":"active-response"},"command":"check_keys",
"parameters":{"keys":[]}}\n'

 read RESPONSE
 COMMAND2=$(echo $RESPONSE | jq -r .command)
 if [ ${COMMAND2} != "continue" ]
 then
  echo "`date '+%Y/%m/%d %H:%M:%S'` $0: $INPUT_JSON Remove threat active response aborted" >> ${LOG_FILE}
  exit 0;
 fi
fi

# Removing file
rm -f $FILENAME
if [ $? -eq 0 ]; then
 echo "`date '+%Y/%m/%d %H:%M:%S'` $0: $INPUT_JSON Successfully removed threat" >> ${LOG_FILE}
else
 echo "`date '+%Y/%m/%d %H:%M:%S'` $0: $INPUT_JSON Error removing threat" >> ${LOG_FILE}
fi

exit 0;
```

abuseipdb-block-ip.py for Linux

```
#!/usr/bin/python3

import sys
import json
import subprocess
import datetime

ADD_COMMAND = 0
DELETE_COMMAND = 1
CONTINUE_COMMAND = 2
ABORT_COMMAND = 3

OS_SUCCESS = 0
OS_INVALID = -1

LOG_FILE = "/var/ossec/logs/active-responses.log"

class message:
    def __init__(self):
        self.alert = ""
        self.command = 0

def write_debug_file(ar_name, msg):
    with open(LOG_FILE, mode="a") as log_file:

log_file.write(str(datetime.datetime.now().strftime('%Y/
%m/%d %H:%M:%S')) + " " + ar_name + ": " + msg
+"\n")

def setup_and_check_message(argv):

    # get alert from stdin
    input_str = ""
    for line in sys.stdin:
        input_str = line
        break

    try:
        data = json.loads(input_str)
    except ValueError:
        write_debug_file(argv[0], 'Decoding JSON has
failed, invalid input format')
        message.command = OS_INVALID
        return message
    message.alert = data

    command = data.get("command")

    if command == "add":
        message.command = ADD_COMMAND
    elif command == "delete":
        message.command = DELETE_COMMAND
    else:
        message.command = OS_INVALID
        write_debug_file(argv[0], 'Not valid command: ' +
command)

    return message

def main(argv):
    write_debug_file(argv[0], "Started")
    msg = setup_and_check_message(argv)

    if msg.command < 0:
        sys.exit(OS_INVALID)

    ip = msg.alert.get("parameters", {}).get("alert",
{}).get("data", {}).get("abuseipdb", {}).get("source",
{}).get("srcip")

    if not ip:
        write_debug_file(argv[0], f"Error accoured, the ip
param is None type!")
        sys.exit(OS_INVALID)

    if msg.command == ADD_COMMAND:
        subprocess.run(["iptables", "-I", "INPUT", "-s", ip,
"-j", "DROP"])
        with open("/etc/iptables/rules.v4", "w") as f:
            subprocess.run(["iptables-save"], stdout=f)
        write_debug_file(argv[0], f"{ip} has been
successfully blocked by iptables!")

    write_debug_file(argv[0], json.dumps(msg.alert))
    write_debug_file(argv[0], f"The end of the script
execution.")

if __name__ == "__main__":
    main(sys.argv)
```

abuseipdb-block-ip.py for MacOS

```
#!/usr/bin/env python3

import sys
import json
import subprocess
import datetime
import os

ADD_COMMAND = 0
DELETE_COMMAND = 1

OS_SUCCESS = 0
OS_INVALID = -1

LOG_FILE = "/Library/Ossec/logs/active-responses.log"
BLOCK_TABLE_NAME = "blocked_ips"
PF_CONF_FILE = "/etc/pf.conf"
BLOCKED_IPS_FILE = f"/etc/{BLOCK_TABLE_NAME}.txt"

class Message:
    def __init__(self):
        self.alert = {}
        self.command = OS_INVALID

def write_log(tag, msg):
    with open(LOG_FILE, "a") as log:
        timestamp = datetime.datetime.now().strftime('%Y/%m/%d
%H:%M:%S')
        log.write(f"{timestamp} {tag}: {msg}\n")

def setup_and_check_message(argv):
    input_str = ""
    for line in sys.stdin:
        input_str = line
        break

    msg = Message()

    try:
        data = json.loads(input_str)
    except Exception as e:
        write_log(argv[0], f"Invalid JSON input: {e}")
        return msg

    msg.alert = data

    command = data.get("command")
    if command == "add":
        msg.command = ADD_COMMAND
    elif command == "delete":
        msg.command = DELETE_COMMAND
    else:
        write_log(argv[0], f"Unsupported command: {command}")

    return msg

def reload_pf():
    subprocess.run(["pfctl", "-f", PF_CONF_FILE])
    subprocess.run(["pfctl", "-t", BLOCK_TABLE_NAME, "-T",
"replace", "-f", BLOCKED_IPS_FILE])

def ensure_pf_table():
    # Create blocked IPs file if missing
    if not os.path.exists(BLOCKED_IPS_FILE):
        with open(BLOCKED_IPS_FILE, "w") as f:
            pass

    with open(PF_CONF_FILE, "r") as f:
        pf_conf = f.read()

    table_def = f"table <{BLOCK_TABLE_NAME}> persist file
\"{BLOCKED_IPS_FILE}\""
    block_rule = f"block in quick from <{BLOCK_TABLE_NAME}>"

    if table_def not in pf_conf or block_rule not in pf_conf:
        with open(PF_CONF_FILE, "a") as f:
            f.write(f"\n{table_def}\n{block_rule}\n")
        reload_pf()

def block_ip(ip):
    with open(BLOCKED_IPS_FILE, "a") as f:
        f.write(f"{ip}\n")
    reload_pf()

def unblock_ip(ip):
    if not os.path.exists(BLOCKED_IPS_FILE):
        return
    with open(BLOCKED_IPS_FILE, "r") as f:
        lines = f.readlines()
    with open(BLOCKED_IPS_FILE, "w") as f:
        for line in lines:
            if line.strip() != ip:
                f.write(line)
    reload_pf()

def extract_ip(msg):
    return msg.alert.get("parameters", {}).get("alert", {}).get("data",
{}).get("abuseipdb", {}).get("source", {}).get("srcip")

def main(argv):
    write_log(argv[0], "Script started")
    ensure_pf_table()

    msg = setup_and_check_message(argv)
    if msg.command == OS_INVALID:
        sys.exit(OS_INVALID)

    ip = extract_ip(msg)
    if not ip:
        write_log(argv[0], "IP is None or invalid")
        sys.exit(OS_INVALID)

    if msg.command == ADD_COMMAND:
        block_ip(ip)
        write_log(argv[0], f"Blocked {ip} via PF")
    elif msg.command == DELETE_COMMAND:
        unblock_ip(ip)
        write_log(argv[0], f"Unblocked {ip} from PF")

    write_log(argv[0], "Script finished")

if __name__ == "__main__":
    main(sys.argv)
```

abuseipdb-block-ip.py for Windows

```
#!/usr/bin/env python3

import sys
import json
import subprocess
import datetime
import ctypes

ADD_COMMAND = 0
DELETE_COMMAND = 1
CONTINUE_COMMAND = 2
ABORT_COMMAND = 3

OS_SUCCESS = 0
OS_INVALID = -1

LOG_FILE = r"C:\Program Files (x86)\ossec-agent\active-response\active-responses.log"

class Message:
    def __init__(self):
        self.alert = {}
        self.command = 0

def write_debug_file(ar_name, msg):
    try:
        with open(LOG_FILE, mode="a") as log_file:

log_file.write(f"{datetime.datetime.now().strftime('%Y/%m/%d %H:%M:%S')} {ar_name}: {msg}\n")
    except Exception as e:
        print(f"Failed to write to log file: {e}")

def setup_and_check_message(argv):
    msg = Message()
    input_str = ""
    for line in sys.stdin:
        input_str = line
        break

    try:
        data = json.loads(input_str)
    except ValueError:
        write_debug_file(argv[0], 'Decoding JSON has failed, invalid input format')
        msg.command = OS_INVALID
        return msg

    msg.alert = data

    command = data.get("command")

    if command == "add":
        msg.command = ADD_COMMAND
    elif command == "delete":
        msg.command = DELETE_COMMAND
    else:
        msg.command = OS_INVALID
        write_debug_file(argv[0], f'Not valid command: {command}')

    return msg

def block_ip(ip, ar_name):
    rule_name = f"WazuhBlock-AbuseIPDB_{ip}"
    powershell_cmd_inbound = f"New-NetFirewallRule -DisplayName '{rule_name}' -Direction Inbound -Action Block -RemoteAddress {ip} -Enabled True"

    try:
        write_debug_file(ar_name, f"Running PowerShell command: {powershell_cmd_inbound}")
        result = subprocess.run(
            ["powershell.exe", "-Command", powershell_cmd_inbound],
            capture_output=True,
            text=True
        )
        if result.returncode == 0:
            write_debug_file(ar_name, f"{ip} has been successfully blocked by Windows Firewall!")
        else:
            write_debug_file(ar_name, f"Failed to block {ip}. STDERR: {result.stderr.strip()} | STDOUT: {result.stdout.strip()}")
    except Exception as e:
        write_debug_file(ar_name, f"Exception occurred while blocking {ip}: {e}")

    powershell_cmd_outbound = f"New-NetFirewallRule -DisplayName '{rule_name}' -Direction Outbound -Action Block -RemoteAddress {ip} -Enabled True"

    try:
        write_debug_file(ar_name, f"Running PowerShell command: {powershell_cmd_outbound}")

        result = subprocess.run(
            ["powershell.exe", "-Command", powershell_cmd_outbound],
            capture_output=True,
            text=True
        )
        if result.returncode == 0:
            write_debug_file(ar_name, f"{ip} has been successfully blocked by Windows Firewall!")
        else:
            write_debug_file(ar_name, f"Failed to block {ip}. STDERR: {result.stderr.strip()} | STDOUT: {result.stdout.strip()}")
    except Exception as e:
        write_debug_file(ar_name, f"Exception occurred while blocking {ip}: {e}")

def main(argv):
    if not ctypes.windll.shell32.IsUserAnAdmin():
        write_debug_file(argv[0], "Script is not running with administrative privileges.")

    write_debug_file(argv[0], "Started")
    msg = setup_and_check_message(argv)

    if msg.command < 0:
        sys.exit(OS_INVALID)

    ip = msg.alert.get("parameters", {}).get("alert", {}).get("data", {}).get("abuseipdb", {}).get("source", {}).get("srcip")

    if not ip:
        write_debug_file(argv[0], "Error occurred, the ip param is None type!")
        sys.exit(OS_INVALID)

    if msg.command == ADD_COMMAND:
        block_ip(ip, argv[0])

    write_debug_file(argv[0], json.dumps(msg.alert))
    write_debug_file(argv[0], "The end of the script execution.")

if __name__ == "__main__":
    main(sys.argv)
```