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

ПОЯСНЮВАЛЬНА ЗАПИСКА

до випускної кваліфікаційної роботи бакалавра

на тему: «Проєкт гібридної інфраструктури виявлення та реагування на кіберінциденти на базі Microsoft Sentinel»

Виконав: студент 4-го курсу групи КБ-22-1
спеціальності 125 «Кібербезпека»
_____ Костянтин ШПИТА
Керівник: старший викладач кафедри КІ та КБ
_____ Олександра ПОКОТИЛО
Рецензент: доцент кафедри КІ та КБ, к.пед.н., доцент
_____ Олена ГОЛОВНЯ

Житомир – 2026

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

ЗАТВЕРДЖУЮ
Завідувач
кафедри комп'ютерної
інженерії та кібербезпеки
Андрій Єфіменко
2 лютого 2026 р.

ЗАВДАННЯ
на випускну кваліфікаційну роботу бакалавра

Студент: Шпита Костянтин Юрійович

Тема роботи: «Проєкт гібридної інфраструктури виявлення та реагування на кіберінциденти на базі Microsoft Sentinel»

затверджена Наказом університету від 2 лютого 2026 р. №58/с

Термін здачі студентом закінченої роботи 12 червня 2026 р.

Вихідні дані роботи (зазначається предмет і об'єкт дослідження):
Метою кваліфікаційної роботи є обґрунтування та проєктування моделі гібридної системи виявлення та реагування на інциденти інформаційної безпеки з використанням платформи Microsoft Sentinel.

Обладнання та програмне забезпечення:

1. Персональний комп'ютер з підтримкою віртуалізації, який має не менше 16 ГБ оперативної пам'яті
2. Операційні системи: Windows Server 2019 (для створення середовища на базі Active Directory) та Windows 10 (як клієнтська ОС)
3. Середовище віртуалізації: Oracle VirtualBox 7.2.6
4. Програмні засоби для розробки і тестування: Visual Studio Code (для написання вебдодатку, та розробки скриптів), веб-браузер Mozilla Firefox (для тестування вебдодатку і роботи з Azure)
5. Офісні програми: Microsoft Word (для оформлення звіту), PowerPoint (для підготовки презентації)

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

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
1	Миколайчук В.В.	10.02.2026	18.02.2026
2	Миколайчук В.В.	02.03.2026	25.03.2026
3	Миколайчук В.В.	01.05.2026	03.06.2026

Студент _____ Костянтин ШПИТА

Керівник _____ Олександра ПОКОТИЛО

Календарний план

№ з/п	Назви етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Постановка завдання	02.02.2026	Виконано
2	Аналіз можливостей	12.02.2026	Виконано
3	Збір літератури	23.02.2026	Виконано
4	Аналіз архітектури SIEM/SOAR систем	10.03.2026	Виконано
5	Аналіз типових загроз гібридного середовища	17.03.2026	Виконано
6	Аналіз ринку SIEM/SOAR-систем	26.03.2026	Виконано
7	Формування архітектурної схеми	03.04.2026	Виконано
8	Дослідження можливостей Sentinel	13.04.2026	Виконано
9	Дослідження KQL та Logic Apps	15.05.2026	Виконано
10	Побудова гібридної інфраструктури	03.06.2026	Виконано
11	Дослідження роботи гібридної інфраструктури	03.06.2026	Виконано

Студент _____ Костянтин ШПИТА

Керівник _____ Олександра ПОКОТИЛО

АНОТАЦІЯ

У кваліфікаційній роботі бакалавра розглянуто питання проєктування гібридної інфраструктури захисту інформаційної безпеки на базі платформи Microsoft Sentinel. Робота присвячена аналізу сучасних підходів до виявлення та реагування на кіберінциденти у середовищах, що поєднують локальну інфраструктуру та хмарні сервіси.

Пояснювальна записка містить 91 сторінку, 91 рисунок, 29 таблиць. Список використаних джерел налічує 27 найменувань та розміщений на 3 сторінках.

Метою кваліфікаційної роботи є обґрунтування, проєктування та дослідження моделі гібридної системи виявлення та реагування на інциденти інформаційної безпеки з використанням SIEM-платформи Microsoft Sentinel.

У роботі проаналізовано сучасні SIEM/SOAR-рішення та моделі їх розгортання, систематизовано вектори кібератак і моделі загроз на основі бази знань MITRE ATT&CK, розглянуто типові загрози локальному доменному середовищу Active Directory у гібридній інфраструктурі. Синтезовано узагальнену структурну схему системи виявлення інцидентів та проведено аналіз підходів інтеграції локальної інфраструктури з Microsoft Sentinel. Реалізовано та протестовано згенеровану схему на ефективність.

Ключові слова: КІБЕРБЕЗПЕКА, SIEM, SOAR, MICROSOFT SENTINEL, MITRE ATT&CK, ГІБРИДНА ІНФРАСТРУКТУРА, ACTIVE DIRECTORY, AZURE, CLOUDFLARE, PLAYBOOK, DASHBOARD.

Змн.	Арк.	№ докум.	Підпис	Дата	ККІКБ.КР.Б – 125 – 19 – ПЗ		
Розроб.		Шпита К.Ю.		10.06.26	Проєкт гібридної інфраструктури виявлення та реагування на кіберінциденти на базі Microsoft Sentinel	Літ.	
Керівник		Покотило О.А.		10.06.26		Арк. 4	Аркушів 109
Реценз.		Головня О.С.		10.06.26		ФІКТ, Гр. КБ-22-1	
Н. контр.		Покотило О.А.		10.06.26			
Затверд.		Єфіменко А.А.		10.06.26			

ANNOTATION

This bachelor’s thesis examines the design of a hybrid information security infrastructure based on the Microsoft Sentinel platform. The thesis analyzes modern approaches to detecting and responding to cyber incidents in environments that combine on-premises infrastructure and cloud services.

The explanatory note contains 91 pages, 91 figures, and 29 tables. The list of references includes 27 items and spans 3 pages.

The aim of this thesis is to justify, design, and investigate a model of a hybrid system for detecting and responding to information security incidents using the Microsoft Sentinel SIEM platform.

The thesis analyzes modern SIEM/SOAR solutions and their deployment models, systematizes cyberattack vectors and threat models based on the MITRE ATT&CK knowledge base, and examines typical threats to the local Active Directory domain environment in a hybrid infrastructure. A generalized structural diagram of the incident detection system is synthesized, and an analysis of approaches to integrating the local infrastructure with Microsoft Sentinel is conducted. The generated diagram is implemented and tested for effectiveness.

Keywords: CYBERSECURITY, SIEM, SOAR, MICROSOFT SENTINEL, MITRE ATT&CK, HYBRID INFRASTRUCTURE, ACTIVE DIRECTORY, AZURE, CLOUDFLARE, PLAYBOOK, DASHBOARD.

ЗМІСТ

ВСТУП 8
РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ В МОЖЛИВИХ СЕРЕДОВИЩАХ 10
1.1 Системи керування подіями та захисту інформації (SIEM) та системи оркестрування, автоматизації та реагування (SOAR) 10
1.2 Систематизація векторів кібератак та моделей загроз на основі бази знань MITRE ATT&CK 16
1.3 Порівняльний аналіз методів побудови SIEM/SOAR архітектури та аналіз рішень на ринку 18
РОЗДІЛ 2 ПРОЄКТУВАННЯ МОДЕЛІ ГІБРИДНОЇ ІНФРАСТРУКТУРИ ЗАХИСТУ 25
2.1 Формалізація об’єкта дослідження та загроз інформаційній безпеці гібридного середовища 25
2.2 Синтез структурної схеми системи виявлення інцидентів на базі платформи Microsoft Sentinel 29
2.3 Проєктування мережної топології та інтеграції з хмарним середовищем . 38
РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ СИСТЕМИ 41
3.1 Реалізація компонентів моніторингу та інтеграція різнорідних джерел даних у єдиний інформаційний простір 41
3.2 Робота з KQL для візуалізації метрик (SIEM) та створення сценаріїв автоматизованого реагування (SOAR) 53
3.3 Експериментальна перевірка ефективності запропонованого рішення та аналіз результатів тестування 73
ВИСНОВКИ 88
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 89
ДОДАТКИ 92

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

SOC – Security Operation Centers

AD – Active Directory

SIEM – Security Information and Event Management

SOAR – Security Orchestration, Automation and Response

IAM – Identity and Access Management

MITRE ATT&CK – Adversarial Tactics, Techniques and Common Knowledge

TTP – Tactics, Techniques and Procedures

VPN – Virtual Private Network

AMA – Azure Monitor Agent

DCR – Data Collection Rules

WEC – Windows Event Collection

ARM – Azure Resource Manager

API – Application Programming Interface

IT – Information Technology

KQL – Kusto Query Language

IIS – Internet Information Services

LAW – Log Analytics Workspace

ВСТУП

Актуальність теми. У сучасних умовах цифровізації та широкого впровадження інформаційних технологій, питання забезпечення інформаційної безпеки набувають особливої актуальності і важливості. Зростання обсягів оброблюваних даних, поширення хмарних сервісів, віддаленої роботи та гібридних моделей інфраструктури призводить до ускладнення архітектури корпоративних мереж і, відповідно, до розширення поверхні атаки, тому необхідно приділяти неабияку увагу безпеці, підтримуючи актуальність рішень і їх ефективність. В умовах стрімкого зростання кількості кіберінцидентів та дефіциту кваліфікованих кадрів у сфері кіберзахисту, критично важливим стає перехід до автоматизованих систем аналізу телеметрії. Одним із найперспективніших сучасних SIEM-рішень є платформа Microsoft Sentinel, яка реалізує хмарну модель моніторингу безпеки та підтримує інтеграцію з локальними й хмарними джерелами телеметрії та може використовувати вбудовані механізми штучного інтелекту для автоматичного реагування на загрози. Використання Microsoft Sentinel у гібридних середовищах потребує обґрунтованого підходу до проєктування архітектури системи, вибору механізмів інтеграції локальної інфраструктури та визначення логіки збору й аналізу подій безпеки з метою оптимізації витрат на збереження даних та мінімізації часу реагування на критичні інциденти.

Метою кваліфікаційної роботи є побудова, дослідження та обґрунтування моделі гібридної інфраструктури захисту інформаційної безпеки на базі платформи Microsoft Sentinel з урахуванням загроз локального доменного середовища.

Визначена мета обумовлює наступні завдання:

- проведення аналізу предметної галузі дослідження;
- аналіз можливостей Microsoft Sentinel як сучасної SIEM/SOAR-системи;
- побудова типової мережі на базі Active Directory для проведення досліджень;
- розгортання та підготовка хмарної інфраструктури для функціонування Microsoft Sentinel;

- інтегрування локальної інфраструктури тестової мережі з хмарними сервісами Azure;
- дослідження функціонування Microsoft Sentinel у побудованій мережі шляхом відтворення сценаріїв зловмисних дій;
- оцінка ефективності впроваджених засобів моніторингу та автоматизованого реагування на інциденти безпеки.

Об’єктом дослідження є процес моніторингу та автоматичного реагування на інциденти безпеки в інфраструктурі корпоративних мереж.

Предметом дослідження є методи, технології та інструментальні засоби побудови гібридних систем виявлення та реагування на кіберінциденти на базі SIEM/SOAR-платформи Microsoft Sentinel та засобів автоматизації Azure Logic Apps.

У процесі виконання кваліфікаційної роботи було розроблено проект захищеної корпоративної мережі, інтегрованої з хмарною SIEM/SOAR-системою Microsoft Sentinel, та досліджено процеси автоматизації реагування на інциденти безпеки.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ ТА ЗАСОБІВ МОНІТОРИНГУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ В МОЖЛИВИХ СЕРЕДОВИЩАХ

1.1 Системи керування подіями та захисту інформації (SIEM) та системи оркестрування, автоматизації та реагування (SOAR)

SIEM - це один із основних компонентів ефективної кібербезпеки. Дані системи збирають, об'єднують і аналізують великі обсяги даних з усіх програм, пристроїв, серверів і від усіх користувачів організації в реальному часі, об'єднуючи цей величезний масив даних у єдину уніфіковану платформу, SIEM забезпечують комплексний огляд захищеності організації та дають змогу швидко й ефективно виявляти, розслідувати інциденти з безпекою, та реагувати на них SOC. Дані рішення можуть допомогти організаціям будь-якого розміру забезпечити видимість захищеності, зводячи ризик порушень безпеки до мінімуму [1].

Ключові компоненти SIEM:

1. *Керування журналами.* Системи SIEM збирають і аналізують журнали в усій організації, зокрема із серверів, мережних пристроїв, брандмауерів, інших рішень безпеки та хмарних програм. Мета цього збирання даних є виявлення аномалії, які вказують на потенційну загрозу. Деякі рішення SIEM також інтегруються з сторонніми каналами інформації про загрози, щоб корелювати свої внутрішні дані безпеки з раніше визнаними сигнатурами та профілями загроз Інтеграція з каналами інформації про загрози в режимі реального часу дозволяє командам безпеки блокувати або виявляти нові типи сигнатур атак [1, 2].

2. *Кореляція подій.* Рішення SIEM є ефективними, оскільки вони об'єднують дані з різних систем підприємства. Наприклад, якщо є докази компрометації одного з облікових записів мережі, і разом з цим, в мережі з'явився підозрілий трафік, SIEM може зробити висновок, що дані дві події можуть бути пов'язаними, і згенерує повідомлення про подію для команди безпеки, щоб вони продовжили розслідування, пов'язане з цією подією. Кореляція подій допомагає виявити активність, яка сама по собі може здаватися безпечною, але у поєднанні з іншими активностями, може бути індикатором порушення. SIEM значно покращують середній

час виявлення та середній час реагування для команд безпеки, звільняючи їх від ручної роботи, пов'язаної з глибоким аналізом подій безпеки [1, 2].

3. Моніторинг інцидентів та сповіщення про порушення безпеки. SIEM об'єднує результати аналізу в дашбордах, де команди безпеки відстежують активність, сортують сповіщення, виявляють загрози та ініціюють заходи реагування або виправлення. Більшість панелей SIEM також містять візуалізацію даних у реальному часі, що допомагає аналітикам безпеки виявляти сплески або тенденції підозрілої активності. За допомогою заздалегідь визначених правил кореляції адміністратори можуть отримувати негайні сповіщення та вживати відповідних заходів для зменшення загроз, перш ніж вони перетворяться на більш серйозні проблеми безпеки [1, 2].

Також варто зазначити, що дані правила кореляції можливо завжди налаштувати для відповідних потреб адміністратора. Багато рішень SIEM також включають можливості автоматичного реагування. У певних випадках SIEM може автоматично вживати заходів на основі правил, визначених командою безпеки.

4. Управління відповідністю та звітність. Рішення SIEM є популярним вибором для організацій, які підлягають різним формам нормативного регулювання. Завдяки автоматизованому збиранню та аналізу даних, SIEM є цінним інструментом для перевірки даних про відповідність вимогам у всій бізнес-інфраструктурі. Рішення SIEM можуть генерувати звіти про відповідність вимогам PCI-DSS, GDPR, HIPAA, SOX та іншим стандартам відповідності в режимі реального часу, зменшуючи навантаження на управління безпекою та виявляючи потенційні порушення на ранній стадії, щоб їх можна було усунути [2].

Основні переваги SIEM:

– *розпізнавання загроз у реальному часі*. Завдяки дашбордам, які оновлюються у режимі реального часу, дозволяють вчасно помічати аномалії.

– *автоматизація на основі штучного інтелекту*. Використовуючи глибоке машинне навчання, яке автоматично навчається на основі поведінки мережі, ці рішення можуть обробляти складні протоколи ідентифікації загроз та реагування на інциденти за менший час, ніж фізичні команди [2].

– *підвищення організаційної ефективності*. Центральна панель управління забезпечує єдиний огляд системних даних, сповіщень та повідомлень, що дозволяє командам ефективно спілкуватися та співпрацювати під час реагування на загрози та інциденти безпеки [2].

– *виявлення складних і невідомих загроз*. Використовуючи інтегровані канали інформації про загрози та технологію штучного інтелекту, рішення SIEM можуть допомогти командам з безпеки ефективніше реагувати на широкий спектр кібератак [2].

– *проведення криміналістичних розслідувань*. Рішення SIEM дозволяють організаціям ефективно збирати та аналізувати дані журналів з усіх своїх цифрових активів в одному місці [2].

– *оцінка та звітність щодо дотримання вимог*. Рішення SIEM значно зменшують витрати ресурсів, необхідних для управління цим процесом, забезпечуючи аудит у режимі реального часу та звітність щодо дотримання нормативних вимог на запит у будь-який необхідний момент [2].

– *моніторинг користувачів і додатків*. Рішення SIEM відстежують всю мережну активність усіх користувачів, пристроїв і додатків, значно підвищуючи прозорість всієї інфраструктури та виявляючи загрози незалежно від того, де здійснюється доступ до цифрових активів і послуг [2].

SOAR – це комплекс рішень для автоматизації та оркестрації процесів реагування на кіберзагрози. Автоматизований підхід реалізується завдяки глибокій інтеграції захисних систем, оптимізації рутинних завдань безпеки та розробці сценаріїв ліквідації інцидентів відповідно до затверджених політик організації [3].

У великих організаціях центри безпеки використовують численні інструменти для відстеження та реагування на кіберзагрози, часто вручну. Таке ручне розслідування загроз призводить до загального уповільнення часу реагування на загрози. Платформи SOAR надають SOC центральну консоль, де вони можуть інтегрувати ці інструменти в оптимізовані робочі процеси реагування на загрози та автоматизувати низько-рівневі, повторювані завдання в цих робочих процесах. Ця

консоль також дозволяє SOC керувати усіма попередженнями про безпеку, що генеруються цими інструментами, в одному центральному місці.

Завдяки оптимізації сортування сповіщень та забезпеченню взаємодії різних інструментів безпеки, SOAR допомагають SOC скоротити середній час виявлення та середній час реагування, покращуючи загальний рівень безпеки. Згідно з останнім звітом IBM про вартість порушення безпеки даних, коротший цикл порушення безпеки даних пов'язаний з меншими витратами на його усунення. Порушення, вирішені менш ніж за 200 днів, обходяться компаніям в середньому на 1,02 млн доларів США менше, що становить різницю в 23% [4].

Ключові компоненти SOAR:

1. *Оркестрування безпеки*. Саме визначення «Оркестрування безпеки» означає те, у який спосіб дана платформа з'єднує і координує апаратними і програмними засобами в системі безпеки компанії. Операційні центри безпеки використовують різні рішення для моніторингу та реагування на загрози, такі як брандмауери, канали інформації про загрози та засоби захисту кінцевих точок. Навіть прості процеси безпеки можуть передбачати використання декількох засобів. За допомогою SOAR, SOC можуть об'єднати ці інструменти в узгоджені, повторювані робочі процеси з безпеки. SOAR використовують API, готові плагіни та спеціальні інтеграції для підключення засобів безпеки. Після інтеграції цих засобів SOC можуть координувати свою діяльність за допомогою плейбуків, можна сказати, інструкцій реагування [4].

2. *Автоматизація безпеки.* Рішення SOAR для забезпечення безпеки можуть автоматизувати низько рівневі, трудомісткі, повторювані завдання, такі як відкриття та закриття квитків підтримки, збагачення подій та пріоритизація сповіщень. SOAR також можуть запускати автоматизовані дії інтегрованих інструментів безпеки. Це означає, що аналітики безпеки можуть використовувати автоматизовані плейбуки, щоб об'єднати кілька інструментів і виконувати більш складні операції з автоматизації безпеки.

Наприклад, розглянемо, як платформа SOAR може автоматизувати розслідування щодо скомпрометованого ноутбука. Перша ознака того, що щось не так, надходить від рішення для виявлення та реагування на кінцевих точках (EDR), яке виявляє підозрілу активність на кінцевому хості. EDR надсилає сповіщення до SOAR, що змушує SOAR виконати заздалегідь визначений сценарій, той самий плейбук. Спочатку SOAR відкриває квиток на інцидент. Він доповнює сповіщення даними з інтегрованих джерел інформації про загрози та інших інструментів безпеки. Потім SOAR виконує автоматизовані відповіді, такі як запуск інструменту виявлення та реагування в мережі (NDR) для ізоляції кінцевої точки або запуск антивірусного програмного забезпечення для пошуку та знищення шкідливого програмного забезпечення. Нарешті, SOAR передає квиток аналітику з безпеки, який визначає, чи інцидент вирішено, чи потрібне втручання людини [4].

3. *Реагування на інциденти.* Можливості оркестрування та автоматизації безпеки SOAR системами дозволяють йому виконувати функції центральної консолі для реагування на інциденти безпеки. Аналітики з безпеки можуть використовувати SOAR для розслідування та вирішення інцидентів без необхідності переходу між різними інструментами. Як і платформи для аналізу загроз, SOAR системи виконують агрегацію метрик та сповіщень з зовнішніх джерел та інтегрованих інструментів безпеки в центральній панелі управління. Аналітики можуть співвідносити дані з різних джерел, відфільтровувати помилкові спрацьовування, визначати пріоритетність сповіщень та ідентифікувати конкретні загрози, з якими вони мають справу. Потім аналітики можуть реагувати, запускаючи відповідні сценарії дій. SOC також можуть використовувати інструменти SOAR для аудиту після інцидентів та більш проактивних процесів безпеки. Панелі інструментів SOAR можуть допомогти командам безпеки зрозуміти, як конкретна загроза проникла в мережу та як запобігти подібним загрозам у майбутньому. Аналогічно, команди безпеки можуть використовувати дані SOAR для виявлення непомічених поточних загроз та зосередити свої зусилля з пошуку загроз у правильних місцях [4].

Завдяки інтеграції засобів безпеки та автоматизації завдань платформи SOAR можуть оптимізувати типові робочі процеси з безпеки, такі як управління випадками, управління вразливостями та реагування на інциденти.

Переваги такої оптимізації включають:

– *обробка більшої кількості сповіщень за менший час*. SOAR можуть зробити сповіщення більш керованими завдяки централізації даних про безпеку, збагаченню подій та автоматизації відповідей. В результаті SOC можуть обробляти більше сповіщень, скорочуючи час реагування [4].

– *більш послідовні плани реагування на інциденти*. SOC можуть використовувати плейбуки SOAR для визначення стандартних, масштабованих робочих процесів реагування на інциденти для типових загроз [4].

– *покращене прийняття рішень*. SOC можуть використовувати інформаційні панелі SOAR, щоб отримати уявлення про свої мережі та загрози, з якими вони стикаються. Ця інформація може допомогти SOC виявляти помилкові спрацьовування, краще розставляти пріоритети сповіщень та вибирати правильні процеси реагування [4].

– *покращена співпраця між SOC*. SOAR централізують дані безпеки та процеси реагування на інциденти, щоб аналітики могли спільно працювати над розслідуваннями. SOAR також дозволяють SOC обмінюватися показниками безпеки із зовнішніми сторонами, такими як відділ кадрів, юридичний відділ та правоохоронні органи [4].

Аналіз показує, що SIEM-системи забезпечують централізований збір і кореляцію даних, подій безпеки, проте їх ефективність значною мірою залежить від правил виявлення. Тому необхідно з неабиякою уважністю та продуманістю підходити до проектування даних систем, щоб запобігти великій кількості хибно позитивних результатів. SOAR-системи, своєю чергою, вирішують проблему великої кількості одноманітних сповіщень шляхом автоматизації реагування, використовуючи різні методи. Тому в сучасних SOC найбільш ефективним є комплексне використання SIEM та SOAR як взаємодоповнюючих технологій.

1.2 Систематизація векторів кібератак та моделей загроз на основі бази знань MITRE ATT&CK

MITRE ATT&CK – це загальнодоступна база знань про тактики та методи зловмисників, що базується на реальних спостереженнях. База знань ATT&CK використовується як основа для розробки конкретних моделей загроз та методологій у приватному секторі, уряді та спільноті виробників продуктів і послуг у сфері кібербезпеки [5].

ATT&CK – це, в основному, база знань про методи та техніки, які використовують зловмисники. У ній ведеться розбивка та класифікація дій, спрямованих на атаку, які можуть бути використані проти певних платформ, таких як Windows. Основна увага приділяється не інструментам та шкідливому програмному забезпеченню, які використовують зловмисники, а тому, як вони взаємодіють із системами під час операції. ATT&CK організовує ці техніки в набір , щоб допомогти пояснити контекст техніки. Кожна техніка містить інформацію, яка є актуальною як для червоної команди або тестувальника на проникнення, для розуміння природи роботи техніки, так і для захисника, для розуміння контексту подій або артефактів, що генеруються технікою, яка використовується [6].

Тактика відображає причину застосування техніки ATT&CK. Тактика – це тактична мета супротивника для виконання дії. Тактика слугує корисною контекстуальною категорією для окремих технік і охоплює стандартні, більш високого рівня позначення дій, які супротивники виконують під час операції, такі як збереження, виявлення інформації, горизонтальне переміщення, виконання файлів та витік даних. Техніки відображають те, як супротивник досягає тактичної мети, виконуючи певну дію. Наприклад, супротивник може викрасти облікові дані, щоб отримати доступ до корисних облікових даних у мережі, які можна буде використовувати пізніше для горизонтального переміщення. Техніки також можуть відображати те, що супротивник отримує, виконуючи певну дію [6]. Взаємозв’язок між тактиками та техніками можна візуалізувати за допомогою матриці ATT&CK [5].

Таким чином, модель ATT&CK варто використовувати не лише як довідник технік та атак, а як основу для формування правил кореляції у SIEM-системах, та сценаріїв автоматичного реагування SOAR.

Якщо говорити про визначення вектору атаки у кібербезпеці, через призму ATT&CK, то вектором атаки можна назвати унікальним, впорядкованим ланцюжком технік, що зловмисник використовує для реалізації однієї, або декількох тактик, починаючи від початкового доступу, і закінчуючи досягненням фінальної мети. Такий ланцюжок часто відображає сам сценарій атаки [5]. Інтерпретація вектору атаки, як послідовності тактик, технік і процедур (TTP) створює основу для переходу від реактивного реагування до проактивного, захисту, орієнтованого на загрози. У цьому випадку захист будується не лише на реагуванні на вже реалізовані інциденти, а на попередньому аналізі можливих сценаріїв атак та їхніх характерних ознак.

Фреймворк ATT&CK є теоретичною основою для побудови ефективної моделі загроз для організації, забезпечуючи систематизований опис можливих дій противника. ATT&CK використовується для створення базової лінії профілю загроз, ідентифікуючи, які техніки найбільш імовірно будуть використані проти критичних активів організації. Поєднання ATT&CK з даними розвідки загроз дозволяє сформувати профіль найбільш релевантних загроз, зосереджуючись на типових TTP конкретних класів зловмисників, а не на абстрактному переліку потенційних атак [7].

Таким чином ATT&CK забезпечує єдину концептуальну основу для систематизації векторів кібератак і побудови моделей загроз, орієнтованих на реальні дії зловмисників. Структурований опис атак у вигляді ланцюжків TTP створює умови для подальшої реалізації цих моделей у процесах моніторингу, виявлення та реагування на інциденти інформаційної безпеки, зокрема в таких системах як SIEM та SOAR.

1.3 Порівняльний аналіз методів побудови SIEM/SOAR архітектури та аналіз рішень на ринку

На сьогоднішній день існує багато різних SIEM/SOAR рішень, орієнтованих на різні платформи та моделі розгортання. Можна виділити два основні середовища розміщення таких систем, локальне, за якого система розгортається в інфраструктурі організації, та хмарне, коли функціонування системи здійснюється на базі хмарного провайдера. Також варто зазначити, що існують гібридні підходи, що поєднують елементи обох моделей. У сучасних організаціях вибір моделі розгортання SIEM системи є важливим стратегічним рішенням, оскільки він безпосередньо впливає на вартість ефективність виявлення інцидентів.

Локальні SIEM системи передбачають розміщення програмного забезпечення та всієї інфраструктури безпосередньо в межах організації. Локальна система SIEM використовується, коли ваша організація зберігає конфіденційні дані локально або на місці. Це може бути пов’язано з багатьма факторами, такими як законодавчі вимоги або те, що ваша компанія не бажає надавати зайву інформацію хмарній системі SIEM. Багато команд з безпеки схильні використовувати локальну систему SIEM, оскільки вона надає їм повний контроль над своїми платформами. Локальна система SIEM означає, що ваша платформа залишається повністю під контролем вашої компанії. Щоб повною мірою скористатися перевагами цього вибору, необхідно мати глибокі знання як про компанію, так і про систему. Локальні рішення потребують значних початкових інвестицій у апаратне забезпечення, ліцензії та кваліфікований персонал, а масштабування таких систем часто пов’язане з додатковими фінансовими та технічними складнощами [8, 9].

Переваги локальних SIEM систем:

– контроль над даними: повний контроль і прозорість щодо місця зберігання даних та осіб, які мають до них доступ.

– індивідуальне налаштування: адаптація SIEM до конкретних потреб та безперебійна інтеграція з існуючою інфраструктурою.

– відповідність вимогам: забезпечення дотримання конкретних нормативних вимог, які можуть передбачати обмеження щодо хмарного зберігання даних.

Недоліки локальної SIEM систем:

– висока вартість: вимагає значних початкових інвестицій в апаратне та програмне забезпечення, а також в IT-персонал для розгортання, обслуговування та оновлення.

– масштабованість: масштабування ресурсів для задоволення зростаючих обсягів даних або потреб у безпеці може бути складним і дорогим.

– навантаження на управління: вимагає спеціалізованих IT-знань для постійного обслуговування, оновлення програмного забезпечення та управління інфраструктурою.

Хмарні SIEM рішення базуються на використанні інфраструктури хмарного провайдера, що дозволяє організаціям швидко впроваджувати систему без необхідності закупівлі та обслуговування власного апаратного забезпечення, через це вона зазвичай простіша у впровадженні та дешевша, ніж локальна система SIEM. Хмарна SIEM зазвичай оплачується за щомісячною передплатою з гнучкими варіантами контракту. Важливою перевагою хмарної SIEM є те, що вона усуває необхідність навчання співробітників тому, як максимально ефективно використовувати обрану платформу SIEM, і надає вашій організації доступ до професійних знань. Ще однією перевагою використання хмарної SIEM або SaaS є те, що вона автоматично оновлюється постачальником (наприклад, Microsoft), тоді як локальна SIEM оновлюється тільки тоді, коли IT-команда вирішує оновити програмне забезпечення, тому іноді вона може дуже швидко застаріти / відставати від поточної версії [9].

Переваги хмарних SIEM систем:

– нижча вартість: усуває необхідність попередніх інвестицій в обладнання та зменшує потреби в IT-персоналі для обслуговування та оновлення;

– масштабованість: легко масштабуйте ресурси для адаптації до змін обсягів даних та потреб у безпеці за допомогою моделі оплати за фактичне використання;

– швидше розгортання: швидко розпочніть роботу з мінімальним залученням IT-персоналу, часто за допомогою послуг на основі передплати;

– автоматичні оновлення: скористайтеся перевагами регулярних оновлень програмного забезпечення та інформації про загрози, які автоматично розгортаються постачальником;

– доступність: доступ до SIEM та даних безпеки з будь-якого місця, де є підключення до Інтернету.

Недоліки хмарних SIEM систем:

– проблеми безпеки даних: деякі організації можуть побоюватися довіряти конфіденційні дані сторонньому хмарному постачальнику;

– обмежена можливість налаштування: рівень налаштування може бути обмеженим порівняно з локальними рішеннями;

– залежність від постачальника: зміна постачальника може бути складною через проблеми з міграцією даних та потенційну несумісність API.

Однією з ключових відмінностей між локальними та хмарними SIEM є підхід до масштабування. Локальні системи, як правило, проектуються з урахуванням пікових навантажень, що може призводити до надлишкових витрат у разі змін обсягу подій безпеки. Натомість хмарні SIEM-рішення використовують гнучкі моделі масштабування, що дозволяють швидко збільшувати або зменшувати обчислювальні ресурси відповідно до актуальних потреб організації. Такий підхід є особливо ефективним у середовищах із динамічним зростанням телеметрії або гібридною інфраструктурою.

Аналіз моделей розгортання SIEM систем свідчить, що вибір між локальним та хмарним підходом залежить від контексту організації, зокрема чутливості даних, доступних ресурсів і вимог до масштабованості. Деякі організації обирають гібридний підхід, поєднуючи локальне та хмарне розгортання SIEM. Ця стратегія може забезпечити баланс між контролем даних та масштабованістю, але вимагає ретельного планування та інтеграції для забезпечення безперебійного моніторингу безпеки. Таким чином, можна зробити висновок, що якщо організація не дуже велика і фінансове становище не дозволяє розгортання локального рішення – то хмарне/гібридне рішення є ідеальним варіантом, звісно, якщо при цьому у хмару не будуть потрапляти конфіденційні дані.

Таблиця 1.1

Порівняльна характеристика моделей розгортання

Критерій	Локальна модель	Хмарна модель	Гібридна модель
Середовище розгортання	Інфраструктура організації	Інфраструктура хмарного провайдера	Поєднання локальної інфраструктури та хмарних сервісів
Контроль над даними	Повний контроль над зберіганням і доступом до даних	Дані зберігаються та обробляються у хмарі провайдера	Критичні дані зберігаються локально, інші – у хмарі
Початкові витрати	Високі капітальні витрати на обладнання та ліцензії	Відсутні або мінімальні початкові витрати	Частина витрат припадає на локальну інфраструктуру
Експлуатаційні витрати	Високі, потребують постійного обслуговування та персоналу	Нижчі, оплата за моделлю передплати або фактичного використання	Помірні, залежать від балансу між локальними та хмарними компонентами
Масштабованість	Обмежена фізичними ресурсами, складне та дороге масштабування	Еластична, з можливістю швидкого збільшення ресурсів	Частково еластична, потребує координації між середовищами
Швидкість впровадження	Повільна через необхідність розгортання інфраструктури	Висока, швидкий запуск без розгортання обладнання	Середня, залежить від складності інтеграції
Гнучкість налаштування	Висока, можливість глибокої кастомізації	Обмежена можливостями хмарної платформи	Середня, залежить від компонентів кожного середовища
Оновлення та підтримка	Виконуються внутрішньою IT-командою	Автоматично забезпечуються провайдером	Частково автоматизовані, частково потребують ручного втручання
Відповідність регуляторним вимогам	Найкраще підходить для жорстких вимог до локального зберігання даних	Може бути обмежена вимогами щодо передачі даних третій стороні	Дозволяє адаптуватися до регуляторних вимог
Типові сценарії використання	Організації з високими вимогами до контролю та конфіденційності	Організації, орієнтовані на масштабованість і швидке впровадження	Організації з гібридною інфраструктурою та змішаними вимогами

В результаті розгляду порівняльної характеристики моделей розгортання, було зроблено рішення для дослідження обрати саме гібридну модель. Дана модель дозволяє підтримувати оптимальний баланс між локальною інфраструктурою і хмарними сервісами, і є досить гнучким, доволі масштабованим та економічно вигідним рішенням для організацій, які вже мають локальну інфраструктуру, але прагнуть використовувати передові технології аналізу загроз.

На сьогоднішній день існує багато різних SIEM/SOAR-рішень, кожне з яких має свої чіткі переваги, і недоліки. Для визначення найбільш ефективного порівняємо декілька найпопулярніших рішень на ринку, проаналізуємо їхню архітектуру, моделі масштабування, фінансову доступність та специфіку інтеграції з різними середовищами. Це дозволить сформувати об'єктивну базу для вибору платформи, що найкраще відповідає обраній моделі розгортання.

Але для початку потрібно виставити певні вимоги, для гібридних середовищ, що поєднують локальну інфраструктуру Active Directory з хмарними сервісами, ключовими критеріями вибору SIEM-платформи є здатність до централізованого збору телеметрії з різних джерел, підтримка кореляції подій у контексті відомих сценаріїв атак, а також можливість інтерпретації інцидентів з урахуванням поведінкових моделей зловмисників. Також дуже важливими є інтеграція з сервісами управління ідентичностями та можливість аналізу подій автентифікації і доступу в межах єдиного інформаційного простору.

З урахуванням відповідних вимог, доцільним є розгляд хмарних SIEM рішень, орієнтованих на роботу в умовах розподіленої інфраструктури та тісну інтеграцію з хмарними сервісами ідентифікації та безпеки. У межах даного дослідження як базову платформу для побудови моделі гібридної системи виявлення інцидентів, за результатами попереднього аналізу моделей розгортання, було обрано Azure Microsoft Sentinel. Для обґрунтування вибору платформи було проведено порівняльний аналіз обраного рішення із провідними комерційними аналогами, такими як: Splunk Enterprise Security, IBM QRadar та популярними Open-Source системами: Elastic Security, Wazuh. Порівняння здійснювалося за ключовими критеріями, що є критичними для захисту гібридної інфраструктури. Порівняння наведено у табл. 1.2.

Аналіз представлених систем дозволяє розділити їх на три групи, кожна з яких має свої переваги та недоліки. Першу групу можна назвати комерційними гігантами, в цю групу входять Splunk ES та IBM QRadar. Вони мають колосальний аналітичний потенціал, глибоку експертизу, та потужні SOAR модулі, проте їх

впровадження супроводжується високим порогом входження, тривалим розгортанням та високою вартістю, також вони вимагають значних капітальних витрат на апаратне забезпечення локальних серверів та збереження логів.

Таблиця 1.2

Порівняльна характеристика популярних рішень

Критерій	Microsoft Sentinel	Splunk ES	IBM QRadar	Elastic Security	Wazuh
Архітектура та розгортання	Хмарна	Гібридна / Локальна / Хмарна	Переважно локальна або хмарна	Гібридна / Локальна / Хмарна	Локальна або приватна хмара
Модель ліцензування	Оплата за фактично проаналізований обсяг	За обсягом даних на добу або за обчислювальні потужності	За кількістю подій за секунду або потоків даних	Безкоштовна базова / Підписка за розширені комерційні функції	Повністю безкоштовна, оплата лише за підтримку
Інтеграція з AD	Безшовна через Azure Arc	Потребує налаштування Splunk Heavy Forwarders та універсальних агентів	Потребує використання WinCollect або сторонніх коллекторів	Через Elastic Agent гнучка конфігурація	Через Wazuh Agent
Вбудовані можливості SOAR	Глибока інтеграція Azure Logic Apps або готові плейбуки	Потужна оркестрація через окремий модуль Splunk SOAR	Інтегрована платформа SOAR на базі рішення IBM Resilient	Обмежена в базовій версії, автоматизація через сторонні інтеграції	Обмежена
Капітальні витрати на старті	Відсутні	Високі	Високі	Низькі/Середні	Низькі
Складність масштабування	Автоматичне хмарне масштабування без обмежень	Складне, потребує ручного додавання індексаторів та кластеризації	Обмежене ліцензією EPS та архітектурою апаратних засобів	Висока масштабованість, але потребує адміністрування кластера Elasticsearch	Середня, масштабується шляхом додавання нових Wazuh-серверів

Друга група це Open-Source, тобто рішення з відкритим кодом, ті самі Elastic Search та Wazuh. Вони приваблюють відсутністю ліцензійних платежів за базовий функціонал, демонструють високу ефективність збору логів, моніторингу цілісності файлів та аналізу конфігурації кінцевих точок, проте їхні SOAR можливості є досить обмеженими, а процес побудови та обслуговування великих кластерів обробки даних лягає безпосередньо на команду адміністраторів, що підвищує загальні витрати і навантаження на локальну інфраструктуру.

І третя група, це суто хмарні платформи, а саме Microsoft Sentinel. Він поєднує у собі переваги комерційного інструменту корпоративного рівня із гнучкістю хмарних технологій. Завдяки моделі SaaS, система розгортається за кілька хвилин і автоматично масштабується під будь-які обсяги даних, повністю нівелюючи витрати на локальну інфраструктуру.

Результати проведеного аналізу показують, що саме Microsoft Sentinel найкраще відповідає поставленим вимогам і є найбільш раціональним для захисту середовищ, що використовують локальну AD в поєднанні з хмарними сервісами. На відміну від Splunk та QRadar, Sentinel не потребує утримання локальних серверів зберігання логів, а модель оплати pay-as-you-go оптимізує операційні витрати. У порівнянні з Wazuh та Elastic, Sentinel надає повноцінний вбудований інструментарій SOAR на базі Azure Logic Apps, що дозволяє створювати складні крос-платформні сценарії автоматичного блокування загроз, а проста інтеграція з Windows-середовищем через Azure Arc та агенти AMA мінімізує затримки при зборі телеметрії з критичних вузлів інфраструктури, що забезпечує найменший показник середнього часу реагування на інциденти.

Висновки до першого розділу

У першому розділі було проведено аналіз сучасних підходів до моніторингу та реагування на інциденти інформаційної безпеки в різних IT-середовищах. Розглянуто основні принципи функціонування систем керування подіями та захисту інформації, а також систем оркестрування, автоматизації та реагування на інциденти, які є ключовими компонентами сучасних центрів операцій безпеки. Розглянуто принципи використання MITRE ATT&CK, де саме застосовується і для чого. Також було зроблено порівняння моделей розгортання SIEM систем, і зроблено висновок на користь гібридної моделі після чого проведено аналіз різних SIEM\SOAR рішень в результаті якого було обґрунтовано вибір хмарної SIEM\SOAR платформи Microsoft Sentinel.

РОЗДІЛ 2 ПРОЄКТУВАННЯ МОДЕЛІ ГІБРИДНОЇ ІНФРАСТРУКТУРИ ЗАХИСТУ

2.1 Формалізація об’єкта дослідження та загроз інформаційній безпеці гібридного середовища

Враховуючи результат аналізу, проведеного у попередньому розділі, а також значну поширеність змішаних IT інфраструктур, у межах даного дослідження об’єктом аналізу було обрано гібридне середовище, що поєднує локальний сегмент корпоративної мережі та хмарні сервіси безпеки.

Об’єктом дослідження є гібридне IT-середовище, що складається з локальної інфраструктури на базі служби каталогів Active Directory, яка містить у собі звичайні робочі станції персоналу та вебсервер з публічним вебдодатком, та хмарної платформи класу SIEM/SOAR, призначеної для централізованого моніторингу, аналізу та реагування на інциденти інформаційної безпеки Microsoft Sentinel.

Локальні доменні середовища на базі Active Directory залишаються однією з основних цілей кібератак, оскільки компрометація облікових записів домену дозволяє зловмиснику отримати розширений контроль над корпоративною інфраструктурою. Перехід до гібридної моделі управління ідентичностями, що поєднує локальний Active Directory з хмарними сервісами ідентифікації, суттєво змінює модель автентифікації та розширює поверхню атаки. Як тільки організація переходить на хмарні технології, поняття традиційного периметра мережі перестає існувати. Для IT-адміністраторів, які останні два десятиліття працювали з AD на місці, це поняття є величезним викликом. У гібридному середовищі ідентифікації організації тепер повинні бути готові захищатися від нескінченної кількості можливих точок входу. Перехід на гібридну модель також кардинально змінює модель дозволів, яку організації повинні забезпечити. У локальному середовищі досить легко контролювати, хто має фізичний доступ до контролерів домену, а загальні точки входу для управління чітко визначені та задокументовані. У гібридному середовищі AD ідентичності тепер також зберігаються в хмарі, що робить їх вразливими для зловживання з

боку будь-кого, хто має доступ до Інтернету. Дослідження також фіксують зростання кількості атак, спрямованих на компрометацію облікових записів Microsoft 365 та Microsoft Entra ID, зокрема через фішингові кампанії та зловживання механізмами синхронізації між локальним і хмарним середовищем [10].

Аналіз типових загроз для доменних середовищ узгоджується з тактиками та техніками, описаними у базі знань MITRE ATT&CK. До найбільш релевантних тактик для Active Directory належать Initial Access, Credential Access, Privilege Escalation, Lateral Movement та Persistence, які реалізуються через характерні техніки компрометації облікових даних і зловживання легітимними механізмами автентифікації [5]. Характеристика даних загроз наведена у табл. 2.1.

Таблиця 2.1

Характеристика загроз інформаційній безпеці Active Directory у гібридному середовищі

Клас загрози	Опис загрози в гібридному середовищі	Тактики MITRE ATT&CK	Типові джерела телеметрії
Компрометація облікових даних	Викрадення або зловживання обліковими записами домену та хмарних ідентичностей, зокрема через фішингові атаки та атаки на механізми автентифікації	Initial Access, Credential Access	Події автентифікації AD, журнали входу хмарного середовища
Ескалація привілеїв	Отримання підвищених прав у домені або хмарному середовищі шляхом зловживання ролями та дозволами	Privilege Escalation	Події змін ролей, події керування обліковими записами
Горизонтальне переміщення	Переміщення зловмисника між системами локального домену з використанням легітимних облікових даних	Lateral Movement	Події доступу до ресурсів, журнали протоколів аутентифікації
Закріплення в системі	Створення або модифікація облікових записів і політик для забезпечення тривалого доступу	Persistence	Події змін облікових записів і груп
Зловживання механізмами синхронізації	Використання взаємодії між локальним AD і хмарними сервісами для поширення компрометації	Persistence	Події синхронізації ідентичностей
Розширення поверхні атаки	Втрата класичного периметра через доступність сервісів ідентичності з мережі Інтернет	Initial Access	Журнали доступу до хмарних сервісів

Наведені загрози характерні для гібридних середовищ управління ідентичностями та узгоджуються з результатами досліджень ризиків безпеки Active Directory при інтеграції з хмарними сервісами [10].

У межах цієї роботи об’єктом дослідження виступає гібридна інфраструктура підприємства, яка інтегрує локальний технологічний контур із хмарними сервісами безпеки, аналітики та керування мережним трафіком і налічує у собі:

– *контролер домену DC-01*. Виконує роль центрального вузла управління ідентифікацією та доступом на базі служб каталогів Active Directory Domain Services. Він забезпечує автентифікацію та авторизацію користувачів і комп’ютерів, а також керує мережними сервісами DNS та DHCP;

– *вебсервер WEB-01*. Функціонує на базі середовища IIS та забезпечує роботу цільового вебдодатку. Додаток розроблено з використанням архітектурного шаблону ASP.NET Core Razor Pages. Цей сервер є критичним об’єктом, оскільки він опублікований у зовнішній мережі й безпосередньо взаємодіє з користувачами та потенційними зловмисниками;

– *робоча станція користувача WS-01*. Клієнтська машина під керуванням ОС Windows 10, яка включена до локального домену Active Directory та отримує динамічну IP-адресацію від DHCP-сервера на DC-01. Вона представляє кінцеві точки користувачів, які є найбільш вразливими до методів соціальної інженерії та запуску шкідливого ПЗ.

Основні компоненти для аналітики телеметрії від хмарної інфраструктури:

1. *Azure Arc та AMA*. Технологічний міст інтеграції, що дозволяє локальним серверам безпечно передавати події через Managed Identity без використання статичних компрометованих ключів автентифікації.

2. *Log Analytics Workspace*. Хмарне сховище великих даних, куди надходить уся сира телеметрія та журнали аудиту з локальних і хмарних джерел;

3. *Microsoft Sentinel*. Центральна хмарна система, яка відповідає за агрегацію, кореляцію та аналіз подій безпеки в реальному часі, а також за оркестрування та автоматизоване реагування на інциденти за допомогою Logic Apps;

Для хостингу вебдодатку у відкриту мережу інтернет буде використано Cloudflare. Він також відіграє роль хмарного зворотного проксі-серверу, який забезпечує базовий захист вебдодатку від DDoS-атак та виступає першим ешелоном виявлення і блокування аномальних HTTP/HTTPS запитів на мережному та прикладному рівнях [25].

Поверхня атаки досліджуваного середовища є неоднорідною та розділена на три основні вектори проникнення і зловмисного впливу:

– *вектор цільового вебдодатку*. Спрямований на компрометацію WEB-01 через експлуатацію архітектурних та програмних вразливостей коду вебдодатку;

– *вектор інфраструктури ідентичностей*. Спрямований на компрометацію служб каталогів Active Directory на DC-01 шляхом підбору, викрадення або несанкціонованого підвищення привілеїв облікових записів користувачів;

– *вектор кінцевих точок*. Спрямований на компрометацію робочої станції WS-01 через доставку та виконання шкідливого коду у внутрішньому периметрі мережі.

З метою систематизації кібератак та формування правил детектування і сценаріїв реагування, виділено 5 фундаментальних класів загроз, які наведено у табл. 2.2.

Таблиця 2.2

Фундаментальні класи загроз досліджуваного проєкту

Клас загрози	Опис загрози	Тактики MITRE ATT&CK	Критичні індикатори
Brute-Force	Масовий автоматизований перебір паролів до облікових записів домену Active Directory.	Credential Access	Аномальний сплеск подій невдалих локальних спроб входу в систему з однієї IP-адреси за короткий проміжок часу.
Account Compromise & Privilege Abuse	Успішний несанкціонований вхід до системи під виглядом легітимного користувача (особливо адміністратора домену) після серії невдалих спроб автентифікації.	Initial Access	Серія помилок автентифікації, яка завершується успішною автентифікацією для того самого облікового запису з ідентичної IP-адреси.

Закінчення таблиці 2.2

Клас загрози	Опис загрози	Тактики MITRE ATT&CK	Критичні індикатори
Web Application Attacks	Проведення атак класу SQL Injection та Cross-Site Scripting через надсилання зловмисних корисних навантажень	Initial Access	Фіксація спеціалізованим конвеєром логування вебдодатку або WAF Cloudflare специфічних сигнатур у HTTP Body або параметрах URI запиту.
Malware Execution	Попадання шкідливого коду або вірусів у файлову систему робочої станції користувача WS-01 та спроба їх запуску.	Execution	Спрацювання локальних сигнатурних або евристичних модулів захисту антивірусного ПЗ з подальшою генерацією критичного сповіщення в системі моніторингу кінцевих точок Microsoft Defender.
Persistence via Backdoor Accounts	Створення несанкціонованого облікового запису користувача локального домену з метою постійного утримання доступу до ресурсів мережі у випадку зміни паролів основних легітимних акаунтів.	Persistence	Генерація в журналах безпеки Active Directory подій створення нового користувача або модифікації груп безпеки

Проаналізовані класи загроз дають чітке підґрунтя для подальшого конструювання логіки пошуку загроз і виділення основних джерел телеметрії для правил їх збору.

2.2 Синтез структурної схеми системи виявлення інцидентів на базі платформи Microsoft Sentinel

У межах даного дослідження, з урахуванням результатів аналізу класів загроз, система виявлення інцидентів інформаційної безпеки розглядається як гібридна система, у якій події безпеки збираються з локальної інфраструктури організації та аналізуються за допомогою хмарної SIEM платформи Microsoft Sentinel.

Концептуальна структура схеми включає декілька логічних рівнів, кожен з яких виконує окрему функцію у процесі забезпечення інформаційної безпеки. До таких рівнів належать:

1. *Рівень джерел телеметрії;*

2. *Рівень збору та передачі подій;*
3. *Аналітичний рівень;*
4. *Рівень реагування на інциденти;*

Почнемо з рівня джерел телеметрії. Телеметрія безпеки – це безперервний збір і передача в режимі реального часу даних, що мають значення для безпеки, з усієї цифрової інфраструктури організації. Ці дані включають журнали, події, сповіщення та метадані з кінцевих точок, мереж, хмарних служб, постачальників ідентифікаційних даних та додатків [11]. У нашому дослідженні, рівень джерел телеметрії охоплює локальні компоненти корпоративної інфраструктури, зокрема служби каталогів Active Directory, серверні та користувацькі системи, а також мережні сервіси. Саме на цьому рівні формуються первинні події безпеки, пов'язані з автентифікацією, авторизацією, доступом до ресурсів і змінами облікових записів, та формується базова видимість стану безпеки системи.

Другий рівень – це збір, індексація та нормалізація даних. Як зазначено в описах сучасних SIEM архітектур, події спочатку збираються, потім агрегуються та нормалізуються для подальшого аналізу; ці функції відповідають за перетворення сирих даних у форму, придатну для кореляції та виявлення аномалій [12]. У нашому дослідженні, цей рівень також реалізує захищену взаємодію між локальною інфраструктурою та хмарним середовищем

Третій рівень – аналітичний рівень системи виявлення інцидентів, який відповідає за кореляцію подій безпеки та інтерпретацію телеметрії у контексті можливих сценаріїв атак. На цьому рівні окремі події не розглядаються ізольовано, а поєднуються у логічні послідовності дій, що дозволяє встановити взаємозв'язок між поведінкою суб'єктів, часовими характеристиками та контекстом доступу. Кореляція може здійснюватися на основі заздалегідь визначених правил, що описують відомі тактики, техніки і процедури зловмисників, а також на основі поведінкових моделей, які виявляють відхилення від нормальної активності користувачів і систем. Такий підхід забезпечує виявлення як відомих сценаріїв атак, так і аномальної поведінки, що не відповідає типовим шаблонам, та створює основу для формування інцидентів або оцінки рівня ризику [13].

Четвертий рівень – рівень реагування на інциденти, який забезпечує підтримку процесів реагування формуючи сповіщення інцидентів та рекомендацій щодо подальших дій. У сучасних архітектурах цей рівень також може включати інтеграцію з механізмами SOAR для автоматичного виконання певних реакційних дій при підтвердженні інцидентів [4].

Таким чином, запропонована структурна схема системи виявлення інцидентів відображає логіку взаємодії між локальними джерелами подій та хмарною SIEM-платформою Microsoft Sentinel у гібридному середовищі. Узагальнена архітектура системи подана у вигляді структурної схеми, яка ілюструє основні функціональні рівні та потоки даних між ними і зображена на рис. 2.1.

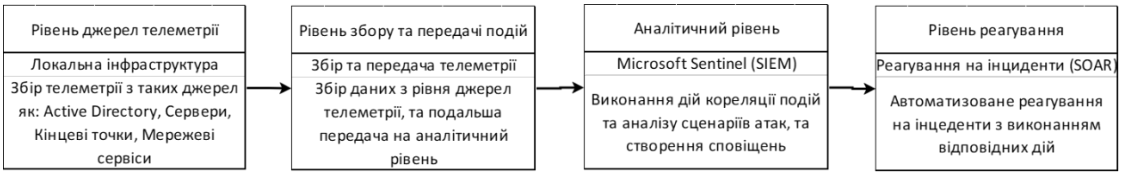


Рисунок 2.1 – Структурна схема системи

Після того, як ми розібралися зі структурною схемою гібридної системи виявлення інцидентів, доцільно буде розглянути основні підходи інтеграції локальної інфраструктури з хмарною SIEM платформою Microsoft Sentinel. У гібридних середовищах саме механізм інтеграції визначає спосіб доставки телеметрії, рівень ізоляції локального сегмента, масштабованість системи та складність її подальшої експлуатації. Microsoft Sentinel не здійснює збір подій безпосередньо, а використовує можливості платформи Azure Monitor та Log Analytics для приймання та завантаження даних у систему й подальшої обробки.

У офіційній документації Microsoft описано декілька базових підходів до інтеграції, які відрізняються архітектурними принципами та способами доставки. До основних підходів інтеграції належать:

– інтеграція на основі використання агентів збору даних за допомогою Azure Monitor Agent;

– інтеграція локальних ресурсів через сервіс Azure Arc;

– централізований збір подій із використанням механізму Windows Event Collection;

– використання захищених мережних з’єднань як допоміжного елемента інтеграції.

Базовим і найбільш універсальним підходом до інтеграції локальних серверів і систем із Microsoft Sentinel є використання AMA. AMA є рекомендованим агентом для збору даних, що замінив попереднього агенту Log Analytics, і дозволяє централізовано збирати журнал подій, включно з Windows Security Events, Syslog, CEF та іншими типами даних, які можуть надходити з локальних і хмарних джерел. Дані, які збирає агент, передаються до Log Analytics Workspace, що використовується Microsoft Sentinel для аналізу та кореляції подій. Основою для цього є можливість задавати DCR, які визначають, які дані і як саме збираються з агентів. Рішення з використанням агентів є гнучким і масштабованим, дозволяє стандартизувати процес збору даних, налаштовувати фільтрацію та обробку подій на джерелі, а також працювати в різних середовищах – від окремих серверів до великих розподілених інфраструктур [14; 15].

Для серверів і систем, що працюють поза межами Azure, але потребують централізованого моніторингу та взаємодії з платформою Microsoft Sentinel, застосовується сервіс Azure Arc. Azure Arc дозволяє зареєструвати локальні або сторонні ресурси, у тому числі сервери, як «керовані об’єкти» Azure, після чого на такі ресурси встановлюється Azure Monitor Agent як додаткове розширення до Azure Connected Machine Agent для збору подій. Після реєстрації через Azure Arc агенти можуть відправляти дані безпосередньо до Log Analytics workspace, що пов’язано з Sentinel, що забезпечує узгоджений потік телеметрії з гібридних систем [16].

Цей підхід дозволяє зменшити залежність від традиційних мережних механізмів передачі даних, забезпечити централізоване керування і розгортання агентів, а також краще інтегрувати моніторинг локальних ресурсів у загальну SIEM-архітектуру.

Інший підхід інтеграції локальної інфраструктури полягає у використанні механізму Windows Event Collection (WEC) для централізованого збору подій безпеки з доменних контролерів та інших серверів. У цій архітектурі локальні журнали спочатку передаються на спеціальний сервер-колектор, після чого агреговані журнали

передаються до Microsoft Sentinel через Azure Monitor Agent або через Azure Arc із установленими агентами [17].

Цей підхід дозволяє зменшити кількість агентів у локальній інфраструктурі, централізувати точку збору і фільтрації подій, але створює залежність від сервера-збирача та може мати обмеження щодо масштабованості у великих або динамічних середовищах. Централізований збір подій через WEC є доцільним у середовищах із чітко визначеними структурами, де можна централізувати збір подій.

Вибір конкретного підходу має базуватися на архітектурі локальної мережі, вимогах безпеки та ресурсах організації. Порівняльна характеристика підходів інтеграції локальної інфраструктури з Microsoft Sentinel наведена у табл. 2.2.

Таблиця 2.2

Порівняльна характеристика підходів інтеграції локальної інфраструктури з Microsoft Sentinel

Критерій	AMA	Azure Arc + AMA	WEC
Архітектурний принцип	Агентний збір подій безпосередньо з кожного вузла	Агентний збір через керовані Arc-ресурси	Централізований збір подій через сервер-колектор
Типи підтримуваних джерел	Windows, Linux, Syslog, CEF	On-prem та сторонні хмарні сервери	Переважно Windows-середовище
Масштабованість	Висока, придатна для розподілених середовищ	Висока, з централізованим керуванням	Обмежена масштабованістю збирача
Складність адміністрування	Середня (керування агентами та DCR)	Середня (керування Arc та агентами)	Нижча кількість агентів, але залежність від WEC
Залежність від топології мережі	Низька	Низька	Середня
Придатність для гібридного середовища	Висока	Дуже висока	Середня
Типові сценарії використання	Універсальний підхід для збору телеметрії	Гібридні та розподілені інфраструктури	Великі доменні середовища з централізацією логів

Порівняння основних підходів інтеграції свідчить, що агентні та Azure Arc-орієнтовані моделі забезпечують найвищий рівень гнучкості, відмовостійкості і масштабованості для гібридних середовищ. На відміну від класичних централізованих або без агентних методів збору логів, цей тандем дозволяє реалізувати децентралізоване зняття телеметрії безпосередньо в точці її генерації та її пряме безпечне

транспортування до репозиторію хмарної SIEM – Log Analytics Workspace. Для детального обґрунтування доцільності впровадження обраної моделі архітектури доцільно розглянути її ключові переваги у порівнянні з альтернативними методами інтеграції:

Можливість детального керування потоками даних за допомогою DCR. Використання сучасного AMA у поєднанні з правилами збору даних DCR надає можливість гнучкої фільтрації подій на рівні самого джерела. Замість надлишкового пересилання всього масиву журналів Windows Security Events, що призводить до нераціонального навантаження на канали зв'язку та суттєвого подорожчання вартості хмарного зберігання, DCR дозволяє ізолювати лише критично важливі ідентифікатори подій, наприклад, Event ID 4624 – успішна автентифікація, Event ID 4625, 4771 – невдала спроба входу для виявлення атак типу Brute-Force.

Централізація та безпека автентифікації через Azure Arc. Традиційні підходи збору логів із локальних середовищ вимагають використання статичних ідентифікаторів та секретних ключів робочої області, компрометація яких дозволяє зловмиснику здійснити атаку типу Log Spoofing або спричинити відмову в обслуговуванні. Інтеграція через Azure Arc нівелює цей ризик, оскільки кожній локальній машині присвоюється керована ідентичність Azure. Автентифікація та шифрування трафіку в процесі передачі логів у хмару здійснюються на основі сертифікатів безпеки в межах моделі Zero Trust, а управління самими агентами відбувається централізовано через Azure ARM.

Таким чином, архітектурне рішення, засноване на поєднанні Azure Arc та AMA, мінімізує операційні ризики, забезпечує стійкість інфраструктури моніторингу до масштабування та гарантує цілісність і незмінність телеметрії, що є критично важливим для якісного проведення розслідувань інцидентів. Отримані результати аналізу створюють основу для формування практичної моделі гібридної системи моніторингу безпеки, яка розглядається у наступному розділі. Також використовуючи результати аналізу, можна сформувати більш чітку архітектурну схему процесу збору-передачі і обробки телеметрії при використанні обраного підходу

інтеграції локальної інфраструктури. Архітектурна схема процесу збору-передачі і обробки телеметрії зображена на рис. 2.2.

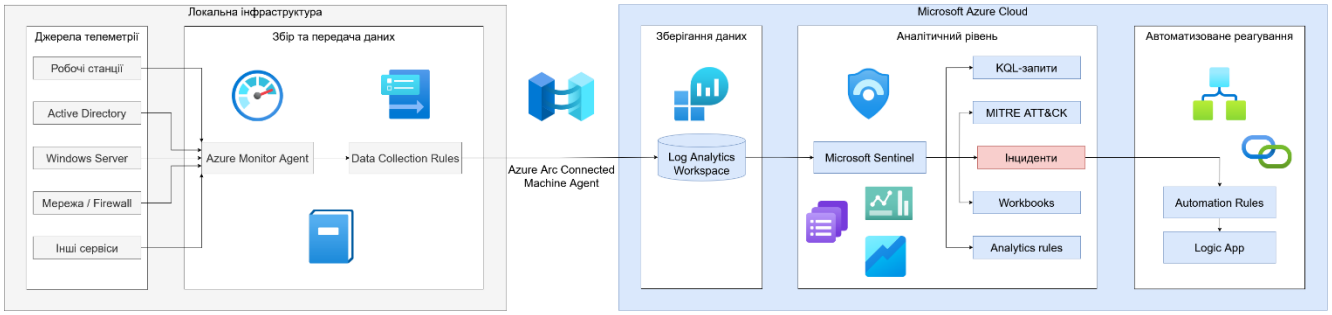


Рисунок 2.2 – Архітектурна схема процесу збору-передачі і обробки телеметрії

Для забезпечення балансу між повнотою охоплення векторів атак та оптимізацією витрат на зберігання даних у Log Analytics Workspace, розроблено точкові правила збору даних. Замість суцільного імпорту журналів операційної системи, що створює надлишковий шум та навантаження, правила DCR налаштовані на передавання лише тих ідентифікаторів подій, які мають критичне значення для виявлення тактик фреймворку MITRE ATT&CK. В загальному для дослідження було виділено 7 категорій, кожна з яких наведена у табл.2.3.

Таблиця 2.3

Специфікація джерел телеметрії

№	Категорія	Event ID	Опис
1	Автентифікація та доступ	4624	Успішні спроби входу в систему
		4625	Невдалі спроби входу в систему
		4771	Невдала попередня автентифікація Kerberos
		4776	Перевірка облікових даних контролером домену
2	Керування обліковими записами	4720	Створення користувачів
		4722	Активація користувачів
		4725	Вимкнення користувачів
		4738	Модифікація властивостей користувачів
3	Аудит процесів	4688	Створення нового процесу
4	Антивірусний захист (Windows Defender)	1116	Виявлення шкідливого ПЗ
		1117	Реагування на шкідливе ПЗ
5	Самозахист системи	1102	Очищення журналу аудиту безпеки Windows
6	Системні служби	7045	Встановлення або реєстрація нової служби у системі
7	Прикладний рівень	0	Специфічні події безпеки, що генеруються цільовим вебдодатком

Кожна категорія містить у собі конкретну аналітичну цінність, і відіграє важливу роль у подальшій роботі SIEM/SOAR платформи (з табл. 2.2):

1. Виявлення атак типу Brute-Force та Password Spraying. Аналіз співвідношення подій дозволяє виявити аномальну активність у позаробочий час або спроби підбору облікових даних локальних користувачів.

2. Моніторинг тактики Persistence та дозволяє виявити несанкціоноване створення зловмисником нових облікових записів або їхнє додавання до груп із підвищеними привілеями.

3. Виявлення тактик Execution та дозволяє аналізувати запуск підозрілих скриптів за допомогою PowerShell або CMD, виконання нетипових системних утиліт та відстежування ланцюжків виконання процесів.

4. Фіксація фактів активності експлойтів, троянських програм чи вірусів на кінцевих точках, а також перевірка статусу реагування антивірусу.

5. Виявлення тактики Defense Evasion. Спрацювання цього лічильника є прямим індикатором ворожих дій зловмисника, спрямованих на приховування слідів своєї присутності.

6. Моніторинг спроб закріплення в системі. Хакери часто реєструють власні утиліти чи шкідливі інструменти як системні служби для забезпечення їх автоматичного запуску після перезавантаження робочої станції.

7. Забезпечення моніторингу вебсервера. Дозволяє корелювати системні події ОС із подіями всередині самого додатка для виявлення специфічних веб-атак.

Відокремлені категорії дозволять якісно слідкувати за найважливішими подіями у досліджуваному середовищі, і дають можливість демонстрації можливостей автоматизованого реагування Microsoft Sentinel.

Для ефективного моніторингу ситуації у мережі аналітики з кібербезпеки використовують дашборди, у середовищі Microsoft Defender та Azure дані їх також називають книгами (workbooks). Для нашої мережі для ефективного моніторингу аномальної активності, можна виділити наступні групи метрик моніторингу:

1. *Метрики автентифікації та контролю доступу*. Дана група метрик агрегує телеметрію служб автентифікації для виявлення аномальної поведінки користувачів. Візуалізація включає:

– динаміку невдалих спроб входу: Відображається у вигляді часового графіку. Раптові сплески сигналізують про проведення активної фази атак типу Brute-Force.

– коефіцієнт успішності автентифікації: Графічне співвідношення успішних та невдалих вхідних запитів, що дозволяє оцінити загальний рівень активності в мережі та відсіяти випадкові помилки легітимних користувачів.

– аудит життєвого циклу облікових записів: Хронологічний перелік подій керування користувачами. Дозволяє миттєво фіксувати створення несанкціонованих облікових записів або раптову зміну прав доступу до критичних ресурсів.

2. *Метрики безпеки вебсервера та прикладного рівня*. Компонент призначений для моніторингу зовнішнього периметра інфраструктури та фіксації спроб компрометації вебдодатку. До цієї групи входять:

– таймлайн критичності загрози додатків: Графік розподілу подій безпеки із вебдодатку, розбитий за рівнями критичності для вибору пріоритету реагування аналітиків.

– класифікатор зафіксованих веб-атак: Статистичний аналіз виявлених спроб впровадження SQL-ін’єкцій, XSS або інших типів веб-атак.

– дайджест інцидентів: Зведена таблиця останніх зафіксованих атак за останні 24 години із відображенням відповідних індикаторів компрометації, таких як підозрілі IP-адреси, типи HTTP-запитів та атаковані URI.

3. *Метрики стану кінцевих точок та антивірусного захисту*. Дашборд орієнтований на контроль цілісності хостів гібридної інфраструктури та моніторинг ефективності антивірусних засобів. Група містить:

– статус активності шкідливого ПЗ: Візуалізація кількості та типів зафіксованих запусків експлойтів.

– індикатор очищення журналів: Візуалізація кількості очищень журналу.

– реєстратор нових системних служб: Лінійний графік появи нових служб у системі, який використовується для проактивного пошуку загроз та виявлення прихованих механізмів закріплення зловмисників.

2.3 Проєктування мережної топології та інтеграції з хмарним середовищем

Для дослідження було прийнято рішення побудови простої віртуальної мережі у середовищі Virtual Box, у якій знаходиться один контролер домену на базі Windows Server 2019, один вебсервер також на базі Windows Server 2019, та одна клієнтська машина з операційною системою Windows 10. На вебсервері, за допомогою Internet Information Services (IIS) буде розгорнуто вебдодаток на базі ASP.NET. Цього буде достатньо для проведення дослідження роботи агентів передачі записів журналу подій, їх обробки Microsoft Sentinel, та використання плейбуків для автоматичного реагування на підозрілі дії, та надсилання автоматичного сповіщення на різні платформи.

У середовищі Virtual Box буде створено просту NAT мережу для розміщення у ній робочих машин. При створенні потрібно враховувати, що Virtual Box виділяє перші три адреси пулу для функціонування шлюзу за замовчуванням, DNS сервера та DHCP сервера. У нашому випадку, DHCP сервер Virtual Box не є необхідним, тому третя адреса звільняється.

Таблиця 2.4

Параметри адресації у мережі

Хост/Мережа	Тип адреси	Адреса	Маска	Префікс
Network	-	192.168.20.0	255.255.255.240	/28
R-01 (VB)	статична	192.168.20.1	255.255.255.240	/28
SW-01 (VB)	статична	192.168.20.2	255.255.255.240	/28
DC-01	статична	192.168.20.3	255.255.255.240	/28
WEB-01	статична	192.168.20.4	255.255.255.240	/28
WS-01	динамічна	DHCP		

Як вже було згадано раніше на контролері домену DC-01 під час базової конфігурації будуть встановлені ролі для забезпечення базового функціонування локальної інфраструктури. Буде проведено базову конфігурацію адресації та встановлено такі ролі як AD Domain Service, DNS та DHCP. Опис даних ролей наведено у табл. А.1.

Для мережної інфраструктури буде використано публічний домен kb221shkyu.pp.ua з безкоштовної доменної зони pp.ua. Для локальної мережі використовується неактивний під-домен публічного домену internal.kb221shkyu.pp.ua. Конфігурація WEB-01 є аналогічною до DC-01 і включає зміну імені комп'ютера, базову конфігурацію параметрів мережного інтерфейсу, а також введення сервера до складу створеного домену Active Directory. Оскільки WEB-01 виконує функцію вебсервера, для забезпечення зручного хостингу та менеджменту вебдодатками на ньому розгортається роль IIS. Опис даної ролі наведено у табл. А.1.

Конфігурація WS-01 буде обмежено підключенням її до створеного домену і отриманням IP-адреси від DHCP-серверу.

Ціллю створення даного стенду є показ впровадження гібридної моделі моніторингу та реагування на інциденти безпеки, та їх дослідження процесів їх аналізу і обробки. Для реалізації цієї архітектури в мережу інтегруються об'єкти моніторингу - робочі станції, сервери а також окремі сервіси. З метою розширення вектору моніторингу, на базі вебсервера WEB-01 додатково розгортається вебдодаток, який виступатиме окремим об'єктом аналізу. Вебдодаток розроблено як тестову модель інтернет-магазину на базі фреймворку ASP.NET core. Для збереження та обробки даних використовується локально розгорнута система управління базами даних, далі СУБД, MS SQL Server.

Планується проведення досліджень обробки і реагування на інциденти безпеки створені в результат атак на цільовий додаток. З метою дослідження методів реагування на інциденти в умовах реальних загроз, додаток буде відкрито для доступу до загальної мережі інтернет. Публікація сервісу та його захист здійснюються за допомогою хмарної платформи Cloudflare. Саме для забезпечення коректної маршрутизації та інтеграції з хмарним провайдером на початковому етапі було зареєстровано безкоштовне публічне доменне ім'я в зоні pp.ua.

Як вже було вирішено в результаті попереднього аналізу, інтеграція локального середовища з хмарним буде виконано за допомогою Azure Arc + AMA. Для автоматичного підключення даних засобів, на порталі Azure, для AD середовищ, для цього рекомендують використовувати GPO, так це і будемо реалізовувати.

Архітектурна діаграма мережі зображена на рис. 2.3.

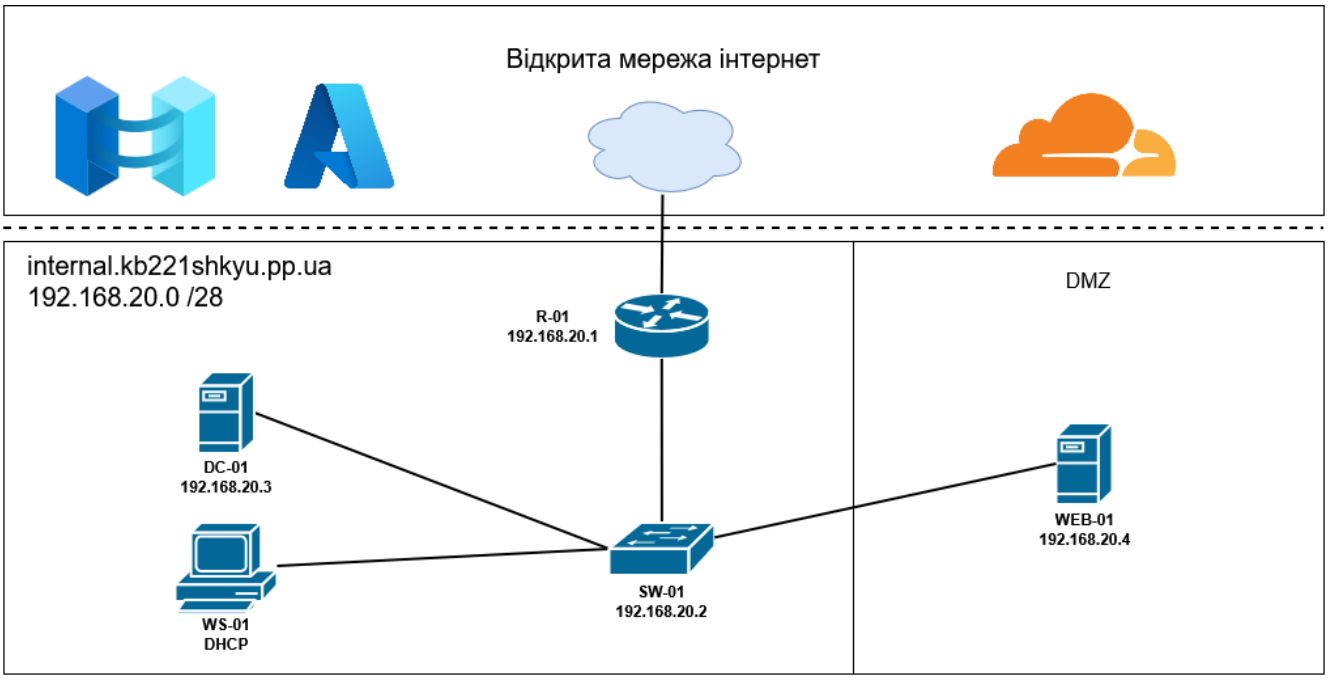


Рисунок 2.3 – Архітектурна діаграма досліджуваної інфраструктури

Висновки до другого розділу

У другому розділі було обґрунтовано вибір гібридної моделі інфраструктури на базі платформи Microsoft Sentinel. Проведено аналіз типових загроз локальному середовищу на базі AD в умовах гібридної інфраструктури. Створено структурну схему, яка складається з чотирьох рівнів, і описує процес роботи системи виявлення інцидентів. Також було проаналізовано основні підходи до інтеграції локальної мережі з хмарною SIEM платформою, та зроблено висновок, що використання агентних і Arc-орієнтованих механізмів інтеграції забезпечує необхідний рівень масштабованості та централізованого аналізу подій безпеки. Було проаналізовано основні класи загроз досліджуваної інфраструктури, після чого, враховуючи проаналізовані класи, сформовано основний список джерел відповідної телеметрії для подальшого ефективного реагування і дослідження. Спроектовано загальну архітектуру з урахуванням обраних технологічних рішень.

РОЗДІЛ 3 ПРОГРАМНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ СИСТЕМИ

3.1 Реалізація компонентів моніторингу та інтеграція різнорідних джерел даних у єдиний інформаційний простір

Налаштування DC-01 починається зі зміни його імені комп'ютера на відповідне, а саме dc-01, після чого відбувається налаштування IP-адресації на мережному інтерфейсі, встановлюємо статичну адресу яка вказана у таблиці 3.1. При подальшому налаштуванні мережного інтерфейсу як первинний DNS-сервер встановлюється його власна IP-адреса, а як вторинний адреса його loopback інтерфейсу. Це необхідно для того, щоб DC-01 правильно розв'язував доменні імена всередині нашого домену, для правильного функціонування AD DS. Після чого встановлюємо на сервер роль контролера домену, використавши при налаштуванні відповідний домен.

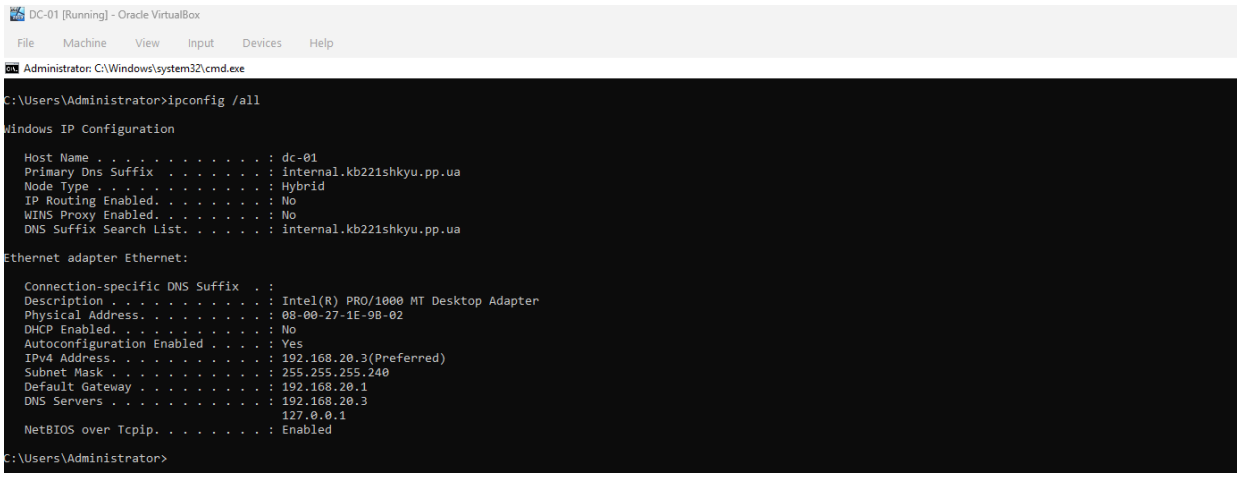


Рисунок 3.1 – Результат початкової конфігурації DC-01

Наступним кроком є розгортання та налаштування служби DHCP на нашому контролері. Для цього на сервері необхідно встановити роль DHCP Server на наш сервер, і після чого через DHCP Manager виконується конфігурація області. Задаються такі базові параметри, як:

- діапазон адрес доступних для оренди;
- діапазон статичних-зарезервованих адрес;
- IP-адрес шлюзу за замовчуванням;

– IP-адрес DNS-серверу;

– доменне ім’я у мережі.

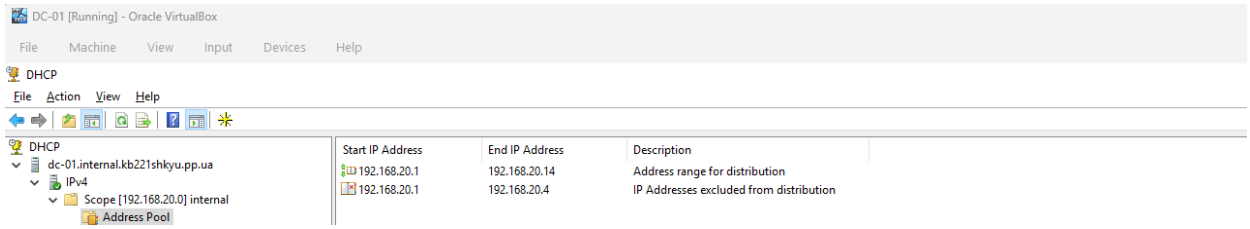


Рисунок 3.2 – Налаштування діапазонів адрес у адресного пулу серверу

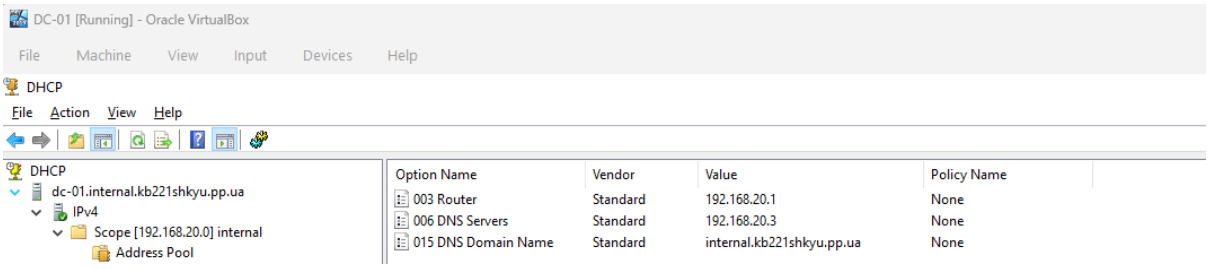


Рисунок 3.3 – Налаштування параметрів адресації клієнтів DHCP-сервера

Роль DNS-сервера інтегрується та налаштовується автоматично під час розгортання доменних служб AD DS. Майстер конфігурації самостійно створює необхідні зони прямого перегляду та інтегрує їх в AD. Після завершення процесу інсталяції додаткова ручна конфігурація базових зон не потрібна.

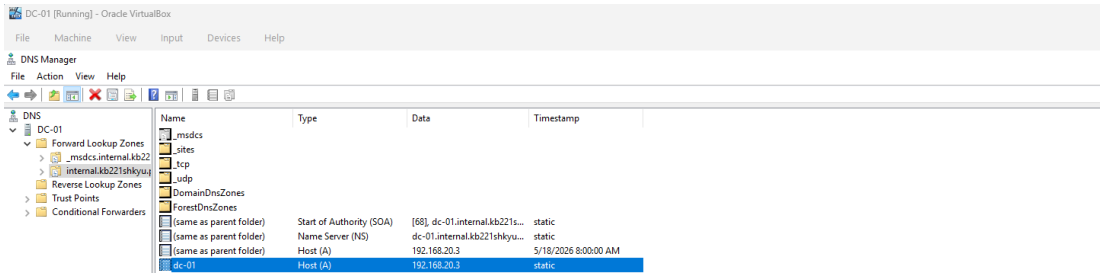


Рисунок 3.4 – Автоматично створення зона прямого перегляду на DC-01

На цьому базова конфігурація DC-01 завершена, наступним у списку є веб-сервер WEB-01. Його первинне налаштування є аналогічним до DC-01 і включає зміну імені комп’ютера, базову конфігурацію параметрів мережного інтерфейсу, а також введення сервера до складу створеного домену Active Directory.

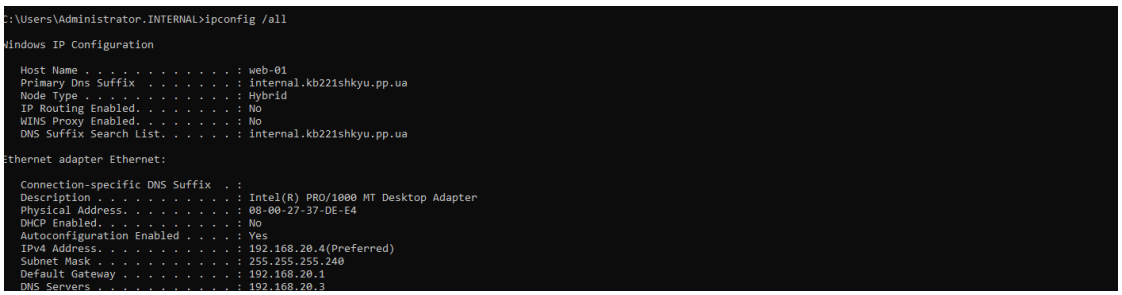


Рисунок 3.5 – Базові налаштування на WEB-01

Оскільки WEB-01 виконує функцію вебсервера, для забезпечення зручного хостингу та менеджменту вебдодатками на ньому розгортається роль IIS.

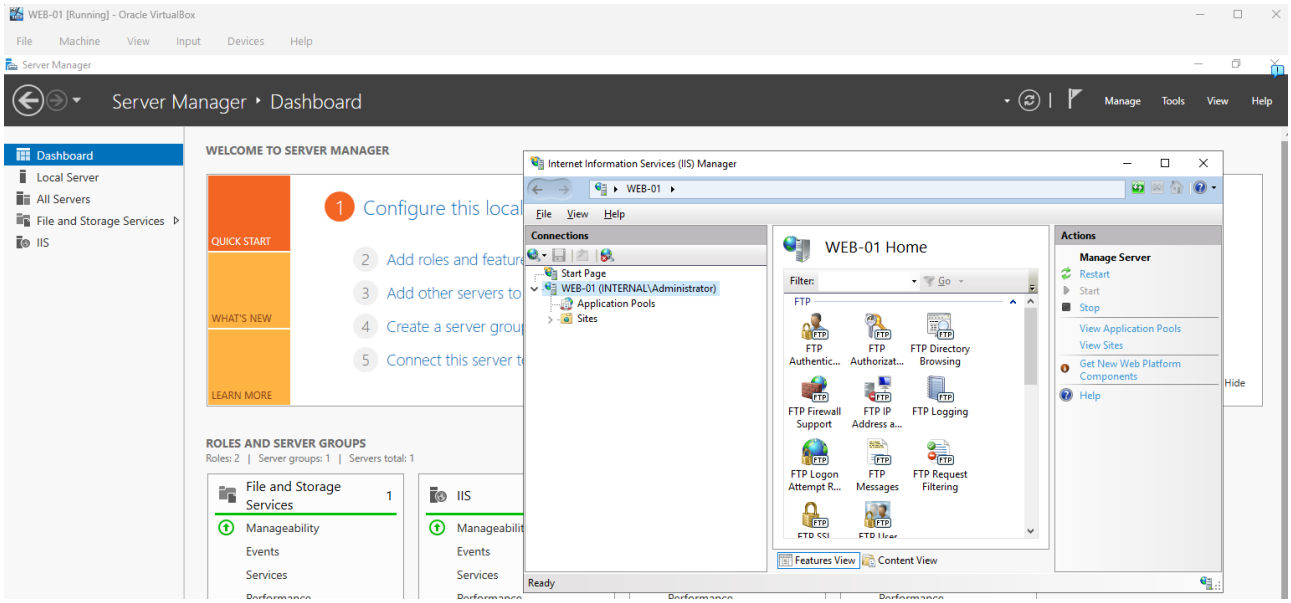


Рисунок 3.6 – Результат додавання ролі на WEB-01

Тепер, коли у нашій мережі є основні сервери, можна додавати робочі станції звичайних користувачів, на нашому стенді буде всього одна така станція – WS-01. На відміну від серверів, у який адресація налаштовувалась статично, на даній робочій станції цього не прийдеться робити, оскільки на контролері домену DC-01 встановлено і налаштовано роль DHCP-серверу, все, що залишається – це приєднати WS-01 до домену, для цього змінимо ім’я комп’ютера на даній станції, та приєднаємося до домену Active Directory.

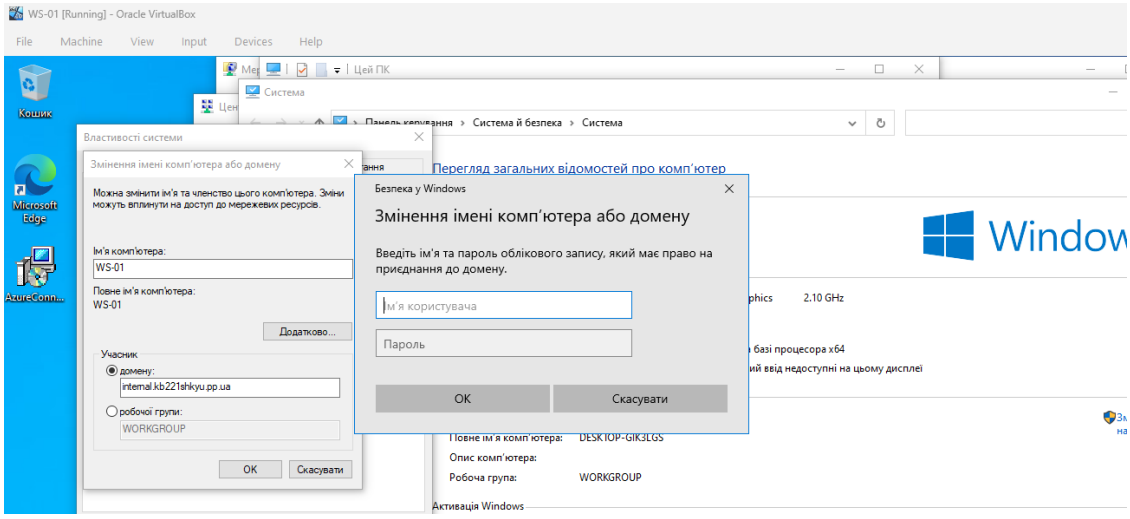


Рисунок 3.7 – Процес зміни імені комп'ютера на робочій станції і приєднання до домену

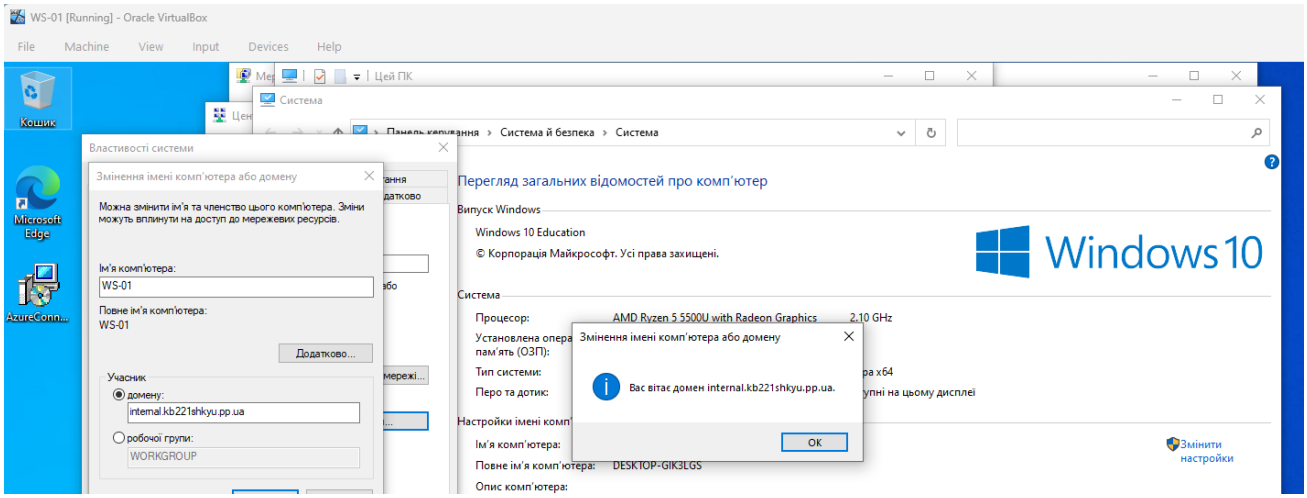


Рисунок 3.8 – Результат приєднання до домену

Після виконання відповідних дій необхідно здійснити перезавантаження станції. Після перезавантаження відбувається отримання інформації про адресацію від DC-01 та завершення базової конфігурації, результат зображено на рисунку 3.9

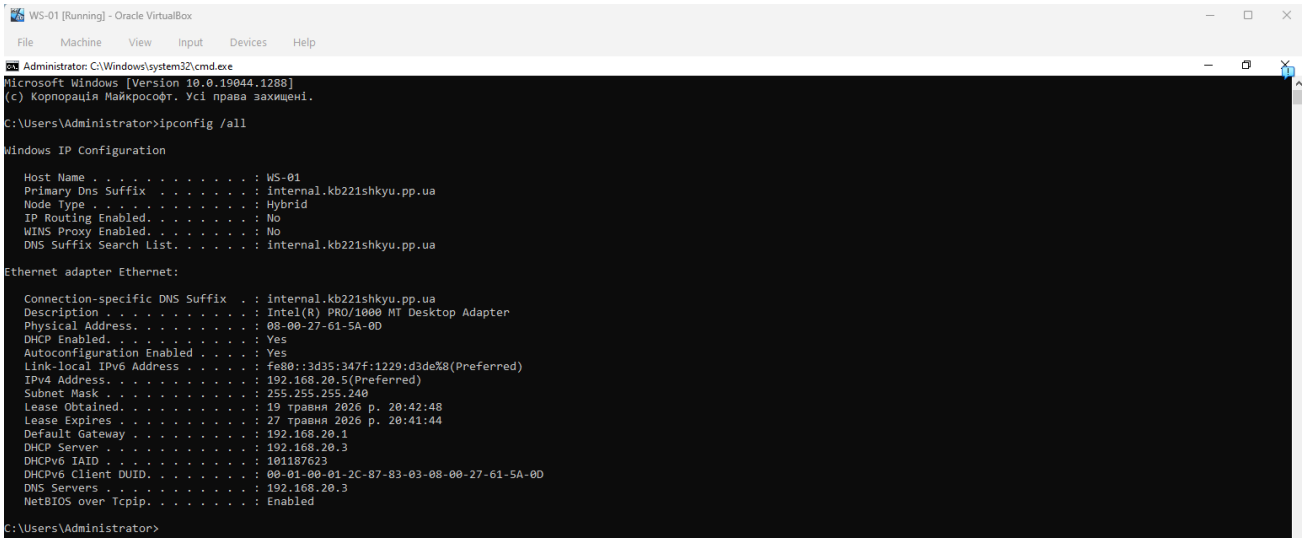


Рисунок 3.9 – Результат базової конфігурації та отримання інформації про адресацію від DC-01

З метою розширення вектору моніторингу, на базі вебсервера WEB-01 додатково розгортається вебдодаток, який виступатиме окремим об’єктом аналізу. Для розробки використовувався редактор коду Visual Studio code. На початку розробки стояв вибір між різними видами вебдодатків фреймворку ASP.NET core і в результаті дослідження було обрано модель Razor Pages. Дана модель лежить на основі сторінок для створення веб-інтерфейсів, що відтворюються на сервері. Інтерфейс Razor Pages динамічно відтворюється на сервері для генерації HTML-та CSS-коду сторінки у відповідь на запит браузера. Сторінка надходить до клієнта у вигляді, готовому до відображення [22].

Для побудови інтерфейсу користувачів використовується відкрита бібліотека UI-елементів Bootstrap. Вебдодаток містить базовий набір сторінок, архітектура яких дозволяє симулювати сценарії реальних веб-атак, для подальшого дослідження реагування на них. Перелік усіх сторінок, які місить вебдодаток та їх опис наведено у табл. А.2.

Дизайн створеного веб-сайту інтернет-магазину досить базовий, бо побудований з використанням відкритої бібліотеки Bootstrap. Зовнішній вигляд усіх вищеописаних сторінок зображено на рисунках 3.10 та Б.1-7.

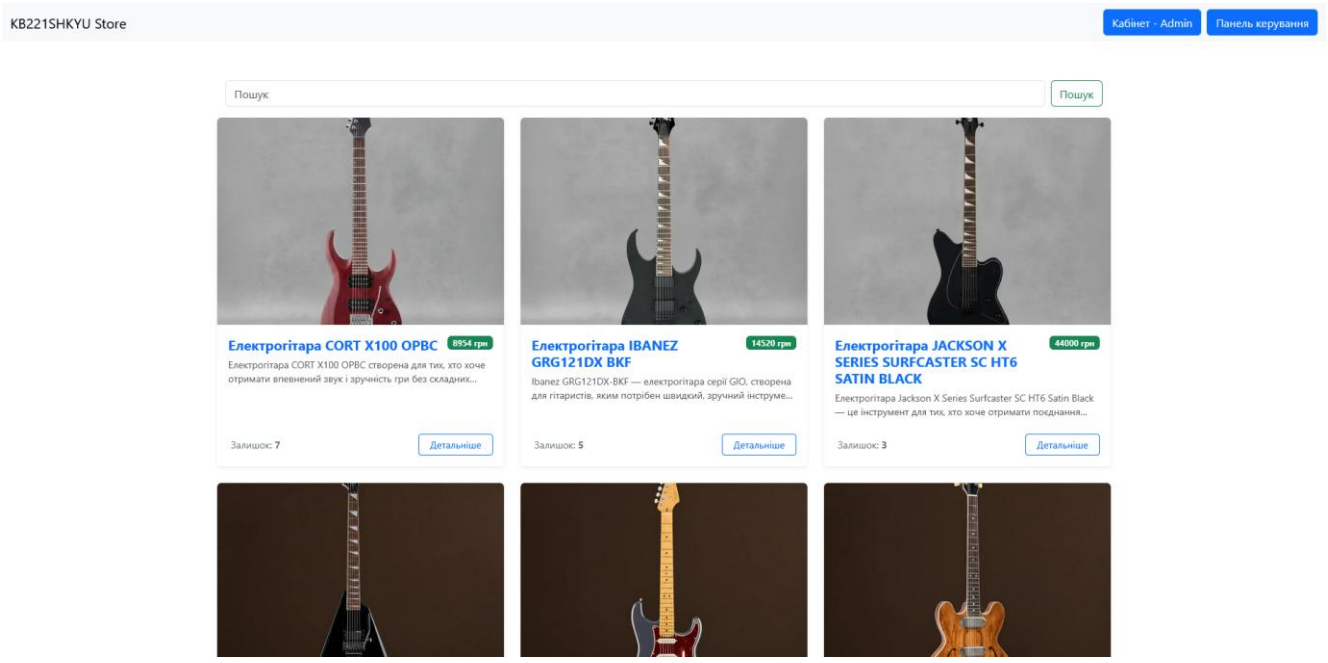


Рисунок 3.10 – Зовнішній вигляд головної сторінки

З урахуванням обраної схеми моніторингу і діагностики, яка передбачає передачу подій безпеки з вебсервера до хмарного середовища, критично необхідним є попереднє фільтрування усіх згенерованих подій. Оскільки вебдодаток інтернет-магазину цілком самостійно може генерувати значну кількість мережного трафіку, це допоможе розвантажити мережу, що в результаті оптимізує використання мережних ресурсів та допоможе уникнути дублювання трафіку. Саме з такою метою у додатку було налаштовано генерацію подій з попереднім переглядом на наявність деяких аномалій.

За попередній перегляд відповідає проміжний модуль *RequestAuditMiddleware*, який виконує перевірку кожного запиту до вебдодатку на наявність у ньому підозрілої активності, та створення подій відповідного рівня після виявлення аномалій. Для перевірки запитів на наявність аномалій використовується метод *IsSuspiciousActivity*, який наведений далі у табл. В.1. Даний метод використовує наступний набір небезпечних корисного навантаження запитів який наведено у табл. В.2. Далі за результатами використання даного метода формується рядок події, який ми отримуємо. Метод створення подій наведено у табл. В.3.

Для публікації додатку у відкриту мережу інтернет використовується Cloudflare. Для того, щоб дана публікація була можливою, необхідно орендувати

домен на відповідному сервісі для орендування, та після чого додати орендований домен до Cloudflare. У досліджуваному проекті використовується безкоштовний тип хостингу – самостійний хостинг. Його суть полягає у тому, що для хостингу використовується агент, який створює пряме з'єднання з Cloudflare і видає, це з'єднання називають тунелем. Далі до цього тунелю на порталі можна додавати свої додатки, ці додатки є лише, можна сказати, адресами, для цільового додатку було обрано адресу з під-домену *testapp*. Після створення, Cloudflare просто направляє трафік через свою інфраструктуру до створеного тунелю і потім до відповідної кінцевої точки на іншому кінці тунелю, для нашого додатку це 192.168.20.4:80 – тобто адреса вебсерверу. Таким чином і відбувається публікація додатку у відкриту мережу інтернет.

Тепер маючі основні об'єкти генерації подій у системі, нам необхідно підключити їх до хмарного середовища. Як вже обговорювалося раніше, це буде зроблено з використанням моделі Azure Arc + AMA. З метою автоматизації процесу розгортання хмарної інфраструктури та її підготовки до інтеграції з агентами моніторингу було розроблено скрипт розгортання з використанням інтерфейсу командного рядка Azure CLI та середовища PowerShell. Процес конфігурування передбачає послідовне створення та налаштування базових хмарних ресурсів у межах визначеної підписки та тенанту. Регіоном для розгортання обрано локацію Switzerland North через її доступність для студентської підписки. Якщо ж підписка звичайна, то жорстких обмежень у виборі регіону немає.

Налаштування інфраструктури складається з таких ключових етапів:

1. *Ініціалізація та автентифікація*: Виконання входу в обліковий запис Azure та встановлення цільового контексту підписки;
2. *Створення логічного контейнера*: Розгортання групи ресурсів rg-szn-arcs, яка об'єднує всі компоненти системи моніторингу для зручності керування та делегування прав;
3. *Створення сервісного облікового запису Service Principal*: Генерація службового запису ArcOnboardingSP із призначенням специфічної ролі «Azure

Connected Machine Onboarding». Цей запис має обмежені права і використовуватиметься безпосередньо агентами на кінцевих точках для їхнього автоматичного онбоардингу в Azure Arc без участі адміністратора;

4. *Розгортання репозиторію логів*: Створення робочої області Log Analytics Workspace law-arcs-monitor;

5. *Конфігурування правил збору даних*: Динамічне формування та створення DCR під назвою dcr-arcs-windows на основі JSON-шаблону;

6. Асоціація підключених ресурсів зі створеним DCR.

Скрипт розгортання хмарної інфраструктури наведено у табл. В.4. В результаті виконання даного скрипту, ми автоматично розгортаємо нашу інфраструктуру у хмарному середовищі Azure, готову до підключення локальних серверів. Для їх підключення використовується згенерований скрипти на порталі Azure.

Після завантаження необхідних скриптів, ми зупиняємось перед вибором методу підключення усіх машин, для цього, як вже було визначено в результаті аналізу раніше, буде використано групові політики. Після вибору відповідного методу потрібно завантажити необхідні інсталяційні пакети агентів та базовий інсталяційний скрипт.

Згенерований Azure шаблон скрипту є узагальненим і потребує додаткової адаптації під параметри нашої локальної мережі, шляхом додавання на відповідні місця нашого домену internal.kb221shkyu.pp.ua, повного імені комп’ютера, де знаходяться файли для розгортання агентів, у нас це наш головний контролер домену DC-01, та назви мережної папки, у якій ці файли розміщені, нехай це буде ArcDeploy.

GPO створюється шляхом запуску попередньо завантаженого скрипту, перед цим у папку ArcDeploy потрібно перенести раніше завантажений інсталяційний пакет агенту. У реальному середовищі секрет облікового запису служби доцільно зберігати у захищеному сховищі, наприклад Azure Key Vault, або використовувати тип SecureString для зменшення ризику його компрометації. У межах лабораторного стенду секрет передавався безпосередньо через параметри сценарію розгортання з

метою спрощення демонстрації процесу налаштування. Даний скрипт наведено у табл. В.5.

Скрипт DeployGPO.ps1 знаходиться серед файлів, які ми завантажували перед генерацією скрипту з табл. В.5. В результаті виконання скрипту, на контролері домену створюється відповідний GPO, який відповідає за автоматичне розгортання агентів на комп'ютерах домену, і їх підключення до хмарної інфраструктури Azure Arc.

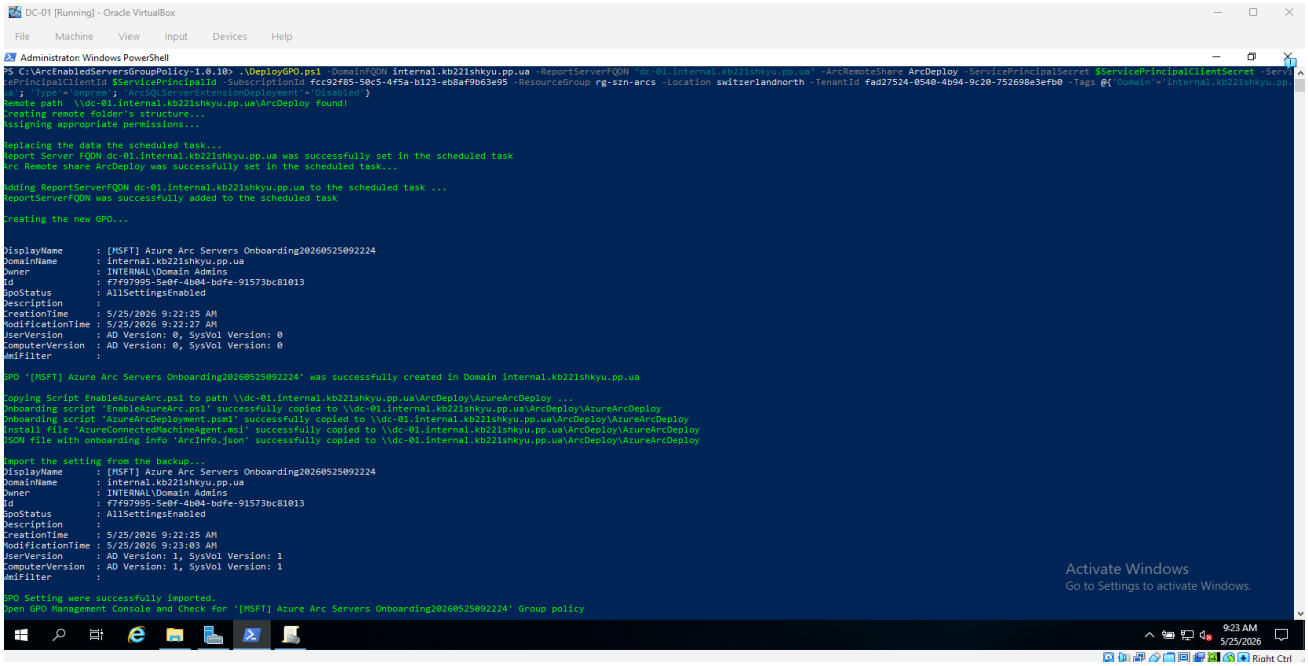


Рисунок 3.11 – Результат виконання скрипту на DC-01

Створений GPO потрібно асоціювати з відповідною організаційною одиницею, у якій знаходяться потрібні нам машини. Створимо OU з назвою Arc Computers, та перенесемо туди усі машини нашого домену, які необхідно підключити до Azure Arc.

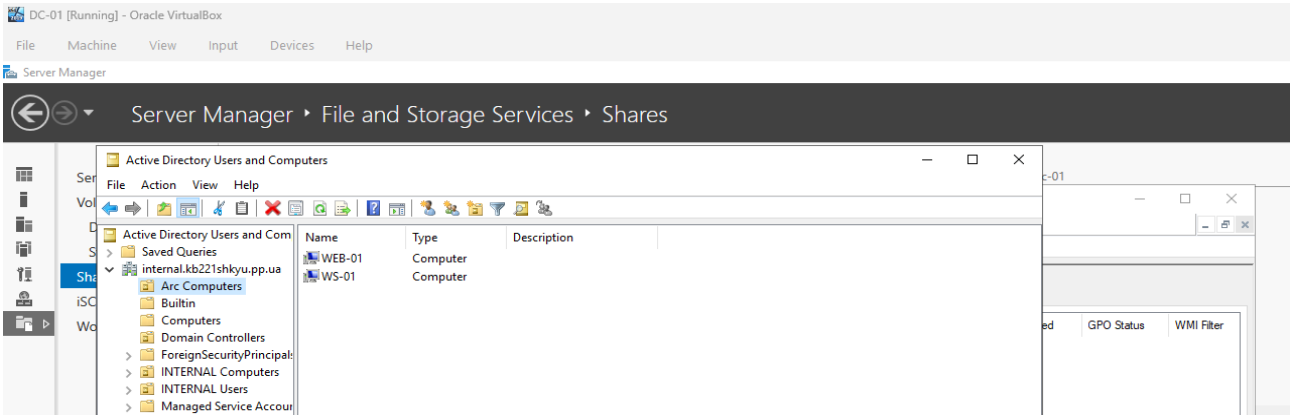


Рисунок 3.12 – Результат створення нової OU та переносу туди інших машин

Контролер домену не переносився до новоствореної OU з метою збереження його працездатності, для розгортання агентів на ньому, новостворену GPO потрібно також асоціювати з OU контролерів домену. архітектура Azure Arc планувалася на додавання машин, які працюють 24/7, тому якщо не вносити зміни у групову політику, будуть підключені лише машини з операційною системою Windows Server, для дослідження роботи таких агентів на звичайних робочих станціях, відредагуємо створений GPO, і додаймо до списку той тип операційних систем, на яких потрібно проводити інсталяцію агенту.

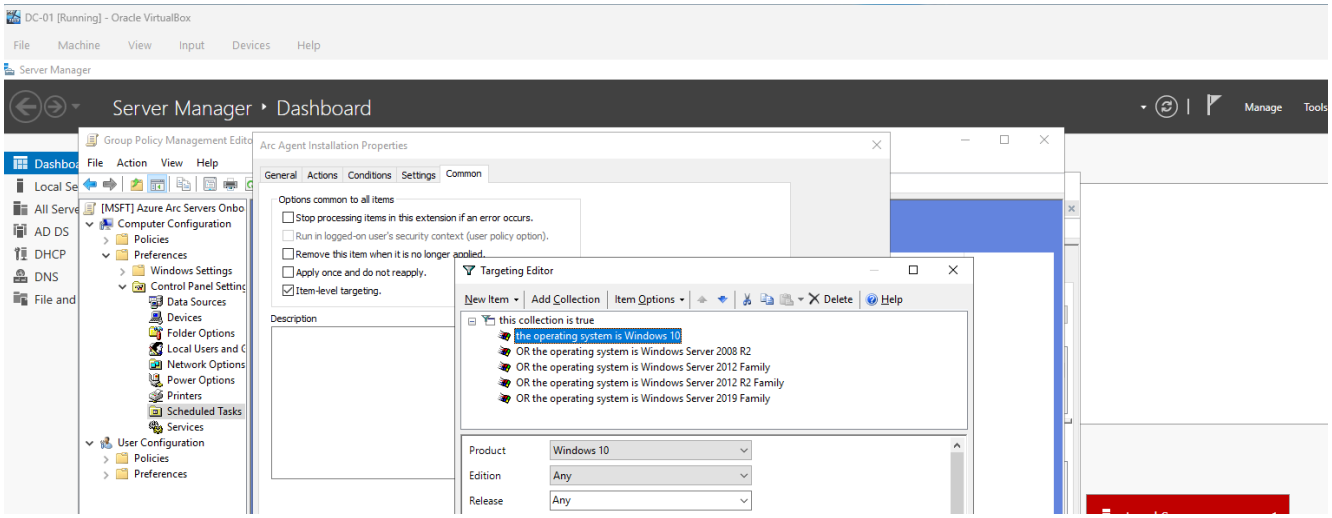


Рисунок 3.13 – Додавання Windows 10 до списку потрібних операційних систем

Автоматичне розгортання відбудеться після оновлення групових політик на кожній машині, що займе приблизно хвилин 90. Для примусового оновлення можемо використати команду *gpupdate /force*.

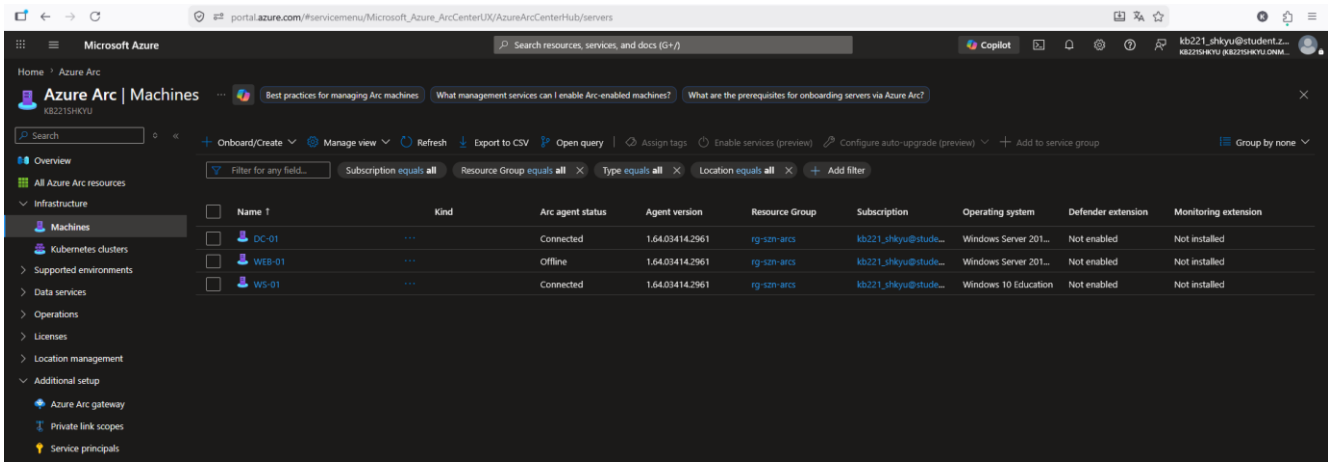


Рисунок 3.14 – Результат онбоардингу усіх машин

Для асоціації підключених ресурсів з DCR, та встановлення AMA використаємо написаний скрипт, який отримає список таких ресурсів, і вручну асоціює кожен з них. Даний скрипт наведено у табл. В.6. Результат виконання скрипту зображено на рис. 3.15

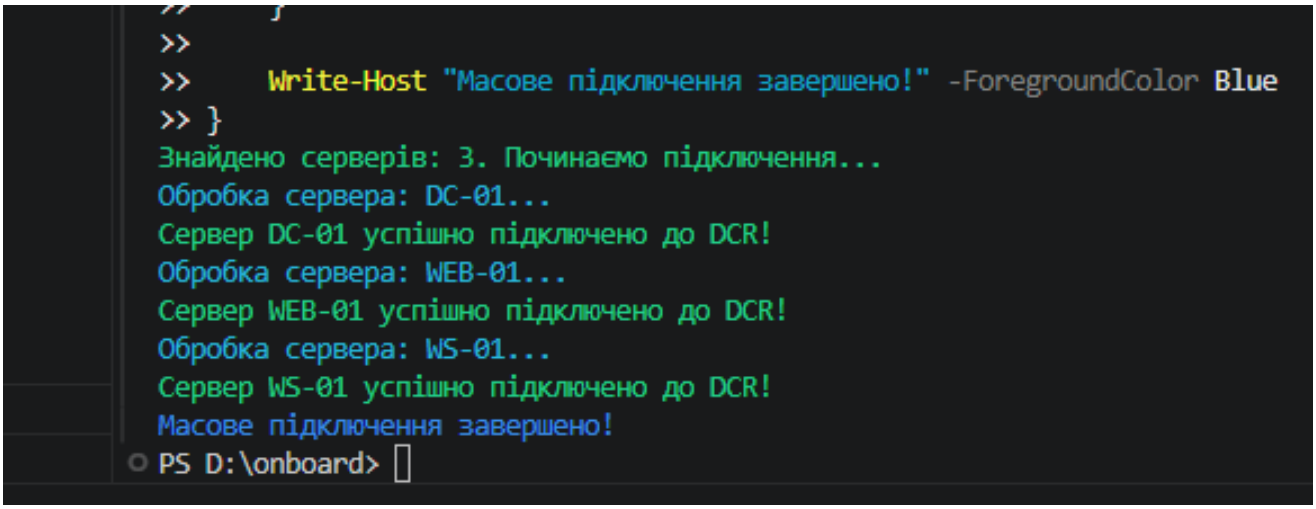


Рисунок 3.15 – Результат виконання скрипту

На цьому підключення усіх локальних ресурсів до хмарної інфраструктури Azure закінчено. Тепер у нас є можливість централізованого перегляду відповідний подій, у створеному LAW використовуючи KQL. Після асоціації і підключення наших ресурсів, у LAW було автоматично створено декілька таблиць, їх опис наведено у табл. 3.1, а значення кожного поля наведено у табл. А.3-4.

Таблиця 3.1

Автоматично створені таблиці у Log Analytics Workspace

Назва таблиці	Опис
Heartbeat	Відповідає за зберігання моніторингових даних про доступність до сервера. Таблиця містить у собі результаті запитів, які повторюються кожну хвилину. Таблиця зазвичай використовується для перевірки статусу сервера, відстеження версії агенту, перевірки основних даних операційної системи та IP-адреси
Event	Містить у собі події з Event Viewer, які зазначені у DCR. Запис у таблицю відбувається динамічно, лише тоді, як відповідна подія з’явиться на відповідній машині. Саме ця таблиця генерує найбільше даних, якщо немає обмежень у DCR.

Спробуємо вивести декілька логів аудиту з DC-01, Результат виконання запиту зображено на рис. 3.16.

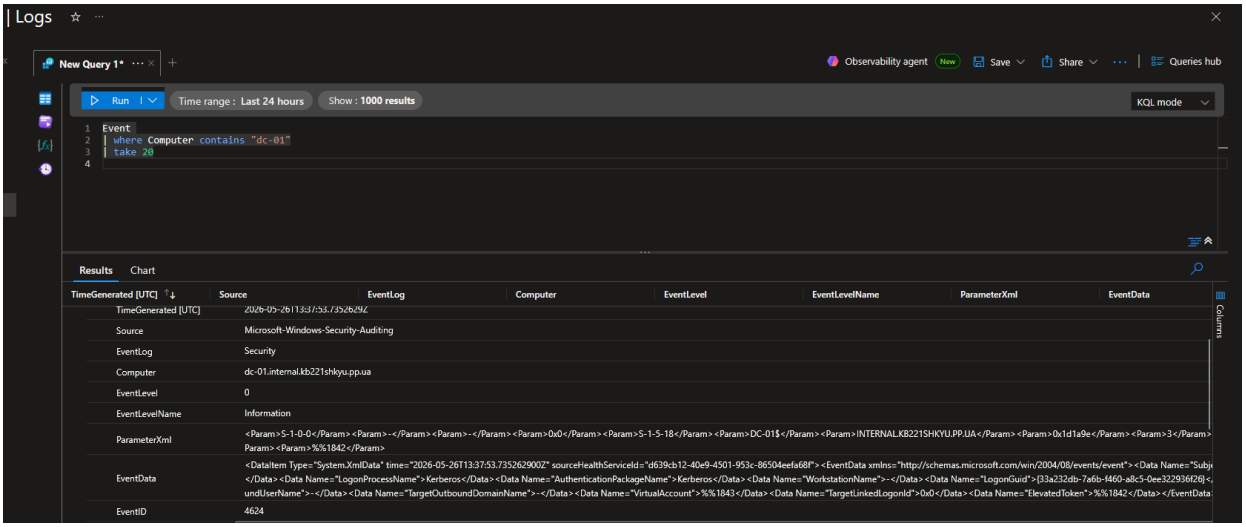


Рисунок 3.16 – Результат виконання KQL запиту

Також можна переглянути загальну статистику логінів на усіх підключених машинах, відфільтрувавши події за Event ID 4624 та об’єднавши ці події за хостами. Результат виконання такого запиту зображено на рис. 3.17.

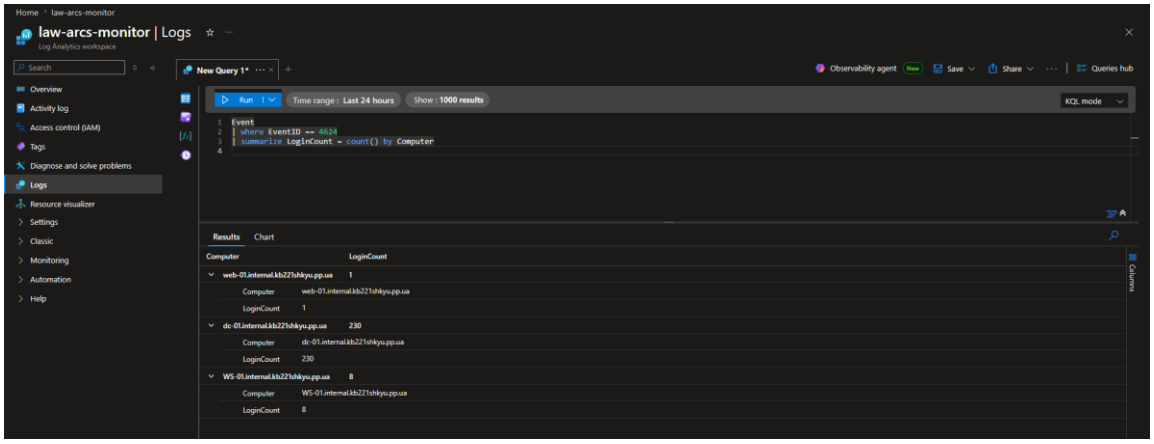


Рисунок 3.17 – Статистика логінів на усіх машинах

Тепер на порталі Azure, нам необхідно підключити наш LAW до Microsoft Sentinel, для інтеграції середовища зберігання подій з усіх підключених машин до середовища аналітики Microsoft Sentinel, що дасть можливість аналізувати дані події з LAW та налаштовувати автоматизовані правила для реагування чи оповіщення. Результат підключення зображено на рис. 3.18.

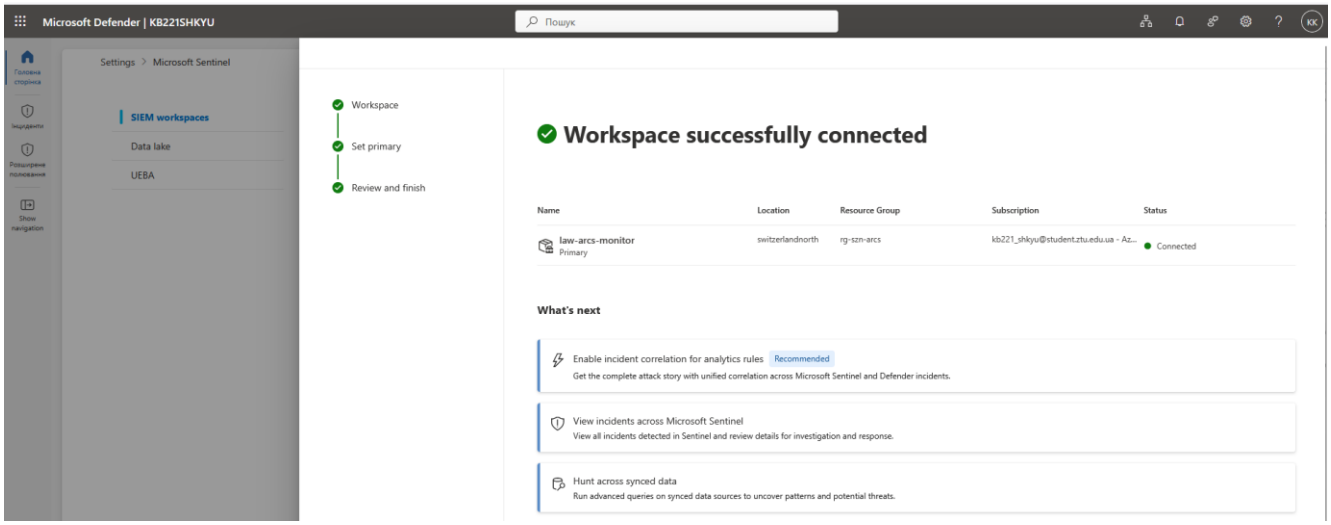


Рисунок 3.18 – Результат підключення LAW до Microsoft Defender

На цьому підготовка інфраструктури і підключення усіх агентів завершена.

3.2 Робота з KQL для візуалізації метрик (SIEM) та створення сценаріїв автоматизованого реагування (SOAR)

KQL використовується для дослідження даних, аналізу логів та телеметрії, а також для побудови аналітичних звітів у хмарному середовищі Azure. У контексті розгорнутої системи моніторингу безпеки, KQL виступає основним інструментом для обробки великих масивів інформації, що надходять від агентів. Мова запитів має декларативну структуру, подібну до SQL, але оптимізована для швидкої фільтрації, агрегації та статистичного аналізу часових рядів.

Структура усіх KQL-запитів досить однакова і базується на концепції потокового конвеєра даних. Кожна операція починається з ініціалізації первинного джерела даних – відповідної таблиці, у нашому дослідженні їх 2 – це Heartbeat та Event і зберігаються вони у нашому LAW. Далі у запиті вибудовується сам пайплайн за допомогою символу «|» та використанням різних операторів, що дозволяє виконувати покрокову трансформацію, фільтрацію та агрегацію інформації. Для побудови ефективних запитів для побудови аналітичних правил та дашбордів було використано та досліджено наступний базовий набір операторів і методів KQL [26]:

1. *Оператори логічної та часової фільтрації*:

– where – базовий оператор конвеєра, призначений для фільтрації рядків таблиці за відповідною умовою. Він дозволяє відсікати нерелевантні події та ізолювати критичні інциденти за допомогою стандартних логічних операторів порівняння.

– ago() – метод динамічного обчислення часового зміщення від поточної мітки системного часу. У поєднанні з оператором where, наприклад, «| where TimeGenerated > ago(24h)», цей метод забезпечує оптимізацію запитів, обмежуючи часове вікно генерації подій, що зменшує кількість логів для подальшої обробки і забезпечує актуальність отриманих подій.

– has / contains – звичайні рядкові оператори для текстового пошуку в тілі логів. Оператор has виконує високоефективний пошук за повними словами, що важливо для швидкої перевірки конкретних кодів помилок чи ідентифікаторів, тоді як contains використовується для пошуку довільних рядків.

2. *Оператори трансформації, декомпозиції та проектування структури*:

– extend – оператор розширення, який динамічно створює нові обчислювальні стовпці у результуючій таблиці на основі аналізу вже існуючих даних. Наприклад, можна діставати за допомогою регулярних виразів потрібні значення IP-адрес з тіла події, і передавати як окремий стовпчик таблиці для подальшого використання.

– extract() – метод екстракції даних, що використовує регулярні вирази для виділення специфічних індикаторів компрометації – таких як IP-адреси, імена користувачів або типи атак – із текстових полів опису події, передаючи їх оператору extend для формування нових структурних полів.

– project – оператор проектування, який виконує роль селектора пам’яті. Він визначає фінальну структуру таблиці, вибираючи лише необхідні стовпці, встановлює їхній чіткий порядок та вилучає з конвеєра інші надлишкові поля, що оптимізує швидкість відображення результатів.

3. *Конструкції розгалуження та умовного вибору*:

– iff() та case() – вбудовані логічні методи для реалізації умовних розгалужень, ті ж самі if else. Вони використовуються для динамічної класифікації та

нормалізації даних. Наприклад, метод case() дозволяє зіставити сирі числові рівні критичності логів із текстовими маркерами для їх зручного сприйняття на дашбордах.

4. *Оператори обчислення метрик, сортування та вибірки*:

– summarize – ключовий оператор агрегації, який групує лінійний потік подій за допомогою директиви by та обчислює зведені метрики (наприклад, загальну кількість подій за допомогою функції count()).

– bin() – метод квантування часу, який обов'язково використовується всередині оператора summarize для розбиття подій у фіксовані часові інтервали з кроками.

– sort by (або order by) – оператор упорядкування результатів, що найчастіше застосовується у спадному порядку за міткою часу для відображення найсвіжіших подій на початку списку.

– take (або limit) – оператор за замовчуванням для обмеження вибірки, який повертає задану кількість випадкових рядків із таблиці.

– join – оператор кореляції, який дозволяє об'єднувати дані з різних таблиць.

У другому розділі було специфіковано джерела телеметрії (табл. 2.3) і розбито їх на три групи ключових метрик. На базі цих метрик буде побудовано два дашборди, перший для відображення аномалій за останні день-два, він допоможе вчасно помічати якісь відхилення від норми у мережі, що сприятиме вчасному реагуванню на інциденти, або навіть допоможе запобігти майбутнім. Другий для відображення загальної статистичної інформації за довгий проміжок часу, що також допоможе при перегляді інформації з першого дашборду, бо він дасть ту саму базову лінію поведінки користувачів у мережі, і буде одразу видно, де поведінка явно аномальна. Усі запити та результативні графіки наведено у таблиці А.5.

В результаті було створено дві панелі для ефективного моніторингу з базовими метриками. Дані панелі можна переглядати як на порталі Azure, так і на Microsoft Defender. Гнучкість інтерфейсу дозволяє розмістити графіки так, щоб мо-

жна було ефективно переглядати найважливіші метрики у першу чергу, а оновлення у реальному часі підтримує загальну ефективність дашбордів, тримаючи усі графіки актуальними. Результат побудови панелей зображено на рис. 3.19-3.20.

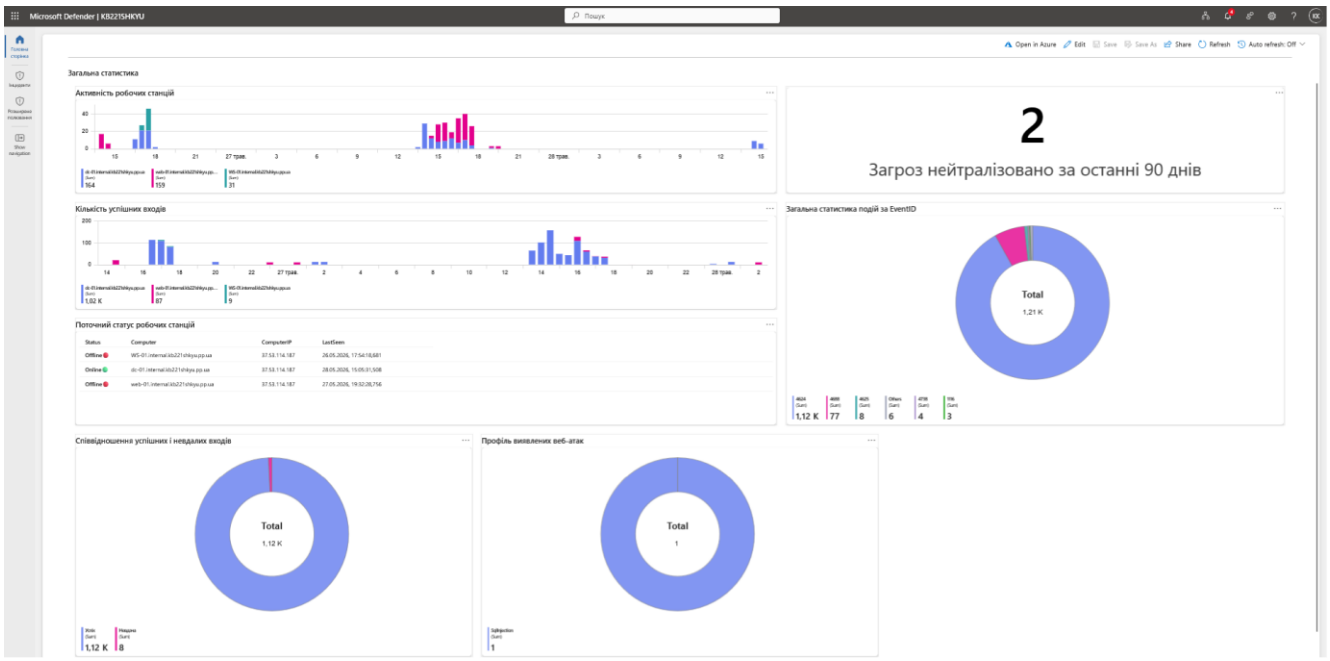


Рисунок 3.19 – Дашборд статистики

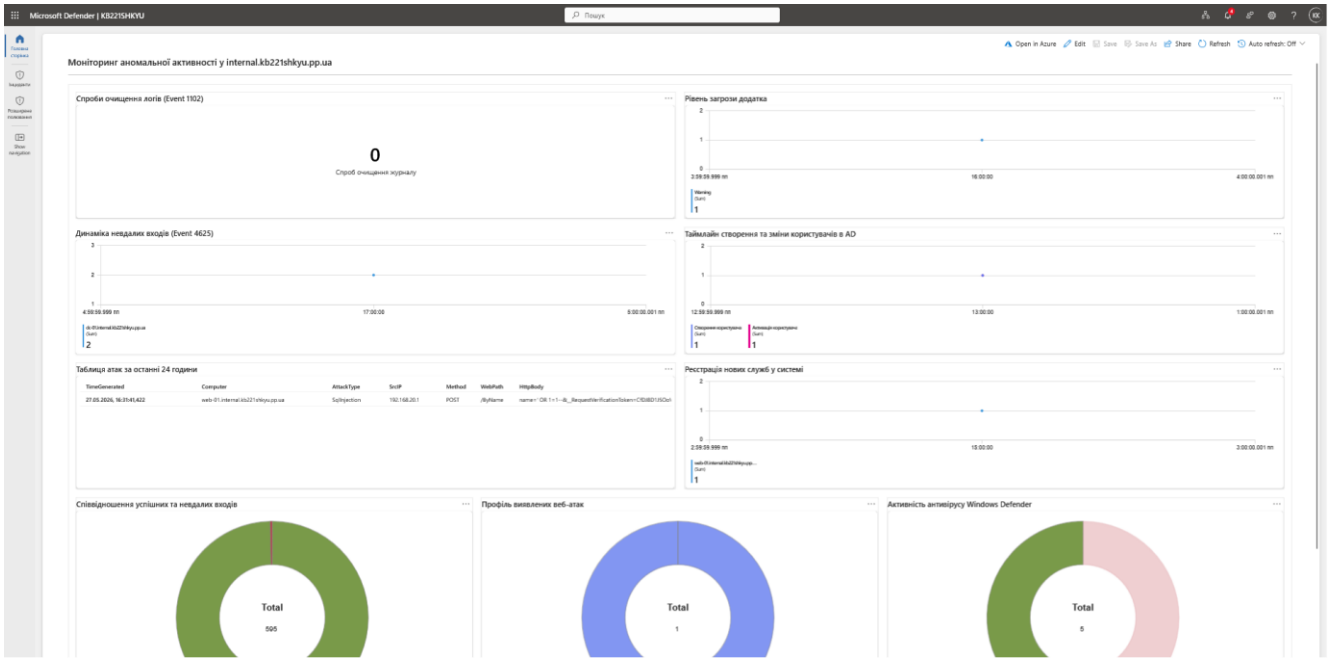


Рисунок 3.20 – Дашборд аномалій

Після успішного налаштування SIEM-складової, логічним наступним етапом буде створення системи оповіщення про інциденти. Цей крок забезпечує плавний перехід у SOAR частину проекту, оскільки механізми, які використовуються для

оповіщення, так само використовуються і у автоматичному реагуванні на інциденти. Будь-яке реагування, чи це звичайне оповіщення, чи виконання конкретних заходів безпеки, можна розділити на два рівні: SIEM рівень, та SOAR.

Рівень SIEM відповідає за аналітику та збір даних, у середовищі Azure для цього використовуються аналітичні правила у Sentinel. Вони базуються на тих самих KQL-запитах, які також і використовувались для створення дашбордів. Аналітичні правила використовують ці запити і перевіряють їх результати зі встановленими умовами для генерації оповіщень безпеки і створення інцидентів.

Рівень SOAR вже відповідає за реагування на відповідні оповіщення, шляхом виконання плейбуків, в Azure для цього використовується Automation Rules та Logic Apps. Таким чином SOAR автоматично бере дані про цей інцидент з оповіщення і надсилає повідомлення на ваш робочий телефон, або інші встановлені канали зв'язку. Крім того SOAR може самостійно, без нашої участі, виконати команду через Azure Arc і виконати відповідні дії, якщо нами було створене таке правило.

Для оповіщення планувалось використати декілька стандартних каналів зв'язку, а саме: Електронну пошту, Teams, Goole Chat Webhook та SMS на мобільний. Однак в результаті аналізу було виявлено та проаналізовано поширену проблему, притаманну корпоративним або академічним хмарним тенантам. Безпекові політики Microsoft Entra ID та Google Workspace часто повністю блокують створення нових додатків сторонніми користувачами, обмежують прямий доступ до Microsoft Graph API та забороняють інтеграцію зовнішніх вхідних вебхуків. З огляду на ці інфраструктурні обмеження, було прийнято рішення, використання не традиційних для підприємств комунікаційних каналів, таких як: Telegram бот та Discord Webhook. Дані канали зв'язку можна розбити за критичністю інциденту. Дане розбиття наведено у таблиці 3.17.

Розпочнемо з оповіщень на електронну пошту, для цього нам необхідно створити плейбук який буде надсилати листа, та після чого, його можна буде використати у автоматизованому правилі, яке використовується у аналітичному правилі, яке і створює інциденти. Перший плейбук назвемо Send-Email. Після створення нас

перекине на сторінку з конструктором логіки створеного плейбуку. Логіка спрацювання починається з отримання відповідного інциденту який і спровокував виконання плейбуку.

Таблиця 3.2

Стратегія пріоритизація та розподілу каналів сповіщення SOAR

Критичність	Канал зв'язку	Приклад
Low / Information	Електронна пошта	Просто звіт за день чи планове оновлення.
Medium	Telegram / Discord	Поодинока веб-атака, яку відбив middleware.
High / Critical	SMS	Успішний вхід адміністратора в неробочий час, очищення логів

Модуль отримання інциденту має назву Microsoft Sentinel incident, він повертає усі дані пов'язані з інцидентом який і викликав даний Logic App. Дані представлені у вигляді JSON-об'єкту, і легко парсяться використовуючи різні модулі роботи з даними у самому Logic App. Для фільтрування відповідних об'єктів використовується метод Filter у який ми передаємо масив об'єктів у якому потрібно знайти окремий об'єкт значення конкретного поля якого відповідає умові, наприклад, для отримання IP-адреси серед Entities, нам потрібно перетворити *triggerBody()?['object']?['properties']?['relatedEntities']* на масив, використавши функцію array(), і в умові порівняти значення поля «*kind*» з «*Ip*». В результаті ми отримаємо об'єкт в якому значення поля «*address*» буде дорівнювати раніше визначеній у інцидентів адресі. Таким чином можна гнучко будувати Logic App передаючи будь-які потрібні дані у інциденті.

Для того щоб ці дані передавати у самому інциденті, потрібно у графічному інтерфейсі визначати ці самі Entities і призначати їм значення колонок вихідної KQL-запиту, або додати Custom Details, які будуть передаватись у вигляді словника. Їх парсинг трохи відмінний від парсингу Entities. Для цього необхідно буде дістати алерти з тіла інциденту, перетворити об'єкт Custom Details у JSON, бо він передається рядком, і вже дістати відповідне значення поля тим самим методом як і з Entities. Структура Send-Email зображена на рис. 3.21. Для забезпечення базової відмовостійкості при інформуванні, було також налаштовано RetRy Policy, який буде повторювати запит на відправку листа при отриманні помилки з кодом 500-511.

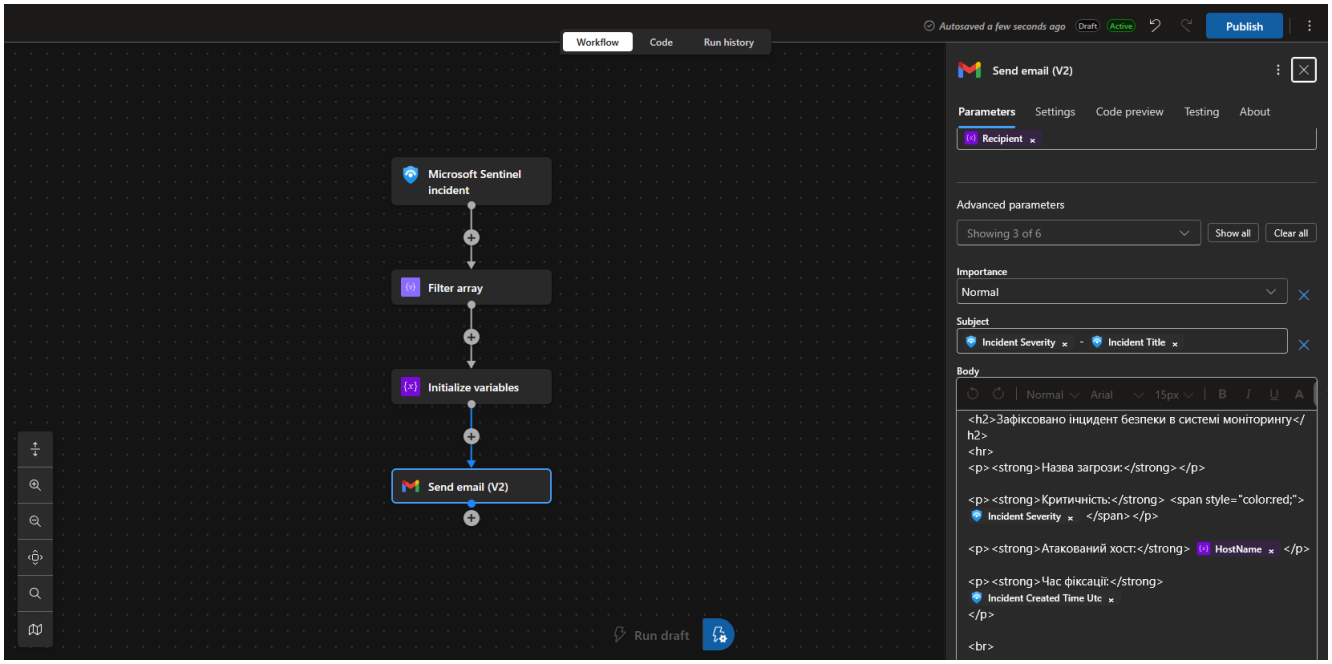


Рисунок 3.21 – Структура Send-Email та самого листа

У конструкторі Logic App також можна проводити тестування побудованої логіки з власним корисним навантаженням, або без нього. Результат перевірки побудованого Logic App зображено на рис. 3.22.

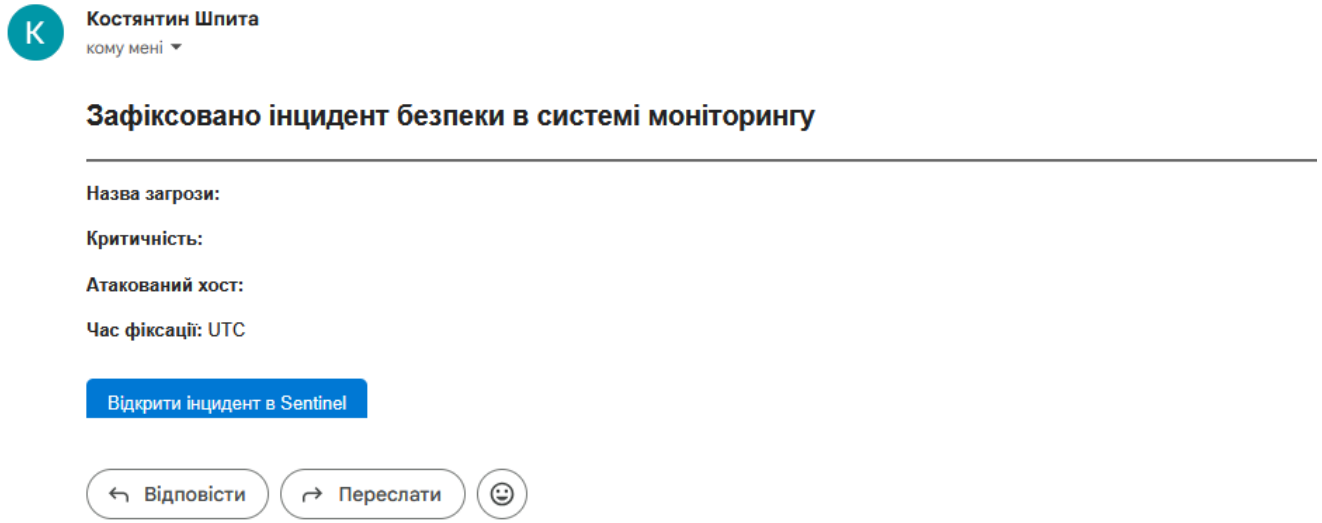


Рисунок 3.22 – Результат тестового виконання

Наступним методом оповіщення буде Discord webhook. Суть та ж, потрібно створити плейбук на базі Logic App, як і у випадку з Send-Email. Створений плейбук назвемо Send-Discord-Webhook. На даному етапі далі будуть створюватись лише плейбуки для оповіщення, аналітичні правила створимо після створення і тестування усіх плейбуків. Для оформлення повідомлення у Discord використаємо

вбудовані елементи платформи – embeds. Структура повідомлення наведена у табл. В.7. Загальна логіка Send-Discord-Webhook зображена на рис. 3.23.

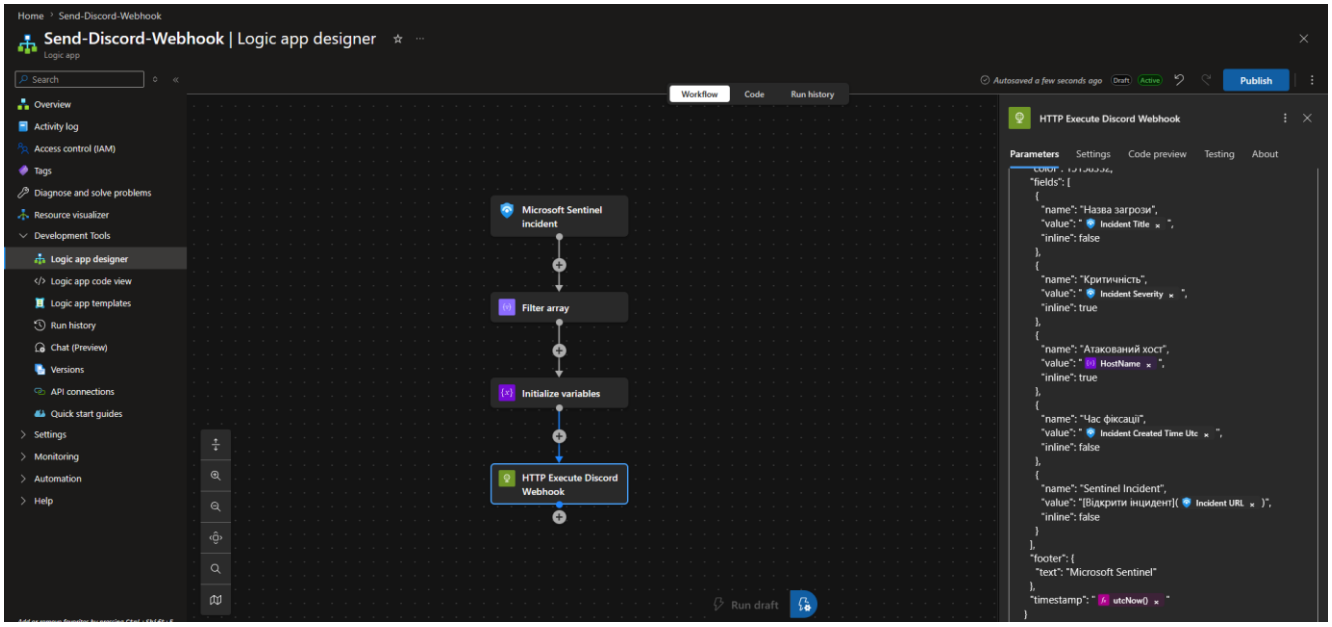


Рисунок 3.23 – Загальна логіка Send-Discord-Webhook

Результат тестової перевірки Send-Discord-Webhook зображено на рис. 3.24.

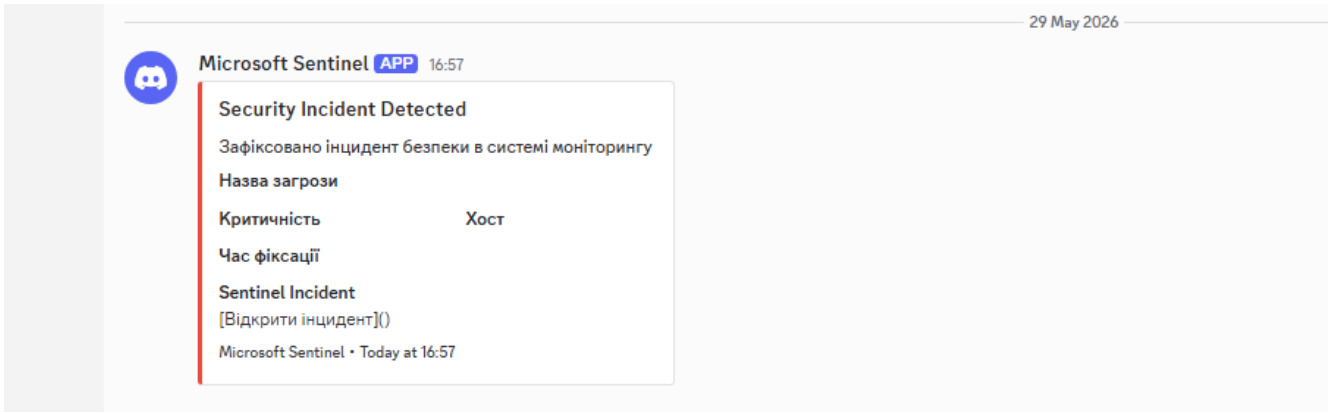


Рисунок 3.24 – Отриманий бланк в результаті виконання тесту

Наступний метод оповіщення – Telegram. На даній платформі для подібного інформування найзручніше використовувати Telegram-бот. Створимо плейбук з назвою Send-Telegram-Message, із загальною логікою виконання схожою до попередніх, тільки в кінцевому етапі використаємо конектор з дією бота. Створимо бота за допомогою *BotFather* у Telegram, після того, як створили бота, копіюємо готовий токен, який вже використовуємо для підключення до бота у Azure.

Важливим моментом є встановлення правильного ідентифікатора чату, щоб бот знав, куди саме необхідно надсилати повідомлення. Для отримання цього id

використовуємо *RawDataBot*. Маючи усе необхідне, можемо будувати наш Logic App. Загальна логіка створеного Logic App зображено на рис. 3.25.

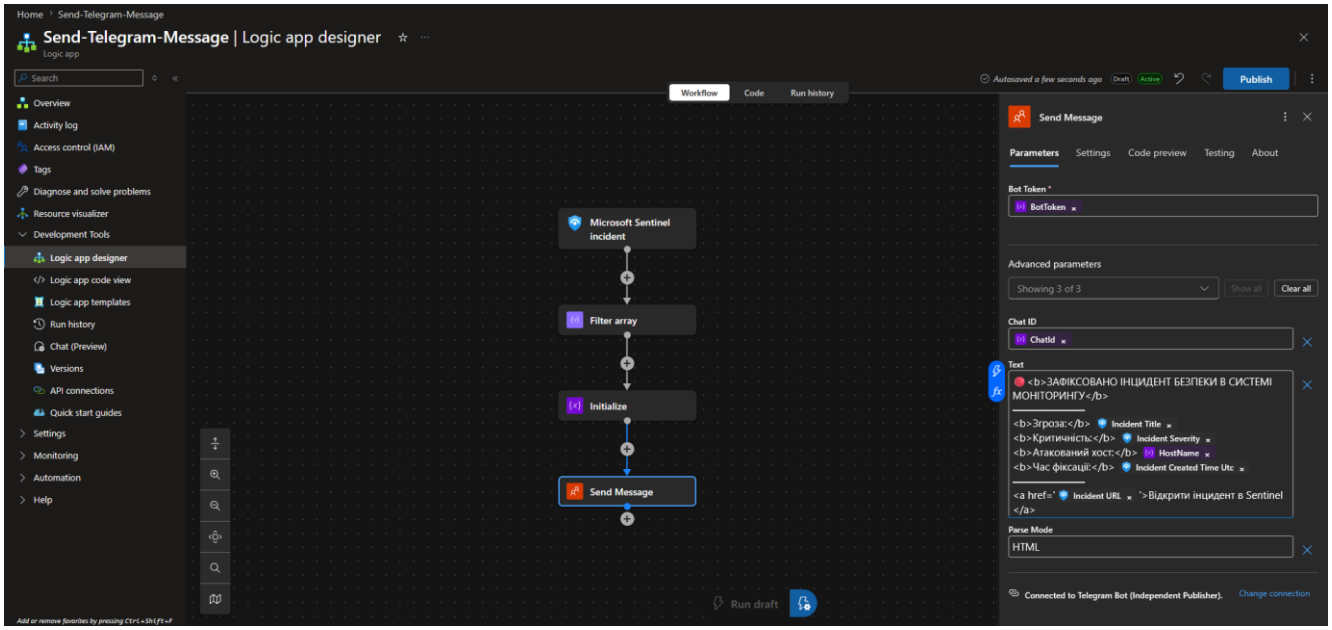


Рисунок 3.25 – Загальна логіка Send-Telegram-Message

Результат тестової перевірки Send-Telegram-Message зображено на рис. 3.26.

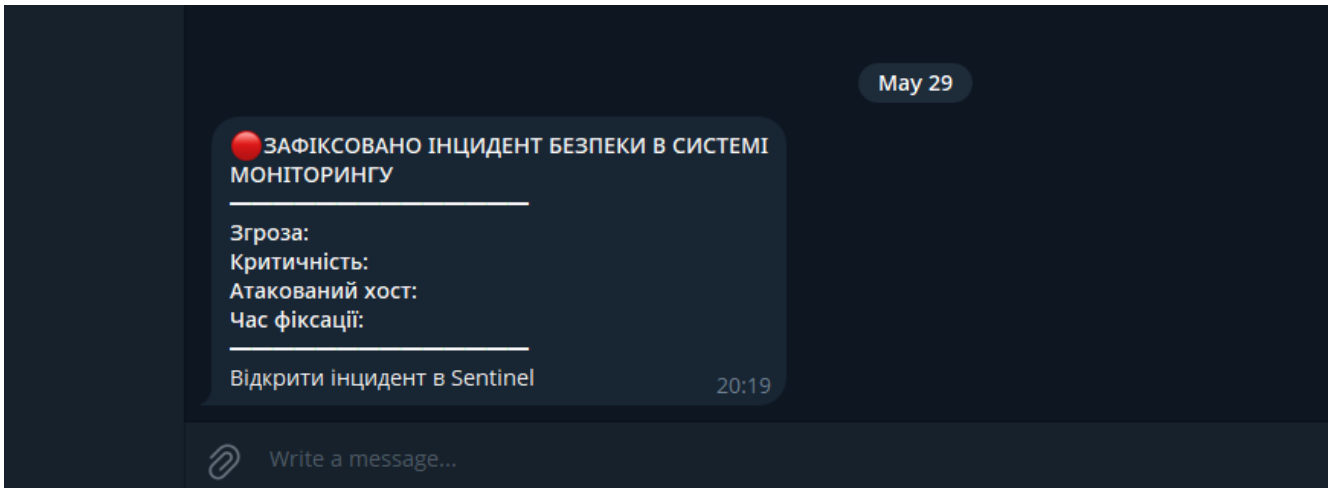


Рисунок 3.26 – Результат перевірки Send-Telegram-Message

Останнім методом оповіщення буде SMS. Існує багато різних способів відправки таких повідомлень, починаючи від офіційних API таких провайдерів як Київстар, або Водафон, закінчуючи різними CPaaS (Communications Platform as a Service) платформами, наприклад, як Twilio. Розглянемо варіант з Kyivstar API. На порталі Azure, у конструкторі Logic App, на жаль, немає ніяких відповідних конекторів до API Київстару, тому прийдеться працювати з API вручну. На порталі

Kyivstar API є достатньо документації про роботу з їх сервісами. Для роботи з API необхідні наступні кроки [23]:

1. Створити обліковий запис на порталі Kyivstar API;
2. За необхідності повного доступу підтвердити свої дані за допомогою ДРПОУ/РНОКПП.

Після реєстрації на порталі ми отримуємо клієнтський ідентифікатор та секрет, який потім використовуємо для доступу до самого API. Процес відправки SMS повідомлення виглядає наступним чином:

1. Отримання токена авторизації;
2. Прямий запит до методу відправки SMS з використанням отриманого токену авторизації.

Для автоматизації цього процесу у Azure Logic App використаємо HTTP. Першим запитом будемо отримувати токен авторизації, для його отримання необхідно передати секрет у запит, його відкрите зберігання у змінній буде досить небезпечним, тому для подальшого зберігання таких секретів використаємо Azure Key Vault Secrets. Додаємо до нашого сховища клієнтський секрет, та дістаємо його звідти у Logic App за допомогою модуля Azure Key Vault [27]. Після чого перевіряємо статус код відповіді, і якщо статус код 200, то переходимо до парсингу відповіді, та діставання самого токену. Далі ми просто використовуємо наш токен і спокійно можемо використовувати API для надсилання SMS. Загальна логіка Send-SMS зображена на рис 3.27.

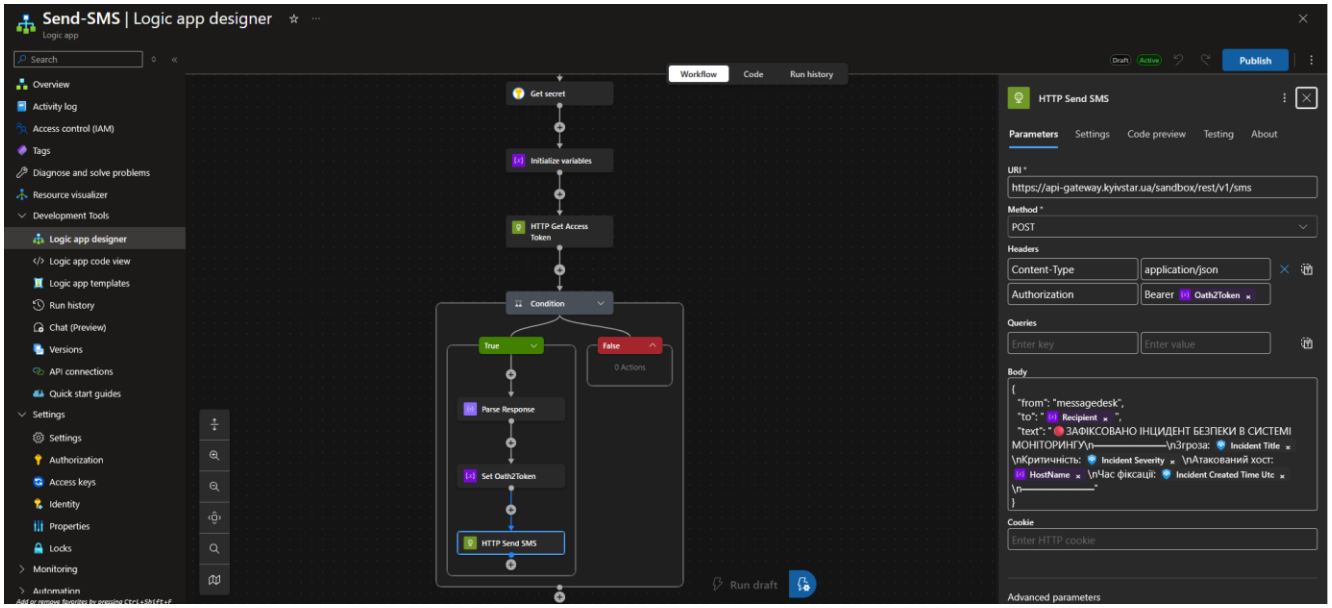


Рисунок 3.27 – Загальна логіка Send-SMS

Результат перевірки Send-SMS зображено на рис. 3.28.

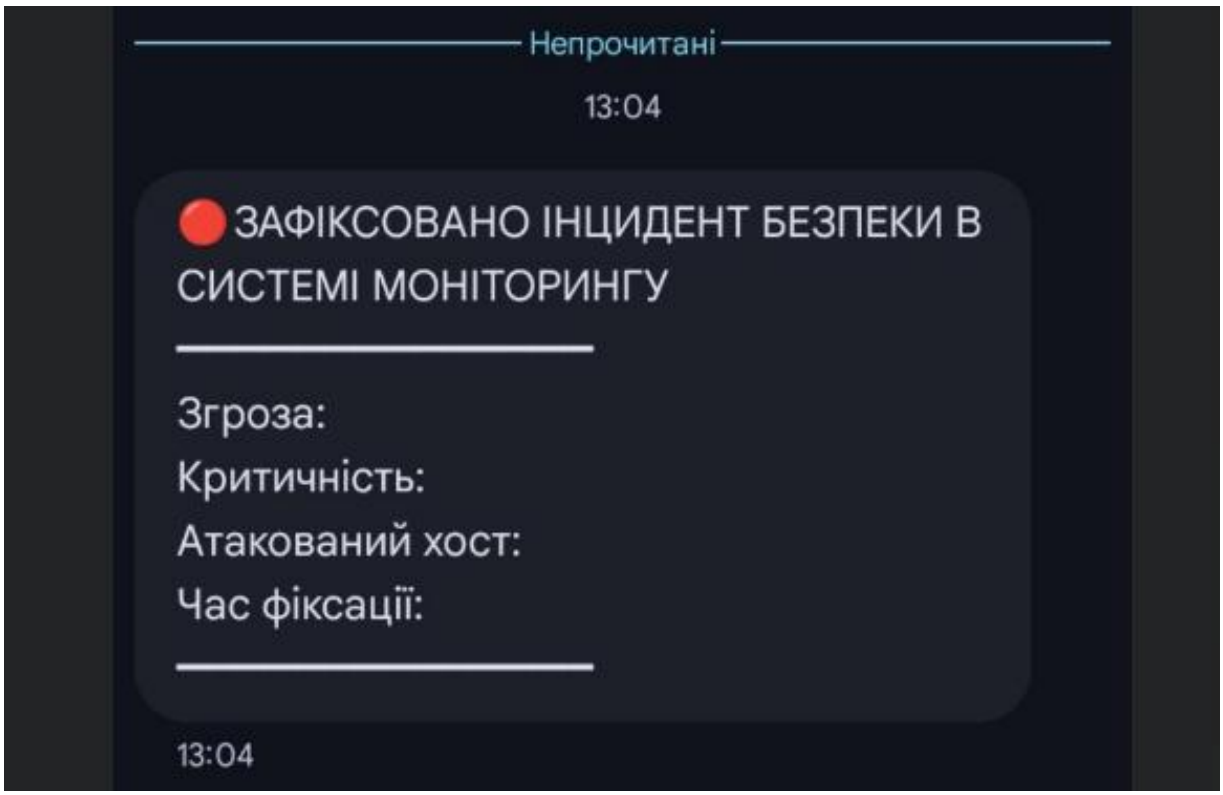


Рисунок 3.28 – Результат перевірки Send-SMS

Маючи усі реалізовані канали оповіщення перейдемо до SOAR частини, а саме реагування. На основі аналізу фундаментальних класів загроз досліджуваного проєкту у табл. 2.2, було виділено основні сценарії для подальшого дослідження.

Для дослідження виділимо наступні сценарії атак та дії у відповідь на виконання сценарію:

Таблиця 3.3

Сценарії та дії у відповідь

Параметр	Опис
Сценарій	Brute-Force на обліковий запис домену
Дії у відповідь	Дія 1: Виконання PowerShell-скрипту на веб-сервері/DC, додаючи IP-адресу атакуючого в локальний Windows Firewall. Дія 2: Оповіщення на електронну пошту
Сценарій	Компрометація облікового запису
Дії у відповідь	Дія 1: Встановлення статусу облікового запису на вимкнений, якщо не адміністратор. Дія 2: Оповіщення за допомогою SMS. Дія 3: Блокування IP
Сценарій	Веб атака
Дії у відповідь	Дія 1: Автоматичне оновлення правил WAF. Дія 2: Оповіщення на Telegram або Discord.
Сценарій	Виявленні вірусу на робочій станції
Дії у відповідь	Дія 1: Автоматичне увімкнення жорстких правил Windows Firewall для заборони будь-якого вихідного трафіку. Дія 2: Оповіщення на Telegram або Discord.
Сценарій	Виявлення створення "Backdoor" користувача
Дії у відповідь	Дія 1: Автоматичне вимкнення створеного користувача. Дія 2: Оповіщення на Telegram або Discord

На кожен з відповідних сценаріїв створимо по аналітичному правилу. Аналітичні правила для спрацювання використовують результаті відповідних KQL-запитів, їх побудова в загальному буде простою, так як більшість запитів вже реалізовано при створенні дашбордів. Для автоматизованого реагування, так само як і у оповіщенні, використовуються плейбуки на базі Logic App.

1. *Brute-Force на веб-сервер або контролер домену*

Створимо перший плейбук, який буде виконувати першу дію, зазначену у таблиці 3.3. Для цього трішки підправимо попередньо написаний KQL-запит, нам також потрібно витягнути IP адресу хосту, з якого відбувається атака, у описі події 4625 або 4771 вона є, і зазвичай, якщо вхід локальний, вона відповідає значенню 127.0.0.1, такі значення ми будемо відсіювати, і будемо ловити тільки ті, де IP-адреса відмінна від попередньої.

Також змінимо загальну логіку, агрегація буде відбуватись не за хостом, а за IP-адресою, з якої відбувається запит що дозволить точніше визначати чи це не

сума інцидентів на одному ПК з різних адрес, щоб не створювати лишніх алертів. Оновлений запит наведено у табл. 3.4.

Таблиця 3.4

KQL-запит

```
Event
| where EventID == 4771
| extend IpAddress = extract(@"<Data Name=""IpAddress"">([^<]+)</Data>", 1, EventData)
| extend SourceIp = replace_string(IpAddress, "::ffff:", "")
| where SourceIp !in ("127.0.0.1", "::1")
| summarize FailedAttempts=count(), FirstSeen=min(TimeGenerated), LastSeen=max(TimeGenerated) by Computer, SourceIp
| where FailedAttempts >= 8
```

Тепер створимо плейбук для реагування, назвемо його Ban-Ip, для виконання команд на відповідних робочих станціях, було зроблено рішення використовувати HTTP-запит до API Azure, оскільки інших готових методів виконання подібних команд не було. Для аутентифікації у запиті потрібно буде використовувати Managed Identity відповідного Logic App, для того, щоб можна було виконувати команди, переходимо до сторінки потрібного сервера, далі Access Control (IAM), обираємо роль Azure Connected Machine Resource Administrator, та пов'язуємо її з нашим Managed Identity створеного Logic App, після чого у конекторі HTTP можна використовувати аутентифікацію за допомогою Managed Identity. Для успішного запиту нам потрібні будуть Id підписки, назва ресурсної групи, та ім'я комп'ютера, який ми отримуємо зі згенерованого інциденту. Далі Логіка виглядає так:

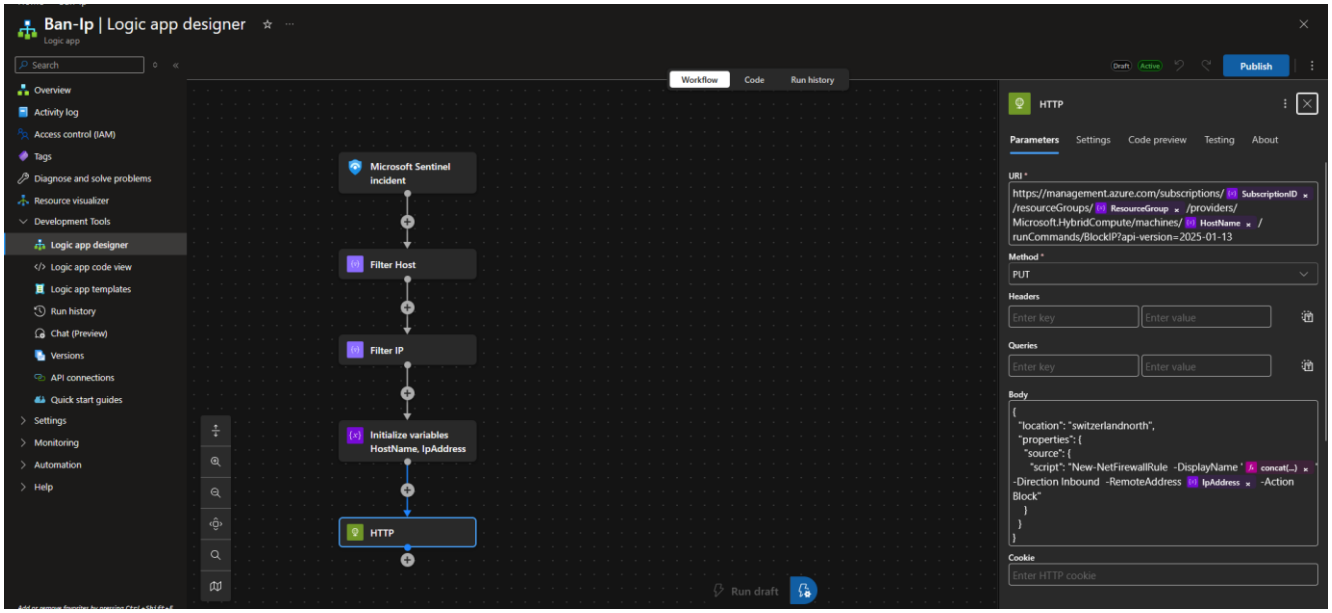


Рисунок 3.29 – Загальна логіка Ban-Ip

Проведемо тестування Ban-Ip статично вписавши адресу 1.2.3.4, результат тестування зображено на рис. 3.30.



Рисунок 3.30 – Результат тестування

2. Компрометація облікового запису

Зазвичай основним показником компрометації облікового запису є вдалий вхід, який передувався великою кількістю невдалих спроб входу. За основу візьмемо KQL-запит з попереднього сценарію, тільки зменшимо кількість невдалих входів, і додаймо запит на успішний вхід, після чого об’єднаємо дані з обох таблиць по IP-адресі, щоб залишились лише події пов’язані з конкретною адресою. KQL-запит наведено у табл. 3.5.

Таблиця 3.5

KQL-запит

```
let FailedLogons = Event
| where TimeGenerated > ago(30m)
| where EventID == 4771
| extend IpAddress = extract(@"<Data Name=""IpAddress"">([^<]+)</Data>", 1, EventData)
| extend SourceIP = replace_string(IpAddress, "::ffff:", "")
| where SourceIP !in ("127.0.0.1", "::1")
| extend Account = extract(@"<Data Name=""TargetUserName"">([^<]+)</Data>", 1, EventData)
| summarize
    FailedAttempts=count(),
    FirstSeen=min(TimeGenerated),
    LastSeen=max(TimeGenerated)
by Computer, SourceIP
| where FailedAttempts >= 3;

let SuccessfulLogons = Event
| where TimeGenerated > ago(30m)
| where EventID == 4624
| extend IpAddress = extract(@"<Data Name=""IpAddress"">([^<]+)</Data>", 1, EventData)
| extend SourceIP = replace_string(IpAddress, "::ffff:", "")
| extend Account = extract(@"<Data Name=""TargetUserName"">([^<]+)</Data>", 1, EventData)
| where SourceIP !in ("127.0.0.1", "::1");
```

Закінчення таблиці 3.5

```
FailedLogons
| join kind=inner SuccessfulLogons on SourceIP, Account
| where TimeGenerated > LastSeen
| project
    Computer,
    SourceIP,
    FailedAttempts,
    LastSeen,
    SuccessTime = TimeGenerated
```

Computer	SourceIP	Account	FailedAttempts
dc-01.internal.kb221shkyu.pp.ua	192.168.20.5	Administrator	51

Computer	dc-01.internal.kb221shkyu.pp.ua
SourceIP	192.168.20.5
Account	Administrator
FailedAttempts	51
FirstSeen [UTC]	2026-06-01T14:09:17.1907202Z
LastSeen [UTC]	2026-06-01T18:40:13.4884679Z

Рисунок 3.31 – Приклад отриманих даних

Створимо плейбук, який буде виконувати вимкнення облікового запису шляхом виконання скрипту на машині, яку атакували. Для точнішої перевірки скрипт самостійно буде переглядати групи, у яких знаходиться атакований обліковий запис, щоб не виконати вимкнення адміністратора. Після чого далі буде надіслано SMS повідомлення для подальшого розгляду та заблоковано IP-адресу.

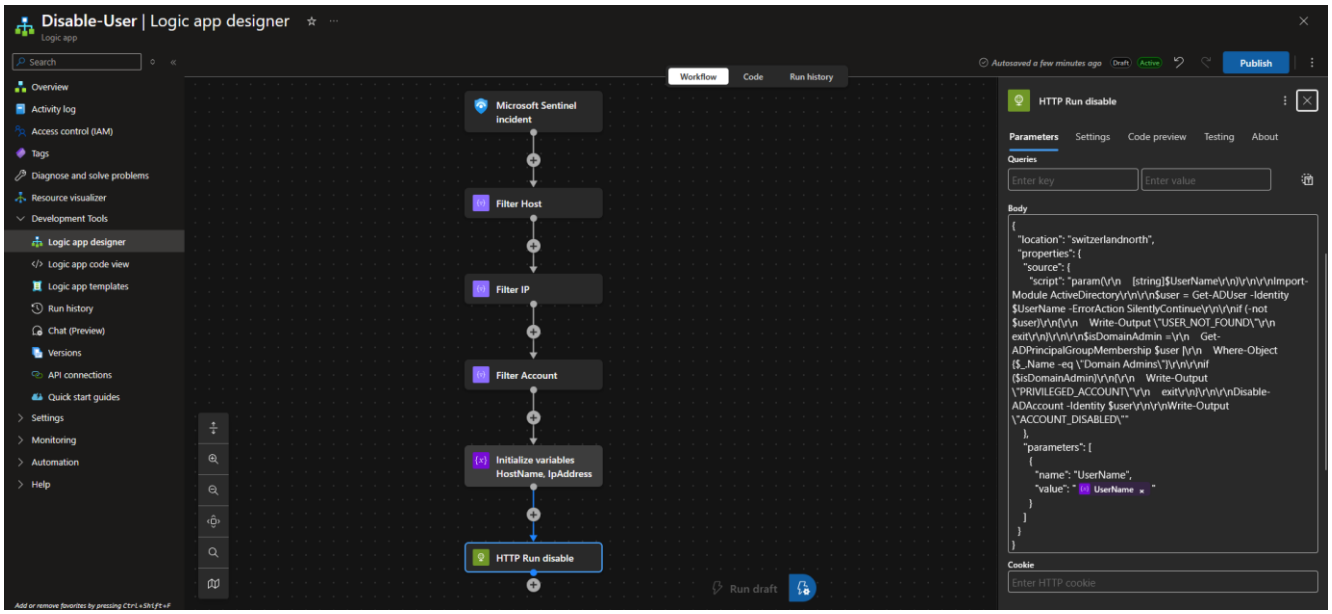


Рисунок 3.32 – Загальна логіка Disable-User

Логіка виконання плейбуків виглядає наступним чином:

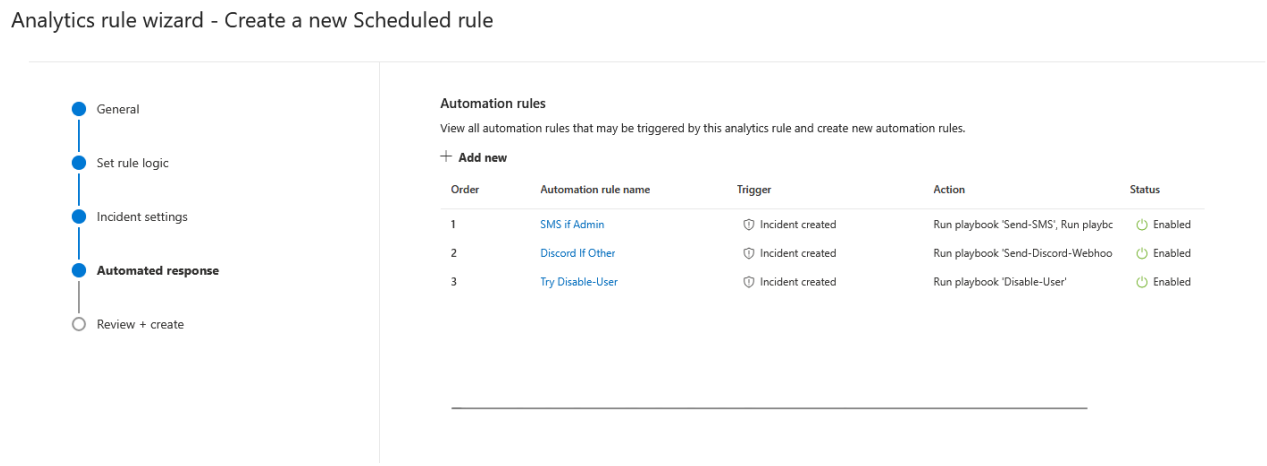


Рисунок 3.33 – Логіка виконання плейбуків

Використовуючи попередньо отримане ім’я користувача можемо визначати чи Адміністратор це. Перевірка наведена на рисунку 3.34.

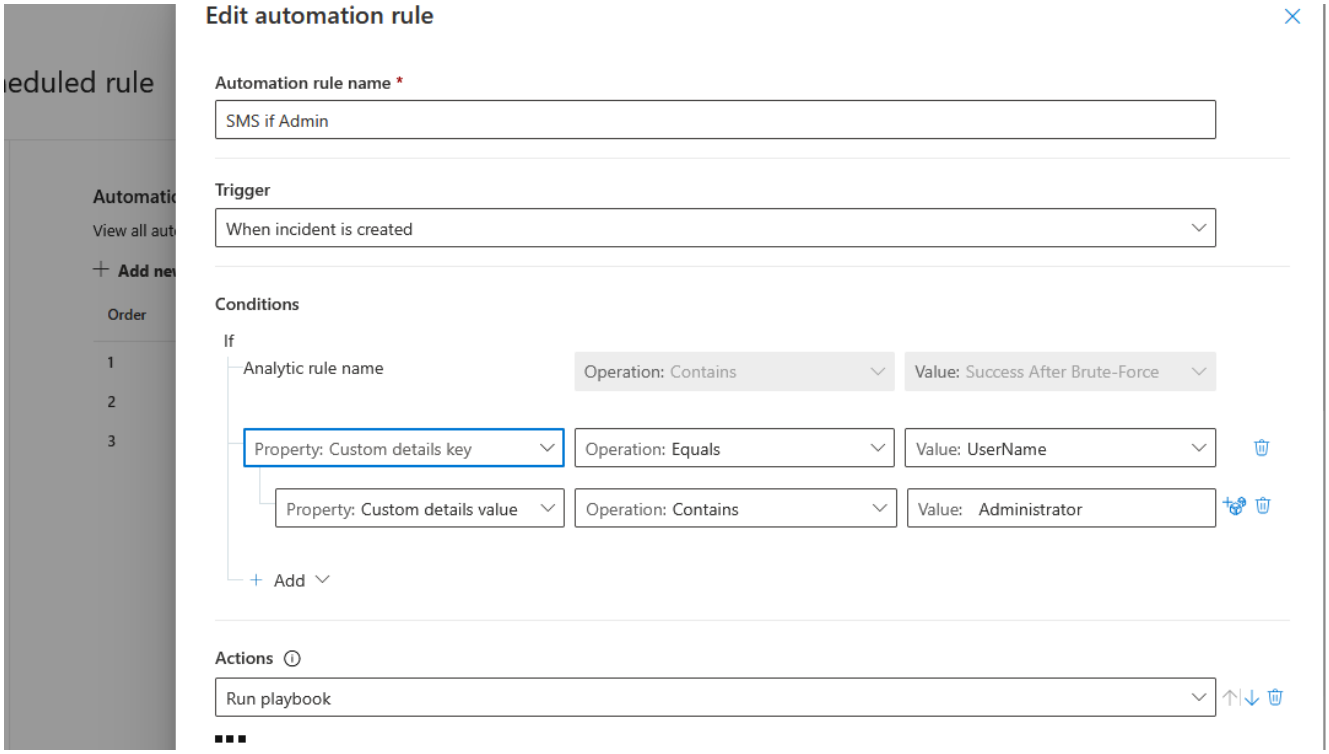


Рисунок 3.34 – Перевірка у правилі

3. Веб атаки різних видів

У нас вже є інструмент для визначення веб-атак у самому додатку, все, що нам залишається – це провести блокування IP-адреси зловмисника на cloudflare. Для цього створимо плейбук, який буде додавати правило блокування використовуючи Cloudflare API. Початковим етапом буде підготовка середовища Cloudflare,

створимо API Token, який діє тільки на зону kb221.shkyu.pp.ua, на порталі, який в подальшому будемо використовувати при запитах [24].

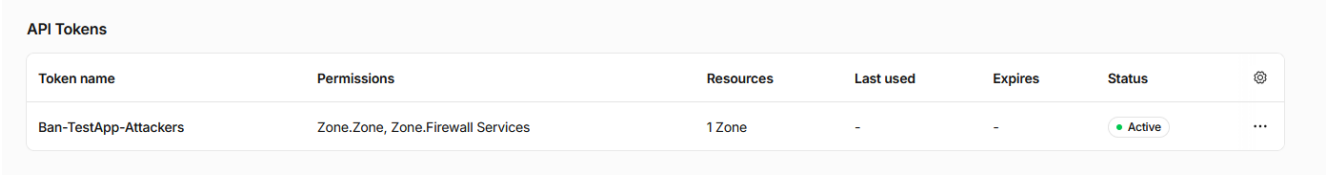


Рисунок 3.35 – Результат створення токену

Наступним кроком буде перевірка тестовим запитом

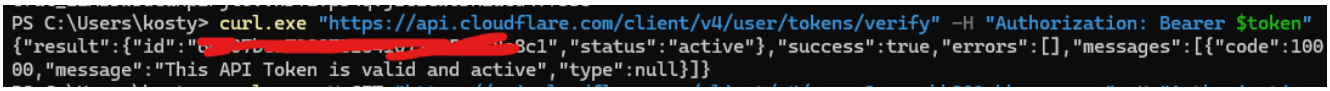


Рисунок 3.36 – Результат перевірки створеного токену

Токен створено, тепер можна переходити до створення плейбуку, назвемо його Ban-Ip-Cloudflare. Перед тим як створити, варто зазначити, що на платформі Cloudflare є і так вбудований WAF, який і відповідає за фільтрування трафіку і блокування зловмисної активності, але даний plug-in є платним, тому під час проектування було вирішено створити саме Logic App який буде блокувати IP запитом з додаванням відповідного адресу до чорного списку. Його логіка буде схожою до усіх інших плейбуків, лише в результаті буде виконуватись додавання ворожої IP-адреси до бан лісту. Для запиту нам також потрібно отримати id зони у Cloudflare.

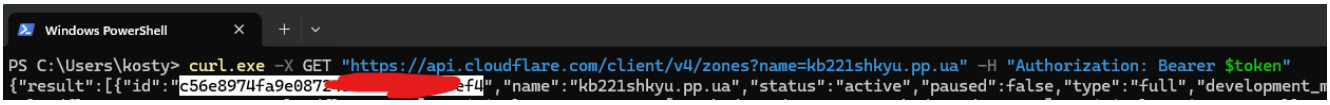


Рисунок 3.37 – Отриманий ідентифікатор зони kb221shkyu.pp.ua

Проведемо тестовий запит на додавання правила блокування IP, результат тестового запиту наведено на рис. 3.38.

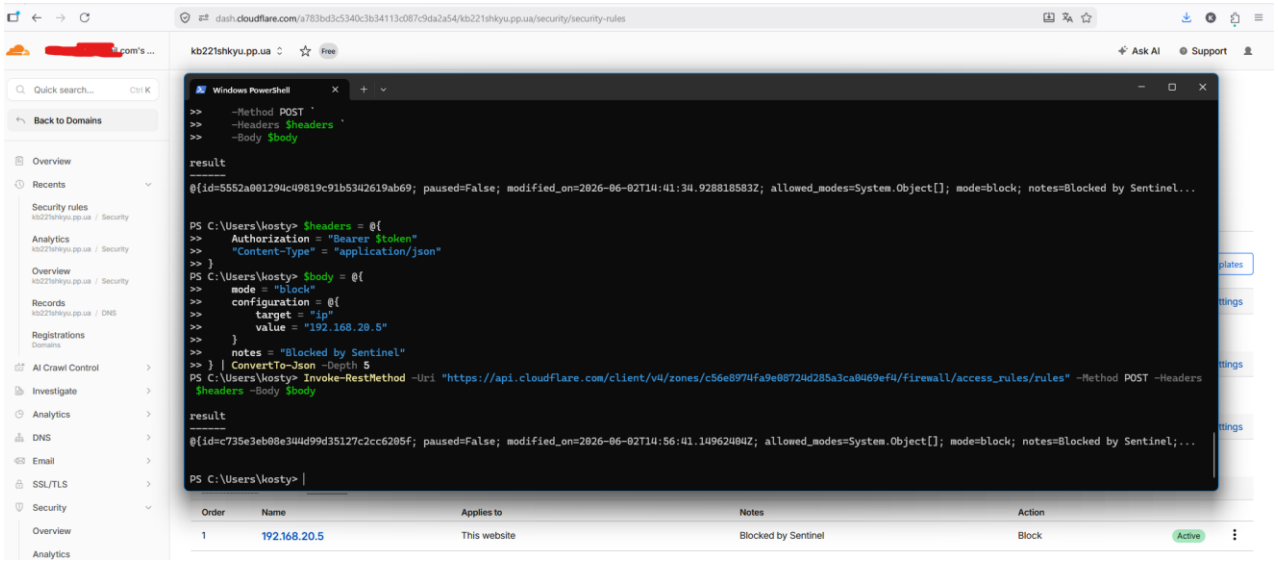


Рисунок 3.38 – Результат додавання тестового правила

Тепер можна використати даний запит у Ban-Ip-Cloudflare, загальна логіка плейбуку зображена на рисунку 3.39.

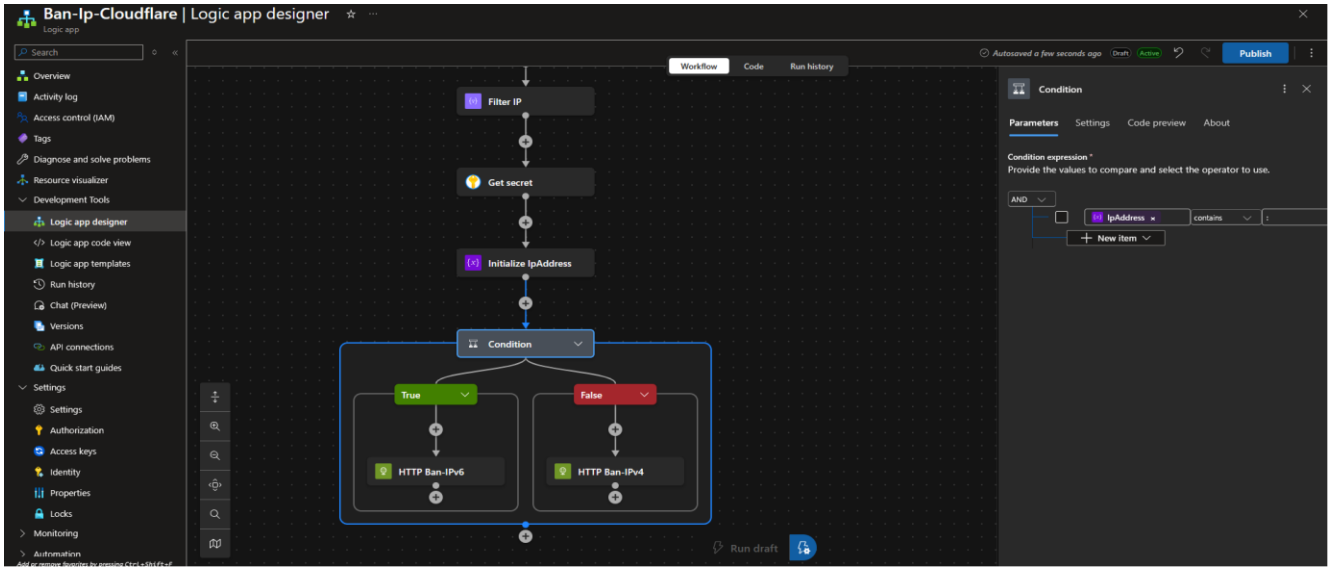


Рисунок 3.39 – Загальна логіка Ban-Ip-Cloudflare

Перейдемо до аналітичного правила, у ньому використаємо наступний KQL-запит зазначений у табл. 3.6

Таблиця 3.6

KQL-запит

```
Event
| where EventID == 0
| where Source == "RequestAuditMiddleware" or RenderedDescription has "AUDIT_EVENT"
| extend SrcIP = extract(@"\| IP:([^|]+)\|", 1, RenderedDescription)
| summarize AttackCount=count() by Computer, SrcIP
| where AttackCount >= 5
```

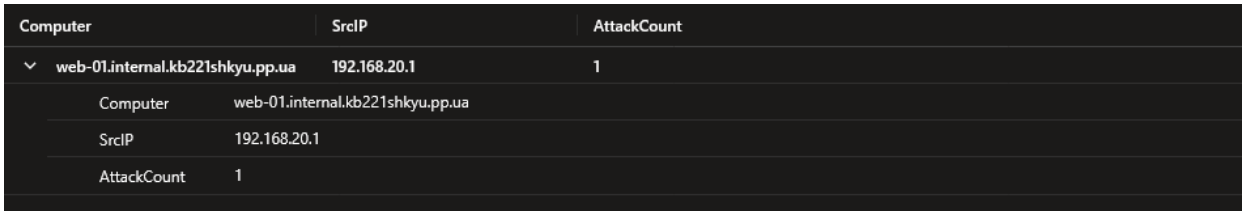


Рисунок 3.40 – Приклад отриманої події

Даний запит буде отримувати кількість атак за IP, якщо кількість зловмисних запитів перевищує 4. Запит буде виконуватись кожні 5 хвилин, і переглядати дані за останні 5 хвилин. В результаті генерації інциденту, буде заблоковано зловмисну IP-адресу та надіслано повідомлення на Telegram.

4. Виявленні вірусу на робочій станції

Сценарій простий, при виявленні подій 1116 блокуємо трафік на інфікованому хості. KQL-запит виглядає наступним чином:

Таблиця 3.7

KQL-запит

```
Event
| where TimeGenerated > ago(5m)
| where EventLog == "Microsoft-Windows-Windows Defender/Operational"
| where EventID == 1116
| summarize Count = count() by Computer
```

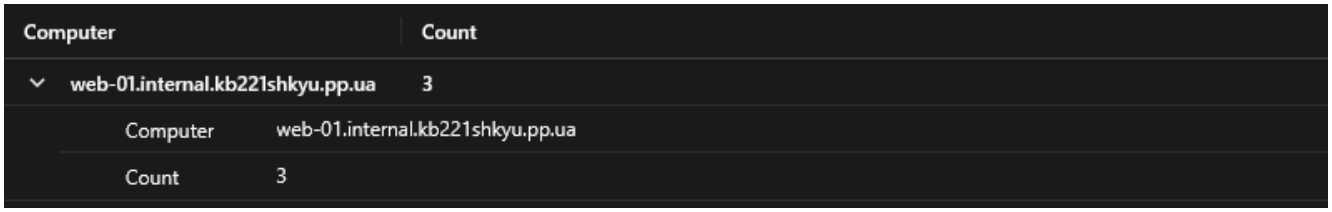


Рисунок 3.41 – Приклад результату виконання запиту

Створимо плейбук, який буде виконувати ізоляцію хосту від усієї мережі, шляхом виконання PowerShell скрипту для зміни профілю локального брандмауеру. У самому правилі перевіряємо вхід

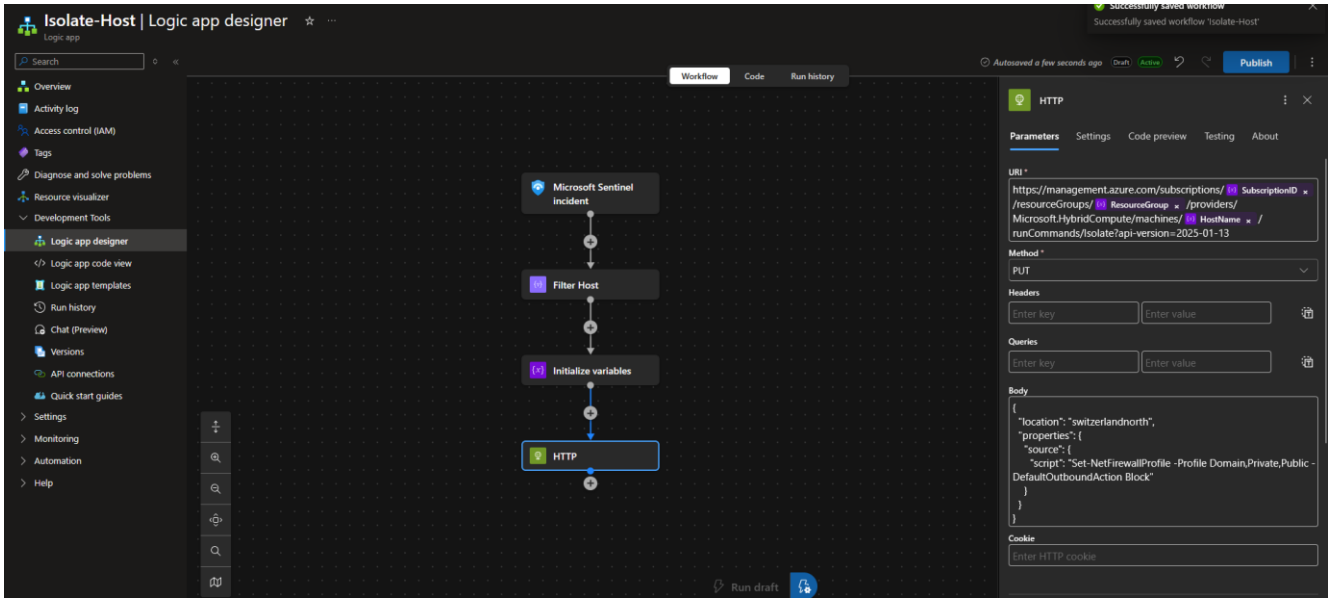


Рисунок 3.42 – Загальна логіка Isolate-Host

5. Виявлення створення "Backdoor" користувача

Загальна логіка схожа на другий сценарій, але при цьому нам потрібно одразу перевіряти ще створення облікового запису після успішного підбору паролю. KQL-запит наведено у табл. В.8. Приклад результату наведено на рис. 3.43.

Computer	SourceIP	FailedAttempts	Account	SuccessTime [UTC]	NewUser	CreatedTime [UTC]
dc-01.internal.kb221shkyu.pp.ua	192.168.20.5	11	Administrator	6/2/2026, 8:56:02.603 PM	eviluser	6/2/2026, 8:56:06.889 PM

Computer	dc-01.internal.kb221shkyu.pp.ua
SourceIP	192.168.20.5
FailedAttempts	11
Account	Administrator
SuccessTime [UTC]	2026-06-02T20:56:02.6038722Z
NewUser	eviluser
CreatedTime [UTC]	2026-06-02T20:56:06.8897911Z

Рисунок 3.43 – Приклад результату запиту

Створимо одразу аналітичне правило з даним запитом, та призначимо на виконання раніше створені плейбуки.

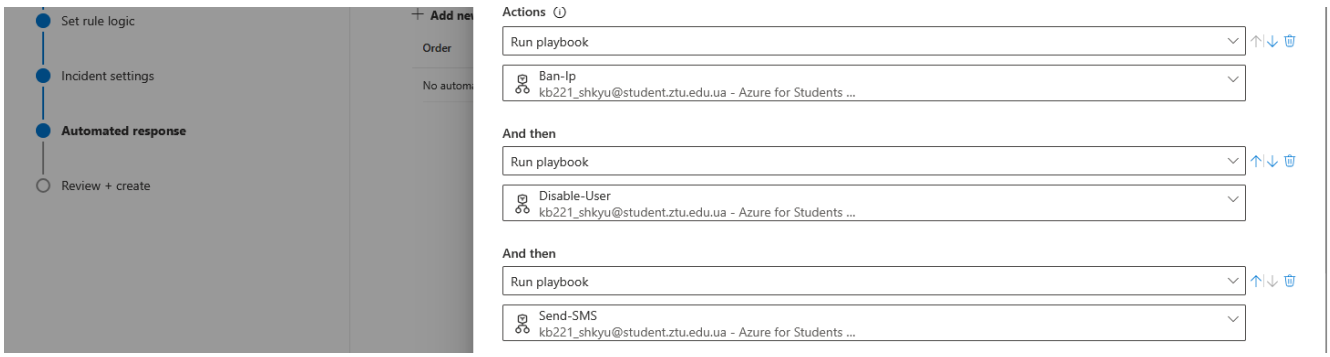


Рисунок 3.44 – Використання відповідних плейбуків

3.3 Експериментальна перевірка ефективності запропонованого рішення та аналіз результатів тестування

2. Brute-Force на обліковий запис домену

Змоделюємо атаку, шляхом генерації невдалих входів, для генерації використаємо станцію WS-01 та PowerShell. Генерувати будемо за допомогою наступного скрипту відправляє 10 мережних запитів до ресурсу на DC-01 від імені адміністратора, при цьому вказавши випадковий пароль:

Таблиця 3.8

Скрипт генерації подій

```
1..10 | ForEach-Object {
   net use \\DC-01\IPC$ /user:internal\Administrator WrongPassword$_
   net use * /delete /y
}
```

Після введення неправильного паролю, у системі генерується послідовність подій 4771 (Kerberos Fail) з віддаленим хостом у вигляді WS-01, після чого, KQL-запит у аналітичному правилі поверне наступну таблицю, яка зображена на рис. 3.45

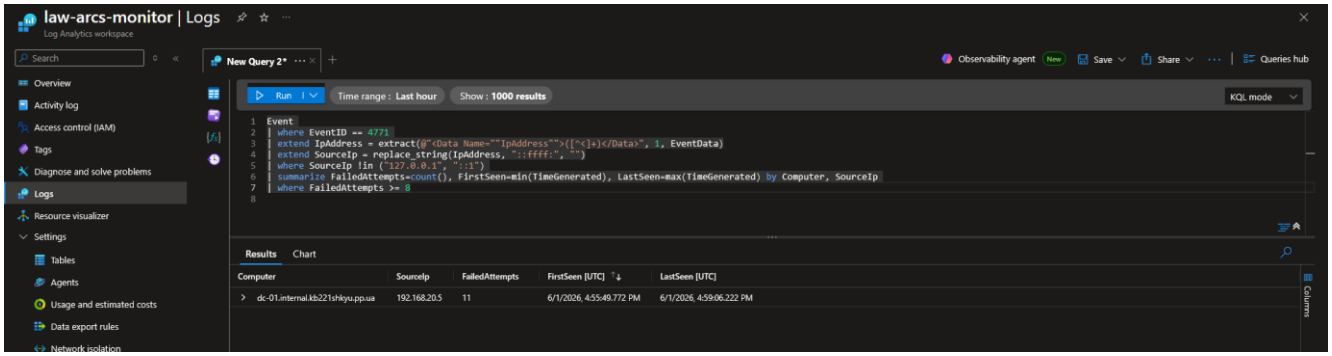


Рисунок 3.45 – Результат виконання запиту

Після проведення атаки спрацювало аналітичне правило, KQL-запит повернув результат, і аналітичне правило створило інцидент, запустивши при цьому відповідні плейбуки інформування і реагування, результат виконання інформування наведено на рис. 3.47.

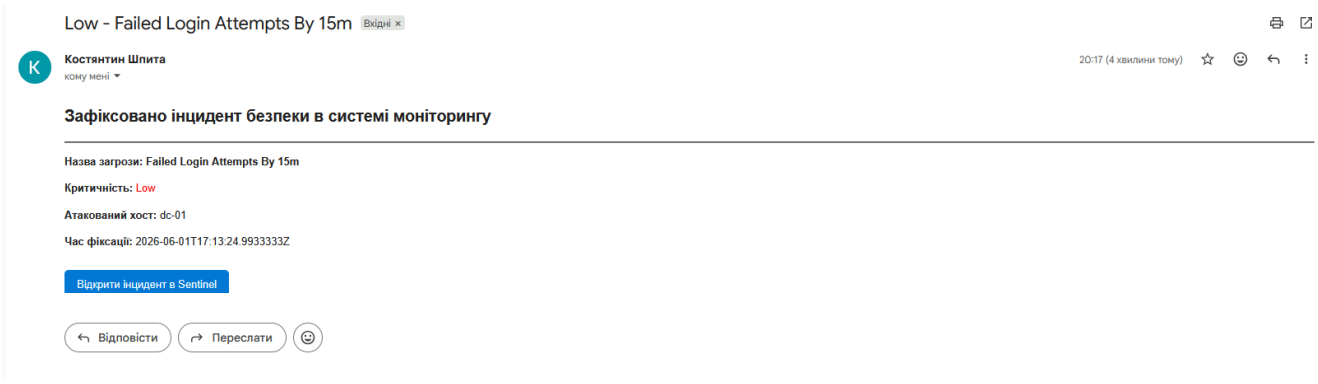


Рисунок 3.47 – Отримане повідомлення на електронну пошту

Результат запуску дій на реагування, зображено на рис. 3.48-49.

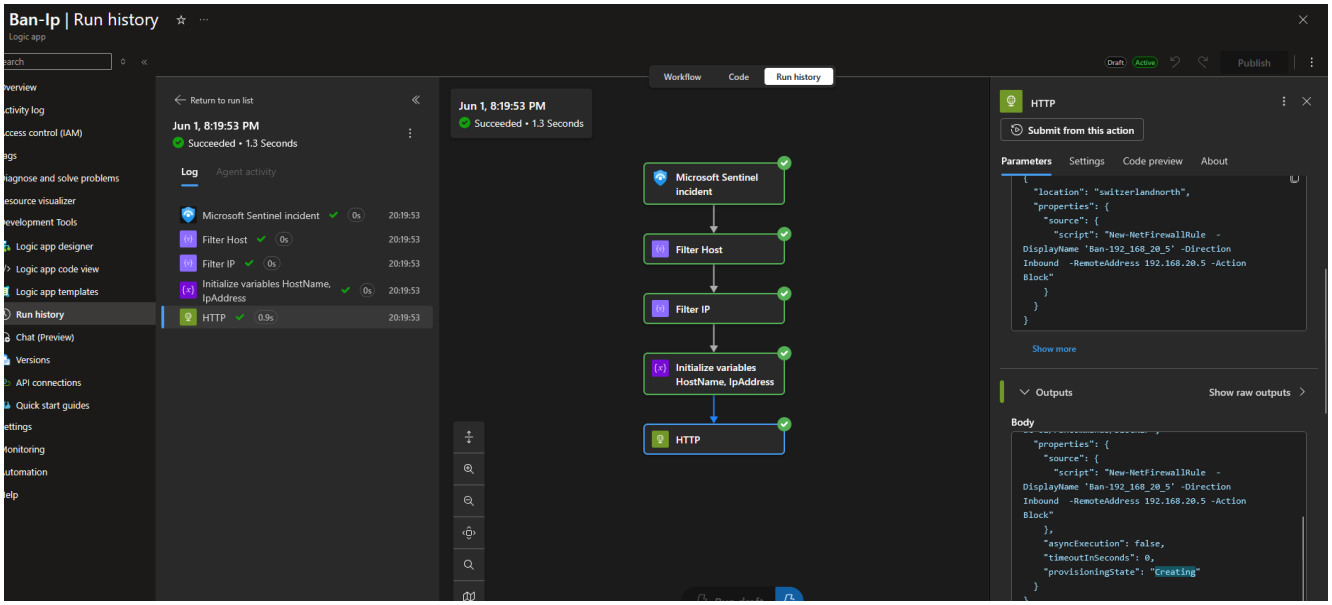


Рисунок 3.48 – Результат виконання Ban-Ip

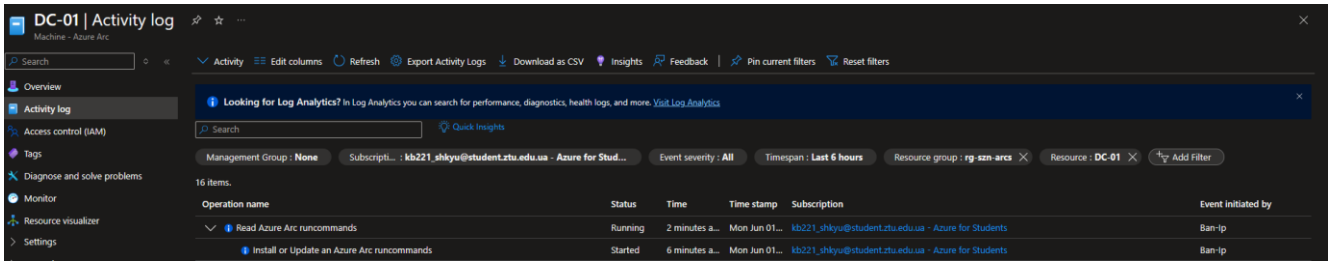


Рисунок 3.49 – Результат виконання Ban-Ip

В результаті виконання скрипту блокування було додано відповідне правило у Windows Defender Firewall на атакованому хості. Результат додавання правило зображено на рис. 3.50.

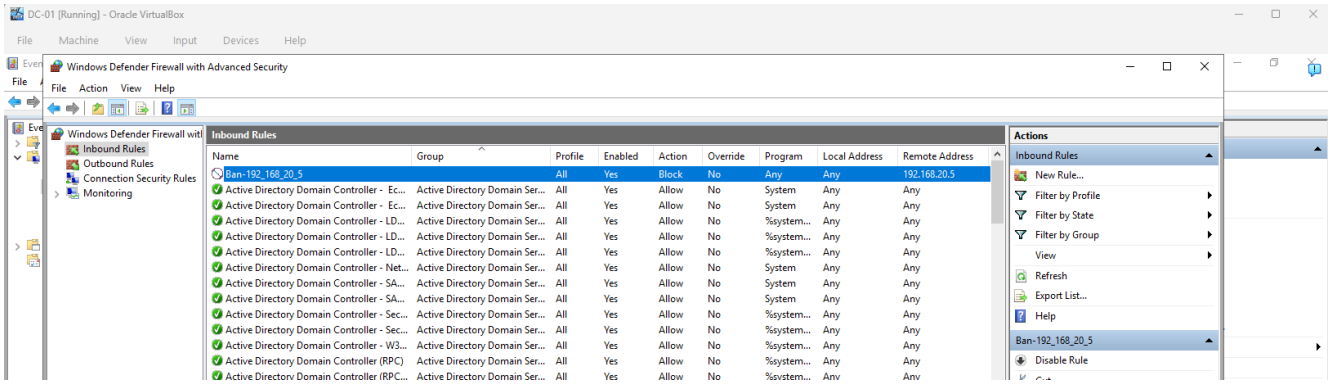


Рисунок 3.50 – Додане правило в результаті виконання команди Ban-Ip

2. *Компрометація облікового запису*

Змоделюємо дану атаку таким самим чином, як і попередню, тільки у даному випадку, невдалих спроб зробимо 4, і після чого успішно увійдемо. Така послідовність дій згенерує саме ту послідовність подій, яка нам необхідна для спрацювання правила. Для початку перевіримо реагування на вхід до облікового запису адміністратора.

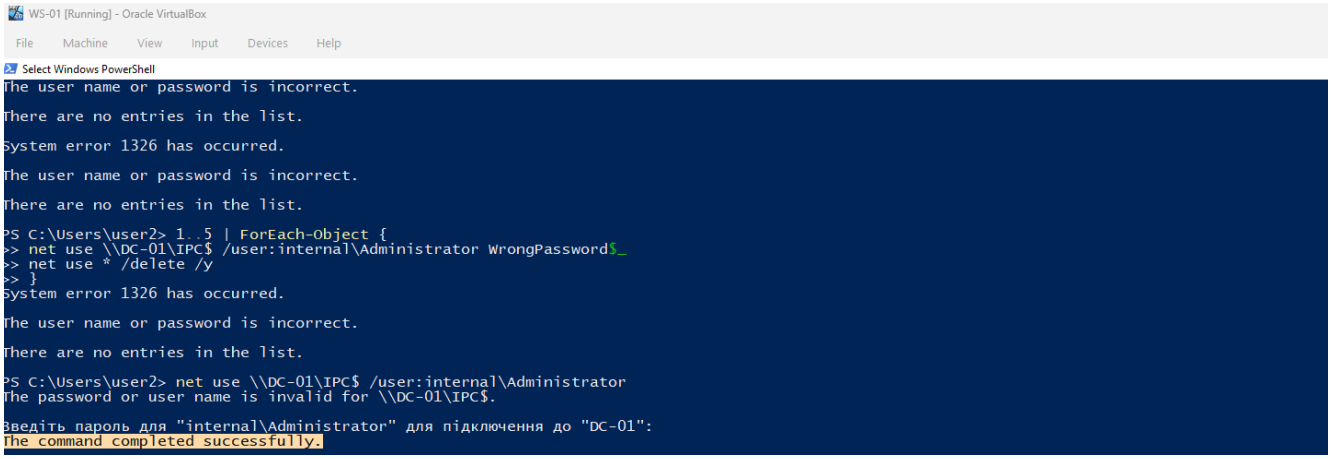


Рисунок 3.51 – Генерація потрібної послідовності подій

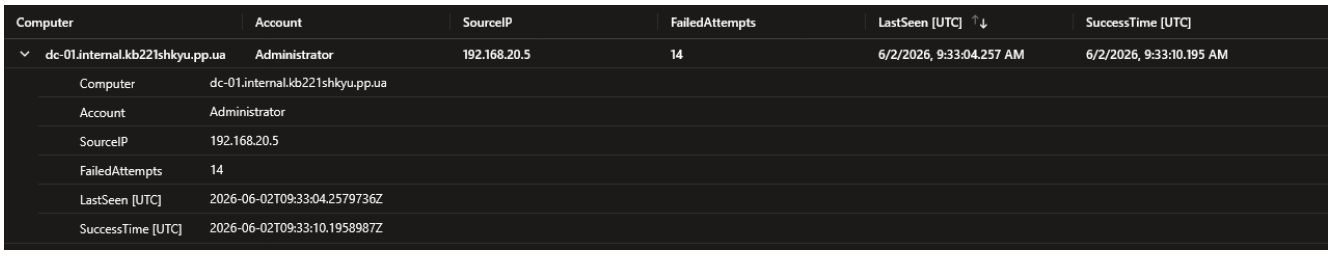


Рисунок 3.52 – Фіксація потрібної послідовності

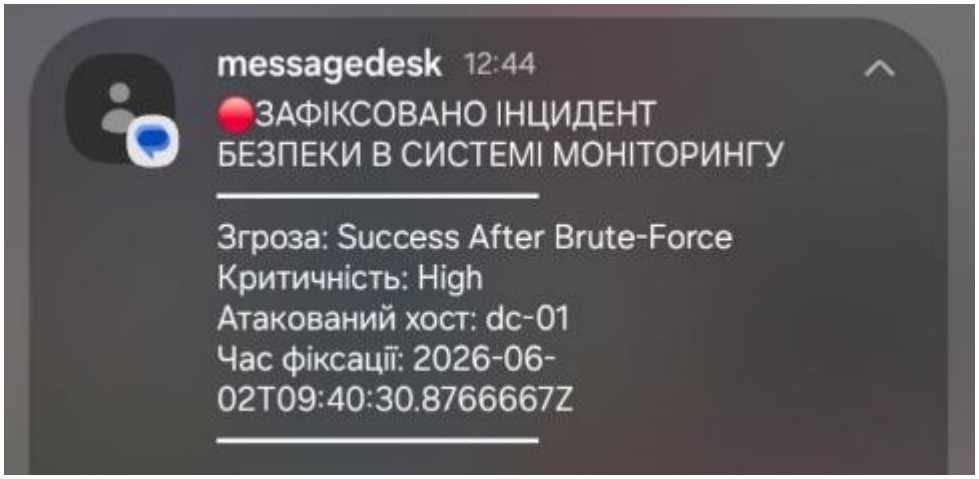


Рисунок 3.53 – Отримане повідомлення про інцидент за допомогою SMS

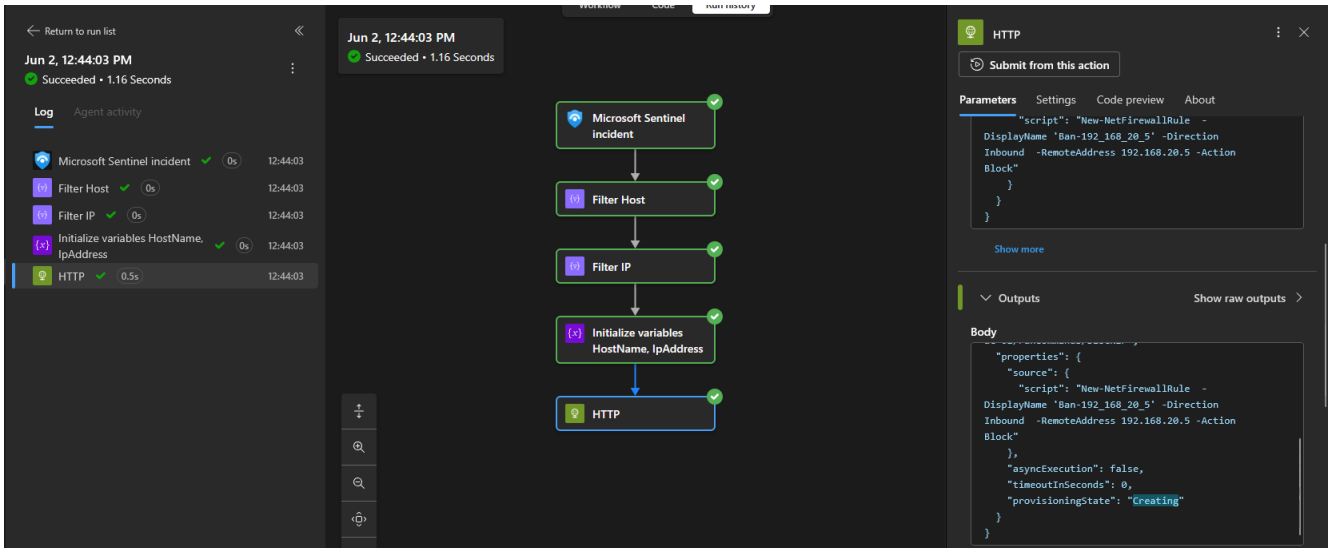


Рисунок 3.54 – Результат спрацювання Ban-Ip

В результаті було заблоковано IP-адресу атакуючого, і надіслано SMS, бо було атаковано обліковий запис адміністратора. Далі також було виконано спробу вимкнення користувача, але так як це адміністратор домену, вимкнення не відбудеться, щоб не створювати ситуацію софт-локу. Результат виконання Logic App для вимкнення користувача зображено на рис. 3.55.

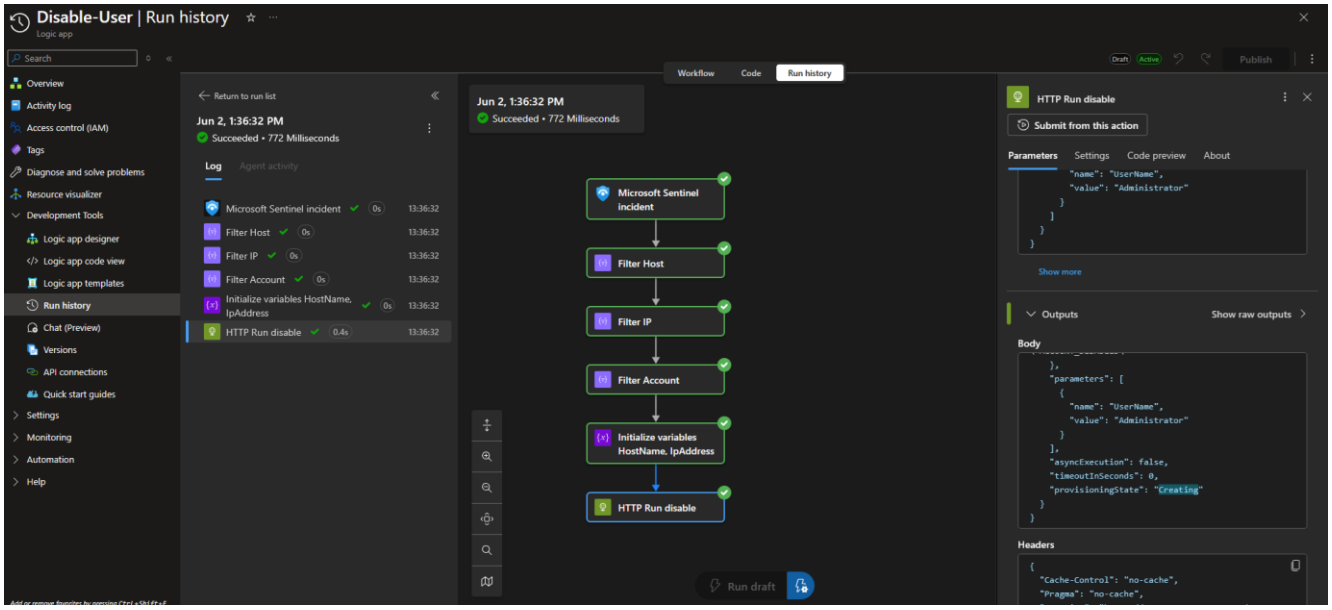


Рисунок 3.55 – Результат спрацювання Disable-User

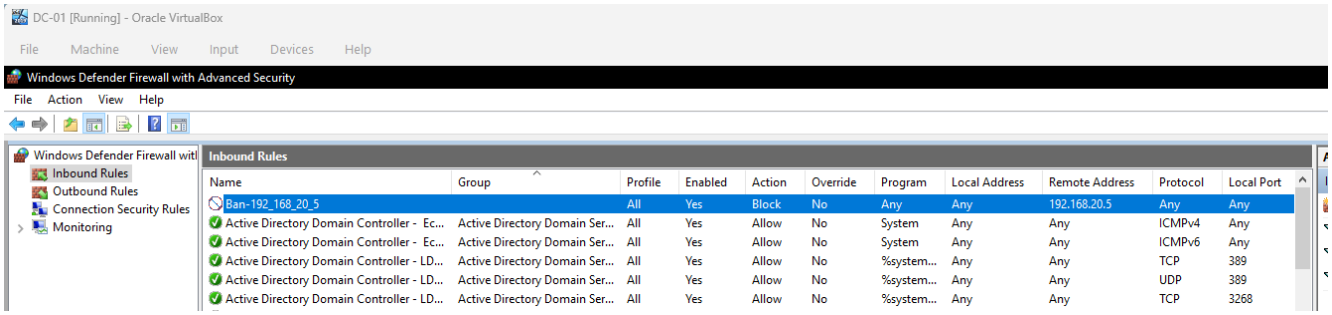


Рисунок 3.56 – Створене правило (нове) після виконання Ban-Ip

Спробуємо виконати такий вхід з облікового запису звичайного користувача, та переглянемо як відбудеться реагування. Одразу після генерації інциденту отримуємо повідомлення у Discord. Результат інформування зображено на рис. 3.57.

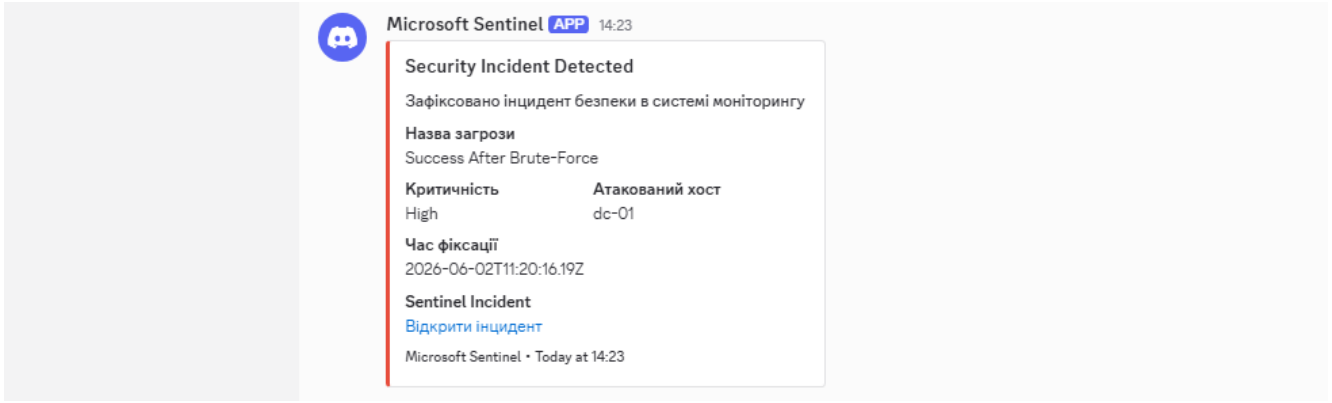


Рисунок 3.57 – Отримане сповіщення про інцидент у Discord

Наступними діями є блокування зловмисної IP та спроба вимкнення атакованого облікового запису. Результат виконання плейбуків зображено на рис. 3.58-59.

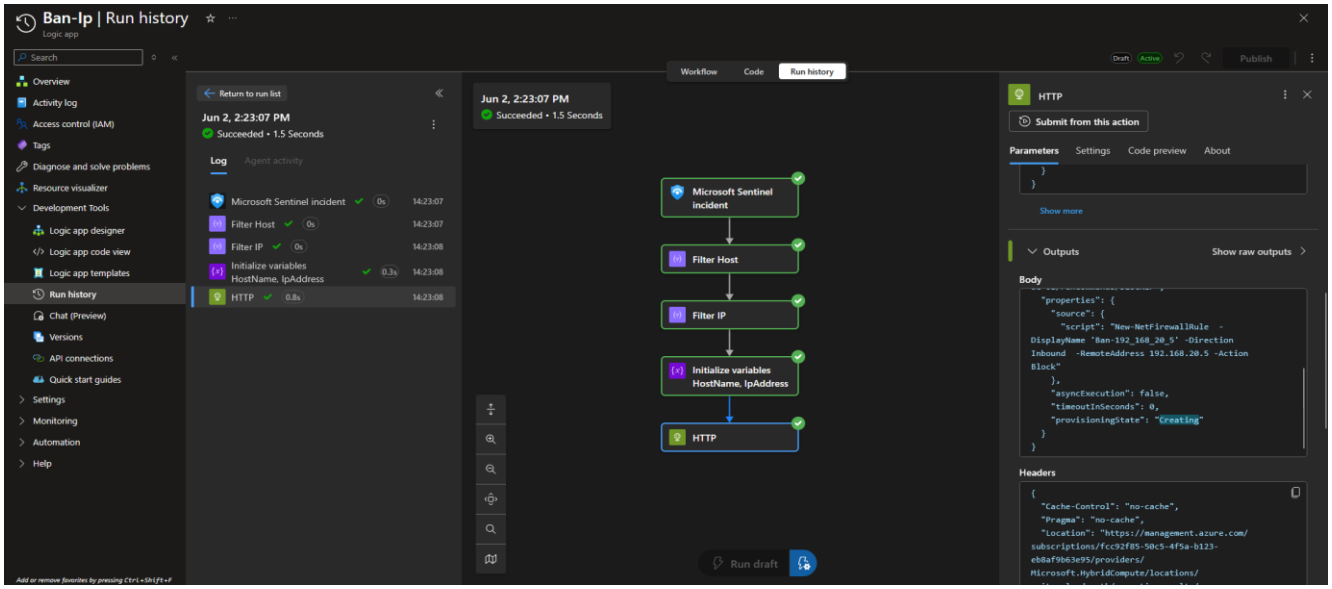


Рисунок 3.58 – Результат виконання Ban-Ip

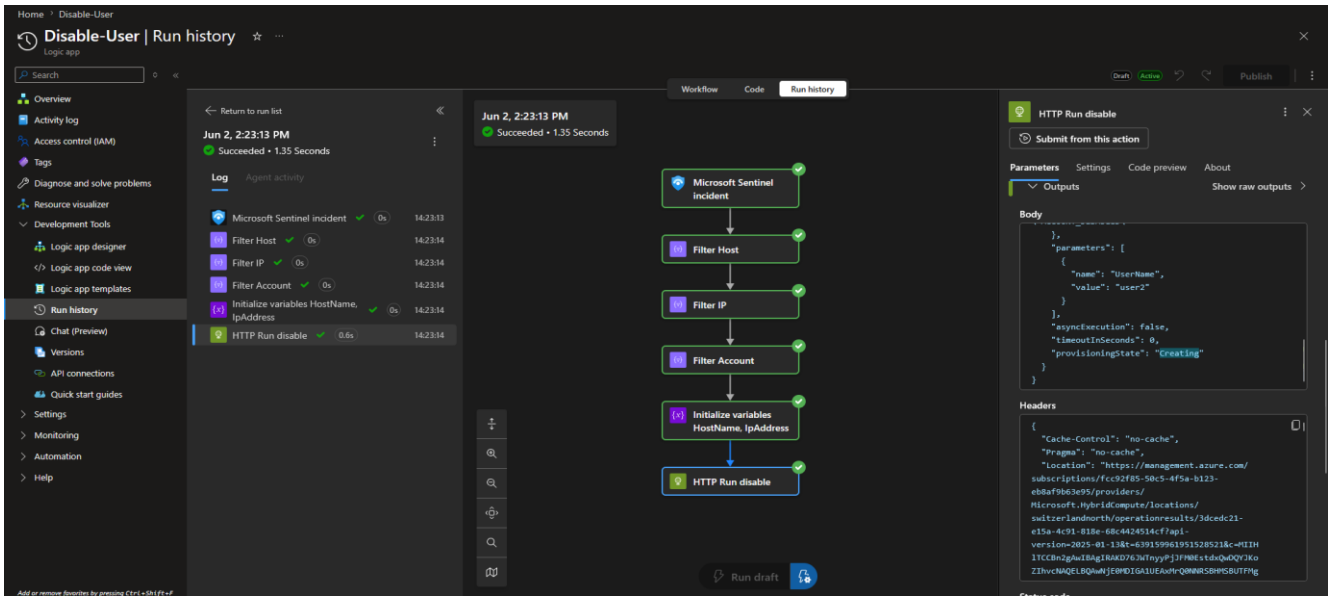


Рисунок 3.59 – Результат виконання Disable-User

Результат реагування зображено на рис.3.60-61.



Рисунок 3.60 – Отриманий статус виконання скрипту

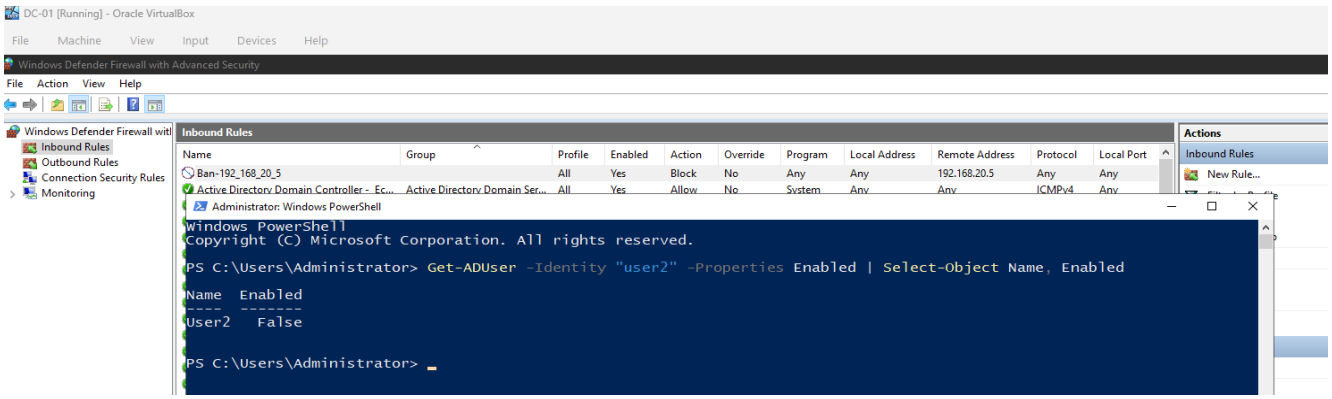


Рисунок 3.61 – Перевірка результату виконання скрипту на фоні нового правила від Ban-Ip

3. *SQLi та XSS*

Використовуючи VPN та веб-браузер, проведемо серію веб-атак на додаток. Спробуємо використати SQL-ін'єкцію та XSS, враховуючи функціонал вебдодатку, цього буде достатньо для перевірки блокування. Почнемо з ін'єкції, використаємо *' OR 1=1 --* для генерації. Результат серії запитів зображено на рисунку 3.62

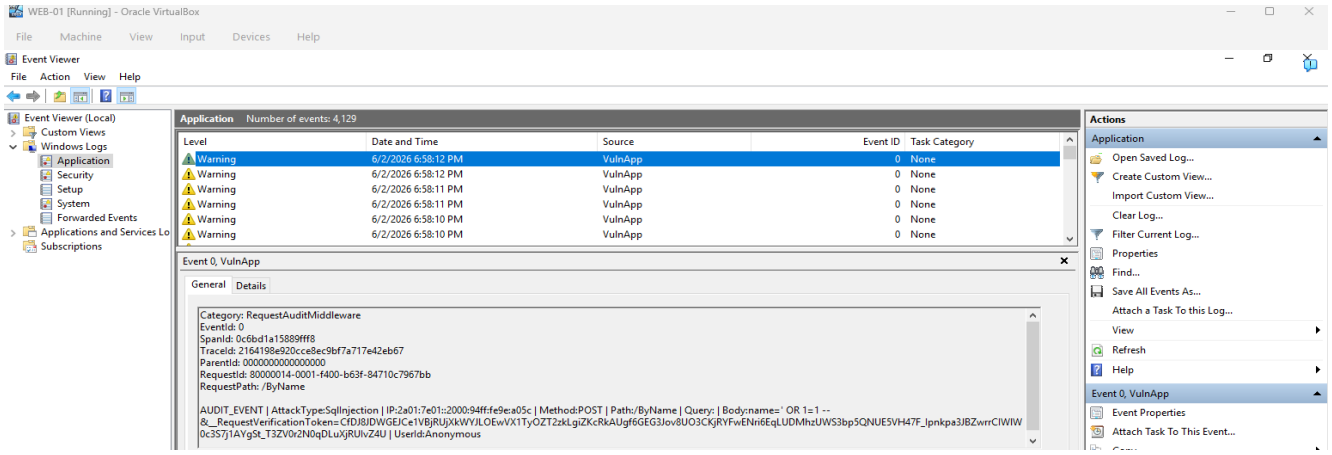


Рисунок 3.62 – Результат виконання серії запитів

Одразу після створення інциденту отримуємо повідомлення про нього у Telegram:

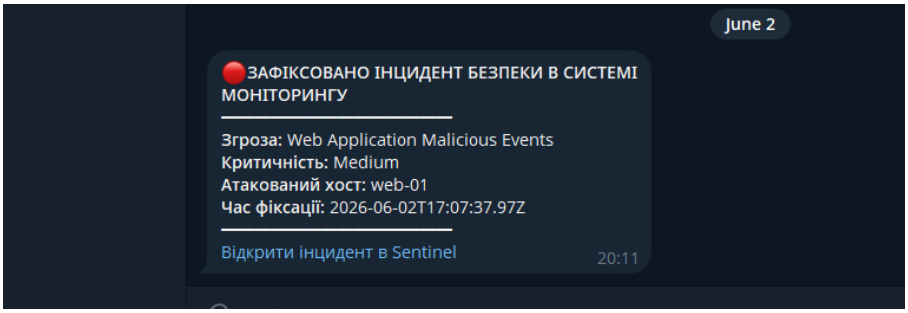


Рисунок 3.63 – Отримане повідомлення про інцидент від Telegram-бота

Одразу після інформування було надіслано запит на блокування відповідної адреси, результат виконання Ban-Ip-Cloudflare зображено на рис. 3.64

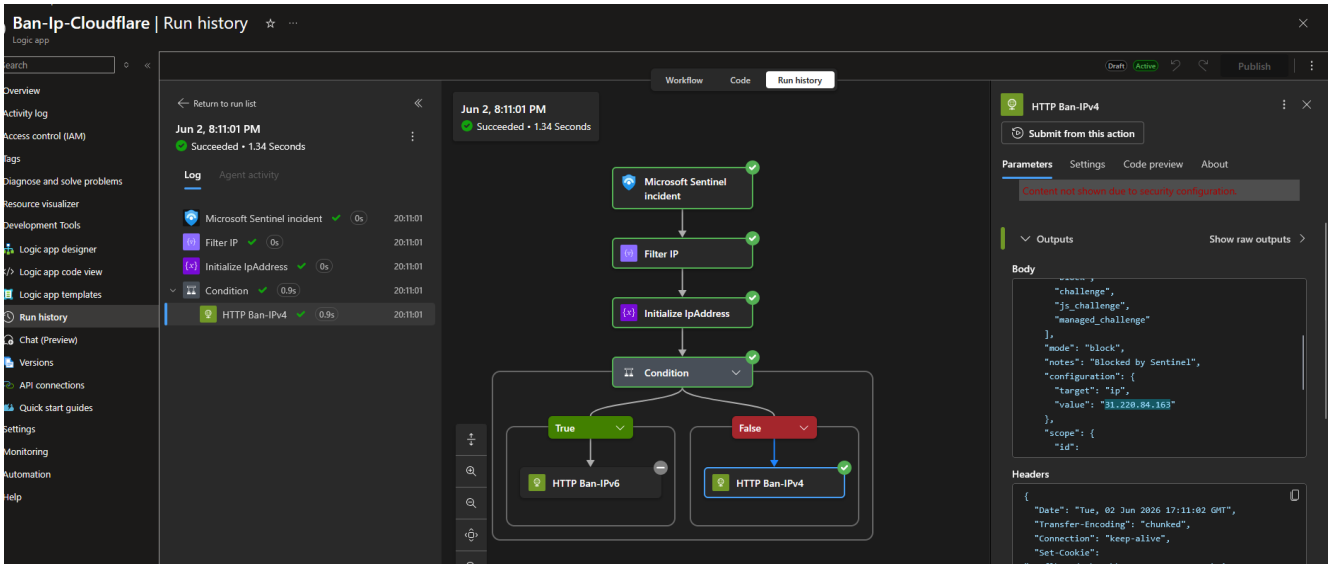


Рисунок 3.64 – Результат виконання Ban-Ip-Cloudflare

Після виконання плейбуку маємо нові заблоковані адреси на порталі Cloudflare:



Рисунок 3.65 – Результат блокування адреси за допомогою Ban-Ip-Cloudflare

Перевіримо, чи дійсно нашу адресу було заблоковано, результат перевірки зображено на рис. 3.66

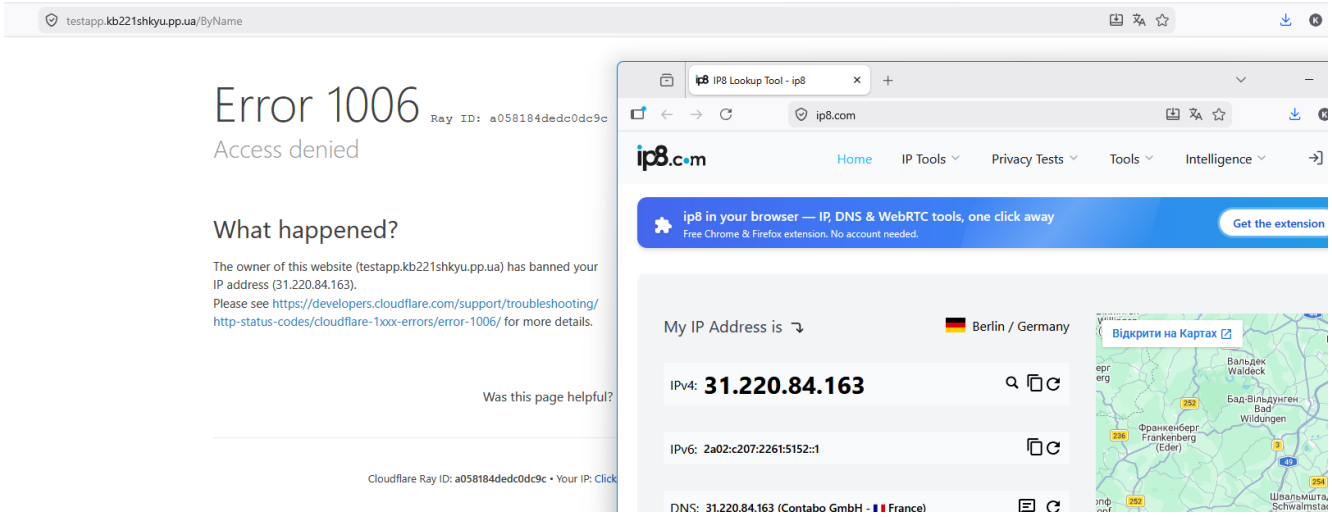


Рисунок 3.66 – Результат блокування адреси

Для XSS використаємо наступне корисне навантаження у секції коментарів до товару *<script>alert(1)</script>*

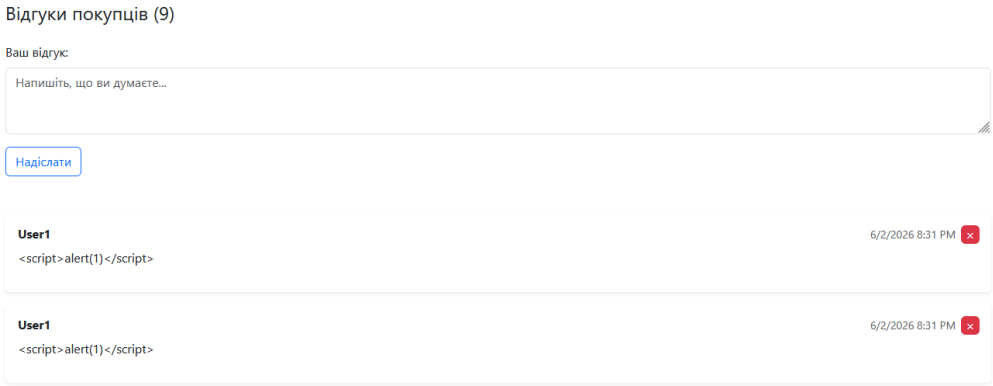


Рисунок 3.67 – Результат використання корисного навантаження

Після використання маємо наступні події з фіксацією атаки типу XSS:

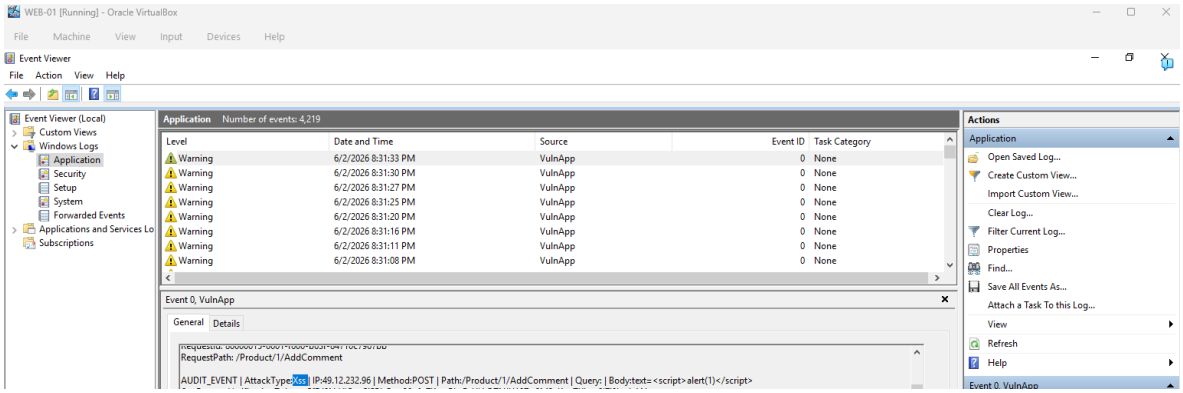


Рисунок 3.68 – Згенеровані події після XSS атаки

Одразу після отримання подій спрацьовує аналітичне правило з генерацією інциденту.

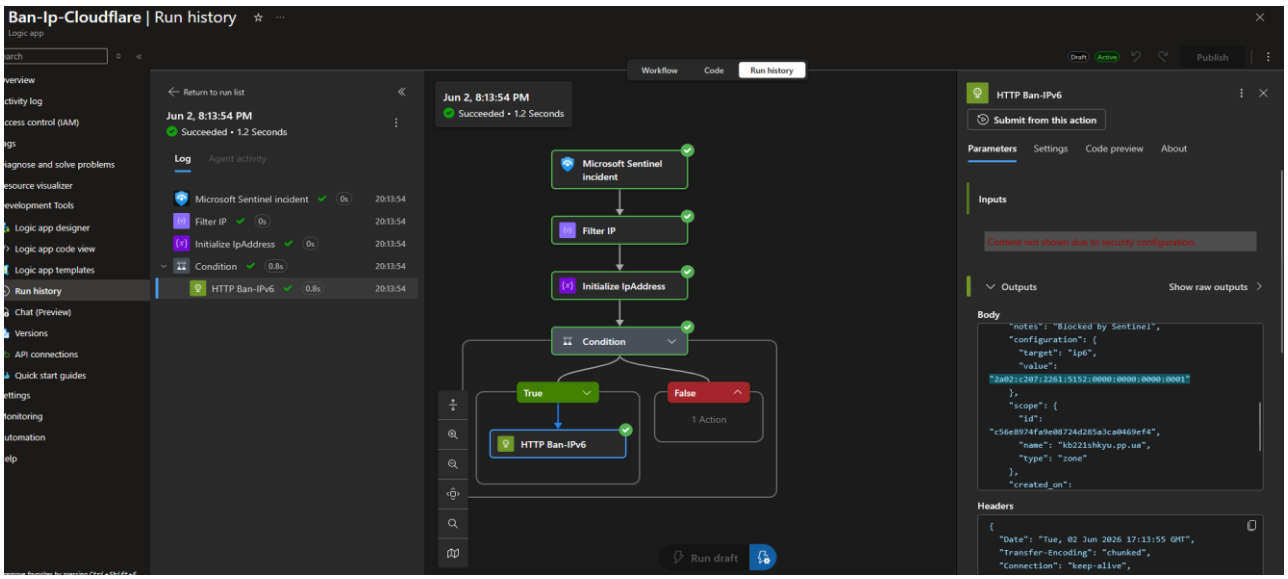


Рисунок 3.69 – Результат виконання Ban-Ip-Cloudflare



Рисунок 3.70 – Додані в результаті виконання Ban-Ip-Cloudflare адреси

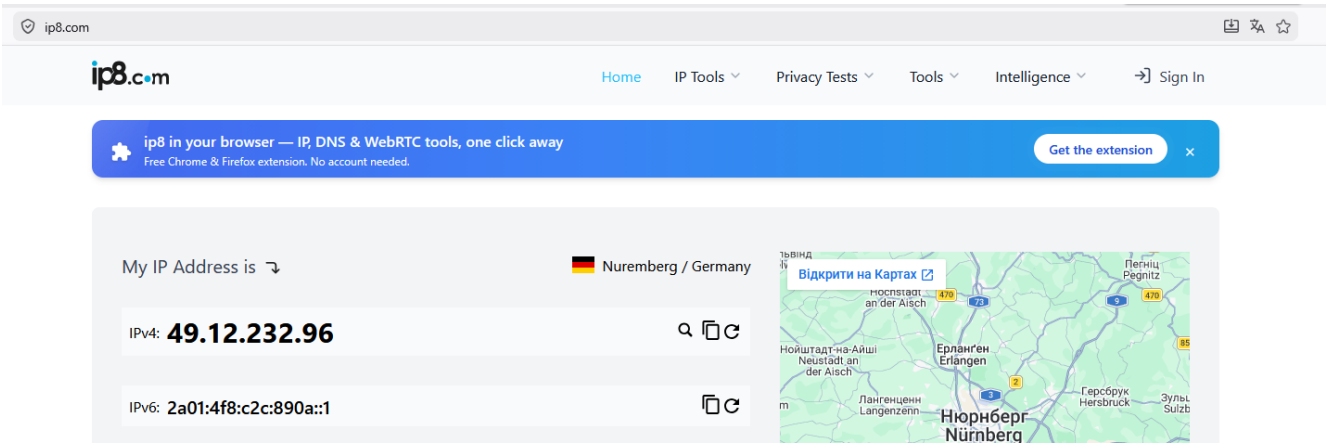


Рисунок 3.71 – Звіряння з новими записаними адресами

4. Виявленні вірусу на робочій станції

Для перевірки реагування на спрацювання створимо файл з тестовою сигнатурою для перевірки антивірусу. Файл містить у собі наступну сигнатуру: *X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H**. В результаті створення файлу з таким вмістом, Windows Defender одразу згенерує подію про фіксацію потенційно небажаного вмісту і подію 1116. Результат створення файлу з таким вмістом зображено на рисунку 3.72.

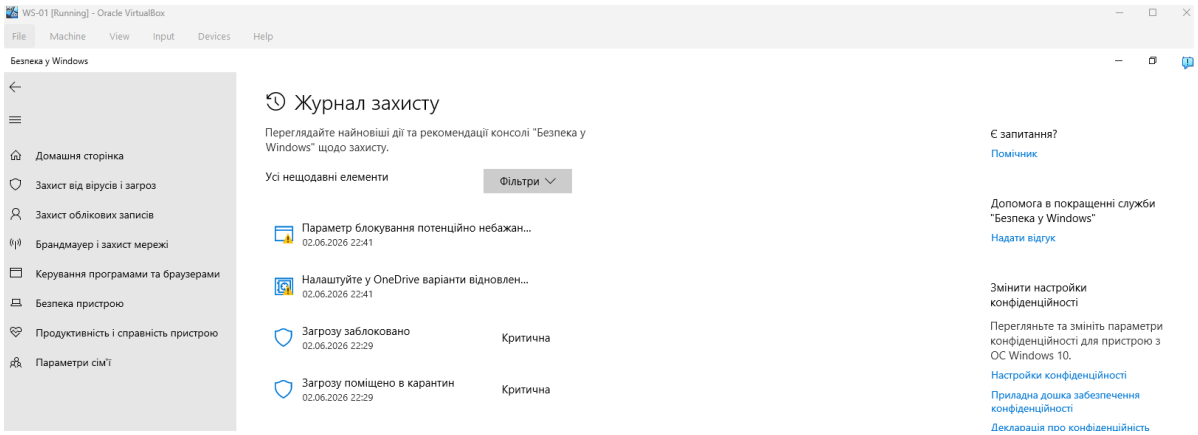


Рисунок 3.72 – Результат створення файлу на робочому столі

В результаті створення одразу отримуємо автоматизоване повідомлення на Discord. Результат отримання зображено на рис. 3.73.

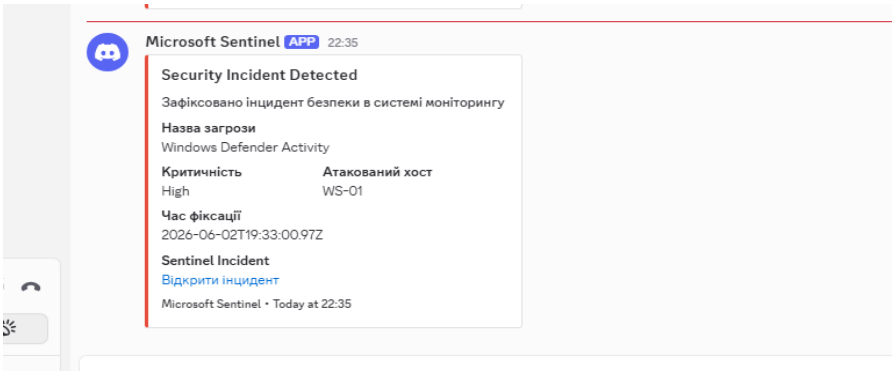


Рисунок 3.73 – Отримане сповіщення про інцидент у Discord

Результат виконання автоматизованої ізоляції хосту зображено на рис. 3.74.

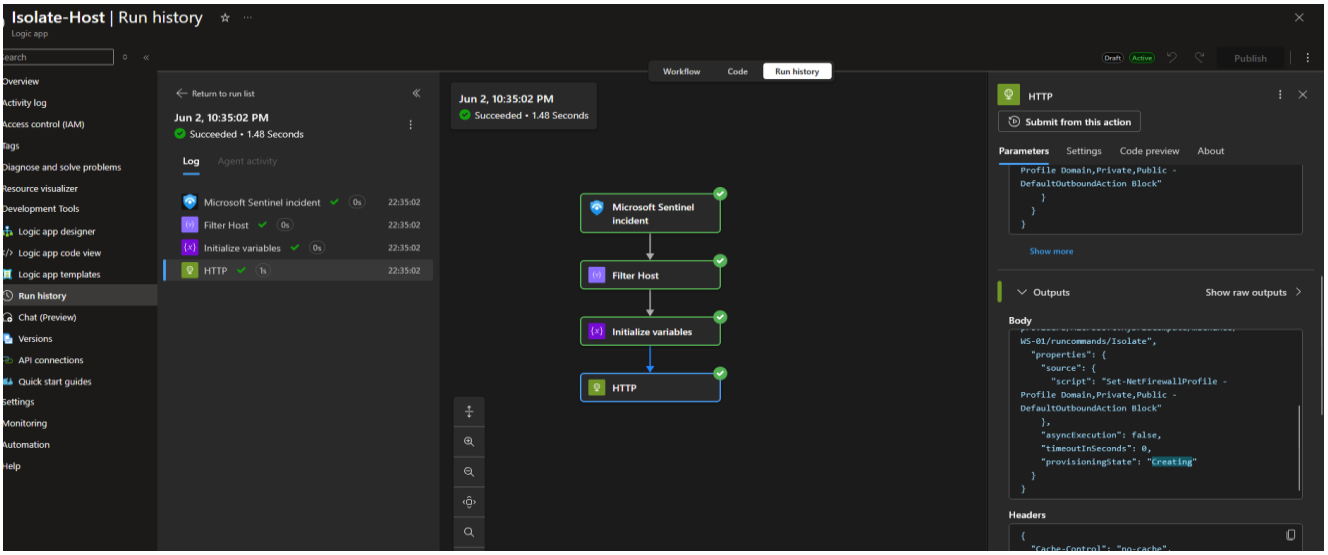


Рисунок 3.74 – Результат виконання Isolate-Host

Для перевірки ізоляції на робочій станції, спробуємо виконати ping до DNS-серверу google, та переглянемо поточний профіль Windows Defender Firewall.

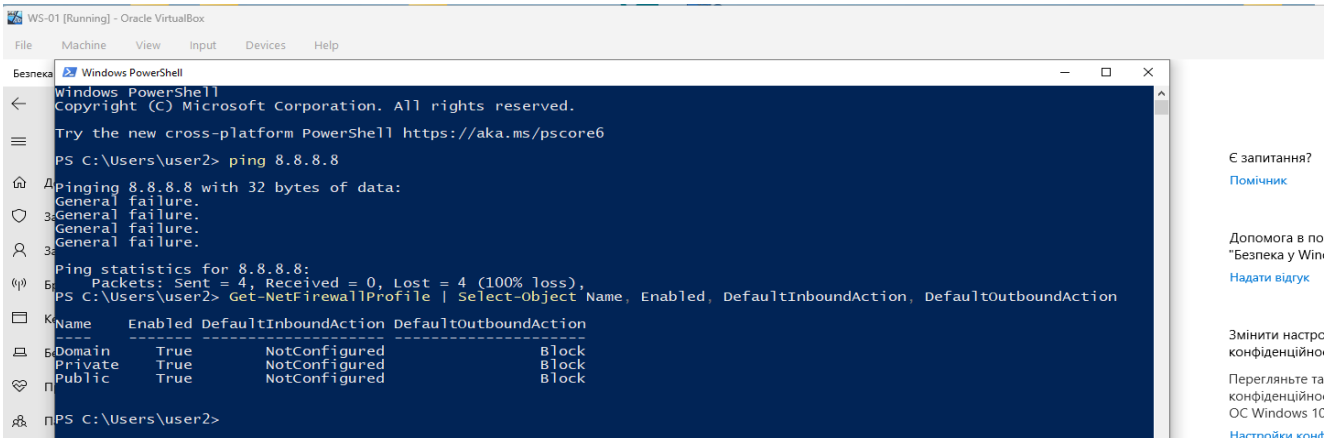


Рисунок 3.75 – Результат ізоляції

На рис. 3.75 бачимо, що ізоляція виконана успішно.

5. *Виявлення створення Backdoor користувача*

Для створення потрібної послідовності подій, використаємо метод зі сценарію номер 2, тільки після нього, додаємо створення нового облікового запису, результат виконання зображено на рисунку 3.76

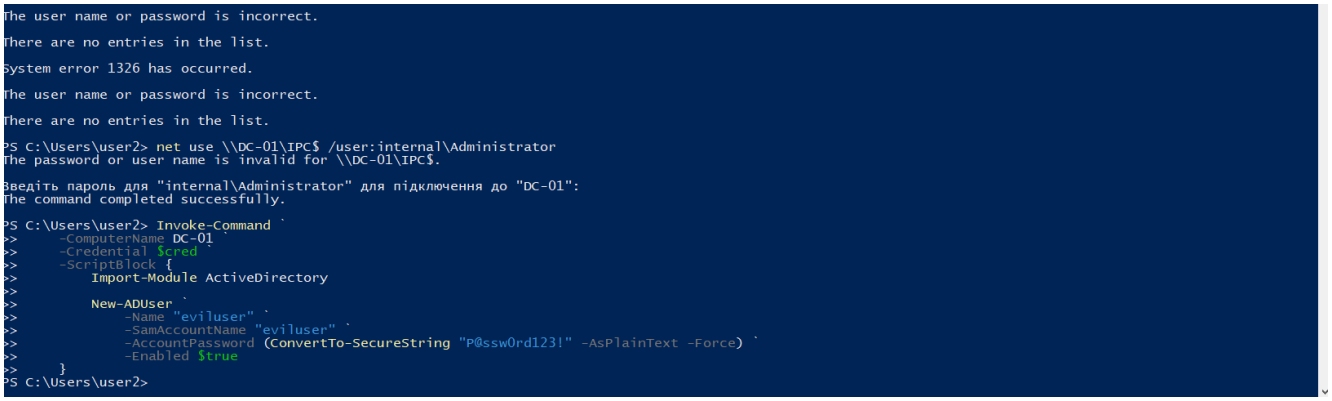


Рисунок 3.76 – Результат виконання відповідних дій

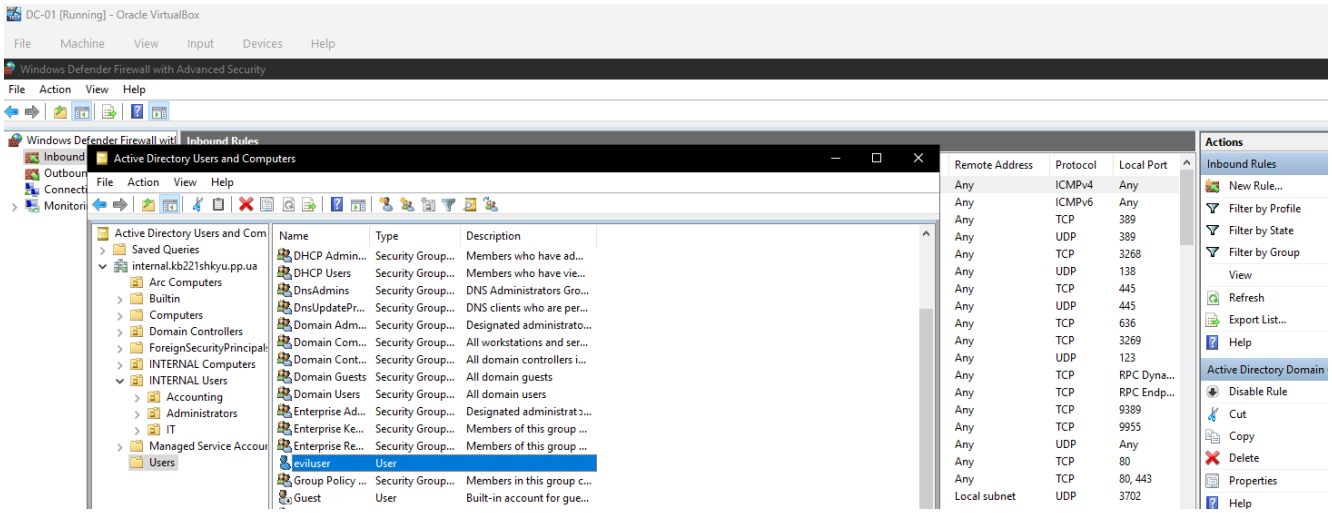


Рисунок 3.77 – Результат створення нового користувача

Після чого було згенеровано необхідну послідовність подій. Результат виконання KQL-запиту зображено на рисунку 3.78.

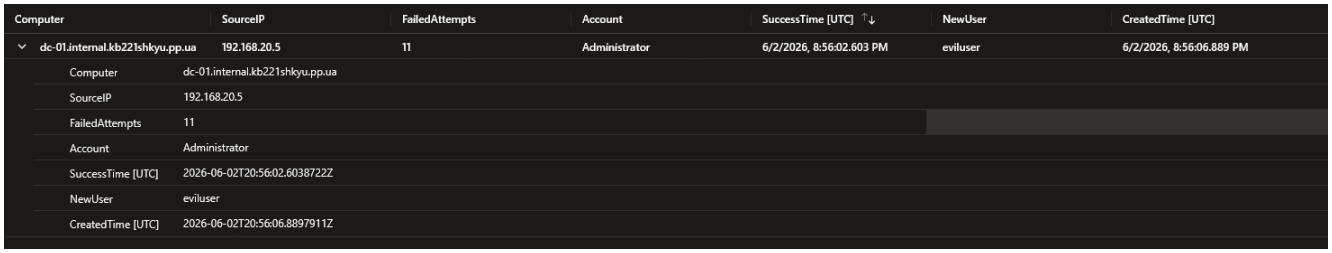


Рисунок 3.78 – Результат виконання KQL-запиту

Після отримання результатів запиту, аналітичне правило створило інцидент.

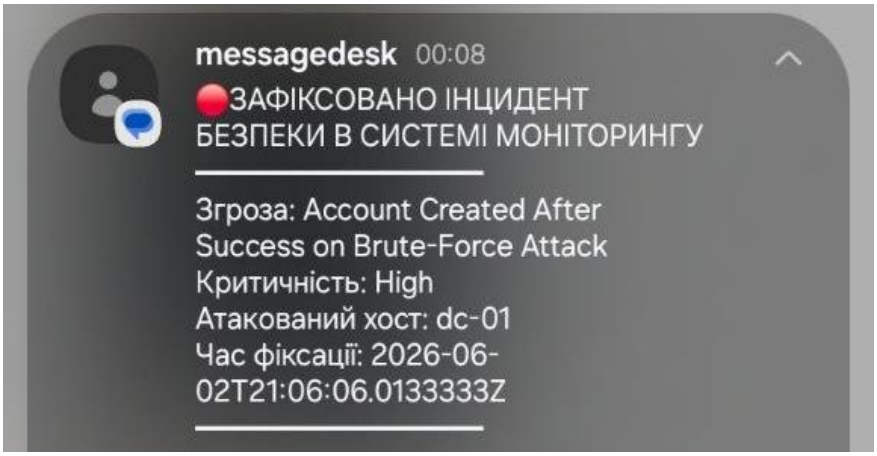


Рисунок 3.79 – Інформування про створений інцидент за допомогою SMS

Одразу після інформування було проведено вимкнення створеного облікового запису і блокування IP-адреси зловмисника.

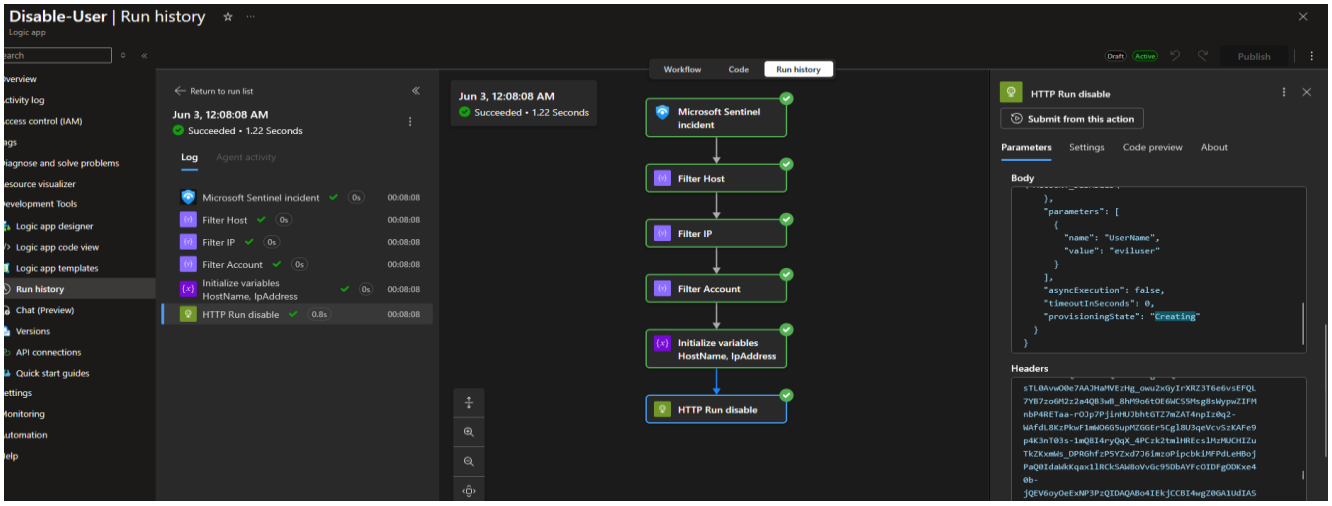


Рисунок 3.80 – Результат виконання Disable-User

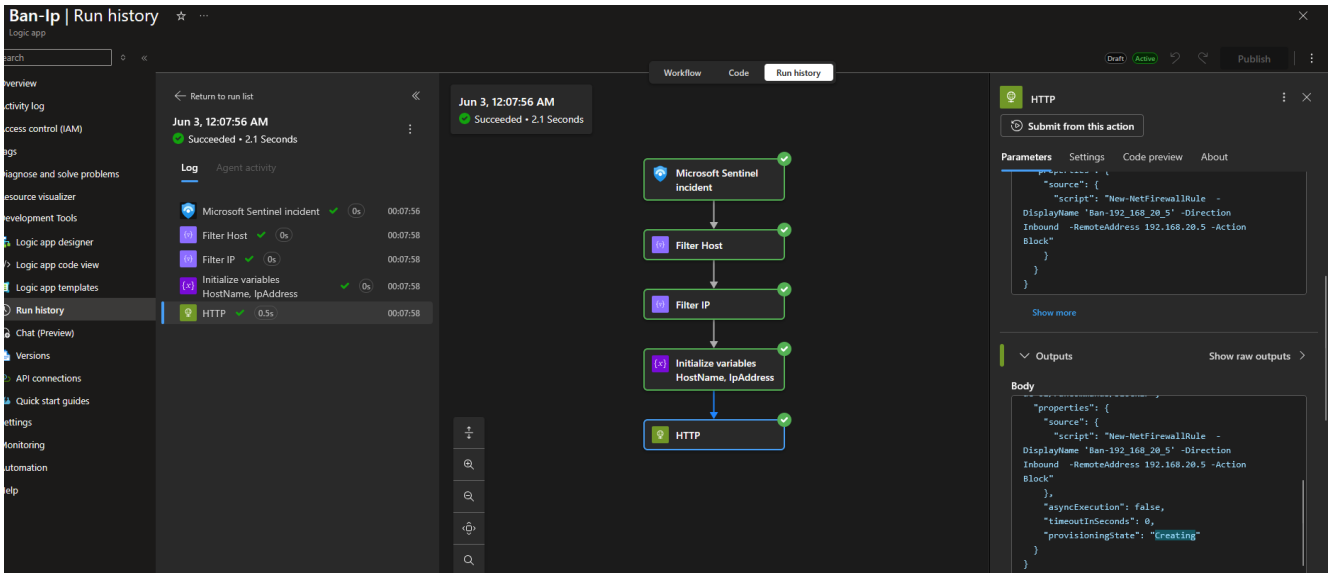


Рисунок 3.81 – Результат виконання Ban-Ip

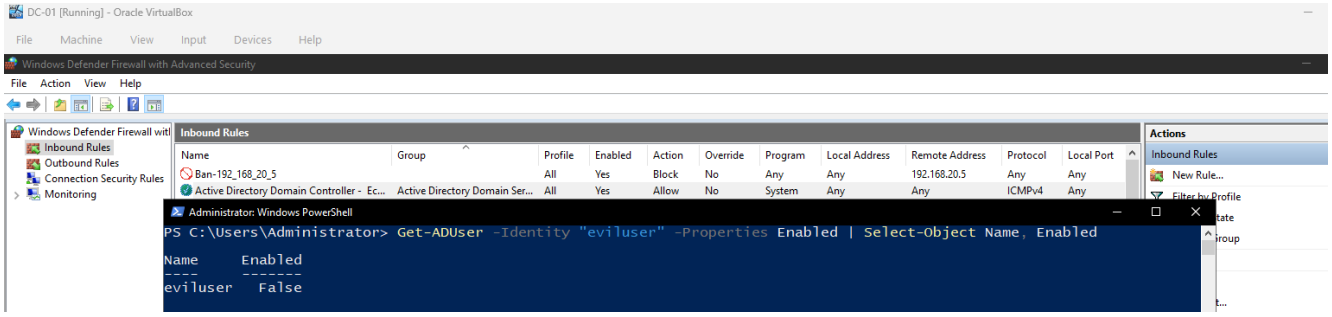


Рисунок 3.82 – Результат виконання плейбуків

Усі сценарії було відтворено успішно, результати виконання тестування наведено у табл. 3.27.

Таблиця 3.27

Результати тестування

Сценарій	Виявлено	Playbook виконано	Дія виконана	Сповіщення отримано
Brute Force	Так	Так	IP заблоковано, надіслано оповіщення на електронну пошту	Так
Компрометація облікового запису	Так	Так	Обліковий запис вимкнено, IP заблоковано, отримано сповіщення на SMS\Discord	Так
Веб-атака	Так	Так	IP заблоковано, отримано сповіщення на Telegram-бот	Так
Виявлення вірусу	Так	Так	Хост ізольовано, отримано сповіщення у Discord	Так
Backdoor-користувач	Так	Так	Користувача вимкнено, IP заблоковано, отримано оповіщення у SMS	Так

Проведене експериментальне тестування підтвердило досить високу ефективність гібридної системи захисту на базі Microsoft Sentinel. Усі змодельовані сценарії атак, від методів Brute-Force і веб-ін'єкцій до маніпуляцій з Active Directory та зараження кінцевих точок, були успішно ідентифіковані системою моніторингу в режимі реального часу. Середній час реагування виявився приблизно 2-3 хвилини, що є достатнім для закриття поширених загроз. Досить важливим є фактор використання KQL, саме його використання дозволило гнучко налаштувати логіку розпізнавання та мінімізувати кількість false positive спрацювань. Найважливішим результатом є успішна інтеграція SIEM з можливостями SOAR. Автоматизовані дії, такі як блокування IP-адрес на рівні робочої машини та Cloudflare, ізоляція ураже-

них робочих станцій та автоматичне вимкнення скомпрометованих облікових записів, довели здатність системи миттєво стримувати кібератаки без необхідності ручного втручання.

Висновки до третього розділу

У третьому розділі було успішно реалізовано програмну та апаратну складову проєкту гібридної інфраструктури виявлення та реагування на кіберінциденти. Було успішно налаштовано збір телеметрії з локального доменного середовища, вебсервера та робочих станцій шляхом розгортання Azure Monitor Agent. На базі зібраних даних розроблено низку дашбордів для візуалізації стану безпеки та написано оптимізовані аналітичні правила для детектування загроз.

Створені SOAR-плейбуки забезпечили ефективне автоматичне реагування на широкий (у перспективі) спектр інцидентів із застосуванням різних каналів оповіщення, таких як: Пошта, SMS, Telegram, Discord.

Експериментальне тестування показало, що побудована система є повністю працездатною та відповідає сучасним вимогам до центрів операцій з кібербезпеки, забезпечуючи надійний проактивний захист змішаних корпоративних IT-інфраструктур.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було реалізовано гібридну інфраструктуру для реагування та виявлення інцидентів безпеки різного типу.

У процесі було проведено аналіз сучасних підходів до моніторингу та реагування на інциденти інформаційної безпеки в різних IT-середовищах. Було виконано порівняльний аналіз моделей розгортання SIEM\SOAR-систем, за результатами якого обґрунтовано доцільність використання гібридної моделі як компромісу між гнучкістю локальних рішень та перевагами централізованого хмарного керування. Проведений аналіз доступних на ринку рішень показав, що вибір хмарної SIEM\SOAR платформи Microsoft Sentinel є найкращим варіантом для обраного для дослідження гібридного середовища.

У межах практичної частини було проведено аналіз типових загроз в умовах гібридної інфраструктури. Створено структурну схему, яка складається з чотирьох рівнів, і описує процес роботи системи виявлення інцидентів. Також було розглянуто основні підходи до інтеграції локальної мережі з хмарною SIEM платформою, та зроблено висновок, що використання агентних і Arc-орієнтованих механізмів інтеграції забезпечує необхідний рівень масштабованості та централізованого аналізу подій безпеки. Після аналізу основних класів загроз досліджуваної інфраструктури, було сформовано основний список джерел відповідної телеметрії для подальшого ефективного дослідження реагування. В результаті чого було спроектовано загальну архітектуру з урахуванням обраних технологічних рішень.

Проведене експериментальне тестування підтвердило працездатність розробленої системи, її здатність до автоматизованого виявлення та реагування на інциденти інформаційної безпеки. Отримані результати демонструють можливість ефективного використання запропонованого підходу для забезпечення проактивного захисту гібридних корпоративних IT-інфраструктур.

Таким чином усі поставлені завдання було виконано у ході дослідження. Результати підтверджують ефективність роботи досліджуваної інфраструктури.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. What is SIEM?. Microsoft. URL: https://www.microsoft.com/en-us/security/business/security-101/what-is-siem (дата звернення: 02.02.2026).

2. What is security information and event management. IBM. URL: https://www.ibm.com/think/topics/siem (дата звернення: 02.02.2026).

3. What is SOAR?. Microsoft. URL: https://www.microsoft.com/en-us/security/business/security-101/what-is-soar (дата звернення: 02.02.2026).

4. What is security orchestration, automation and response. IBM. URL: https://www.ibm.com/think/topics/security-orchestration-automation-response (дата звернення: 02.02.2026).

5. ATT&CK Matrix. Mitre. URL: https://attack.mitre.org/ (дата звернення: 06.02.2026).

6. Storm B. ATT&CK 101. Medium. 21.09.2018. URL: https://medium.com/mitre-attack/att-ck-101-17074d3bc62 (дата звернення: 07.02.2026).

7. What is MITRE ATT&CK® Framework and How Your SOC Can Benefit. Exabeam. URL: https://www.exabeam.com/explainers/mitre-attck/what-is-mitre-attck-framework-and-how-your-soc-can-benefit/ (дата звернення: 10.02.2026).

8. On-Premise vs Cloud-Based SIEM. Amsat. URL: https://amsat.ai/on-premise-vs-cloud-based-siem-which-is-right-for-you (дата звернення: 10.02.2026).

9. On-Premises Vs Cloud-Based SIEM. Wizardcyber. URL: https://wizardcyber.com/on-premises-vs-cloud-based-siem/ (дата звернення: 16.02.2026).

10. Doug D. Top Security Risks to Watch for in Shifting to Hybrid Identity Management. Semperis. URL: https://www.semperis.com/blog/security-risks-to-watch-for-in-shifting-to-hybrid-identity-management/ (дата звернення: 18.02.2026).

11. Security Telemetry. Deepwatch. URL: https://www.deepwatch.com/glossary/security-telemetry/ (дата звернення: 19.02.2026).

12. SIEM Architecture: Technology, Process and Data. Exabeam. URL: https://www.exabeam.com/explainers/siem/siem-architecture/ (дата звернення: 21.02.2026).

13. SIEM Correlation Rules: Enhancing Your Threat Detection. Stellarcyber. URL: https://stellarcyber.ai/learn/siem-correlation-rules/ (дата звернення: 01.03.2026).

14. Microsoft Sentinel data connectors. Microsoft. Learn. URL: https://learn.microsoft.com/en-us/azure/sentinel/connect-data-sources?tabs=defender-portal (дата звернення: 03.03.2026).

15. Data collection best practices. Microsoft. Microsoft. Learn. URL: https://learn.microsoft.com/en-us/azure/sentinel/best-practices-data (дата звернення: 03.03.2026).

16. Onboard Azure Arc-enabled servers to Microsoft Sentinel. Microsoft. Learn. URL: https://learn.microsoft.com/en-us/azure/azure-arc/servers/scenario-onboard-azure-sentinel (дата звернення: 07.03.2026).

17. paulberg. Forward On-Premises Windows Security Event Logs to Microsoft Sentinel. Microsoft. techcommunity. 17.12.2021. URL: https://techcommunity.microsoft.com/blog/coreinfrastructureandsecurityblog/forward-on-premises-windows-security-event-logs-to-microsoft-sentinel/3040784 (дата звернення: 08.03.2026).

18. Active Directory Domain Services overview. Microsoft. learn. URL: https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview (дата звернення: 14.03.2026).

19. Domain Name System (DNS). Microsoft. learn. URL: https://learn.microsoft.com/en-us/windows-server/networking/dns/dns-overview (дата звернення: 07.04.2026).

20. Dynamic Host Configuration Protocol (DHCP) Server. Microsoft. learn. URL: https://learn.microsoft.com/en-us/windows-server/networking/technologies/dhcp/dhcp-top (дата звернення: 07.04.2026).

21. Web Server (IIS) Overview. Microsoft. learn. URL: https://learn.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/hh831725(v=ws.11) (дата звернення: 07.04.2026).

22. Choose an ASP.NET Core web UI. Microsoft. learn. URL: https://learn.microsoft.com/en-us/aspnet/core/tutorials/choose-web-ui?view=aspnetcore-10.0 (дата звернення: 20.05.2026).

23. API Портал від Київстар. kyivstar. URL: https://api-gateway.kyivstar.ua/#overview (дата звернення: 31.05.2026).

24. Cloudflare REST API. Cloudflare. developers. URL: https://developers.cloudflare.com/pages/configuration/api/ (дата звернення: 02.06.2026).

25. About Cloudflare. *Cloudflare*. URL: https://www.cloudflare.com/about-overview/ (дата звернення: 04.12.2025).

26. KQL - Learn common operators. Microsoft. learn. URL: https://learn.microsoft.com/en-us/kusto/query/tutorials/learn-common-operators?view=microsoft-fabric (дата звернення: 02.06.2026).

27. Maddukuri V. How to Use Azure Key Vault Secrets for Logic App. Medium. 11.01.2024. URL: https://medium.com/@venky89.ai/configure-key-vault-for-logic-app-part-1-2011d40738a7 (дата звернення: 02.06.2026).

ДОДАТКИ

Додаток А

Описові таблиці

Таблиця А.1

Опис встановлених ролей

Роль	Опис
AD DS	Надає засоби для зберігання даних каталогу та забезпечення доступу до них для користувачів і адміністраторів мережі. AD DS зберігає інформацію про облікові записи користувачів, таку як імена, паролі, номери телефонів тощо. AD DS також забезпечує авторизованим користувачам у тій самій мережі доступ до цієї інформації [18].
DNS	DNS відіграє ключову роль у роботі служб домену Active Directory (AD DS), виступаючи механізмом визначення розташування контролерів домену для таких операцій, як автентифікація, оновлення та пошук. Контролери домену також використовують DNS для взаємного пошуку. Контролери домену також використовують DNS для взаємного пошуку. Сервер DNS відповідає за зберігання та реплікацію зон DNS, що містять імена та IP-адреси всіх комп'ютерів у домені. Це дозволяє клієнтам перетворювати доменні імена в IP-адреси та знаходити контролери домену [19].
DHCP	Сервер протоколу динамічної конфігурації хостів (DHCP) у Windows Server автоматизує процес призначення та управління IP-адресами та відповідними мережними налаштуваннями. Завдяки централізації цих завдань сервер DHCP зменшує обсяг ручної роботи, мінімізує помилки та забезпечує ефективне використання ресурсів IP-адрес у мережі [20].
IIS	Роль вебсервера (IIS) у Windows Server 2012 забезпечує безпечну, просту в управлінні, модульну та розширювану платформу для надійного розміщення веб-сайтів, служб і додатків.

Таблиця А.2

Сторінки вебдодатку та їх опис

Сторінка	Опис
/Index	Головна сторінка інтернет-магазину, доступна усім користувачам. Дозволяє переглядати список усіх товарів у магазині та виконувати їх сортування. Сторінка має потенційну загрозу використання SQL ін'єкції, якщо система пошуку вразлива до цього .
/Product	Сторінка з конкретним товаром, його описом та коментарями інших користувачів, сторінка доступна усім користувачам. Коментарі можуть залишати лише авторизовані користувачі. Секція коментарів потенційно може бути вразливою до XSS.
/Login та /Register	Сторінки логіну та реєстрації, доступні усім користувачам. Сторінка реєстрації надає можливість створення облікового запису, в результаті реєстрації створюється обліковий запис і видається JWT для авторизації користувача у вебдодатку. Сторінка логіну має аналогічний функціонал, але на відміну від реєстрації не створює новий обліковий запис. Також потенційні місця використання SQL ін'єкції для обходу ідентифікації.
/Me	Сторінка поточного облікового запису, доступна тільки авторизованим користувачам. На даній сторінці можна отримати інформацію про поточний обліковий запис, включно з його правами доступу у системі. Також на даній сторінці можна вийти з поточного облікового запису.

Закінчення таблиці А.2

Сторінка	Опис
/Panel/Products	Сторінка з панеллю керування товарами, доступна тільки авторизованим користувачам з правами адміністратора. Сторінка дає можливість перегляду усіх наявних товарів, їх редагування, видалення та додавання нових. Можливе місце використання File Upload вразливостей.
/Panel/Users	Сторінка має ті ж властивості і функціонал, що і попередня.
/FirstStartup	Сторінка яка доступна тільки при першому запуску вебдодатку. Використовується для створення першого облікового запису адміністратора.

Таблиця А.3

Опис основних полів таблиці Heartbeat

Поле	Опис
TimeGenerated	Точний час, коли агент відправив «пульс»
Computer	Мережне ім’я машини
OSType	Тип операційної системи
OSName	Конкретна версія
ResourceProvider	Для Arc-машин тут буде вказано
ComputerIP	IPv4 адреса сервера

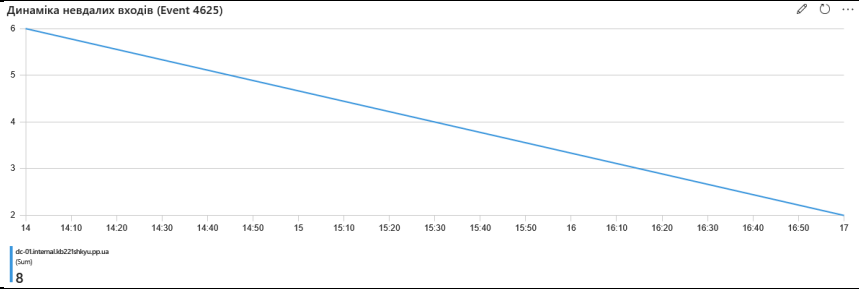
Таблиця А.4

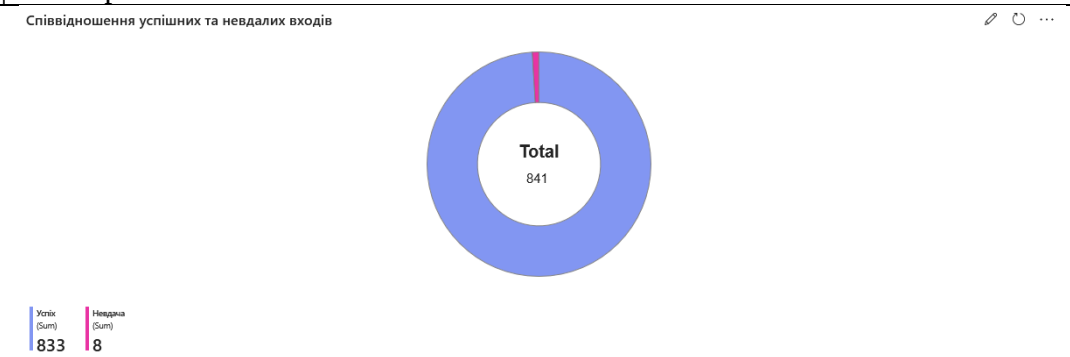
Опис основних полів таблиці Event

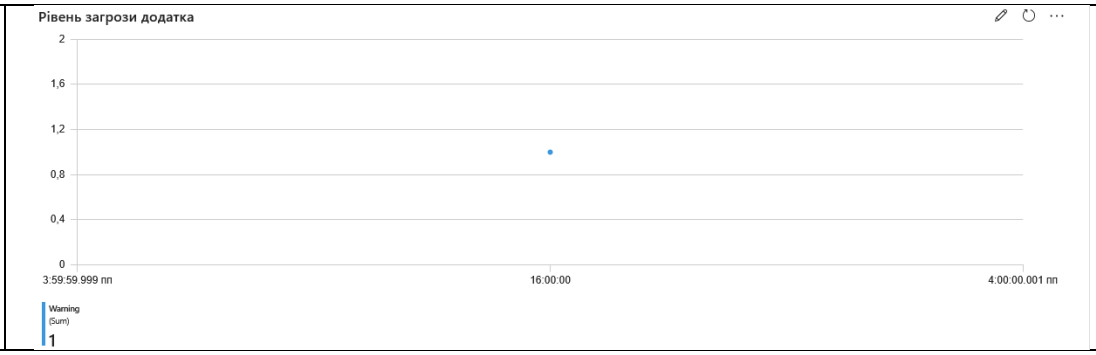
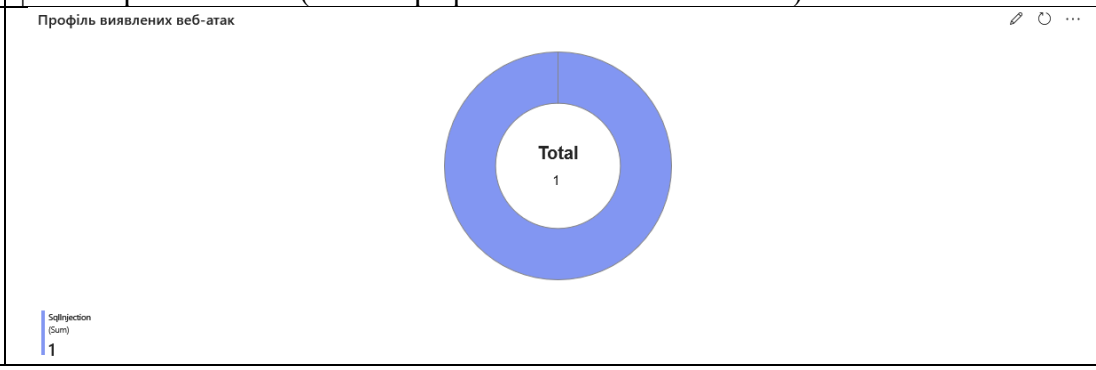
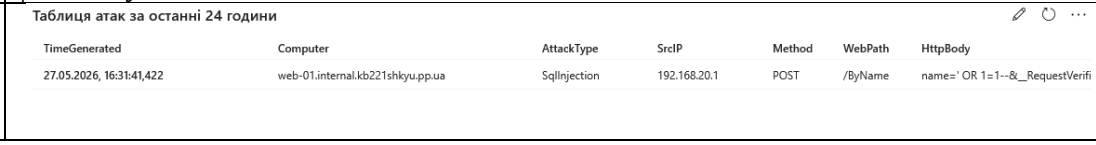
Поле	Опис
TimeGenerated	Точний час, коли подія запису локально в Event Viewer
Computer	Ім’я комп’ютера, де сталася подія
EventID	Унікальний код події Windows
EventLog	З якого журналу взято подій
EventLevelName	Рівень критичності
RenderedDescription	Повний текст опису події
Source	Програма або компонент системи, що створив подію

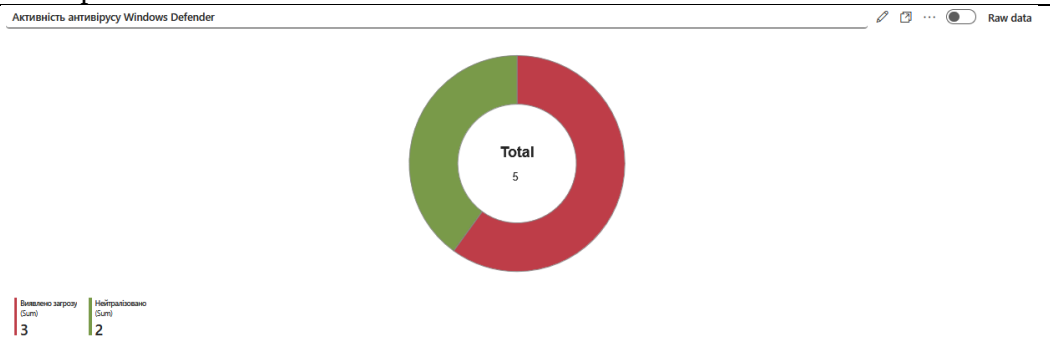
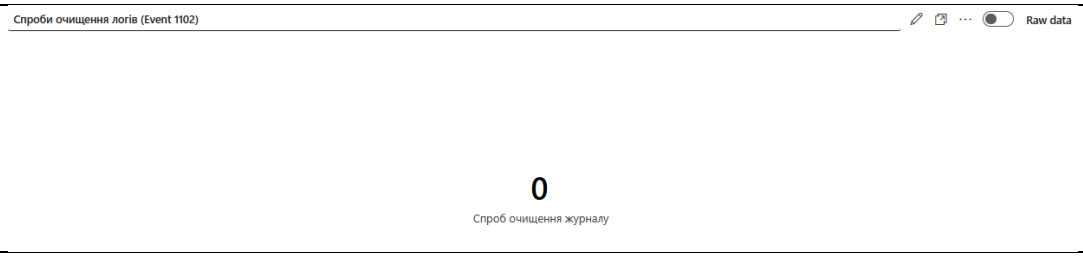
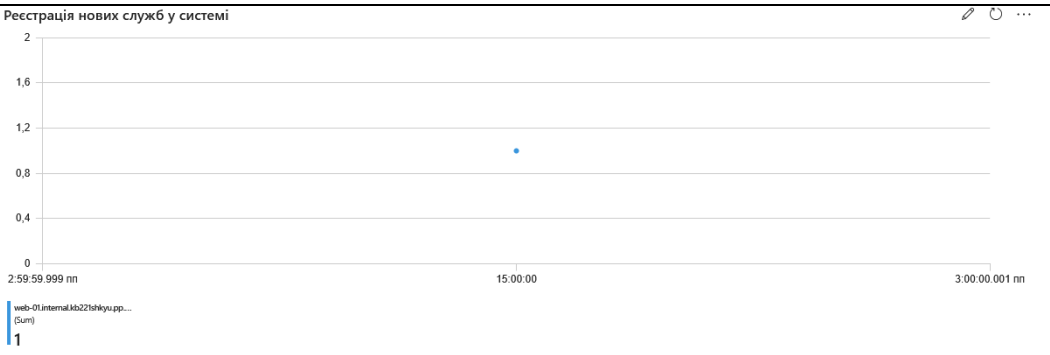
Таблиця А.5

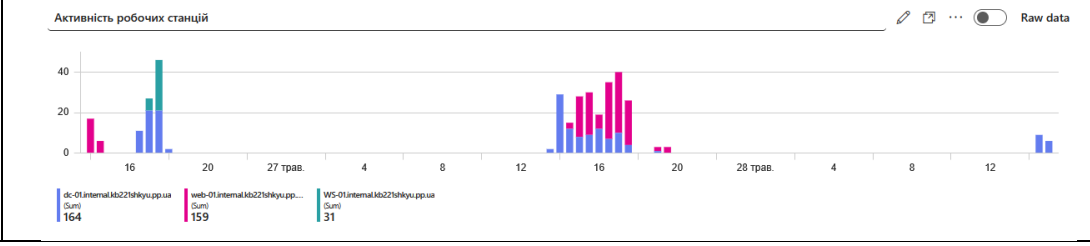
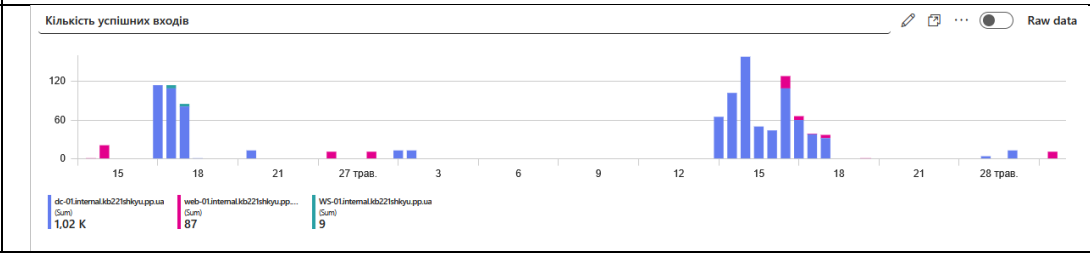
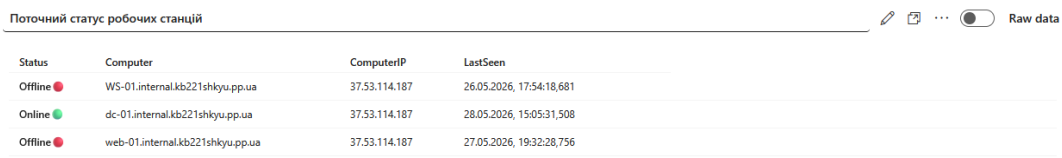
KQL запити та результати їх виконання

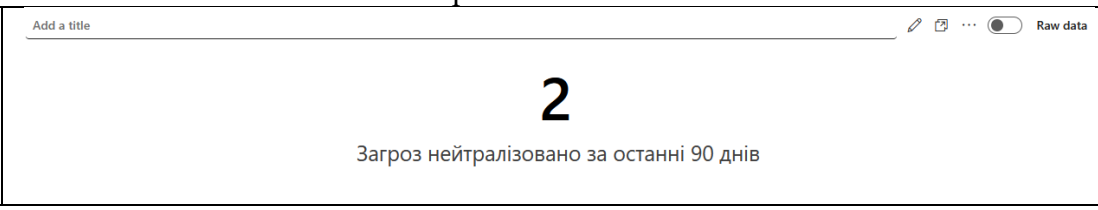
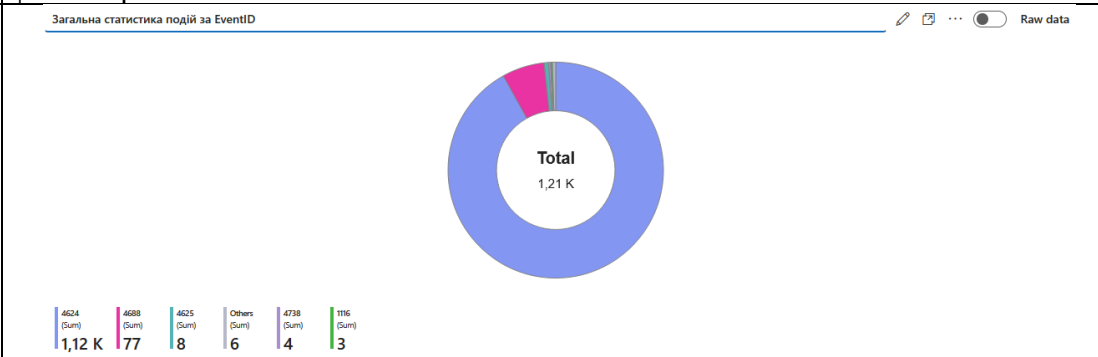
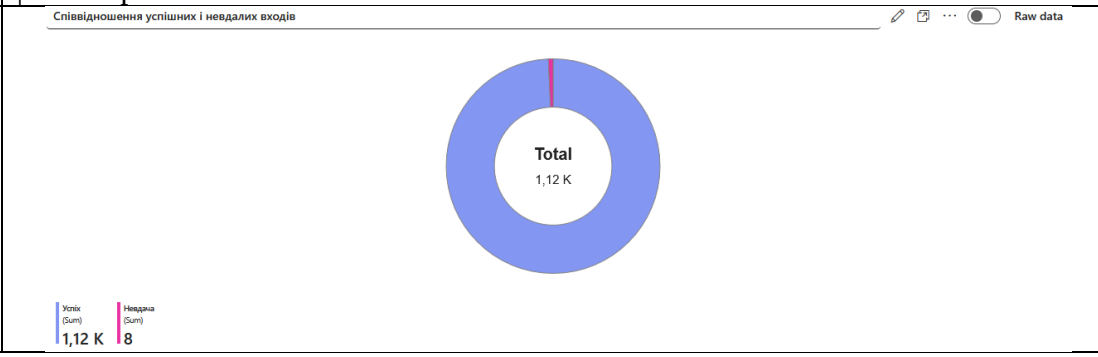
Метрика	
Динаміка невдалих входів	
KQL запит	Event | where TimeGenerated > ago(24h) | where EventID == 4625 | summarize FailedLogins = count() by bin(TimeGenerated, 30m), Computer | render timechart
Результат	

<table>
<tr><td colspan="2">Метрика</td></tr>
<tr><td colspan="2">Співвідношення успішних та невдалих входів</td></tr>
<tr><td>KQL запит</td><td>Event
| where TimeGenerated > ago(24h)
| where EventLog == "Security" and EventID in (4624, 4625)
| extend Status = iff(EventID == 4624, "Успіх", "Невдача")
| summarize Count = count() by Status
| render piechart</td></tr>
<tr><td>Результат</td><td> 
</td></tr>
<tr><td colspan="2">Співвідношення успішних та невдалих входів</td></tr>
<tr><td>KQL запит</td><td>Event
| where TimeGenerated > ago(7d)
| where EventLog == "Security" and EventID in (4720, 4722, 4724, 4732)
| extend EventAction = case(
EventID == 4720, "Створення користувача",
EventID == 4722, "Активація користувача",
EventID == 4724, "Скидання пароля",
EventID == 4732, "Додавання в Security Group",
"Інші зміни"
)
| summarize Count = count() by bin(TimeGenerated, 2h), EventAction
| render timechart</td></tr>
<tr><td>Результат</td><td> 
</td></tr>
<tr><td colspan="2">Рівень загрози додатка</td></tr>
<tr><td>KQL запит</td><td>Event
| where TimeGenerated > ago(24h)
| where EventID == 0
| summarize Count = count() by bin(TimeGenerated, 1h), EventLevelName
| render areachart</td></tr>
</table>

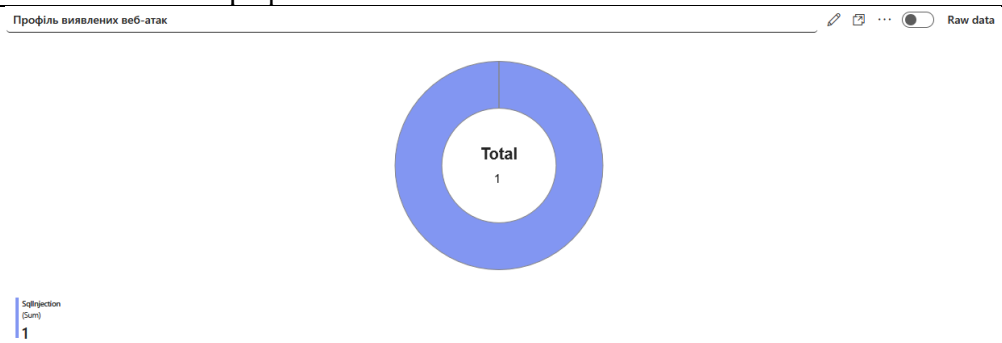
Результат	
Профіль виявлених веб-атак	
KQL запит	Event | where TimeGenerated > ago(3d) | where Source == "RequestAuditMiddleware" or RenderedDescription has "AUDIT_EVENT" | extend ExtractedAttack = extract(@"AttackType:([a-zA-Z0-9_-]+)\s*|", 1, RenderedDescription) | extend AttackType = iff(isempty(ExtractedAttack), "Unknown Anomaly", ExtractedAttack) | summarize Count = count() by AttackType | render piechart with (title="Профіль виявлених веб-атак")
Результат	
Таблиця атак за останні 24 години	
KQL запит	Event | where TimeGenerated > ago(24h) | where Source == "RequestAuditMiddleware" or RenderedDescription has "AUDIT_EVENT" | project TimeGenerated, Computer, AttackType = extract(@"AttackType:([a-zA-Z0-9_-]+)\s*|", 1, RenderedDescription), SrcIP = extract(@"IP:([0-9\.]+)\s*|", 1, RenderedDescription), Method = extract(@"Method:([A-Z]+)\s*|", 1, RenderedDescription), WebPath = extract(@"Path:([^\s|]+)\s*|", 1, RenderedDescription), HttpBody = extract(@"Body:(.*?)\s*|", 1, RenderedDescription) | order by TimeGenerated desc
Результат	

Метрика	
Активність антивірусу Windows Defender	
KQL запит	Event | where TimeGenerated > ago(7d) | where EventLog == "Microsoft-Windows-Windows Defender/Operational" | where EventID in (1116, 1117) | extend ThreatStatus = iff(EventID == 1116, "Виявлено загрозу", "Нейтралізовано") | summarize Count = count() by ThreatStatus, Computer | render piechart
Результат	
Спроби очищення логів	
KQL запит	Event | where TimeGenerated > ago(24h) | where EventLog == "Security" and EventID == 1102 | summarize Count = count()
Результат	
Реєстрація нових служб у системі	
KQL запит	Event | where TimeGenerated > ago(7d) | where EventLog == "System" and EventID == 7045 | summarize NewServices = count() by bin(TimeGenerated, 12h), Computer | render timechart
Результат	

Метрика	
Активність робочих станцій	
KQL запит	Heartbeat | where TimeGenerated > ago(90d) | summarize Ping = count() by bin(TimeGenerated, 30m), Computer
Результат	
Спроби очищення логів	
KQL запит	Event | where TimeGenerated > ago(90d) | where EventID == 4624 | summarize FailedLogins = count() by bin(TimeGenerated, 30m), Computer
Результат	
Поточний статус робочих станцій	
KQL запит	Heartbeat | summarize arg_max(TimeGenerated, *) by Computer | extend Status = iff(TimeGenerated < ago(5m), "Offline 🔴", "Online 🟢") | project Status, Computer, ComputerIP, LastSeen = TimeGenerated | order by Computer asc
Результат	
Загроз нейтралізовано за останні 90 днів	
KQL запит	Event | where TimeGenerated > ago(7d) | where EventLog == "Microsoft-Windows-Windows Defender/Operational" | where EventID == 1117 | extend ThreatStatus = "Нейтралізовано" | summarize Count = count() by ThreatStatus, Computer | render barchart with (title="Статус реагування Windows Defender")

Метрика	
Результат	
Загальна статистика подій за EventID	
KQL запит	Event | where TimeGenerated > ago(90d) | summarize Count = count() by EventID = tostring(EventID) | render piechart
Результат	
Співвідношення успішних і невдалих входів	
KQL запит	Event | where TimeGenerated > ago(90d) | where EventLog == "Security" and EventID in (4624, 4625) | extend Status = iff(EventID == 4624, "Успіх", "Невдача") | summarize Count = count() by Status | render piechart
Результат	
Профіль виявлених веб-атак	
KQL запит	Event | where TimeGenerated > ago(90d) | where Source == "RequestAuditMiddleware" or RenderedDescription has "AUDIT_EVENT" | extend ExtractedAttack = extract(@"AttackType:([a-zA-Z0-9_-]+)\s*|", 1, RenderedDescription) | extend AttackType = iff(isempty(ExtractedAttack), "Unknown Anomaly", ExtractedAttack) | summarize Count = count() by AttackType | render piechart with (title="Профіль виявлених веб-атак")

Закінчення таблиці А.5

Метрика	
Профіль виявлених веб-атак	
Результат	

Описові рисунки

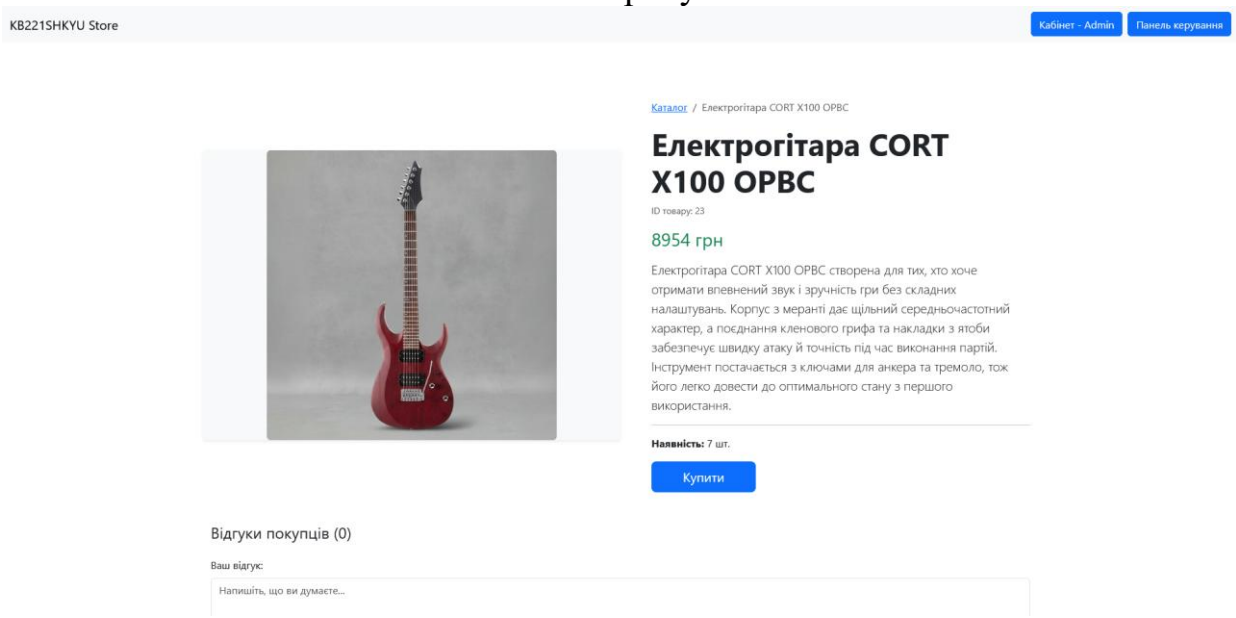


Рисунок Б.1 – Сторінка з конкретним товаром

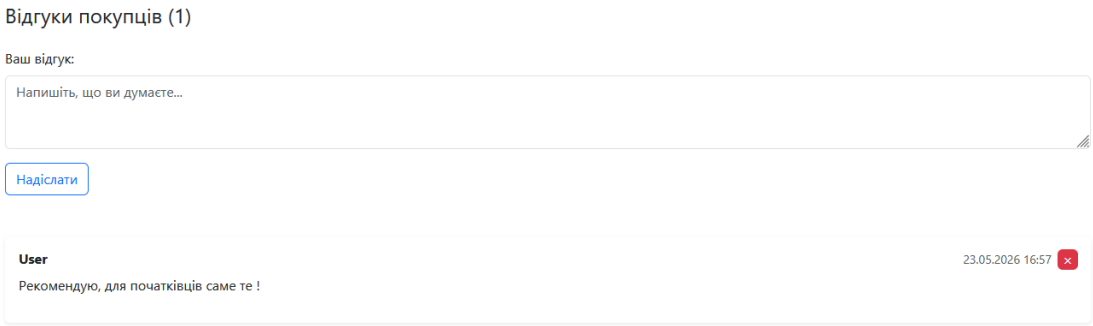


Рисунок Б.2 – Секція коментарів

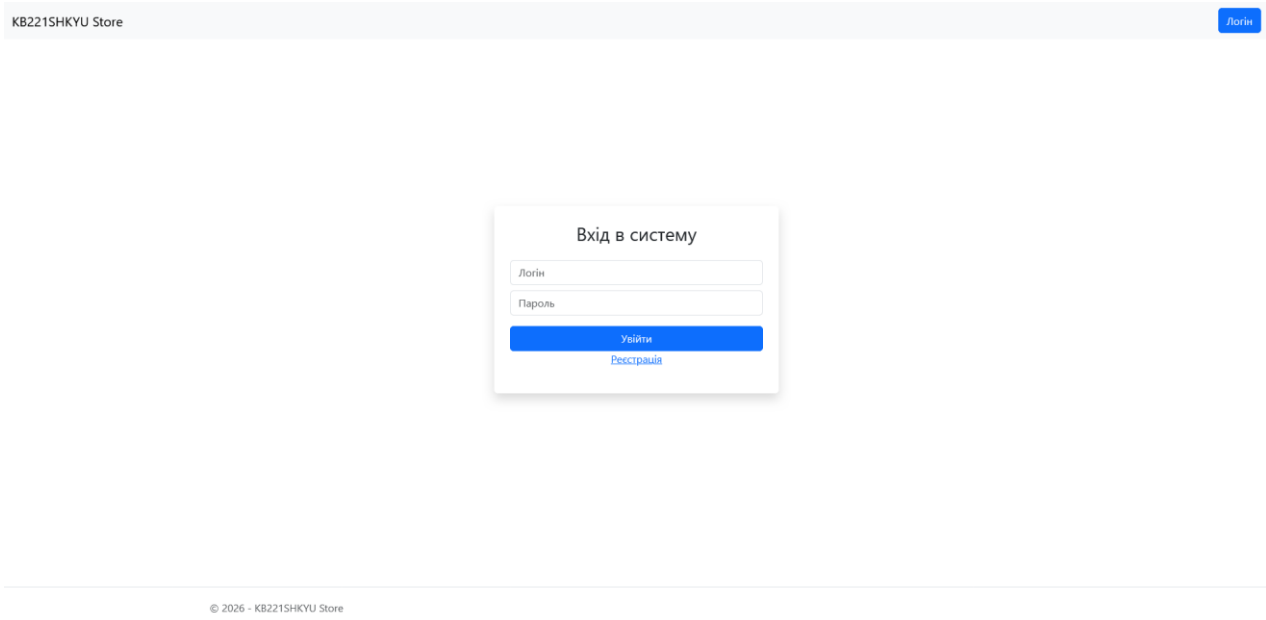


Рисунок Б.3. Сторінка логіну чи реєстрації

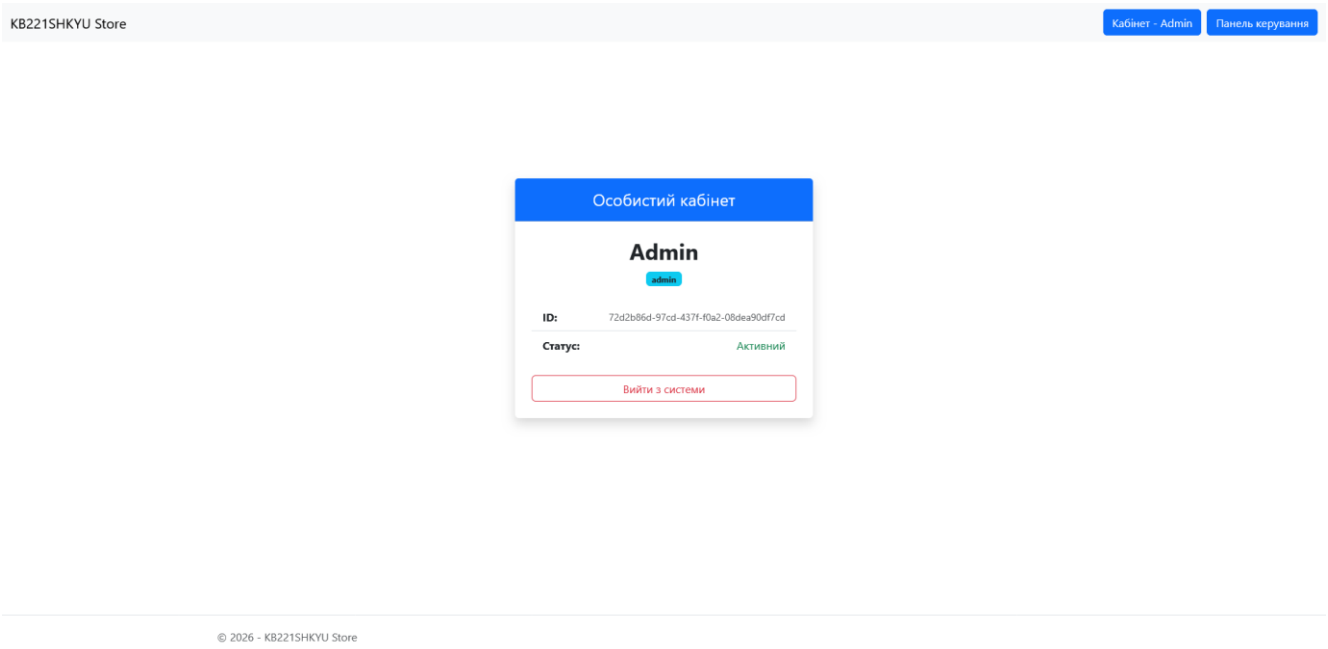


Рисунок Б.4 – Сторінка поточного користувача

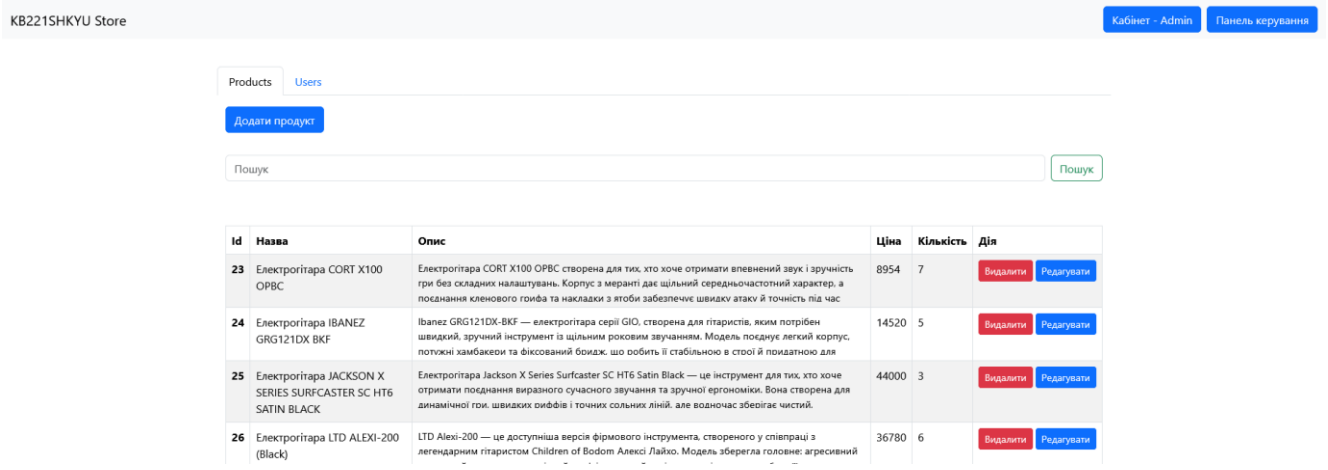


Рисунок Б.5 – Панель керування товарами

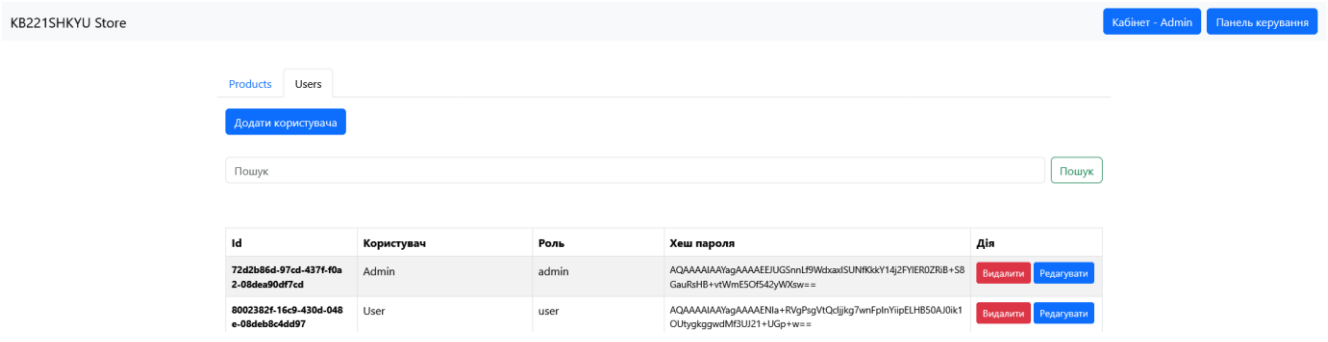


Рисунок Б.6 – Панель керування користувачами

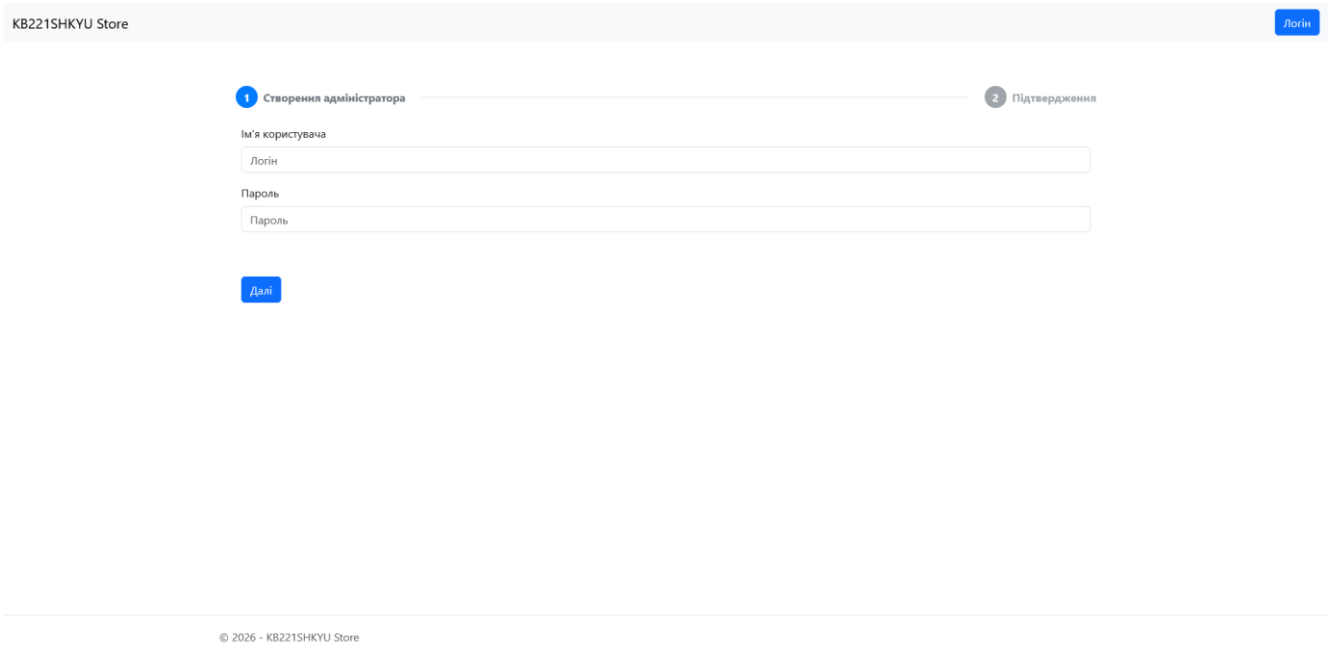


Рисунок Б.7 – Сторінка першого запуску

Додаток В

Лістинги коду

Таблиця В.1

Метод для перевірки на підозрілу активність

```
public static (bool IsSuspicious, AttackType DetectedType) IsSuspiciousActivity(string input)
    {
        if (string.IsNullOrWhiteSpace(input))
            return (false, AttackType.None);

        string decodedQuery = System.Net.WebUtility.UrlDecode(input).ToLower();

        foreach (var item in DangerousPatterns)
        {
            if (decodedQuery.Contains(item.Pattern))
            {
                return (true, item.Type);
            }
        }

        return (false, AttackType.None);
    }
```

Таблиця В.2

Набір сигнатур сучасних веб-атак

```
private static readonly (string Pattern, AttackType Type)[] DangerousPatterns = new (string,
AttackType)[]
{
    // --- SQL Injection (SQLi) ---
    ("'", AttackType.SqlInjection), ("--", AttackType.SqlInjection),
    (";", AttackType.SqlInjection), ("/*", AttackType.SqlInjection),
    ("*/", AttackType.SqlInjection), ("@@", AttackType.SqlInjection),
    ("@", AttackType.SqlInjection), ("drop table", AttackType.SqlInjection),
    ("drop database", AttackType.SqlInjection), ("insert into", AttackType.SqlInjection),
    ("update ", AttackType.SqlInjection), ("delete from", AttackType.SqlInjection),
    ("select ", AttackType.SqlInjection), ("union select", AttackType.SqlInjection),
    ("union all", AttackType.SqlInjection), ("truncate table", AttackType.SqlInjection),
    ("alter table", AttackType.SqlInjection), ("exec(", AttackType.SqlInjection),
    ("execute(", AttackType.SqlInjection), ("xp_cmdshell", AttackType.SqlInjection),
    ("sp_executesql", AttackType.SqlInjection), ("waitfor delay", AttackType.SqlInjection),
    ("cast(", AttackType.SqlInjection), ("convert(", AttackType.SqlInjection),
    ("declare ", AttackType.SqlInjection), ("varchar(", AttackType.SqlInjection),
    ("nvarchar(", AttackType.SqlInjection), ("1=1", AttackType.SqlInjection),
    ("or 1=1", AttackType.SqlInjection), ("and 1=1", AttackType.SqlInjection),

    // --- Cross-Site Scripting (XSS) ---
    ("<script", AttackType.Xss), ("</script>", AttackType.Xss),
    ("javascript:", AttackType.Xss), ("vbscript:", AttackType.Xss),
    ("data:text/html", AttackType.Xss), ("onload=", AttackType.Xss),
    ("onerror=", AttackType.Xss), ("onmouseover=", AttackType.Xss),
    ("onfocus=", AttackType.Xss), ("onblur=", AttackType.Xss),
    ("onclick=", AttackType.Xss), ("eval(", AttackType.Xss),
    ("settimeout(", AttackType.Xss), ("document.cookie", AttackType.Xss),
    ("document.location", AttackType.Xss), ("window.location", AttackType.Xss),
    ("<iframe", AttackType.Xss), ("<object", AttackType.Xss),
    ("<embed", AttackType.Xss), ("<meta", AttackType.Xss),
    ("<form", AttackType.Xss), ("<svg", AttackType.Xss),
    ("alert(", AttackType.Xss), ("prompt(", AttackType.Xss),
    ("confirm(", AttackType.Xss),

    // --- Windows & IIS Path Traversal ---
```

Закінчення таблиці В.2

```
    ("..\\", AttackType.PathTraversal), ("../", AttackType.PathTraversal),
    ("%2e%2e%2f", AttackType.PathTraversal), ("%2e%2e\\", AttackType.PathTraversal),
    ("%u002e%u002e", AttackType.PathTraversal), ("%u2216", AttackType.PathTraversal),
    ("c:\\", AttackType.PathTraversal), ("d:\\", AttackType.PathTraversal),
    ("c:/", AttackType.PathTraversal), ("\\windows\\", AttackType.PathTraversal),
    ("\\system32\\", AttackType.PathTraversal), ("boot.ini", AttackType.PathTraversal),
    ("win.ini", AttackType.PathTraversal), ("web.config", AttackType.PathTraversal),
    ("global.asax", AttackType.PathTraversal), ("app_data", AttackType.PathTraversal),
    ("app_code", AttackType.PathTraversal), ("~1", AttackType.PathTraversal),
    ("~2", AttackType.PathTraversal), ("::$data", AttackType.PathTraversal),

    // --- Windows Command Injection ---
    ("|", AttackType.CommandInjection), ("||", AttackType.CommandInjection),
    ("&", AttackType.CommandInjection), ("&&", AttackType.CommandInjection),
    ("cmd.exe", AttackType.CommandInjection), ("cmd ", AttackType.CommandInjection),
    ("/c ", AttackType.CommandInjection), ("powershell", AttackType.CommandInjection),
    ("pwsh", AttackType.CommandInjection), ("%comspec%", AttackType.CommandInjection),
    ("whoami", AttackType.CommandInjection), ("net user", AttackType.CommandInjection),
    ("net localgroup", AttackType.CommandInjection), ("ipconfig", AttackType.CommandInjection),
    ("systeminfo", AttackType.CommandInjection), ("tasklist", AttackType.CommandInjection),
    ("type ", AttackType.CommandInjection), ("dir ", AttackType.CommandInjection),
    ("copy ", AttackType.CommandInjection), ("del ", AttackType.CommandInjection),
    ("sc ", AttackType.CommandInjection), ("reg ", AttackType.CommandInjection),

    // --- SSRF ---
    ("file:///", AttackType.Ssrf)
                                                                                    };
```

Таблиця В.3

Метод формування подій

```
public async Task Invoke(HttpContext context)
{
    string body = "";
    if (HasBody(context.Request))
    {
        body = await ReadBody(context.Request);
    }
    var decodedBody = WebUtility.UrlDecode(body);
    var query = WebUtility.UrlDecode(context.Request.QueryString.ToString());
    var header = context.Request.Headers["CF-Connecting-IP"].ToString();
    var        ip        =        !string.IsNullOrEmpty(header)        ?        header        :
context.Connection.RemoteIpAddress?.ToString();
    var path = context.Request.Path;
    var method = context.Request.Method;
    var userId = context.User?.Claims?.FirstOrDefault(c => c.Type == "id")?.Value ?? "Anonymous";
    var (isSuspicious, attackType) = AttackDetector.IsSuspiciousActivity(query);
    if (!isSuspicious && !string.IsNullOrWhiteSpace(body))
    {
        (isSuspicious, attackType) = AnalyzeBodyStructure(context.Request, body);
    }
    if (isSuspicious)
    {
       _logger.LogWarning("AUDIT_EVENT | AttackType:{0} | IP:{1} | Method:{2} | Path:{3} | Query:{4}
| Body:{5} | UserId:{6}", attackType, ip, method, path, query, decodedBody, userId);
    }
    else
    {
        _logger.LogInformation("AUDIT_EVENT | IP:{0} | Method:{1} | Path:{2} | Query:{3} | Body:{4}
| UserId:{5}", ip, method, path, query, decodedBody, userId);
    }
    await _next(context);
}
```

Таблиця В.4

Скрипт розгортання хмарної інфраструктури

```
# ============================================================
#                     КОНФІГУРАЦІЯ ЗМІННИХ
# ============================================================

$TenantId = "fad27524-0540-4b94-9c20-752698e3efb0"
$SubscriptionId = "fcc92f85-50c5-4f5a-b123-eb8af9b63e95"
$Location = "switzerlandnorth"
$ResourceGroup = "rg-szn-arcs"
$SpName = "ArcOnboardingSP"
$LawName = "law-arcs-monitor"
$DcrName = "dcr-arcs-windows"

# ============================================================
#                   ПІДГОТОВКА ГРУПИ РЕСУРСІВ
# ============================================================
az login --tenant $TenantId
az account set --subscription $SubscriptionId

az group create --name $ResourceGroup --location $Location

# ============================================================
#         СТВОРЕННЯ SERVICE PRINCIPAL ДЛЯ ОНБОРДИНГУ
# ============================================================
# Збережіть appId (ServicePrincipalId) та password (ServicePrincipalClientSecret) для подальшого
онбоардингу
az ad sp create-for-rbac --name $SpName --role "Azure Connected Machine Onboarding" --scopes
/subscriptions/$SubscriptionId/resourceGroups/$ResourceGroup

# ============================================================
#             СТВОРЕННЯ LOG ANALYTICS WORKSPACE
# ============================================================

az monitor log-analytics workspace create --resource-group $ResourceGroup --name $LawName --location
$Location

$LawId = az monitor log-analytics workspace show --resource-group $ResourceGroup --workspace-name
$LawName --query id --output tsv

# ============================================================
#             СТВОРЕННЯ LOG ANALYTICS WORKSPACE
# ============================================================

(Get-Content dcr-template.json) -replace '__WORKSPACE_ID__', $LawId | Set-Content dcr-config.json

az monitor data-collection rule create --resource-group $ResourceGroup --name $DcrName --location
$Location --rule-file "dcr-config.json"
```

Таблиця В.5

Скрипт створення GPO

```
$ServicePrincipalId="d1480c82-170b-40a5-8496-9a5508570f52";
$ServicePrincipalClientSecret="<ОТРИМАНИЙ СЕКРЕТ З ЗАВАНТАЖЕНОГО ФАЙЛУ>";

.\DeployGPO.ps1      -DomainFQDN      internal.kb221shkyu.pp.ua      -ReportServerFQDN      "dc-
01.internal.kb221shkyu.pp.ua"       -ArcRemoteShare       ArcDeploy       -ServicePrincipalSecret
$ServicePrincipalClientSecret   -ServicePrincipalClientId   $ServicePrincipalId   -SubscriptionId
fcc92f85-50c5-4f5a-b123-eb8af9b63e95  -ResourceGroup  rg-szn-arcs  -Location  switzerlandnorth  -
TenantId   fad27524-0540-4b94-9c20-752698e3efb0   -Tags   @{'Domain'='internal.kb221shkyu.pp.ua';
'Type'='onprem'; 'ArcSQLServerExtensionDeployment'='Disabled'}
```

Таблиця В.6

Скрипт асоціації серверів х DCR

```
# ============================================================
#                    КОНФІГУРАЦІЯ ЗМІНИХ
# ============================================================
$ResourceGroup = "rg-szn-arcs"
$DcrName = "dcr-arcs-windows"
# ============================================================
#           АСОЦІАЦІЯ DCR ІЗ ПІДКЛЮЧЕНИМИ СЕРВЕРАМИ
# ============================================================
$DcrId = az monitor data-collection rule show --resource-group $ResourceGroup --name $DcrName --
query id --output tsv
$ArcServers = az connectedmachine list --resource-group $ResourceGroup --query "[].{name:name,
id:id}" --output json | ConvertFrom-Json
if ($ArcServers.Count -eq 0) {
     Write-Host "У ресурсній групі $ResourceGroup не знайдено жодного Azure Arc сервера." -
ForegroundColor Red
} else {
    Write-Host "Знайдено серверів: $($ArcServers.Count). Починаємо підключення..." -ForegroundColor
Green

    foreach ($Server in $ArcServers) {
        Write-Host "Обробка сервера: $($Server.name)..." -ForegroundColor Cyan
        $AssocName = "assoc-dcr-$($Server.name)"

        az monitor data-collection rule association create --name $AssocName --resource $Server.id
--rule-id $DcrId --output none
        az connectedmachine extension create --name "AzureMonitorWindowsAgent" --publisher
"Microsoft.Azure.Monitor" --type "AzureMonitorWindowsAgent" --machine-name $Server.name --resource-
group "rg-szn-arcs" --location "switzerlandnorth"
        Write-Host "Сервер $($Server.name) успішно підключено до DCR!" -ForegroundColor Green
    }

    Write-Host "Масове підключення завершено!" -ForegroundColor Blue
}
```

Таблиця В.7

Структура повідомленні Discord Webhook

```
{
  "embeds":[
  {
    "title":"Security Incident Detected",
    "description":"Зафіксовано інцидент безпеки в системі моніторингу",
    "color":15158332,
    "fields":[
      {
        "name":"Назва загрози",
        "value":"@{variables('Title')}",
        "inline":false
      },
      {
        "name":"Критичність",
        "value":"@{variables('Severity')}",
        "inline":true
      },
      {
        "name":"Атакований хост",
        "value":"@{variables('HostName')}",
        "inline":true
```

Закінчення таблиці В.7

```
            },
            {
               "name":"Час фіксації",
               "value":"@{variables('CreatedTimeUtc')}",
               "inline":false
            },
            {
               "name":"Sentinel Incident",
               "value":"[Відкрити інцидент](@{variables('Url')})",
               "inline":false
            }
         ],
         "footer":{
            "text":"Microsoft Sentinel"
         },
         "timestamp":"@{utcNow()}"
      }
      ]
}
```

Таблиця В.8

KQL-запит до п’ятого сценарію атаки

```
let FailedLogons = Event
| where TimeGenerated > ago(30m)
| where EventID == 4771
| extend IpAddress = extract(@"<Data Name=""IpAddress"">([^<]+)</Data>", 1, EventData)
| extend SourceIP = replace_string(IpAddress, "::ffff:", "")
| where SourceIP !in ("127.0.0.1", "::1")
| extend Account = extract(@"<Data Name=""TargetUserName"">([^<]+)</Data>", 1, EventData)
| summarize
   FailedAttempts=count(),
   FirstSeen=min(TimeGenerated),
   LastSeen=max(TimeGenerated)
by Computer, SourceIP
| where FailedAttempts >= 3;
let SuccessfulLogons = Event
| where TimeGenerated > ago(30m)
| where EventID == 4624
| extend IpAddress = extract(@"<Data Name=""IpAddress"">([^<]+)</Data>", 1, EventData)
| extend SourceIP = replace_string(IpAddress, "::ffff:", "")
| extend Account = extract(@"<Data Name=""TargetUserName"">([^<]+)</Data>", 1, EventData)
| where SourceIP !in ("127.0.0.1", "::1");

let UserCreated = Event
| where TimeGenerated > ago(30m)
| extend CreatedTime = TimeGenerated
| where EventID == 4720
| extend NewUser = extract(@"<Data Name=""TargetUserName"">([^<]+)</Data>", 1, EventData);
FailedLogons
| join kind=inner SuccessfulLogons on SourceIP
| where TimeGenerated > LastSeen
| join kind=inner UserCreated on Computer
```

Закінчення таблиці В.8

```
| where CreatedTime > TimeGenerated
| project
    Computer,
    SourceIP,
    FailedAttempts,
    Account,
    SuccessTime = TimeGenerated,
    NewUser,
    CreatedTime
```