Міністерство освіти і науки України
Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра інформаційної та кібернетичної безпеки
імені професора Володимира Бурячка

«Допущено до захисту»
Завідувач кафедри інформаційної та кібернетичної безпеки імені професора Володимира Бурячка
кандидат технічних наук, доцент
Складанний П.М.

_____
(підпис)
« ___ » _____ 20__ р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття другого (магістерського)
рівня вищої освіти

Спеціальність 125 Кібербезпека та захист інформації

Тема роботи:
ТЕХНОЛОГІЯ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ КІБЕРЗАГРОЗАМИ З ВИКОРИСТАННЯМ SOAR ТА ІНСТРУМЕНТІВ THREAT INTELLIGENCE

Виконав
студент групи БІКСм-1-24-1.4.д

Яблоков Ілля Ростиславович
(прізвище, ім'я, по батькові)

_____
(підпис)

Науковий керівник
доктор філософії, доцент
(науковий ступінь, наукове звання)
Киричок Р. В.
(прізвище, ініціали)

_____
(підпис)

Київ – 2025

Міністерство освіти і науки України
Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра інформаційної та кібернетичної безпеки
імені професора Володимира Бурячка

Освітньо-кваліфікаційний рівень – магістр
Спеціальність 125 Кібербезпека та захист інформації
Освітня програма 125.00.01 Безпека інформаційних і комунікаційних систем

«Затверджую»
Завідувач кафедри інформаційної та
кібернетичної безпеки імені
професора Володимира Бурячка
кандидат технічних наук, доцент
Складанний П.М.

_____
(підпис)
« ___ » _____ 20__ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Яблокову Іллі Ростиславовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Технологія автоматизованого управління кіберзагрозами з використанням SOAR та інструментів Threat Intelligence;
   керівник Киричок Роман Васильович, доктор філософії, доцент;
   затверджені наказом ректора від «__» _____ 2025 року №__..
2. Термін подання студентом роботи «__» _____ 2025 р.
3. Вихідні дані до роботи: міжнародна та українська нормативно-правові бази, стандарти й рекомендації у сфері управління кіберінцидентами та реагування на кіберзагрози (зокрема ISO/IEC 27001, 27002, 27035, 30111, 29147; NIST SP 800-61, 800-53; ENISA Incident Response Guidelines; документи Держспецзв'язку, CERT-UA та Кіберполіції); методичні матеріали щодо оркестрації, автоматизації та інтеграції засобів безпеки (SOAR), а також використання платформ Threat

Intelligence (MITRE ATT&CK, D3FEND); аналітичні звіти та наукові публікації про автоматизоване реагування на кіберінциденти; відкриті фіди та OSINT-джерела кіберзагроз; тестові журнали подій SIEM/EDR-систем, сце реагування (playbooks) для SOAR-платформи.

4. Зміст текстової частини роботи (перелік питань, які потрібно розробити):
   4.1. Аналітичний огляд процесу управління кіберзагрозами в корпоративних інформаційних системах.
   4.2. Концептуальні засади та архітектура технології автоматизованого управління кіберзагрозами.
   4.3. Експериментальне дослідження технології автоматизованого управління кіберзагрозами.
5. Перелік графічного матеріалу:
   5.1. Схема типових векторів кіберзагроз на криптовалютні платформи; архітектура традиційного SOC з інтеграцією систем безпеки; діаграма життєвого циклу управління кіберінцидентом; концептуальна архітектура інтеграції SOAR-платформи з джерелами Threat Intelligence; алгоритм автоматизованого реагування на основі індикаторів компрометації; модель процесу збагачення подій даними Threat Intelligence; архітектура розробленої системи автоматизованого управління кіберзагрозами; UML-діаграми варіантів використання, послідовності та активностей; блок-схема алгоритму автоматичного порівняння подій з базою IoC; графіки результатів експериментального дослідження ефективності системи.
   5.2. Презентація доповіді, виконана в Microsoft PowerPoint.
6. Дата видачі завдання «__» _____ 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів підготовки роботи	Термін виконання	Примітка
1.	Уточнення постановки завдання		
2.	Аналіз літератури		
3.	Обґрунтування вибору рішення		
4.	Збір даних		
5.	Виконання та оформлення розділу 1.		
6.	Виконання та оформлення розділу 2.		
7.	Виконання та оформлення розділу 3.		
8.	Вступ, висновки, реферат		
9.	Апробація роботи на науково-методичному семінарі та науково-технічній конференції		
10.	Оформлення та друк текстової частини роботи		
11.	Оформлення презентацій		
12.	Отримання рецензій		
13.	Попередній захист роботи		
14.	Захист в ЕК		

Студент Яблоков Ілля Ростиславович
(прізвище, ім'я, по батькові)

Науковий керівник Киричок Роман Васильович
(прізвище, ім'я, по батькові)

РЕФЕРАТ

Кваліфікаційна робота присвячена технології автоматизованого управління кіберзагрозами з використанням SOAR та інструментів Threat Intelligence.

Робота складається зі вступу, трьох розділів, що містять 5 рисунків та 13 таблиці, висновків та списку використаних джерел, що містить 50 найменувань. Загальний обсяг роботи становить 116 сторінки, з яких 5 сторінок займають ілюстрації і таблиці на окремих аркушах, а також додатки, перелік умовних скорочень та список використаних джерел.

Об'єктом дослідження в роботі є процес управління кіберзагрозами в корпоративних інформаційних системах.

Предметом дослідженння є технологія автоматизованого управління кіберзагрозами на основі інтеграції SOAR-платформи з інструментами Threat Intelligence.

Метою роботи є підвищення ефективності управління кіберзагрозами в корпоративних інформаційних системах шляхом автоматизації процесів виявлення, аналізу та реагування на інциденти безпеки з використанням оркестрованої взаємодії засобів SOAR і платформ Threat Intelligence.

Для досягнення поставленої мети у роботі:

- проведено аналітичний огляд процесу управління кіберзагрозами в корпоративних інформаційних системах;
- сформовано концептуальні засади та архітектуру технології автоматизованого управління кіберзагрозами;
- проведено експериментальне дослідження розробленої технології, оцінити її ефективність і сформулювати практичні рекомендації щодо впровадження.

Наукова новизна одержаних результатів полягає в тому, що в роботі проведено удосконалення архітектури системи управління кіберінцидентами шляхом інтеграції платформи автоматизованого реагування з джерелами Threat Intelligence для

автоматичного збагачення подій контекстом про актуальні загрози та формування сценаріїв реагування на основі індикаторів компрометації, що дозволяє підвищити швидкість виявлення і блокування кіберзагроз.

Галузь застосування. Матеріали роботи можуть бути використані для автоматизації рутинних операцій, зменшення впливу людського чинника та забезпечення безперервного циклу адаптивного реагування на сучасні кіберзагрози. Розроблена технологія може бути використана у центрах моніторингу інформаційної безпеки (SOC), підрозділах кіберзахисту банківських, енергетичних, державних та IT-структур, які працюють із великим обсягом подій безпеки в реальному часі.

Ключові слова: ВРАЗЛИВІСТЬ, ІНЦИДЕНТ, АВТОМАТИЗОВАНЕ РЕАГУВАННЯ, THREAT INTELLIGENCE, SOAR, ІНДИКАТОРИ КОМПРОМЕТАЦІЇ, КОРЕЛЯЦІЯ ПОДІЙ, КІБЕРЗАГРОЗИ, SIEM, КРИПТОВАЛЮТНІ БІРЖІ, ЦЕНТР ОПЕРАЦІЙ БЕЗПЕКИ, PLAYBOOK, ВИЯВЛЕННЯ ЗАГРОЗ, KEYLOGGER, PHISHING

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

API – Application Programming Interface, програмний інтерфейс застосування

APT – Advanced Persistent Threat, цільова тривала загроза

AV – Antivirus, антивірусне програмне забезпечення

C2 – Command and Control, командно-контрольний сервер

CACAO – Collaborative Automated Course of Action Operations, стандарт опису playbook

CTI – Cyber Threat Intelligence, аналітична інформація про кіберзагрози

DNS – Domain Name System, система доменних імен

EDR – Endpoint Detection and Response, засоби виявлення та реагування на кінцевих точках

IAM – Identity and Access Management, система управління ідентифікацією та доступом

IDS – Intrusion Detection System, система виявлення вторгнень

IoC – Indicators of Compromise, індикатори компрометації

IP – Internet Protocol, протокол міжмережевої взаємодії

IPS – Intrusion Prevention System, система запобігання вторгненням

JSON – JavaScript Object Notation, формат обміну даними

MITRE ATT&CK – фреймворк класифікації тактик та техніік зловмисників

MTTR – Mean Time To Response, середній час реагування на інцидент

MTTD – Mean Time To Detect, середній час виявлення загрози

NDR – Network Detection and Response, засоби виявлення та реагування на мережеві загрози

OASIS – Organization for the Advancement of Structured Information Standards

OSINT – Open Source Intelligence, розвідка з відкритих джерел

Playbook – формалізований сценарій реагування на інцидент безпеки

REST – Representational State Transfer, архітектурний стиль для API

SIEM – Security Information and Event Management, система управління інформацією та подіями безпеки

SOAR – Security Orchestration, Automation and Response, платформа оркестрації, автоматизації та реагування на загрози

SOC – Security Operations Center, центр операцій безпеки

STIX – Structured Threat Information Expression, стандарт структурованого представлення інформації про загрози

TAXII – Trusted Automated Exchange of Indicator Information, протокол автоматизованого обміну індикаторами

TI – Threat Intelligence, аналітична інформація про загрози

TTP – Tactics, Techniques and Procedures, тактики, техніки та процедури зловмисників

UEBA – User and Entity Behavior Analytics, аналітика поведінки користувачів та сутностей

URL – Uniform Resource Locator, уніфікований локатор ресурсу

VT – VirusTotal, сервіс перевірки репутації файлів та індикаторів

ВСТУП

Актуальність. Стрімке зростання кількості кіберзагроз, їхня складність і динамічність створюють суттєві виклики для корпоративних інформаційних систем. Традиційні методи моніторингу безпеки, що базуються на ручному аналізі інцидентів, уже не забезпечують необхідної швидкості реагування та ефективності. У сучасних умовах особливого значення набуває автоматизація процесів виявлення, класифікації та реагування на кіберінциденти з використанням інтелектуальних технологій, здатних аналізувати великі обсяги даних у реальному часі.

Попри значний розвиток технологій автоматизації кіберзахисту, сучасні системи управління інцидентами залишаються фрагментованими та часто орієнтованими лише на реактивне реагування. Більшість існуючих платформ типу SOAR (SecurityOrchestration, AutomationandResponse) реалізують автоматизацію переважно на основі статичних сценаріїв (playbooks), які не враховують контексту поточної загрози, динаміки її розвитку та взаємозв’язків між індикаторами компрометації. У свою чергу, системи ThreatIntelligence (TI), що акумулюють дані про загрози, зазвичай функціонують автономно – вони надають інформацію про індикатори атак, але не інтегруються у механізми автоматичного реагування. Відсутність двостороннього зв’язку між SOAR і TI обмежує можливість формування адаптивних сценаріїв реагування, які б ураховували контекст, достовірність і релевантність отриманої розвідувальної інформації.

Дослідження вітчизняних і зарубіжних авторів [1, 4, 5] свідчать, що існуючі SOAR-рішення переважно зосереджені на автоматизації окремих етапів реагування (наприклад, ізоляції вузла, блокування IP-адреси чи відправлення сповіщення), проте не забезпечують динамічної адаптації сценаріїв залежно від характеристик загрози, визначених інструментами ThreatIntelligence [1]. Додатковою проблемою є низька інтероперабельність між різними форматами обміну даними (STIX, TAXII, OpenIOC) [2, 3, 6], що ускладнює інтеграцію розвідувальних джерел у єдине середовище

реагування. Відсутність уніфікованих моделей взаємодії SOAR ↔ TI призводить до втрати релевантних даних, затримок у реагуванні та надмірного навантаження на аналітиків SOC [4, 5].

Отже, актуальність даного дослідження полягає у необхідності створення інтегрованої технології автоматизованого управління кіберзагрозами, що поєднує Threat Intelligence з адаптивними сценаріями SOAR для забезпечення динамічного, контекстно орієнтованого реагування. Запропонований підхід спрямований на подолання існуючих обмежень традиційних SOAR-систем і підвищення ефективності управління кіберінцидентами в умовах швидко змінюваного середовища.

Мета роботи полягає у підвищенні ефективності управління кіберзагрозами в корпоративних інформаційних системах шляхом автоматизації процесів виявлення, аналізу та реагування на інциденти безпеки з використанням оркестрованої взаємодії засобів SOAR і платформ Threat Intelligence.

Для досягнення поставленої мети необхідно розв’язати такі **завдання***:*

1. Провести аналітичний огляд процесу управління кіберзагрозами в корпоративних інформаційних системах.
2. Сформувати концептуальні засади та архітектуру технології автоматизованого управління кіберзагрозами.
3. Провести експериментальне дослідження розробленої технології, оцінити її ефективність і сформулювати практичні рекомендації щодо впровадження.

Об’єкт дослідження – процес управління кіберзагрозами в корпоративних інформаційних системах.

Предмет дослідження – технологія автоматизованого управління кіберзагрозами на основі інтеграції SOAR-платформи з інструментами Threat Intelligence.

Методи дослідження. Для досягнення поставленої мети використано комплекс наукових і прикладних методів:

– аналітичні методи – для огляду сучасних технологій SOAR,

ThreatIntelligence та аналізу тенденцій у сфері кібербезпеки;

- системний аналіз – для формалізації вимог до інтегрованої системи та визначення її структури;
- моделювання процесів – для побудови архітектури технології автоматизованого управління кіберзагрозами;
- експериментальні методи – для тестування прототипу системи на вибірці типових кіберінцидентів;
- порівняльний аналіз – для оцінки ефективності впровадженої технології у порівнянні з традиційними підходами реагування.

Наукова новизна одержаних результатів полягає в тому, що в роботі проведено удосконалення архітектури системи управління кіберінцидентами шляхом інтеграції платформи автоматизованого реагування з джерелами Threat Intelligence для автоматичного збагачення подій контекстом про актуальні загрози та формування сценаріїв реагування на основі індикаторів компрометації, що дозволяє підвищити швидкість виявлення і блокування кіберзагроз.

Теоретичне та практичне значення полягає у створенні технологічних рішень, що дозволяють підвищити ефективність процесів реагування на кіберінциденти завдяки інтеграції систем оркестрації та автоматизації реагування (SOAR) з платформами розвідки кіберзагроз (Threat Intelligence).

Галузь застосування. Матеріали роботи можуть бути використані для автоматизації рутинних операцій, зменшення впливу людського чинника та забезпечення безперервного циклу адаптивного реагування на сучасні кіберзагрози. Розроблена технологія може бути використана у центрах моніторингу інформаційної безпеки (SOC), підрозділах кіберзахисту банківських, енергетичних, державних та IT-структур, які працюють із великим обсягом подій безпеки в реальному часі.

Апробація результатів дипломної роботи. Основні положення роботи викладалися:

1) В тезах доповіді на https://zenodo.org/records/17822524 (https://doi.org/10.5281/zenodo.17822524)

РОЗДІЛ 1 СУЧАСНИЙ СТАН ТА ПІДХОДИ ДО ВИЯВЛЕННЯ Й УПРАВЛІННЯ КІБЕРІНЦИДЕНТАМИ

1.1 Порушення інформаційної безпеки та їх наслідки в контексті сучасних кіберзагроз

У сучасному цифровому середовищі забезпечення захисту інформації є критичним завданням для будь якої організації. Порушення інформаційної безпеки можуть призвести до фінансових втрат, пошкодження репутації та порушення безперервності бізнес-процесів. Технологічний прогрес створив нові типи загроз, включаючи витоки даних, атаки на інфраструктуру та поширення шкідливого програмного забезпечення. Це вимагає від організацій комплексних заходів захисту, що гарантують конфіденційність, цілісність і доступність інформаційних ресурсів.

Зловмисні порушення є однією з найнебезпечніших категорій загроз для інформаційних систем. До них належать хакерські атаки, зокрема *DDoS (Distributed Denial of Service) атаки*, які перевантажують сервери та блокують доступ до ресурсів. Розподілена атака на відмову в обслуговуванні реалізується шляхом перевантаження цільових ресурсів великою кількістю запитів, які генеруються з багатьох пристроїв, об'єднаних у бот мережу. Основна мета таких атак полягає у тому, щоб зробити ресурси недоступними для законних користувачів, що може завдати фінансових збитків або порушити роботу критичних систем.

Атаку ініціює зловмисник, який організовує групу заражених пристроїв, що працюють під його контролем. Запити, що надсилаються під час атаки зазвичай є легітимними, але їхній надмірний обсяг призводить до перевантаження ресурсів жертви. У результаті цього сервери не можуть обробляти нові запити, що викликає зниження продуктивності або повне відключення від мережі. Окрім організації атак на відмову в обслуговуванні, бот-мережі можуть використовуватись для розсилки

шкідливого контенту, що дестабілізує роботу інформаційних систем.

Атака Man-in-the-Middle (MitM), або «людина посередині», є іншим небезпечним видом кібератак, що полягає у перехопленні комунікації між двома сторонами. Основною метою є отримання доступу до конфіденційної інформації, такої як паролі або банківські дані, без відома учасників. Зловмисник втручається у процес обміну даними, підміняючи з'єднання та стаючи посередником у каналі зв'язку. Користувач та веб застосунок взаємодіють, не підозрюючи, що всі дані проходять через пристрій зловмисника. Така атака дозволяє не лише перехоплювати, але й маніпулювати інформацією, створюючи додаткові загрози для цілісності даних.

Phishing - використовує методи соціальної інженерії для отримання доступу до конфіденційної інформації. Зловмисник створює фішинговий електронний лист та надсилає його потенційній жертві. Жертва, довіряючи вигляду листа, переходить на фальшивий веб-сайт, що імітує реальний. Цей сайт збирає облікові дані, які вводить жертва. Зібрана інформація надходить до зловмисника, який використовує її для доступу до справжніх облікових записів. Фішингові атаки поєднують технічні та психологічні методи для обману користувачів.

Keylogger - призначене для перехоплення введення з клавіатури. Таке програмне забезпечення фіксує всі натискання клавіш, включаючи паролі, банківські дані та приватні ключі від криптовалютних гаманців. Сучасні варіанти також встановлюють Keylogger зі знімком екрана вже та транслюють зображення екрана в режимі реального часу. Зловмисник може спостерігати за всіма діями користувача, включаючи роботу з криптовалютними біржами та також перехоплювати введення з клавіатури.

Програмне забезпечення потрапляє на комп'ютер через заражені вкладення в електронних листах або завантаження з ненадійних джерел наприклад реклами яка може подарувати щось (вебінар). Після запуску воно маскується під системний процес, щоб уникнути виявлення антивірусом. Зібрана інформація передається

зловмиснику через зашифровані з'єднання. Особливу небезпеку становлять випадки, коли отримується доступ до електронної пошти користувача. Багато користувачів зберігають резервні коди для відновлення доступу до програм автентифікації, таких як *Google Authenticator*, у своїй поштовій скриньці. Сучасні програми автентифікації дозволяють синхронізувати налаштування через хмарні сервіси, пов'язані з обліковим записом пошти. Отримавши доступ до пошти, зловмисник може відновити всі налаштування автентифікації на своєму пристрої і генерувати коди для входу в захищені сервіси. Це особливо критично для облікових записів на криптовалютних біржах, де двофакторна автентифікація є основним механізмом захисту від несанкціонованого виведення активів.

Фотографії, зроблені на мобільному пристрої, часто автоматично синхронізуються з хмарними сервісами, такими як Google Photos. Користувач може зробити знімок екрана з резервними кодами автентифікації, не усвідомлюючи, що він автоматично завантажується у хмарне сховище, доступне через веб-інтерфейс пошти. Шкідливе програмне забезпечення також здатне отримувати доступ до файлів на комп'ютері. Якщо користувач зберігає приватні ключі або початкові фрази для відновлення, до прикладу, криптовалютних гаманців у текстових файлах, зловмисник може легко їх виявити.

Основна мета таких атак полягає в отриманні доступу до фінансових активів користувача. У контексті криптовалют це означає можливість виведення всіх коштів з облікового запису на біржі або з особистого гаманця на адресу зловмисника. На відміну від традиційних банківських транзакцій, операції з криптовалютами є незворотними після того, як кошти переведені, їх практично неможливо повернути.

На основі практичного аналізу процедур безпеки декількох криптовалютних бірж було виявлено ряд ключових кіберзагроз, які представлено далі у вигляді *конкретних прикладів,* що демонструють їх ефективність. На таких біржах, як Binance або HTX, користувачі можуть стати жертвами шкідливого програмного забезпечення, яке перехоплює паролі та коди автентифікації. Зловмисник може

спостерігати в режимі реального часу, як користувач заповнює форму виведення криптовалюти, отримуючи інформацію про адреси гаманців та суми транзакцій.

Категорію загроз становлять вразливості в реалізації захисних механізмів самих криптовалютних сервісів. Аналіз процедур безпеки на біржах виявив випадки, коли захисні механізми могли бути обійдені через недоліки в бізнес логіці або недостатню захисту.

Одним із виявлених типів вразливостей є можливість обходу процедур багатофакторної верифікації операцій виведення. У типовому сценарії користувач налаштовує кілька методів підтвердження критичних операцій, включаючи верифікацію через SMS, електронну пошту та TOTP. Після ініціації операції виведення виявилася можливість змінити налаштування методів верифікації та завершити операцію без застосування тих механізмів захисту, що були активні на момент її початку. Технічно це відбувалося через відсутність повторної валідації операції при зміні критичних налаштувань безпеки. Користувач міг встановити новий метод автентифікації, наприклад Passkey, та одразу відключити попередній метод, такий як SMS-верифікація. Система не вимагала підтвердження операції з урахуванням нових налаштувань та не застосовувала обов'язковий період утримання. У результаті транзакція виведення завершувалася успішно без застосування жодного з методів верифікації. Цей випадок було виявлено на біржі HTX та згодом виправлено після повідомлення через програму винагород за виявлення вразливостей.

Іншим типом вразливостей є недостатня валідація операцій на серверній стороні при наявності обмежень, реалізованих виключно в клієнтському інтерфейсі. У деяких випадках блокування операцій виведення реалізовувалося лише через модифікацію елементів веб-інтерфейсу, без відповідної перевірки на сервері. Користувач з базовими технічними навичками міг видалити блокуючі елементи через інструменти розробника браузера (Developer Tools) та ініціювати заборонену операцію. Коли виведення коштів було заблоковано, веб-інтерфейс відображав

відповідне повідомлення та деактивував елементи керування. Однак ці обмеження були реалізовані виключно на рівні представлення в браузері користувача. Через консоль розробника можна було знайти та видалити елементи структури документа, що містили блокування. Після їх видалення кнопка виведення коштів ставала активною. Критична проблема полягала в тому, що сервер не виконував незалежну перевірку наявності активних обмежень при отриманні запиту на виведення. Така вразливість була виявлена на біржі Binance та також виправлена після повідомлення.

Додатковим прикладом є ситуація на біржі Bitstamp, де зміна пароля облікового запису не призводила до автоматичного блокування операцій виведення на встановлений період. Стандартна практика безпеки передбачає введення тимчасових обмежень на критичні операції після будь-яких змін у налаштуваннях безпеки. У даному випадку користувач міг змінити пароль та одразу ініціювати виведення коштів без додаткових затримок. Це створювало ризик, що зловмисник міг змінити пароль та негайно вивести всі активи.

Виявлені вразливості були повідомлені відповідним платформам через програми винагород за виявлення помилок та згодом виправлені. Це підкреслює важливість систематичного тестування безпеки та активної співпраці між дослідниками безпеки та операторами криптовалютних платформ.

Навмисні порушення інформаційної безпеки здійснюються зловмисниками з конкретними цілями, які можуть включати фінансову вигоду, корпоративне шпигунство або саботаж. Основними тригерами є можливість отримання доступу до критично важливої інформації або завдання шкоди репутації компанії. У таблиці 1.1 наведено основні причини навмисних порушень та їх наслідки в контексті криптовалютних систем.

Таблиці 1.1

Основні причини навмисних порушень та їх наслідки

Причина порушення	**Характеристика**	**Метод реалізації**	**Можливі наслідки**	**Приклад**
Фінансова вигода через компрометацію облікових записів	Викрадення криптовалютних активів з облікових записів користувачів	Використання шкідливого ПЗ для перехоплення паролів і кодів автентифікації	Повна втрата коштів користувача, незворотність транзакцій	Впровадження Keylogger з метою доступу до бірж
Експлуатація вразливостей процедур верифікації	Обхід механізмів багатофакторної автентифікації	Маніпуляція налаштуваннями безпеки під час виконання критичних операцій	Несанкціоноване виведення активів без підтвердження	Обхід SMS-верифікації через встановлення Passkey на HTX
Використання недоліків клієнтської валідації	Обхід блокувань операцій через маніпуляцію інтерфейсом	Видалення елементів блокування через Developer Tools браузера	Виконання заборонених операцій виведення коштів	Модифікація веб-інтерфейсу на Binance
Компрометація систем двофакторної автентифікації	Отримання доступу до резервних кодів або синхронізованих налаштувань	Доступ через скомпрометовану пошту до хмарних резервних копій Google Authenticator	Повний контроль над обліковим записом	Відновлення налаштувань автентифікації через пошту
Викрадення приватних ключів та початкових фраз	Пошук конфіденційних даних у файловій системі або хмарних сховищах	Автоматичне сканування файлів на наявність криптовалютних даних	Втрата доступу до криптовалютних гаманців	Виявлення текстових файлів з seed-фразами
Корпоративне шпигунство	Збір інформації про методи захисту та архітектуру систем	Аналіз процедур безпеки через тестування на проникнення	Виявлення системних вразливостей для подальших атак	Систематичне дослідження механізмів верифікації

Ненавмисні порушення інформаційної безпеки виникають через людські помилки, недбалість або недостатнє розуміння безпеки. Ці інциденти не є результатом злого вчинку, але їхній вплив може бути критичним. Наприклад, неправильне налаштування доступу, відправлення конфіденційної інформації на невірну адресу або недотримання рекомендацій щодо створення надійних паролів можуть призводити до витоку даних.

Часто ненавмисні порушення виникають через відсутність належного навчання персоналу або недостатню автоматизацію процесів безпеки. В умовах сучасної цифрової трансформації такі порушення є одним із найпоширеніших джерел інцидентів і відповідно ризиків ІБ. У таблиці 1.2 представлено аналіз основних типів ненавмисних порушень.

Таблиця 1.2

Типи ненавмисних порушень безпеки в криптовалютних системах

Тип порушення	**Причина**	**Механізм виникнення**	**Можливі наслідки**	**Приклад сценарію**
Зберігання резервних кодів у незахищених місцях	Відсутність обізнаності про ризики	Збереження кодів автентифікації в пошті або хмарних сервісах	Компрометація двофакторної автентифікації через витік пошти	Користувач надсилає собі знімок екрана з кодами Google Authenticator
Автоматична синхронізація конфіденційних даних	Налаштування за замовчуванням у мобільних пристроях	Автоматичне завантаження фото з seed-фразами у Google Photos	Несанкціонований доступ до приватних ключів через хмарне сховище	Фотографія паперового гаманця синхронізується автоматично
Використання простих або повторюваних паролів	Недостатнє розуміння важливості унікальних паролів	Використання одного пароля для пошти та криптовалютної біржі	Ланцюгова компрометація облікових записів після витоку	Користувач використовує однаковий пароль для Gmail та Binance
Ігнорування оновлень безпеки	Відкладання встановлення критичних патчів	Використання застарілих версій ПЗ з відомими вразливостями	Експлуатація публічно відомих вразливостей	Користувач не оновлює браузер з відомою вразливістю
Відсутність перевірки адрес при виведенні	Неуважність під час копіювання криптовалютних адрес	Помилка в символі адреси призводить до втрати коштів	Незворотна втрата криптовалюти через неіснуючу адресу	Користувач помиляється в символі при введенні адреси
Довіра до підозрілих повідомлень	Відсутність базових знань про Phishing	Перехід за посиланнями в листах, що імітують офіційні повідомлення	Введення облікових даних на фішинговому сайті	Користувач вводить пароль на фальшивій копії Binance
Збереження конфіденційних даних у текстових файлах	Незручність запам'ятовування складних даних	Створення документів з паролями на робочому столі	Компрометація всіх облікових записів при зараженні Keylogger	Файл "passwords.txt" на робочому столі

Розглядати вектори загроз для зловмисних і ненавмисних порушень важливо для розуміння їх природи та впливу на інформаційні системи. Зловмисні дії мають цілеспрямованість і часто використовують складні подходи. Ненавмисні порушення є наслідком помилок або недбалості. У таблиці 1.3 ключові відмінності між векторами загроз.

Таблиця 1.3

Порівняння векторів загроз для криптовалютних систем

Критерій	**Зловмисні порушення**	**Ненавмисні порушення**	**Реальні кейси**
Мотивація	Фінансове збагачення через викрадення криптоактивів	Зручність використання без розуміння ризиків	Впровадження Keylogger vs збереження паролів у пошті
Складність реалізації	Вимагає технічних знань або спеціалізованих інструментів	Виникає природним чином при звичайному користуванні	Обхід SMS через Passkey vs випадкова синхронізація фото
Час до виявлення	Може залишатися непоміченим тижнями або місяцями	Зазвичай виявляється після настання наслідків	Тривала робота шкідливого ПЗ vs виявлення після втрати доступу
Можливість відновлення	Практично неможливо через незворотність криптовалютних транзакцій	Залежить від типу порушення та наявності резервних копій	Виведені кошти не повертаються vs можливість відновлення доступу
Масштаб впливу	Цільові атаки на користувачів з високими балансами	Може вплинути на будь-якого користувача	Цілеспрямований вибір жертв vs випадкова компрометація
Методи запобігання	Технічні засоби захисту багаторівнева верифікація	Навчання користувачів, автоматизація захисту	Серверна валідація vs примусове використання менеджерів паролів

Вплив на репутацію	Серйозні репутаційні втрати при масових інцидентах	Відповідальність розподіляється між платформою і користувачем	Вразливості в системі HTX/Binance vs помилки користувачів
Типові сценарії	Експлуатація логічних вразливостей, Phishing, Keylogger	Слабкі паролі, незахищене зберігання даних	Обхід SMS-верифікації vs клік на фішинговий лінк

Порушення інформаційної безпеки призводять до різних наслідків залежно від типу інциденту та швидкості реагування. Зловмисні атаки зазвичай спричиняють серйозні фінансові втрати та репутаційні ризики, тоді як ненавмисні помилки частіше призводять до тимчасових збоїв у роботі систем. Основними що визначають масштаб наслідків, є рівень захищеності системи, підготовки персоналу та наявність планів реагування на інциденти. У криптовалютних системах наслідки порушень мають специфічний характер через незворотність транзакцій. Якщо зловмисник виводить кошти з облікового запису повернути їх практично неможливо. Це кардинально відрізняється від традиційних банківських систем, де можна оскаржити підозрілу транзакцію або заблокувати рахунок. У таблиці 1.4 наведено порівняння наслідків для різних типів порушень.

Таблиця 1.4

Наслідки порушень безпеки в криптовалютних системах

Тип порушення	**Безпосередні наслідки**	**Короткострокові ризики**	**Довгострокові наслідки**
Компрометація через Keylogger	Викрадення паролів та кодів 2FA	Виведення всіх активів протягом годин	Повна втрата коштів без можливості повернення
Експлуатація вразливостей біржі	Обхід механізмів верифікації	Несанкціоноване виведення без SMS-підтвердження	Масові втрати користувачів судові позови до біржі
Phishing	Введення облікових даних на фальшивому сайті	Зловмисник отримує доступ до облікового запису	Втрата контролю над обліковим записом назавжди
Компрометація електронної пошти	Доступ до кодів Google Authenticator	Відновлення 2FA на пристрої зловмисника	Повний контроль над усіма захищеними сервісами

Помилка користувача при виведенні	Відправка коштів на неправильну адресу	Миттєва втрата криптовалюти	Незворотна втрата через помилку в одному символі
Автоматична синхронізація фото	Завантаження фото з seed-фразою в Google Photos	Потенційна компрометація через витік хмари	Втрата доступу до всіх гаманців при витоку облікового запису

Порушення безпеки впливають на три основні аспекти: конфіденційність, цілісність та доступність. Конфіденційність порушується коли несанкціоновані особи отримують доступ до приватних даних. Цілісність страждає при модифікації або видаленні інформації без дозволу. Доступність порушується, коли законні користувачі не можуть отримати доступ до своїх ресурсів. У таблиці 1.5 показано, як різні типи атак впливають на ці аспекти.

Таблиця 1.5

Вплив порушень на основні аспекти безпеки

Аспект безпеки	**Тип загрози**	**Механізм впливу**	**Конкретний приклад**
Конфіденційність	Keylogger	Перехоплення всіх введених даних включно з паролями	Викрадення приватних ключів від Ethereum-гаманця
	Компрометація пошти	Доступ до листів з резервними кодами 2FA	Зловмисник бачить всі backup codes від Binance
	Phishing	Користувач добровільно вводить дані на фальшивому сайті	Введення seed-фрази на фейковому сайті MetaMask
Цілісність	Зміна налаштувань безпеки	Зловмисник змінює email для відновлення доступу	Власник втрачає можливість повернути обліковий запис HTX
	Модифікація адреси виведення	Заміна адреси в буфері обміну через malware	Кошти йдуть на адресу зловмисника замість призначеної
	Обхід клієнтської валідації	Видалення блокування через Developer Tools	Виконання операції виведення через встановлені обмеження
Доступність	Втрата доступу до 2FA	Користувач втратив телефон без резервних кодів	Неможливість увійти в обліковий запис Binance

	Зміна пароля зловмисником	Блокування доступу законного власника	Користувач не може увійти після фішингової атаки
	DDoS-атака на біржу	Перевантаження серверів біржі запитами	Неможливість виконати термінові операції під час обвалу ринку

Реальні інциденти демонструють масштаб можливих наслідків у криптовалютній індустрії. Аналіз найбільших злочинів за період 2014-2022 років показує зростаючу складність атак та їх фінансовий вплив.

У 2014 році біржа Mt.Gox оголосила про банкротство після втрати 850 тисяч Bitcoin через хакерську атаку. Це був найбільший злом криптовалютної біржі на той момент, що призвело до втрати близько 450 мільйонів доларів. Біржа так і не змогла повернути всі кошти користувачам, і справа розглядалася в судах протягом багатьох років. [43]

У 2016 році сталася атака на Bitfinex, внаслідок якої було викрадено 120 тисяч Bitcoin вартістю близько 72 мільйонів доларів. Зловмисники використали вразливість у системі мультипідпису, що дозволяло виконувати транзакції без належної верифікації. Біржа була змушена призупинити торгівлю на кілька днів, а всі користувачі зазнали втрат через розподіл збитків пропорційно до балансів. [44]

У 2018 році атака на Coincheck призвела до викрадення 523 мільйонів токенів NEM вартістю близько 530 мільйонів доларів. Причиною стало зберігання коштів у гарячому гаманці замість холодного сховища, що дало зловмисникам змогу отримати доступ до приватних ключів. Це був один із найбільших злочинів в історії криптовалют за обсягом викрадених коштів. [45]

У 2019 році біржа Binance зазнала атаки, під час якої зловмисники вивели 7 тисяч Bitcoin вартістю 40 мільйонів доларів. Хакери використовували комбінацію методів, включаючи Phishing-атаки, шкідливе програмне забезпечення та соціальну інженерію для отримання великої кількості API-ключів і кодів двофакторної автентифікації. Binance відшкодувала всі втрати користувачів зі свого резервного

фонду SAFU, але інцидент продемонстрував вразливість навіть найбільших криптовалютних платформ. [46]

У 2021 році атака на Poly Network стала однією з найбільших у сфері децентралізованих фінансів. Зловмисник викрав криптовалюти на суму понад 600 мільйонів доларів, використавши вразливість у смарт-контракті між різними блокчейн-мережами. Цікаво, що хакер згодом повернув більшу частину коштів, заявивши, що хотів продемонструвати слабкі місця в системі. Цей випадок показав, наскільки вразливими можуть бути навіть найсучасніші DeFi-протоколи. [47]

У 2022 році міст Ronin Network, що використовується для гри Axie Infinity, зазнав атаки з викраденням 625 мільйонів доларів у Ethereum та USDC. Зловмисники скомпрометували приватні ключі валідаторів мережі, що дозволило їм схвалювати фальшиві транзакції виведення. Атака залишалася непоміченою протягом шести днів, що дало хакерам достатньо часу для виведення всіх коштів. Цей інцидент продемонстрував критичну важливість розподіленого управління валідаторами в блокчейн-мостах. [48]

Також у 2022 році виявився скандал з біржею FTX, коли з'ясувалося, що платформа використовувала кошти клієнтів для ризикових операцій через пов'язану компанію Alameda Research. Це призвело до втрати понад 8 мільярдів доларів клієнтських коштів та банкрутства біржі. Засновник біржі Сем Бенкман-Фрід був засуджений до 25 років ув'язнення за шахрайство та відмивання грошей [49]

Ці приклади демонструють, що навіть великі біржі з потужними системами безпеки можуть стати жертвами атак.

Спільною рисою більшості інцидентів є використання комбінації різних методів – технічних вразливостей, соціальної інженерії та людських помилок [50-64]. У таблиці 1.6 систематизовано основні інциденти та їх наслідки.

Таблиця 1.6

Великі інциденти безпеки в криптовалютній індустрії

Інцидент	Рік	Метод атаки	Сума втрат	Основні наслідки
Mt.Gox	2014	Злом гарячих гаманців	850 тис. BTC (~$450 млн)	Банкрутство біржі судові справи протягом років
Bitfinex	2016	Вразливість мультипідпису	120 тис. BTC (~$72 млн)	Розподіл збитків між усіма користувачами
Coincheck	2018	Компрометація гарячого гаманця	523 млн NEM (~$530 млн)	Найбільша крадіжка за обсягом на той момент
Binance	2019	Phishing + API-ключі + 2FA	7 тис. BTC (~$40 млн)	Відшкодування зі страхового фонду біржі
Poly Network	2021	Вразливість смарт-контракту	$600+ млн	Повернення коштів після громадського тиску
Ronin Network	2022	Компрометація ключів валідаторів	$625 млн	Виявлення через 6 днів після атаки
FTX	2022	Внутрішнє зловживання коштами	$8+ млрд	Банкрутство кримінальне переслідування засновника

1.2Типові ознаки порушень інформаційної безпеки як індикатори кіберінцидентів

Виявлення порушень інформаційної безпеки є критично важливим етапом для мінімізації збитків та запобігання подальшим атакам. Типові ознаки порушень залежать від їх природи: зловмисні дії часто супроводжуються спробами приховати сліди, тоді як ненавмисні помилки зазвичай помітні через їх наслідки. Розпізнавання таких ознак дозволяє своєчасно реагувати на інциденти та запобігати їх розвитку.

До ознак зловмисних дій належать багаторазові спроби входу з невірними обліковими даними, підозріла мережна активність та незвичні зміни у файлах або конфігураціях систем. Ненавмисні порушення виявляються через некоректну роботу систем, помилки у налаштуваннях або випадковий витік інформації. У таблиці 1.7 наведено порівняльний аналіз ключових ознак для обох типів порушень.

Таблиця 1.7

Ключові ознаки порушень інформаційної безпеки

Ознака	**Зловмисні порушення**	**Ненавмисні порушення**	**Приклад у криптосистемах**
Аномальна активність входу	Численні спроби входу з різних IP-адрес за короткий час	Кілька невдалих спроб через забутий пароль	Brute-force атака на обліковий запис Binance
Незвичні операції виведення	Спроби виведення на нові адреси одразу після входу	Помилкове введення неправильної адреси	Keylogger-атака: виведення на адресу зловмисника
Зміни налаштувань безпеки	Відключення 2FA та зміна email для відновлення	Випадкове відключення SMS-верифікації	Компрометований обліковий запис: зміна всіх налаштувань
Використання Developer Tools	Модифікація DOM-елементів для обходу блокувань	Випадкове відкриття консолі розробника	Видалення блокування виведення на Binance через F12
Доступ з незвичних локацій	Вхід з IP-адреси з іншої країни без VPN	Вхід під час подорожі без попередження біржі	Вхід з України потім через 10 хвилин з Китаю
Маніпуляції з методами верифікації	Швидка зміна з SMS на Passkey під час операції виведення	Зміна методу 2FA через втрату телефону	Обхід SMS на HTX через встановлення Passkey
Підозрілі файлові операції	Keylogger сканує диск в пошуках seed-фраз	Збереження приватних ключів у текстовому файлі	Malware знаходить "wallet_backup.txt" на робочому столі
Аномалії в API-активності	Масове використання API-ключів для виведення	Помилкова конфігурація API-дозволів	Викрадені API-ключі використовуються для автоматичного виведення

Ключові ознаки порушень інформаційної безпеки дозволяють вчасно ідентифікувати зловмисні дії та ненавмисні помилки. Зловмисні дії часто супроводжуються аномаліями, які не відповідають типовим паттернам поведінки. Наприклад, якщо користувач зазвичай входить у свій обліковий запис на біржі з IP-

адреси в Україні, а раптом з'являються спроби входу з Китаю або Росії, це вказує на компрометацію облікових даних. Особливо підозрілою є ситуація одночасних спроб входу, що фізично неможливо для легітимного користувача.

Ознакою Keylogger може бути незвичне завантаження системних ресурсів навіть у стані спокою. Користувач може помітити уповільнення роботи комп'ютера при введенні тексту або незвичну мережну активність навіть коли браузер закритий. При компрометації електронної пошти характерною ознакою є наявність прочитаних листів, які користувач не відкривав, або автоматичне переміщення листів від бірж у папку сміття.

Ознакою експлуатації вразливостей у бізнес-логіці біржі є успішне виконання операцій, які мали бути заблоковані. На HTX була виявлена можливість обійти SMS-верифікацію шляхом встановлення Passkey та негайного відключення SMS під час очікування підтвердження операції.

Ненавмисні порушення пов'язані з людськими помилками або недбалістю через недостатню обізнаність або неправильну організацію процесів. Їх наслідки можуть бути не менш серйозними, ніж у випадку навмисних дій. Основними причинами є помилки користувачів, відсутність чітких процедур та недостатній рівень навчання персоналу. У таблиці 1.8 наведено типові ознаки ненавмисних порушень.

Таблиця 1.8

Ознаки ненавмисних порушень інформаційної безпеки

Ознака	**Причина виникнення**	**Механізм виявлення**	**Типовий сценарій**
Файли з конфіденційними даними на робочому столі	Недостатня обізнаність про ризики	Виявлення файлів типу "passwords.txt" або "seeds.txt"	Користувач зберігає seed-фрази в блокноті
Синхронізовані фото з приватними ключами	Автоматичні налаштування Google Photos	Фотографії з QR-кодами або текстом seed-фраз у хмарі	Знімок екрана з MetaMask backup автоматично в хмарі
Використання однакових паролів	Зручність запам'ятовування	Компрометація через витік на стороннньому сервісі	Пароль від Gmail використовується для Binance
Листи з backup	Побоювання втратити	Пошук по ключовим	Email собі з темою

codes у поштовій скриньці	доступ	словам у пошті	"Google Authenticator backup"
Відсутність верифікації адрес при виведенні	Неуважність під час копіювання	Кошти відправлені на неправильну адресу	Помилка в останньому символі Ethereum-адреси
Клік на підозрілі посилання	Недостатнє розуміння Phishing	Введення даних на фальшивому сайті	Перехід з листа "Binance Security Alert" на fake-сайт
Відсутність оновлень безпеки	Відкладання встановлення патчів	Експлуатація відомих вразливостей	Браузер з вразливістю, через яку встановлюється Keylogger
Публічне обговорення балансів	Бажання поділитися успіхом	Користувач стає ціллю для зловмисників	Публікація скріншота з балансом у Telegram-чаті

Ознакою ненавмисного порушення є наявність текстових файлів з конфіденційною інформацією на робочому столі. Користувачі створюють файли "passwords.txt" або "crypto_wallets.txt" для зберігання seed-фраз та приватних ключів. Такі файли стають першою ціллю для Keylogger після зараження системи якщо в него встроенна така функція.

Автоматична синхронізація фотографій створює ризик випадкового розкриття даних. Знімок екрана з seed-фразою автоматично завантажується в Google Photos або iCloud. Збереження резервних кодів Google Authenticator в електронній пошті призводить до того, що пошти автоматично компрометує всі облікові записи.

Використання однакового пароля для пошти та біржі є критичною помилкою. При витоку даних на стороньому сервісі зловмисники перевіряють скомпрометовані комбінації на криптобіржах. Відсутність верифікації адреси перед відправленням криптовалюти призводить до втрати коштів через помилку в одному символі.

Перехід за посиланнями з листів без перевірки домену дозволяє зловмисникам імітувати офіційні повідомлення від Binance або Coinbase. Публічне обговорення балансів у Telegram або Twitter робить користувача ціллю для атак.

1.3 Сучасні системи виявлення та реагування на кіберінциденти: стан і тенденції розвитку

Виявлення та реагування на кіберінциденти вимагає використання комплексу технічних засобів, кожен з яких виконує специфічні функції в загальній системі безпеки. Сучасні організації застосовують різні класи систем для забезпечення повного циклу захисту: від виявлення загроз до їх блокування та аналізу наслідків.

Основними класами технічних засобів є системи виявлення та запобігання вторгненням (IDS/IPS), системи управління інформацією та подіями безпеки (SIEM), рішення для захисту кінцевих точок (EDR/XDR), системи виявлення мережевих загроз (NDR), аналітика поведінки користувачів (UEBA) та антишкідливе програмне забезпечення. Кожен клас систем вирішує специфічні задачі в процесі забезпечення безпеки.

Системи виявлення вторгнень (IDS - Intrusion Detection System) є пасивними системами, що здійснюють моніторинг мережевого трафіку та повідомляють адміністраторів про виявлені загрози. Вони аналізують пакети даних, порівнюють їх з базою відомих сигнатур атак та виявляють аномалії в поведінці мережі. Системи запобігання вторгненням (IPS - Intrusion Prevention System) є активними системами, які не тільки виявляють загрози, але й автоматично вживають заходів для їх нейтралізації в реальному часі. IPS може вікидати шкідливі пакети, блокувати IP-адреси джерел атак або відєднувати підозрілі з'єднання без участі адміністратора.

SIEM-системи (Security Information and Event Management) виконують централізований збір та кореляцію подій з різних джерел інформації. Основними задачами є збір, обробка та аналіз подій безпеки, виявлення атак та порушень політик безпеки в реальному часі. Класична реалізація SIEM складається з компонента збору подій, ядра для кореляції та зберігання, і веб-консолі управління. Компонент збору подій, який у різних виробників може називатися агентом, конектором або колектором, відповідає за отримання подій з джерел. Збір може

здійснюватись в активному режимі, коли збирач підключається до джерела по протоколах RPC або SMB, або в пасивному режимі через Syslog та SNMP. Та компонент збору виконує агрегацію, нормалізацію та фільтрацію сирих подій. Агрегація об'єднує кілька однакових подій в одну, економлячи пропускну здатність каналу. Нормалізація приводить події до загального стандарту та витягує значення різних полів для подальшої обробки.

Наприклад, з події невдалого входу можна витягнути час створення, службу, ім'я користувача, адресу та порт джерела. У ядрі SIEM кожна подія перевіряється правилами кореляції. При відповідності одному або кільком правилам створюється інцидент. Кілька спроб невдалого входу можуть свідчити про спробу підбору пароля. SIEM дозволяє виявляти складні багатоетапні атаки, які неможливо ідентифікувати при аналізі окремих подій.

А рішення для захисту кінцевих точок EDR (Endpoint Detection and Response) забезпечують моніторинг активності на робочих станціях та серверах. Вони відстежують запуск процесів, зміни файлової системи, мережеві з'єднання на рівні операційної системи. EDR дозволяє виявляти шкідливе програмне забезпечення, що використовує раніше невідомі вразливості, та аналізувати ланцюжок дій зловмисника після компрометації системи. Також розширені рішення XDR (Extended Detection and Response) інтегрують дані з кінцевих точок, мережі, хмарних сервісів та інших джерел для комплексного виявлення загроз.

Системи виявлення мережевих загроз (NDR - Network Detection and Response) спеціалізуються на аналізі мережевого трафіку для виявлення аномалій та підозрілої активності. На відміну від IDS/IPS, які базуються на сигнатурах, NDR використовують методи машинного навчання для виявлення незвичної поведінки в мережі. Це дозволяє ідентифікувати нові типи атак, латеральне переміщення зловмисників всередині мережі та витоки даних без відомих сигнатур.

Аналітика поведінки користувачів та сутностей (UEBA - User and Entity Behavior Analytics) фокусується на виявленні аномалій в діях користувачів та систем.

UEBA створює базові профілі нормальної поведінки для кожного користувача та виявляє відхилення від встановлених паттернів. Якщо користувач раптом починає завантажувати великі обсяги даних у незвичний час або отримує доступ до ресурсів, з якими зазвичай не працює, UEBA генерує сповіщення про потенційну компрометацію облікового запису.

Антивірусне програмне забезпечення залишається базовим захистом, незважаючи на розвиток складніших систем вірусів. Сучасні антивірусні рішення використовують не лише сигнатурний аналіз, але й поведінковий аналіз, машинне навчання та хмарні технології для виявлення нових загроз. Вони інтегруються з EDR та іншими системами для багаторівневого захисту.

За аналіз поведінки відповідають UEBA та NDR системи. UEBA моніторить час доступу до ресурсів, обсяги завантажених даних та використання облікових записів. NDR застосовує машинне навчання для виявлення аномалій та латерального переміщення зловмисників всередині мережі.

Реагування на загрози забезпечують IPS, EDR та антивірусне програмне забезпечення. IPS працює на мережевому рівні та здійснює фільтрацію трафіку у реальному часі. EDR функціонує на рівні кінцевих точок та може відключати скомпрометовані вузли від мережі, припиняти виконання підозрілих процесів або блокувати спроби зміни критичних файлів. Антивірусне програмне забезпечення виявляє та видаляє шкідливі файли, блокує запуск підозрілих додатків та запобігає експлуатації вразливостей.

Інтеграція всіх цих систем відбувається в рамках центру операцій безпеки (SOC - Security Operations Center). SOC є централізованим підрозділом, який здійснює цілодобовий моніторинг стану безпеки організації. Аналітики SOC використовують дані з усіх систем безпеки для виявлення, аналізу та реагування на інциденти. SIEM виступає основною платформою для агрегації та кореляції подій, тоді як інші системи надають спеціалізовані дані для різних аспектів безпеки.

Процес роботи в SOC включає кілька етапів. Спочатку різні системи

виявлення генерують сповіщення про потенційні загрози. Ці сповіщення надходять до SIEM, де відбувається їх кореляція та пріоритизація. Аналітики SOC переглядають сповіщення з високим пріоритетом, проводять додаткове розслідування та приймають рішення про необхідність реагування. Для автоматизації рутинних процесів використовуються платформи SOAR (Security Orchestration, Automation and Response), які можуть автоматично виконувати певні дії у відповідь на типові інциденти, включаючи блокування IP-адрес в брандмауері, ізоляцію компрометованих систем або запуск сканування антивірусом. Однак традиційні системи управління кіберінцидентами мають суттєві обмеження, які ускладнюють ефективне реагування на сучасні загрози. Ці обмеження стосуються як архітектури процесів моніторингу, так і рівня автоматизації наявних рішень.

1.3.1 Архітектура типового процесу центру моніторингу та його обмеження

Архітектура типового центру моніторингу базується на централізованій моделі збору та обробки подій безпеки. У центрі цієї моделі знаходиться SIEM-система, яка отримує логи з різноманітних джерел: мережевого обладнання, серверів, робочих станцій, систем захисту та бізнес-додатків. Дані нормалізуються, зберігаються та аналізуються за допомогою правил кореляції для виявлення потенційних інцидентів.

Типовий цикл роботи з інцидентом включає кілька послідовних етапів, на кожному з яких залучені різні системи та процеси. На першому етапі збору даних системи IDS/IPS, EDR, антишкідливе програмне забезпечення, NDR та мережеве обладнання генерують події про підозрілу активність. Ці події автоматично передаються до SIEM через протоколи Syslog, SNMP або спеціалізовані агенти збору. На цьому етапі процес повністю автоматизований та не вимагає участі аналітика.

На другому етапі нормалізації та кореляції SIEM обробляє отримані події, приводить їх до єдиного формату та застосовує правила кореляції для виявлення

паттернів атак. Система автоматично об'єднує пов'язані події з різних джерел та створює алерти при виявленні відповідності правилам. UEBA додатково аналізує поведінку користувачів та виявляє аномалії, що відхиляються від нормальних профілів активності. Цей етап також є переважно автоматизованим, однак вимагає попереднього ручного налаштування правил кореляції та базових профілів поведінки аналітиками.

У третьому етапі тріажу та пріоритизації SIEM автоматично оцінює критичність кожного алерту на основі заздалегідь визначених параметрів, таких як важливість атакованого ресурсу, тип загрози та потенційний вплив на бізнес. Система класифікує алерти за рівнями пріоритету та розміщує їх у черзі для розгляду аналітиками. Однак фінальне рішення про те, які інциденти потребують негайного розслідування, а які можуть бути відкладені, приймає аналітик SOC на основі власного досвіду та розуміння поточної ситуації в організації.

Четвертому етапі розслідування аналітик вручну переглядає деталі алерту, збирає додатковий контекст з різних джерел, аналізує логи та визначає, чи є подія справжнім інцидентом безпеки чи хибно-позитивним спрацюванням. Аналітик може використовувати інтерфейси SIEM для перегляду пов'язаних подій, перевіряти репутацію IP-адрес та доменів через зовнішні сервіси, аналізувати зразки файлів у пісочниці. Цей етап залишається переважно ручним процесом, що вимагає значних часових витрат та залежить від кваліфікації окремих аналітиків.

На п'ятому етапі реагування, після підтвердження справжності інциденту, запускаються дії для його локалізації та усунення. Якщо організація використовує платформу SOAR, частина дій може виконуватись автоматично за заздалегідь визначеними playbook: блокування IP-адрес в брандмауері через IPS, ізоляція скомпрометованих робочих станцій через EDR, запуск сканування антивірусом на підозрілих системах. Однак критичні рішення про масштаб реагування, необхідність відключення сервісів або залучення додаткових ресурсів приймає аналітик вручну.

На шостому етапі документування та звітності система автоматично зберігає всі події, пов'язані з інцидентом, у базі даних SIEM. Аналітик вручну готує детальний звіт про інцидент, що включає опис виявленої загрози, виконані дії з реагування, оцінку впливу на організацію та рекомендації щодо запобігання подібним інцидентам у майбутньому. Цей етап вимагає значних зусиль на збір інформації з різних систем та її систематизацію.

Така архітектура має численні обмеження, які суттєво впливають на ефективність роботи SOC. Надмірна кількість алертів є однією з найсерйозніших проблем. Типовий SOC отримує тисячі сповіщень щодня, більшість з яких виявляються хибно-позитивними спрацюваннями. Аналітики витрачають значний час на перевірку кожного алерту, що призводить до втоми від сповіщень та зниження уваги. У результаті справжні інциденти можуть бути пропущені серед великої кількості помилкових алертів.

Високий рівень хибно-позитивних спрацювань пов'язаний з обмеженнями сигнатурних методів виявлення та недостатньою точністю налаштування правил кореляції. Системи IDS/IPS генерують алерти на основі відповідності трафіку відомим сигнатурам атак, але легітимна активність може іноді відповідати цим паттернам. SIEM створює інциденти на основі правил кореляції, однак ці правила не завжди враховують специфічний контекст організації, що призводить до хибних спрацювань на нормальну діяльність користувачів. Довгий час виявлення та реагування на інциденти залишається значною проблемою. Середній час виявлення (MTTD - Mean Time To Detect) складних атак може становити тижні або навіть місяці, оскільки зловмисники поступово просуваються через мережу, намагаючись залишатися непоміченими. Середній час реагування (MTTR - Mean Time To Respond) залежить від доступності кваліфікованих аналітиків та складності інциденту. Затримки на етапі ручного розслідування значно збільшують загальний час від виявлення загрози до її нейтралізації.

Складність кореляції подій з різних джерел створює додаткові виклики. SIEM повинна обробляти логи з десятків або сотень різних систем, кожна з яких має власний формат даних. Навіть після нормалізації виявлення взаємозв'язків між подіями з різних джерел вимагає складних правил кореляції. Багатоетапні атаки, де зловмисник використовує різні техніки на різних етапах компрометації, можуть залишатися непоміченими, оскільки окремі події не виглядають підозрілими самі по собі. Відсутність єдиного вікна управління інцидентами змушує аналітиків працювати одночасно з кількома різними інтерфейсами. Для повного розслідування інциденту необхідно переключатися між SIEM, EDR, системами управління мережею, інструментами аналізу шкідливого програмного забезпечення та іншими рішеннями. Інформація з різних систем не завжди автоматично синхронізується, що вимагає ручного перенесення даних та збільшує ймовірність помилок. Залежність від кваліфікації окремих аналітиків є критичним обмеженням для багатьох організацій. Ефективне розслідування інцидентів вимагає глибоких знань про інфраструктуру організації, сучасні тактики та техніки зловмисників, специфіку різних типів атак. Досвідчені аналітики можуть швидко ідентифікувати справжні інциденти серед хибних спрацювань та приймати правильні рішення щодо реагування. Проте кількість таких фахівців обмежена, а їх підготовка вимагає значного часу. При звільненні досвідченого аналітика організація втрачає накопичені знання та досвід, що негативно впливає на ефективність роботи SOC. Обмеження пропускної здатності також створюють проблеми для великих організацій. SIEM-системи повинні обробляти та зберігати величезні обсяги логів, що вимагає значних обчислювальних ресурсів та дискового простору. Це призводить до високої вартості експлуатації та необхідності компромісів щодо глибини моніторингу або тривалості зберігання даних. Деякі організації змушені обмежувати кількість джерел даних або зменшувати деталізацію логів для відповідності технічним можливостям системи.

Відсутність достатнього контексту для прийняття рішень ускладнює процес розслідування. SIEM може виявити незвичну активність, наприклад велику кількість

спроб доступу до певного ресурсу, але часто не має інформації про те, чи є ця активність легітимною діяльністю у рамках поточного проекту чи справжньою загрозою. Аналітики повинні вручну збирати додатковий контекст, звертаючись до власників ресурсів, переглядаючи календарі заходів або аналізуючи історичні дані про схожу активність.

1.3.2 Поточний рівень автоматизації в традиційних системах управління інцидентами інформаційної безпеки

Автоматизація в традиційних системах управління інцидентами залишається обмеженою та фрагментованою. Існуючі механізми охоплюють лише частину процесів, тоді як критичні етапи все ще вимагають значної участі людини.

Основними механізмами автоматизації у звичайних системах є правила кореляції SIEM для виявлення підозрілих паттернів. Автоматичне блокування на мережевому рівні реалізується через IPS, який може блокувати IP-адреси джерел атак або відкидати шкідливі пакети без участі адміністратора. Скрипти автоматизації використовуються для виконання рутинних операцій, таких як збір додаткової інформації про підозрілі IP-адреси або перевірка репутації доменів. Шаблони реакцій визначають стандартні процедури для різних типів загроз, однак їх виконання залишається переважно ручним процесом. EDR-рішення можуть автоматично ізолювати скомпрометовані робочі станції від мережі або припиняти виконання підозрілих процесів на основі поведінкового аналізу.

Аналіз життєвого циклу інциденту показує чітке розмежування між автоматизованими процесами та етапами, що вимагають участі аналітика. На етапі збору даних автоматизація є найбільш повною. SIEM автоматично отримує логи з налаштованих джерел цілодобово без ручного втручання. На етапі нормалізації системи автоматично приводять події до єдиного формату та застосовують правила збагачення даних. На етапі кореляції SIEM автоматично виявляє паттерни атак та

створює інциденти, UEBA виявляє аномалії в поведінці користувачів.

На етапі тріажу автоматизація є частковою. Системи оцінюють критичність алертів автоматично, але фінальне рішення про пріоритетність приймає аналітик, враховуючи поточну ситуацію та контекстуальні фактори. На етапі розслідування автоматизація мінімальна. Аналітик вручну збирає додатковий контекст з різних джерел, аналізує логи, перевіряє репутацію IP-адрес. Цей етап є найбільш часозатратним та критично залежить від досвіду аналітика. На етапі реагування автоматизація залежить від наявності SOAR. Без неї всі дії виконуються вручну, з SOAR частина типових дій може виконуватись автоматично за playbook, але критичні рішення все ще приймає людина.

Нестача системної автоматизації виражається у кількох ключових аспектах. Системи не можуть автоматично збагачувати події контекстом для визначення, чи є активність частиною легітимного процесу чи аномалією. Відсутня наскрізна автоматизація життєвого циклу інциденту. Переходи між етапами вимагають ручного втручання, аналітик вручну перемикається між системами. Правила кореляції та playbook є статичними та вимагають ручного оновлення при появі нових загроз. Існуючі системи є реактивними та виявляють загрози після того, як підозріла активність вже відбулася, без можливості прогнозування ймовірних векторів атак.

Зростаюча складність кіберзагроз стимулює розвиток нових технологій. Ключовою тенденцією є перехід від ізольованих механізмів автоматизації до комплексної оркестрації процесів безпеки через платформи SOAR. Оркестрація дозволяє координувати роботу різних систем безпеки, автоматизувати складні багатокрокові процедури реагування та забезпечувати наскрізну видимість всього життєвого циклу інциденту.

Інтеграція з Threat Intelligence стає критично важливим компонентом сучасних систем. Threat Intelligence надає контекст про актуальні загрози, тактики зловмисників та індикатори компрометації. Автоматичне збагачення подій даними Threat Intelligence дозволяє системам краще розуміти природу виявленої активності

та приймати більш обґрунтовані рішення. Платформи SOAR можуть автоматично отримувати оновлення індикаторів компрометації та застосовувати їх для проактивного виявлення загроз ще до реальної атаки.

Використання Threat Intelligence в інтеграції з SOAR дозволяє реалізувати адаптивну автоматизацію, яка динамічно коригує правила виявлення на основі актуальної інформації про загрози. Playbook реагування можуть автоматично оновлюватися при появі нових технік атак, знижуючи залежність від ручного оновлення конфігурацій. Повна реалізація потенціалу оркестрації вимагає нових підходів до побудови архітектури систем управління кіберінцидентами. Необхідні методології для автоматизованого прийняття рішень, механізми для самонавчання систем, інтелектуальні алгоритми для аналізу складних взаємозв'язків між подіями. Ці питання формують основу для подальшого розвитку систем автоматизованого управління кіберінцидентами, що буде детально розглянуто в наступному розділі.

Висновки до першого розділу

У рамках розділу досліджено процеси порушення інформаційної безпеки в інформаційних системах. Розглянуто механізми реалізації зловмисних атак DDoS, Man-in-the-Middle та Phishing. Особливу увагу приділено Keylogger-атакам та їх впливу на криптовалютні системи, включаючи компрометацію двофакторної автентифікації через електронну пошту та доступ до синхронізованих резервних кодів Google Authenticator.

На основі практичного аналізу процедур безпеки криптовалютних бірж виявлено ключові вразливості: обхід SMS-верифікації на HTX через використання Passkey під час очікування підтвердження операції, обхід клієнтських обмежень на Binance через модифікацію веб-інтерфейсу за допомогою Developer Tools, відсутність блокування виведення після зміни пароля на Bitstamp. Ці приклади

демонструють необхідність серверної валідації всіх критичних операцій та важливість багаторівневого підходу до забезпечення безпеки.

На основі аналізу реальних інцидентів Mt.Gox, Bitfinex, Coincheck, Binance, Poly Network, Ronin Network та FTX продемонстровано масштаб можливих втрат у криптовалютній індустрії за період 2014-2022 років. Незворотність криптовалютних транзакцій робить наслідки порушень особливо серйозними порівняно з традиційними фінансовими системами.

Визначено ключові індикатори порушень інформаційної безпеки. Ознаки зловмисних дій включають аномальну активність входу з різних IP-адрес, незвичні операції виведення одразу після входу, швидкі зміни налаштувань безпеки, використання Developer Tools для обходу обмежень. Ознаки ненавмисних порушень включають зберігання конфіденційних даних у текстових файлах, автоматичну синхронізацію фотографій з приватними ключами у хмарні сервіси, використання однакових паролів для різних сервісів, збереження резервних кодів в електронній пошті.

Досліджено сучасні системи виявлення та реагування на кіберінциденти. Визначено основні класи технічних засобів: IDS/IPS для моніторингу та блокування трафіку, SIEM для централізованого збору та кореляції подій, EDR/XDR для захисту кінцевих точок, NDR для аналізу мережевих загроз, UEBA для виявлення аномалій у поведінці користувачів. Розподіл задач: SIEM відповідає за збір та кореляцію подій, UEBA та NDR за аналіз поведінки, IPS, EDR та антишкідливе програмне забезпечення за реагування та блокування. Інтеграція відбувається в рамках центру операцій безпеки (SOC), де SIEM виступає основною платформою агрегації, а SOAR автоматизує рутинні процеси.

Проаналізовано архітектуру типового процесу центру моніторингу та виявлено основні обмеження: надмірну кількість алертів з високим рівнем хибно-позитивних спрацювань, значні затримки між компрометацією та виявленням, складність кореляції подій з різних джерел, відсутність єдиного вікна управління, критичну

залежність від кваліфікації окремих аналітиків. Етапи збору, нормалізації та кореляції є високо автоматизованими, тоді як розслідування та прийняття критичних рішень залишаються переважно ручними процесами.

Визначено нестачу системної автоматизації: відсутність автоматичного збагачення контекстом, наскрізної автоматизації життєвого циклу інциденту, адаптивної автоматизації правил, проактивного виявлення загроз. Визначено ключові тенденції: перехід до комплексної оркестрації через платформи SOAR та інтеграція з Threat Intelligence для автоматичного збагачення подій контекстом про актуальні загрози та адаптивного коригування правил виявлення.

Отримані результати підкреслюють необхідність розробки нових підходів до побудови архітектури систем управління кіберінцидентами з використанням методологій автоматизованого прийняття рішень, механізмів самонавчання та інтелектуальних алгоритмів аналізу подій, що буде детально розглянуто в наступному розділі.

РОЗДІЛ 2 КОНЦЕПТУАЛЬНІ ЗАСАДИ ТА АРХІТЕКТУРА ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ КІБЕРЗАГРОЗАМИ

2.1 Концепція автоматизації процесів реагування на кіберзагрози з використанням технологій SOAR та Threat Intelligence

Традиційні підходи до управління кіберінцидентами, як було встановлено в попередньому розділі, демонструють критичні обмеження щодо швидкості реагування та ефективності обробки великих обсягів оповіщень безпеки. У відповідь на виявлені обмеження в межах даної дипломної роботи пропонується концепція автоматизації процесів реагування на кіберзагрози з використанням технологій SOAR та Threat Intelligence. Передбачається, що автоматизація процесів реагування становить фундаментальну трансформацію операційної парадигми центрів моніторингу, де технологічні засоби координуються для виконання складних завдань з мінімальним залученням людських ресурсів.

Розроблювана концепція базується на переході від моделі Human-in-the-Loop, де аналітик контролює кожну дію системи, до моделі Human-on-the-Loop, де автономні системи виконують визначені завдання під наглядом фахівців. Islam, Babar та Nepal [64] визначають оркестрацію безпеки як планування, інтеграцію, співпрацю та координацію діяльності інструментів безпеки для автоматизації дій у відповідь на інциденти через множину технологічних парадигм. Це визначення підкреслює багатовимірний характер автоматизації, що охоплює не лише виконання технічних операцій, але й координацію між різнорідними системами. Платформи Security Orchestration, Automation and Response інтегрують три основні компоненти для управління інцидентами.

Оркестрація забезпечує координацію різнорідних інструментів безпеки через стандартизовані інтерфейси програмування, дозволяючи системам від різних

виробників взаємодіяти в єдиному операційному процесі. Автоматизація реалізує виконання послідовностей дій без втручання оператора, перетворюючи повторювані процедури в автоматизовані робочі процеси. Реагування структурує обробку інцидентів через формалізовані сценарії дій для різних типів загроз. Таке поєднання створює основу для концепції Force Multiplier, де платформа розширює можливості аналітиків обробляти значно більшу кількість інцидентів без пропорційного збільшення ресурсів. Islam, Babar та Nepal [64] запропонували шестирівневу архітектурну модель SOAR платформ з чітким розподілом відповідальності між шарами. Шар інструментів безпеки забезпечує інтеграцію гетерогенних систем виявлення та захисту. Інтеграційний шар забезпечує зв'язок через різноманітні протоколи комунікації та механізми трансформації даних. Шар обробки даних виконує агрегацію, нормалізацію та зберігання інформації від різних джерел.

Семантичний шар реалізує онтологічну інтерпретацію даних безпеки, використовуючи методи обробки природної мови для класифікації завдань. Шар оркестрації містить механізм виконання робочих процесів та координує складні процедури через інтегровані системи. Шар інтерфейсу забезпечує централізовану візуалізацію, управління справами та конфігурацію процесів.Фундаментальна відмінність між SOAR та традиційними системами управління інформацією про події безпеки полягає в комплементарності операційних ролей. SIEM системи зосереджені на збиранні логів, кореляції подій для виявлення паттернів атак та генерації оповіщень про потенційні інциденти. SOAR платформи споживають оповіщення від SIEM та фокусуються на автоматизації розслідування, тріажу, збагачення контекстом та виконання дій реагування.

Це створює операційний ланцюг, де SIEM виконує моніторинг та виявлення, після чого SOAR автоматизує перевірку, аналіз та нейтралізацію загроз. У таблиці 2.1 наведено порівняльну характеристику SIEM та SOAR систем.

Таблиця 2.1.

Порівняльна характеристика SIEM та SOAR систем

Характеристика	**SIEM**	**SOAR**
Основний фокус	Виявлення загроз через кореляцію подій	Автоматизація реагування на виявлені загрози
Первинні джерела	Логи з інфраструктури організації	Оповіщення систем виявлення та threat intelligence
Ключова функція	Агрегація, кореляція подій	Оркестрація процесів, автоматизація дій
Метод виявлення	Rule-based кореляція	AI/ML-посилена пріоритизація з контекстом
Модель роботи	Human-in-the-Loop	Human-on-the-Loop
Основний результат	Оповіщення про інциденти	Автоматизовані координовані відповіді
Метрика	MTTD (час виявлення)	MTTR (Час реагування)

Threat Intelligence надає контекст для оповіщень безпеки. Це структурована інформація про загрози, що дозволяє приймати обґрунтовані рішення щодо захисту інфраструктури. Академічна література класифікує аналітичну інформацію на чотири типи залежно від аудиторії та часового горизонту. Стратегічна інформація призначена для керівництва, описує високорівневі тренди та геополітичні фактори для довгострокового планування. Тактична інформація фокусується на методах зловмисників, використовуючи MITRE ATT&CK як онтологію для категоризації технік атак. Операційна інформація описує конкретні кампанії та групи для проактивного полювання на загрозі. Технічна інформація містить індикатори компрометації: IP-адреси, домени, хеші файлів для створення правил виявлення.

Технічні індикатори мають найкоротший життєвий цикл, оскільки зловмисники швидко змінюють інфраструктуру атак. Стратегічна інформація залишається актуальною місяцями або роками. Ця темпоральна різниця визначає вимоги до інтеграції: технічні індикатори потребують оновлення в режимі реального часу, стратегічна інформація може інтегруватись періодично.

Стандартизація форматів обміну аналітичною інформацією забезпечує інтероперабельність. STIX, затверджений OASIS у червні 2021 року, являє собою

JSON-based мову для виразу інформації про кіберзагрози. Версія 2.1 визначає об'єкти предметної області: паттерни атак, кампанії, індикатори, шкідливе програмне забезпечення, акторів загроз, вразливості. Об'єкти відношень описують зв'язки між сутностями через типізовані відношення.TAXII забезпечує транспортний протокол для автоматизованого обміну через HTTPS на основі RESTful API. Версія 2.1 визначає два сервіси: колекції для моделі запит-відповідь та канали для моделі публікація-підписка. SOAR використовують STIX та TAXII для автоматизації отримання та розповсюдження інформації, забезпечуючи стандартизовану інтеграцію з джерелами threat intelligence. Інтеграція SOAR з джерелами threat intelligence створює інтелектуально-керовану автоматизацію.

Рішення про дії базуються на поточному контексті загроз, а не на статичних правилах. Збагачення подій виконується в реальному часі через паралельні запити до кількох джерел. Система отримує багатовимірний контекст за секунди. Збагачення індикаторів включає репутацію IP-адрес, геолокацію, історію в чорних списках, дані WHOIS для доменів, ідентифікацію сімейства шкідливого ПЗ для файлів.

Контекстуальне збагачення додає критичність активу, роль користувача, рівень привілеїв. Threat intelligence збагачення включає атрибуцію до груп зловмисників, кореляцію з кампаніями, мапування на MITRE ATT&CK. Playbook являють собою формалізовані сценарії реагування на специфічні типи інцидентів. CACAO версії 2.0, стандарт OASIS, визначає структуру через метадані, граф кроків та переходів, цільові системи, змінні для конфігурації, цифрові підписи для автентичності. Типи кроків включають виконання команд, виклик інших playbook для модульності, паралельне виконання, умовне розгалуження, ітеративні цикли. Інтеграція з STIX дозволяє playbook споживати індикатори для автоматичної відповіді та мапувати тактики на дії. Адаптивна автоматизація модифікує playbook динамічно на основі актуальної threat intelligence. Механізми включають зміну сценаріїв відповідно до поточних даних про загрози, вибір playbook залежно від критичності активу та часу доби, аналіз результатів для налаштування логіки.

Машинне навчання забезпечує автоматичну пріоритизацію оповіщень, категоризацію інцидентів, передбачення часу вирішення. Комерційні платформи впроваджують supervised learning для класифікації за критичністю, unsupervised learning для виявлення аномалій у патернах подій, як відзначають Islam, Babar та Nepal [64].

Узагальнюючи, запропонована концепція передбачає такий операційний ланцюг (див. рис. 2.1):

1. **Агрегування подій безпеки** — система SIEM збирає логи з різнорідних джерел (мережеві пристрої, сервери, кінцеві точки, хмарні сервіси) та нормалізує їх до єдиного формату для подальшої обробки.

2. **Кореляція та виявлення аномалій** — SIEM застосовує правила кореляції та моделі поведінкового аналізу для виявлення підозрілих патернів активності, що можуть свідчити про інцидент безпеки.

3. **Генерація оповіщень** — події, що відповідають критеріям загроз, перетворюються на структуровані оповіщення та передаються до SOAR-платформи для автоматизованої обробки.

4. **Збагачення через Threat Intelligence** — SOAR автоматично запитує зовнішні та внутрішні джерела аналітичної інформації (VirusTotal, AbuseIPDB, внутрішні бази репутації) для отримання контексту про індикатори компрометації, що містяться в оповіщенні.

5. **Багатофакторна пріоритизація** — система оцінює критичність інциденту на основі репутації індикаторів, важливості атакованого активу, рівня привілеїв користувача та інших контекстних параметрів для визначення порядку обробки.

6. **Вибір та виконання playbook** — залежно від типу загрози та пріоритету автоматично обирається відповідний сценарій реагування, що може включати локалізацію загрози, блокування індикаторів, збір додаткових доказів.

7. **Інтеграція з системами захисту** — SOAR виконує дії через API інтеграції з EDR, брандмауерами, системами управління ідентифікацією для реалізації заходів локалізації та нейтралізації.

8. **Передача аналітику або документування** — залежно від результатів автоматичної обробки інцидент або передається фахівцю SOC для ручного розслідування складних випадків, або автоматично документується при успішному вирішенні.

9. **Моніторинг ефективності та адаптація** — система збирає метрики виконання playbook (час обробки, точність виявлення, хибні спрацювання) для подальшого вдосконалення логіки автоматизації, оновлення правил кореляції та налаштування моделей пріоритизації.

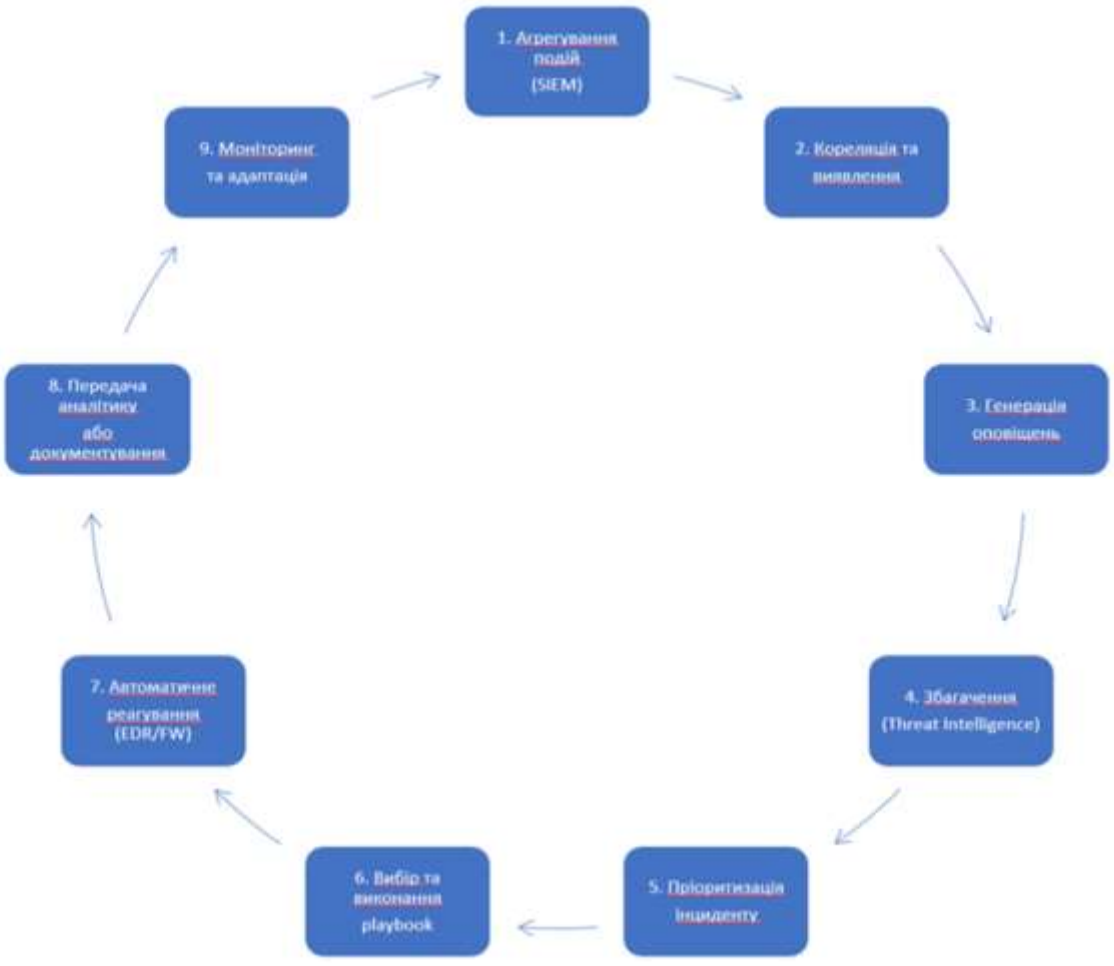

Рис. 2.1. Операційний ланцюг автоматизації процесу реагування на кіберзагрози з використанням технологій SOAR та Threat Intelligence

Таким чином, запропонована концепція автоматизації процесів реагування через інтеграцію SOAR з джерелами threat intelligence забезпечує перехід від реактивної моделі мануального розслідування до проактивної моделі інтелектуально-керованого автоматизованого реагування. Передбачається, що організації зможуть досягти суттєвого скорочення часу виявлення та нейтралізації загроз при ефективнішому використанні обмежених ресурсів аналітиків безпеки.

2.2 Формалізація сценаріїв реагування і алгоритмів автоматизації

У межах запропонованої концепції автоматизації реагування на кіберзагрози критичним елементом є формалізація сценаріїв дій, які система має виконувати в кожній ситуації. Playbook в розробленій моделі представляють собою машинозчитувані інструкції для SOAR-платформи, що визначають послідовність кроків, умови їх виконання та логіку прийняття рішень. Без чіткої формалізації процесів автоматизація втрачає передбачуваність і може призвести до непередбачуваних наслідків для інфраструктури організації.

Життєвий цикл обробки інциденту в запропонованій концепції структурується як послідовність взаємопов'язаних етапів, кожен з яких виконує специфічну роль у процесі реагування. Ідентифікація відбувається при отриманні алерту від систем виявлення (SIEM, EDR, NDR), що сигналізує про потенційну загрозу. Збагачення передбачає автоматичне доповнення події контекстом через запити до джерел threat intelligence для верифікації індикаторів компрометації, перевірки репутації IP-адрес, пошуку кореляцій з попередніми інцидентами. Пріоритизація в розробленій моделі визначає критичність інциденту на основі багатофакторної оцінки, що дозволяє системі вирішити, чи потребує ситуація негайного автоматичного реагування, чи передачі аналітику для детального розслідування. Локалізація реалізується через дії для обмеження поширення загрози: ізоляція скомпрометованих вузлів, блокування

шкідливих IP-адрес у мережевому периметрі. Усунення включає видалення артефактів атаки, відновлення системних конфігурацій, зміну скомпрометованих облікових даних. Завершальний етап документування забезпечує фіксацію всіх виконаних дій для аудиту, аналізу ефективності та вдосконалення playbook. Структуру життєвого циклу інциденту в запропонованій концепції показано на рисунку 2.2.

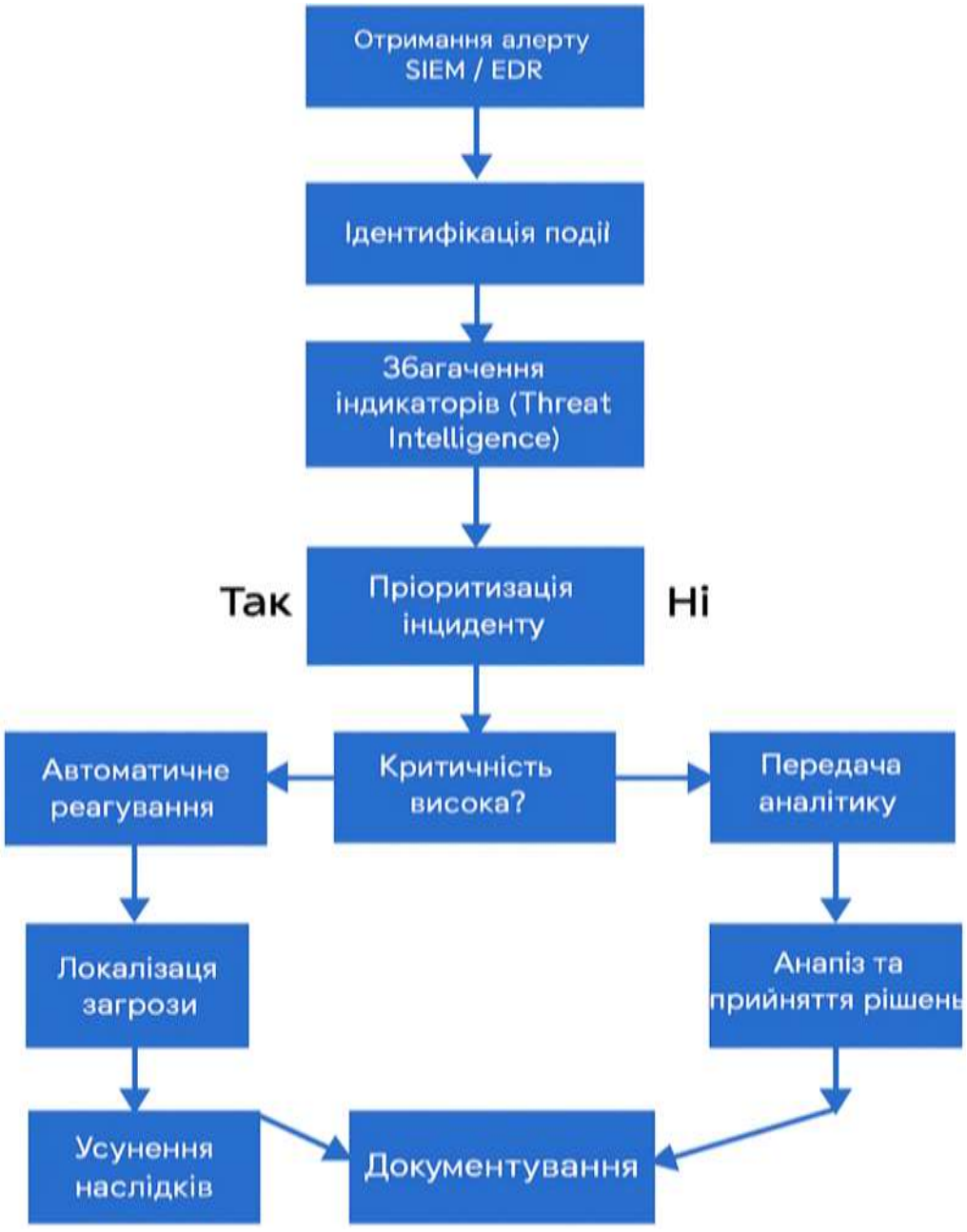

Рис. 2.2 Структура життєвого циклу інциденту в запропонованій концепції автоматизації

У розробленій моделі різні типи інцидентів вимагають диференційованих сценаріїв реагування, оскільки компрометація облікового запису потребує принципово інших дій порівняно з виявленням шкідливого програмного забезпечення на кінцевій точці. Запропонована концепція передбачає створення спеціалізованих playbook для кожної категорії загроз: компрометація облікових записів через підбір паролів або фішинг, шкідливе програмне забезпечення на робочих станціях, мережеві атаки через сканування портів або експлуатацію вразливостей, витік даних при несанкціонованій передачі конфіденційної інформації, інсайдерські загрози від легітимних користувачів з аномальною поведінкою. Відповідність типів інцидентів конкретним автоматизованим діям реагування в запропонованій моделі представлено в таблиці 2.2.

Таблиця 2.2

Матриця відповідності типів інцидентів та дій реагування

Тип інциденту	**Критичність**	**Час реагування**	**Ключові автоматизовані дії**
Компрометація облікового запису	Висока	<15 хв	Блокування облікового запису, примусова зміна пароля, завершення активних сесій
Шкідливе ПЗ на робочій станції	Висока	<10 хв	Ізоляція вузла від мережі, припинення процесу, видалення файлів
Мережеве сканування	Середня	<30 хв	Блокування IP адреси в IPS, створення правила firewall

Витік даних	Критична	<5 хв	Блокування передачі, ізоляція джерела, нотифікація керівництва
Інсайдерська загроза	Висока	<20 хв	Збір доказів активності, обмеження привілеїв, ескалація до HR

У запропонованій концепції структура playbook базується на стандарті CACAO (Collaborative Automated Course of Action Operations), розробленому організацією OASIS, що дозволяє формалізувати сценарії реагування в машинозчитуваному форматі. Розроблена модель передбачає, що кожен playbook містить метадані для ідентифікації (назва, версія, автор, рівень критичності), граф виконання, що визначає послідовність кроків та переходи між ними, змінні для зберігання проміжних даних під час виконання, опис цільових систем з параметрами підключення до інтегрованих інструментів безпеки. В запропонованій моделі кроки playbook класифікуються за типами відповідно до їх функціонального призначення. Крок дії виконує конкретну команду на цільовій системі через API — наприклад, викликає функцію EDR для ізоляції скомпрометованої робочої станції від корпоративної мережі. Крок виклику playbook запускає інший сценарій як підпрограму, що в розробленій архітектурі забезпечує модульність та можливість повторного використання компонентів без дублювання логіки.

Паралельний крок дозволяє запускати кілька дій одночасно — в запропонованій реалізації це використовується, наприклад, для одночасного блокування IP-адреси в брандмауері та додавання її до чорного списку системи запобігання вторгненням. Умовний крок реалізує логіку розгалуження: якщо репутаційна оцінка індикатора перевищує встановлений поріг, система виконує

блокування, інакше — передає інцидент на ручну перевірку. Крок циклу в розробленій моделі обробляє списки об'єктів ітеративно — наприклад, перебирає всі виявлені скомпрометовані хости та послідовно виконує їх ізоляцію. Крок затримки додає контрольовану паузу між діями, що критично для асинхронних операцій, де наступний крок залежить від завершення попереднього. Крок нотифікації забезпечує інформування аналітиків SOC через налаштовані канали комунікації (email, Slack, системи управління інцидентами).

Алгоритм збагачення подій

Збагачення подій реалізується як один з критичних алгоритмів автоматизації. Розроблений алгоритм EnrichAlert передбачає, що при отриманні алерту система автоматично витягує індикатори компрометації (IP-адреси, доменні імена, хеші файлів, URL) та виконує паралельні запити до налаштованих джерел threat intelligence — VirusTotal, AlienVault OTX, внутрішні бази репутації організації. Запити виконуються асинхронно з дотриманням rate limits джерел, що дозволяє отримати відповіді від усіх сервісів за час найповільнішого джерела, а не суму часів всіх запитів.

Зібрана інформація агрегується в єдину репутаційну оцінку з урахуванням вагових коефіцієнтів надійності джерел, доповнюється геолокацією, історією присутності в чорних списках, зв'язками з відомими кампаніями зловмисників, ідентифікацією сімейства шкідливого програмного забезпечення для файлових артефактів. Додатково система збагачує подію організаційним контекстом: критичність атакованого активу для бізнес-процесів, роль та рівень привілеїв користувача, підрозділ організації. Цей комплексний збагачений контекст в запропонованій моделі дозволяє системі відрізнити реальну загрозу від хибного спрацювання.

Алгоритм EnrichAlert(alert)

```
Вхід:
    alert – подія безпеки, отримана від SIEM/EDR/NDR
Вихід:
    enriched_alert – подія з додатковим контекстом TI та організаційними атрибутами
Алгоритм EnrichAlert(alert):
    indicators ← ExtractIndicators(alert)
        // Виділити з алерту всі індикатори: IP, домени, URL, хеші файлів тощо
    indicators ← Unique(indicators)
        // Уникнути дублювання запитів
    if indicators порожній:
        return alert
        // Немає чого збагачувати – повертаємо подію як є
    sources ← [VT, OTX, InternalTI, ...]
        // Набір підключених джерел threat intelligence
    ti_results ← порожня_структура
    // Паралельно опрацьовуємо індикатори, але дотримуємось лімітів TI-джерел
    ПАРАЛЕЛЬНО для кожного ind ∈ indicators:
        для кожного s ∈ sources (з обмеженим пулом запитів):
            resp ← GetFromCacheOrQueryTI(s, ind, timeout=TI_TIMEOUT)
            if resp помилка або timeout:
                resp ← MarkAsFailed(s)
            safe_append(ti_results, ind, resp)
    КІНЕЦЬ_ПАРАЛЕЛЬНОГО_БЛОКУ
    enriched_indicators ← порожній_список
    для кожного ind ∈ indicators:
        data ← ti_results[ind]
        rep_score ← AggregateReputation(data)
            // Агрегація репутаційних оцінок з урахуванням ваг джерел
            // Має коректно працювати, навіть якщо частина джерел не дала даних
        geo        ← ExtractGeo(data)
        blacklist  ← ExtractBlacklistHistory(data)
        campaigns  ← ExtractRelatedCampaigns(data)
        malwarefam ← ExtractMalwareFamily(data)
        enriched_ind ← {
            "value": ind,
            "reputation": rep_score,
            "geo": geo,
            "blacklist_history": blacklist,
            "related_campaigns": campaigns,
            "malware_family": malwarefam
        }
        enriched_indicators.append(enriched_ind)
    КІНЕЦЬ_ЦИКЛУ
    // Додавання організаційного контексту
    asset_ctx ← GetAssetContext(alert)
        // критичність активу, власник, бізнес-процес
    user_ctx ← GetUserContext(alert)
        // роль користувача, рівень привілеїв, підрозділ
    enriched_alert ← Clone(alert)
    enriched_alert.enriched_indicators ← enriched_indicators
    enriched_alert.asset_context ← asset_ctx
    enriched_alert.user_context  ← user_ctx
    return enriched_alert
```

Алгоритм пріоритизації інцидентів

Розроблений алгоритм PrioritizeIncident використовує багатофакторну модель оцінювання для визначення критичності інциденту. Система враховує агреговану репутацію індикаторів з джерел threat intelligence (нормалізовану до шкали 0-100), критичність атакованого активу в інфраструктурі організації (класифікація low/medium/high/critical), аномальність поведінки користувача відносно його базової активності, рівень системних привілеїв облікованого запису, темпоральні фактори (час доби, день тижня). У запропонованій моделі ці фактори комбінуються з ваговими коефіцієнтами (40% — репутація TI, 30% — критичність активу, 20% — привілеї користувача, 10% — аномальність поведінки) для обчислення зведеного ризикового балу, нормалізованого до шкали 0-10. Інциденти з балом понад 7 в розробленій системі тригерують негайне автоматичне реагування через відповідні playbook, інциденти з балом 4-7 передаються аналітикам SOC для детального розслідування, події з балом нижче 4 логуються для подальшого аналізу паттернів.

Алгоритму PrioritizeIncident:

```
Вхід: enriched_alert — подія з TI-контекстом та організаційними атрибутами
Вихід: priority_score — числова оцінка пріоритету (0-10)

Алгоритм PrioritizeIncident(enriched_alert):
    rep ← AggregateIndicatorReputation(enriched_alert.enriched_indicators)
    asset_criticality ← enriched_alert.asset_context.criticality
    user_privileges ← enriched_alert.user_context.privilege_level
    anomaly ← DetectUserOrHostAnomaly(enriched_alert)

    // Нормалізація факторів до шкали 0-1
    rep_normalized ← Normalize(rep, 0, 100)
    asset_normalized ← Normalize(asset_criticality, 1, 4)
    user_normalized ← Normalize(user_privileges, 1, 3)
    anomaly_normalized ← Normalize(anomaly, 0, 1)

    // Зважена комбінація факторів
    priority_score ← 0.4 × rep_normalized +
                     0.3 × asset_normalized +
                     0.2 × user_normalized +
                     0.1 × anomaly_normalized

    // Масштабування до шкали 0-10
    priority_score ← priority_score × 10

    return priority_score
```

Алгоритм автоматизованого розслідування

Автоматизоване розслідування реалізується через алгоритм InvestigateIncident, що розширює контекст інциденту шляхом запитів до різнорідних систем організації. Розроблена модель передбачає, що при виявленні компрометації облікового запису система автоматично витягує історію входів користувача за визначений період (за замовчуванням 30 днів), перелік ресурсів, до яких здійснювався доступ, IP-адреси та географічні локації підключень, інші облікові записи, що використовували ті самі IP-адреси, попередні оповіщення безпеки, пов'язані з цим користувачем. Якщо виявлено шкідливий файл, запропонована система автоматично аналізує батьківський процес (який процес ініціював створення файлу), мережеві з'єднання, встановлені процесом, файли, створені або модифіковані під час виконання, записи в системному реєстрі Windows, наявність того самого хешу на інших хостах в інфраструктурі. У розробленій моделі всі зібрані дані структуруються у вигляді графу зв'язків між об'єктами (користувачі, хости, процеси, файли, мережеві підключення), що дозволяє аналітикам візуалізувати повну картину атаки, включно з можливими шляхами латерального переміщення зловмисника в мережі.

Алгоритм InvestigateIncident:

```
Вхід: enriched_alert — подія з контекстом TI, активу та користувача
Вихід: investigation_report — структура з знахідками та графом зв'язків

     Алгоритм InvestigateIncident(enriched_alert):
         objects ← ExtractObjects(enriched_alert)
             // Витягти об'єкти: користувачі, хости, файли, процеси

         graph ← NewGraph()
             // Створити граф для зв'язків

         для кожного obj ∈ objects:
             siem_events ← QuerySIEM(obj, time_window=30 днів)
             edr_events ← QueryEDR(obj, time_window=30 днів)
             auth_logs ← QueryAuthLogs(obj, time_window=30 днів)

             related_events ← MergeEvents(siem_events, edr_events, auth_logs)

             для кожного event ∈ related_events:
                 graph.add_edge(obj, event)

                 // Додати зв'язані об'єкти
                 linked_objects ← ExtractLinkedObjects(event)
                 для кожного linked ∈ linked_objects:
                     graph.add_edge(obj, linked)
```

```
    КІНЕЦЬ_ЦИКЛУ
КІНЕЦЬ_ЦИКЛУ

attack_paths ← DetectSuspiciousPaths(graph)
    // Виявити підозрілі шляхи атаки

investigation_report ← {
    graph: graph,
    attack_paths: attack_paths,
    objects: objects,
    timeline: BuildTimeline(graph)
}

return investigation_report
```

Умовна логіка в playbooks

Розроблена модель playbooks передбачає використання умовної логіки для адаптації дій залежно від контексту інциденту. Прості булеві умови в запропонованій реалізації перевіряють факт присутності індикатора в попередньо визначених списках: якщо IP-адреса міститься в чорному списку відомих зловмисних джерел, система виконує блокування на мережевому периметрі. Числові порівняння дозволяють встановлювати порогові значення: якщо кількість невдалих спроб автентифікації перевищує 5 за 10 хвилин, розроблений алгоритм блокує обліковий запис та ініціює примусову зміну пароля. Темпоральна логіка в запропонованій концепції враховує час події: інциденти, що відбулися поза робочим часом або у вихідні дні, автоматично отримують підвищений пріоритет, оскільки легітимна активність в ці періоди мінімальна. Логіка множин перевіряє приналежність об'єктів до визначених категорій: чи належить IP-адреса до довірених мереж організації, чи є користувач членом привілейованої групи адміністраторів. Складні умови в розробленій системі комбінують кілька перевірок через логічні операції AND, OR, NOT — наприклад, блокування виконується лише за одночасного виконання умов: репутація індикатора нижче порогу І активність зафіксована в неробочий час І користувач звертався до критичного бізнес-ресурсу.

Алгоритм локалізації загрози

У запропонованій концепції локалізація загрози реалізується через алгоритм ContainThreat, що виконує дії для обмеження поширення атаки в інфраструктурі.

Розроблена модель передбачає автоматичну ізоляцію скомпрометованих хостів від корпоративної мережі через інтеграцію з EDR-рішеннями, що фізично розриває мережеву зв'язність, залишаючи тільки канал управління для подальших дій реагування. Якщо виявлено компрометацію облікового запису, запропонована система автоматично блокує обліковий запис в системі управління ідентифікацією (IAM), ініціює примусову зміну пароля, завершує всі активні сесії користувача в усіх підключених системах. Для мережевих індикаторів з низькою репутацією розроблений алгоритм виконує блокування на рівні брандмауера та додавання в чорні списки системи запобігання вторгненням (IPS), для шкідливих доменів — конфігурацію DNS-sinkholes для переспрямування запитів. У запропонованій моделі всі дії локалізації логуються з фіксацією точного часу виконання, параметрів команд, результатів виконання для забезпечення можливості аудиту та, при необхідності, відкату змін.

Алгоритм ContainThreat(enriched_alert)

```
Вхід: enriched_alert — подія з підтвердженим інцидентом та пріоритетом
Вихід: containment_actions — список виконаних дій локалізації

Алгоритм ContainThreat(enriched_alert):
    actions ← порожній_список

    host ← enriched_alert.asset_context.host
    user ← enriched_alert.user_context.user_id
    indicators ← enriched_alert.enriched_indicators

    // Ізоляція скомпрометованого хоста
    if host ≠ NULL та host.under_attack = TRUE:
        result ← EDR.IsolateHost(host)
        actions.append({
            action: "isolate_host",
            target: host,
            result: result,
            timestamp: NOW()
        })

    // Блокування облікового запису
    if user ≠ NULL та enriched_alert.type = "account_compromise":
        result1 ← IAM.DisableAccount(user)
        result2 ← IAM.ForcePasswordReset(user)
        result3 ← IAM.TerminateActiveSessions(user)

        actions.append({action: "disable_account", target: user, result: result1})
        actions.append({action: "force_password_reset", target: user, result: result2})
        actions.append({action: "terminate_sessions", target: user, result: result3})

    // Блокування шкідливих індикаторів
```

```
для кожного ind ∈ indicators:
    if ind.type = "ip" та ind.reputation < THRESHOLD_IP:
        result ← Firewall.BlockIP(ind.value)
        actions.append({action: "block_ip", target: ind.value, result: result})

    if ind.type = "domain" та ind.reputation < THRESHOLD_DOMAIN:
        result ← DNS.SinkholeDomai(ind.value)
        actions.append({action: "sinkhole_domain", target: ind.value, result: result})

    if ind.type = "hash" та ind.malicious = TRUE:
        result ← AV.QuarantineFile(ind.value)
        actions.append({action: "quarantine_file", target: ind.value, result: result})
КІНЕЦЬ_ЦИКЛУ

return actions
```

Паралелізація виконання

У запропонованій концепції паралелізація виконання використовується для скорочення загального часу реагування на інциденти. Розроблена модель передбачає, що при необхідності ізоляції множини скомпрометованих хостів (наприклад, 20 робочих станцій при виявленні поширення шкідливого програмного забезпечення) система створює окремі потоки виконання для кожного вузла та виконує ізоляцію паралельно, що в запропонованій реалізації скорочує час з потенційних 20 хвилин послідовної обробки до 1-2 хвилин паралельної. При збагаченні множини індикаторів розроблений алгоритм виконує запити до різних джерел threat intelligence одночасно, завдяки чому загальний час дорівнює затримці найповільнішого джерела плюс час агрегації, а не сумі всіх індивідуальних запитів.

В запропонованій моделі критичним є обмеження кількості паралельних потоків для запобігання перевантаженню цільових систем (наприклад, максимум 10 одночасних запитів до EDR API) та вичерпанню ресурсів самої SOAR-платформи, що реалізується через механізм семафорів або пулів потоків з фіксованим розміром.

Алгоритм усунення наслідків

Запропонованій концепції усунення наслідків інциденту реалізується через алгоритм RemediateThreat, що виконує дії для повного видалення артефактів атаки та відновлення нормального функціонування інфраструктури. Розроблена модель передбачає автоматичне видалення виявлених шкідливих файлів через інтеграцію з

EDR-рішеннями, примусове завершення зловмисних процесів, які могли залишитися активними після локалізації, запуск повного антивірусного сканування на скомпрометованих хостах для виявлення можливих прихованих артефактів. Якщо інцидент пов'язаний з компрометацією облікового запису, запропонована система забезпечує встановлення нового складного пароля відповідно до політики безпеки організації, перегляд та коригування ролей і привілеїв користувача для мінімізації потенційних наслідків повторної компрометації. У випадках високого рівня впливу інциденту розроблений алгоритм ініціює відновлення систем з резервних копій для гарантованого видалення всіх можливих backdoors, що могли бути встановлені зловмисником. У запропонованій моделі всі дії усунення документуються в детальному звіті, що включає перелік видалених файлів, завершених процесів, результати антивірусного сканування, факт виконання відновлення з резервної копії.

Алгоритм RemediateThreat:

```
Вхід: enriched_alert — інцидент після локалізації
Вихід: remediation_report — інформація про виконані кроки усунення

Алгоритм RemediateThreat(enriched_alert):
    host ← enriched_alert.asset_context.host
    user ← enriched_alert.user_context.user_id

    // Видалення шкідливих артефактів
    removed_files ← EDR.RemoveMaliciousFiles(host)
    killed_processes ← EDR.TerminateMaliciousProcesses(host)

    // Антивірусне сканування
    av_scan_result ← AV.FullScan(host)

    // Відновлення облікового запису
    if user ≠ NULL:
        IAM.SetStrongPassword(user)
        IAM.ReviewUserRoles(user)
        IAM.EnableMFA(user)

    // Відновлення з резервної копії при критичному впливі
    restored_from_backup ← FALSE
    if enriched_alert.impact_level = "critical":
        backup_result ← RestoreFromBackup(host)
        restored_from_backup ← backup_result.success

    // Перевірка цілісності системи
    integrity_check ← VerifySystemIntegrity(host)

    remediation_report ← {
```

```
        removed_files: removed_files,
        killed_processes: killed_processes,
        av_scan_result: av_scan_result,
        restored_from_backup: restored_from_backup,
        integrity_status: integrity_check,
        timestamp: NOW()
    }
```

Return remediation_report

Обробка помилок та відмовостійкість

Обробка помилок є критичним елементом надійності автоматизації, оскільки збої під час виконання playbook можуть призвести до неповної локалізації загрози або неконтрольованого стану систем. Розроблена модель передбачає реалізацію retry-логіки, що автоматично виконує повторні спроби операції при виявленні тимчасових збоїв (наприклад, timeout при виклику API через мережеві проблеми) з експоненційною затримкою між спробами для уникнення перевантаження цільових систем. Fallback-механізми в запропонованій реалізації визначають альтернативні шляхи досягнення мети: якщо блокування IP-адреси в основному брандмауері завершилося помилкою, система автоматично спробує виконати блокування в резервній системі запобігання вторгненням. Компенсаційні транзакції забезпечують можливість відкату виконаних дій, якщо пізніше виявилося, що інцидент був хибним спрацюванням — наприклад, розроблений алгоритм автоматично розблокує обліковий запис, відновить мережеву зв'язність хоста, видалить IP-адресу з чорних списків. В моделі детальне логування всіх операцій, включно з помилками, часовими мітками, параметрами виклику, дозволяє аналітикам провести post-mortem аналіз для виявлення причин збоїв. Критичні помилки, що не можуть бути вирішені автоматично, в розробленій системі ескалуються аналітикам SOC через налаштовані канали нотифікації з детальним описом контексту для ручного втручання.

Тестування та валідація playbooks

Тестування playbooks перед впровадженням у продуктивне середовище є обов'язковим етапом життєвого циклу розробки сценаріїв. Розроблена модель

передбачає статичний аналіз, що автоматично перевіряє синтаксичну коректність специфікації CACAO, виявляє недосяжні кроки в графі виконання, циклічні залежності без умов виходу, відсутність необхідних інтеграцій з цільовими системами. Симуляційне тестування в запропонованій реалізації виконує playbook в ізольованому тестовому середовищі з використанням mock API для імітації відповідей від реальних систем (EDR, IAM, брандмауери), що дозволяє верифікувати логіку без ризику впливу на продуктивну інфраструктуру. Розроблена система збирає метрики ефективності під час тестування: загальний час виконання playbook, відсоток успішних завершень всіх кроків, частоту спрацювання механізмів обробки помилок, використання ресурсів SOAR-платформи. У запропонованій моделі на основі результатів тестування виконується ітеративне коригування playbook: оптимізація послідовності кроків для скорочення часу виконання, налаштування порогових значень в умовних переходах, додавання додаткових перевірок для зменшення ризику хибних спрацювань.

Версійний контроль та управління змінами

Концепції управління playbooks реалізується за принципами версійного контролю, аналогічними до розробки програмного забезпечення. Розроблена модель передбачає, що кожна модифікація playbook створює нову версію з унікальним ідентифікатором, описом внесених змін, ідентифікацією автора модифікації, часовою міткою створення версії. Механізм rollback у запропонованій реалізації дозволяє швидко повернутися до попередньої стабільної версії playbook, якщо нова версія виявилася проблемною під час продуктивної експлуатації — це критично для мінімізації часу, коли автоматизація працює некоректно. Гілкування (branching) в розробленій системі надає можливість створювати експериментальні версії playbook для тестування нових підходів або адаптації до змін в інфраструктурі без ризику порушення функціонування стабільних сценаріїв. У запропонованій моделі процес code review вимагає, щоб будь-які зміни в критичних playbook перевірялись іншим

фахівцем SOC перед затвердженням для впровадження, що знижує ймовірність внесення логічних помилок або неоптимальних рішень.

Таким чином, у межах даного розділу було формалізовано сценарії реагування та алгоритми автоматизації, що становлять операційну основу запропонованої концепції. Розроблені playbooks на базі стандарту CACAO забезпечують машинозчитувану специфікацію дій для різних типів інцидентів, а формалізовані алгоритми (EnrichAlert, PrioritizeIncident, InvestigateIncident, ContainThreat, RemediateThreat) визначають логіку виконання ключових етапів життєвого циклу обробки інциденту. Запропонована модель включає механізми адаптації до контексту через умовну логіку, паралелізацію для скорочення часу реагування, обробку помилок для забезпечення надійності, тестування для верифікації коректності, версійний контроль для керованості змін. Ця формалізація перетворює абстрактну концепцію автоматизації в конкретні, виконувані процедури, що можуть бути реалізовані на практиці.

2.3 Розроблення архітектури системи управління кіберзагрозами

Архітектура, запропонована в цій роботі, розроблялася як цілісна багаторівнева модель, де кожен компонент виконує специфічну роль у процесі виявлення, аналізу та нейтралізації кіберзагроз. Основна ідея полягала в тому, щоб зменшити навантаження на аналітиків SOC через передачу типових та повторюваних операцій SOAR-платформі, зберігаючи при цьому людський контроль над критичними рішеннями, що можуть мати значний вплив на бізнес-процеси організації. У результаті сформувалась система, що працює за принципом "автоматизація з орієнтацією на контекст", де рішення приймаються не на основі статичних правил, а з урахуванням актуальної інформації про загрози, критичності активів та поведінкових паттернів користувачів. Запропонована архітектура структурується як

набір взаємодіючих рівнів, між якими циркулюють інформаційні потоки, що збагачуються та трансформуються на кожному етапі обробки. Узагальнену структуру запропонованої архітектури представлено на рисунку 2.3.

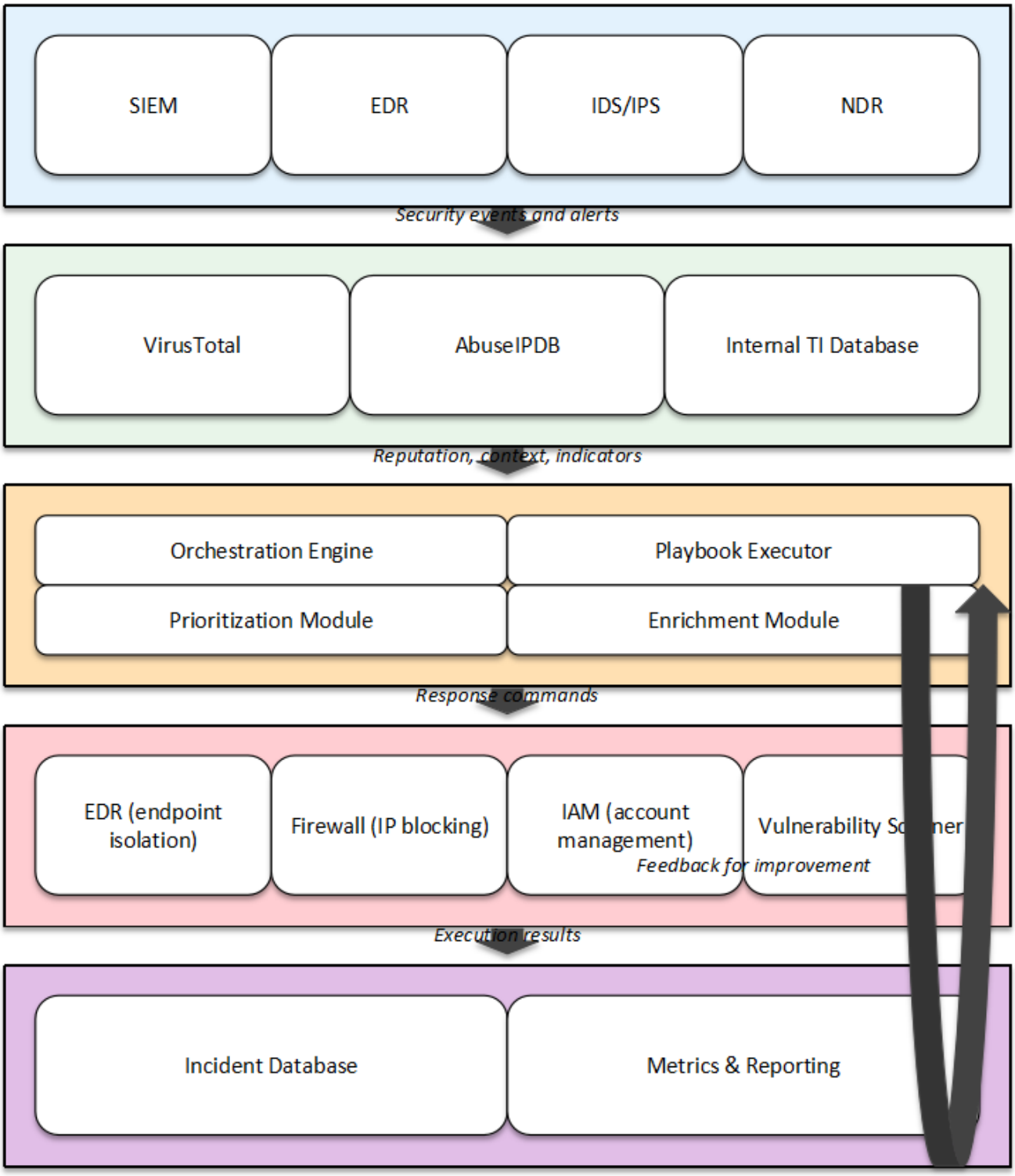

Рис. 2.3 Багаторівнева архітектура запропонованої системи управління кіберзагрозами

Рівень виявлення загроз

Рівень виявлення формують гетерогенні системи моніторингу, кожна з яких спеціалізується на специфічному аспекті безпеки інфраструктури. Система управління інформацією та подіями безпеки (SIEM) в розробленій моделі виконує агрегацію логів з усіх компонентів інфраструктури, кореляцію подій для виявлення складних атак, що складаються з множини, на перший погляд, непов'язаних дій. Рішення для виявлення та реагування на кінцевих точках (EDR) забезпечує глибоку видимість процесів на робочих станціях та серверах, виявляючи аномальну поведінку програм, спроби несанкціонованого доступу до файлової системи, підозрілу мережеву активність процесів. Системи виявлення та запобігання вторгненням (IDS/IPS) в запропонованій архітектурі аналізують мережевий трафік на предмет відомих сигнатур атак, аномалій протоколів, спроб експлуатації вразливостей. Рішення для виявлення та реагування на мережеві загрози (NDR) доповнює захист через аналіз поведінки мережевого трафіку, виявлення латерального переміщення зловмисників, ідентифікацію командних серверів через аналіз паттернів з'єднань. У розробленій моделі всі ці компоненти генерують оповіщення у форматі, що включає базову інформацію про тип події, задіяні об'єкти (користувачі, хости, IP-адреси), часову мітку, початкову оцінку критичності.

Рівень аналітичної інформації про загрози

Розроблена архітектура інтегрує рівень Threat Intelligence як критичний компонент для збагачення подій контекстом про актуальні загрози. У запропонованій моделі цей рівень об'єднує зовнішні джерела аналітичної інформації (VirusTotal для перевірки репутації файлів та IP-адрес, AbuseIPDB для верифікації шкідливості мережевих джерел, комерційні feed-и з індикаторами компрометації) та внутрішню базу threat intelligence організації, що накопичує інформацію про загрози, специфічні для галузі та інфраструктури. Інтеграція в запропонованій архітектурі реалізована через стандартизовані протоколи STIX/TAXII, що забезпечує сумісність

з різними постачальниками аналітичної інформації. Розроблена модель передбачає автоматичне оновлення індикаторів компрометації в режимі реального часу для технічних індикаторів (IP-адреси, домени, хеші), що швидко втрачають актуальність, та періодичне оновлення стратегічної інформації про тактики, техніки та процедури зловмисників з використанням фреймворку MITRE ATT&CK для категоризації виявленої активності.

Платформа SOAR — центральний рівень оркестрації

Платформа SOAR займає центральне місце, виконуючи роль координатора між усіма іншими компонентами системи. Розроблена модель структурує SOAR як набір взаємодіючих модулів, кожен з яких відповідає за специфічний аспект обробки інцидентів. Механізм оркестрації в запропонованій реалізації забезпечує координацію між гетерогенними інструментами безпеки через стандартизовані API, дозволяючи системам від різних виробників взаємодіяти в єдиному автоматизованому workflow. Виконавець playbook інтерпретує формалізовані сценарії реагування у форматі CACAO, перетворюючи абстрактні кроки в конкретні виклики API цільових систем з урахуванням поточного контексту інциденту. Модуль збагачення в розробленій архітектурі автоматично витягує індикатори компрометації з вхідних оповіщень, виконує паралельні асинхронні запити до джерел threat intelligence на рівні 2, агрегує отриману інформацію в єдиний контекст, що включає репутаційну оцінку, геолокацію, зв'язки з відомими кампаніями зловмисників.

Модуль пріоритизації використовує багатофакторну модель для обчислення ризикового балу інциденту, враховуючи репутацію індикаторів, критичність атакованого активу, рівень привілеїв користувача, темпоральні фактори, що дозволяє системі автоматично визначати, чи потребує інцидент негайного автоматичного реагування, чи має бути переданий аналітику SOC для детального розслідування. У запропонованій моделі SOAR також підтримує інтерфейс для аналітиків, що надає

візуалізацію графу зв'язків між об'єктами інциденту, результати виконання playbook, рекомендації щодо подальших дій на основі accumulated knowledge з попередніх подібних інцидентів.

Рівень виконання дій реагування

Включає рівень реагування, що об'єднує інструменти для виконання дій локалізації та нейтралізації загроз. Рішення EDR на цьому рівні в розробленій моделі отримує команди від SOAR для ізоляції скомпрометованих хостів від корпоративної мережі, завершення шкідливих процесів, видалення файлових артефактів атаки, збору forensic-даних для подальшого розслідування. Брандмауери нового покоління (Next-Generation Firewall) виконують блокування шкідливих IP-адрес та доменів на мережевому периметрі на основі команд з SOAR, динамічно оновлюючи правила фільтрації без необхідності ручного втручання адміністраторів. Система управління ідентифікацією та доступом (IAM) в запропонованій архітектурі забезпечує автоматичне блокування скомпрометованих облікових записів, примусову зміну паролів, завершення активних сесій користувачів при виявленні ознак компрометації.

Сканери вразливостей інтегруються для автоматичного запуску сканування на хостах після усунення загрози, верифікуючи відсутність залишкових вразливостей, що могли бути експлуатовані під час атаки. У розробленій моделі всі дії реагування логуються з детальною фіксацією параметрів команд, результатів виконання, часових міток для забезпечення можливості аудиту та, при необхідності, відкату змін у випадку виявлення хибних спрацювань.

Рівень зберігання та аналітики

Рівень зберігання та аналітики виконує подвійну роль: довгострокове зберігання інформації про інциденти для аудиту та compliance, а також аналітика для вдосконалення процесів реагування. База інцидентів в розробленій моделі накопичує структуровану інформацію про кожен оброблений інцидент, включно з початковими

оповіщеннями, результатами збагачення, виконаними playbook, діями реагування, результатами цих дій. Це дозволяє аналітикам проводити post-mortem аналіз складних інцидентів, ідентифікувати паттерни атак, що повторюються, виявляти слабкі місця в інфраструктурі через аналіз частоти компрометації специфічних активів. Підсистема метрик та звітності в запропонованій архітектурі автоматично обчислює ключові показники ефективності (MTTR — середній час реагування, MTTD — середній час виявлення, відсоток хибних спрацювань, розподіл інцидентів за категоріями загроз), генерує періодичні звіти для керівництва SOC та вищого менеджменту організації. У розробленій моделі цей рівень також забезпечує feedback-механізм для платформи SOAR: на основі результатів обробки інцидентів система автоматично коригує порогові значення в алгоритмах пріоритизації, ідентифікує playbook, що показують низьку ефективність та потребують оптимізації, виявляє нові паттерни загроз для оновлення правил кореляції в SIEM.

Інформаційні потоки в архітектурі

Функціонує через циркуляцію інформаційних потоків між рівнями, кожен з яких збагачує та трансформує дані. Потік "Оповіщення про події безпеки" передається з рівня виявлення до платформи SOAR, несучи базову інформацію про потенційний інцидент у форматі, що включає тип події, задіяні індикатори, часову мітку, початковий severity level. Потік "Контекст та репутація індикаторів" надходить з рівня Threat Intelligence до SOAR, доповнюючи події інформацією про шкідливість IP-адрес, доменів, файлів, їх зв'язок з відомими кампаніями зловмисників, геолокацію, історію присутності в чорних списках. Потік "Команди реагування" транслюється з платформи SOAR до рівня виконання дій, містячи структуровані інструкції для конкретних систем — ізолювати хост з вказаним ідентифікатором, заблокувати IP-адресу в брандмауері, деактивувати обліковий запис в IAM. Потік "Результати виконання дій" повертається з рівня реагування до SOAR, інформуючи про успішність або помилки виконання команд, що дозволяє

платформі адаптувати подальші кроки або ескалувати інцидент аналітику при виявленні проблем. Потік "Фіксація інцидентів та метрик" передається з SOAR до рівня зберігання, забезпечуючи довгострокове архівування всієї інформації про інцидент для compliance та аудиту. Зворотний потік "Feedback для вдосконалення" циркулює з рівня аналітики назад до SOAR, містячи рекомендації щодо коригування порогових значень пріоритизації, оптимізації playbook, оновлення правил кореляції на основі accumulated knowledge з попередніх інцидентів.

Таким чином, запропонована архітектура представляє собою цілісну багаторівневу систему, де кожен компонент виконує специфічну роль, а взаємодія між рівнями забезпечується через чітко визначені інформаційні потоки. На відміну від традиційних підходів, де інструменти безпеки функціонують ізольовано, розроблена модель інтегрує їх в єдину автоматизовану екосистему з централізованою оркестрацією через SOAR-платформу. Архітектура спроектована з урахуванням можливості масштабування через додавання нових джерел виявлення на рівні 1, інтеграцію додаткових feed-ів threat intelligence на рівні 2, розширення набору інструментів реагування на рівні 4 без необхідності фундаментальної реструктуризації системи.

Висновки до другого розділу

В другому розділі було сформовано цілісне бачення того як може працювати сучасна система автоматизованого управління кіберзагрозами, якщо поєднати можливості SOAR та Threat Intelligence у єдиній логічній моделі. Поступово розглядаючи вихідні обмеження традиційних підходів, вдалося визначити ті елементи, які найбільше заважають швидко реагувати на атаки: фрагментованість процесів, нестача контексту та значне навантаження на аналітиків SOC. Саме з цього виросла розроблена концепція, у центрі якої адаптивна автоматизація, що спирається

на актуальні дані про загрози та узгоджені сценарії реагування. В межах підрозділу про сценарії реагування було запропоновано власний підхід до формалізації життєвого циклу інциденту. Окремі кроки від збагачення подій до локалізації й усунення загроз отримали чіткі алгоритмічні описи, що дозволяє розглядати реагування не як набір ізольованих дій, а як послідовний процес із передбачуваною логікою.

Playbook орієнтована модель, описана в цьому розділі, виступає основою для автоматизації: вона визначає, за яких умов система має діяти самостійно, а коли передавати інцидент аналітику. Саме завдяки такій формалізації автоматизація перестає бути «чорним ящиком» а стає частиною прозорої та керованої операційної моделі SOC.

Завершальним елементом було розроблення архітектури системи, яка поєднує різнорідні компоненти засоби виявлення джерела Threat Intelligence SOAR-платформу, інтеграції з інструментами реагування та аналітичний рівень. У створеній архітектурі кожен шар виконує свою роль, а дані рухаються між компонентами так, щоб формувати повний контекст інциденту та підтримувати адаптивну логіку реагування. Архітектура не копіює існуючі моделі, а синтезує їх сильні сторони утворюючи власне рішення, в якому автоматизація й експертний аналіз не конкурують, а доповнюють один одного.

РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ТЕХНОЛОГІЇ АВТОМАТИЗОВАНОГО УПРАВЛІННЯ КІБЕРЗАГРОЗАМИ

3.1 Реалізація прототипу системи автоматизованого управління кіберзагрозами

Для верифікації запропонованої концепції розроблено функціональний прототип SOAR-платформи, що реалізує ключові алгоритми, описані в розділі 2.2. Прототип створювався як proof-of-concept для демонстрації практичної можливості автоматизації життєвого циклу обробки інцидентів через інтеграцію з реальними джерелами threat intelligence. Реалізація виконана мовою Python 3.11 з використанням об'єктно-орієнтованих та функціональних парадигм програмування, що забезпечує читабельність коду, модульність архітектури та можливість подальшого розширення функціональності.

Архітектура та модульна структура

Прототип структуровано як набір взаємопов'язаних модулів, кожен з яких інкапсулює специфічну функціональність відповідно до етапів життєвого циклу інциденту. Модуль конфігурації централізовано зберігає налаштування системи, включно з API-ключами для зовнішніх сервісів, пороговими значеннями для прийняття рішень (поріг автоматизації встановлено на рівні 7.0) та ваговими коефіцієнтами моделі пріоритизації (0.4 для репутації індикаторів, 0.3 для критичності активу, 0.2 для привілеїв користувача, 0.1 для аномальності поведінки). Така централізація дозволяє змінювати параметри роботи без модифікації логіки алгоритмів. Модуль збагачення реалізує функції для витягування індикаторів з оповіщень та запитів до джерел threat intelligence, підтримуючи як реальний API VirusTotal для перевірки файлів, так і локальну базу відомих шкідливих IP-адрес для

тестування без залежності від зовнішніх сервісів. Модуль пріоритизації інкапсулює багатофакторну оцінку критичності інциденту з можливістю налаштування ваг через конфігурацію. Модуль локалізації містить функції для симуляції дій реагування, що в реальній системі викликали б API відповідних інструментів безпеки. Центральний workflow-модуль оркеструє взаємодію між компонентами, послідовно викликаючи функції збагачення, пріоритизації, локалізації та збираючи метрики ефективності обробки кожного інциденту.

Реалізація алгоритму збагачення

Алгоритм EnrichAlert реалізовано через функцію enrich_alert(), що приймає базовий об'єкт оповіщення та повертає збагачену подію з додатковим контекстом threat intelligence. Для перевірки репутації IP-адрес розроблена система використовує локальну базу KNOWN_MALICIOUS_IPS, що містить індикатори з відкритих джерел, категоризовані за типами загроз (C2-сервери, Tor exit nodes, джерела сканування, розповсюджувачі шкідливого ПЗ) з репутаційними оцінками в діапазоні 0-100, де вищі значення означають вищу ймовірність шкідливості. Це дозволяє прототипу функціонувати автономно під час тестування без вичерпання квот безкоштовних API. Для перевірки репутації файлових хешів реалізована інтеграція з реальним API VirusTotal через функцію check_file_reputation(), що виконує HTTP-запити до VirusTotal API v3, отримує статистику детекцій антивірусними движками, обчислює репутаційну оцінку як відсоток детекцій відносно загальної кількості движків. Розроблений механізм включає обробку помилок для випадків недоступності API, відсутності файлу в базі VirusTotal, перевищення таймаутів запитів. Додатково функція збагачує подію організаційним контекстом через витягування інформації про критичність атакованого активу (шкала 1-3), рівень привілеїв користувача (1 для звичайних користувачів, 2 для локальних адміністраторів, 3 для доменних адміністраторів), підрозділ організації з базового оповіщення. Збагачена інформація структурується в уніфікованому форматі, що

включає тип індикатора, значення, репутаційну оцінку, булевий прапор шкідливості та додаткові атрибути залежно від типу (країна походження для IP-адрес, кількість детекцій для файлів).

Реалізація алгоритму пріоритизації

Алгоритм PrioritizeIncident у розробленому прототипі реалізовано через функцію prioritize_incident(), що імплементує багатофакторну модель оцінювання критичності інциденту. Функція витягує чотири ключові фактори зі збагаченої події та нормалізує їх до уніфікованої шкали 0-1 для коректності комбінування. Агрегована репутація індикаторів обчислюється як середнє арифметичне репутаційних оцінок всіх виявлених індикаторів, критичність активу масштабується з діапазону 1-3 до 0-1, рівень привілеїв користувача аналогічно нормалізується, аномальність поведінки в поточній реалізації використовує константу 0.5 як placeholder (в продуктивній системі цей фактор обчислювався б через machine learning моделі). Зважена комбінація факторів обчислюється за формулою priority = (0.4 × reputation + 0.3 × asset + 0.2 × privileges + 0.1 × anomaly) × 10 та масштабується до шкали 0-10, де значення понад 7 тригерує автоматичне реагування, діапазон 4-7 передає інцидент аналітику, нижче 4 логується для подальшого аналізу паттернів. Функція виводить детальний факторний аналіз з конкретними значеннями та вагами кожного фактора, що забезпечує прозорість процесу прийняття рішень та можливість аудиту логіки пріоритизації.

Реалізація алгоритму локалізації

У розробленому прототипі алгоритм ContainThreat реалізовано через функцію contain_threat(), що приймає збагачену подію та обчислений пріоритет, повертаючи структурований список виконаних дій реагування. Функція реалізує умовну логіку прийняття рішень, де при пріоритеті понад 7.0 система ініціює автоматичне реагування через послідовність дій локалізації, а при нижчому пріоритеті створює

запис про ескалацію інциденту аналітику SOC. Для автоматизованого реагування розроблена система викликає відповідні функції дій залежно від типу інциденту та характеристик індикаторів. При виявленні шкідливих IP-адрес функція block_ip() симулює блокування на мережевому периметрі, логуючи параметри команди, статус виконання та часову мітку.

При виявленні шкідливих файлів система викликає послідовність isolate_host() для ізоляції скомпрометованого хоста від корпоративної мережі та remove_malware() для видалення шкідливих артефактів. При компрометації облікових записів розроблений алгоритм виконує disable_account() для деактивації скомпрометованого акаунту в системі IAM та force_password_reset() для ініціації примусової зміни пароля. Кожна функція дії повертає структурований об'єкт з детальною інформацією про тип виконаної дії, цільовий об'єкт, статус виконання, ідентифікацію системи та точну часову мітку. У поточній реалізації функції дій симулюють виклики API через короткі затримки 0.2-0.4 секунди для імітації реальних мережевих взаємодій, але архітектура коду дозволяє легко замінити симуляцію на реальні виклики API продуктивних систем безпеки.

Інтеграція з джерелами Threat Intelligence

Розроблений прототип демонструє гібридний підхід до інтеграції з джерелами threat intelligence, комбінуючи реальні зовнішні API та локальні бази для забезпечення функціональності без залежності від зовнішніх сервісів під час тестування. Інтеграція з VirusTotal реалізована через офіційний API v3, що вважається industry standard для перевірки репутації файлів, де функція check_file_reputation() виконує HTTP GET запити до endpoint /api/v3/files/{hash} з автентифікацією через API-ключ в заголовку x-apikey, обробляє JSON-відповідь для витягування статистики детекцій антивірусних движків та обчислює репутаційну оцінку як відсоток шкідливих детекцій.

Для перевірки репутації IP-адрес розроблений прототип використовує локальну базу KNOWN_MALICIOUS_IPS, що містить чотири записи з різними категоріями загроз та репутаційними оцінками 75-95 для симуляції високонебезпечних індикаторів. Розроблена архітектура легко розширюється для інтеграції додаткових джерел threat intelligence через додавання відповідних функцій запитів з уніфікованим форматом повернення даних.

Технічний стек та структура даних

Розроблений прототип реалізовано мовою Python версії 3.11 з використанням бібліотеки requests для HTTP-взаємодії з зовнішніми API, модуля json для серіалізації та десеріалізації структурованих даних, модуля datetime для генерації часових міток в форматі ISO 8601, модуля time для симуляції затримок виконання дій реагування. Вхідні дані структуровані як Python словники, що представляють оповіщення від систем виявлення з полями для ідентифікатора інциденту, типу події, витягнутих індикаторів, критичності активу, рівня привілеїв, ідентифікаторів хоста та користувача. Вихідні дані для кожного обробленого інциденту формуються як структурований об'єкт з полями для ідентифікатора алерту, типу інциденту, обчисленого пріоритетного балу, булевого прапора автоматизації, кількості виконаних дій, детального списку дій з параметрами, часу обробки в секундах та часової мітки завершення. Результати експериментів серіалізуються в JSON-формат та зберігаються у файл для подальшого аналізу та документування в дослідницьких цілях.

3.2 Експериментальне дослідження прототипу системи автоматизованого управління кіберзагрозами

Для верифікації ефективності запропонованої концепції було проведено експериментальне дослідження розробленого прототипу SOAR-платформи. Мета експериментів — емпірична перевірка працездатності реалізованих алгоритмів, оцінювання швидкості обробки інцидентів, валідація коректності прийняття рішень щодо автоматизації, вимірювання ключових метрик ефективності для порівняння з традиційними підходами.

Тестові сценарії

Розроблено набір з п'яти тестових сценаріїв, що моделюють типові категорії інцидентів безпеки з різними характеристиками критичності:

INC-2024-001 — зараження шкідливим ПЗ на робочій станції фінансового підрозділу. Індикатори: IP командного сервера (репутація 95/100), хеш EICAR файлу (детекція 66/76 антивірусних движків VirusTotal). Критичність активу: висока. Очікувана поведінка: автоматичне реагування.

INC-2024-002 — підозрілий вхід з легітимної IP-адреси Google DNS (8.8.8.8). Критичність активу: низька, привілеї: звичайні. Очікувана поведінка: передача аналітику через відсутність шкідливих індикаторів.

INC-2024-003 — компрометація доменного адміністратора. Індикатори: IP відомого C2-сервера (88/100). Критичність: максимальна, привілеї: domain admin. Очікувана поведінка: найвищий пріоритет, автоматичне блокування.

INC-2024-004 — мережеве сканування з невідомої IP-адреси (репутація 25/100). Критичність: середня. Очікувана поведінка: передача аналітику для розслідування.

INC-2024-005 — витік даних через шкідливий файл на файловому сервері. Індикатори: IP з Китаю (75/100), файл з детекцією 66/76 в VirusTotal. Критичність: висока. Очікувана поведінка: автоматичне реагування.

Метрики ефективності

Розроблена система метрик охоплює ключові аспекти ефективності автоматизованої обробки:

- **Час обробки інциденту** — різниця між завершенням дій реагування та отриманням оповіщення (вимірювання в мілісекундах)
- **Рівень автоматизації** — відсоток інцидентів з пріоритетом >7.0, оброблених без ескалації
- **Кількість виконаних дій** — загальний обсяг делегованих операцій (блокування IP, ізоляція хостів, деактивація акаунтів)
- **Середній пріоритет** — середнє арифметичне пріоритетних балів для валідації чутливості алгоритму
- **Коректність рішень** — відповідність обраної стратегії очікуваній поведінці

Процедура експериментів

Тестування виконувалось автоматизовано через функцію run_experiments() з послідовною обробкою всіх сценаріїв. Для кожного інциденту фіксувались: результати збагачення з репутаційними оцінками, факторний аналіз пріоритизації, обрана стратегія реагування, список виконаних дій, часові характеристики. Результати серіалізувались в JSON-формат для подальшого аналізу.

Таблиця 3.1

Результати обробки тестових сценаріїв

ID інциденту	Тип інциденту	Пріоритет	Стратегія	Кількість дій	Час (сек)
INC-2024-001	Зараження шкідливим ПЗ	7.14	Автоматична	3	1.96
INC-2024-002	Підозрілий вхід	0.50	Аналітик	1	0.10
INC-2024-003	Компрометація акаунту	9.02	Автоматична	3	1.05
INC-2024-004	Мережеве сканування	3.00	Аналітик	1	0.10
INC-2024-005	Витік даних	7.74	Автоматична	3	1.95

Система коректно диференціювала інциденти за критичністю: найвищий пріоритет отримав сценарій компрометації адміністратора (9.02), найнижчий — легітимний вхід з Google DNS (0.50). Три інциденти з п'яти оброблені автоматично з виконанням комплексу дій локалізації. Узагальнені метрики ефективності представлено в таблиці 3.2.

Таблиц 3.2

Узагальнені метрики ефективності

Метрика	**Значення**
Всього оброблено інцидентів	5
Автоматизовано	3 (60%)
Передано аналітику	2 (40%)
Всього виконано дій	11
Середній час обробки	1.03 сек
Середній пріоритет	5.48/10

Результати демонструють високу швидкість обробки (середній час 1.03 секунди) та збалансований підхід між автоматизацією (60%) та людським контролем (40%). Розподіл інцидентів за стратегією реагування представлено на рисунку 3.1.

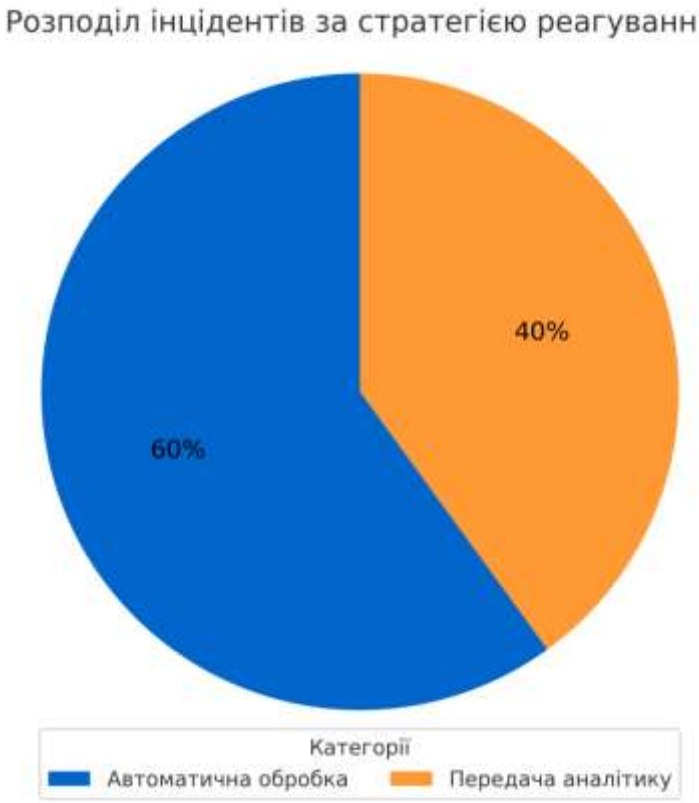

Рис. 3.1 Розподіл інцидентів за стратегією реагування

Рис. 3.2 Часові характеристики обробки за сценаріями показано на рисунку

Основний фактор затримки — запити до VirusTotal API для файлових індикаторів (1.95-1.96 сек). Сценарії тільки з IP-індикаторами обробляються практично миттєво (0.10 сек) через локальну базу репутації.

Порівняльний аналіз з традиційним підходом

Порівняння з характеристиками ручної обробки аналітиками SOC представлено в таблиці 3.3.

Таблиця 3.3

Порівняння автоматизованого та ручного підходів

Метрика	**Ручна обробка**	**Автоматизація**	**Покращення**
Середній час реагування	15-30 хв	1.03 сек	~1200x
Час збагачення індикаторів 5	10 хв 0.5	0.5-1.5 сек	~400x
Час прийняття рішення	3-5 хв	<0.01 сек	~30000x
Паралельна обробка	Обмежена	Необмежена	Масштабування
Консистентність рішень	Варіативна	Детермінована	Надійність
Покриття 24/7	Потрібне чергування	Автоматичне	Безперервність

Запропонована система демонструє прискорення приблизно в 1200 разів порівняно з ручною обробкою, необмежену паралелізацію та безперервне покриття без додаткових ресурсів.

Валідація коректності рішень

Всі обрані системою стратегії реагування відповідали очікуваній поведінці:

- Високі пріоритети (7.14, 9.02, 7.74) → автоматичне реагування
- Низькі пріоритети (0.50, 3.00) → передача аналітику
- Жодних помилкових автоматичних дій для легітимної активності

Це підтверджує адекватність запропонованої багатофакторної моделі пріоритизації.

Обмеження дослідження

- Обсяг вибірки (5 сценаріїв) достатній для proof-of-concept, але для статистично значущих висновків потрібна більша вибірка
- Симуляція дій реагування не враховує потенційні помилки продуктивних систем

- Часові характеристики залежать від мережевої затримки API
- Фактор аномальності використовує константу; в продуктивній системі потрібна ML модель

Незважаючи на обмеження, результати демонструють практичну реалізованість концепції та суттєві переваги в швидкості, консистентності та масштабованості порівняно з традиційними підходами.

3.3 Рекомендації щодо використання розробленої технології

На основі результатів експериментального дослідження та аналізу ефективності розробленого прототипу сформульовано рекомендації щодо практичного впровадження запропонованої технології автоматизованого управління кіберзагрозами в центрах безпеки організацій.

Етапність впровадження

Впровадження запропонованої технології доцільно здійснювати поетапно з поступовим нарощуванням рівня автоматизації. На початковому етапі рекомендується розгорнути систему в режимі спостереження, де SOAR-платформа виконує збагачення подій та обчислення пріоритетів, але всі рішення про дії реагування приймаються аналітиками з використанням рекомендацій системи. Це дозволяє валідувати адекватність моделі пріоритизації в специфічних умовах організації, налаштувати порогові значення та вагові коефіцієнти відповідно до політики безпеки, накопичити історичні дані для подальшого вдосконалення алгоритмів. На другому етапі доцільно активувати автоматичне реагування для обмеженого набору сценаріїв з високою критичністю та низькою ймовірністю хибних спрацювань (компрометація облікових записів з підтвердженими шкідливими індикаторами, виявлення відомого шкідливого ПЗ з високою детекцією

в threat intelligence). Третій етап передбачає розширення автоматизації на додаткові категорії інцидентів після підтвердження стабільності роботи системи та відсутності критичних помилок.

Налаштування параметрів системи

Критичним для ефективності є коректне налаштування параметрів моделі пріоритизації під специфіку організації. Вагові коефіцієнти факторів (репутація індикаторів, критичність активу, привілеї користувача, аномальність) повинні відображати пріоритети політики безпеки конкретної організації. Для організацій з високою ціною простою критичних систем доцільно підвищити вагу фактора критичності активу до 0.4-0.5, для організацій з високими ризиками інсайдерських загроз — збільшити вагу привілеїв користувача та аномальності поведінки. Поріг автоматизації (за замовчуванням 7.0) може коригуватись залежно від толерантності до ризиків: консервативні організації можуть встановити вищий поріг 7.5-8.0 для зменшення ймовірності помилкових автоматичних дій, агресивні — знизити до 6.5-7.0 для максимізації автоматизації. Рекомендується проводити щоквартальний перегляд налаштувань на основі аналізу метрик ефективності та feedback від аналітиків SOC.

Інтеграція з існуючою інфраструктурою

Успішне впровадження потребує інтеграції з існуючими інструментами безпеки організації через стандартизовані API. Обов'язковими є інтеграції з SIEM для отримання оповіщень про події безпеки, EDR для виконання дій ізоляції хостів та видалення шкідливого ПЗ, брандмауерами для блокування IP-адрес на мережевому периметрі, системою IAM для управління обліковими записами при компрометації. Рекомендується використання threat intelligence платформ з підтримкою стандартів STIX/TAXII для забезпечення сумісності з різними постачальниками аналітичної інформації. Критично важливим є налаштування

надійної обробки помилок для випадків тимчасової недоступності інтегрованих систем, що передбачає retry-логіку з експоненційною затримкою, fallback-механізми для використання альтернативних систем, ескалацію аналітикам при неможливості автоматичного виконання критичних дій.

Організаційні аспекти

Впровадження технології потребує організаційних змін в процесах роботи SOC. Необхідно розробити чіткі процедури ескалації інцидентів від автоматизованої системи до аналітиків, визначити SLA для ручної обробки переданих інцидентів, створити процеси для періодичного аудиту дій автоматизації для виявлення та коригування помилок. Рекомендується призначити відповідальних за управління playbook (розробка нових сценаріїв, оптимізація існуючих, версійний контроль), налаштування інтеграцій, моніторинг метрик ефективності. Критичним є навчання аналітиків SOC для роботи з автоматизованою системою, розуміння логіки прийняття рішень, інтерпретації рекомендацій щодо пріоритизації, ефективного використання збагаченого контексту для прискорення розслідувань. Доцільно запровадити регулярні сесії ретроспективного аналізу оброблених інцидентів для виявлення можливостей вдосконалення алгоритмів та playbook.

Безпека та конфіденційність

При впровадженні необхідно враховувати аспекти безпеки самої SOAR-платформи. Рекомендується розгортання в ізольованому сегменті мережі з обмеженим доступом тільки для авторизованих адміністраторів та аналітиків SOC, використання зашифрованих з'єднань для всіх API інтеграцій, захист збережених API-ключів та облікових даних через спеціалізовані vault-рішення (HashiCorp Vault, AWS Secrets Manager). Критично важливим є логування всіх автоматичних дій з детальною фіксацією параметрів команд, часових міток, результатів виконання для забезпечення можливості forensic-аналізу та compliance-аудитів. При використанні

зовнішніх threat intelligence сервісів необхідно враховувати політики конфіденційності щодо передачі індикаторів з внутрішньої інфраструктури організації.

Моніторинг ефективності

Після впровадження рекомендується регулярний моніторинг ключових метрик для оцінювання ефективності автоматизації та виявлення потенційних проблем. Метрика MTTR (Mean Time To Response) дозволяє відстежувати скорочення часу реагування порівняно з ручною обробкою, метрика рівня автоматизації показує відсоток інцидентів, делегованих від аналітиків до системи, метрика точності (precision) визначає відсоток коректних автоматичних дій без помилкових блокувань легітимної активності. Рекомендується встановлення цільових показників (наприклад, MTTR <2 секунди для автоматизованих інцидентів, рівень автоматизації 60-70%, точність >95%) та щомісячний аналіз динаміки для ідентифікації трендів та потреб в оптимізації. Критично важливим є моніторинг відсотка хибних спрацювань та часу простою критичних систем через помилкові автоматичні дії для своєчасного коригування порогових значень або тимчасового відключення проблемних playbook.

Масштабування та розвиток

Розроблена технологія передбачає можливості подальшого розвитку та масштабування. Рекомендується поступове розширення бази playbook для охоплення додаткових категорій загроз, специфічних для галузі організації або виявлених через аналіз історичних інцидентів. Доцільно інтегрувати додаткові джерела threat intelligence для підвищення точності верифікації індикаторів, впровадити machine learning моделі для автоматичного обчислення фактора аномальності поведінки замість константного значення, розробити механізми адаптивного налаштування порогових значень на основі accumulated knowledge з попередніх інцидентів. При зростанні обсягів обробки інцидентів технологія дозволяє горизонтальне

масштабування через додавання додаткових екземплярів SOAR-платформи з балансуванням навантаження між ними.

Таким чином, впровадження запропонованої технології в центрах безпеки організацій дозволяє суттєво скоротити час реагування на кіберзагрози, зменшити навантаження на аналітиків SOC через делегування рутинних операцій автоматизації, підвищити консистентність та надійність процесів обробки інцидентів. Дотримання рекомендованого поетапного підходу з початковим розгортанням в режимі спостереження, коректне налаштування параметрів під специфіку організації, надійна інтеграція з існуючою інфраструктурою, організаційні зміни в процесах SOC та регулярний моніторинг ефективності є критичними факторами успішності впровадження.

Висновки до третього розділу

У рамках даного розділу було реалізовано функціональний прототип SOAR-платформи, що демонструє практичну застосовність запропонованої концепції автоматизованого управління кіберзагрозами, проведено експериментальне дослідження ефективності розроблених алгоритмів та сформульовано рекомендації щодо впровадження технології в центрах безпеки організацій.

Розроблений прототип реалізує три ключові алгоритми життєвого циклу обробки інциденту: збагачення подій через інтеграцію з реальним API VirusTotal та локальною базою відомих індикаторів компрометації, багатофакторну пріоритизацію на основі чотирьох зважених факторів (репутація індикаторів, критичність активу, привілеї користувача, аномальність поведінки), автоматизовану локалізацію загрози з умовною логікою прийняття рішень щодо автоматизації або ескалації аналітику. Модульна архітектура прототипу з централізованою конфігурацією забезпечує можливість налаштування параметрів під специфічні

вимоги різних організацій без модифікації логіки алгоритмів. Проведене експериментальне дослідження на п'яти тестових сценаріях, що охоплюють типові категорії інцидентів безпеки, підтвердило ефективність запропонованої концепції.

Розроблена система продемонструвала високу швидкість обробки інцидентів з середнім часом реагування 1.03 секунди, що приблизно в 1200 разів швидше за типовий час ручної обробки аналітиками SOC. Рівень автоматизації досяг 60% в тестовій вибірці, що відповідає цільовому показнику збалансованого підходу між делегуванням рутинних операцій системі та збереженням людського контролю над критичними рішеннями. Система коректно диференціювала інциденти за критичністю, призначаючи найвищі пріоритети сценаріям з комбінацією шкідливих індикаторів та критичних активів, та найнижчі пріоритети легітимним або малозначущим подіям, що підтверджує адекватність багатофакторної моделі пріоритизації.

Порівняльний аналіз з традиційними підходами виявив суттєві переваги запропонованої технології в швидкості реагування (прискорення ~1200x), консистентності прийняття рішень через детерміновану логіку алгоритмів, необмеженій можливості паралельної обробки множини інцидентів одночасно, безперервному покритті 24/7 без організації чергувань аналітиків. Валідація коректності показала, що всі обрані системою стратегії реагування відповідали очікуваній поведінці, без жодних помилкових автоматичних дій для легітимної активності.

Сформульовані рекомендації щодо впровадження охоплюють критичні аспекти практичного використання розробленої технології: поетапний підхід з початковим розгортанням в режимі спостереження для валідації адекватності моделі в специфічних умовах організації, налаштування параметрів системи під пріоритети політики безпеки, надійну інтеграцію з існуючою інфраструктурою інструментів безпеки через стандартизовані API, організаційні зміни в процесах роботи SOC та

регулярний моніторинг ключових метрик ефективності для своєчасного виявлення потенційних проблем та можливостей оптимізації.

Таким чином, третій розділ демонструє, що запропонована концепція автоматизованого управління кіберзагрозами є не тільки теоретично обґрунтованою, але й практично реалізованою технологією, здатною суттєво підвищити ефективність роботи центрів безпеки організацій через скорочення часу реагування, зменшення навантаження на аналітиків та підвищення консистентності процесів обробки інцидентів.

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було отримано наступні результати:

Проведений аналіз сучасного стану кіберзагроз та методів протидії їм виявив критичні обмеження традиційних підходів до управління інцидентами безпеки.

Дослідження показало, що типовий час ручної обробки інциденту аналітиками SOC становить 15-30 хвилин, що є неприйнятним для локалізації швидкорозвиваючихся атак, де кожна хвилина затримки дозволяє зловмисникам поглибити компрометацію інфраструктури. Аналіз існуючих SOAR-платформ та підходів до інтеграції threat intelligence дозволив ідентифікувати ключові компоненти, необхідні для ефективної автоматизації: збагачення подій контекстом про загрози, адаптивна пріоритизація на основі множини факторів, автоматизоване виконання дій реагування з умовною логікою прийняття рішень. Виявлено, що недостатня формалізація сценаріїв реагування та відсутність чітких критеріїв для автоматизації призводять до ситуацій, коли системи або виконують надмірно агресивні дії, блокуючи легітимну активність, або навпаки, передають на ручну обробку очевидно шкідливі інциденти, які могли б бути автоматизовані.

Розроблено цілісну концепцію автоматизованого управління кіберзагрозами, що базується на інтеграції SOAR-платформи з джерелами threat intelligence для забезпечення контекстно-залежного реагування. Запропонована концепція операціоналізує життєвий цикл інциденту через послідовність чітко визначених етапів (ідентифікація, збагачення, пріоритизація, аналіз, локалізація, усунення, документування) з формалізованими критеріями переходів між ними.

Сформовано багатофакторну модель пріоритизації, що комбінує репутацію індикаторів з джерел threat intelligence, критичність атакованого активу, рівень привілеїв користувача та аномальність поведінки з ваговими коефіцієнтами, налаштованими під пріоритети політики безпеки організації.

Розроблено набір формалізованих алгоритмів для ключових етапів обробки інциденту: EnrichAlert для автоматичного збагачення подій через паралельні запити до множини джерел threat intelligence, PrioritizeIncident для обчислення ризикового балу на основі багатофакторної моделі, InvestigateIncident для розширення контексту через запити до різнорідних систем організації, ContainThreat для виконання дій локалізації залежно від обчисленого пріоритету, RemediateThreat для усунення артефактів атаки. Запропонована багаторівнева архітектура системи інтегрує гетерогенні компоненти (SIEM, EDR, NDR, IDS/IPS на рівні виявлення, threat intelligence платформи, SOAR як центральний оркестратор, інструменти реагування, аналітичний рівень) через чітко визначені інформаційні потоки, що забезпечує масштабованість та можливість розширення без фундаментальної реструктуризації.

Реалізовано функціональний прототип SOAR-платформи мовою Python з модульною архітектурою, що демонструє практичну застосовність запропонованої концепції. Прототип інтегрує реальний API VirusTotal для перевірки репутації файлів та локальну базу відомих індикаторів компрометації, реалізує багатофакторну модель пріоритизації з налаштовуваними ваговими коефіцієнтами, виконує умовну логіку прийняття рішень щодо автоматизації або ескалації аналітику.

Проведене експериментальне дослідження на п'яти тестових сценаріях підтвердило ефективність розробленої технології, продемонструвавши середній час обробки інциденту 1.03 секунди, що приблизно в 1200 разів швидше за ручну обробку. Рівень автоматизації досяг 60% в тестовій вибірці при збереженні коректності прийняття рішень без жодних помилкових автоматичних дій для легітимної активності. Порівняльний аналіз виявив суттєві переваги запропонованої системи в швидкості реагування, консистентності рішень через детерміновану логіку алгоритмів, необмеженій можливості паралельної обробки та безперервному покритті 24/7 без організації чергувань.

Оформлення результатів цього дослідження здійснювалося згідно з методичними рекомендаціями кафедри [65].

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ismail K., Kurnia R., Brata Z. A., Nelistiani G. A., Heo S., Kim H., Kim H. Toward Robust Security Orchestration and Automated Response in Security Operations Centers with a Hyper-Automation Approach Using Agentic Artificial Intelligence // Information. — 2025. — Vol. 16, No. 5. — Article 365. DOI: https://www.mdpi.com/2078-2489/16/5/365

2. Kumar A., Sharma S., Ahmad M., Alharbi A. A., Tariq U. AI/ML in Security Orchestration, Automation and Response (SOAR) // Intelligent Automation & Soft Computing. — 2021. — Vol. 28, No. 2. — P. 347-360. URL: https://www.techscience.com/iasc/v28n2/42057/html

3. Mavroeidis V., Bromander S. Cyber Threat Intelligence Model: An Evaluation of Taxonomies, Sharing Standards, and Ontologies within Cyber Threat Intelligence // arXiv preprint arXiv:2103.03530. — 2021. — URL: https://arxiv.org/abs/2103.03530

4. Sinha A. Challenges and Best Practices for Integrating Security Orchestration, Automation, and Response (SOAR) Platforms : Master’s Thesis. — Iowa State University, 2024. — URL: https://dr.lib.iastate.edu/bitstreams/48ab713c-0f6c-472a-9564-9d6bee1b394b/download

5. Das A. Automation and Orchestration in Cyber Threat Intelligence (CTI): A Survey // International Journal for Research in Applied Science & Engineering Technology (IJRASET). — 2025. — Vol. 13, Issue IV. URL: https://www.ijraset.com/research-paper/automation-and-orchestration-in-cyber-threat-intelligence

6. National Institute of Standards and Technology (NIST). Computer Security Incident Handling Guide (SP 800 61 Rev. 3). Gaithersburg, MD: NIST, 2023. URL: https://csrc.nist.gov/pubs/sp/800/61/r3/final

7. González Granadillo G., González Zarzosa S., Diaz R. Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures. Sensors, 2021, 21(14):4759. URL: https://www.mdpi.com/1424-8220/21/14/4759

8. Karantzas G., Patsakis C. Quantitative Assessment of Endpoint Detection and Response Systems. Journal of Cybersecurity and Privacy, 2021, 1(3):508–532. URL: https://www.mdpi.com/2624-800X/1/3/21

9. Tounsi W., Rais H. A survey of cyber threat intelligence. Computers & Security, 2018, 72:212–233. URL: https://www.sciencedirect.com/science/article/pii/S0167404817301839

10. OASIS. STIX™ Version 2.1. Committee Specification 02 (OASIS Open). 2021. URL: https://docs.oasis-open.org/cti/stix/v2.1/os/stix-v2.1-os.html

11. OASIS. TAXII™ Version 2.1. Committee Specification 02 (OASIS Open). 2021. URL: https://docs.oasis-open.org/cti/taxii/v2.1/os/taxii-v2.1-os.html

12. MITRE. ATT&CK® — Adversary Tactics, Techniques, and Procedures Knowledge Base. URL: https://attack.mitre.org/

13. Strom B.E., Applebaum A., Miller D.P., et al. MITRE ATT&CK®: Design and Philosophy. The MITRE Corporation, 2020. URL: https://attack.mitre.org/docs/ATTACK_Design_and_Philosophy_March_2020.pdf

14. NIST. SP 800 150: Guide to Cyber Threat Information Sharing. 2016. URL: https://csrc.nist.gov/pubs/sp/800/150/final

15. Du M., Li F., Zheng G., Srikumar V. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS'17). URL: https://dl.acm.org/doi/10.1145/3133956.3134015

16. Kent K., Johnson C., et al. Guide to Computer Security Log Management (SP 800 92 Rev. 1, Initial Public Draft). NIST, 2024. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-92r1.ipd.pdf

17. Karlsen R., Nyre A., Lund M.S., Bjønnes E. Large Language Models for Cybersecurity Incident Response Automation: A Systematic Review. arXiv preprint, 2023. URL: https://arxiv.org/abs/2311.14519

18. Ismail R., Kurnia R., Brata Z.A., Nelistiani G.A., Heo S., Kim H., Kim H. Toward Robust Security Orchestration and Automated Response in Security Operations Centers with a Hyper Automation Approach Using Agentic Artificial Intelligence. Information (MDPI), 2025, 16(5):365. URL: https://www.mdpi.com/2078-2489/16/5/365

19. Zhang J., Chen L., Wang W., et al. When LLMs meet cybersecurity: a systematic literature review. Cybersecurity (SpringerOpen), 2025. URL: https://cybersecurity.springeropen.com/articles/10.1186/s42400-025-00361-w

20. MITRE. ATT&CK® Data & Tools (incl. ATT&CK Navigator). URL: https://attack.mitre.org/resources/attack-data-and-tools/

21. Mezzi E., Massacci F., Tuma K. Large Language Models are Unreliable for Cyber Threat Intelligence. arXiv preprint, 2025. URL: https://arxiv.org/abs/2503.23175

22. Griffioen T., Doerr C., Blom F., et al. Quality Evaluation of Cyber Threat Intelligence Feeds. In: Security and Privacy in Communication Networks (SecureComm 2020). Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer, 2020. URL: https://link.springer.com/chapter/10.1007/978-3-030-57878-7_14

23. Wagner C., Mahbub K., Palomar E., Abdallah A.E. Technical Threat Intelligence Sharing: Survey, Research Directions and Case Studies. Computers & Security, 2019, 87:101589. URL: https://www.sciencedirect.com/science/article/pii/S016740481830467X

24. Khan Z.A., Shiaeles S., Kolokotronis N., etc. Impact of log parsing on deep learning based anomaly detection. PeerJ Computer Science, 2024, 10:e1862. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC11330418/

25. Kirillov I., Parnitzke S., Kushner D., et al. D3FEND™: Toward a Knowledge Graph of Cybersecurity Countermeasures. MITRE, 2021. URL: https://d3fend.mitre.org/resources/D3FEND.pdf

26. Bridges R.A., Benjamin R., Iyer R.K., et al. Testing SOAR tools in use: Characterization and gaps. Computers & Security, 2023, 126:102993. URL: https://www.sciencedirect.com/science/article/pii/S0167404823001116

27. Fernandes R., Romano S., Pujol Perich A., et al. A Review of SOAR Frameworks and Platforms. In: Lecture Notes in Computer Science, 2025. URL: https://link.springer.com/chapter/10.1007/978-3-031-95963-9_38

28. Tilbury J., Flowerday S. Humans and Automation: Augmenting Security Operation Centers. Journal of Cybersecurity and Privacy, 2024, 4(3):388–409. URL: https://www.mdpi.com/2624-800X/4/3/20

29. Ammi M., M'bika M., Bedr E. Cyber Threat Hunting Case Study using MISP. Journal of Information Systems and Industrial Security (JISIS), 2023. URL: https://jisis.org/wp-content/uploads/2023/06/2023.I2.001.pdf

30. Ackermann T. Standardisierung von SOAR Plattformen: Anforderungen und Referenzarchitektur. Automatisierungstechnik (De Gruyter), 2023. URL: https://www.degruyter.com/document/doi/10.1515/auto-2023-0057/html

31. Dwivedi S., Dutta A., Tulachan A., et al. AI4SOAR: A Security Intelligence Tool for Automated Incident Response. In: Proceedings of the 2024 ACM AsiaCCS Workshops (AISec/SOUPS/…). URL: https://dl.acm.org/doi/10.1145/3664476.3670450

32. Md Omar A.Y., Karlzén H., Sommestad T. Automatic incident response solutions: a review of proposed solutions' input and output. In: Proceedings of the 18th International Conference on Availability, Reliability and Security (ARES 2024). URL: https://dl.acm.org/doi/10.1145/3600160.3605066

33. Schlette D., Caselli M., Pernul G. A Comparative Study on Cyber Threat Intelligence: The Security Incident Response Perspective. IEEE Communications Surveys & Tutorials, 2021, 23(4):2525–2556. URL: https://ieeexplore.ieee.org/document/9564691

34. Chatziamanetoglou D., Rantos K. Weighted quality criteria for cyber threat intelligence: assessment and prioritisation in the MISP data model. International Journal of Information Security, 2025, 24(4). URL: https://link.springer.com/article/10.1007/s10207-025-01080-6

35. NIST. SP 800 55 v1: Measurement Guide for Information Security — Volume 1: Identifying and Selecting Measures (Final). 2024. URL: https://csrc.nist.gov/pubs/sp/800/55/v1/final

36. NIST. SP 800 55 v2: Measurement Guide for Information Security — Volume 2: Developing an Information Security Measurement Program (Final). 2024. URL: https://csrc.nist.gov/pubs/sp/800/55/v2/final

37. Dempsey K., Pillitteri V., Baer M. SP 800 137A: Assessing Information Security Continuous Monitoring (ISCM) Programs. NIST, 2020. URL: https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-137A.pdf

38. Stine K., Quinn S., Gardner R., Witte G. NISTIR 8286: Integrating Cybersecurity and Enterprise Risk Management (ERM). NIST, 2020. URL: https://csrc.nist.gov/pubs/ir/8286/final

39. Tariq S., Ahmad A., Slay J., et al. Alert Fatigue in Security Operations Centres: Causes and Mitigations. Communications of the ACM, 2025. URL: https://dl.acm.org/doi/10.1145/3723158

40. MITRE Engenuity. ATT&CK® Evaluations — Methodology Overview. URL: https://evals.mitre-engenuity.org/methodology-overview/

41. Cybersecurity and Infrastructure Security Agency (CISA). Automated Indicator Sharing (AIS). URL: https://www.cisa.gov/ais

42. Agyepong E., Cherdantseva Y., Reinecke P., Burnap P. A systematic method for measuring the performance of a cyber security operations centre analyst. Computers & Security, 2023, 124:102959. URL: https://www.sciencedirect.com/science/article/pii/S0167404822003510

43. Soble J. Суд над главою Mt. Gox: заперечення нових звинувачень. The Guardian. 2017 URL: https://www.theguardian.com/technology/2017/jul/11/gox-bitcoin-exchange-mark-karpeles-on-trial-japan-embezzlement-loss-of-millions

44. BBC News. Хакери вкрали $72 мільйони з біржі Bitfinex. 2016. URL: https://en.wikipedia.org/wiki/2016_Bitfinex_hack#:~:text=The%20Bitfinex%20cryptocurrency%20exchange%20was,at%20the%20time%2C%20were%20stolen.

45. BBC News. Криптовалюта на суму $530 мільйонів викрадена з японської біржі Coincheck. 2018. URL: https://www.bbc.com/news/world-asia-42845505

46. BBC News. Криптовалютна біржа Binance зазнала хакерської атаки. 2019. URL: https://www.bbc.com/news/technology-37009319

47. The Verge. Poly Network зазнала хакерської атаки на суму понад $600 мільйонів. 2021. URL: https://www.cnbc.com/2021/08/13/poly-network-hack-nearly-all-of-600-million-in-crypto-returned.html

48. BBC News. Ronin Network: хакери вкрали $625 мільйонів у криптовалюті. 2022. URL: https://www.bbc.com/news/technology-60933174

49. Міністерство юстиції США. Семюел Бенкман-Фрід визнаний винним у шахрайстві щодо клієнтів та інвесторів FTX. 2023. URL:https://www.justice.gov/usao-sdny/pr/samuel-bankman-fried-sentenced-25-years-prison

50. Гулак Г., Жданова Ю., Складанний П., Гулак Є., Корнієць В. (2022) Уразливості шифрування коротких повідомлень в мобільних інформаційно-комунікаційних системах об'єктів критичної інфраструктури. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 1(17), 145–158. https://doi.org/10.28925/2663-4023.2022.17.145158

51. Romaniuk, O., Skladannyi, P., & Shevchenko, S. (2022). Comparative analysis of solutions to provide control and management of privileged access in the IT environment. Cybersecurity: Education, Science, Technique, 4(16), 98–112. https://doi.org/10.28925/2663-4023.2022.16.98112

52. Скіцько, О., Складанний, П., Ширшов, Р., Гуменюк, М., & Ворохоб, М. (2023). Загрози та ризики використання штучного інтелекту. Кібербезпека: освіта, наука, техніка, 2(22), 6–18. https://doi.org/10.28925/2663-4023.2023.22.618

53. Марценюк , М., Козачок, В., Богданов, О., Іосіфов, Є., & Бржевська , З. (2023). Аналіз методів виявлення дезінформації в соціальних мережах за допомогою машинного навчання. Кібербезпека: освіта, наука, техніка, 2(22), 148–155. https://doi.org/10.28925/2663-4023.2023.22.148155

54. Спасітєлєва, С., Чичкань, І., Шевченко, С., & Жданова, Ю. (2023). Розробка безпечних контейнерних застосунків з мікросервісною архітектурою. Кібербезпека: освіта, наука, техніка, 1(21), 193–210. https://doi.org/10.28925/2663-4023.2023.21.193210

55. Ворохоб, М., Киричок, Р., Яскевич, В., Добришин, Ю., & Сидоренко, С. (2023). Сучасні перспективи застосування концепції zero trust при побудові політики інформаційної безпеки підприємства. Кібербезпека: освіта, наука, техніка, 1(21), 223–233. https://doi.org/10.28925/2663-4023.2023.21.223233

56. Добришин, Ю., Сидоренко, С., & Ворохоб, М. (2023). Автоматизована система підтримки прийняття рішення щодо відновлення пошкодженого програмного забезпечення внаслідок впливу кібератак. Кібербезпека: освіта, наука, техніка, 4(20), 174–182. https://doi.org/10.28925/2663-4023.2023.20.174182

57. Крючкова Л., Складанний П., Ворохоб М. (2023). Передпроєктні рішення щодо побудови системи авторизації на основі концепції Zero Trust. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 3(19), 226–242. https://doi.org/10.28925/2663-4023.2023.13.226242

58. Корнієць, В., & Складанний, П. (2024). Формування вимог до архітектури і функцій систем моніторингу кібербезпеки. Телекомунікаційні та інформаційні технології, 4(85), 90–96. https://doi.org/10.31673/2412-4338.2024.040224

59. Костюк, Ю., Складанний, П., Рзаєва, С., Мазур, Н., Черевик, В., & Аносов, А. (2025). Особливості реалізації мережевих атак через TCP/IP-протоколи.

Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 1(29), 571–597. https://doi.org/10.28925/2663-4023.2025.29.915

60. Складанний , П., Костюк, Ю., Рзаєва, С., Самойленко, Ю., & Савченко, Т. (2025). Розробка модульних нейронних мереж для виявлення різних класів мережевих атак. Кібербезпека: освіта, наука, техніка, 3(27), 534–548. https://doi.org/10.28925/2663-4023.2025.27.772

61. Костюк, Ю., Бебешко, Б., Складанний, П., Рзаєва, С., & Хорольська, К. (2025). Оптимізація буфера та пріоритетів для забезпечення безпеки у Bluetooth-мережах. Безпека інформаційних систем і технологій, 2(8), 5–16. https://doi.org/10.17721/ISTS.2024.8.5-16

62. V. Shapoval, et al., Automation of Data Management Processes in Cloud Storage, in: Workshop on Cybersecurity Providing in Information and Telecommunication Systems, CPITS, vol. 3654 (2024) 410–418.

63. Korshun, N., Myshko, I., Tkachenko, O. Automation and Management in Operating Systems: The Role of Artificial Intelligence and Machine Learning. CEUR Workshop Proceedings, 2023, 3687, pp. 59–68.

64. Islam C., Babar M. A., Nepal S. A Multi-Vocal Review of Security Orchestration [Мультивокальний огляд оркестрації безпеки]. ACM Computing Surveys. 2019. Vol. 52, No. 2. Art. 37. P. 1–45. URL: https://dl.acm.org/doi/10.1145/3305268

65. Жданова, Ю. Д., Складанний, П. М., & Шевченко, С. М. (2023). Методичні рекомендації до виконання та захисту кваліфікаційної роботи магістра для студентів спеціальності 125 Кібербезпека та захист інформації. https://elibrary.kubg.edu.ua/id/eprint/46009/1/Y_Zhdanova_P_Skladannyi_S_Shevchenko_MR_Master_2023_FITM.pdf

ДОДАТОК

```
import requests
import json
import time
from datetime import datetime
from typing import Dict, List, Any

# ============================================================================
# КОНФІГУРАЦІЯ
# ============================================================================

CONFIG = {
    "virustotal_api_key": "d1a21834cdf5ea1e789527285cf13bc009653e0ce5d37bdcc51ca3bb1b92161d",
    "reputation_threshold": 50,  # Поріг репутації для блокування (0-100)
    "priority_weights": {
        "reputation": 0.4,
        "asset_criticality": 0.3,
        "user_privileges": 0.2,
        "anomaly": 0.1
    }
}

# Симульована база шкідливих IP (для тестування)
KNOWN_MALICIOUS_IPS = {
    '185.220.101.1': {'score': 95, 'country': 'RU', 'category': 'tor_exit_node'},
    '45.142.212.61': {'score': 88, 'country': 'NL', 'category': 'c2_server'},
    '1.2.3.4': {'score': 75, 'country': 'CN', 'category': 'scanning'},
    '192.0.2.1': {'score': 82, 'country': 'US', 'category': 'malware_distribution'},
}

# ============================================================================
# АЛГОРИТМ 1: ЗБАГАЧЕННЯ ПОДІЙ (EnrichAlert)
# ============================================================================

def enrich_alert(alert: Dict[str, Any]) -> Dict[str, Any]:
    """
    Збагачує алерт даними з Threat Intelligence джерел

    Вхід: alert - базова подія від SIEM
    Вихід: збагачена подія з контекстом TI
    """
    print(f"\n[ЗБАГАЧЕННЯ] Обробка алерту: {alert['id']}")
    print(f"  Тип інциденту: {alert.get('type', 'unknown')}")

    # Витягуємо індикатори
    indicators = alert.get('indicators', {})
    enriched_indicators = []

    # Збагачуємо IP-адресу (симуляція через локальну базу)
    if 'ip' in indicators:
```

```
            ip = indicators['ip']
            print(f"  → Перевірка IP: {ip}")

            ip_reputation = check_ip_reputation(ip)
            enriched_indicators.append({
                'type': 'ip',
                'value': ip,
                'reputation': ip_reputation['score'],
                'malicious': ip_reputation['is_malicious'],
                'country': ip_reputation.get('country', 'Unknown'),
                'category': ip_reputation.get('category', 'unknown')
            })
            print(f"    Репутація: {ip_reputation['score']}/100 | "
                  f"Шкідливий: {ip_reputation['is_malicious']} | "
                  f"Країна: {ip_reputation.get('country')}")

        # Збагачуємо хеш файлу (реальний API VirusTotal)
        if 'file_hash' in indicators:
            file_hash = indicators['file_hash']
            print(f"  → Перевірка хешу файлу: {file_hash}")

            file_reputation = check_file_reputation(file_hash)
            enriched_indicators.append({
                'type': 'file_hash',
                'value': file_hash,
                'reputation': file_reputation['score'],
                'malicious': file_reputation['is_malicious'],
                'detections': file_reputation.get('detections', 0),
                'total_engines': file_reputation.get('total_engines', 0)
            })
            print(f"    Репутація: {file_reputation['score']:.1f}/100 | "
                  f"Детекцій:
{file_reputation.get('detections')}/{file_reputation.get('total_engines')}")

        # Додаємо організаційний контекст
        asset_context = {
            'criticality': alert.get('asset_criticality', 2),  # 1-low, 2-medium, 3-high
            'business_impact': ['low', 'medium', 'high'][alert.get('asset_criticality', 2) - 1],
            'host': alert.get('host', 'unknown')
        }

        user_context = {
            'privilege_level': alert.get('user_privilege', 1),  # 1-user, 2-admin, 3-domain_admin
            'user_id': alert.get('user', 'unknown'),
            'department': 'IT'
        }

        # Формуємо збагачений алерт
        enriched_alert = alert.copy()
        enriched_alert['enriched_indicators'] = enriched_indicators
        enriched_alert['asset_context'] = asset_context
```

```
    enriched_alert['user_context'] = user_context
    enriched_alert['enrichment_timestamp'] = datetime.now().isoformat()

    print(f"  ✓ Збагачення завершено ({len(enriched_indicators)} індикаторів)")
    return enriched_alert


def check_ip_reputation(ip: str) -> Dict[str, Any]:
    """
    Перевіряє репутацію IP через локальну базу відомих загроз
    (Симуляція Threat Intelligence для тестування)
    """
    # Перевірка в базі відомих шкідливих IP
    if ip in KNOWN_MALICIOUS_IPS:
        threat_data = KNOWN_MALICIOUS_IPS[ip]
        return {
            'score': threat_data['score'],
            'is_malicious': True,
            'country': threat_data['country'],
            'category': threat_data['category']
        }

    # Безпечні IP (Google DNS, Cloudflare DNS, тощо)
    safe_ips = ['8.8.8.8', '1.1.1.1', '8.8.4.4', '1.0.0.1']
    if ip in safe_ips:
        return {
            'score': 0,
            'is_malicious': False,
            'country': 'US',
            'category': 'legitimate_service'
        }

    # Невідомі IP — низька репутація за замовчуванням
    return {
        'score': 25,
        'is_malicious': False,
        'country': 'Unknown',
        'category': 'unknown'
    }


def check_file_reputation(file_hash: str) -> Dict[str, Any]:
    """
    Перевіряє репутацію файлу через VirusTotal API
    """
    try:
        headers = {'x-apikey': CONFIG['virustotal_api_key']}

        print(f"    Запит до VirusTotal API...")
        response = requests.get(
            f'https://www.virustotal.com/api/v3/files/{file_hash}',
```

```
            headers=headers,
            timeout=15
        )

        if response.status_code == 200:
            data = response.json()['data']['attributes']
            stats = data['last_analysis_stats']

            malicious = stats.get('malicious', 0)
            suspicious = stats.get('suspicious', 0)
            total = sum(stats.values())

            # Обчислюємо репутаційну оцінку
            detections = malicious + suspicious
            score = (detections / total * 100) if total > 0 else 0

            return {
                'score': score,
                'is_malicious': malicious > 3,  # Більше 3 AV детектують
                'detections': malicious,
                'total_engines': total,
                'first_seen': data.get('first_submission_date', 'unknown')
            }

        elif response.status_code == 404:
            print(f"    ⚠ Файл не знайдено в базі VirusTotal")
            return {
                'score': 0,
                'is_malicious': False,
                'detections': 0,
                'total_engines': 0
            }

        else:
            print(f"    ⚠ Помилка API: {response.status_code}")
            return {
                'score': 50,  # Невідома репутація
                'is_malicious': False,
                'detections': 0,
                'total_engines': 0
            }

    except Exception as e:
        print(f"    ⚠ Помилка перевірки файлу: {e}")
        return {
            'score': 50,
            'is_malicious': False,
            'detections': 0,
            'total_engines': 0
        }
```

```
# ============================================================================
# АЛГОРИТМ 2: ПРІОРИТИЗАЦІЯ (PrioritizeIncident)
# ============================================================================

def prioritize_incident(enriched_alert: Dict[str, Any]) -> float:
    """
    Обчислює пріоритет інциденту на основі багатофакторної моделі

    Вхід: enriched_alert - збагачена подія
    Вихід: пріоритетний бал (0-10)
    """
    print(f"\n[ПРІОРИТИЗАЦІЯ] Оцінка критичності інциденту...")

    # Фактор 1: Репутація індикаторів (0-100)
    reputation_scores = [
        ind['reputation']
        for ind in enriched_alert.get('enriched_indicators', [])
    ]
    avg_reputation = sum(reputation_scores) / len(reputation_scores) if reputation_scores else
0
    rep_normalized = avg_reputation / 100  # Нормалізація до 0-1

    # Фактор 2: Критичність активу (1-3)
    asset_criticality = enriched_alert['asset_context']['criticality']
    asset_normalized = (asset_criticality - 1) / 2  # 1-3 → 0-1

    # Фактор 3: Рівень привілеїв користувача (1-3)
    user_privilege = enriched_alert['user_context']['privilege_level']
    user_normalized = (user_privilege - 1) / 2  # 1-3 → 0-1

    # Фактор 4: Аномальність поведінки (спрощена модель)
    # У реальній системі тут була б ML модель
    anomaly_normalized = 0.5

    # Зважена комбінація факторів
    weights = CONFIG['priority_weights']
    priority_score = (
        weights['reputation'] * rep_normalized +
        weights['asset_criticality'] * asset_normalized +
        weights['user_privileges'] * user_normalized +
        weights['anomaly'] * anomaly_normalized
    ) * 10  # Масштабування до 0-10

    print(f"  Факторний аналіз:")
    print(f"                →    Репутація    індикаторів:    {avg_reputation:.1f}/100    (вага:
{weights['reputation']})")
    print(f"                    →     Критичність     активу:     {asset_criticality}/3     (вага:
{weights['asset_criticality']})")
    print(f"                    →     Привілеї     користувача:     {user_privilege}/3     (вага:
{weights['user_privileges']})")
```

```
    print(f"    → Аномальність: {anomaly_normalized:.2f} (вага: {weights['anomaly']})")
    print(f"  ✓ Пріоритетний бал: {priority_score:.2f}/10")

    return priority_score


# ============================================================================
# АЛГОРИТМ 3: ЛОКАЛІЗАЦІЯ ЗАГРОЗИ (ContainThreat)
# ============================================================================

def contain_threat(enriched_alert: Dict[str, Any], priority: float) -> List[Dict[str, Any]]:
    """
    Виконує дії для локалізації загрози

    Вхід: enriched_alert - збагачена подія, priority - пріоритет
    Вихід: список виконаних дій
    """
    print(f"\n[ЛОКАЛІЗАЦІЯ] Визначення стратегії реагування...")

    actions = []

    # Автоматична локалізація при високому пріоритеті (>7)
    if priority > 7.0:
        print(f"  → Пріоритет {priority:.2f}/10 — автоматичне реагування")

        # Блокування шкідливих IP-адрес
        for ind in enriched_alert.get('enriched_indicators', []):
            if ind['type'] == 'ip' and ind['malicious']:
                action = block_ip(ind['value'], ind.get('category', 'unknown'))
                actions.append(action)

        # Ізоляція хоста при виявленні шкідливого файлу
        for ind in enriched_alert.get('enriched_indicators', []):
            if ind['type'] == 'file_hash' and ind['malicious']:
                host = enriched_alert.get('host', 'unknown')
                action = isolate_host(host)
                actions.append(action)

                # Додаткова дія: видалення шкідливого файлу
                action = remove_malware(host, ind['value'])
                actions.append(action)

        # Блокування облікового запису при компрометації
        if enriched_alert.get('type') == 'account_compromise':
            user = enriched_alert['user_context']['user_id']
            action = disable_account(user)
            actions.append(action)

            # Примусова зміна пароля
            action = force_password_reset(user)
            actions.append(action)
```

```
    else:
        print(f"  → Пріоритет {priority:.2f}/10 — передача аналітику SOC")
        actions.append({
            'action': 'escalate_to_analyst',
            'reason': f'priority_below_threshold ({priority:.2f} < 7.0)',
            'status': 'pending_manual_review',
            'timestamp': datetime.now().isoformat()
        })

    print(f"  ✓ Виконано дій: {len(actions)}")
    return actions


def block_ip(ip: str, category: str) -> Dict[str, Any]:
    """Симуляція блокування IP в firewall"""
    print(f"  → Блокування IP: {ip} (категорія: {category})")
    time.sleep(0.3)  # Імітація мережевої затримки
    return {
        'action': 'block_ip',
        'target': ip,
        'category': category,
        'status': 'success',
        'system': 'firewall',
        'timestamp': datetime.now().isoformat()
    }


def isolate_host(host: str) -> Dict[str, Any]:
    """Симуляція ізоляції хоста через EDR"""
    print(f"  → Ізоляція хоста від мережі: {host}")
    time.sleep(0.4)
    return {
        'action': 'isolate_host',
        'target': host,
        'status': 'success',
        'system': 'EDR',
        'timestamp': datetime.now().isoformat()
    }


def remove_malware(host: str, file_hash: str) -> Dict[str, Any]:
    """Симуляція видалення шкідливого файлу"""
    print(f"  → Видалення шкідливого файлу: {file_hash[:16]}... на {host}")
    time.sleep(0.3)
    return {
        'action': 'remove_malware',
        'target': host,
        'file_hash': file_hash,
        'status': 'success',
        'system': 'EDR',
```

```
        'timestamp': datetime.now().isoformat()
    }


def disable_account(user: str) -> Dict[str, Any]:
    """Симуляція блокування облікового запису"""
    print(f"  → Блокування облікового запису: {user}")
    time.sleep(0.3)
    return {
        'action': 'disable_account',
        'target': user,
        'status': 'success',
        'system': 'IAM',
        'timestamp': datetime.now().isoformat()
    }


def force_password_reset(user: str) -> Dict[str, Any]:
    """Симуляція примусової зміни пароля"""
    print(f"  → Примусова зміна пароля: {user}")
    time.sleep(0.2)
    return {
        'action': 'force_password_reset',
        'target': user,
        'status': 'success',
        'system': 'IAM',
        'timestamp': datetime.now().isoformat()
    }


# ============================================================================
# ГОЛОВНИЙ WORKFLOW
# ============================================================================

def process_incident(alert: Dict[str, Any]) -> Dict[str, Any]:
    """
    Повний цикл обробки інциденту через SOAR платформу
    """
    start_time = time.time()

    print("\n" + "=" * 80)
    print(f"□ ОБРОБКА ІНЦИДЕНТУ: {alert['id']}")
    print("=" * 80)

    # Етап 1: Збагачення подій через Threat Intelligence
    enriched_alert = enrich_alert(alert)

    # Етап 2: Пріоритизація на основі багатофакторної моделі
    priority = prioritize_incident(enriched_alert)

    # Етап 3: Локалізація загрози (автоматично або ескалація)
```

```
    actions = contain_threat(enriched_alert, priority)

    # Підсумкові метрики
    processing_time = time.time() - start_time
    automated = priority > 7.0

    result = {
        'alert_id': alert['id'],
        'incident_type': alert.get('type', 'unknown'),
        'priority_score': round(priority, 2),
        'automated_response': automated,
        'actions_taken': len(actions),
        'actions_details': actions,
        'processing_time_seconds': round(processing_time, 2),
        'completion_timestamp': datetime.now().isoformat()
    }

    print(f"\n{'—' * 80}")
    print(f"✓ ІНЦИДЕНТ ОБРОБЛЕНО")
    print(f"  • Пріоритет: {priority:.2f}/10")
    print(f"  • Режим: {'Автоматичний' if automated else 'Ручний (аналітик)'}")
    print(f"  • Виконано дій: {len(actions)}")
    print(f"  • Час обробки: {processing_time:.2f} сек")
    print("=" * 80)

    return result


# ============================================================================
# ТЕСТОВІ СЦЕНАРІЇ ДЛЯ ЕКСПЕРИМЕНТАЛЬНОГО ДОСЛІДЖЕННЯ
# ============================================================================

def run_experiments():
    """
    Запуск експериментальних тестів для розділу 3.2 диплому
    """

    # Набір реалістичних тестових сценаріїв
    test_scenarios = [
        {
            'id': 'INC-2024-001',
            'type': 'malware_infection',
            'description': 'Виявлено шкідливий файл на робочій станції',
            'indicators': {
                'file_hash': '44d88612fea8a8f36de82e1278abb02f',  # EICAR тестовий вірус
                'ip': '185.220.101.1'  # Відомий C2 сервер
            },
            'asset_criticality': 3,  # High
            'user_privilege': 1,     # Regular user
            'host': 'WKS-FINANCE-15'
        },
```

```
        {
            'id': 'INC-2024-002',
            'type': 'suspicious_login',
            'description': 'Підозрілий вхід з безпечної IP-адреси',
            'indicators': {
                'ip': '8.8.8.8'  # Google DNS (легітимний)
            },
            'asset_criticality': 1,  # Low
            'user_privilege': 1,     # Regular user
            'user': 'john.doe',
            'host': 'WKS-HR-08'
        },
        {
            'id': 'INC-2024-003',
            'type': 'account_compromise',
            'description': 'Компрометація адміністративного облікового запису',
            'indicators': {
                'ip': '45.142.212.61'  # Відомий малверний IP
            },
            'asset_criticality': 3,  # High
            'user_privilege': 3,     # Domain admin
            'user': 'admin.petrov',
            'host': 'SRV-DC-01'
        },
        {
            'id': 'INC-2024-004',
            'type': 'network_scanning',
            'description': 'Сканування мережі з невідомої IP-адреси',
            'indicators': {
                'ip': '203.0.113.42'  # Невідомий IP
            },
            'asset_criticality': 2,  # Medium
            'user_privilege': 1,
            'host': 'WKS-IT-22'
        },
        {
            'id': 'INC-2024-005',
            'type': 'data_exfiltration',
            'description': 'Спроба витоку даних через шкідливий файл',
            'indicators': {
                'file_hash':
'275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f',
                'ip': '1.2.3.4'
            },
            'asset_criticality': 3,  # High
            'user_privilege': 2,     # Local admin
            'user': 'backup.service',
            'host': 'SRV-FILE-03'
        }
    ]
```

```
    print("\n" + "=" * 80)
    print("□ ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОТОТИПУ SOAR")
    print("   Магістерська робота: Технологія автоматизованого управління кіберзагрозами")
    print("=" * 80)
    print(f"\nКількість тестових сценаріїв: {len(test_scenarios)}")
    print(f"Дата початку: {datetime.now().strftime('%Y-%m-%d %H:%M:%S')}\n")

    results = []

    # Обробка кожного сценарію
    for i, scenario in enumerate(test_scenarios, 1):
        print(f"\n{'▶' * 40}")
        print(f"СЦЕНАРІЙ {i}/{len(test_scenarios)}")
        print(f"Опис: {scenario['description']}")
        print(f"{'▶' * 40}")

        result = process_incident(scenario)
        results.append(result)

        # Пауза між сценаріями
        if i < len(test_scenarios):
            time.sleep(2)

    # Збереження результатів у JSON
    output_file = 'experiment_results.json'
    with open(output_file, 'w', encoding='utf-8') as f:
        json.dump(results, f, indent=2, ensure_ascii=False)

    # Обчислення підсумкової статистики
    print("\n" + "=" * 80)
    print("□ ПІДСУМКОВА СТАТИСТИКА ЕКСПЕРИМЕНТІВ")
    print("=" * 80)

    total_incidents = len(results)
    automated_incidents = sum(1 for r in results if r['automated_response'])
    manual_incidents = total_incidents - automated_incidents

    total_actions = sum(r['actions_taken'] for r in results)
    avg_processing_time = sum(r['processing_time_seconds'] for r in results) / total_incidents

    avg_priority = sum(r['priority_score'] for r in results) / total_incidents

    print(f"\n□ Загальні показники:")
    print(f"  • Всього оброблено інцидентів: {total_incidents}")
    print(f"                        •           Автоматизовано:           {automated_incidents}
({automated_incidents/total_incidents*100:.1f}%)")
    print(f"                  •         Передано         аналітику:         {manual_incidents}
({manual_incidents/total_incidents*100:.1f}%)")
    print(f"  • Всього виконано дій: {total_actions}")
    print(f"  • Середній час обробки: {avg_processing_time:.2f} сек")
    print(f"  • Середній пріоритет: {avg_priority:.2f}/10")
```

```
    print(f"\n Детальні результати збережено у файл: {output_file}")
    print("=" * 80 + "\n")

    return results


# ============================================================================
# ТОЧКА ВХОДУ
# ============================================================================

if __name__ == '__main__':
    print("\n ЗАПУСК SOAR ПРОТОТИПУ\n")
    run_experiments()
    print("\n✓ ЕКСПЕРИМЕНТ ЗАВЕРШЕНО\n")
```

ВІДГУК

на кваліфікаційну роботу магістра

студента **Яблокова Іллі Ростиславовича**

на тему: Технологія автоматизованого управління кіберзагрозами з використанням SOAR та інструментів Threat Intelligence

Кваліфікаційна робота є ґрунтовним дослідженням, присвяченим актуальній та практично важливій темі – розробленню технології автоматизованого управління кіберзагрозами з використанням SOAR-платформи та джерел Threat Intelligence. Актуальність роботи зумовлена зростанням складності кіберзагроз та потребою зменшення навантаження на аналітиків SOC за рахунок формалізованої автоматизації реагування.

Позитивною стороною роботи є чітке теоретико-методичне обґрунтування запропонованої концепції: проведено аналіз ролі SOAR-платформ у зв'язці з SIEM, системами виявлення загроз та TI-платформами, сформовано концептуальний операційний ланцюг обробки інцидентів, формалізовано життєвий цикл інциденту та сценарії реагування на базі стандарту CACAO з використанням STIX/TAXII. Окремо слід відзначити формалізацію ключових алгоритмів автоматизації (EnrichAlert, PrioritizeIncident, InvestigateIncident, ContainThreat, RemediateThreat), які перетворюють загальну ідею автоматизованого реагування на чітко структуровані, машинозчитувані процедури.

Практична значущість роботи полягає в реалізації прототипу SOAR-рішення мовою Python, що інтегрується з реальними джерелами аналітичної інформації (VirusTotal) та локальними базами репутації, реалізує багатофакторну модель пріоритизації інцидентів та симулює дії локалізації й усунення загроз. Проведене експериментальне дослідження на низці типових сценаріїв (зараження шкідливим ПЗ, компрометація облікових записів, мережеве сканування тощо) продемонструвало істотне скорочення часу реагування, збалансований розподіл між автоматизацією та участю аналітика, а також коректність обраних стратегій. Сформульовані автором рекомендації щодо етапного впровадження, налаштування параметрів пріоритизації, інтеграції з існуючою інфраструктурою SOC та організаційних аспектів експлуатації свідчать про високий рівень практичної орієнтованості роботи.

Перелік використаних джерел свідчить про вміння студента розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання кваліфікаційної роботи Яблоков І.Р. показав хорошу теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконував сумлінно, акуратно та вчасно за планом.

Висновок: Рекомендую допустити роботу до захисту.

Науковий керівник

доктор філософії, доцент — Киричок Роман Васильович

(посада, вчений ступінь, вчене звання) (прізвище, ім'я, по батькові)

« ___ » _____ 2025 р.

РЕЦЕНЗІЯ

на кваліфікаційну роботу магістра

студента **Яблокова Іллі Ростиславовича**

на тему: Технологія автоматизованого управління кіберзагрозами з використанням SOAR та інструментів Threat Intelligence

Актуальність. Робота присвячена актуальній проблемі сучасної кібербезпеки – автоматизованому управлінню кіберзагрозами на основі інтеграції SOAR-платформ та джерел Threat Intelligence. В умовах зростання обсягів оповіщень, дефіциту кваліфікованих аналітиків SOC та необхідності скорочення часу реагування на інциденти, запропонована тематика є вкрай затребуваною як для корпоративних SOC, так і для сектору критичної інфраструктури. Автор обґрунтовано акцентує на переході від ручної, фрагментованої обробки інцидентів до адаптивної автоматизації, що спирається на формалізовані сценарії та контекстне збагачення подій.

Позитивні сторони.

1. Важливою сильною стороною є формалізація playbook-орієнтованої моделі з опорою на стандарт CACAO та поданням низки ключових алгоритмів (EnrichAlert, PrioritizeIncident, InvestigateIncident, ContainThreat, RemediateThreat) у вигляді псевдокоду. Це переводить абстрактну ідею автоматизації в конкретні, відтворювані процедури, що можуть бути реалізовані в реальних системах.

2. Запропонована багаторівнева архітектура системи (рівні виявлення, TI, SOAR-оркестрації, виконання дій, зберігання та аналітики) описана послідовно й логічно, із виокремленням інформаційних потоків між компонентами.

Недоліки та зауваження.

1. Експериментальне дослідження, хоча й добре структуроване, базується на відносно невеликій кількості сценаріїв і частково симульованому середовищі (локальна база IP, спрощений фактор аномальності). Для більшої переконливості результатів доцільно було б розширити вибірку інцидентів та/або показати роботу прототипу на близьких до реальних журналах.

2. У роботі якісно порівнюється запропонований підхід з «традиційною» ручною обробкою, однак бракує детальнішого зіставлення із сучасними комерційними або open-source рішеннями (хоча б на концептуальному рівні – за набором функцій, моделлю автоматизації, архітектурними принципами). Таке порівняння дозволило б чіткіше позиціонувати внесок автора відносно існуючих практик.

Відзначені зауваження не впливають суттєво на загальну позитивну оцінку кваліфікаційної роботи магістра.

Висновок: Кваліфікаційна робота магістра заслуговує оцінку «добре», а її автор Яблоков І.Р. – присвоєння кваліфікації магістра спеціальності 125 Кібербезпека та захист інформації.

Рецензент

_____,_____

(посада, вчений ступінь, вчене звання) (прізвище, ім’я, по батькові)

« ___ » _____ 2025 р.

КИЇВСЬКИЙ СТОЛИЧНИЙ УНІВЕРСИТЕТ ІМЕНІ БОРИСА ГРІНЧЕНКА

ПОДАННЯ
ГОЛОВІ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ
ЩОДО ЗАХИСТУ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Направляється студент Яблоков І.Р. до захисту кваліфікаційної роботи
(прізвище та ініціали)
магістра за спеціальністю 125 Кібербезпека та захист інформації освітньої програми 125.00.01 Безпека інформаційних і комунікаційних систем на тему:
«Технологія автоматизованого управління кіберзагрозами з використанням SOAR та інструментів Threat Intelligence»
(назва теми)

Кваліфікаційна робота і рецензія додаються.
Завідувач кафедри Складанний П.М.
(підпис)
(прізвище та ініціали)

Довідка про успішність

Яблоков І.Р. за період навчання на факультеті з 2023 року по 2024 рік
(прізвище та ініціали студента)
повністю виконав навчальний план за спеціальністю з таким розподілом оцінок за:
національною шкалою: відмінно ___%, добре ___%, задовільно ___%;
шкалою ECTS: A ____%; B ____%; C ____%; D ____%; E ____%.

Методист факультету _____
(підпис) (прізвище та ініціали)

Висновок керівника кваліфікаційної роботи

Студент ***Яблоков І.Р.*** обрав тему роботи, метою якої було підвищення ефективності управління кіберзагрозами в корпоративних інформаційних системах шляхом автоматизації процесів виявлення, аналізу та реагування на інциденти безпеки з використанням оркестрованої взаємодії засобів SOAR і платформ Threat Intelligence. Перелік використаних джерел свідчить про вміння студента розбиратись в наукових питаннях та застосовувати їх при дослідженнях. Під час виконання кваліфікаційної роботи Яблоков І.Р. показав хорошу теоретичну та практичну підготовку, вміння самостійно вирішувати питання і робити висновки. Роботу виконував сумлінно, акуратно та вчасно за планом.

Все це дозволяє оцінити виконану магістерську роботу студента ***Яблокова І.Р.*** на оцінку **«добре»** та присвоїти йому кваліфікацію магістра спеціальності 125 Кібербезпека та захист інформації.

Керівник роботи Киричок Р.В.
(підпис) (прізвище та ініціали)
“____” _____ 2024 року

Висновок кафедри про кваліфікаційну роботу

Кваліфікаційна робота розглянута. Студент Яблоков І.Р.
(прізвище та ініціали)
допускається до захисту даної роботи в Державній екзаменаційній комісії.
Завідувач кафедри інформаційної
та кібернетичної безпеки
імені професора Володимира Бурячка
кандидат технічних наук, доцент

Складанний П.М.
(підпис) (прізвище та ініціали)
“____” _____ 2025 року