

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 магістра
на тему: «Методи та технології виявлення і реагування
на інциденти в гетерогенних системах»

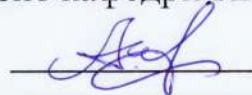
Виконав: студент 2-го курсу групи КБм-24-1

спеціальності 125 «Кібербезпека та захист інформації»



Павло СКОЧКО

Керівник: Доцент кафедри КІ та КБ, к.т.н.



Олексій ШЕЛУХА

Рецензент: Доцент кафедри КІ та КБ, к.т.н., доцент



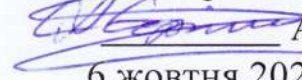
Юрій РОССІНСЬКИЙ

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

ЗАТВЕРДЖУЮ

Завідувач

кафедри комп'ютерної
інженерії та кібербезпеки

 Андрій Єфіменко
6 жовтня 2025 р.

ЗАВДАННЯ

на випускн у кваліфікаційну роботу магістра

Студент: Скочко Павло Андрійович

Тема роботи: «Методи та технології виявлення і реагування на інциденти в гетерогенних системах»

затверджена Наказом університету від 6 жовтня 2025 р. №566/с

Термін здачі студентом закінченої роботи 24 грудня 2025 р.

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

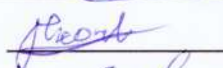
- Система виявлення і реагування на інциденти в гетерогенних системах;
- Процес функціонування мережної інфраструктури;

Обладнання та програмне забезпечення:

1. Wazuh
2. Ubuntu Server
3. Kali Linux

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Андрій Єфіменко	07.11.25 	09.11.25 
2	Андрій Єфіменко	10.11.25 	12.11.25 
3	Андрій Єфіменко	13.11.25 	15.11.25 

Студент 

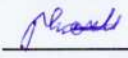
Павло СКОЧКО

Керівник 

Олексій ШЕЛУХА

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	до 10.10.2025	Виконано
2	Розробка плану роботи	до 16.10.2025	Виконано
3	Аналіз об'єкта дослідження	до 16.10.2025	Виконано
4	Опрацювання літературних джерел	до 03.11.2025	Виконано
5	Розробка теоретичного макету	до 06.11.2025	Виконано
6	Вибір обладнання	до 19.11.2025	Виконано
7	Налаштування системи	до 29.11.2025	Виконано
8	Тестування системи	до 29.11.2025	Виконано
9	Аналіз результатів тестування	до 29.11.2025	Виконано
10	Написання пояснювальної записки	до 07.12.2025	Виконано
11	Отримання відгуку та рецензій	до 13.12.2025	Виконано
12	Захист	до 28.12.2025	Виконано

Студент 

Павло СКОЧКО

Керівник 

Олексій ШЕЛУХА

АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи на тему «Методи та технології виявлення і реагування на інциденти в гетерогенних системах», складається з 80 сторінок, 46 рисунків, 1 таблиці, список використаних джерел містить 23 наіменувань та займає 2 сторінки.

Метою роботи є проєктування, розгортання та дослідження процесу роботи комплексної системи виявлення та реагування на інциденти інформаційної безпеки в гетерогенних системах.

У ході роботи визначено основні завдання для проєктування та дослідження комплексної системи виявлення та реагування на інциденти, проаналізовано сучасні підходи до моніторингу та обрано оптимальне рішення для захисту гетерогенних мереж. Розглянуто особливості та архітектуру провідних SIEM та XDR-систем, таких як Splunk, IBM QRadar, ELK Stack та Wazuh. Показано процес розгортання та виконано дослідження роботи обраної платформи Wazuh у поєднанні з pfSense та Snort, включаючи налаштування моніторингу контейнерів Docker, розробку правил кореляції та сценаріїв автоматичного реагування. Наведено результати тестування, які підтверджують ефективність розгорнутої системи у виявленні різнотипних загроз та надійність механізмів активного блокування атак у реальному часі.

Ключові слова: SIEM, XDR, WAZUH, SNORT, PFSENSE, ГЕТЕРОГЕННІ МЕРЕЖІ, МОНІТОРИНГ, РЕАГУВАННЯ НА ІНЦИДЕНТИ, DOCKER, ACTIVE RESPONSE.

ANNOTATION

Explanatory note to the qualification paper on the topic "Methods and technologies for incident detection and response in heterogeneous systems" consists of 80 pages, 46 figures, 1 table; the list of references contains 23 items and occupies 2 pages.

The purpose of the work is the design, deployment, and investigation of the operation process of a comprehensive system for information security incident detection and response in heterogeneous systems.

In the course of the work, the main tasks for the design and research of a comprehensive incident detection and response system were defined, modern approaches to monitoring were analyzed, and the optimal solution for the protection of heterogeneous networks was selected. The features and architecture of leading SIEM and XDR systems, such as Splunk, IBM QRadar, ELK Stack, and Wazuh, were examined. The deployment process is demonstrated, and an investigation of the operation of the selected Wazuh platform in combination with pfSense and Snort was performed, including the configuration of Docker container monitoring, the development of correlation rules, and automatic response scenarios. The test results are presented, confirming the effectiveness of the deployed system in detecting various types of threats and the reliability of active attack blocking mechanisms in real-time.

Keywords: SIEM, XDR, WAZUH, SNORT, PFSENSE, HETEROGENEOUS NETWORKS, MONITORING, INCIDENT RESPONSE, DOCKER, ACTIVE RESPONSE.

ЗМІСТ

ВСТУП	8
РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМ МОНІТОРИНГУ ТА УПРАВЛІННЯ ПОДІЯМИ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ.....	10
1.1. Поняття та основні функції SIEM-систем	10
1.2. Архітектура та компоненти SIEM-систем.....	13
1.3. Аналіз та класифікація сучасних SIEM-рішень.....	17
Висновки до першого розділу:	21
РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ	23
2.1. Обґрунтування вибору інструментальних засобів	23
2.2. Розробка архітектури та топології тестової мережі	27
2.3. Планування сценаріїв моделювання кібератак.....	30
Висновки до другого розділу:.....	35
РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ МОНІТОРИНГУ	36
3.1. Розгортання та налаштування SIEM-системи.....	36
3.2. Тестування системи шляхом моделювання кібератак	55
3.3. Розробка власних правил кореляції та механізмів реагування	68
Висновки до третього розділу:	77
ВИСНОВКИ.....	78
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	79

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

DSS – Data Security Standard

DVWA – Damn Vulnerable Web Application

ELK – Elasticsearch, Logstash, Kibana

FIM – File Integrity Monitoring

HTTP – HyperText Transfer Protocol

IBM – International Business Machines

IP – Internet Protocol

IT – Information Technology

JSON – JavaScript Object Notsation

LAN – Local Area Network

MITRE – The MITRE Corporation (Massachusetts Institute of Technology Research and Engineering)

NIDS – Network Intrusion Detection System

OWASP – Open Web Application Security Project

PCI – Payment Card Industry

SIEM – Security Information and Event Management

SPL – Search Processing Language

SQL – Structured Query Language

SSH – Secure Shell

TCP – Transmission Control Protocol

UDP – User Datagram Protocol

WAN – Wide Area Network

XDR – Extended Detection and Response

ВСТУП

Актуальність теми. Ускладнення архітектури через перехід до гетерогенних середовищ вимагає впровадження централізованих систем безпеки, здатних агрегувати телеметрію, корелювати події та автоматично блокувати загрози в реальному часі. У межах цієї роботи розглядаються методи та засоби розгортання та дослідження процесу роботи систем виявлення і реагування на інциденти в гетерогенних системах.

Мета роботи. Метою кваліфікаційної роботи є проектування, розгортання та дослідження процесу роботи комплексної системи виявлення та реагування на інциденти інформаційної безпеки в гетерогенних системах.

Визначена мета обумовлює наступні завдання:

- Аналіз сучасних підходів до моніторингу подій безпеки та порівняльна характеристика існуючих SIEM та XDR-рішень.
- Обґрунтування вибору оптимального інструментарію для побудови системи захисту гетерогенної мережевої інфраструктури.
- Розробка проєкту розгортання компонентів системи, що включає інтеграцію мережевих шлюзів, хостових агентів та середовищ контейнеризації.
- Налаштування механізмів кореляції подій та тестування ефективності сценаріїв автоматизованого реагування на кібератаки.

Об’єктом дослідження є процеси виявлення, аналізу та нейтралізації кіберзагроз у розподілених інформаційно-телекомунікаційних системах зі змішаною архітектурою.

Предметом дослідження є методи, моделі та програмно-апаратні засоби централізованого моніторингу подій безпеки, алгоритми кореляції різнорідних даних та технології автоматизованого реагування на інциденти в умовах гетерогенного мережевого середовища.

Наукова новизна одержаних результатів. Набув подальшого розвитку метод комплексного захисту гетерогенних інформаційних систем шляхом поєднання агентського та безагентського збору телеметрії в межах єдиної платформи класу XDR. Удосконалено підхід до моніторингу ефемерних

контейнеризованих середовищ через реалізацію кастомного механізму перехоплення та нормалізації потокових логів, що дозволяє включати події ізольованих додатків у загальні ланцюжки кореляції інцидентів безпеки.

Практичне значення одержаних результатів визначається можливістю безпосереднього застосування розробленої архітектури для побудови бюджетних центрів моніторингу та реагування на інциденти в організаціях малого та середнього бізнесу. Запропоноване рішення на базі відкритого програмного забезпечення дозволяє уніфікувати процеси захисту в змішаних мережах, де одночасно використовуються платформи Windows та Linux, забезпечуючи автоматизоване реагування на інциденти та відповідність вимогам регуляторних стандартів без залучення дорогих комерційних платформ.

У процесі виконання кваліфікаційної роботи було розроблено проєкт комплексної системи моніторингу та реагування на інциденти безпеки на базі платформи Wazuh. Проєкт включає аналіз існуючої мережевої інфраструктури, вибір відповідного обладнання, налаштування системи та її тестування.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМ МОНІТОРИНГУ ТА УПРАВЛІННЯ ПОДІЯМИ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

1.1. Поняття та основні функції SIEM-систем

Сучасна парадигма управління інформаційною безпекою базується на комплексному підході до обробки даних, технологічним втіленням якого стали системи класу SIEM. Ця концепція виникла як еволюційний результат конвергенції двох історично розрізнених напрямків: SEM, що фокусувалося на моніторингу та оповіщенні в реальному часі, та SIM, орієнтованого на довготривале зберігання, аналіз журналів та формування аналітичної звітності [1]. Об'єднання цих дисциплін в єдину платформу дозволило подолати обмеження фрагментарних засобів захисту, надаючи фахівцям інструмент для виявлення загроз, які неможливо ідентифікувати шляхом аналізу ізольованих джерел даних. Стрімке ускладнення архітектури корпоративних мереж, де одночасно функціонують різноманітні операційні системи, мережеве обладнання, віртуалізовані середовища та хмарні сервіси, призвело до експоненційного зростання обсягів генерованої телеметрії. В умовах гетерогенного середовища, коли кількість подій може досягати мільйонів записів на добу, традиційні методи ручного аудиту втрачають свою ефективність і не можуть гарантувати своєчасного виявлення інцидентів. Технологія SIEM виступає як інтелектуальна надбудова над інфраструктурою, забезпечуючи централізований збір та автоматизовану обробку різноманітної інформації. Її впровадження дозволяє трансформувати хаотичні потоки технічних логів у структуровані знання про стан захищеності активів організації. Фундаментальна сутність сучасних SIEM-рішень полягає у забезпеченні контекстної обізнаності, що виходить за межі простої агрегації даних. Система створює логічні зв'язки між діями користувачів, мережевим трафіком та змінами у файлових системах, формуючи цілісну картину подій у хронологічному порядку. Такий підхід уможливорює виявлення складних, багатовекторних атак, які можуть бути розтягнуті у часі або розподілені між різними вузлами мережі. Розвиток

технології продовжується у напрямку інтеграції з методами поведінкового аналізу, що дозволяє переходити від реактивної моделі захисту до проактивного виявлення аномалій, які свідчать про підготовку або початок кібератаки ще до нанесення шкоди інформаційним ресурсам.

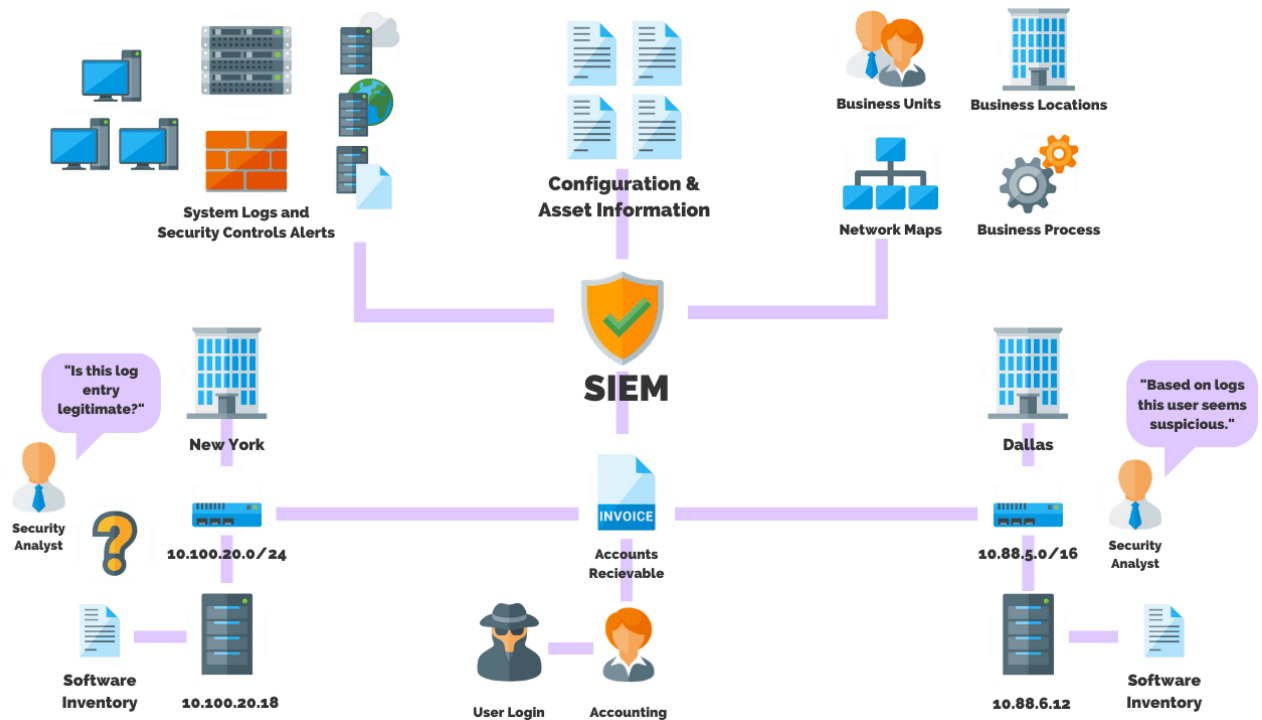


Рис. 1.1 Схематичне зображення роботи SIEM

Функціонування систем управління інцидентами базується на безперервному циклічному процесі перетворення даних, який розпочинається з етапу агрегації. Ця фаза передбачає консолідацію інформаційних потоків від широкого спектра джерел інфраструктури, включаючи мережеве обладнання, сервери, робочі станції та прикладне програмне забезпечення. Збір телеметрії може здійснюватися як за допомогою спеціалізованих агентів, встановлених безпосередньо на кінцевих точках, так і через стандартні мережеві протоколи передачі журналів, наприклад Syslog. Критичним завданням на цьому етапі є забезпечення цілісності та гарантованої доставки даних до центрального сховища, оскільки втрата навіть незначної частини пакетів може призвести до порушення логічних ланцюжків під час подальшого розслідування. Отримані "сирі" дані зазвичай представлені у різномірних форматах, що унеможлиблює їх безпосередній перехресний аналіз. Процес нормалізації вирішує цю проблему

шляхом парсингу вхідних повідомлень та приведення їх до єдиної уніфікованої таксономії. Під час цієї процедури система виокремлює з неструктурованого тексту ключові атрибути, такі як IP-адреси, імена користувачів, ідентифікатори процесів та мітки часу, записуючи їх у стандартні поля бази даних. Особлива увага приділяється синхронізації часових поясів, що дозволяє вибудувати правильну хронологію подій, які відбуваються на географічно розподілених вузлах. Результатом нормалізації є створення впорядкованого масиву даних, придатного для машинної обробки незалежно від походження початкового логу. Найбільш інтелектуально містким етапом циклу є кореляція подій, де застосовуються аналітичні алгоритми для виявлення взаємозв'язків між окремими фактами. Кореляційний рушій зіставляє нормалізовані записи з заздалегідь визначеними правилами, сигнатурами атак та поведінковими моделями. Цей механізм дозволяє системі фільтрувати інформаційний шум, відсіюючи легітимні дії користувачів, та ідентифікувати складні патерни загроз, які стають помітними лише при аналізі сукупності подій. Виявлення таких залежностей трансформує набір розрізнених технічних повідомлень у кваліфікований інцидент інформаційної безпеки, що містить контекстну інформацію, необхідну для прийняття рішення про реагування [2].

Впровадження систем централізованого моніторингу трансформує підхід до оцінки стану захищеності інформаційної інфраструктури, надаючи операторам центрів безпеки засоби для досягнення повної ситуаційної обізнаності. Візуалізація агрегованих даних через інтерактивні панелі приладів дозволяє транслювати складні технічні метрики у зрозумілі індикатори ризику, забезпечуючи миттєву оцінку поточного рівня загроз та статусу критичних активів. Графічне представлення тенденцій мережевої активності та статистики атак дає змогу виявляти приховані аномалії, які можуть не викликати спрацювання автоматичних тривог, але свідчать про підготовку складних цілеспрямованих кампаній. Це значно скорочує час прийняття управлінських рішень, дозволяючи змістити фокус уваги фахівців з рутинної перевірки журналів на аналіз стратегічних векторів атак. Ретроспективний аналіз

інцидентів та проведення цифрових розслідувань спираються на здатність платформи забезпечувати надійне довгострокове зберігання історичних даних у незмінному вигляді. Архітектура SIEM створює необхідні умови для детальної реконструкції хронології подій, дозволяючи відновити повний ланцюжок дій зломисника від моменту первинного проникнення до ексфільтрації даних. Можливість швидкого пошуку та гнучкої фільтрації великих масивів архівної інформації є критичною при проведенні комп'ютерної криміналістики, коли необхідно точно визначити масштаб компрометації та ідентифікувати всі уражені вузли. Гарантування цілісності збережених логів за допомогою криптографічних методів забезпечує юридичну значущість зібраних доказів, що є необхідною передумовою для правового переслідування порушників або проведення внутрішніх службових розслідувань. Регуляторний тиск та необхідність дотримання галузевих та міжнародних стандартів безпеки, таких як ISO/IEC 27001, PCI DSS або вимог GDPR, перетворюють автоматизацію контролю відповідності на обов'язковий елемент корпоративної стратегії захисту [3,4]. Системи моніторингу виступають ключовим інструментом документального підтвердження ефективності впроваджених контрольних процедур, дозволяючи автоматично генерувати необхідну звітність для проходження зовнішніх та внутрішніх аудитів. Централізація журналів аудиту спрощує процес демонстрації відповідності вимогам щодо розмежування доступу, захисту персональних даних та моніторингу змін у конфігураціях. Використання попередньо налаштованих шаблонів звітності дозволяє організації мінімізувати ресурсовитрати на підготовку до перевірок та уникнути фінансових або репутаційних втрат через порушення нормативів інформаційної безпеки.

1.2. Архітектура та компоненти SIEM-систем

Фундаментальним рівнем архітектури будь-якої системи управління подіями інформаційної безпеки є підсистема збору телеметрії, яка відповідає за акумуляцію даних від усіх компонентів інформаційної інфраструктури. Ефективність подальшого аналізу безпосередньо залежить від повноти та якості

отриманої інформації, що вимагає застосування гібридних підходів до моніторингу, особливо в умовах гетерогенних мереж. Побудова цього рівня базується на поєднанні двох основних методологій отримання даних: агентського моніторингу, що передбачає встановлення спеціалізованого програмного забезпечення на кінцеві точки, та безагентського збору, який використовує стандартні мережеві протоколи для комунікації з пристроями, де інсталяція сторонніх модулів неможлива. Реалізація агентського методу дозволяє отримати найбільш деталізовану картину стану операційної системи, надаючи доступ до внутрішніх механізмів ядра, файлової системи та процесів у пам'яті. Локальні агенти функціонують як інтелектуальні сенсори, здатні не лише транслювати вміст журналів подій, але й виконувати попередню фільтрацію та стиснення даних, знижуючи навантаження на пропускну здатність мережі. У розрізі захисту різнорідних платформ, таких як Windows та Linux, використання кросплатформних агентів вирішує проблему несумісності форматів на ранньому етапі, уніфікуючи структуру повідомлень перед їх відправкою на центральний сервер. Додатковою перевагою такого підходу є можливість локального буферування подій, що гарантує збереження критичних даних навіть у випадку тимчасової втрати зв'язку з керуючим вузлом. Архітектура моніторингу закритих систем, зокрема мережевих шлюзів, маршрутизаторів та промислових контролерів, спирається на використання стандартних протоколів передачі журналів, серед яких домінуючу роль відіграє Syslog [5]. Цей метод дозволяє інтегрувати в єдиний контур безпеки обладнання, яке не підтримує встановлення додаткових агентів, шляхом перенаправлення потоків подій на спеціалізовані колектори. Організація такого каналу зв'язку вимагає підвищеної уваги до захисту транспортного рівня, оскільки передача чутливої інформації про мережеві з'єднання та конфігурації у відкритому вигляді створює ризики перехоплення. Забезпечення цілісності та конфіденційності телеметрії досягається впровадженням механізмів взаємної автентифікації джерел та шифрування трафіку, що унеможливорює ін'єкцію фальшивих подій злоумисником.

Інтелектуальним ядром системи моніторингу виступає сервер обробки подій, який виконує функції глибокого аналізу вхідних потоків даних у режимі реального часу. Процес обробки розпочинається з етапу декодування, де спеціалізовані програмні модулі розбирають "сирі" рядки логів, що надходять від різномірних джерел. На цьому рівні відбувається синтаксичний аналіз тексту за допомогою регулярних виразів, що дозволяє виокремити з неструктурованого повідомлення ключові інформаційні поля, такі як мітка часу, ідентифікатор користувача, джерельна IP-адреса та тип події. Ефективність цього етапу визначає здатність системи "розуміти" мову конкретних додатків та операційних систем, трансформуючи набір байтів у структурований об'єкт даних, придатний для подальшої машинної обробки. Нормалізація даних виступає наступною ланкою в ланцюгу обробки, забезпечуючи приведення різноформатних подій до єдиної уніфікованої таксономії. Оскільки різні операційні системи та пристрої використовують власну термінологію для опису ідентичних дій, система повинна мапити ці значення на стандартні поля внутрішньої бази даних. Процедура уніфікації дозволяє абстрагуватися від специфіки конкретного джерела, створюючи єдиний логічний шар, де подія "Failed Password" з Linux-сервера та "Event ID 4625" з Windows-станції інтерпретуються як ідентичний інцидент помилки автентифікації. Такий підхід створює фундамент для написання універсальних правил безпеки, які будуть ефективно працювати у гетерогенному середовищі незалежно від архітектури підключених хостів. Кульмінаційним етапом аналізу є робота кореляційного рушія, який зіставляє нормалізовані події із закладеною логікою виявлення загроз. Цей компонент оперує складними алгоритмами, що враховують не лише сигнатурні ознаки окремих подій, але й їх частотні характеристики, послідовність появи та часові інтервали між ними. Механізм кореляції дозволяє виявляти складні багатовекторні атаки, пов'язуючи, наприклад, сканування портів мережевим екраном та подальшу підозрілу активність на сервері в єдиний логічний ланцюжок інциденту. Результатом роботи рушія є генерація алерту з

присвоєнням йому певного рівня критичності, що ініціює процедури оповіщення персоналу або автоматичного реагування.

Ефективна робота з великими масивами подій безпеки вимагає застосування спеціалізованих архітектурних рішень для зберігання даних, здатних забезпечити високу швидкість запису та пошуку інформації. Традиційні реляційні бази даних часто виявляються неефективними при обробці неструктурованих текстових логів у масштабах Big Data, тому сучасні SIEM-системи базуються на технологіях повнотекстової індексації з використанням інвертованих індексів. Такий підхід дозволяє здійснювати пошукові запити серед мільйонів записів за частки секунди, що є критичним фактором при проведенні оперативних розслідувань інцидентів. Організація сховища передбачає реалізацію політик ротації та архівування даних, що дозволяє балансувати між потребою у швидкому доступі до оперативної інформації та необхідністю довготривалого зберігання історичних журналів для відповідності вимогам нормативних документів. Трансформація технічної телеметрії у зрозумілу для аналітика форму здійснюється через підсистему візуалізації, яка виступає основним інтерфейсом взаємодії користувача з платформою. Веб-консолі управління надають інструментарій для побудови інтерактивних інформаційних панелей, де агрегована статистика відображається у вигляді графіків, діаграм розподілу та теплових карт. Графічне представлення даних дозволяє миттєво оцінювати загальний стан безпеки інфраструктури та виявляти аномальні відхилення від базових показників нормальної активності без необхідності ручного аналізу кожного окремого запису. Функціонал деталізації забезпечує можливість переходу від узагальнених віджетів до конкретних подій, дозволяючи дослідити першопричину виявленої аномалії на рівні "сирого" логу. Завершальним елементом ланцюжка обробки інформації є модуль генерації звітності, який автоматизує процес документування результатів моніторингу для потреб аудиту та менеджменту. Система дозволяє налаштовувати періодичне створення звітів у стандартизованих форматах, що містять зведену інформацію про виявлені загрози, статус відповідності стандартам та ефективність роботи

засобів захисту. Автоматизація звітності не лише економить час фахівців безпеки, але й забезпечує об'єктивність наданих даних, виключаючи людський фактор при формуванні доказової бази інцидентів. Наявність документованої історії подій є необхідною умовою для проведення ретроспективного аналізу та вдосконалення стратегії захисту організації.

1.3. Аналіз та класифікація сучасних SIEM-рішень

Ринок систем моніторингу подій безпеки характеризується чіткою сегментацією за типом ліцензування програмного забезпечення, що визначає стратегію впровадження та структуру витрат на кіберзахист. Комерційні платформи корпоративного класу, до яких відносяться рішення від провідних вендорів, пропонують замовникам готовий набір правил кореляції, гарантовану технічну підтримку та широкі можливості інтеграції «з коробки». Використання таких систем зазвичай пов'язане зі значними фінансовими інвестиціями, модель яких часто базується на обсязі оброблюваних даних або обсязі збережених логів. На противагу їм, рішення з відкритим вихідним кодом надають повний доступ до вихідного коду та свободу від ліцензійних відрахувань, дозволяючи адаптувати архітектуру під специфічні потреби організації. Впровадження відкритих платформ зміщує акцент витрат з купівлі ліцензій на утримання кваліфікованого інженерного персоналу, здатного самостійно підтримувати та налаштовувати складні конфігурації. Диференціація за моделлю розгортання є критичним архітектурним критерієм, розділяючи системи на локальні та хмарні. Класичний локальний підхід передбачає інсталяцію всіх компонентів SIEM на власних апаратних потужностях організації, що гарантує повний контроль над даними та відповідність жорстким регуляторним вимогам щодо заборони передачі інформації за межі контрольованого периметра. Ця модель забезпечує максимальну конфіденційність, проте вимагає капітальних витрат на серверну інфраструктуру та її постійне технічне обслуговування. Хмарні моделі, що стрімко набирають популярності, пропонують SIEM як сервіс, забезпечуючи швидке розгортання та еластичне масштабування ресурсів без необхідності

закупівлі "заліза". Вибір на користь хмарної архітектури перекладає відповідальність за доступність сервісу на провайдера, але вимагає надійних каналів зв'язку та детальної оцінки ризиків передачі чутливих журналів аудиту через публічні мережі. Сучасні тенденції демонструють зростання популярності гібридних архітектур та відкритих платформ, які дозволяють організаціям балансувати між вартістю та ефективністю. Відкриті рішення, завдяки активній спільноті розробників, часто випереджають пропрієтарні продукти у швидкості адаптації до нових векторів атак, пропонуючи функціонал рівня Enterprise без штучних обмежень продуктивності. Для гетерогенних мереж, де необхідно уніфікувати моніторинг різнорідних систем при обмеженому бюджеті, саме Open Source платформи стають оптимальним вибором, дозволяючи побудувати повноцінний центр реагування на інциденти на базі доступного програмного забезпечення.

Аналіз ринку систем моніторингу дозволяє виділити декілька ключових гравців, які формують технологічні стандарти галузі, серед яких безумовним лідером у сегменті комерційних рішень є платформа Splunk. Цей продукт характеризується потужним аналітичним апаратом, що дозволяє обробляти величезні масиви неструктурованих машинних даних з будь-яких джерел, перетворюючи їх на оперативну інформацію. Конкурентною перевагою системи є власна мова пошукових запитів SPL, яка надає аналітикам гнучкий інструментарій для проведення глибоких розслідувань та кореляції подій "на льоту" [6]. Попри широкі функціональні можливості та розвинену екосистему додатків, масове впровадження Splunk часто обмежується агресивною ціновою політикою, прив'язаною до обсягу індексації даних, що робить його економічно недоцільним для проєктів з обмеженим бюджетом або наукових досліджень. Альтернативою пропрієтарним гігантам виступає стек технологій ELK, який де-факто став стандартом у світі відкритих систем логування. Архітектура стеку забезпечує виняткову продуктивність при індексації та пошуку текстової інформації, дозволяючи будувати масштабовані кластери для зберігання терабайтів логів. Специфіка використання "чистого" ELK у контексті

інформаційної безпеки полягає в тому, що це рішення є універсальним конструктором, а не готовою SIEM-системою. Воно потребує значних трудовитрат на розробку власної логіки нормалізації подій, написання правил виявлення загроз та налаштування візуалізації, оскільки базовий дистрибутив не містить попередньо налаштованих модулів безпеки. Платформа Wazuh займає унікальну нішу, поєднуючи продуктивність пошукового рушія Elastic Stack із спеціалізованим функціоналом для захисту кінцевих точок та хмарних середовищ. На відміну від класичного ELK, це рішення постачається з інтегрованим набором декодерів та тисячами правил кореляції, адаптованих до фреймворку MITRE ATT&CK та вимог регуляторних стандартів, таких як PCI DSS. Архітектурною особливістю Wazuh є наявність повнофункціональних агентів для різних операційних систем, які виконують задачі інвентаризації, пошуку вразливостей та контролю цілісності файлів безпосередньо на хостах. Така інтеграція перетворює платформу на комплексне рішення класу XDR, здатне не лише збирати логи, але й активно реагувати на інциденти, що вигідно відрізняє її від систем, сфокусованих виключно на аналізі журналів.

Таблиця 1

Порівняння SIEM-систем [7-13]

SIEM-система	Тип рішення	Основні характеристики	Переваги	Недоліки
Splunk	Комерційне	Потужні можливості аналізу даних, гнучке корелювання подій, інтеграція з іншими системами	Потужний інтерфейс, масштабованість, підтримка великої кількості джерел даних	Висока вартість ліцензії, вимоги до ресурсів
IBM QRadar	Комерційне	Автоматизація процесів аналізу загроз, інтеграція з продуктами IBM, глибокий аналіз трафіку	Висока точність виявлення загроз, масштабованість	Складність налаштування, висока вартість

Таблиця 1 (Продовження)

SIEM-система	Тип рішення	Основні характеристики	Переваги	Недоліки
ArcSight	Комерційне	Гнучкі можливості кореляції подій, потужний аналітичний потенціал	Висока продуктивність, можливість налаштування під специфічні потреби	Складність у впровадженні, тривала крива навчання
Microsoft Sentinel	Комерційне	Хмарне рішення, інтеграція з Azure, автоматизація виявлення загроз	Легке масштабування, зручна інтеграція з іншими сервісами Microsoft	Може бути обмеженим для організацій без Azure
Graylog	Open-source	Веб-інтерфейс, потужна система обробки логів, можливість налаштування через плагіни	Гнучкість, безкоштовне використання, масштабованість	Потребує ресурсу для налаштування та підтримки
Wazuh	Open-source	Безпека кінцевих точок, інтеграція з OSSEC, можливості SIEM	Гнучкість налаштування, безкоштовне рішення	Більше орієнтоване на моніторинг безпеки кінцевих точок
ELK Stack	Open-source	Висока швидкість обробки даних, потужні можливості візуалізації через Kibana	Безкоштовне, потужний інструмент для обробки даних і візуалізації	Потребує великих зусиль для налаштування та підтримки

Еволюція засобів інформаційної безпеки останніми роками характеризується зміною фундаментальної парадигми від пасивного збору даних до активної протидії загрозам. Традиційні SIEM-системи, фокусуючись переважно на агрегації журналів та генерації звітів, зіткнулися з проблемою розриву між часом виявлення інциденту та моментом фактичної реакції на нього.

Зростання швидкості проведення кібератак, коли час від проникнення до ексфільтрації даних вимірюється хвилинами, зробило модель ручного реагування неефективною. Відповіддю на цей виклик стала поява концепції XDR, яка передбачає глибшу інтеграцію інструментів моніторингу з механізмами управління інфраструктурою, об'єднуючи захист кінцевих точок, мережевий аналіз та контроль хмарних середовищ у єдину екосистему реагування [14]. Ключова відмінність підходу XDR полягає у зміщенні акцентів з простої реєстрації подій на автоматизовану оркестрацію захисних дій. Сучасні платформи безпеки перестають бути ізольованими архівами логів, отримуючи можливості безпосереднього впливу на операційні системи та мережеве обладнання. Використання активних агентів на хостах дозволяє системі не лише фіксувати наявність шкідливого процесу, але й примусово завершувати його роботу, ізолювати заражену станцію від мережі або блокувати облікові записи скомпрометованих користувачів. Така архітектура усуває "сліпі зони" між різними засобами захисту, створюючи цілісний контур оборони, де подія на рівні мережевого шлюзу може миттєво ініціювати карантинні заходи на рівні робочої станції. Впровадження механізмів автоматичного реагування стає стандартом для систем нового покоління, дозволяючи мінімізувати час перебування зломисника в мережі. Алгоритми безпеки тепер включають сценарії SOAR, які виконуються без участі людини при спрацюванні критичних правил кореляції. Ця технологічна трансформація дозволяє організаціям протистояти автоматизованим атакам у режимі реального часу, перетворюючи систему моніторингу з інструменту посмертного аналізу інцидентів на активний щит, здатний самостійно нейтралізувати загрози на початкових етапах ланцюжка атаки Cyber Kill Chain [15].

Висновки до першого розділу:

Аналіз теоретичних засад побудови систем моніторингу засвідчив, що еволюція засобів інформаційної безпеки призвела до трансформації класичних SIEM-рішень у комплексні платформи класу XDR, здатні забезпечувати не лише

пасивну реєстрацію подій, а й активну протидію загрозам у реальному часі. Архітектурна конвергенція методів збору, нормалізації та кореляції даних дозволяє подолати проблему фрагментарності захисту в гетерогенних мережах, створюючи єдиний інформаційний простір для виявлення складних багатовекторних атак. Порівняння ринкових пропозицій виявило, що відкриті платформи з гнучкою агентською архітектурою, такі як Wazuh, надають необхідний функціонал для побудови адаптивних систем безпеки без надмірних фінансових витрат, стаючи повноцінною альтернативою комерційним продуктам та обґрунтованим вибором для подальшої практичної реалізації механізмів реагування на інциденти.

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ МОНІТОРИНГУ

2.1. Обґрунтування вибору інструментальних засобів

Центральним елементом системи захисту гетерогенних мереж є підсистема збору, аналізу та кореляції подій безпеки. Специфіка досліджуваної інфраструктури, яка об'єднує робочі станції на базі Windows, сервери під управлінням Linux та контейнеризовані додатки, висуває жорсткі вимоги до сумісності агентів та форматів логування. Критерії вибору платформи базувалися на необхідності забезпечення повного циклу обробки інцидентів: від детектування аномалій на рівні окремих хостів до централізованого реагування на мережеві загрози. Пріоритет надавався рішенням з відкритим вихідним кодом, що дозволяє детально аналізувати алгоритми кореляції подій та адаптувати їх під специфічні вектори атак без ліцензійних обмежень. Аналіз ринку комерційних рішень, зокрема платформи Splunk, демонструє широкі можливості щодо візуалізації даних та підтримки пропрієтарних протоколів. Використання подібних систем у дослідницьких цілях часто обмежується закритістю програмного коду та високою вартістю ліцензування, що ускладнює масштабування тестового стенда. Застосування "чистого" стеку ELK (Elasticsearch, Logstash, Kibana) хоч і забезпечує гнучкість у зберіганні логів, проте вимагає значних трудовитрат на розробку власної логіки безпеки, написання парсерів та правил виявлення загроз «з нуля», що зміщує фокус роботи з аналізу інцидентів на підтримку інфраструктури. Оптимальним варіантом для реалізації поставлених завдань обрано платформу Wazuh, яка поєднує функціонал класичної SIEM-системи та розширені можливості XDR. Архітектура рішення базується на використанні легковагових кросплатформних агентів, здатних збирати інформацію з різних джерел, включаючи системні журнали, події Docker та дані про цілісність файлової системи. Інтеграція Wazuh з базою знань MITRE ATT&CK та наявність вбудованих механізмів активного реагування дозволяє не лише фіксувати інциденти, але й автоматизувати протидію атакам, наприклад, блокувати мережеві з'єднання або зупиняти

шкідливі процеси в реальному часі [16]. Відкрита архітектура дозволяє інтегрувати платформу з зовнішніми інструментами, такими як NIDS Snort, забезпечуючи комплексний погляд на стан безпеки гетерогенної мережі.

Моделювання складних гетерогенних мереж вимагає використання середовища віртуалізації, яке забезпечує ізоляцію експериментальних вузлів від хостової системи при збереженні високої продуктивності. Гіпервізор Oracle VirtualBox було обрано як базову платформу для розгортання віртуальних машин завдяки його широкій сумісності з різними типами гостьових операційних систем, включаючи дистрибутиви Linux, Windows Server та BSD-системи [17]. Використання даного програмного забезпечення дозволяє емулювати апаратне забезпечення для кожного вузла тестового стенда, гнучко розподіляючи обчислювальні ресурси та дисковий простір відповідно до вимог розгорнутих сервісів, що є критичним для відтворення реалістичних сценаріїв навантаження під час кібератак. Для оркестрації мережевої взаємодії та візуального проєктування топології використано графічний симулятор мережі GNS3. Цей інструмент виконує роль сполучної ланки, інтегруючись безпосередньо з гіпервізором VirtualBox та дозволяючи об'єднувати віртуальні машини в єдину інфраструктуру через емульовані комутаційні пристрої [18]. Застосування GNS3 надає можливість детального моделювання фізичного та каналного рівнів мережі, дозволяючи налаштовувати логічне розділення мережі на сегменти та керувати маршрутизацією трафіку між атакуючими та захищеними вузлами без необхідності використання фізичного мережевого обладнання. Інтеграція GNS3 та VirtualBox створює цілісну екосистему для розгортання досліджуваної архітектури, де центральним елементом управління трафіком виступає програмний маршрутизатор pfSense. Така конфігурація забезпечує прозорість мережеских потоків, дозволяючи перехоплювати та аналізувати пакети даних на будь-якій ділянці віртуальної мережі. Реалізація тестового стенда на базі цих інструментів гарантує відтворюваність експериментів та дозволяє безпечно тестувати механізми виявлення інцидентів та активного реагування, не наражаючи на небезпеку основну робочу станцію дослідника.

Відтворення реалістичної моделі гетерогенної мережі вимагає використання операційних систем з принципово різною архітектурою, що дозволяє дослідити специфіку збору телеметрії в різномірних середовищах. У ролі базової платформи для розгортання серверних компонентів та керуючого вузла системи моніторингу обрано дистрибутив Ubuntu Server. Ця операційна система є галузевим стандартом для розміщення веб-сервісів та інфраструктурних рішень, забезпечуючи стабільну роботу компонентів Wazuh Manager та підтримку широкого спектра мережеских утиліт. Використання Linux-середовища дозволяє дослідити механізми аудиту системних викликів, контролю конфігураційних файлів та обробки потоків Syslog, що є критичним для захисту серверного сегмента корпоративної мережі. Для емуляції клієнтських робочих станцій та специфічних корпоративних сервісів до складу тестового стенда введено операційну систему сімейства Windows. Її наявність є необхідною умовою для перевірки кросплатформних можливостей агентів моніторингу, оскільки підсистема аудиту Windows базується на пропрієтарних механізмах Event Log та системному реєстрі, які суттєво відрізняються від Unix-подібних систем. Інтеграція Windows-хоста дозволяє реалізувати сценарії виявлення загроз, характерних для цієї екосистеми, зокрема аналіз цілісності системних бібліотек, виявлення шкідливого програмного забезпечення та моніторинг змін у критичних гілках реєстру, забезпечуючи повноту покриття векторів атак у змішаній інфраструктурі. Невіддільною складовою сучасного ландшафту загроз є контейнеризовані додатки, для розгортання яких використано платформу Docker. Впровадження технології контейнеризації у тестовий стенд зумовлене необхідністю дослідження специфіки моніторингу ефемерних середовищ, де життєвий цикл сервісів може бути коротким, а файлова система — віртуалізованою. Контейнери генерують потоки подій у структурованому форматі JSON, що вимагає від системи SIEM застосування спеціалізованих декодерів та правил нормалізації даних. Використання Docker дозволяє ізолювати вразливі веб-додатки від основної операційної системи,

створюючи безпечний полігон для моделювання атак та одночасно ускладнюючи задачу кореляції подій між хостом та контейнером [19].

Ефективна верифікація налаштованої системи моніторингу вимагає наявності виділеного середовища, здатного емулювати дії потенційного злоумисника з використанням актуальних технік проникнення. У ролі центрального вузла для генерації шкідливого трафіку обрано спеціалізований дистрибутив Kali Linux, який є загальновизнаним галузевим стандартом для проведення тестувань на проникнення та аудиту безпеки. Використання цієї операційної системи дозволяє зосередити широкий спектр атакуючих векторів в межах одного ізолюваного хоста, усуваючи необхідність ручної компіляції та налаштування окремих утиліт [20]. Це забезпечує відтворюваність результатів експериментів та відповідність генерованих мережових патернів тим слідам, що залишають реальні кіберінциденти. Автоматизація сценаріїв атаки реалізована за допомогою комплексу спеціалізованих утиліт, спрямованих на експлуатацію конкретних вразливостей. Для перевірки стійкості підсистем аутентифікації до спроб несанкціонованого доступу застосовано інструментарій Hydra, що дозволяє моделювати інтенсивні паралельні атаки методом повного перебору на протокол SSH [21]. Паралельно з цим, перевірка механізмів захисту веб-застосунків здійснювалася з використанням SQLMap — автоматизованого засобу для виявлення та експлуатації вразливостей типу SQL Injection. Вибір саме цих інструментів зумовлений їхньою здатністю генерувати складні, структуровані послідовності шкідливих запитів, які SIEM-система повинна ідентифікувати серед легітимного трафіку, перевіряючи таким чином коректність роботи правил кореляції. Моделювання цільової інфраструктури передбачало розгортання спеціалізованого програмного забезпечення, що навмисно містить дефекти безпеки. Об'єктом для веб-атак виступив додаток DVWA, розгорнутий у середовищі Docker, який надає контрольований полігон для виконання ін'єкцій коду без правових та етичних ризиків, притаманних тестуванню реальних систем. Додаткова перевірка механізмів захисту кінцевих точок, зокрема модулів контролю цілісності файлів на платформі Windows,

проводилася за допомогою тестового файлу EICAR. Ця стандартизована сигнатура дозволяє безпечно імітувати появу шкідливого програмного забезпечення у файловій системі, викликаючи реакцію засобів захисту без нанесення фактичної шкоди, що є оптимальним для калібрування рівнів тривоги в системі моніторингу.

2.2. Розробка архітектури та топології тестової мережі

Для реалізації практичної частини дослідження в середовищі емуляції GNS3 було розгорнуто мережеву інфраструктуру, центром якої виступає програмний маршрутизатор pfSense. Цей вузол виконує функцію шлюзу, забезпечуючи з'єднання та контроль трафіку між двома ізольованими сегментами. Топологія побудована за класичною схемою розмежування на внутрішню довірену зону та зовнішню мережу, що дозволяє емулювати типове підключення корпоративної інфраструктури до мережі Інтернет та досліджувати проходження мережевих пакетів крізь засоби захисту. Внутрішній сегмент мережі організовано з використанням віртуального некерованого комутатора, до якого приєднано основні обчислювальні ресурси. Ця група пристроїв підключена до LAN-інтерфейсу маршрутизатора та включає сервер моніторингу Wazuh, веб-сервер на базі Ubuntu для розміщення цільового додатка та клієнтську машину з ОС Windows. Об'єднання цих вузлів в одній підмережі спрощує процес налаштування взаємодії між агентами та сервером збору логів, забезпечуючи пряму передачу телеметрії без необхідності налаштування додаткових правил маршрутизації всередині захищеного периметра. Роль зовнішнього загрозового фактора виконує віртуальна машина з операційною системою Kali Linux, яку підключено безпосередньо до WAN-інтерфейсу маршрутизатора pfSense. Таке розміщення атакуючого вузла імітує спроби несанкціонованого доступу з глобальної мережі, змушуючи весь вхідний трафік проходити крізь міжмережевий екран та систему виявлення вторгнень. Запропонована схема дозволяє чітко фіксувати момент перетину периметра

безпеки та аналізувати реакцію захисних механізмів на мережевому рівні ще до того, як шкідливі запити досягнуть цільових серверів у внутрішній мережі.

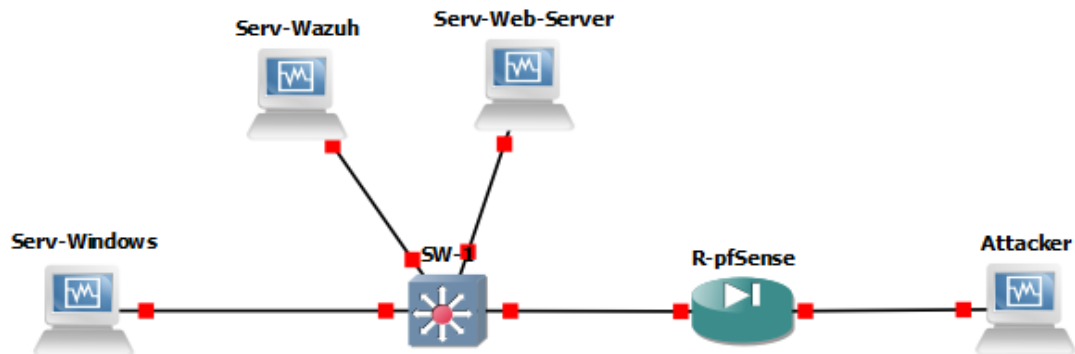


Рис. 2.1 Топологія мережі

Ядром побудованої системи захисту виступає сервер управління Wazuh Manager, розгорнутий на базі операційної системи Ubuntu Server. Цей вузол розміщено у довіреному сегменті внутрішньої мережі, що мінімізує ризики його безпосередньої компрометації з боку зовнішніх загроз та гарантує стабільний канал зв'язку з підконтрольними хостами. Реалізація серверної частини виконана за архітектурою «все-в-одному», де компоненти індексації даних, аналізу подій та веб-інтерфейс візуалізації функціонують у межах однієї віртуальної машини, що дозволяє оптимізувати використання обчислювальних ресурсів тестового стенда без втрати аналітичної функціональності. Взаємодія з агентами безпеки, встановленими на цільових вузлах, організована через виділені TCP-порти 1514 та 1515, які слугують для реєстрації агентів та передачі подій. Цей комунікаційний канал використовує транспортне шифрування для забезпечення конфіденційності та цілісності телеметрії, яка надходить від різноманітних операційних систем. Сервер управління виконує роль центрального аналітичного вузла, акумулюючи різноманітні потоки інформації про стан

файлової системи, зміни конфігурації та системні події, після чого проводить їх нормалізацію та кореляцію в реальному часі відповідно до налаштованих правил виявлення загроз. Для інтеграції подій мережевого рівня в загальну систему моніторингу налаштовано механізм прийому системних повідомлень від прикордонного маршрутизатора pfSense. На стороні сервера Wazuh активовано модуль збору віддалених логів, який прослуховує стандартний UDP-порт 514. Таке архітектурне рішення надає можливість централізовано обробляти журнали брандмауера та сповіщення від системи виявлення вторгнень Snort, дозволяючи корелювати спроби мережевих атак із підозрілою активністю на кінцевих точках та формувати єдиний контекст інцидентів інформаційної безпеки.

Захист серверного сегмента реалізовано на базі віртуальної машини під управлінням Ubuntu Linux, яка виконує роль хоста для контейнеризованих веб-додатків. У цьому середовищі розгорнуто вразливий сервіс DVWA за допомогою платформи Docker, що дозволяє ізолювати прикладне програмне забезпечення від базової операційної системи. Специфіка моніторингу такої архітектури вимагає нестандартного підходу до обробки журналів подій, оскільки стандартний драйвер логування Docker інкапсулює повідомлення веб-сервера у формат JSON, роблячи їх нечитабельними для стандартних декодерів веб-логів. Для вирішення цієї проблеми розроблено та впроваджено механізм потокової обробки даних, який у реальному часі витягує змістовну частину запитів Apache з контейнера та перенаправляє їх у локальний файл на хостовій системі, надаючи агенту Wazuh доступ до «чистих» логів для коректної ідентифікації сигнатур веб-атак. Клієнтський сегмент мережі представлено віртуальною машиною під управлінням операційної системи Windows, що дозволяє дослідити особливості захисту пропрієтарних платформ у змішаній інфраструктурі. На цьому вузлі інстальовано агент Wazuh, який функціонує у фоновому режимі як системна служба та забезпечує безперервний збір телеметрії з журналів подій Windows Event Log. Конфігурація агента включає налаштування модуля контролю цілісності файлів, орієнтованого на відстеження змін у критичних директоріях користувача та системних каталогах. Активація даного функціонала дозволяє

системі фіксувати створення нових файлів у реальному часі, що є необхідною умовою для своєчасного виявлення спроб завантаження шкідливого програмного забезпечення та перевірки реакції системи на появу відомих загроз.

Архітектура тестового середовища передбачає виділення окремої зони для розміщення джерела кібератак, що дозволяє змодельовати сценарій несанкціонованого втручання з боку зовнішніх глобальних мереж. У ролі атакуючого вузла використано віртуальну машину з операційною системою Kali Linux, яку підключено до зовнішнього інтерфейсу маршрутизатора pfSense. Таке топологічне рішення імітує розташування зловмисника за межами контрольованого периметра безпеки, змушуючи весь генерований шкідливий трафік проходити крізь механізми прикордонного захисту. Це створює умови для перевірки ефективності роботи міжмережевого екрана та системи запобігання вторгненням Snort, які повинні аналізувати вхідні пакети до моменту їх потрапляння у внутрішню мережу. Мережева конфігурація зовнішнього сегмента реалізована з використанням окремого адресного простору, ізольованого від внутрішньої локальної мережі, що вимагає налаштування спеціальних правил фільтрації на шлюзі для дозволу проходження трафіку до цільових сервісів. Відсутність прямої маршрутизації за замовчуванням дозволяє контролювати вектори атак, обмежуючи їх лише відкритими портами веб-сервісу та протоколу SSH, які підлягають тестуванню. Генерація атак із зовнішнього сегмента надає можливість дослідити процес кореляції подій у SIEM-системі, об'єднуючи в один логічний ланцюжок алерти мережевого рівня від pfSense та події безпеки, зафіксовані агентами на кінцевих точках усередині захищеного периметра.

2.3. Планування сценаріїв моделювання кібератак

Одним із найбільш розповсюджених векторів компрометації серверної інфраструктури в гетерогенних мережах залишаються спроби несанкціонованого втручання в роботу механізмів віддаленого управління, зокрема протоколу SSH. Для перевірки ефективності налаштованих правил

кореляції подій у тестовому середовищі обрано сценарій автоматизованого підбору облікових даних, відомий як атака методом повного перебору. Цей тип загрози характеризується генерацією аномально високої кількості запитів на встановлення з'єднання за короткий проміжок часу, що створює специфічний патерн у системних журналах та дозволяє однозначно ідентифікувати спробу злому на тлі звичайної адміністративної активності. Реалізація даного сценарію передбачає застосування спеціалізованого інструментарію Hydra, розгорнутого на зовнішньому атакуючому вузлі в зоні WAN. Методика експерименту базується на виконанні багатопотокової атаки за словником, де вхідними параметрами виступають заздалегідь підготовлені масиви типових ідентифікаторів користувачів та найпоширеніших парольних комбінацій. Програмний засіб ініціює серію циклічних спроб автентифікації на цільовому сервері під управлінням ОС Ubuntu через стандартний TCP-порт 22, імітуючи дії зломисника, який намагається отримати первинний доступ до системи шляхом експлуатації слабкої парольної політики. Реєстрація інциденту відбувається на рівні операційної системи жертви, де служба SSH-демона фіксує кожну невдалу спробу входу у локальному файлі журналу безпеки `/var/log/auth.log` або системному журналі. Агент моніторингу здійснює безперервне зчитування нових записів у цьому файлі та їх нормалізацію перед відправкою на сервер аналізу. Ключовим індикатором успішного виявлення атаки в цьому сценарії є спрацювання правила кореляції, яке реагує на перевищення порогового значення кількості подій помилки автентифікації з однієї джерельної IP-адреси протягом заданого часового вікна, що дозволяє підтвердити здатність системи розрізняти поодинокі помилки користувачів та цілеспрямовані атаки на підбір паролів.

Дослідження безпеки прикладного рівня вимагає відтворення сценаріїв атак на веб-сервіси, які функціонують в ізольованих середовищах мікросервісної архітектури. Об'єктом для проведення експерименту обрано веб-додаток DVWA, розгорнутий у контейнері Docker, що дозволяє емулювати типову структуру сучасних хмарних сервісів. Основний фокус зосереджено на моделюванні вразливості типу SQL Injection, яка залишається критичною

загрозою для конфіденційності даних. Специфіка контейнеризації накладає суттєві обмеження на доступність журналів подій, оскільки стандартні механізми аудиту всередині контейнера ізольовані від хостової системи, що вимагає побудови спеціалізованих каналів передачі телеметрії для коректного аналізу інцидентів та виявлення аномалій у запитах до бази даних. Автоматизація процесу експлуатації вразливості здійснюється за допомогою інструментарію SQLMap, здатного генерувати складні варіації шкідливих запитів для обходу базових фільтрів веб-додатків. Під час планування експерименту було ідентифіковано технічну проблему, пов'язану з обмеженнями стандартних конфігурацій логування веб-серверів при обробці HTTP-запитів методу POST. Шкідливе навантаження, що містить синтаксис SQL-ін'єкцій, у такому випадку передається в тілі запиту та за замовчуванням не фіксується у стандартному журналі доступу, створюючи "сліпу зону" для систем сигнатурного аналізу. Відсутність деталізації тіла запиту у логах призводить до ситуації, коли факт атаки залишається невидимим для агента моніторингу, навіть за умови успішного виконання ін'єкції на стороні серверної частини. Подолання виявленого обмеження видимості реалізовано шляхом адаптації методики атаки з використанням методу HTTP GET, що змушує передавати параметри запиту безпосередньо в адресному рядку URL. Такий підхід гарантує повне відображення шкідливого синтаксису, включаючи оператори об'єднання запитів та логічні маніпулятори, у журналах подій, що експортуються з контейнера через потоки вводу-виводу. Наявність повної інформації про структуру запиту дозволяє системі моніторингу застосовувати сигнатурний метод виявлення, зіставляючи патерни в URL із правилами безпеки. Успішна ідентифікація загрози в цьому сценарії підтверджує коректність налаштування ланцюжка обробки даних, що включає перехоплення JSON-потoku з Docker, його декодування та аналіз на предмет наявності ознак компрометації веб-додатка.

Забезпечення безпеки кінцевих точок у гетерогенній мережі вимагає впровадження механізмів контролю за станом файлової системи, здатних виявляти несанкціоновані зміни та появу потенційно небезпечних об'єктів. Для

перевірки ефективності роботи підсистеми FIM та модулів антивірусного сканування обрано сценарій імітації зараження клієнтської робочої станції під управлінням ОС Windows. Ця операційна система є найбільш вразливою ланкою корпоративної інфраструктури через широке розповсюдження шкідливого програмного забезпечення, орієнтованого на кінцевого користувача. Методика експерименту передбачає доставку тестового зразка загрози на цільовий хост через мережевий канал, відтворюючи типовий вектор атаки типу Drive-by Download або завантаження файлу користувачем із недовіреного джерела. У ролі імітатора шкідливого об'єкта використано стандартизований тестовий файл EICAR. Цей файл являє собою безпечний рядок ASCII-символів, який, згідно з міжнародними стандартами кібербезпеки, повинен ідентифікуватися будь-яким засобом захисту як вірусна сигнатура [22]. Використання EICAR дозволяє безпечно протестувати весь ланцюжок реагування SIEM-системи без ризику реального зараження інфраструктури чи пошкодження даних. Для реалізації сценарію на атакуючому вузлі розгортається тимчасовий веб-сервер, який обслуговує файл із сигнатурою, а на стороні жертви ініціюється примусове завантаження цього об'єкта у директорію, що перебуває під наглядом агента моніторингу. Критерієм успішності експерименту є фіксація системою моніторингу двох послідовних подій: факту створення нового файлу у файловій системі та результату його сканування на наявність відомих загроз. Агент Wazuh повинен в автоматичному режимі виявити зміни в атрибутах файлової системи, передати метадані новоствореного об'єкта на сервер для аналізу та зіставити їх із базою сигнатур загроз або інтегрованими сервісами Threat Intelligence. Коректна обробка цього інциденту дозволяє підтвердити працездатність механізмів виявлення прихованих загроз, які можуть оминати традиційні засоби периметрового захисту та потрапляти безпосередньо на робочі станції користувачів.

Критичним компонентом сучасної системи кіберзахисту є здатність не лише виявляти інциденти, а й оперативно нейтралізувати загрози в автоматичному режимі, мінімізуючи час перебування зловмисника в мережі. Для

реалізації цієї функції в рамках дослідження розроблено алгоритм активного реагування, який базується на динамічній взаємодії між сервером аналізу та агентами на кінцевих точках. Сценарій реагування спроектовано для протидії атакам на мережеві сервіси, зокрема спробам підбору паролів, де швидкість реакції є вирішальним фактором. Логіка роботи механізму передбачає виконання скриптів блокування на рівні брандмауера хоста у відповідь на спрацювання специфічних правил кореляції високого рівня критичності. Процес налаштування автоматизації включає створення кастомних правил кореляції, адаптованих під конкретні пороги спрацювання. Стандартні правила виявлення атак методом повного перебору часто мають консервативні налаштування для уникнення хибних спрацювань, тому в рамках експерименту передбачено модифікацію параметрів частоти та часового вікна для правила, що відстежує невдалі спроби SSH-автентифікації. Це дозволяє підвищити чутливість системи та ініціювати захисні дії на ранніх етапах атаки. Додатково розроблено логіку виявлення складних, багатоетапних інцидентів, де факт успішного входу в систему після серії невдалих спроб розглядається як критична подія компрометації облікового запису, що вимагає негайної ізоляції хоста. Технічна реалізація механізму блокування здійснюється через інтеграцію модуля Active Response із системним міжмережевим екраном операційної системи. При детектуванні атаки сервер Wazuh надсилає команду агенту на атакованому вузлі, який, у свою чергу, виконує скрипт `firewall-drop` для внесення IP-адреси зломисника до списку заборонених. Алгоритм передбачає тимчасове блокування джерела загрози на фіксований проміжок часу, що дозволяє зупинити атаку без необхідності постійного втручання адміністратора, але зберігає доступність сервісу для легітимних користувачів після завершення інциденту. Перевірка ефективності цього механізму є завершальним етапом експерименту, підтверджуючи здатність системи функціонувати в режимі реального часу як повноцінна платформа XDR.

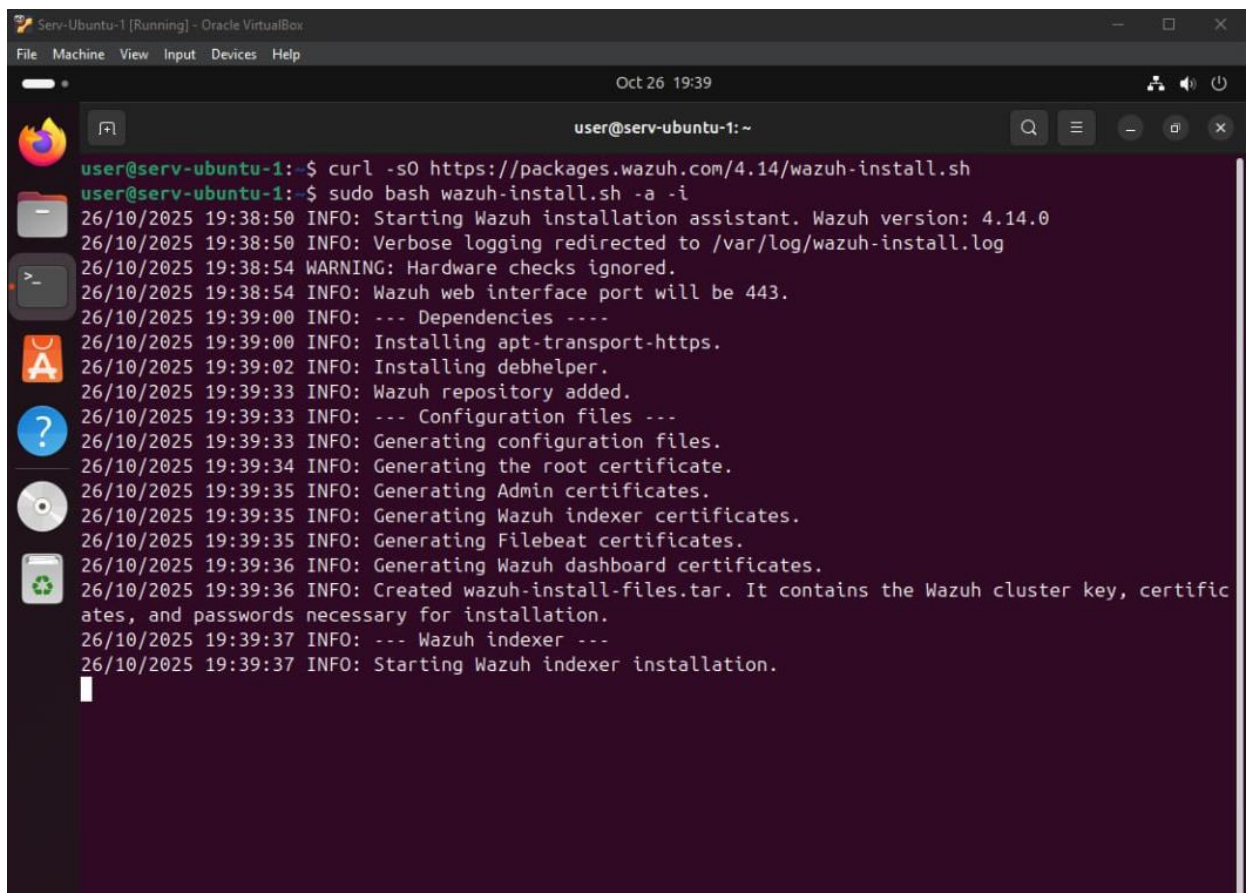
Висновки до другого розділу:

Обґрунтування вибору інструментального комплексу на базі платформи Wazuh та середовища віртуалізації дозволило спроектувати гнучку архітектуру системи моніторингу, адаптовану до специфіки гетерогенних мереж. Розроблена топологія тестового стенда, що поєднує різноманітні операційні системи, контейнеризовані додатки та засоби периметрового захисту, забезпечує реалістичне моделювання корпоративної інфраструктури та створює умови для дослідження повного циклу обробки інцидентів. Визначення ключових векторів атак, включаючи спроби несанкціонованого доступу, веб-ін'єкції та розповсюдження шкідливого програмного забезпечення, разом із плануванням специфічних каналів збору телеметрії, сформувало необхідний методологічний фундамент для подальшої практичної верифікації ефективності механізмів виявлення та автоматизованого реагування.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ СИСТЕМИ МОНІТОРИНГУ

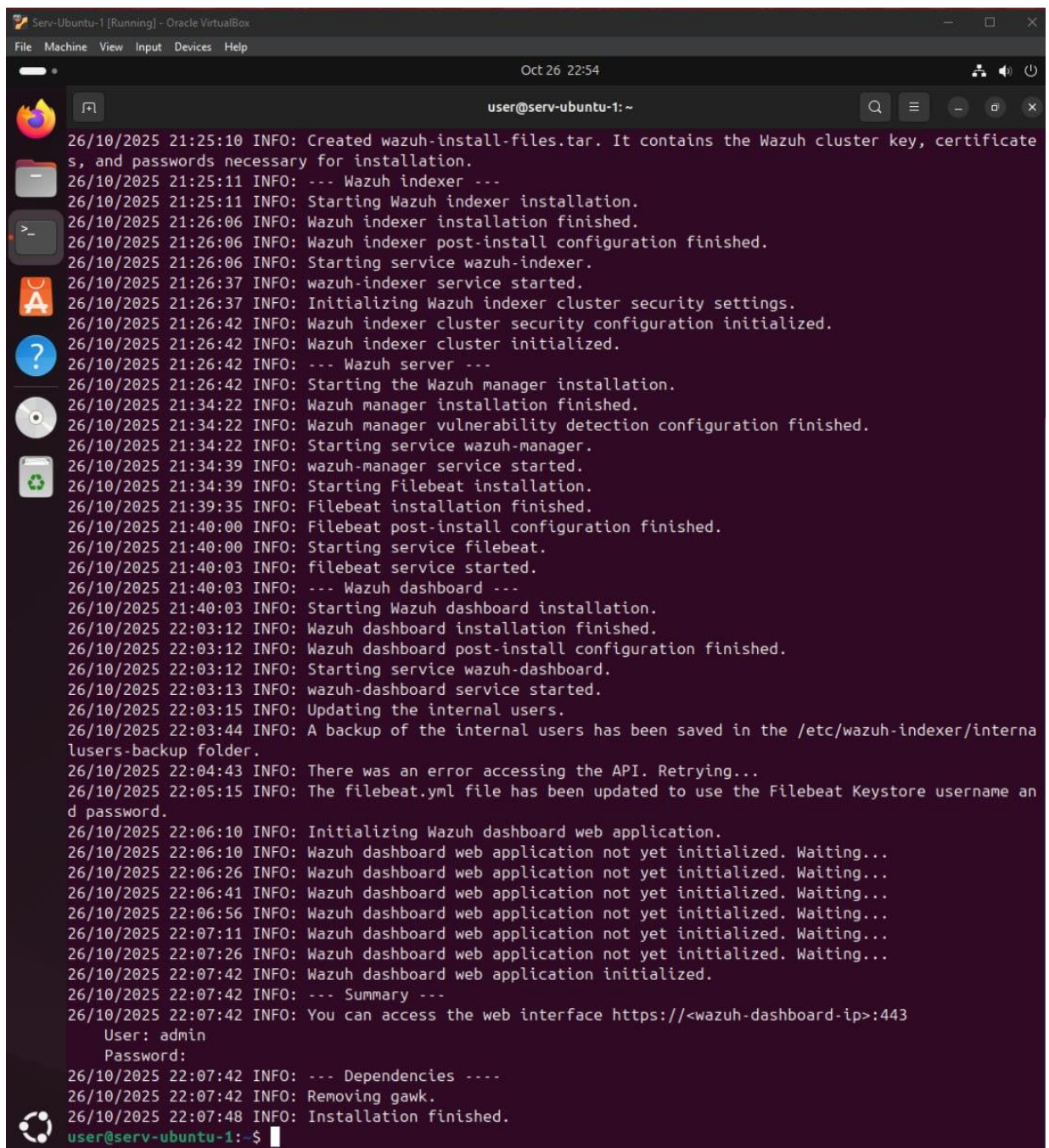
3.1. Розгортання та налаштування SIEM-системи

Розгортання центрального вузла системи моніторингу розпочалося з інсталяції керуючого сервера Wazuh Manager на базі операційної системи Ubuntu Server. Вибір даної платформи зумовлений її стабільністю та нативною підтримкою компонентів архітектури ELK, що лежать в основі підсистеми індексації та візуалізації даних. Процес встановлення здійснювався з використанням автоматизованого скрипта розгортання, який забезпечує коректну інтеграцію трьох ключових модулів: сервера обробки подій, індексатора для зберігання історичних даних та веб-інтерфейсу Dashboard. Така архітектура типу «все-в-одному» дозволяє зосередити обчислювальні потужності в межах однієї віртуальної машини, спрощуючи адміністрування тестового середовища.



```
user@serv-ubuntu-1: ~  
user@serv-ubuntu-1:~$ curl -sO https://packages.wazuh.com/4.14/wazuh-install.sh  
user@serv-ubuntu-1:~$ sudo bash wazuh-install.sh -a -i  
26/10/2025 19:38:50 INFO: Starting Wazuh installation assistant. Wazuh version: 4.14.0  
26/10/2025 19:38:50 INFO: Verbose logging redirected to /var/log/wazuh-install.log  
26/10/2025 19:38:54 WARNING: Hardware checks ignored.  
26/10/2025 19:38:54 INFO: Wazuh web interface port will be 443.  
26/10/2025 19:39:00 INFO: --- Dependencies ---  
26/10/2025 19:39:00 INFO: Installing apt-transport-https.  
26/10/2025 19:39:02 INFO: Installing debhelper.  
26/10/2025 19:39:33 INFO: Wazuh repository added.  
26/10/2025 19:39:33 INFO: --- Configuration files ---  
26/10/2025 19:39:33 INFO: Generating configuration files.  
26/10/2025 19:39:34 INFO: Generating the root certificate.  
26/10/2025 19:39:35 INFO: Generating Admin certificates.  
26/10/2025 19:39:35 INFO: Generating Wazuh indexer certificates.  
26/10/2025 19:39:35 INFO: Generating Filebeat certificates.  
26/10/2025 19:39:36 INFO: Generating Wazuh dashboard certificates.  
26/10/2025 19:39:36 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificates, and passwords necessary for installation.  
26/10/2025 19:39:37 INFO: --- Wazuh indexer ---  
26/10/2025 19:39:37 INFO: Starting Wazuh indexer installation.
```

Рис. 3.1 Інсталяція пакунку керуючого сервера Wazuh Manager на Serv-Wazuh



```
Serv-Ubuntu-1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Oct 26 22:54
user@serv-ubuntu-1: ~
26/10/2025 21:25:10 INFO: Created wazuh-install-files.tar. It contains the Wazuh cluster key, certificate
s, and passwords necessary for installation.
26/10/2025 21:25:11 INFO: --- Wazuh indexer ---
26/10/2025 21:25:11 INFO: Starting Wazuh indexer installation.
26/10/2025 21:26:06 INFO: Wazuh indexer installation finished.
26/10/2025 21:26:06 INFO: Wazuh indexer post-install configuration finished.
26/10/2025 21:26:06 INFO: Starting service wazuh-indexer.
26/10/2025 21:26:37 INFO: wazuh-indexer service started.
26/10/2025 21:26:37 INFO: Initializing Wazuh indexer cluster security settings.
26/10/2025 21:26:42 INFO: Wazuh indexer cluster security configuration initialized.
26/10/2025 21:26:42 INFO: Wazuh indexer cluster initialized.
26/10/2025 21:26:42 INFO: --- Wazuh server ---
26/10/2025 21:26:42 INFO: Starting the Wazuh manager installation.
26/10/2025 21:34:22 INFO: Wazuh manager installation finished.
26/10/2025 21:34:22 INFO: Wazuh manager vulnerability detection configuration finished.
26/10/2025 21:34:22 INFO: Starting service wazuh-manager.
26/10/2025 21:34:39 INFO: wazuh-manager service started.
26/10/2025 21:34:39 INFO: Starting Filebeat installation.
26/10/2025 21:39:35 INFO: Filebeat installation finished.
26/10/2025 21:40:00 INFO: Filebeat post-install configuration finished.
26/10/2025 21:40:00 INFO: Starting service filebeat.
26/10/2025 21:40:03 INFO: filebeat service started.
26/10/2025 21:40:03 INFO: --- Wazuh dashboard ---
26/10/2025 21:40:03 INFO: Starting Wazuh dashboard installation.
26/10/2025 22:03:12 INFO: Wazuh dashboard installation finished.
26/10/2025 22:03:12 INFO: Wazuh dashboard post-install configuration finished.
26/10/2025 22:03:12 INFO: Starting service wazuh-dashboard.
26/10/2025 22:03:13 INFO: wazuh-dashboard service started.
26/10/2025 22:03:15 INFO: Updating the internal users.
26/10/2025 22:03:44 INFO: A backup of the internal users has been saved in the /etc/wazuh-indexer/interna
lusers-backup folder.
26/10/2025 22:04:43 INFO: There was an error accessing the API. Retrying...
26/10/2025 22:05:15 INFO: The filebeat.yml file has been updated to use the Filebeat Keystore username an
d password.
26/10/2025 22:06:10 INFO: Initializing Wazuh dashboard web application.
26/10/2025 22:06:10 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:06:26 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:06:41 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:06:56 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:07:11 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:07:26 INFO: Wazuh dashboard web application not yet initialized. Waiting...
26/10/2025 22:07:42 INFO: Wazuh dashboard web application initialized.
26/10/2025 22:07:42 INFO: --- Summary ---
26/10/2025 22:07:42 INFO: You can access the web interface https://<wazuh-dashboard-ip>:443
User: admin
Password:
26/10/2025 22:07:42 INFO: --- Dependencies ----
26/10/2025 22:07:42 INFO: Removing gawk.
26/10/2025 22:07:48 INFO: Installation finished.
user@serv-ubuntu-1:~$
```

Рис. 3.2 Інсталяція пакунку керуючого сервера Wazuh Manager на Serv-Wazuh

Після завершення інсталяції було проведено верифікацію статусу служб та налаштування параметрів споживання оперативної пам'яті Java-процесами індексатора для забезпечення стабільної продуктивності в умовах обмежених ресурсів віртуального стенда.

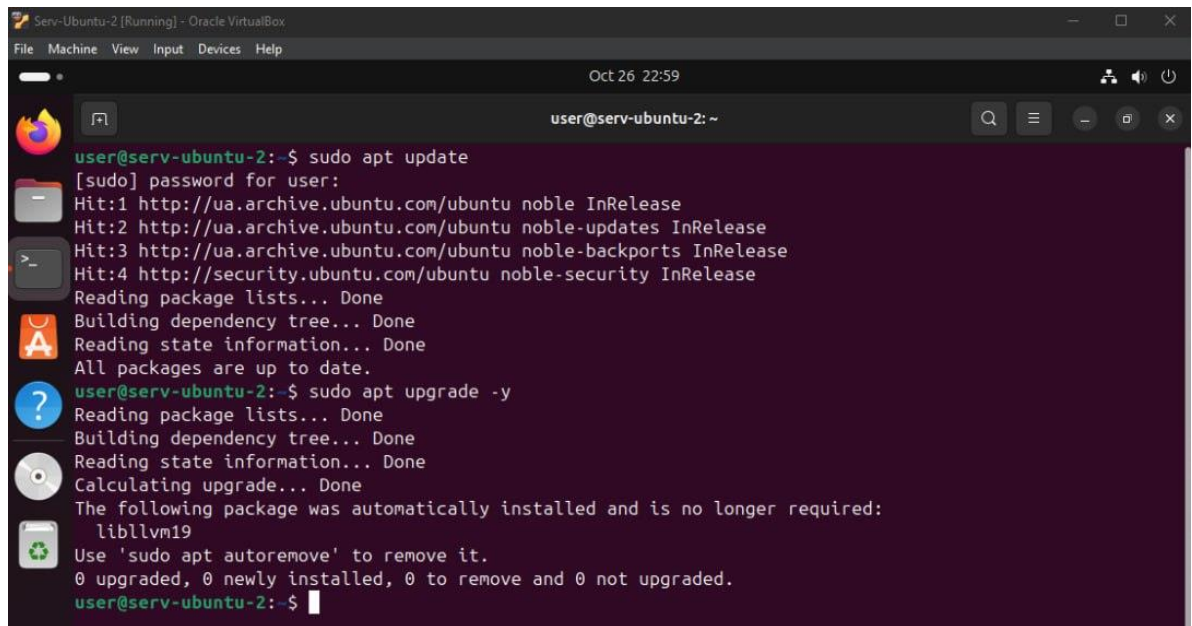
Забезпечення комунікації між сервером та периферійними пристроями вимагало внесення змін до налаштувань мережевого екрана операційної системи. Для коректної реєстрації нових агентів та прийому телеметрії було відкрито

TCP-порти 1514 та 1515, які відповідають за захищений транспорт даних та сервіс авторизації. Окрім стандартних каналів зв'язку з агентами, конфігурація системи передбачала інтеграцію мережевого обладнання, що не підтримує встановлення стороннього програмного забезпечення. З цією метою у головному конфігураційному файлі `ossec.conf` було активовано модуль `remote`, налаштований на прослуховування вхідного трафіку на стандартному UDP-порту 514. Це дозволило серверу приймати та обробляти події у форматі Syslog, що надходять від маршрутизатора pfSense, розширюючи зону видимості SIEM-системи на мережевий рівень.

Етап початкового налагодження системи супроводжувався тимчасовою модифікацією параметрів логування для перевірки цілісності надходження даних. У конфігурації менеджера було активовано опцію `<logall>`, що змушує систему зберігати повний архів усіх отриманих подій у текстовому форматі, незалежно від того, чи спрацювало на них правило кореляції. Такий підхід надав можливість детально аналізувати "сирі" дані, що надходять від різних джерел, та виявляти проблеми з декодуванням нестандартних форматів логів на ранніх стадіях. Після підтвердження коректності парсингу вхідних потоків та налаштування базових правил фільтрації, режим повного логування було деактивовано для економії дискового простору та зменшення навантаження на підсистему вводу-виводу.

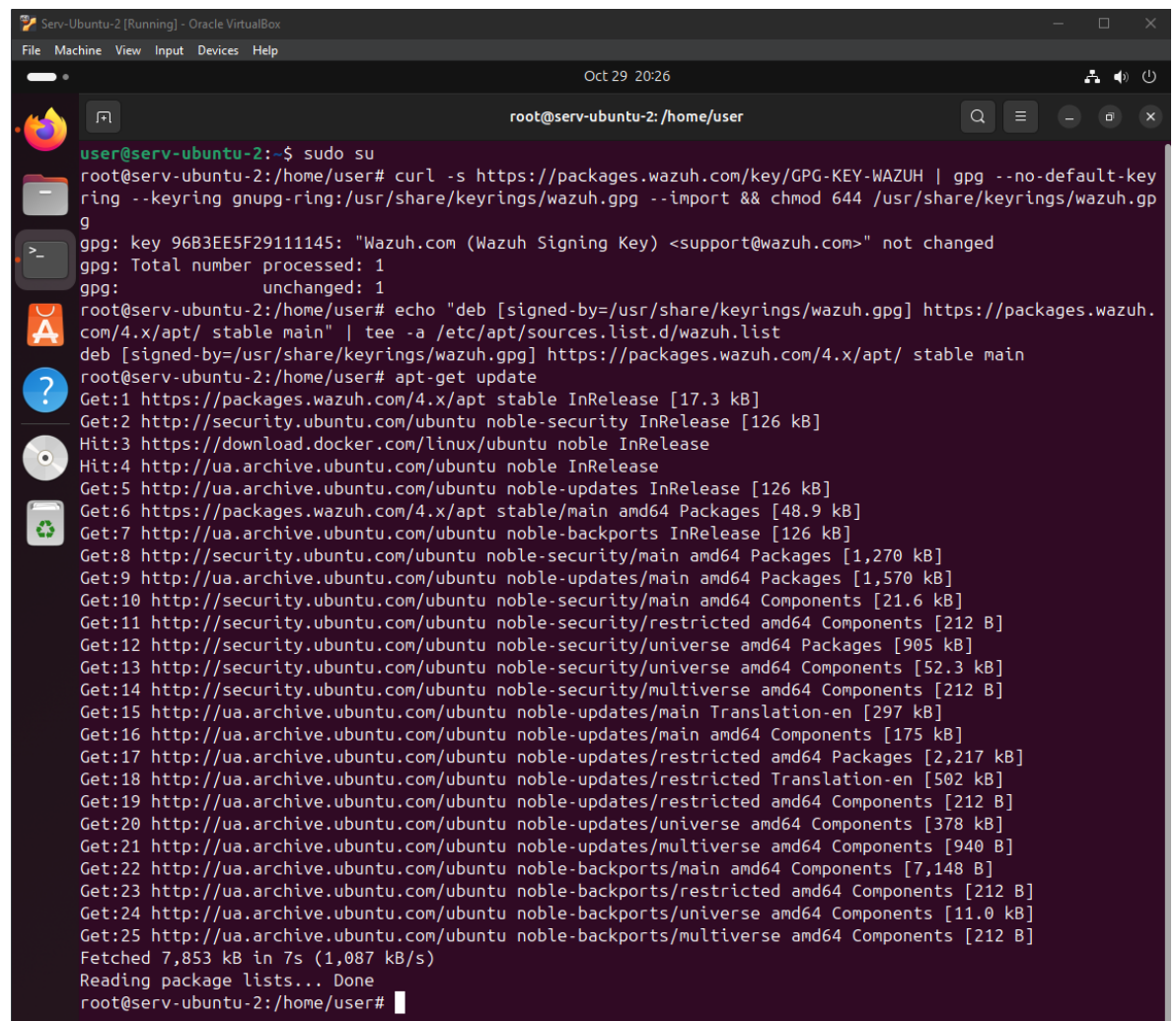
Побудова розподіленої архітектури збору даних розпочалася з інтеграції серверного сегмента мережі, що функціонує під управлінням операційної системи Ubuntu. Процедура розгортання агента на цій платформі здійснювалася шляхом додавання офіційного репозиторію розробника до системного менеджера пакунків, що гарантує отримання актуальної версії програмного забезпечення та перевірку цифрових підписів. Конфігурація клієнтської частини вимагала внесення адреси керуючого сервера у файл налаштувань, після чого ініціювався процес реєстрації. Під час першого запуску служби агент автоматично виконав обмін криптографічними ключами з сервером, отримавши унікальний ідентифікатор та встановивши захищений тунель для передачі

телеметрії, що дозволило негайно розпочати моніторинг стану системних демонів та журналів безпеки.



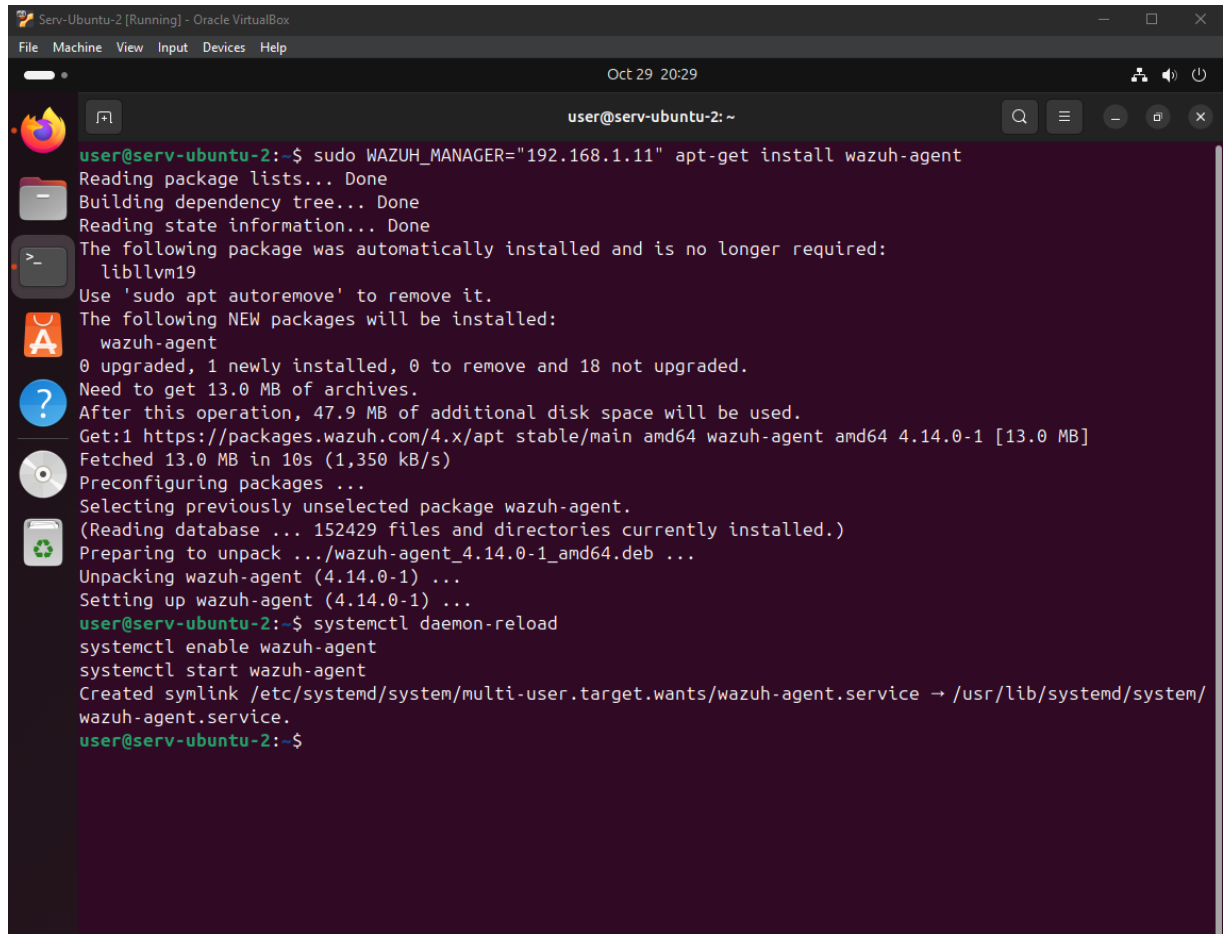
```
user@serv-ubuntu-2:~$ sudo apt update
[sudo] password for user:
Hit:1 http://ua.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ua.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ua.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
user@serv-ubuntu-2:~$ sudo apt upgrade -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following package was automatically installed and is no longer required:
  liblvm19
Use 'sudo apt autoremove' to remove it.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
user@serv-ubuntu-2:~$
```

Рис. 3.3 Оновлення пакунків на Serv-Web-Server



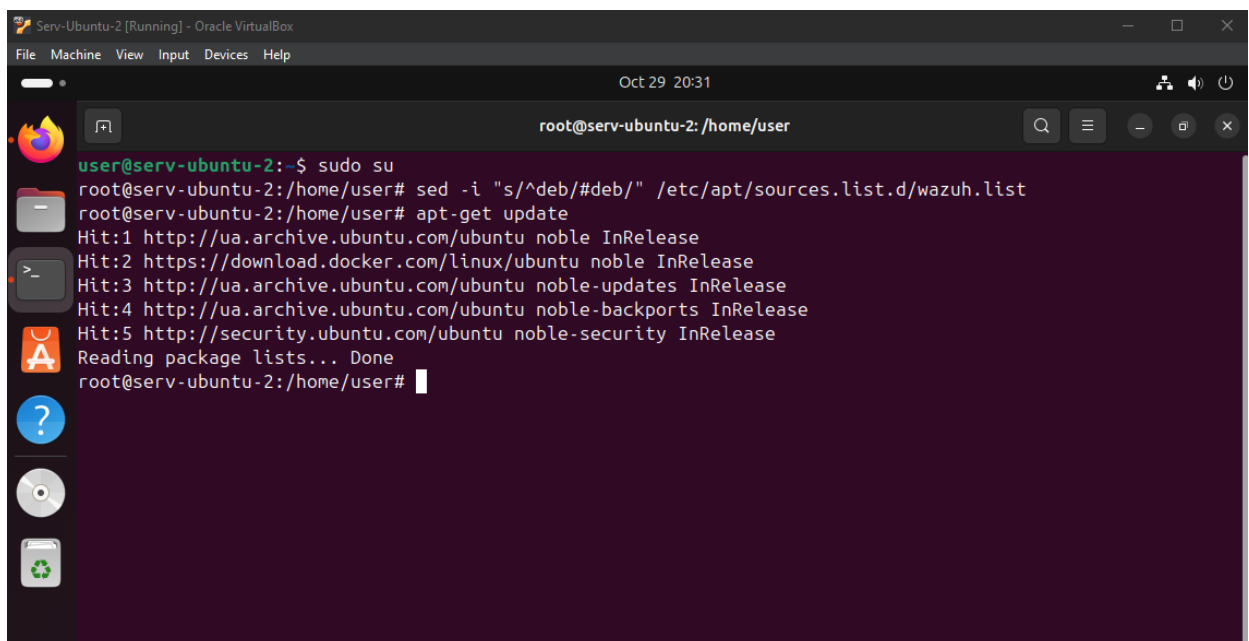
```
user@serv-ubuntu-2:~$ sudo su
root@serv-ubuntu-2:/home/user# curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | gpg --no-default-keyring --keyring gnupg-ring:/usr/share/keyrings/wazuh.gpg --import && chmod 644 /usr/share/keyrings/wazuh.gpg
gpg: key 96B3EE5F29111145: "Wazuh.com (Wazuh Signing Key) <support@wazuh.com>" not changed
gpg: Total number processed: 1
gpg: unchanged: 1
root@serv-ubuntu-2:/home/user# echo "deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable main" | tee -a /etc/apt/sources.list.d/wazuh.list
deb [signed-by=/usr/share/keyrings/wazuh.gpg] https://packages.wazuh.com/4.x/apt/ stable main
root@serv-ubuntu-2:/home/user# apt-get update
Get:1 https://packages.wazuh.com/4.x/apt stable InRelease [17.3 kB]
Get:2 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Hit:3 https://download.docker.com/linux/ubuntu noble InRelease
Hit:4 http://ua.archive.ubuntu.com/ubuntu noble InRelease
Get:5 http://ua.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:6 https://packages.wazuh.com/4.x/apt stable/main amd64 Packages [48.9 kB]
Get:7 http://ua.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1,270 kB]
Get:9 http://ua.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1,570 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [905 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.3 kB]
Get:14 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:15 http://ua.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [297 kB]
Get:16 http://ua.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [175 kB]
Get:17 http://ua.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [2,217 kB]
Get:18 http://ua.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [502 kB]
Get:19 http://ua.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:20 http://ua.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [378 kB]
Get:21 http://ua.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:22 http://ua.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [7,148 B]
Get:23 http://ua.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [212 B]
Get:24 http://ua.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [11.0 kB]
Get:25 http://ua.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Fetched 7,853 kB in 7s (1,087 kB/s)
Reading package lists... Done
root@serv-ubuntu-2:/home/user#
```

Рис. 3.4 Встановлення агенту Wazuh на Serv-Web-Server



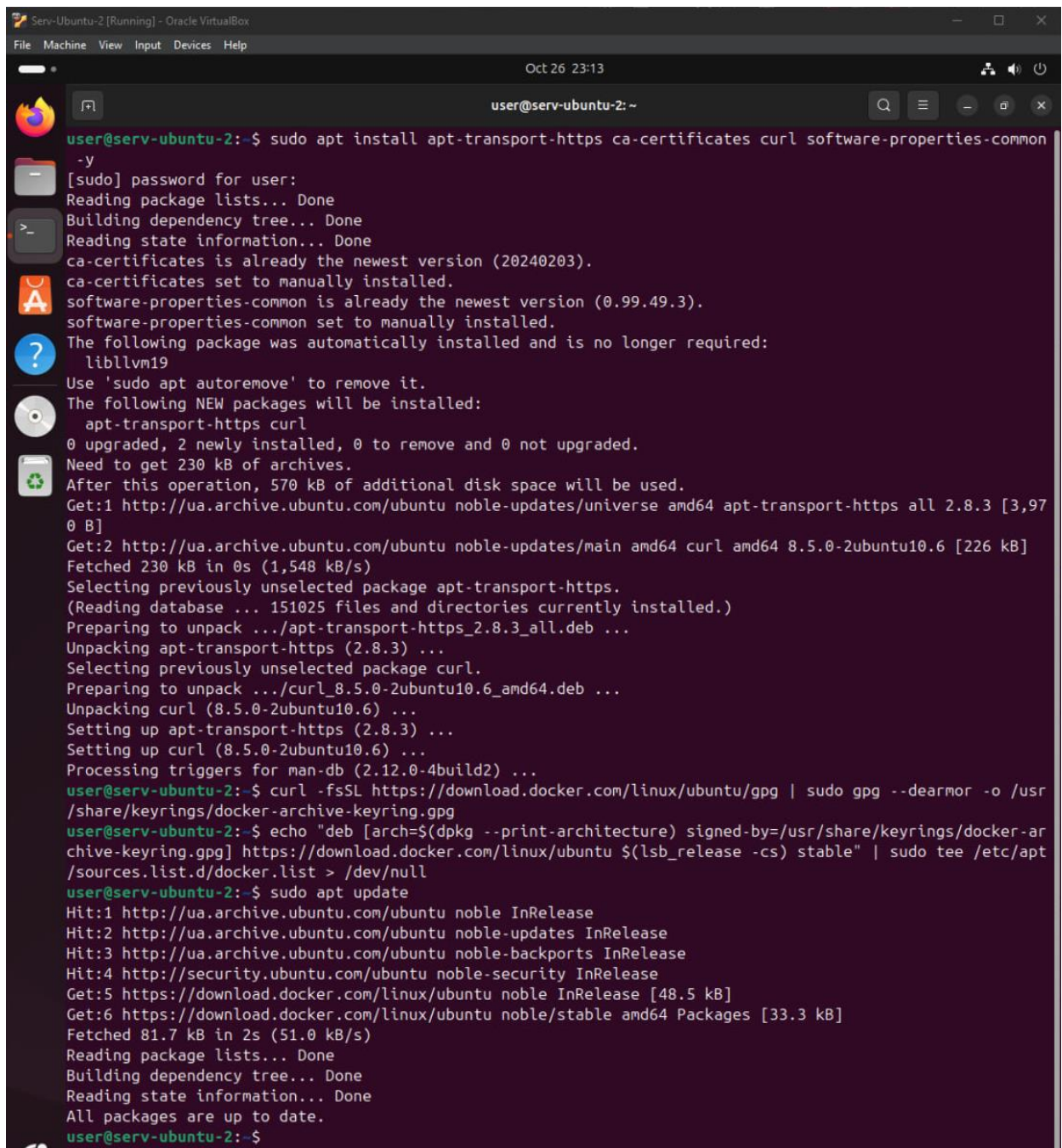
```
Serv-Ubuntu-2 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Oct 29 20:29
user@serv-ubuntu-2: ~
user@serv-ubuntu-2:~$ sudo WAZUH_MANAGER="192.168.1.11" apt-get install wazuh-agent
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  wazuh-agent
0 upgraded, 1 newly installed, 0 to remove and 18 not upgraded.
Need to get 13.0 MB of archives.
After this operation, 47.9 MB of additional disk space will be used.
Get:1 https://packages.wazuh.com/4.x/apt/stable/main amd64 wazuh-agent amd64 4.14.0-1 [13.0 MB]
Fetched 13.0 MB in 10s (1,350 kB/s)
Preconfiguring packages ...
Selecting previously unselected package wazuh-agent.
(Reading database ... 152429 files and directories currently installed.)
Preparing to unpack .../wazuh-agent_4.14.0-1_amd64.deb ...
Unpacking wazuh-agent (4.14.0-1) ...
Setting up wazuh-agent (4.14.0-1) ...
user@serv-ubuntu-2:~$ systemctl daemon-reload
systemctl enable wazuh-agent
systemctl start wazuh-agent
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service → /usr/lib/systemd/system/
wazuh-agent.service.
user@serv-ubuntu-2:~$
```

Рис. 3.5 Встановлення агенту Wazuh на Serv-Web-Server



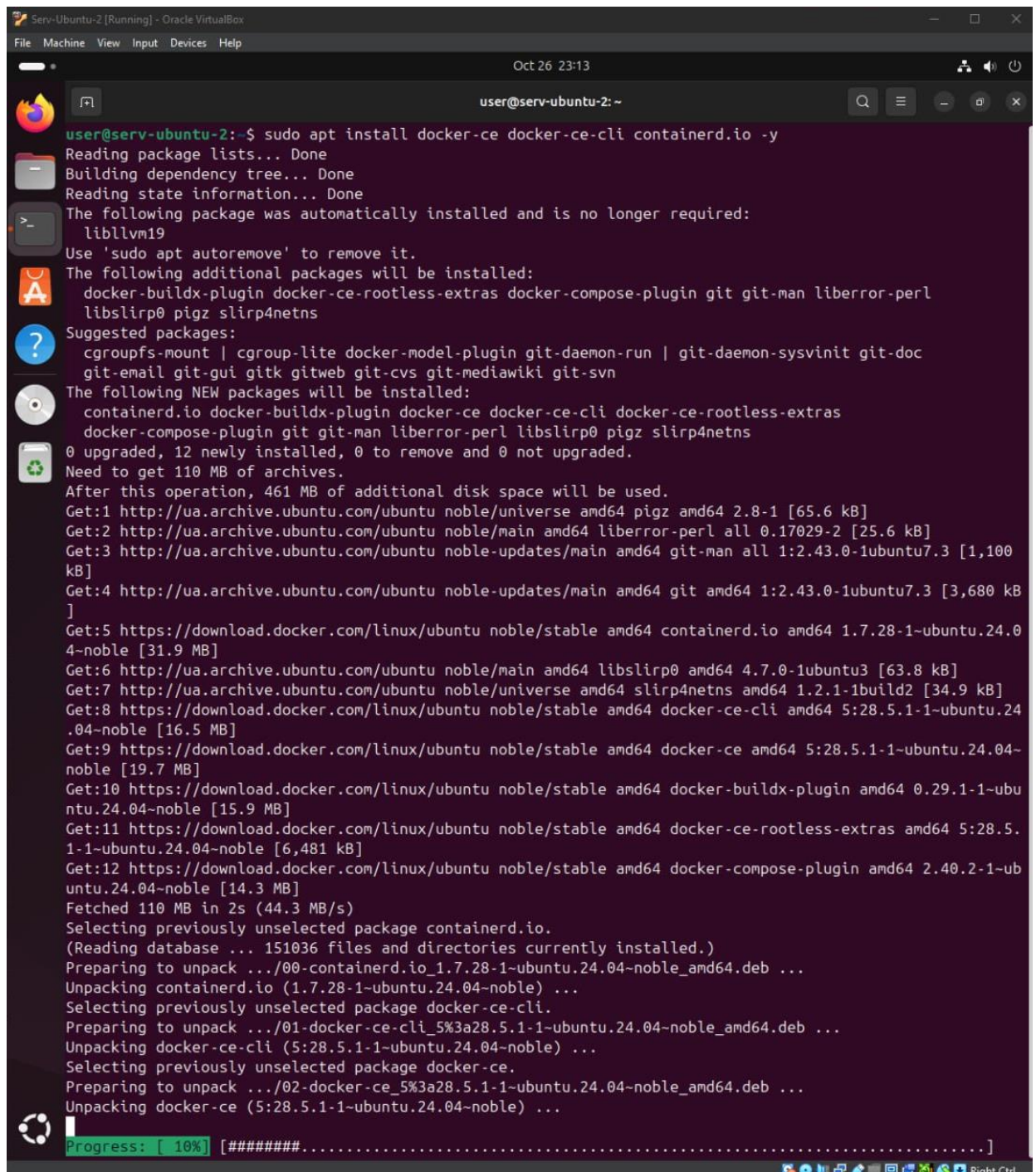
```
Serv-Ubuntu-2 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Oct 29 20:31
root@serv-ubuntu-2: /home/user
user@serv-ubuntu-2:~$ sudo su
root@serv-ubuntu-2:/home/user# sed -i "s/^deb/#deb/" /etc/apt/sources.list.d/wazuh.list
root@serv-ubuntu-2:/home/user# apt-get update
Hit:1 http://ua.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 https://download.docker.com/linux/ubuntu noble InRelease
Hit:3 http://ua.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:4 http://ua.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
root@serv-ubuntu-2:/home/user#
```

Рис. 3.6 Відміна оновлень для агенту Wazuh на Serv-Web-Server



```
user@serv-ubuntu-2:~$ sudo apt install apt-transport-https ca-certificates curl software-properties-common
-y
[sudo] password for user:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
ca-certificates is already the newest version (20240203).
ca-certificates set to manually installed.
software-properties-common is already the newest version (0.99.49.3).
software-properties-common set to manually installed.
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  apt-transport-https curl
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 230 kB of archives.
After this operation, 570 kB of additional disk space will be used.
Get:1 http://ua.archive.ubuntu.com/ubuntu noble-updates/universe amd64 apt-transport-https all 2.8.3 [3,97
0 B]
Get:2 http://ua.archive.ubuntu.com/ubuntu noble-updates/main amd64 curl amd64 8.5.0-2ubuntu10.6 [226 kB]
Fetched 230 kB in 0s (1,548 kB/s)
Selecting previously unselected package apt-transport-https.
(Reading database ... 151025 files and directories currently installed.)
Preparing to unpack .../apt-transport-https_2.8.3_all.deb ...
Unpacking apt-transport-https (2.8.3) ...
Selecting previously unselected package curl.
Preparing to unpack .../curl_8.5.0-2ubuntu10.6_amd64.deb ...
Unpacking curl (8.5.0-2ubuntu10.6) ...
Setting up apt-transport-https (2.8.3) ...
Setting up curl (8.5.0-2ubuntu10.6) ...
Processing triggers for man-db (2.12.0-4build2) ...
user@serv-ubuntu-2:~$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr
/share/keyrings/docker-archive-keyring.gpg
user@serv-ubuntu-2:~$ echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-ar
chive-keyring.gpg] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee /etc/apt
/sources.list.d/docker.list > /dev/null
user@serv-ubuntu-2:~$ sudo apt update
Hit:1 http://ua.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ua.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ua.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://download.docker.com/linux/ubuntu noble InRelease [48.5 kB]
Get:6 https://download.docker.com/linux/ubuntu noble/stable amd64 Packages [33.3 kB]
Fetched 81.7 kB in 2s (51.0 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
user@serv-ubuntu-2:~$
```

Рис. 3.7 Встановлення необхідних пакунків на Serv-Web-Server



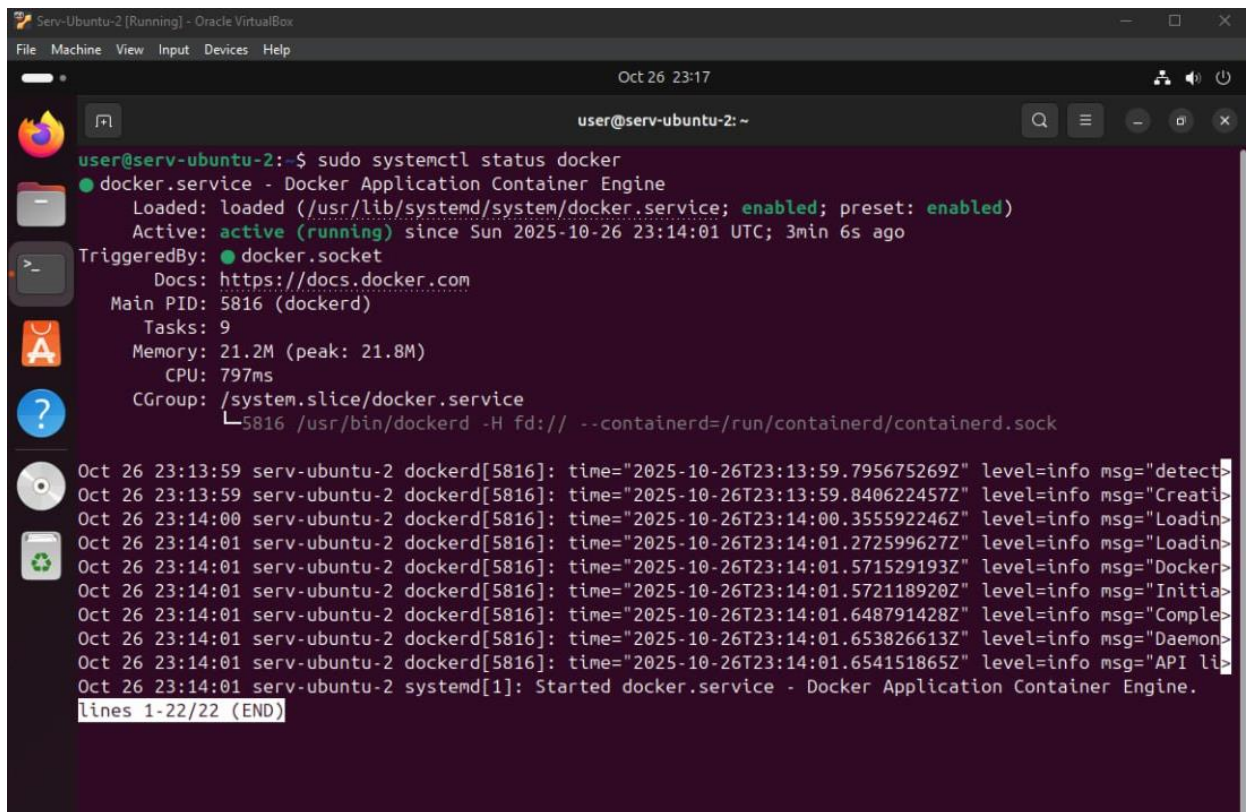
```
user@serv-ubuntu-2:~$ sudo apt install docker-ce docker-ce-cli containerd.io -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following package was automatically installed and is no longer required:
  libllvm19
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  docker-buildx-plugin docker-ce-rootless-extras docker-compose-plugin git git-man liberror-perl
  libslirp0 pigz slirp4netns
Suggested packages:
  cgroupfs-mount | cgroup-lite docker-model-plugin git-daemon-run | git-daemon-sysvinit git-doc
  git-email git-gui gitk gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  containerd.io docker-buildx-plugin docker-ce docker-ce-cli docker-ce-rootless-extras
  docker-compose-plugin git git-man liberror-perl libslirp0 pigz slirp4netns
0 upgraded, 12 newly installed, 0 to remove and 0 not upgraded.
Need to get 110 MB of archives.
After this operation, 461 MB of additional disk space will be used.
Get:1 http://ua.archive.ubuntu.com/ubuntu noble/universe amd64 pigz amd64 2.8-1 [65.6 kB]
Get:2 http://ua.archive.ubuntu.com/ubuntu noble/main amd64 liberror-perl all 0.17029-2 [25.6 kB]
Get:3 http://ua.archive.ubuntu.com/ubuntu noble-updates/main amd64 git-man all 1:2.43.0-1ubuntu7.3 [1,100
kB]
Get:4 http://ua.archive.ubuntu.com/ubuntu noble-updates/main amd64 git amd64 1:2.43.0-1ubuntu7.3 [3,680 kB
]
Get:5 https://download.docker.com/linux/ubuntu noble/stable amd64 containerd.io amd64 1.7.28-1-ubuntu.24.0
4~noble [31.9 MB]
Get:6 http://ua.archive.ubuntu.com/ubuntu noble/main amd64 libslirp0 amd64 4.7.0-1ubuntu3 [63.8 kB]
Get:7 http://ua.archive.ubuntu.com/ubuntu noble/universe amd64 slirp4netns amd64 1.2.1-1build2 [34.9 kB]
Get:8 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-cli amd64 5:28.5.1-1-ubuntu.24
.04~noble [16.5 MB]
Get:9 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce amd64 5:28.5.1-1-ubuntu.24.04~
noble [19.7 MB]
Get:10 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-buildx-plugin amd64 0.29.1-1-ubu
ntu.24.04~noble [15.9 MB]
Get:11 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-ce-rootless-extras amd64 5:28.5.
1-1-ubuntu.24.04~noble [6,481 kB]
Get:12 https://download.docker.com/linux/ubuntu noble/stable amd64 docker-compose-plugin amd64 2.40.2-1-ub
untu.24.04~noble [14.3 MB]
Fetched 110 MB in 2s (44.3 MB/s)
Selecting previously unselected package containerd.io.
(Reading database ... 151036 files and directories currently installed.)
Preparing to unpack .../00-containerd.io_1.7.28-1-ubuntu.24.04~noble_amd64.deb ...
Unpacking containerd.io (1.7.28-1-ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../01-docker-ce-cli_5%3a28.5.1-1-ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce-cli (5:28.5.1-1-ubuntu.24.04~noble) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../02-docker-ce_5%3a28.5.1-1-ubuntu.24.04~noble_amd64.deb ...
Unpacking docker-ce (5:28.5.1-1-ubuntu.24.04~noble) ...
Progress: [ 10%] [#####.....]
```

Рис. 3.8 Встановлення пакунку Docker на Serv-Web-Server

```
Serv-Ubuntu-2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Oct 26 23:16
user@serv-ubuntu-2: ~

Preparing to unpack .../02-docker-ce_5%3a28.5.1-1-ubuntu.24.04-noble_amd64.deb ...
Unpacking docker-ce (5:28.5.1-1-ubuntu.24.04-noble) ...
Selecting previously unselected package pigz.
Preparing to unpack .../03-pigz_2.8-1_amd64.deb ...
Unpacking pigz (2.8-1) ...
Selecting previously unselected package docker-buildx-plugin.
Preparing to unpack .../04-docker-buildx-plugin_0.29.1-1-ubuntu.24.04-noble_amd64.deb ...
Unpacking docker-buildx-plugin (0.29.1-1-ubuntu.24.04-noble) ...
Selecting previously unselected package docker-ce-rootless-extras.
Preparing to unpack .../05-docker-ce-rootless-extras_5%3a28.5.1-1-ubuntu.24.04-noble_amd64.deb ...
Unpacking docker-ce-rootless-extras (5:28.5.1-1-ubuntu.24.04-noble) ...
Selecting previously unselected package docker-compose-plugin.
Preparing to unpack .../06-docker-compose-plugin_2.40.2-1-ubuntu.24.04-noble_amd64.deb ...
Unpacking docker-compose-plugin (2.40.2-1-ubuntu.24.04-noble) ...
Selecting previously unselected package liberror-perl.
Preparing to unpack .../07-liberror-perl_0.17029-2_all.deb ...
Unpacking liberror-perl (0.17029-2) ...
Selecting previously unselected package git-man.
Preparing to unpack .../08-git-man_1%3a2.43.0-1ubuntu7.3_all.deb ...
Unpacking git-man (1:2.43.0-1ubuntu7.3) ...
Selecting previously unselected package git.
Preparing to unpack .../09-git_1%3a2.43.0-1ubuntu7.3_amd64.deb ...
Unpacking git (1:2.43.0-1ubuntu7.3) ...
Selecting previously unselected package libslirp0:amd64.
Preparing to unpack .../10-libslirp0_4.7.0-1ubuntu3_amd64.deb ...
Unpacking libslirp0:amd64 (4.7.0-1ubuntu3) ...
Selecting previously unselected package slirp4netns.
Preparing to unpack .../11-slirp4netns_1.2.1-1build2_amd64.deb ...
Unpacking slirp4netns (1.2.1-1build2) ...
Setting up liberror-perl (0.17029-2) ...
Setting up docker-buildx-plugin (0.29.1-1-ubuntu.24.04-noble) ...
Setting up containerd.io (1.7.28-1-ubuntu.24.04-noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /usr/lib/systemd/system/containerd.service.
Setting up docker-compose-plugin (2.40.2-1-ubuntu.24.04-noble) ...
Setting up docker-ce-cli (5:28.5.1-1-ubuntu.24.04-noble) ...
Setting up libslirp0:amd64 (4.7.0-1ubuntu3) ...
Setting up pigz (2.8-1) ...
Setting up git-man (1:2.43.0-1ubuntu7.3) ...
Setting up docker-ce-rootless-extras (5:28.5.1-1-ubuntu.24.04-noble) ...
Setting up slirp4netns (1.2.1-1build2) ...
Setting up docker-ce (5:28.5.1-1-ubuntu.24.04-noble) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /usr/lib/systemd/system/docker.socket.
Setting up git (1:2.43.0-1ubuntu7.3) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.6) ...
user@serv-ubuntu-2:~$
```

Рис. 3.9 Встановлення пакунку Docker на Serv-Web-Server

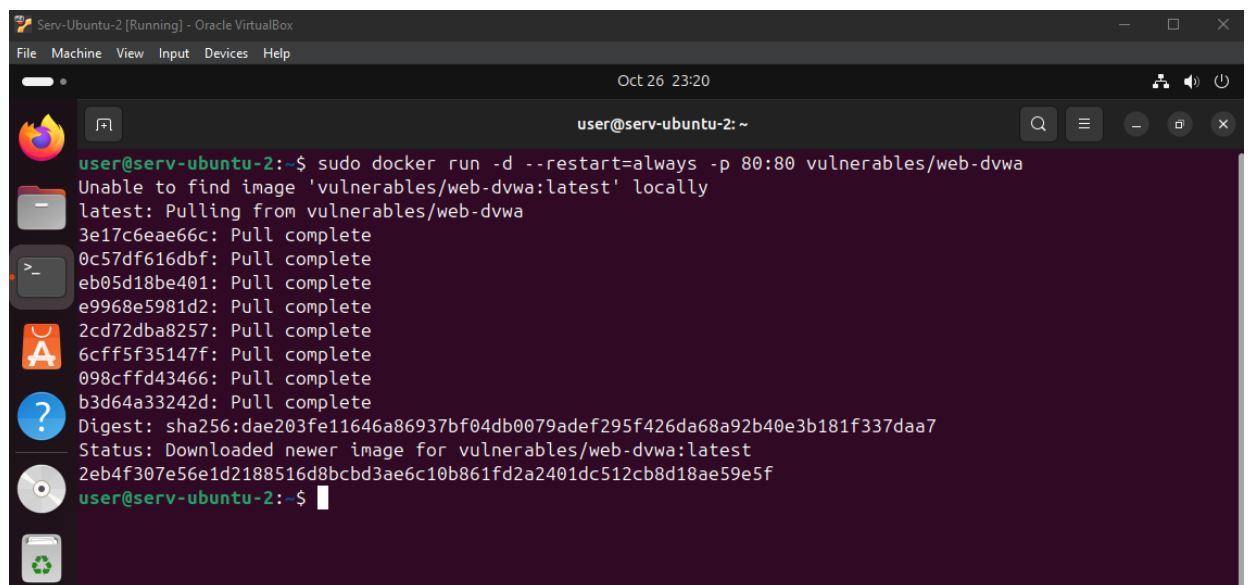


```
user@serv-ubuntu-2:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-10-26 23:14:01 UTC; 3min 6s ago
     TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
      Main PID: 5816 (dockerd)
         Tasks: 9
        Memory: 21.2M (peak: 21.8M)
           CPU: 797ms
      CGroup: /system.slice/docker.service
              └─5816 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 26 23:13:59 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:13:59.795675269Z" level=info msg="detect>
Oct 26 23:13:59 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:13:59.840622457Z" level=info msg="Creati>
Oct 26 23:14:00 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:00.355592246Z" level=info msg="Loadin>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.272599627Z" level=info msg="Loadin>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.571529193Z" level=info msg="Docker>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.572118920Z" level=info msg="Initia>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.648791428Z" level=info msg="Comple>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.653826613Z" level=info msg="Daemon>
Oct 26 23:14:01 serv-ubuntu-2 dockerd[5816]: time="2025-10-26T23:14:01.654151865Z" level=info msg="API li>
Oct 26 23:14:01 serv-ubuntu-2 systemd[1]: Started docker.service - Docker Application Container Engine.

lines 1-22/22 (END)
```

Рис. 3.10 Перевірка статусу служби Docker



```
user@serv-ubuntu-2:~$ sudo docker run -d --restart=always -p 80:80 vulnerables/web-dvwa
Unable to find image 'vulnerables/web-dvwa:latest' locally
latest: Pulling from vulnerables/web-dvwa
3e17c6eae66c: Pull complete
0c57df616dbf: Pull complete
eb05d18be401: Pull complete
e9968e5981d2: Pull complete
2cd72dba8257: Pull complete
6cff5f35147f: Pull complete
098cffd43466: Pull complete
b3d64a33242d: Pull complete
Digest: sha256:dae203fe11646a86937bf04db0079adef295f426da68a92b40e3b181f337daa7
Status: Downloaded newer image for vulnerables/web-dvwa:latest
2eb4f307e56e1d2188516d8bcd3ae6c10b861fd2a2401dc512cb8d18ae59e5f
user@serv-ubuntu-2:~$
```

Рис. 3.11 Встановлення веб-додатку у Docker

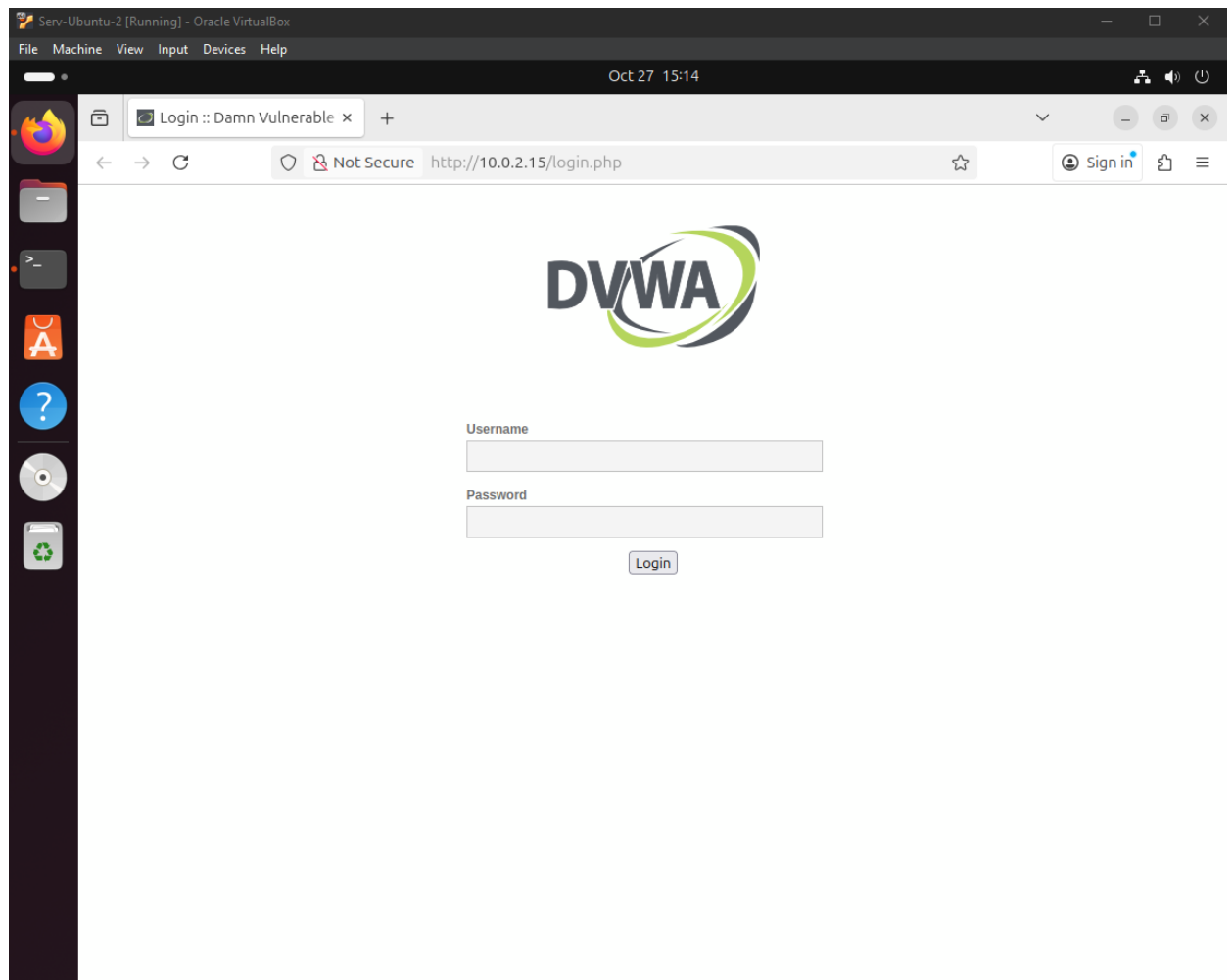


Рис. 3.12 Перевірка встановленого веб-додатку

Інтеграція клієнтського сегмента мережі на базі операційної системи Windows здійснювалася з урахуванням архітектурних особливостей пропрієтарного середовища. Для інсталяції використовувався спеціалізований пакет у форматі MSI, який автоматизує створення системних служб та налаштування правил локального брандмауера для дозволу вихідних з'єднань. Процес налаштування передбачав введення IP-адреси сервера Wazuh через графічний інтерфейс інсталятора, що ініціювало підключення до менеджера через порт 1514. Успішний запуск агента у фоновому режимі дозволив системі отримати доступ до специфічних джерел даних Windows, зокрема журналів подій Security та System, забезпечуючи уніфікацію форматів логів від різномірних вузлів в єдиній консолі управління.

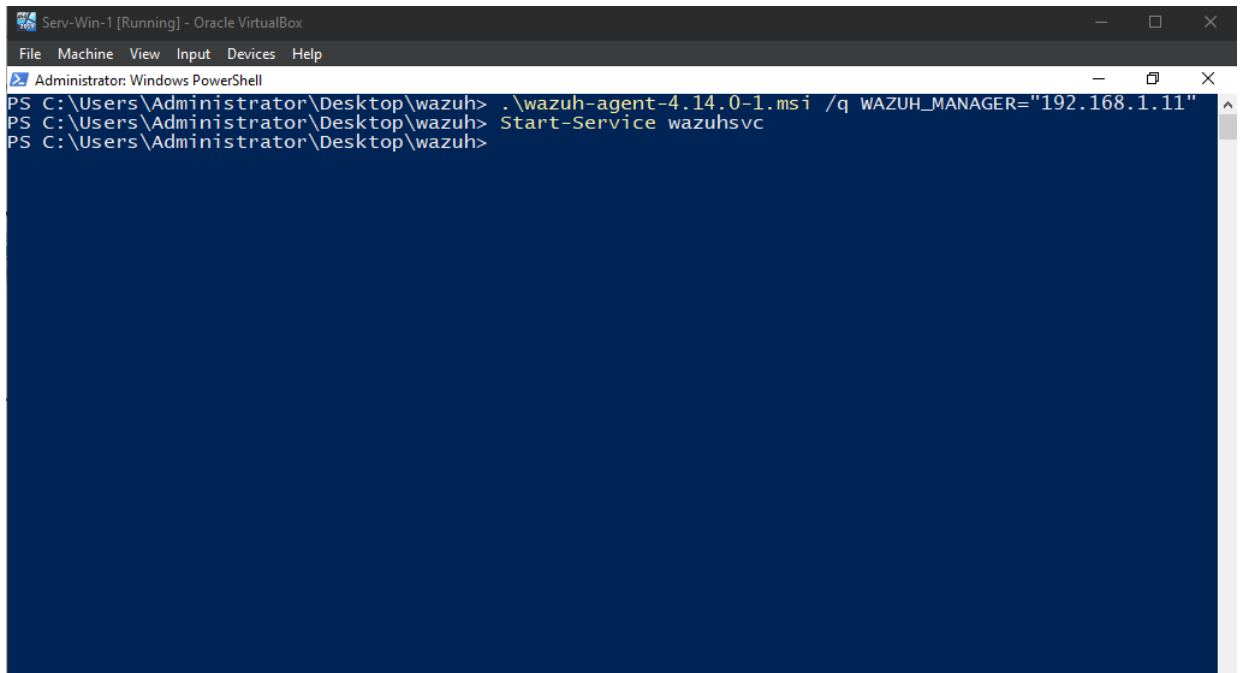


Рис. 3.13 Встановлення агенту Wazuh на Serv-Windows

Верифікація працездатності розгорнутої мережі агентів проводилася через веб-інтерфейс системи, де статус обох підключених вузлів змінився на активний, підтверджуючи стабільність каналів зв'язку. Для робочої станції під управлінням Windows було додатково проведено тонке налаштування модуля контролю цілісності файлів, розширивши область моніторингу на користувацькі директорії, які найчастіше стають об'єктом атак із використанням шкідливого програмного забезпечення. Ця модифікація конфігурації є підготовчим етапом до проведення експериментів із виявленням аномалій у файловій системі, гарантуючи фіксацію змін атрибутів файлів у реальному часі.

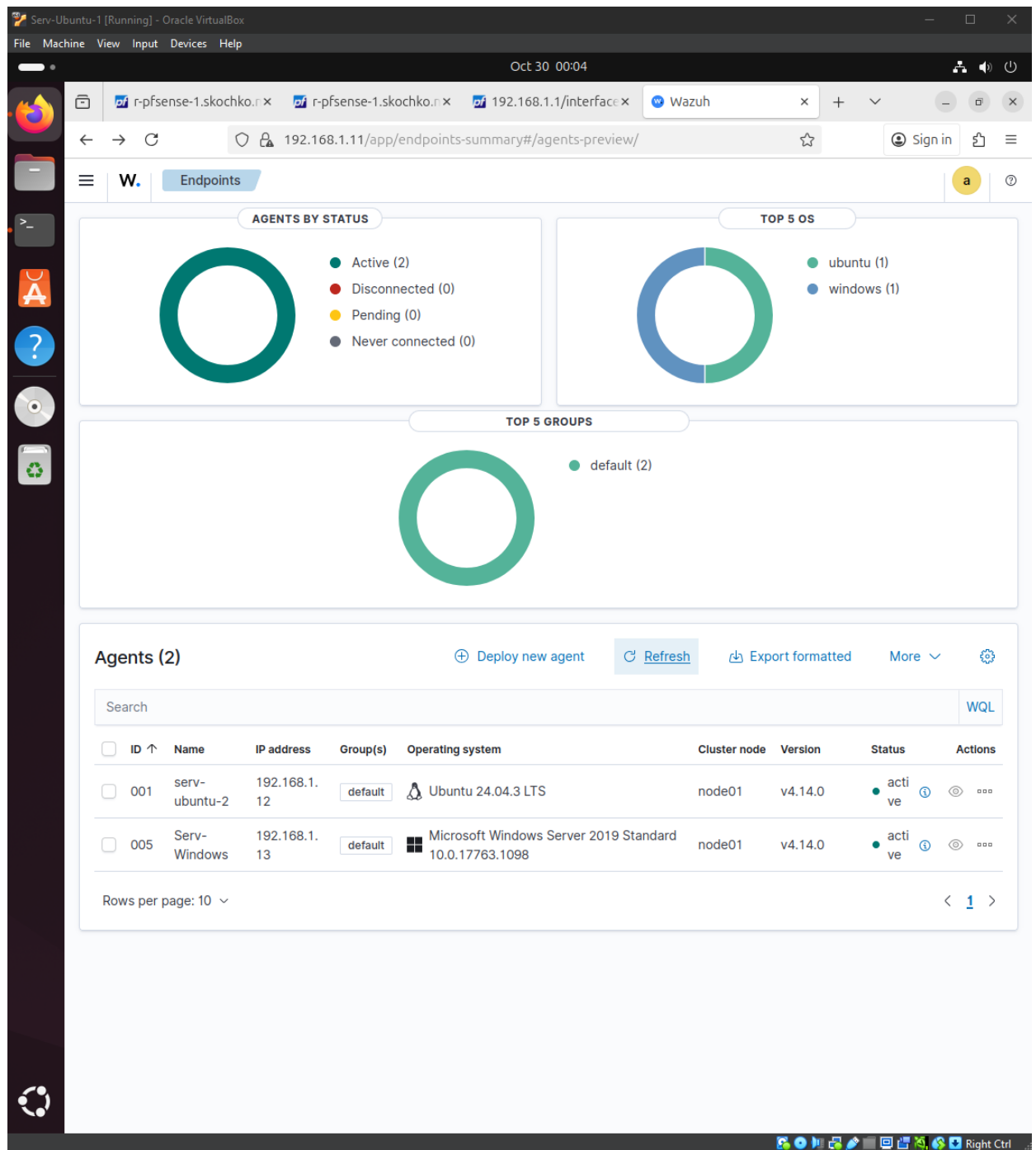
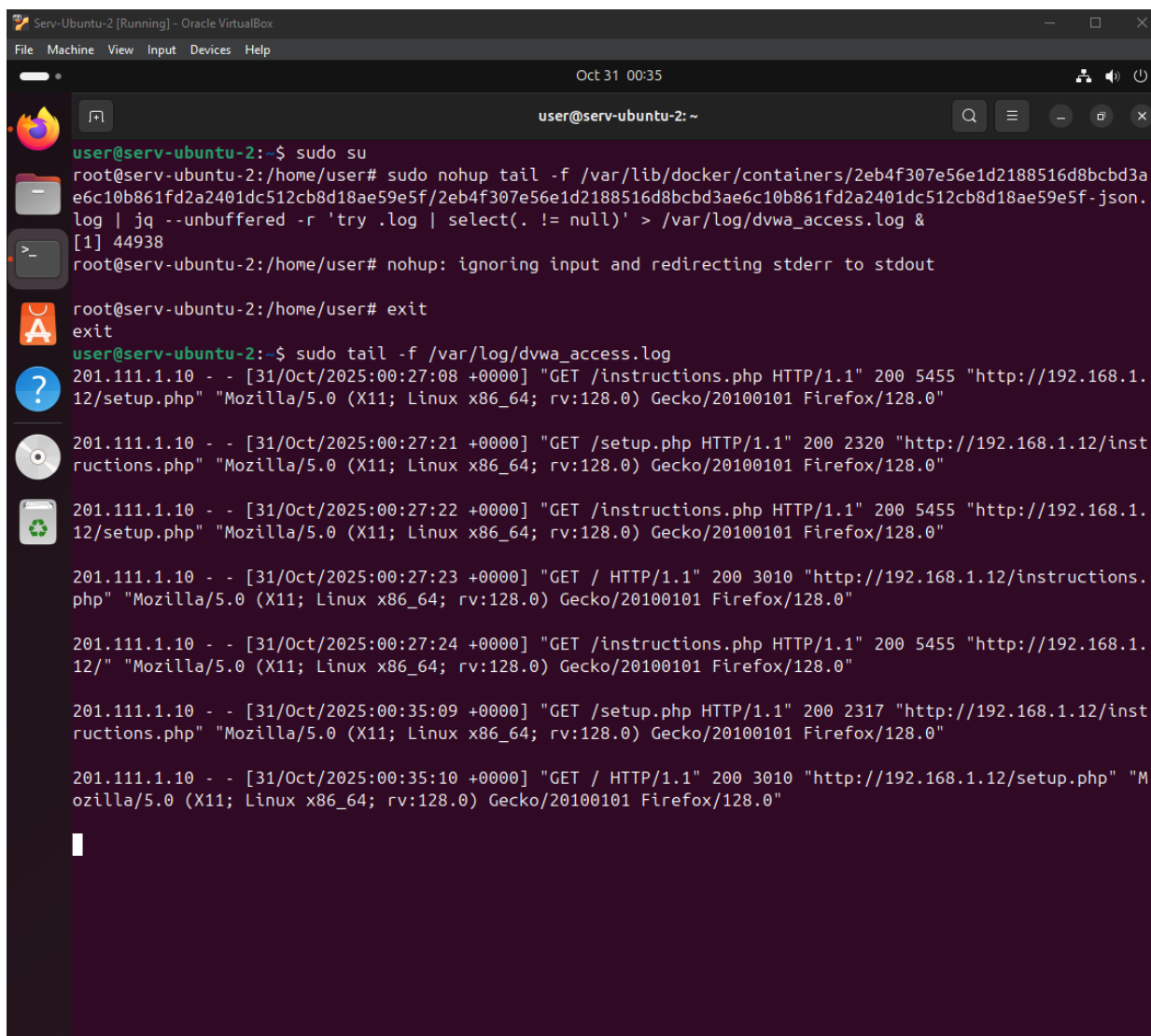


Рис. 3.14 Перевірка агентів

Моніторинг веб-застосунку DVWA вимагав подолання архітектурного обмеження середовища Docker, пов'язаного зі специфікою інкапсуляції потоків логування. Стандартна конфігурація контейнера передбачає виведення системних журналів у форматі JSON, що робить безпосереднє зчитування запитів Apache агентом Wazuh неефективним через відсутність підтримки такого синтаксису вбудованими веб-декодерами. Для забезпечення читабельності вхідних подій розроблено процедуру демаршалізації даних у реальному часі.

Використання утиліти `jq` дозволило динамічно вилучати вміст поля `.log` із загального JSON-потoku, перетворюючи службові повідомлення контейнера на стандартні рядки доступу веб-сервера. Процес налаштовано у вигляді фонові служби, яка безперервно перенаправляє оброблену інформацію у локальний файл `/var/log/dvwa_access.log`, створюючи прозорий канал телеметрії для подальшого аналізу.



```
user@serv-ubuntu-2:~$ sudo su
root@serv-ubuntu-2:/home/user# sudo nohup tail -f /var/lib/docker/containers/2eb4f307e56e1d2188516d8bcd3ae6c10b861fd2a2401dc512cb8d18ae59e5f/2eb4f307e56e1d2188516d8bcd3ae6c10b861fd2a2401dc512cb8d18ae59e5f.json.log | jq --unbuffered -r 'try .log | select(. != null)' > /var/log/dvwa_access.log &
[1] 44938
root@serv-ubuntu-2:/home/user# nohup: ignoring input and redirecting stderr to stdout

root@serv-ubuntu-2:/home/user# exit
exit
user@serv-ubuntu-2:~$ sudo tail -f /var/log/dvwa_access.log
201.111.1.10 - - [31/Oct/2025:00:27:08 +0000] "GET /instructions.php HTTP/1.1" 200 5455 "http://192.168.1.12/setup.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:27:21 +0000] "GET /setup.php HTTP/1.1" 200 2320 "http://192.168.1.12/instructions.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:27:22 +0000] "GET /instructions.php HTTP/1.1" 200 5455 "http://192.168.1.12/setup.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:27:23 +0000] "GET / HTTP/1.1" 200 3010 "http://192.168.1.12/instructions.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:27:24 +0000] "GET /instructions.php HTTP/1.1" 200 5455 "http://192.168.1.12/" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:35:09 +0000] "GET /setup.php HTTP/1.1" 200 2317 "http://192.168.1.12/instructions.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"

201.111.1.10 - - [31/Oct/2025:00:35:10 +0000] "GET / HTTP/1.1" 200 3010 "http://192.168.1.12/setup.php" "Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0"
```

Рис. 3.15 Перенаправлення журналу у локальний файл

Інтеграція створеного джерела даних у систему моніторингу реалізована шляхом модифікації локальної конфігурації агента на хості Ubuntu. У секції `<localfile>` конфігураційного файлу `ossec.conf` вказано шлях до новоствореного журналу, причому формат логів визначено як `syslog` для спрощення процесу нормалізації подій сервером. Такий підхід усунув необхідність у написанні складних кастомних декодерів на стороні менеджера, оскільки витягнуті рядки

```
Serv-Ubuntu-2 [Running] - Oracle VirtualBox
File Machine View Input Devices Help
Oct 31 00:37
user@serv-ubuntu-2: ~
GNU nano 7.2 /var/ossec/etc/ossec.conf
<log_format>full_command</log_format>
<command>netstat -tulpn | sed 's/\([[[:alnum:]]\+\\) \+[[[:digit:]]\+\\) \+[[[:digit:]]\+\\) \+(\(.*)\):\([[:d
```

Завершальний етап налаштування підсистеми збору веб-логів включав валідацію цілісності ланцюжка передачі даних. Через веб-інтерфейс управління було підтверджено коректність реєстрації агентом подій, що генеруються при

взаємодії з веб-додатком, при цьому дані відображаються у чистому текстовому вигляді без JSON-обгортки. Це дозволяє вбудованим правилам кореляції Wazuh безперешкодно зіставляти запити з відомими сигнатурами SQL-ін'єкцій. Реалізована методика підготовки даних демонструє спосіб уніфікації логів у гетерогенних середовищах, де додатки мають різні рівні абстракції та формати звітності, забезпечуючи повноту моніторингу контейнеризованої інфраструктури.

Забезпечення контролю за мережевим периметром реалізовано шляхом підключення програмного маршрутизатора pfSense до центральної системи моніторингу.



Рис. 3.17 Підготовка маршрутизатора, встановлення pfSense на маршрутизатор

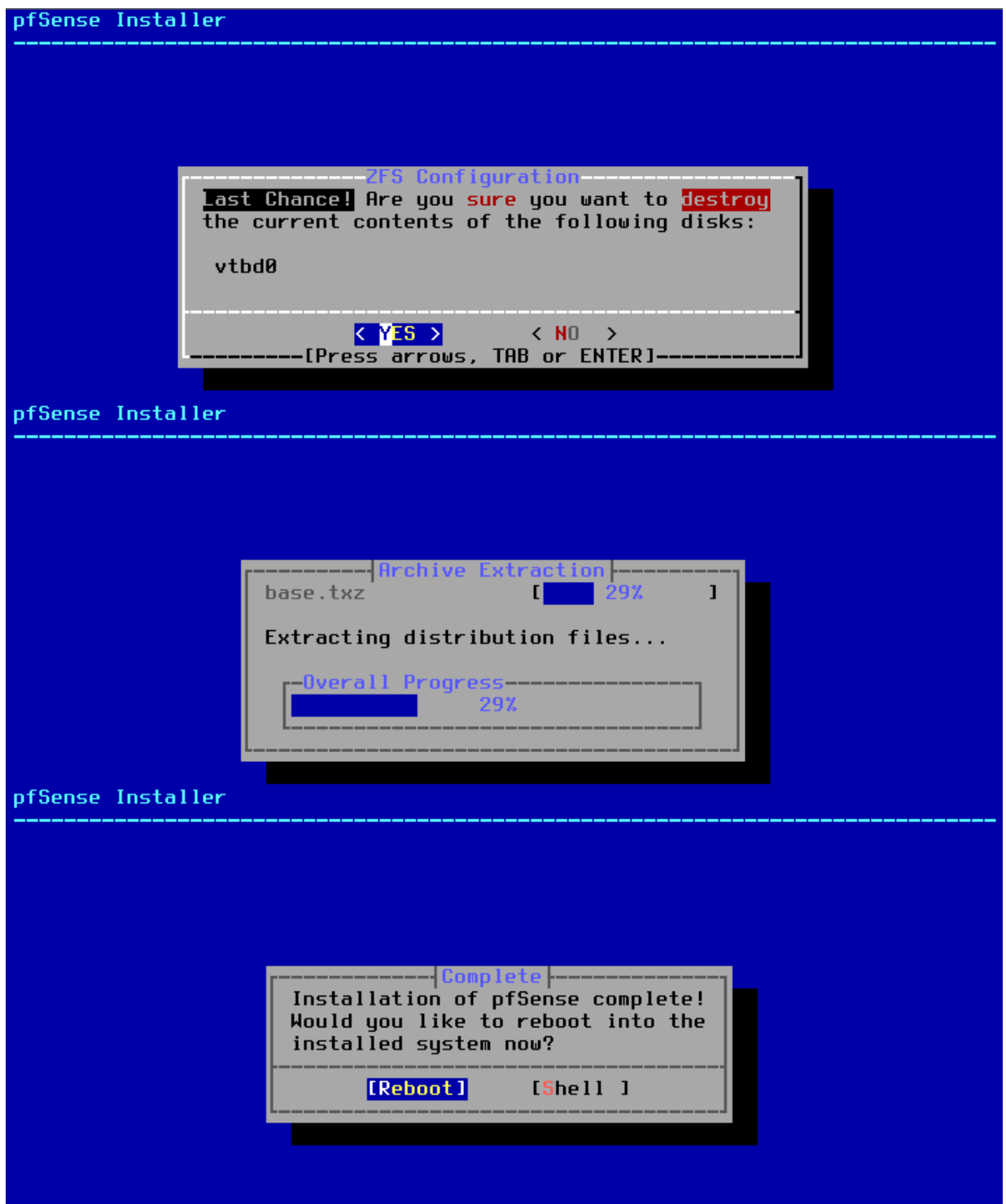


Рис. 3.18 Підготовка маршрутизатора, встановлення pfSense на маршрутизатор

Оскільки спеціалізована архітектура даної платформи не передбачає встановлення стандартних агентів Wazuh, обмін даними організовано з використанням протоколу Syslog, що є галузевим стандартом для мережевого обладнання. У налаштуваннях підсистеми логування шлюзу активовано механізм віддаленої передачі подій, де в ролі отримувача вказано IP-адресу сервера управління. Використання транспортного протоколу UDP на порту 514

забезпечило сумісність із попередньо налаштованим модулем прийому даних на стороні SIEM-системи, дозволяючи агрегувати інформацію про заблоковані пакети, встановлені з'єднання та системні помилки маршрутизатора в єдиному аналітичному просторі.

The screenshot displays the pfSense web interface for configuring logging. The browser address bar shows '192.168.1.1/status_logs_settings.php'. The page is divided into two main sections: 'General Logging Options' and 'Remote Logging Options'.

General Logging Options:

- Log Message Format:** A dropdown menu is set to 'BSD (RFC 3164, default)'. Below it, a note states: 'The format of syslog messages written to disk locally and sent to remote syslog servers (if enabled). Changing this value will only affect new log messages.'

Remote Logging Options:

- Enable Remote Logging:** A checkbox is checked, with the label 'Send log messages to remote syslog server'.
- Source Address:** A dropdown menu is set to 'Default (any)'. Below it, a note states: 'This option will allow the logging daemon to bind to a single IP address, rather than all IP addresses. If a single IP is picked, remote syslog servers must all be of that IP type. To mix IPv4 and IPv6 remote syslog servers, bind to all interfaces. NOTE: If an IP address cannot be located on the chosen interface, the daemon will bind to all addresses.'
- IP Protocol:** A dropdown menu is set to 'IPv4'. Below it, a note states: 'This option is only used when a non-default address is chosen as the source above. This option only expresses a preference; If an IP address of the selected type is not found on the chosen interface, the other type will be tried.'
- Remote log servers:** Three input fields are present. The first contains '192.168.1.11:514', the second contains 'IP[:port]', and the third contains 'IP[:port]'.
- Remote Syslog Contents:** A list of checkboxes for different event types:
 - ☐ Everything
 - ☒ System Events
 - ☒ Firewall Events
 - ☐ DNS Events (Resolver/unbound, Forwarder/dnsmasq, filterdns)
 - ☐ DHCP Events (DHCP Daemon, DHCP Relay, DHCP Client)
 - ☐ PPP Events (PPPoE WAN Client, L2TP WAN Client, PPTP WAN Client)
 - ☐ General Authentication Events
 - ☐ Captive Portal Events
 - ☐ VPN Events (IPsec, OpenVPN, L2TP, PPPoE Server)
 - ☐ Gateway Monitor Events
 - ☐ Routing Daemon Events (RADVD, UPnP, RIP, OSPF, BGP)
 - ☐ Network Time Protocol Events (NTP Daemon, NTP Client)
 - ☐ Wireless Events (hostapd)

At the bottom of the page, there is a blue 'Save' button.

Рис. 3.19 Налаштування відправки журналів за допомогою Syslog

Функціональність виявлення мережових атак розширено за допомогою інтеграції системи виявлення вторгнень Snort, інсталяцію якої здійснено через вбудований менеджер пакунків. Конфігурація сенсора передбачає моніторинг зовнішнього інтерфейсу WAN, що дозволяє аналізувати вхідний трафік у "сирому" вигляді до моменту застосування правил трансляції мережових адрес та фільтрації брандмауером. Таке розміщення точки аналізу надає можливість

фіксувати спроби сканування портів та експлуатації вразливостей ще на етапі розвідки зловмисником, забезпечуючи видимість загроз, які можуть бути автоматично відхилені правилами міжмережевого екрана, але залишаються важливими індикаторами компрометації.

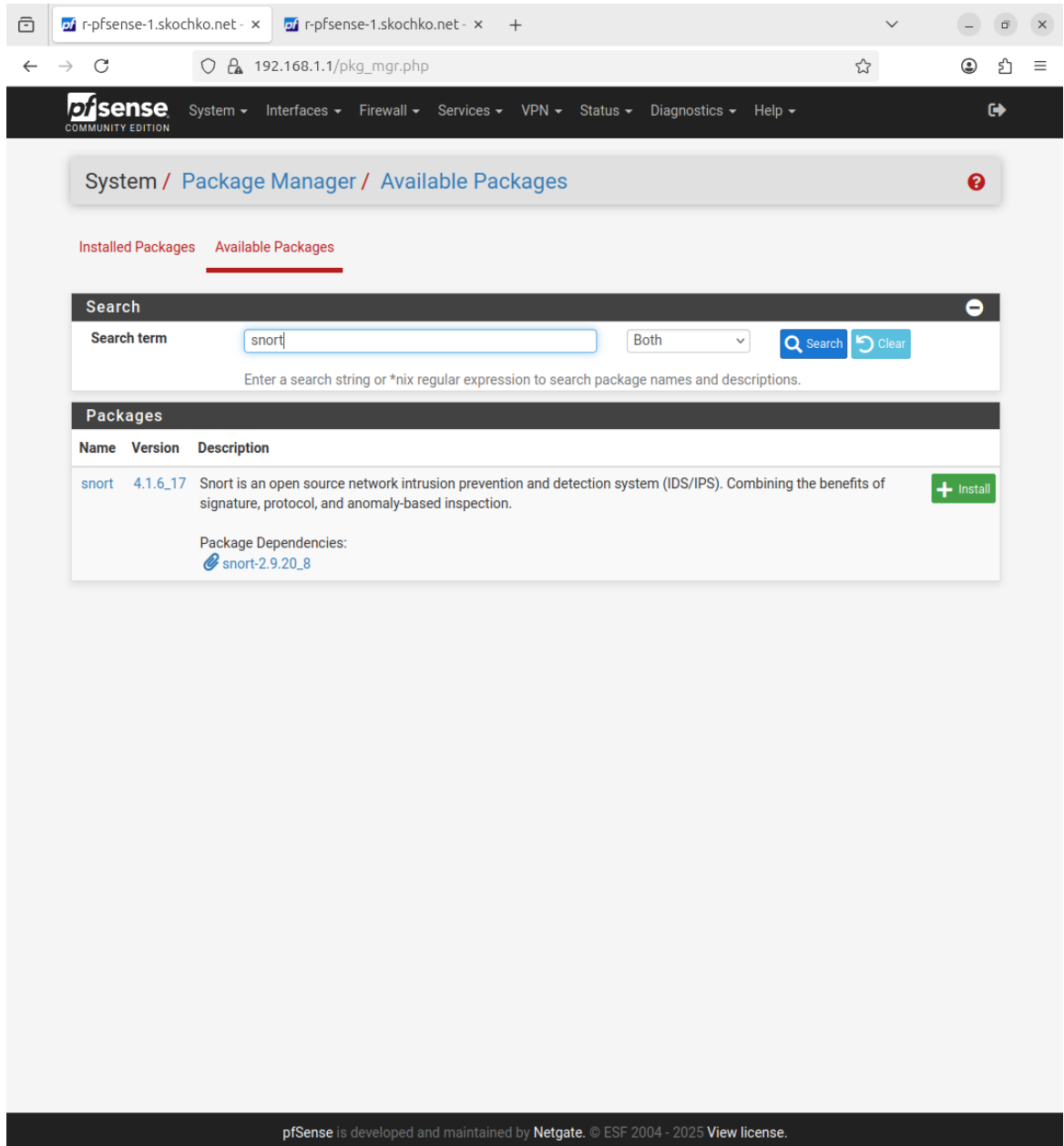


Рис. 3.20 Встановлення пакунту Snort

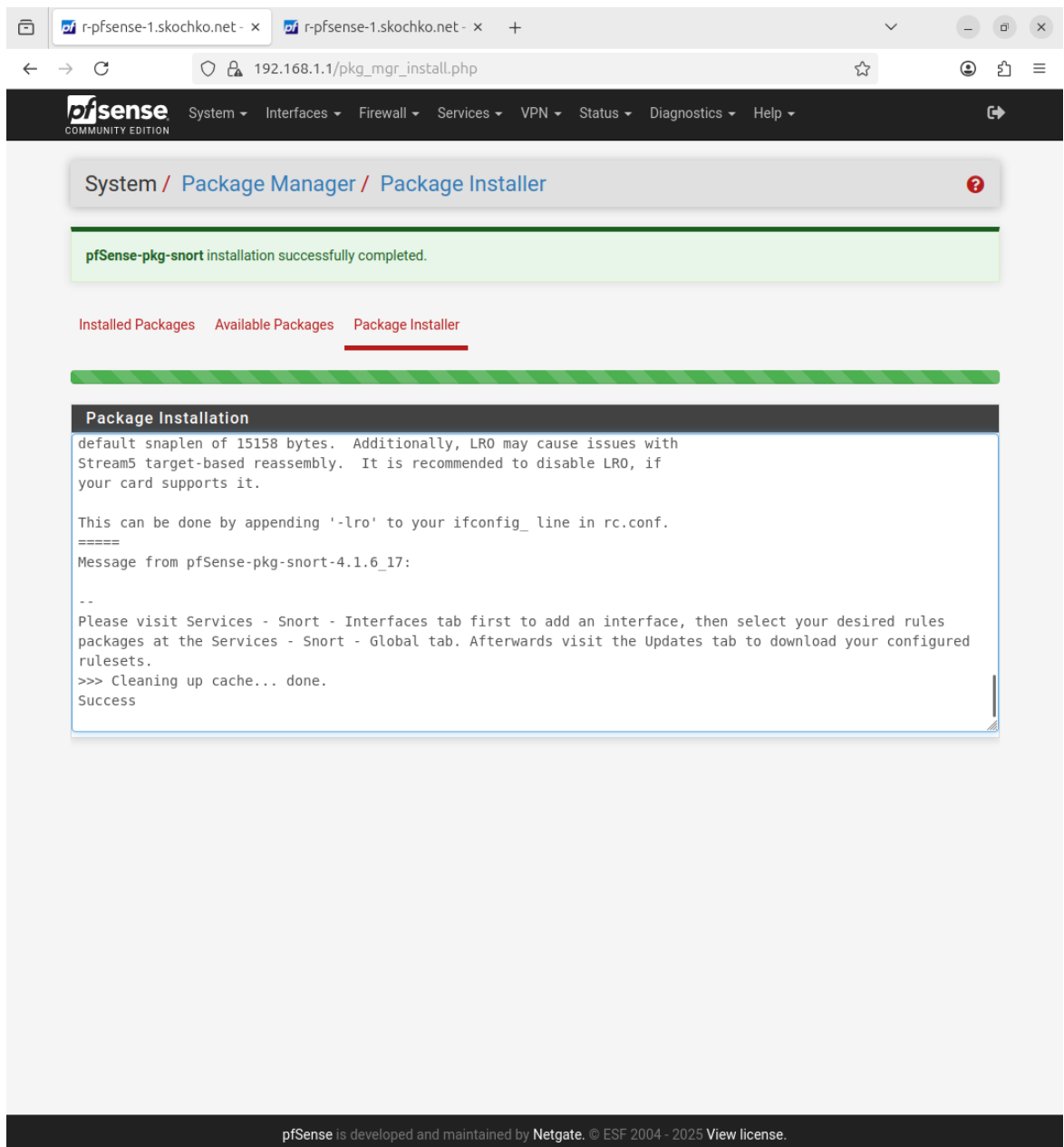


Рис. 3.21 Встановлення пакунку Snort

Сигнатурний аналіз трафіку базується на підключенні актуальних баз правил, що охоплюють широкий спектр веб-атак та аномалій протоколів. Критичним етапом налаштування взаємодії між компонентами системи стала активація опції перенаправлення тривожних сповіщень Snort у системний журнал маршрутизатора. Цей механізм створює логічний міст між NIDS та SIEM-системою, оскільки алерти про виявлені вторгнення інкапсулюються у стандартні повідомлення Syslog та автоматично транслуються на сервер Wazuh разом з іншими подіями шлюзу, дозволяючи корелювати мережеві інциденти з подіями на кінцевих точках без необхідності встановлення додаткових сенсорів.

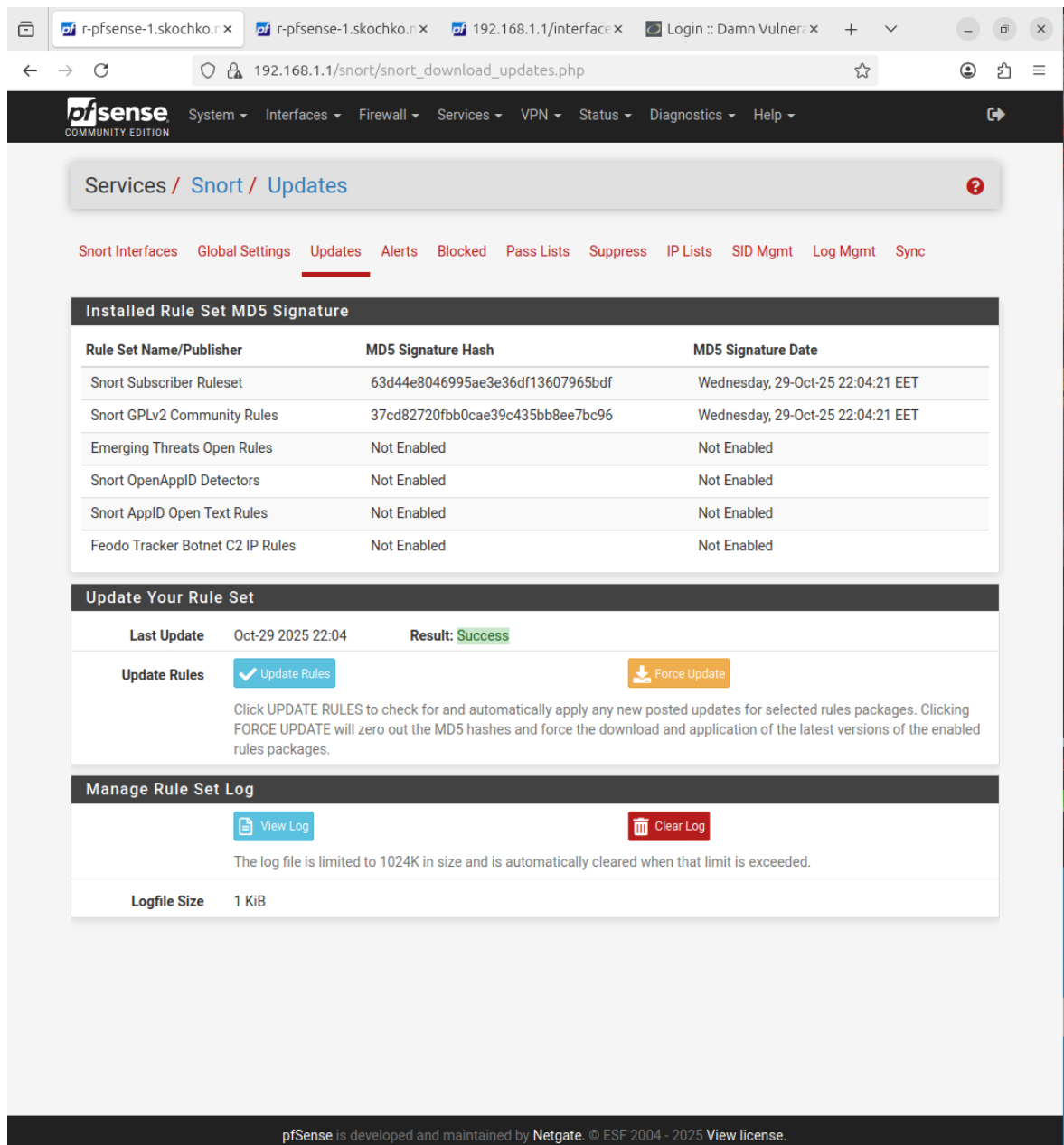
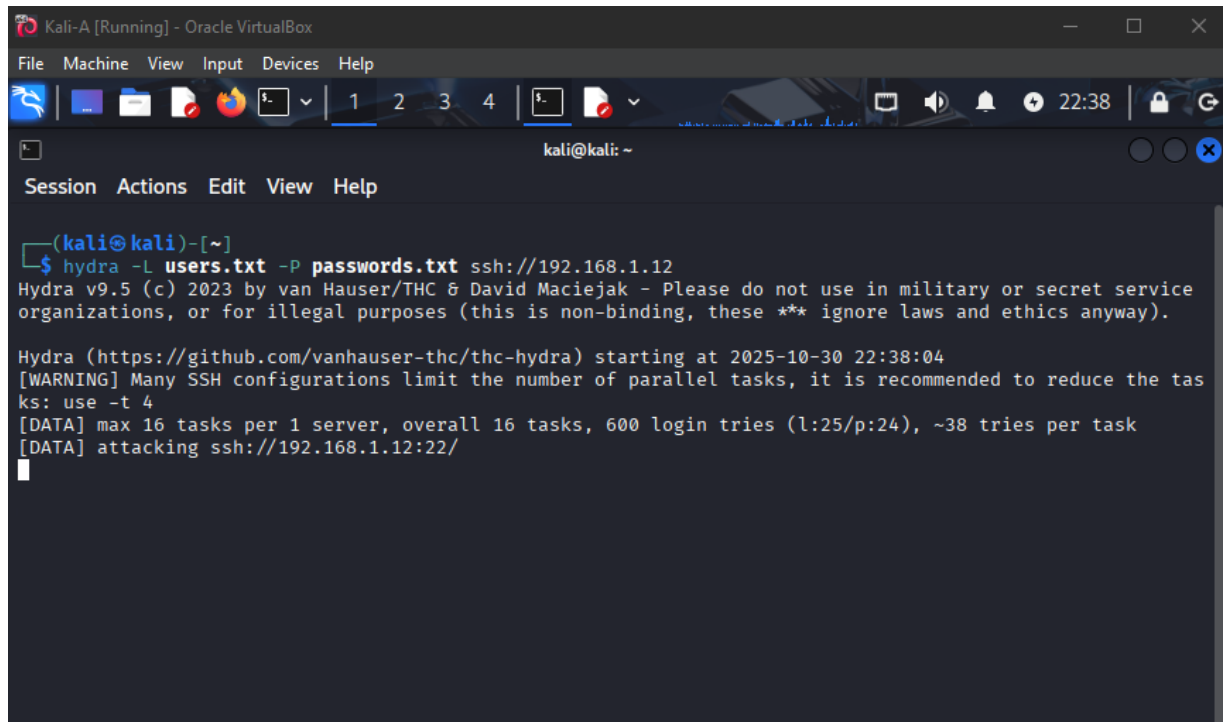


Рис. 3.22 Оновлення баз правил для виявлення загроз

3.2. Тестування системи шляхом моделювання кібератак

Експериментальна перевірка підсистеми виявлення аномалій розпочалася з ініціювання процесу автоматизованого підбору облікових даних до служби SSH внутрішнього веб-сервера. Атакуючий вузол згенерував інтенсивний потік запитів на з'єднання, використовуючи заздалегідь підготовлений словник типових комбінацій логінів та паролів. Моніторинг локальних системних журналів на цільовому хості підтвердив миттєву реєстрацію цих подій, де демон автентифікації зафіксував численні помилки входу із зовнішньої IP-адреси. Цей

етап підтвердив наявність мережевої зв'язності між ізольованими сегментами та доступність цільового сервісу для зовнішніх запитів, створивши необхідні умови для перевірки логіки детектування.



```
Kali-A [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4 5
kali@kali: ~
Session Actions Edit View Help

(kali@kali)-[~]
$ hydra -L users.txt -P passwords.txt ssh://192.168.1.12
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-30 22:38:04
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tas
ks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 600 login tries (l:25/p:24), ~38 tries per task
[DATA] attacking ssh://192.168.1.12:22/
```

Рис. 3.23 Виконання brute-force атаки на SSH серверу Serv-Web-Server

Паралельно з виконанням атаки консоль моніторингу Wazuh зареєструвала відповідний сплеск подій безпеки. Аналіз потоку сповіщень продемонстрував, що система успішно нормалізувала вхідні рядки логів, коректно ідентифікувавши IP-адресу джерела, атаковані облікові записи та специфіку помилки. Механізм кореляції активував стандартне правило з ідентифікатором 5710 на кожну окрему невдалу спробу, проте критично важливим стало спрацювання правила високого рівня з ідентифікатором 5712, класифікованого як спроба повного перебору паролів. Перехід від реєстрації одиничних подій до виявлення поведінкового патерну підтверджує ефективність частотного аналізу, закладеного в алгоритми SIEM-системи.

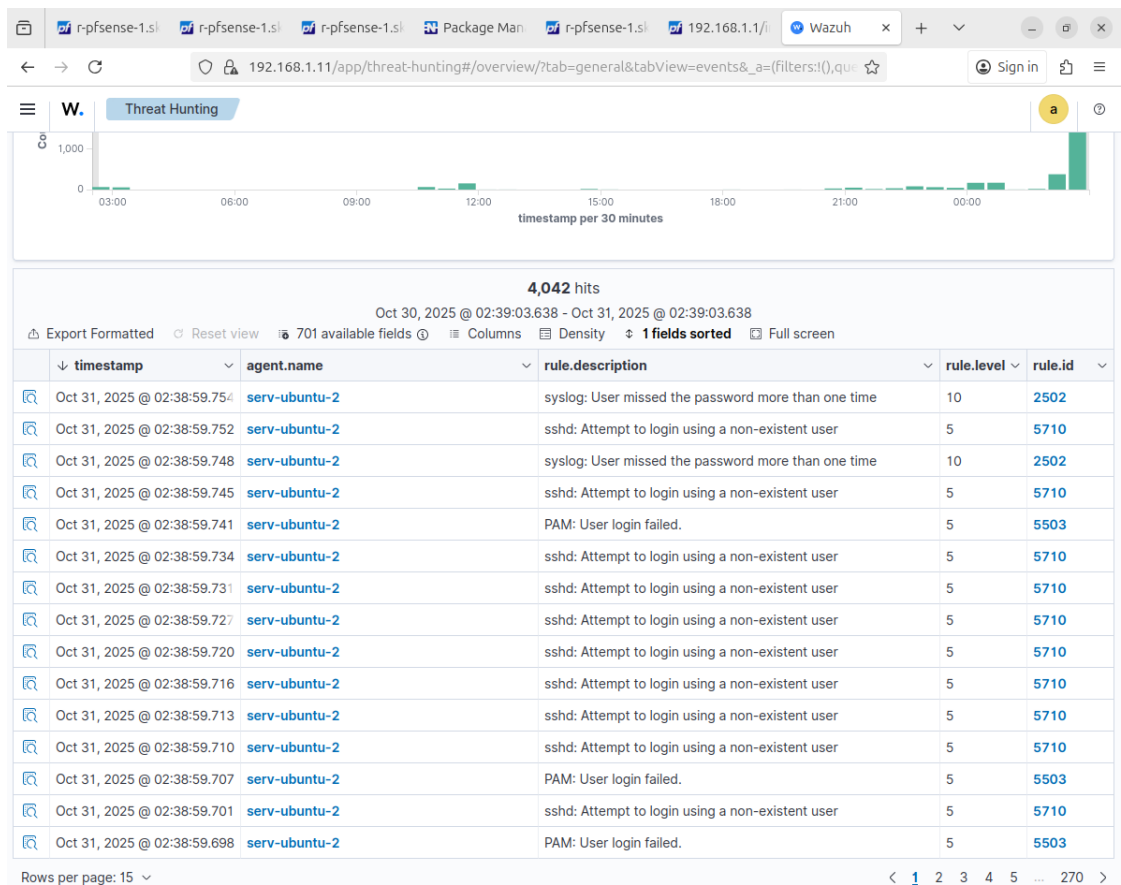


Рис. 3.24 Реакція Wazuh на загрозу

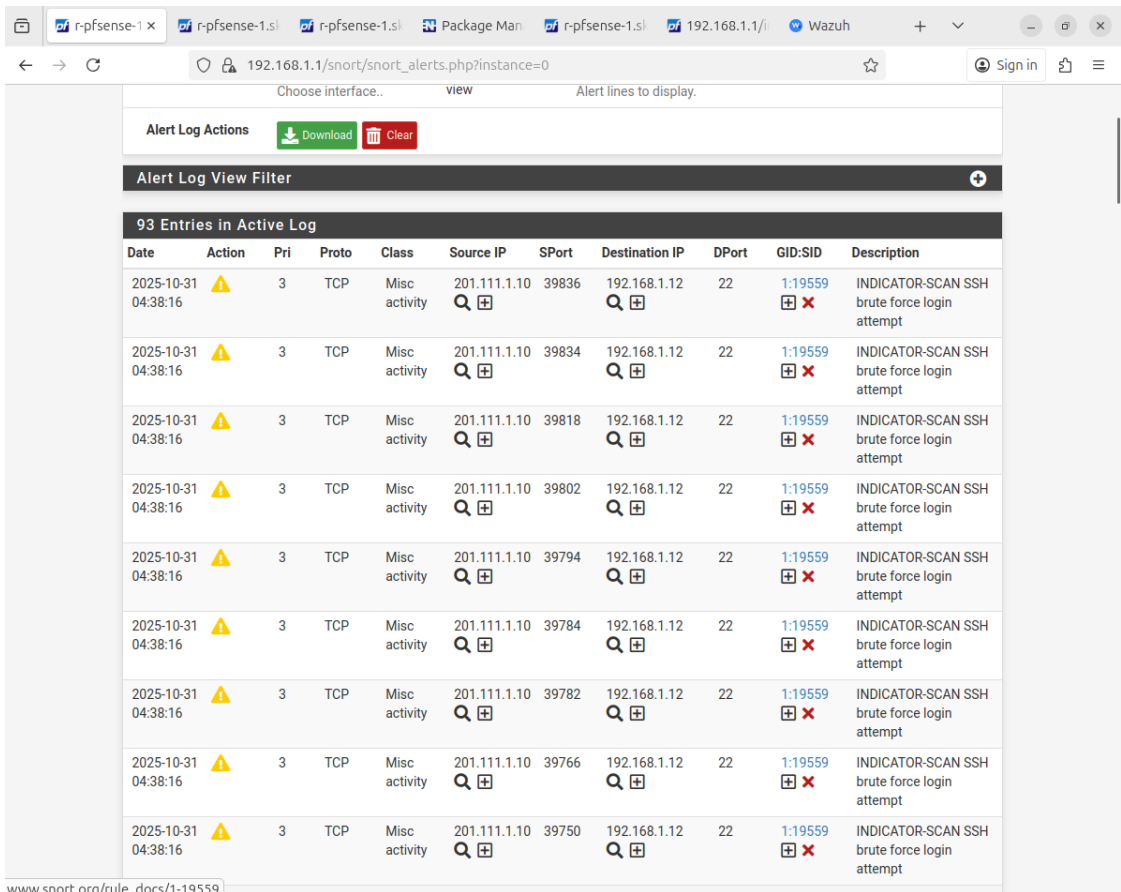


Рис. 3.25 Одночасна реакція Snort на загрозу

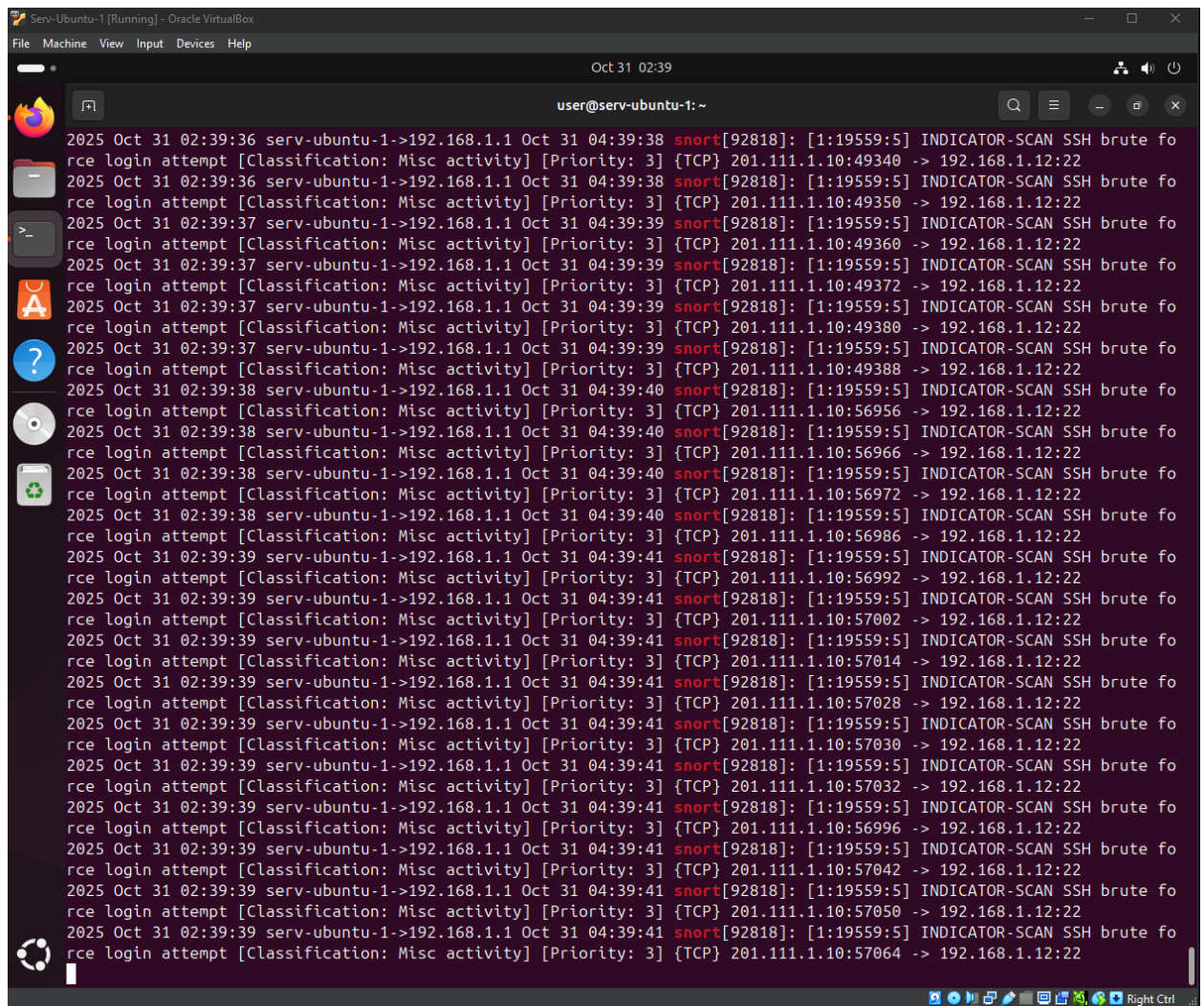


Рис. 3.26 Сповідання від Snort у журналі Wazuh

Візуалізація даних на інформаційній панелі відобразила чітку часову шкалу інциденту з різким зростанням кількості помилок автентифікації протягом короткого інтервалу. Агрегація отриманих подій дозволила точно встановити точку входу зловмисника та тривалість шкідливої активності. Результати експерименту засвідчили, що розгорнута архітектура «агент-сервер» забезпечує видимість атак типу brute-force у середовищі Linux у режимі, наближеному до реального часу, надаючи необхідну телеметрію для подальшого застосування механізмів активного реагування.

Верифікація механізмів захисту веб-додатків здійснювалася шляхом моделювання атаки на вразливий сервіс DVWA, функціонування якого забезпечується засобами контейнеризації Docker. Для генерації шкідливого трафіку було застосовано автоматизований інструмент SQLMap, налаштований

на використання методу HTTP GET для передачі параметрів ін'єкції безпосередньо в адресному рядку запиту. Генерація серії запитів, що містили оператори об'єднання таблиць та логічні умови, мала на меті імітувати спробу несанкціонованого отримання даних із бази даних, створюючи характерний сигнатурний слід у потоці подій.

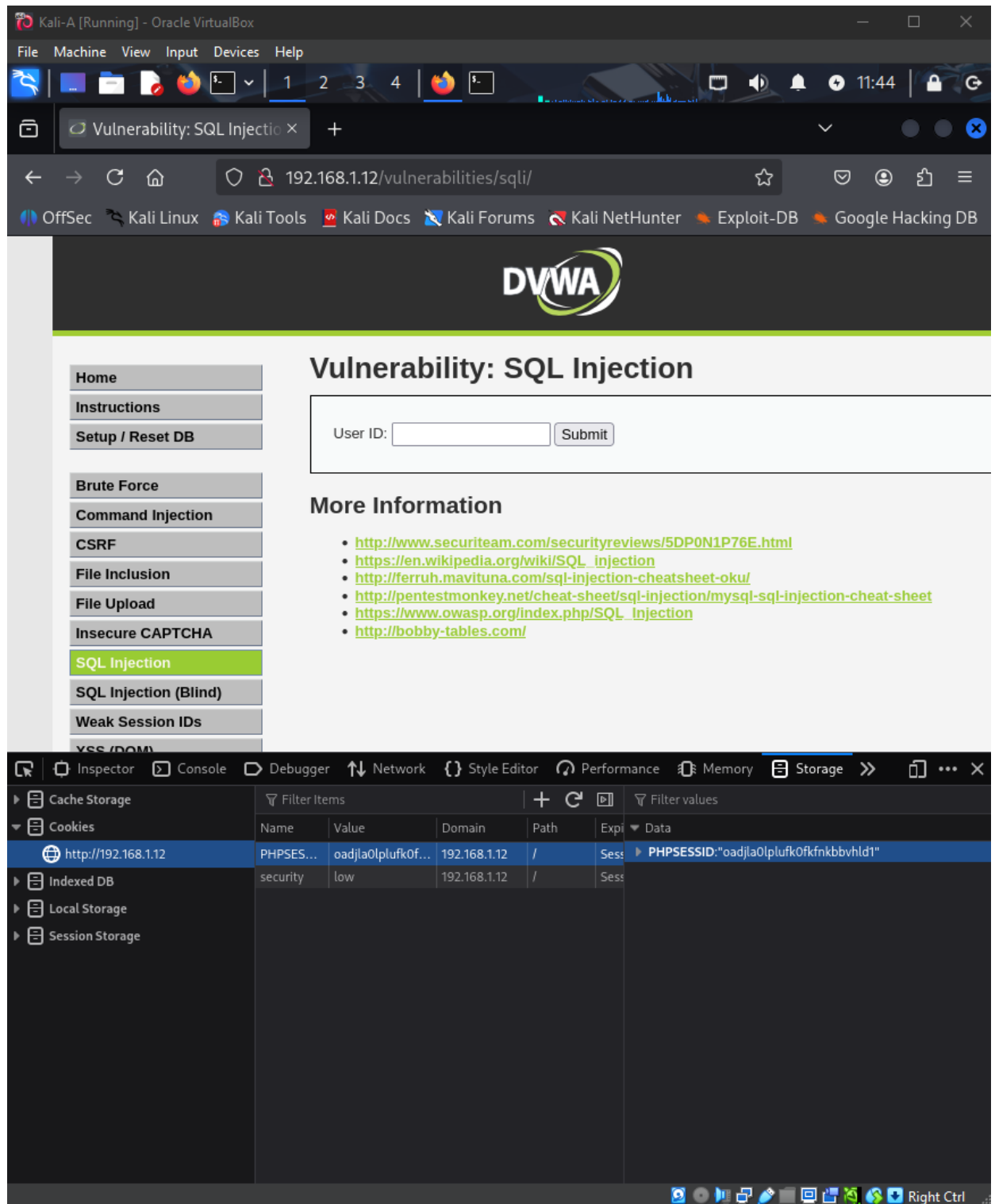
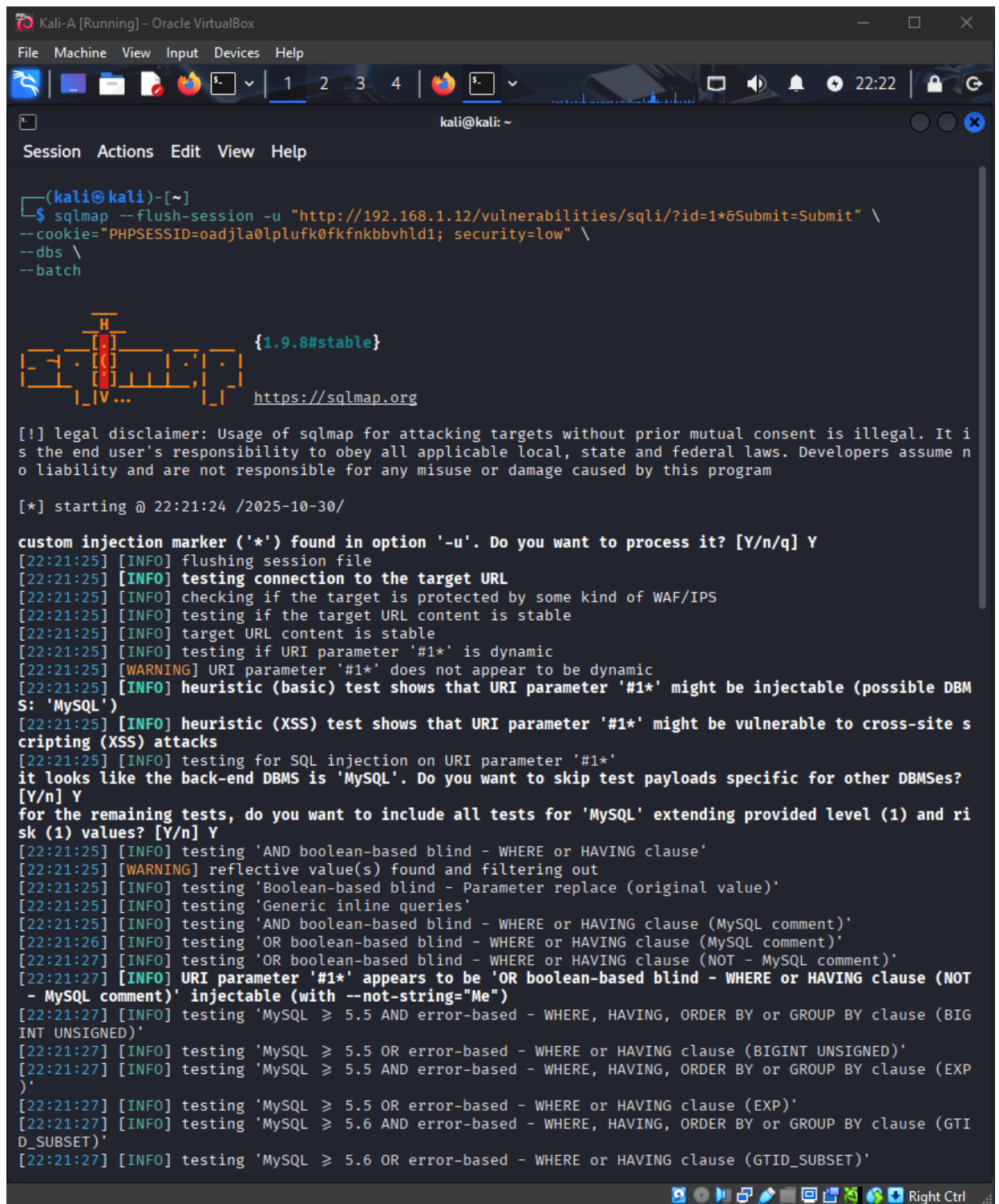


Рис. 3.27 Підготовка до атаки



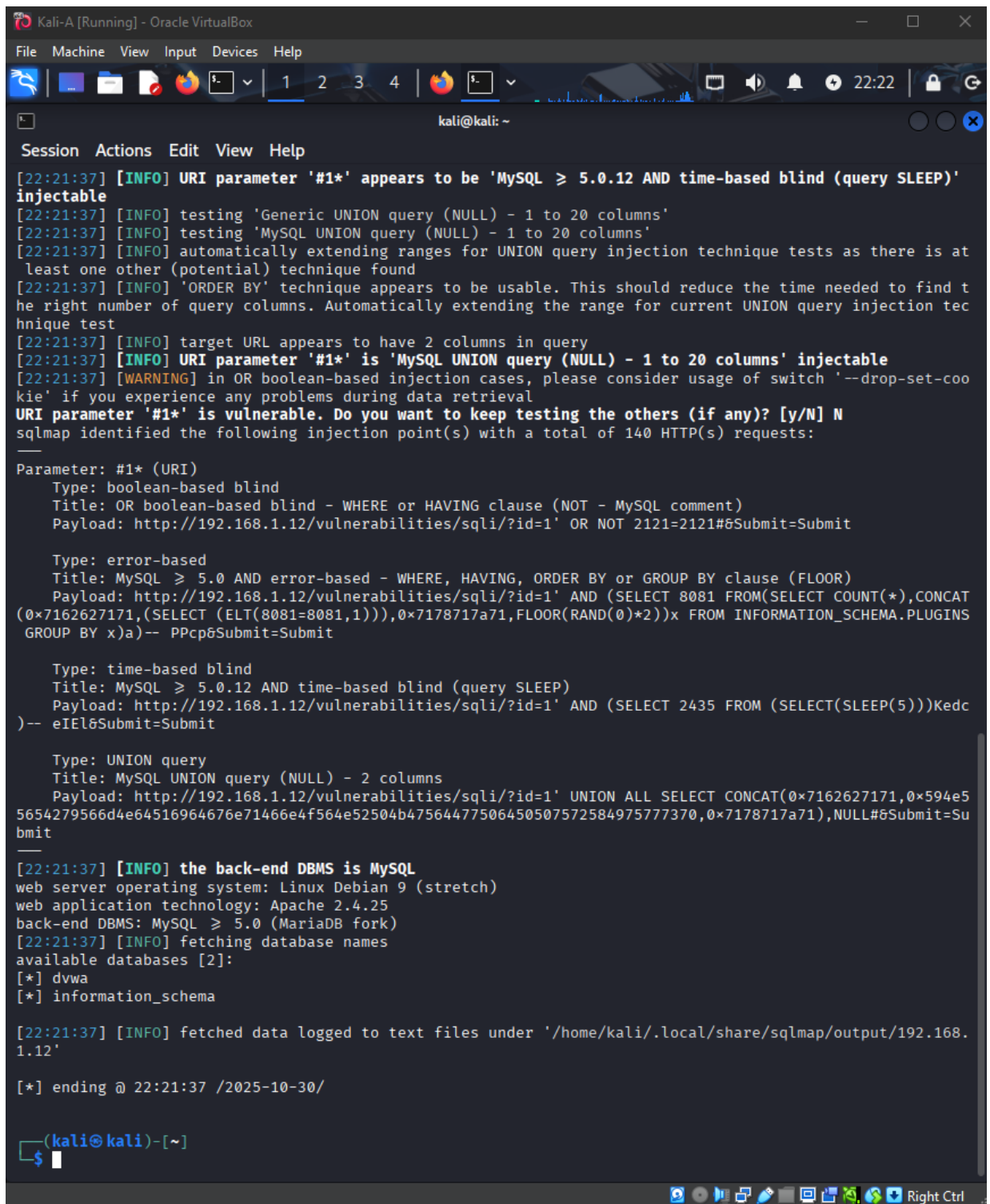
```
(kali@kali)-[~]
$ sqlmap --flush-session -u "http://192.168.1.12/vulnerabilities/sqli/?id=1*8Submit=Submit" \
--cookie="PHPSESSID=oadjla0lpufk0fkfnkbbvhld1; security=low" \
--dbs \
--batch

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is
the end user's responsibility to obey all applicable local, state and federal laws. Developers assume n
o liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 22:21:24 /2025-10-30/

custom injection marker ('*') found in option '-u'. Do you want to process it? [Y/n/q] Y
[22:21:25] [INFO] flushing session file
[22:21:25] [INFO] testing connection to the target URL
[22:21:25] [INFO] checking if the target is protected by some kind of WAF/IPS
[22:21:25] [INFO] testing if the target URL content is stable
[22:21:25] [INFO] target URL content is stable
[22:21:25] [INFO] testing if URI parameter '#1*' is dynamic
[22:21:25] [WARNING] URI parameter '#1*' does not appear to be dynamic
[22:21:25] [INFO] heuristic (basic) test shows that URI parameter '#1*' might be injectable (possible DBM
S: 'MySQL')
[22:21:25] [INFO] heuristic (XSS) test shows that URI parameter '#1*' might be vulnerable to cross-site s
cripting (XSS) attacks
[22:21:25] [INFO] testing for SQL injection on URI parameter '#1*'
it looks like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes?
[Y/n] Y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (1) and ri
sk (1) values? [Y/n] Y
[22:21:25] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[22:21:25] [WARNING] reflective value(s) found and filtering out
[22:21:25] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[22:21:25] [INFO] testing 'Generic inline queries'
[22:21:25] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[22:21:26] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (MySQL comment)'
[22:21:27] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)'
[22:21:27] [INFO] URI parameter '#1*' appears to be 'OR boolean-based blind - WHERE or HAVING clause (NOT
- MySQL comment)' injectable (with --not-string="Me")
[22:21:27] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIG
INT UNSIGNED)'
[22:21:27] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[22:21:27] [INFO] testing 'MySQL >= 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP
)'
[22:21:27] [INFO] testing 'MySQL >= 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[22:21:27] [INFO] testing 'MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTI
D_SUBSET)'
[22:21:27] [INFO] testing 'MySQL >= 5.6 OR error-based - WHERE or HAVING clause (GTID_SUBSET)'
```

Рис. 3.28 Виконання SQL-ін'єкції



```
Kali-A [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4
kali@kali: ~
Session Actions Edit View Help
[22:21:37] [INFO] URI parameter '#1*' appears to be 'MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)' injectable
[22:21:37] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[22:21:37] [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
[22:21:37] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[22:21:37] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[22:21:37] [INFO] target URL appears to have 2 columns in query
[22:21:37] [INFO] URI parameter '#1*' is 'MySQL UNION query (NULL) - 1 to 20 columns' injectable
[22:21:37] [WARNING] in OR boolean-based injection cases, please consider usage of switch '--drop-set-cookie' if you experience any problems during data retrieval
URI parameter '#1*' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 140 HTTP(s) requests:
Parameter: #1* (URI)
Type: boolean-based blind
Title: OR boolean-based blind - WHERE or HAVING clause (NOT - MySQL comment)
Payload: http://192.168.1.12/vulnerabilities/sqli/?id=1' OR NOT 2121=2121#&Submit=Submit

Type: error-based
Title: MySQL ≥ 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
Payload: http://192.168.1.12/vulnerabilities/sqli/?id=1' AND (SELECT 8081 FROM (SELECT COUNT(*), CONCAT(0x7162627171, (SELECT (ELT(8081=8081,1))), 0x7178717a71, FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)a)-- PPcp&Submit=Submit

Type: time-based blind
Title: MySQL ≥ 5.0.12 AND time-based blind (query SLEEP)
Payload: http://192.168.1.12/vulnerabilities/sqli/?id=1' AND (SELECT 2435 FROM (SELECT(SLEEP(5)))Kedc) -- eIEl&Submit=Submit

Type: UNION query
Title: MySQL UNION query (NULL) - 2 columns
Payload: http://192.168.1.12/vulnerabilities/sqli/?id=1' UNION ALL SELECT CONCAT(0x7162627171,0x594e55654279566d4e64516964676e71466e4f564e52504b47564477506450507572584975777370,0x7178717a71),NULL#&Submit=Submit

[22:21:37] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Debian 9 (stretch)
web application technology: Apache 2.4.25
back-end DBMS: MySQL ≥ 5.0 (MariaDB fork)
[22:21:37] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[22:21:37] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.1.12'

[*] ending @ 22:21:37 /2025-10-30/

(kali@kali)-[~]
$
```

Рис. 3.29 Результат атаки

Функціонування розробленого каналу збору телеметрії підтвердилося успішним перехопленням та перетворенням даних у реальному часі. Проміжний шар обробки, реалізований за допомогою утиліти для роботи з JSON-структурами, коректно виокремив змістовну частину логів веб-сервера Apache із загального потоку виводу контейнера. Перевірка на стороні агента засвідчила,

що попередньо очищені дані надходять у систему у форматі стандартних повідомлень Syslog, позбавлених надлишкових метаданих середовища віртуалізації. Це дозволило центральному серверу застосувати стандартні декодери веб-логів без необхідності розробки складних кастомних правил парсингу, підтверджуючи життєздатність обраної архітектури моніторингу ефемерних сервісів.

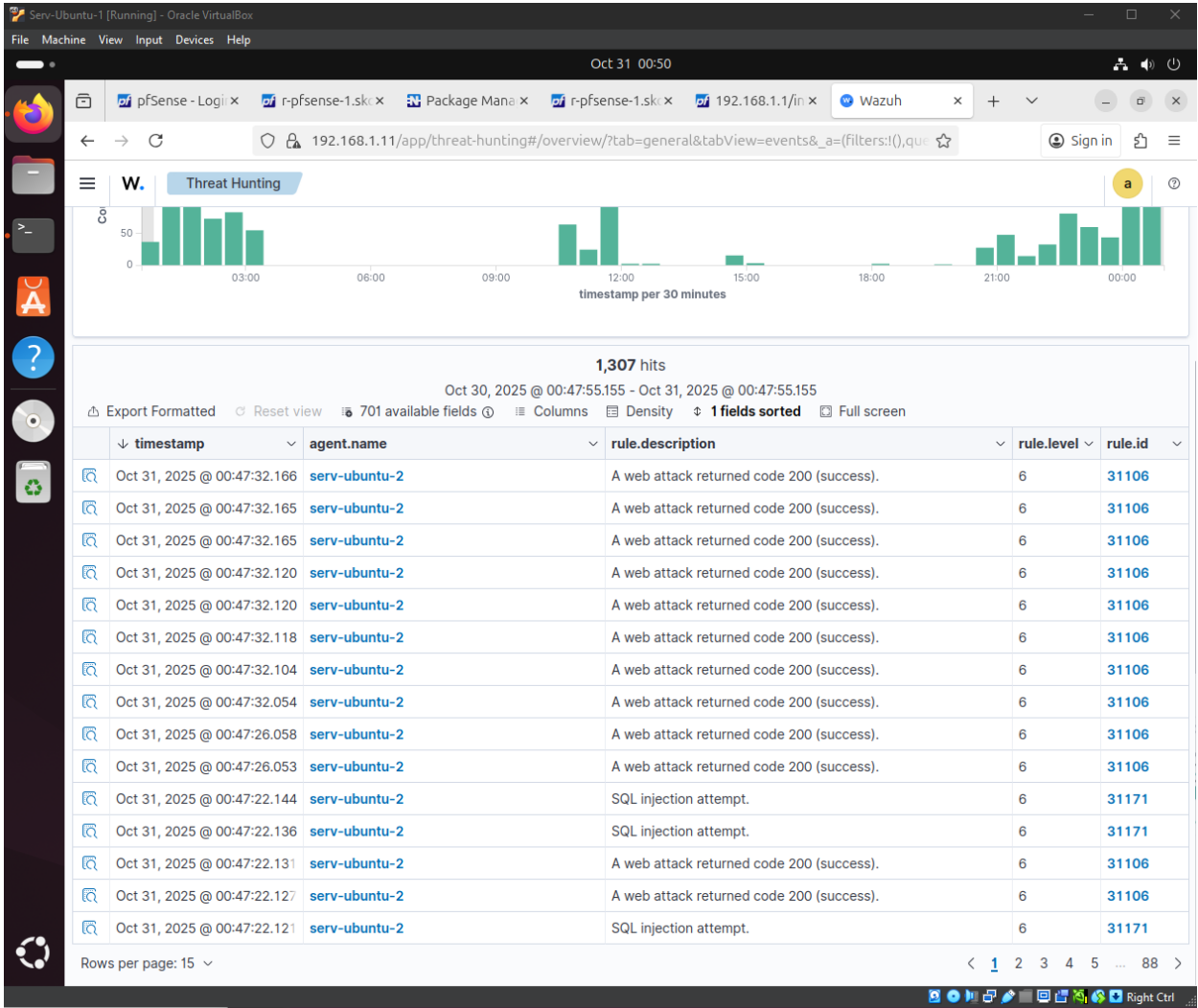


Рис. 3.30 Реакція Wazuh на атаку

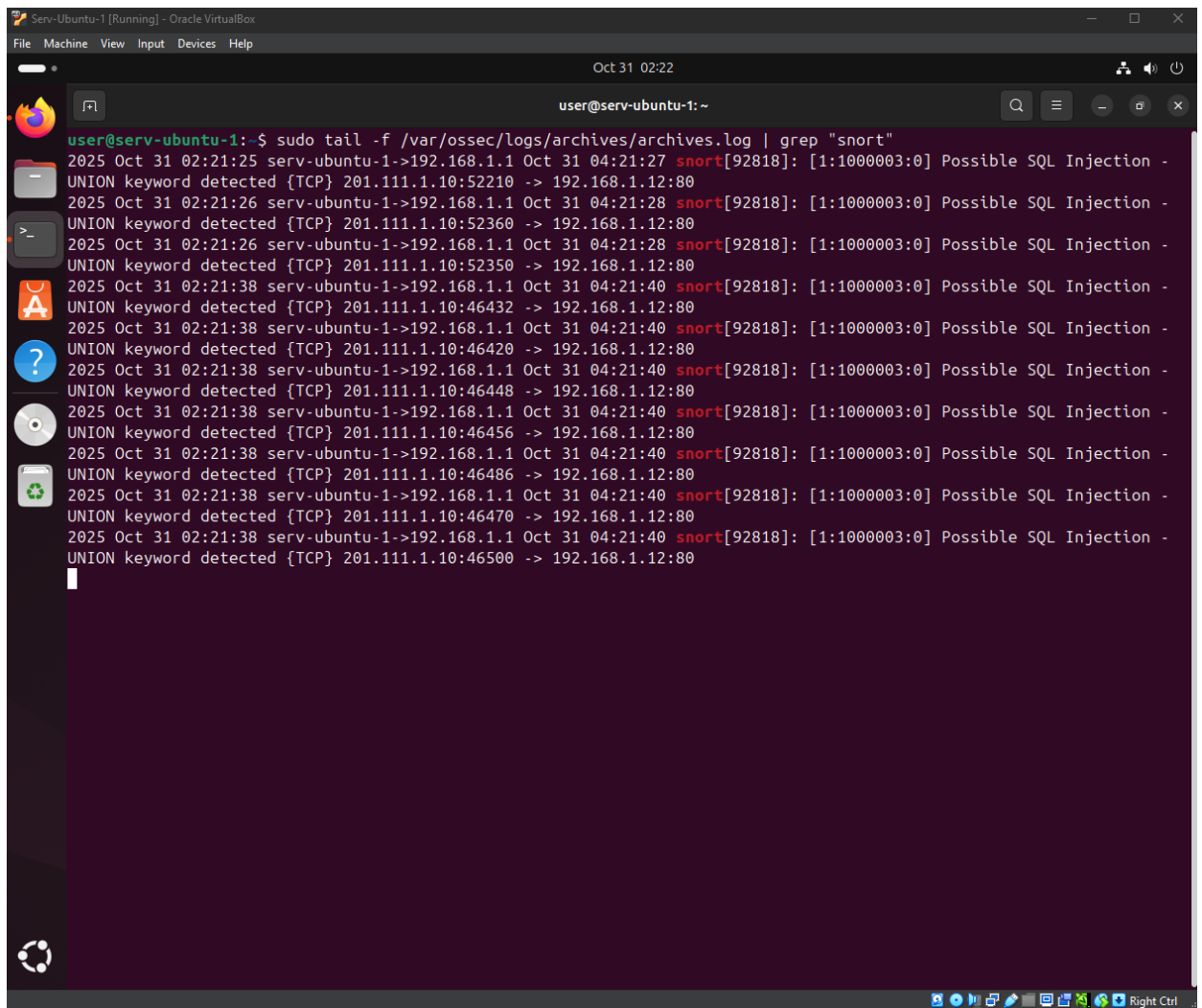


Рис. 3.31 Сповіщення від Snort про SQL-ін'єкцію у журналі Wazuh

Обробка нормалізованих журналів кореляційним рушієм призвела до реєстрації інцидентів високого рівня критичності в консолі управління. Система успішно зіставила виявлені патерни в URL-адресах із базою сигнатур веб-атак, активувавши правила, які класифікують подію як спробу SQL-ін'єкції. Деталізація отриманих сповіщень містила вичерпну інформацію про джерело атаки, цільовий ресурс та конкретний вектор ін'єкції, що надає аналітику безпеки необхідний контекст для розслідування. Результати тестування доводять, що інтеграція логів контейнеризованих додатків у загальний контур SIEM дозволяє ефективно виявляти вразливості зі списку OWASP Top 10 [23], нівелюючи ізоляційний бар'єр технології Docker.

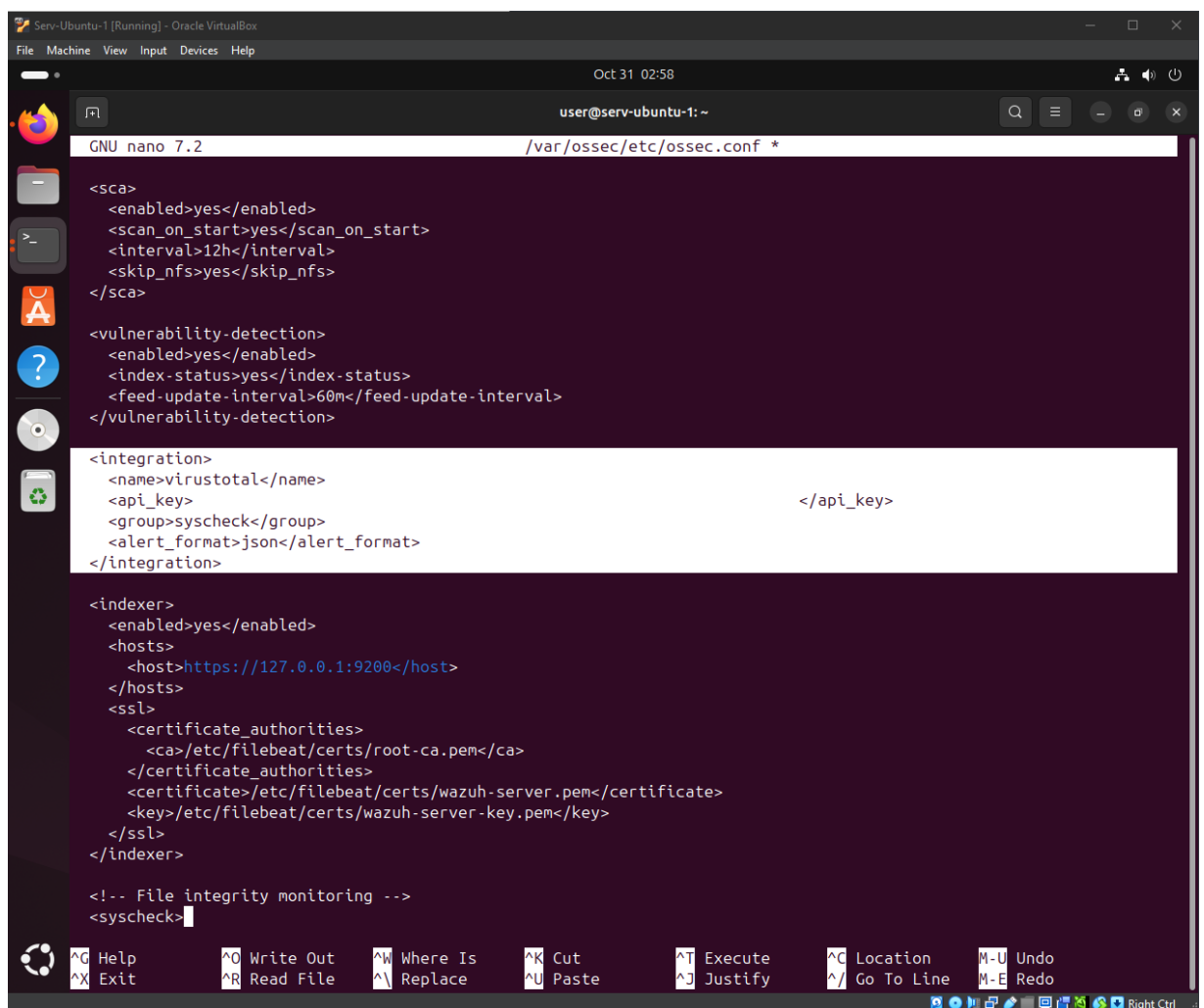
Комплексний підхід до моніторингу безпеки передбачає контроль не лише кінцевих точок, а й транзитного трафіку на межі захищеного периметра. Під час

моделювання веб-атак на сервер додатків весь вхідний потік даних проходив крізь маршрутизатор pfSense, де функціонувала система виявлення вторгнень Snort. Аналіз мережевих пакетів у реальному часі дозволив сенсору ідентифікувати ознаки шкідливої активності ще до моменту обробки запитів цільовим веб-сервером. Застосування баз сигнатур забезпечило детектування специфічних патернів SQL-ін'єкцій у тілі HTTP-запитів, що класифікувалися системою як спроби експлуатації вразливостей веб-додатків. Цей етап експерименту підтвердив здатність NIDS функціонувати як перша лінія оборони, фіксуючи факти розвідки та активних атак незалежно від стану захищеності внутрішніх хостів.

Інтеграція подій мережевого рівня в єдину консоль моніторингу відбулася завдяки налаштованому механізму трансляції системних повідомлень. Згенеровані Snort попередження автоматично пересилалися засобами служби Syslog на сервер Wazuh, де проходили процедуру декодування та нормалізації. Архітектура рішення дозволила обійти обмеження, пов'язані з відсутністю нативних агентів для спеціалізованих мережевих ОС, використавши універсальний транспортний протокол UDP. Сервер коректно розпізнав джерело подій як мережевий шлюз та зіставив отримані текстові повідомлення з відповідними правилами безпеки, сформувавши структуровані інциденти з деталізацією джерела та типу атаки.

Візуалізація результатів у веб-інтерфейсі продемонструвала кореляцію між мережевими алармами та подіями на веб-сервері, створюючи цілісну картину інциденту. Одночасна реєстрація атаки на рівні NIDS та на рівні прикладного програмного забезпечення дозволила верифікувати загрозу з високою точністю, мінімізуючи ймовірність хибних спрацювань. Наявність записів від Snort у базі даних SIEM надає аналітику додатковий контекст, зокрема інформацію про повний мережевий шлях атаки та використані протоколи, що є критично важливим для розслідування інцидентів у гетерогенних інфраструктурах, де не всі вузли можуть підтримувати встановлення агентського програмного забезпечення.

Заключний етап верифікації системи моніторингу був спрямований на оцінку захищеності клієнтських робочих станцій під управлінням операційної системи Windows, які традиційно є найбільш вразливим елементом корпоративної мережі. Сценарій експерименту передбачав імітацію завантаження шкідливого програмного забезпечення з зовнішнього ресурсу, для чого було використано штатний системний інструмент PowerShell. Виконання команди на завантаження тестового файлу EICAR з веб-сервера зловмисника дозволило відтворити типовий вектор зараження, коли шкідливий об'єкт потрапляє у файлову систему через дії користувача або автоматизовані скрипти. Цей процес відбувався у директорії, попередньо доданий до зони відповідальності модуля контролю цілісності, що мало на меті перевірити швидкість реакції агента на зміни у файловій структурі.



```
GNU nano 7.2 /var/ossec/etc/ossec.conf *
<sca>
  <enabled>yes</enabled>
  <scan_on_start>yes</scan_on_start>
  <interval>12h</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>

<integration>
  <name>virustotal</name>
  <api_key>                                </api_key>
  <group>syscheck</group>
  <alert_format>json</alert_format>
</integration>

<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>
```

Рис. 3.32 Налаштування для виявлення шкідливих файлів за допомогою VirusTotal

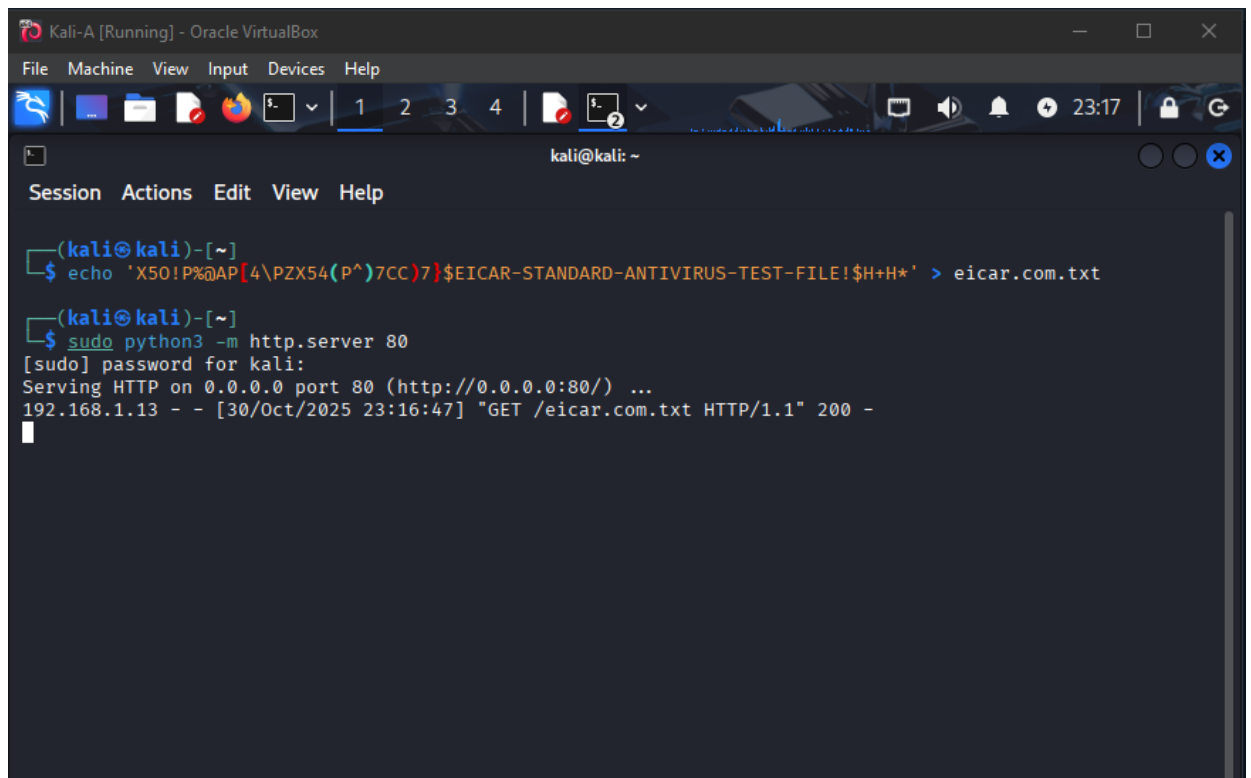


Рис. 3.33 Створення шкідливого файлу та підняття серверу

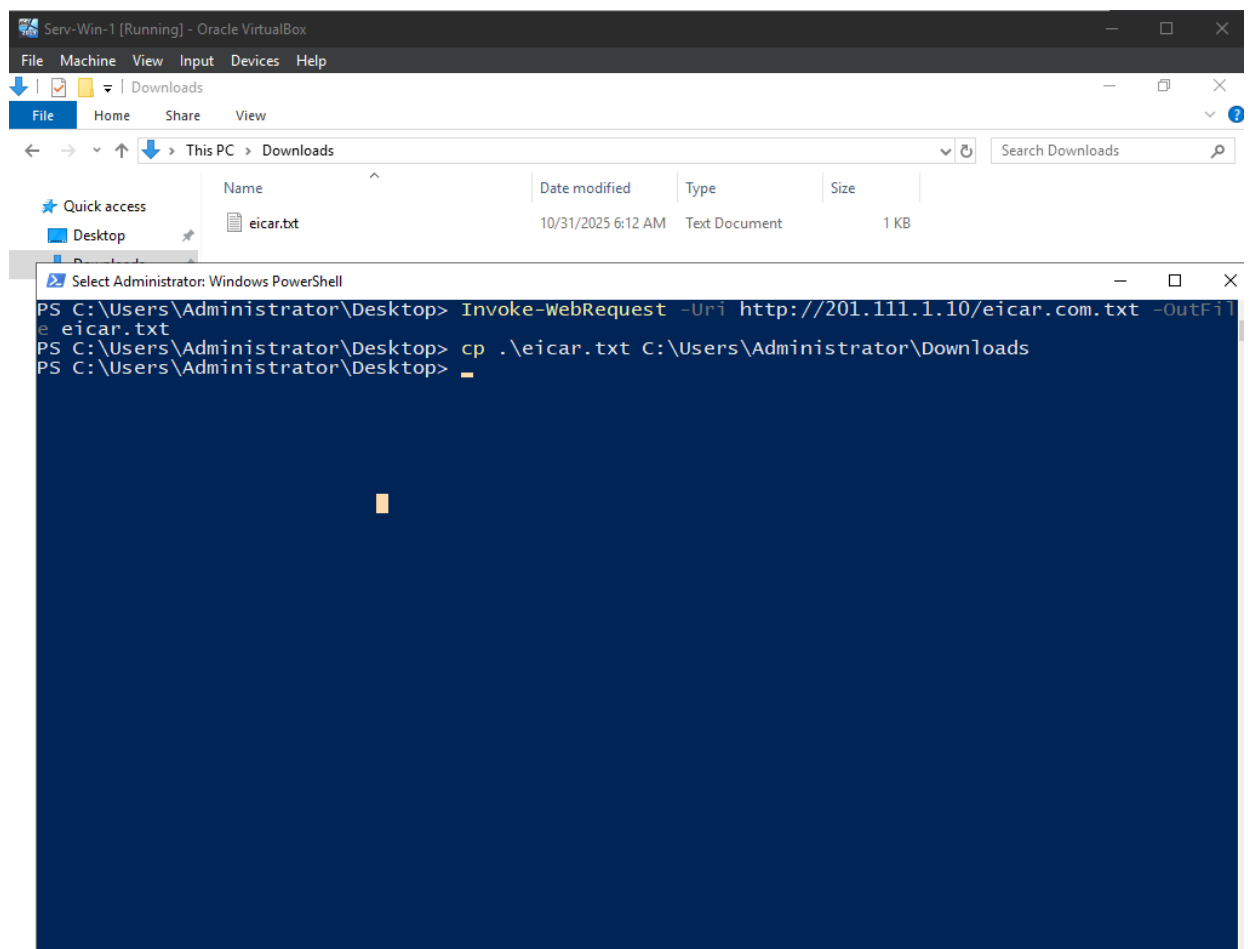


Рис. 3.34 Завантаження шкідливого файлу

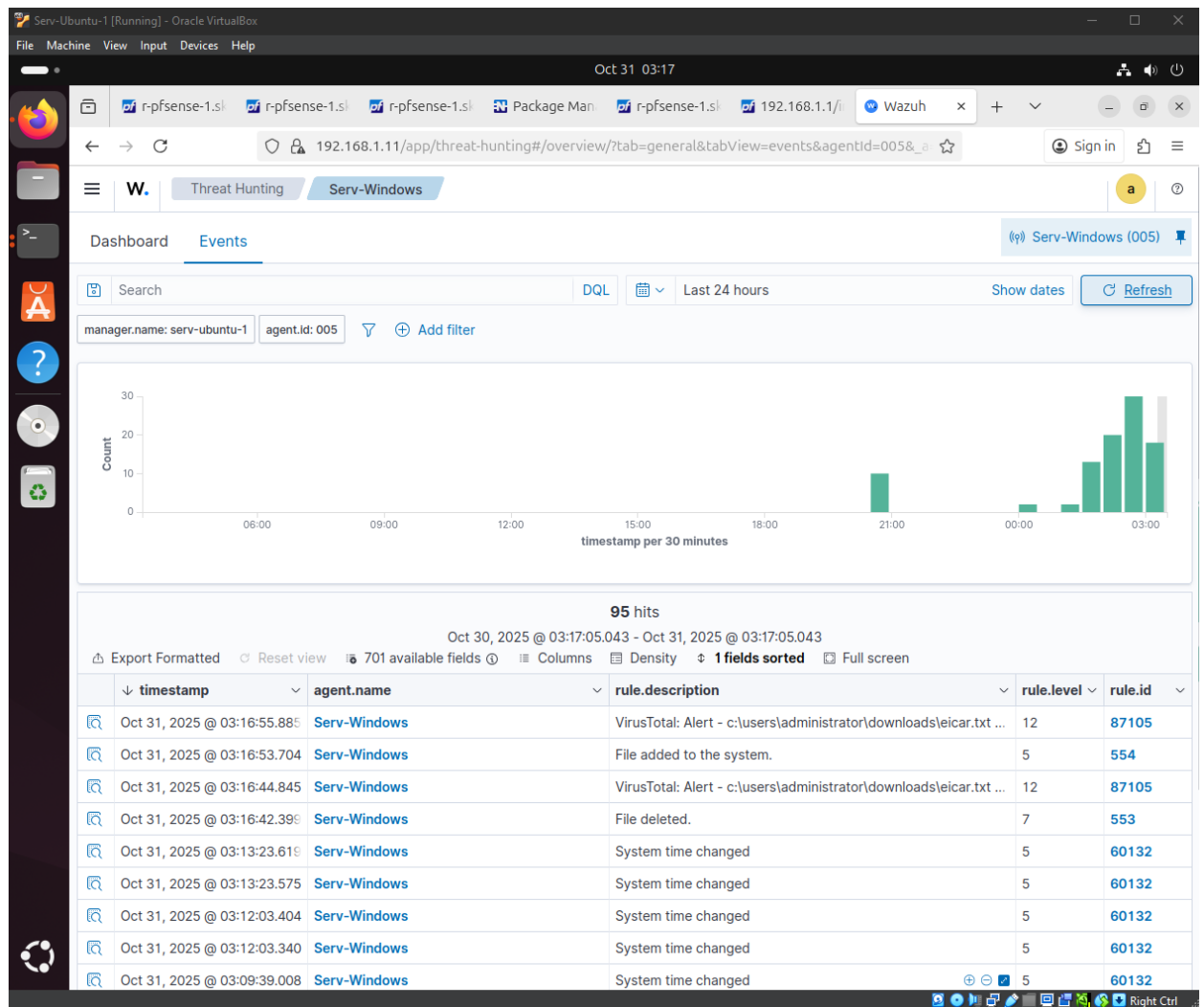


Рис. 3.35 Реакція Wazuh на шкідливий файл

Реакція системи безпеки на появу нового об'єкта була зафіксована миттєво завдяки активному стану підсистеми File Integrity Monitoring. Агент Wazuh в автоматичному режимі виявив подію створення файлу, провів обчислення його контрольних сум та передав метадані на центральний сервер для аналізу. Цей етап підтвердив коректність функціонування механізмів аудиту Windows, які здатні відстежувати зміни файлової системи в реальному часі без необхідності проведення повного планового сканування диска. Важливим аспектом стало те, що моніторинг здійснювався у фоновому режимі, не впливаючи на продуктивність робочої станції, що є критичним показником для експлуатації в реальних умовах.

Обробка отриманої телеметрії на стороні сервера завершилася успішною класифікацією інциденту як критичної загрози безпеці. Кореляційний механізм

здійснив порівняння сигнатури виявленого файлу з вбудованою базою загроз, ідентифікувавши об'єкт як тестовий вірус EICAR. В результаті в консолі управління було сформовано сповіщення високого рівня критичності, яке містило вичерпну інформацію про шлях до файлу, час його створення та обліковий запис, під яким було ініційовано завантаження. Успішна реалізація цього сценарію демонструє здатність побудованої системи забезпечувати комплексний захист гетерогенної мережі, поєднуючи контроль цілісності даних із функціями антивірусного виявлення на рівні кінцевих точок.

3.3. Розробка власних правил кореляції та механізмів реагування

Стандартна політика безпеки платформи Wazuh класифікує поодинокі помилки автентифікації як події середнього рівня значущості, присвоюючи їм п'ятий рівень критичності. Такий підхід спрямований на зменшення інформаційного шуму в великих мережах, де помилки введення паролів користувачами є буденним явищем, проте в умовах підвищених вимог до безпеки це створює ризик ігнорування початкових етапів цілеспрямованих атак. У ході дослідження було визначено, що стандартне правило з ідентифікатором 5710, яке реєструє спробу входу неіснуючого користувача або помилку пароля, недостатньо акцентує увагу адміністратора на інциденті. Для реалізації політики "нульової толерантності" до несанкціонованого доступу було прийнято рішення про примусове підвищення пріоритету даної події до дванадцятого рівня, що переводить її в розряд критичних інцидентів безпеки.

Технічна реалізація цього завдання здійснювалася шляхом редагування локального файлу правил `local_rules.xml`, що дозволяє зберігати кастомні налаштування окремо від системних, забезпечуючи їх збереження під час оновлення платформи. Замість створення нового правила було використано механізм перезапису існуючого об'єкта шляхом додавання атрибута `overwrite` до тегу правила з ідентифікатором 5710. У новій конфігурації параметр `level` було змінено з базового значення 5 на 12. Така модифікація змушує аналітичний рушій системи розглядати кожну окрему невдалу спробу підключення по SSH як

серйозну загрозу, незалежно від частоти повторення запитів або джерела походження трафіку.

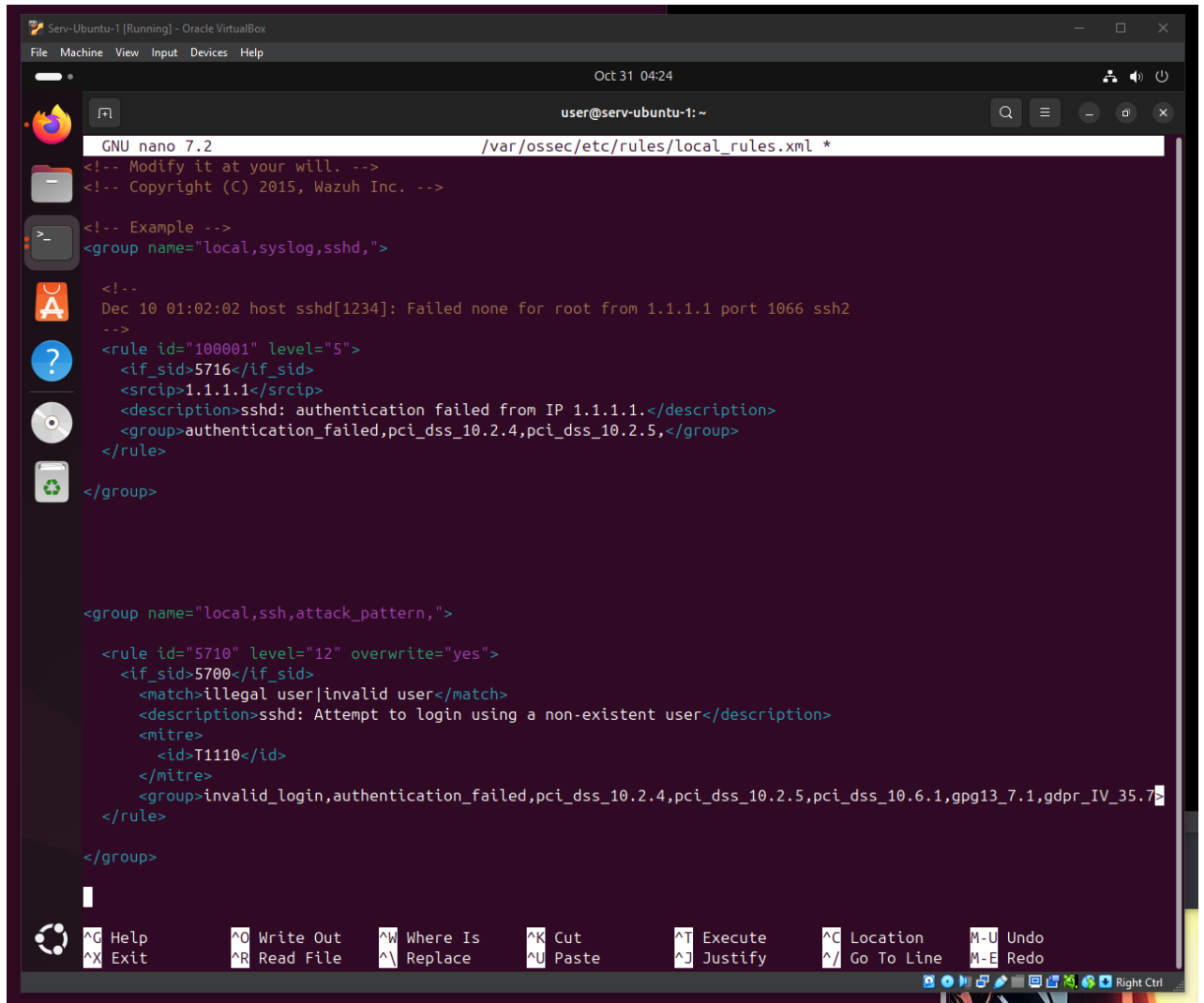


Рис. 3.36 Створення нового правила

Впровадження змін у логіку кореляції призвело до суттєвого підвищення чутливості системи моніторингу. Тестування оновленої конфігурації за допомогою генерації поодиноких запитів із некоректними даними підтвердило миттєву появу алерту високого пріоритету на інформаційній панелі. Такий підхід забезпечив гарантовану видимість будь-яких спроб зондування служби віддаленого доступу, усуваючи необхідність накопичення певної кількості помилок для спрацювання механізму детектування. Це налаштування стало фундаментом для подальшої автоматизації реагування, оскільки скрипти блокування зазвичай прив'язуються саме до подій високого рівня критичності,

дозволяючи системі реагувати на загрозу вже після першого "стуку" ЗЛОВМИСНИКА.

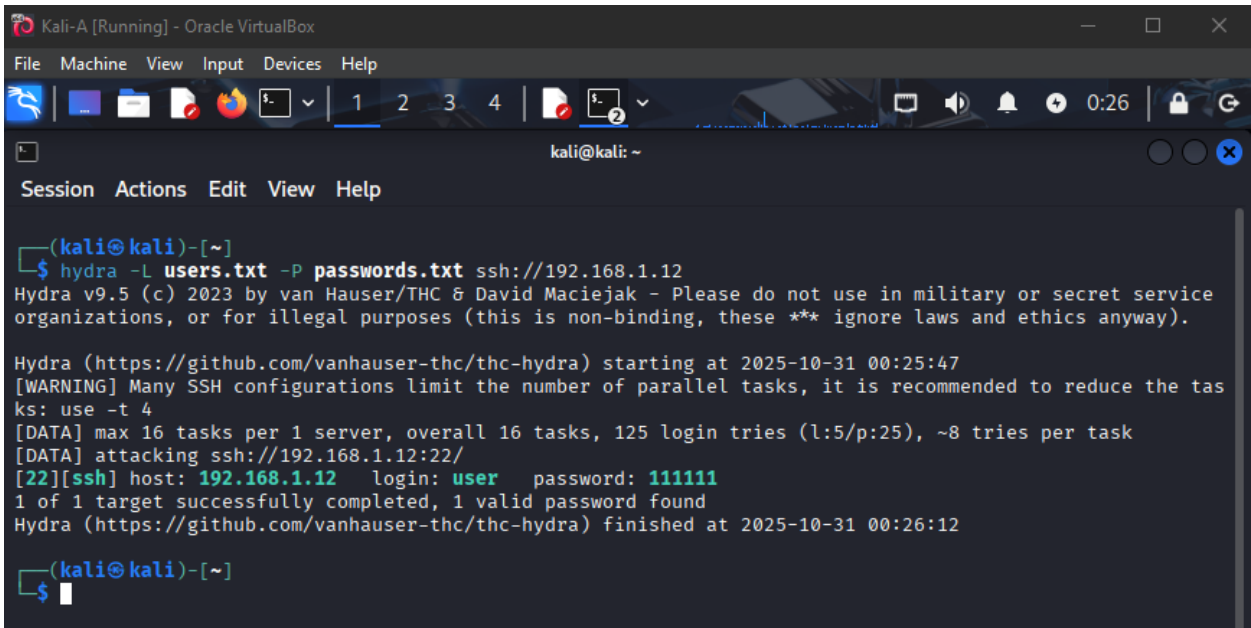


Рис. 3.37 Повторна атака

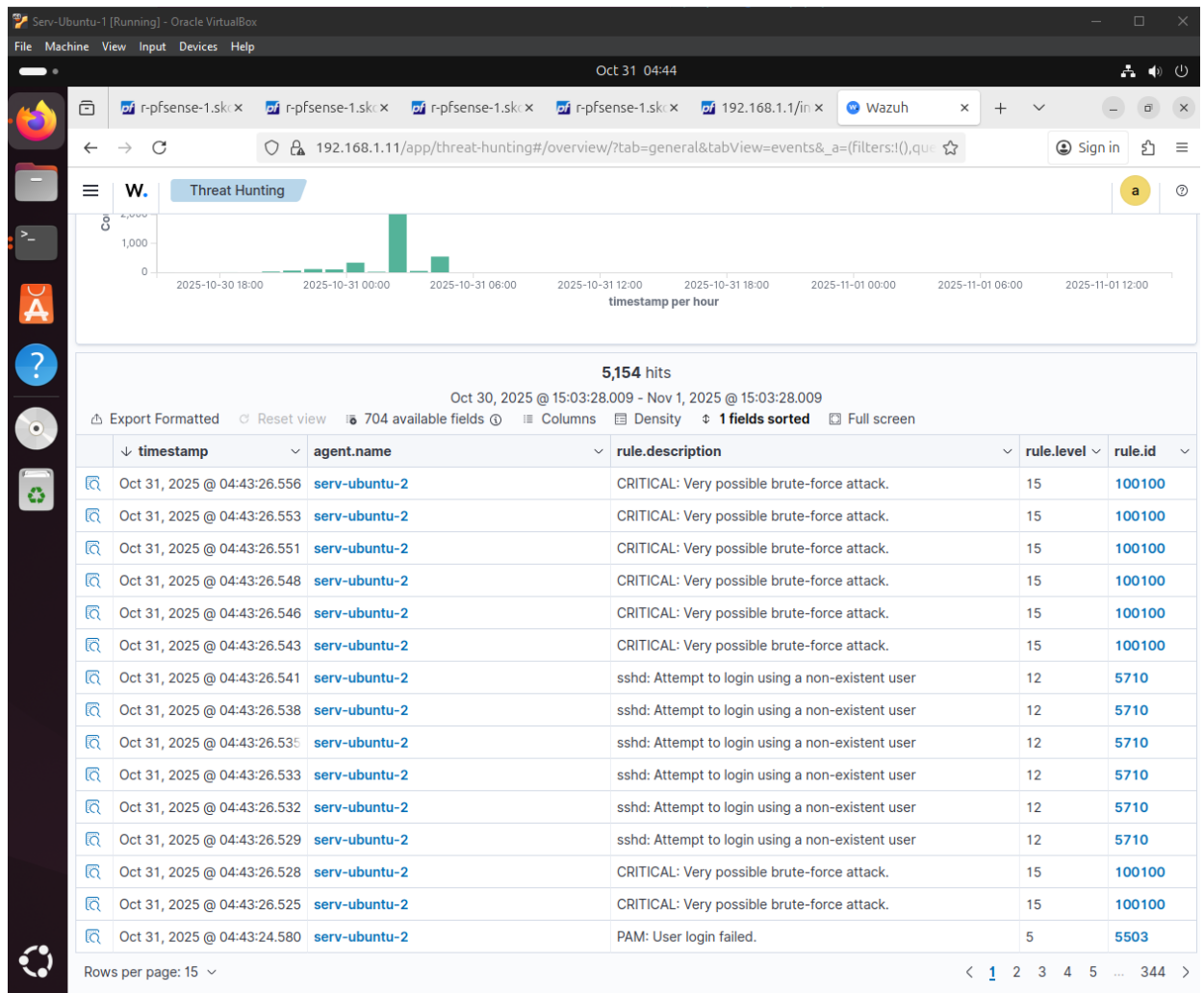
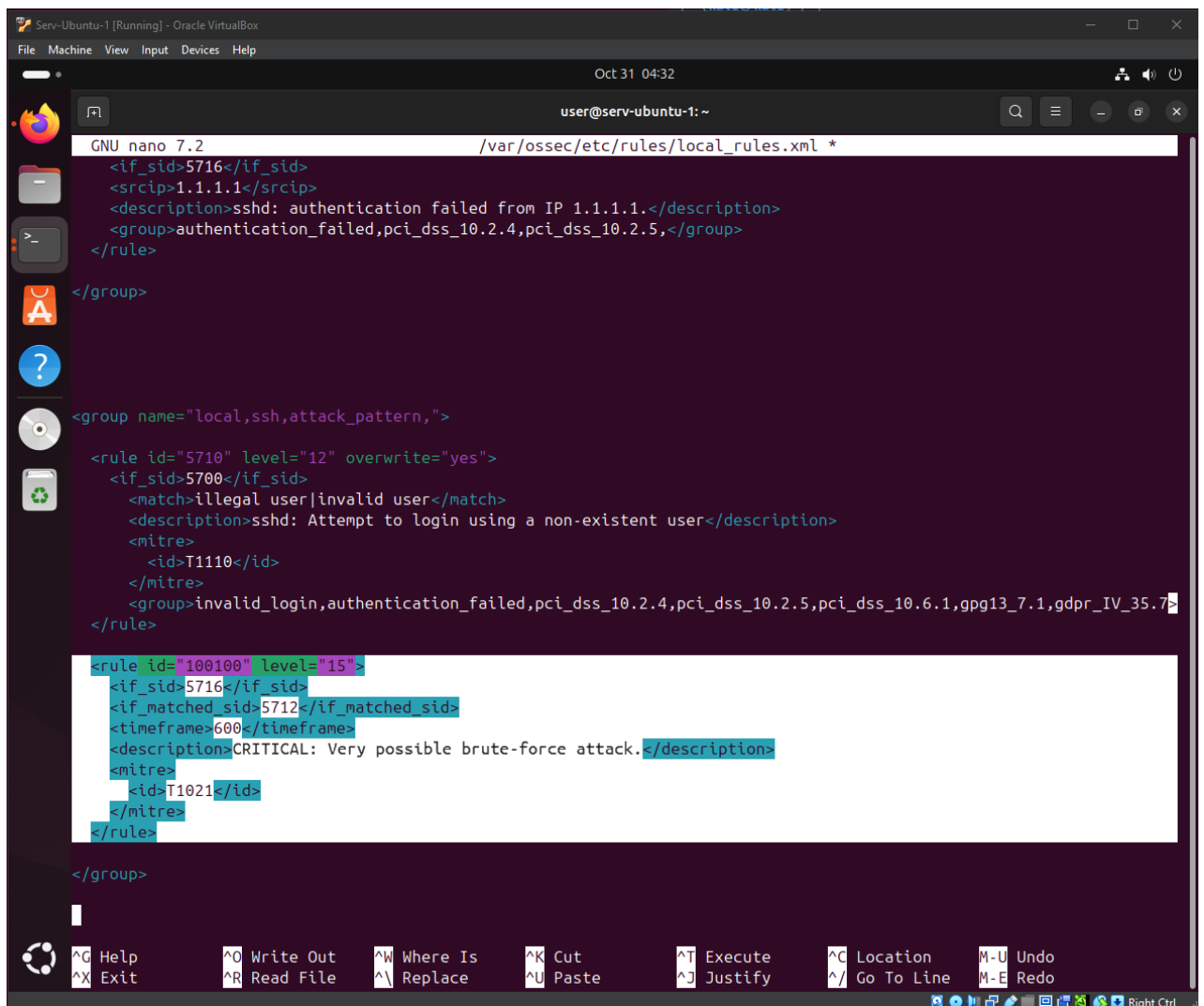


Рис. 3.38 Реакція Wazuh на повторну атаку

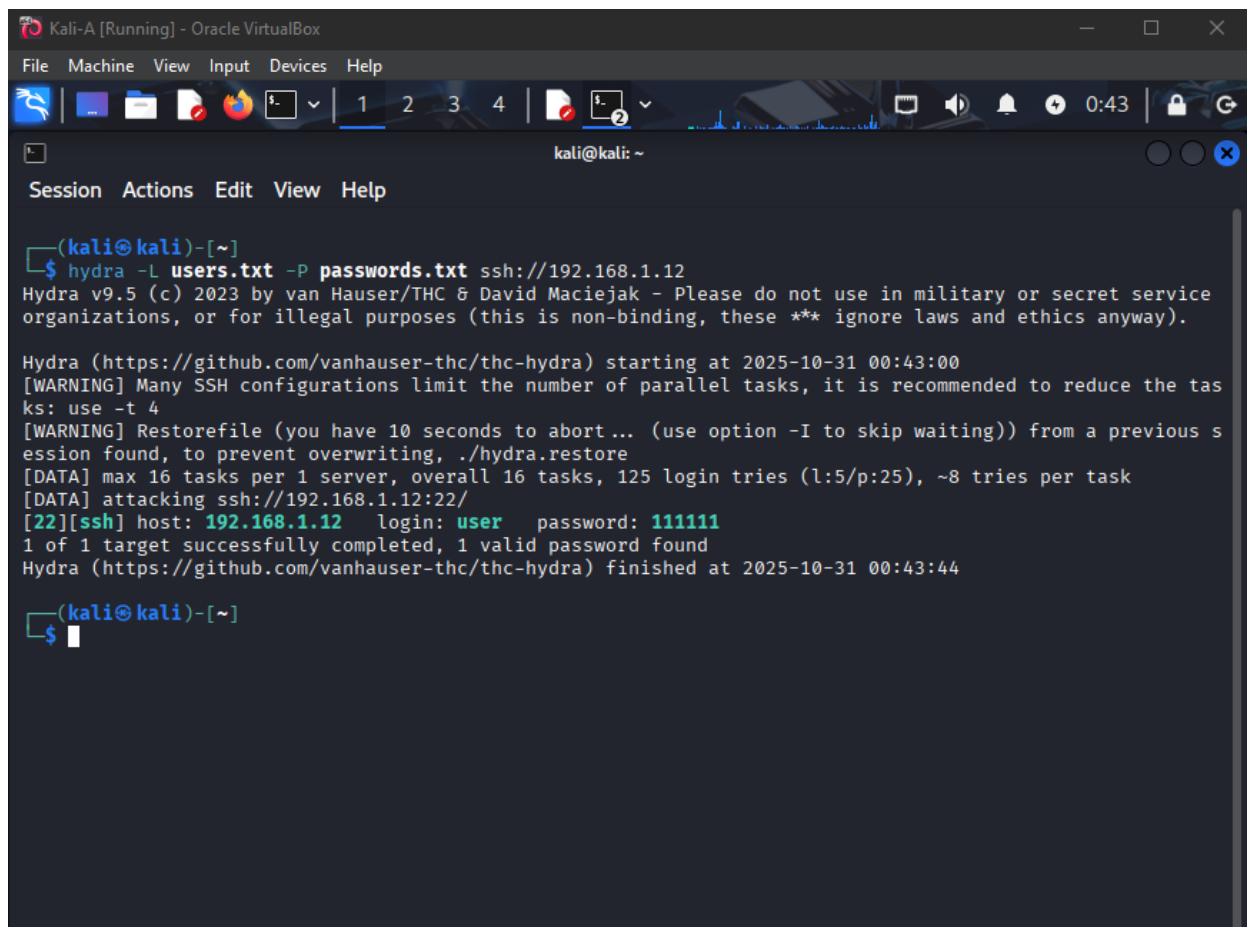
Ефективність систем моніторингу визначається їхньою здатністю виявляти причинно-наслідкові зв'язки між розрізненими подіями, які окремо можуть виглядати легітимно, але у певній послідовності свідчать про критичний інцидент. У рамках дослідження було розроблено сценарій виявлення успішної компрометації облікового запису, що відбулася одразу після серії невдалих спроб входу. Стандартні засоби SIEM часто розглядають атаку перебору паролів та подальший успішний вхід як дві незалежні події, що може призвести до втрати контексту адміністратором безпеки. Для вирішення цієї проблеми створено спеціалізоване правило кореляції з ідентифікатором 100100, яке об'єднує ці події в єдиний логічний ланцюжок, сигналізуючи про те, що атака злоумисника завершилася успішним підбором пароля.



```
GNU nano 7.2 /var/ossec/etc/rules/local_rules.xml *
<rule id="5716" level="12" overwrite="yes">
  <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>
<group name="local,ssh,attack_pattern,">
  <rule id="5710" level="12" overwrite="yes">
    <if_sid>5700</if_sid>
    <match>illegal user|invalid user</match>
    <description>sshd: Attempt to login using a non-existent user</description>
    <mitre>
      <id>T1110</id>
    </mitre>
    <group>invalid_login,authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5,pci_dss_10.6.1,gpg13_7.1,gdpr_IV_35.7</group>
  </rule>
  <rule id="100100" level="15">
    <if_sid>5716</if_sid>
    <if_matched_sid>5712</if_matched_sid>
    <timeframe>600</timeframe>
    <description>CRITICAL: Very possible brute-force attack.</description>
    <mitre>
      <id>T1021</id>
    </mitre>
  </rule>
</group>
```

Рис. 3.39 Створення нового правила

Реалізація композитного правила здійснювалася у файлі `local_rules.xml` із застосуванням умовних операторів залежності. Логіка роботи правила базується на відстеженні події успішної автентифікації SSH як тригера. Додатковою умовою активації стала перевірка наявності попередніх спрацювань правила про brute-force атаку або серію помилок входу з тієї ж самої IP-адреси протягом визначеного часового вікна, встановленого на рівні 600 секунд. Використання тегів `<if_sid>` для поточної події та `<if_matched_sid>` для історичного контексту дозволило навчити систему розрізняти звичайний вхід користувача, який міг помилитися один раз, від ситуації, коли входу передувала масована атака.



```
Kali-A [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4
kali@kali: ~
Session Actions Edit View Help

(kali@kali)-[~]
$ hydra -L users.txt -P passwords.txt ssh://192.168.1.12
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-31 00:43:00
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tas
ks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous s
ession found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 125 login tries (l:5/p:25), ~8 tries per task
[DATA] attacking ssh://192.168.1.12:22/
[22][ssh] host: 192.168.1.12 login: user password: 111111
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-10-31 00:43:44

(kali@kali)-[~]
$
```

Рис. 3.40 Повторна атака для тестування правила

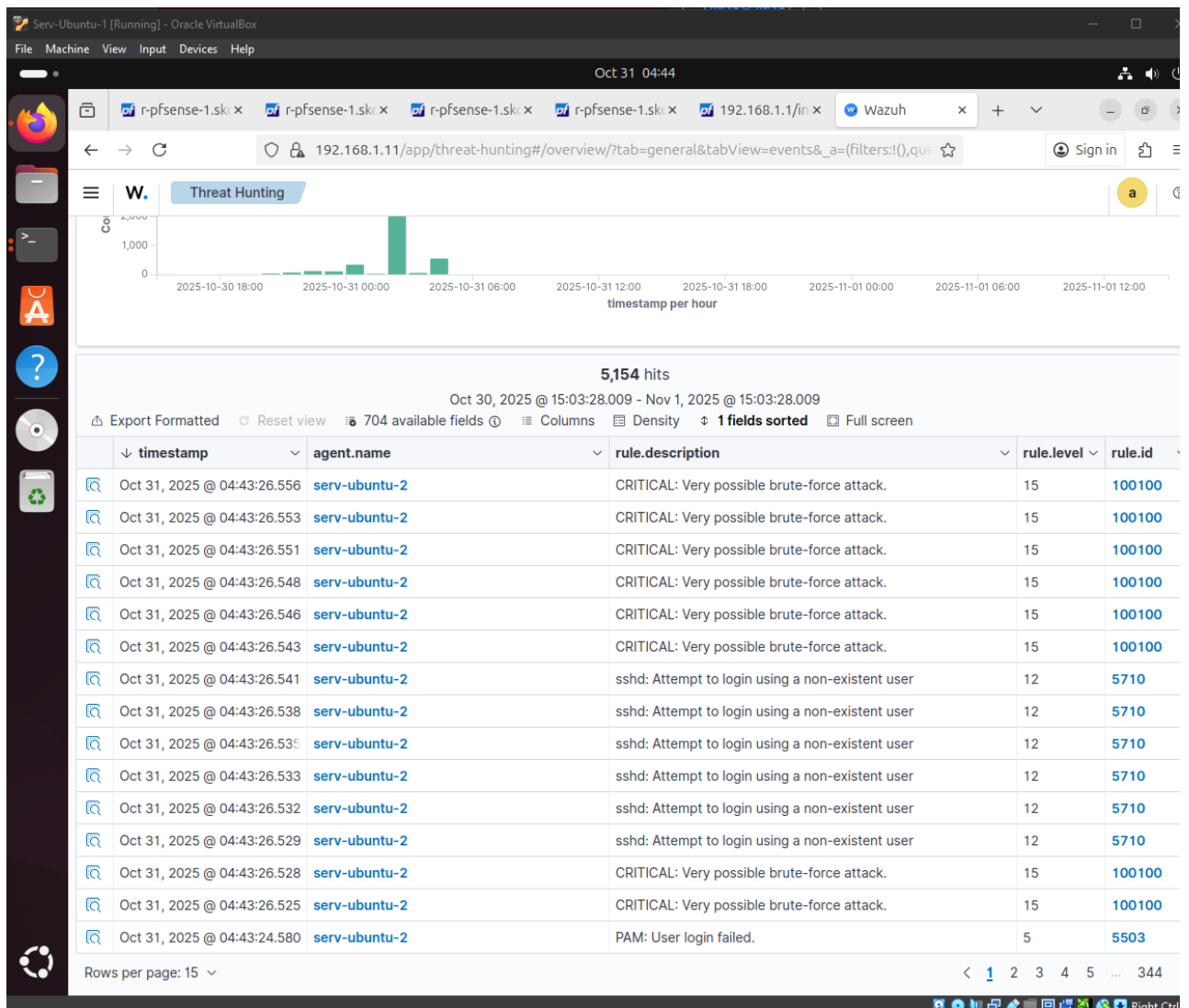
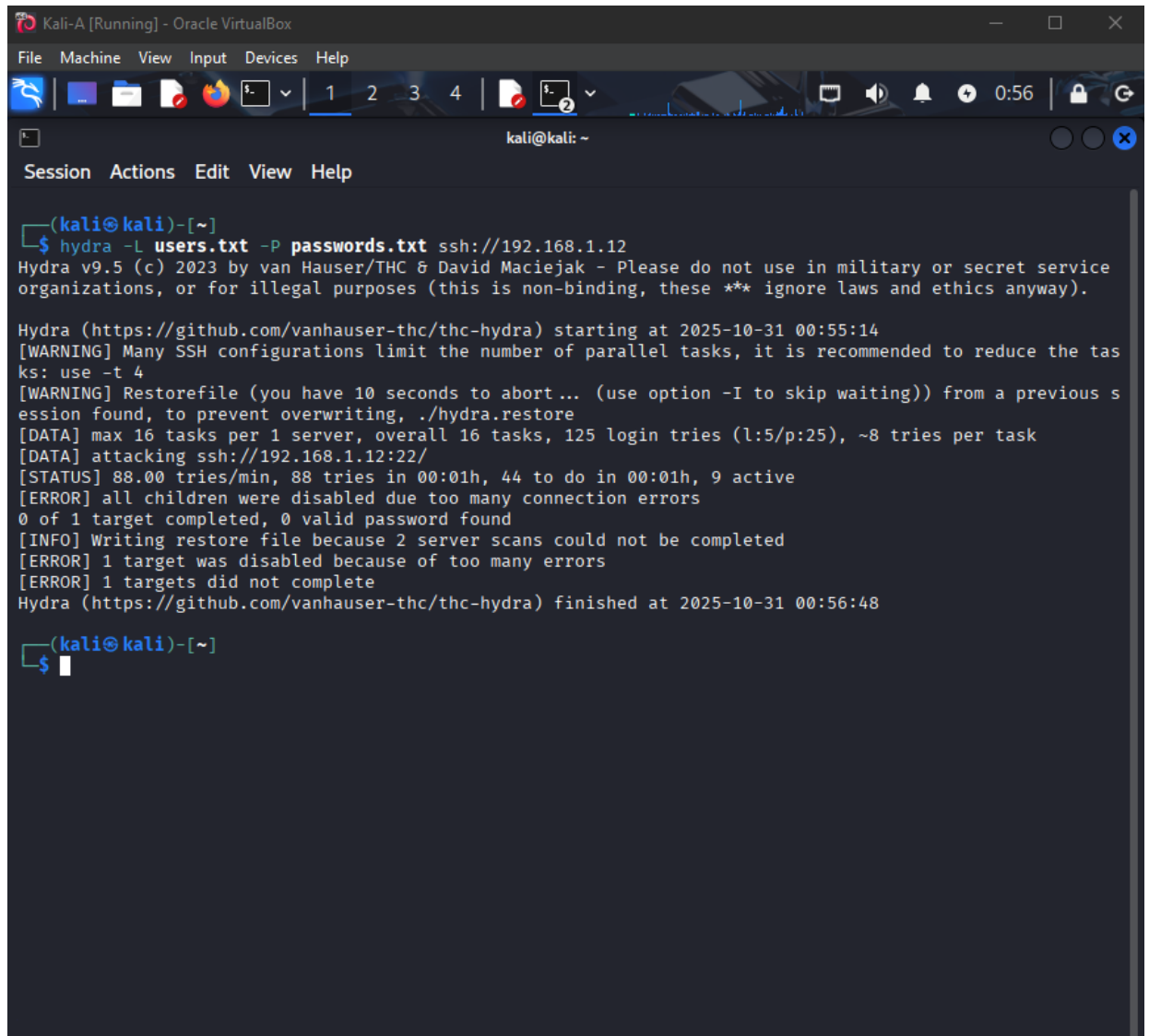


Рис. 3.41 Реакція Wazuh на повторну атаку

Впровадження розробленого правила дозволило автоматизувати процес виявлення інцидентів високого ступеня ризику, де факт проникнення підтверджується поведінковим аналізом. Тестування механізму проводилося шляхом імітації дій зловмисника: спочатку генерувалася серія запитів із неправильними паролями, після чого здійснювався вхід із використанням валідних облікових даних. Система коректно зіставила послідовність дій, згенерувавши критичне сповіщення про успішний злам після перебору паролів. Такий підхід значно підвищує ситуаційну обізнаність оператора центру безпеки, фокусуючи увагу на тих інцидентах, де захисний периметр було фактично подолано.

Трансформація системи моніторингу з пасивного спостерігача в активний інструмент протидії загрозам була досягнута шляхом налаштування модуля

виконання віддалених команд. При надходженні сигналу тривоги від менеджера агент ініціює взаємодію з підсистемою фільтрації пакетів iptables операційної системи Linux. Скрипт автоматично додає правило, яке відхиляє всі вхідні пакети від підозрілої IP-адреси, фактично розриваючи активні сесії та унеможлиблюючи створення нових з'єднань. Цей процес відбувається в автоматичному режимі без необхідності втручання адміністратора, що мінімізує час реакції на інцидент до часток секунди.



```
Kali-A [Running] - Oracle VirtualBox
File Machine View Input Devices Help
1 2 3 4
kali@kali: ~
Session Actions Edit View Help

(kali@kali)-[~]
$ hydra -L users.txt -P passwords.txt ssh://192.168.1.12
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-10-31 00:55:14
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tas
ks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous s
ession found, to prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 125 login tries (l:5/p:25), ~8 tries per task
[DATA] attacking ssh://192.168.1.12:22/
[STATUS] 88.00 tries/min, 88 tries in 00:01h, 44 to do in 00:01h, 9 active
[ERROR] all children were disabled due too many connection errors
0 of 1 target completed, 0 valid password found
[INFO] Writing restore file because 2 server scans could not be completed
[ERROR] 1 target was disabled because of too many errors
[ERROR] 1 targets did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-10-31 00:56:48

(kali@kali)-[~]
$
```

Рис. 3.43 Повторна атака

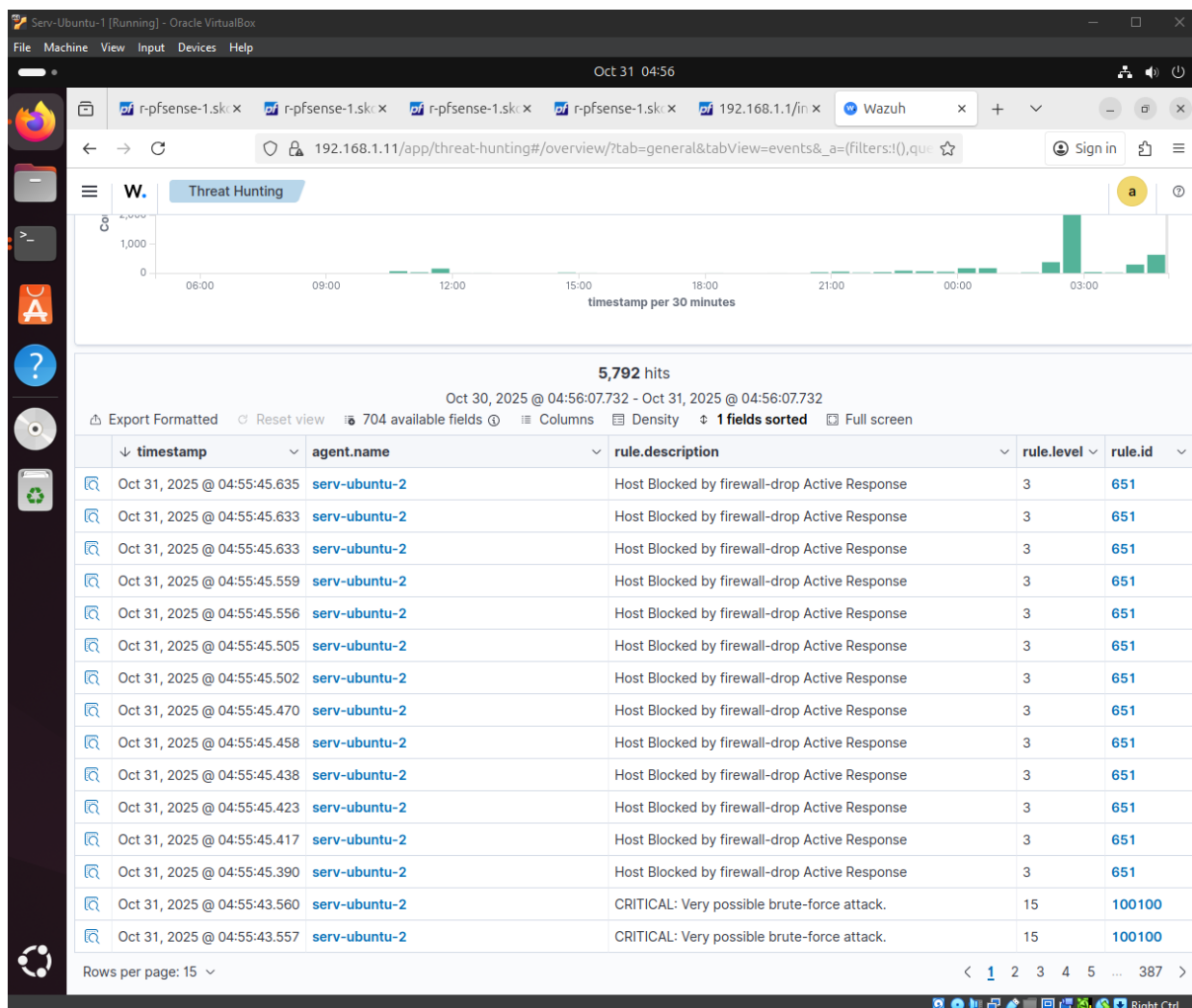


Рис. 3.44 Реакція Wazuh на повторну атаку

Експериментальна верифікація налаштованого механізму проводилася шляхом повторного запуску атаки з вузла Kali Linux. Після реєстрації першої події невдалої автентифікації, яка завдяки попереднім налаштуванням отримала статус критичної, система миттєво активувала процедуру блокування. Результати тестування мережевої доступності показали повну втрату зв'язку між атакуючою машиною та цільовим сервером: утиліта ping фіксувала втрату пакетів, а спроби встановлення нових SSH-сесій завершувалися тайм-аутом. Одночасно в журналі подій Wazuh з'явився запис про успішне виконання команди активного реагування, що підтвердило здатність системи самостійно нейтралізувати виявлені загрози в реальному часі.

Висновки до третього розділу:

Практична реалізація та всебічне тестування розгорнутої системи підтвердили ефективність обраної архітектури для захисту гетерогенних інфраструктур, де поєднання агентських та безагентських методів збору телеметрії забезпечило повну видимість подій безпеки на всіх рівнях мережевої моделі. Експериментальна верифікація сценаріїв атак продемонструвала здатність комплексу коректно детектувати різноманітні вектори загроз, від SQL-ін'єкцій у специфічних контейнеризованих середовищах до спроб підбору паролів та появи шкідливого програмного забезпечення на кінцевих точках. Критичним досягненням роботи стало успішне впровадження та налагодження механізмів активного реагування, які трансформували систему моніторингу з інструменту пасивного аудиту в засіб автоматизованої протидії, здатний самостійно блокувати джерела атак у реальному часі. Отримані результати свідчать про те, що розроблена конфігурація правил кореляції та інтеграція компонентів Wazuh, Snort і Docker створюють надійний, адаптивний контур безпеки, готовий до протистояння актуальним кіберзагрозам.

ВИСНОВКИ

У даній кваліфікаційній роботі було вирішено завдання з проектування та практичної реалізації комплексної системи виявлення та реагування на інциденти інформаційної безпеки в гетерогенних середовищах.

При виконанні роботи було проведено ґрунтовний аналіз сучасних підходів до моніторингу, зокрема розглянуто функціональні можливості комерційних платформ, таких як Splunk та IBM QRadar, а також рішень з відкритим кодом, зокрема ELK Stack та Wazuh. Для кожної системи вивчалися архітектурні особливості, моделі ліцензування та здатність працювати в змішаних інфраструктурах. На основі аналізу було зроблено вибір на користь платформи Wazuh як оптимального інструменту, що поєднує можливості SIEM та XDR.

Дослідження показало, що інтеграція Wazuh у поєднанні з мережевим шлюзом pfSense та системою виявлення вторгнень Snort є найбільш ефективним архітектурним рішенням для захисту різнорідних мереж, де одночасно функціонують Windows, Linux та контейнеризовані сервіси. Ця конфігурація забезпечує багаторівневий контроль безпеки, дозволяючи корелювати події з кінцевих точок та мережевого периметра в єдиній консолі управління. У результаті проведеного дослідження було розроблено та розгорнуто діючий макет системи, який враховує специфіку моніторингу ефемерних середовищ Docker та особливості аудиту пропрієтарних операційних систем. Реалізований проєкт дозволяє не лише збирати телеметрію, а й вибудовувати логічні зв'язки між розрізненими подіями для ідентифікації складних векторів атак.

Тестування розгорнутої системи, проведене шляхом моделювання кібератак із використанням спеціалізованого інструментарію, дозволило підтвердити її ефективність та високу швидкість реакції. Отримані результати засвідчили здатність комплексу успішно виявляти та блокувати як мережеві вторгнення, так і шкідливу активність на рівні файлової системи хостів. Розгортання цього проєкту значно підвищило рівень захищеності модельованої інфраструктури.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Miller D. R. et al. Security Information and Event Management (SIEM) Implementation. New York : McGraw-Hill, 2010. 432 p.
2. Kent K., Souppaya M. Guide to Computer Security Log Management : SP 800-92. Gaithersburg : National Institute of Standards and Technology, 2006.
3. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection — Information security management systems — Requirements. 3rd ed. Geneva : ISO/IEC, 2022. 19 p.
4. Payment Card Industry (PCI) Data Security Standard. Requirements and Security Assessment Procedures. Version 4.0. PCI Security Standards Council, 2022.
5. Gerhards R. The Syslog Protocol : RFC 5424. Internet Engineering Task Force (IETF), 2009.
6. Gormley C., Tong Z. Elasticsearch: The Definitive Guide. Sebastopol : O'Reilly Media, 2015. 724 p.
7. Splunk Enterprise Documentation. Splunk Inc.. URL: <https://docs.splunk.com/Documentation/Splunk>
8. IBM QRadar SIEM 7.4.0 Documentation. IBM Corporation. URL: https://www.ibm.com/docs/en/qsip/7.4.0?topic=SS42VS_7.4/com.ibm.qradar.doc/c_qradar_pdfs.htm
9. ArcSight Documentation. Micro Focus. URL: <https://www.microfocus.com/documentation/arcsight/#gsc.tab=0>
10. Microsoft Sentinel documentation. Microsoft. URL: <https://learn.microsoft.com/en-us/azure/sentinel/>
11. Graylog Documentation. Graylog, Inc. URL: <https://go2docs.graylog.org/current/home.htm>
12. Wazuh Documentation. Wazuh, Inc. URL: <https://documentation.wazuh.com/current/index.html>
13. Elastic Docs. Elasticsearch B.V. URL: <https://www.elastic.co/docs>
14. Innovation Insight for Extended Detection and Response (XDR). Gartner Research, 2020.

15. Hutchins E. M., Cloppert M. J., Amin R. M. Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains. Bethesda : Lockheed Martin Corporation, 2011.
16. MITRE ATT&CK® Framework. The MITRE Corporation. URL: <https://attack.mitre.org/>
17. Oracle VM VirtualBox User Manual. Oracle Corporation. URL: <https://www.virtualbox.org/manual/UserManual.html>
18. GNS3 Documentation. GNS3 Technologies Inc. URL: <https://docs.gns3.com/>
19. Merkel D. Docker: lightweight linux containers for consistent development and deployment. Linux Journal. 2014. Vol. 2014, no. 239. Article 2.
20. Hertzog R., O'Gorman J., Aharoni M. Kali Linux Revealed: Mastering the Penetration Testing Distribution. Offsec Press, 2017. 342 p.
21. Ylonen T., Lonvick C. The Secure Shell (SSH) Protocol Architecture : RFC 4251. Internet Engineering Task Force (IETF), 2006.
22. The Anti-Virus test file. European Institute for Computer Antivirus Research (EICAR). URL: <https://www.eicar.org/download-anti-malware-testfile/>
23. OWASP Top 10:2025. The OWASP Foundation. URL: https://owasp.org/Top10/2025/A05_2025-Injection/