Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ДОПУСТИТИ ДО ЗАХИСТУ:
В.о. завідувача кафедри
кібербезпеки та захисту
інформації
_____ Іван ПАРХОМЕНКО
«___» червня 2025 р.

ПОЯСНЮВАЛЬНА ЗАПИСКА

кваліфікаційної роботи

галузь знань 12 Інформаційні технології
(шифр і назва галузі знань)
спеціальність 125 Кібербезпека
(код і назва спеціальності)
освітній ступень бакалавр
освітня програма Кібербезпека
(назва освітньо-професійної програми)
на тему: «Механізм виявлення вразливостей системного реєстру Windows методами машинного навчання»

Виконавець: студент IV курсу, групи КБ-43

_____ **Дмитро ОНАЦЬКИЙ**
(підпис) (ім'я, прізвище)

	Підпис	**Ім'я ПРІЗВИЩЕ**
Керівник		Юрій ЩЕБЛАНІН
Нормоконтроль		Яніна ШЕСТАК

Київ 2025

Міністерство освіти і науки України
«Київський національний університет імені Тараса Шевченка»

Факультет інформаційних технологій
Кафедра кібербезпеки та захисту інформації

ЗАТВЕРДЖЕНО:
В.о. завідувача кафедрою
кібербезпеки та захисту інформації
_____Іван ПАРХОМЕНКО
«29» листопада 2024 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи

спеціальності 125 Кібербезпека
(код і назва спеціальності)

освітньої програми Кібербезпека
(назва освітньої програми)

студентові **КБ-43** **Онацькому Дмитру Андрійовичу**
(група) (прізвище ім'я по-батькові)

Тема кваліфікаційної роботи Механізм виявлення вразливостей системного реєстру Windows методами машинного навчання

1. ПІДСТАВИ ДЛЯ ПРОВЕДЕННЯ РОБОТИ

Тема кваліфікаційної роботи затверджена на засіданні кафедри кібербезпеки та захисту інформації протокол №6 від 28.11.2024 р.

2. ВИХІДНІ ДАНІ ДЛЯ ПРОВЕДЕННЯ РОБІТ

Структура і функції системного реєстру Windows, методи машинного навчання

3. ЗМІСТ РОЗРАХУНКОВО-ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Нормативно-правова база у сфері захисту інформації, засоби і методи аналізу захищеності інформаційної системи, структура і функції реєстру операційної системи Windows, основні вразливості системного реєстру, методи аналізу реєстру, методи машинного навчання для аналізу реєстру, побудова архітектури механізму виявлення вразливостей, підвищення захисту операційної системи.

4. ВИМОГИ ДО РЕЗУЛЬТАТІВ ВИКОНАННЯ РОБОТИ

Практична цінність Підвищення захищеності реєстру ОС Windows

шляхом розробки механізму виявлення вразливостей методами ML

5. ДАТА ВИДАЧІ ЗАВДАННЯ

Дата видачі завдання: «29» листопада 2024 року

Завдання видав _____ (підпис) Юрій ЩЕБЛАНІН (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Дмитро ОНАЦЬКИЙ (ініціали, прізвище)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів робіт	Строки виконання робіт (початок-кінець)	Відмітка про виконання
1	Уточнення постановки задачі	18.11.2024 – 23.11.2024	*виконано*
2	Аналіз наукової літератури	24.11.2024 – 15.01.2025	*виконано*
3	Дослідження вразливостей ОС	16.01.2025 – 24.02.2025	*виконано*
4	Аналіз впливу вразливостей на стан безпеки ІС	25.02.2025 – 04.03.2025	*виконано*
5	Аналіз методів і засобів захищеності операційних систем	05.03.2025 – 24.03.2025	*виконано*
6	Дослідження структури і функцій реєстру ОС Windows і особливостей його даних	25.03.2025 – 07.04.2025	*виконано*
7	Розгляд інструментів аналізу реєстру і журналу подій	08.04.2025 – 20.04.2025	*виконано*
8	Аналіз можливостей методів ML для виявлення вразливостей	21.04.2025 – 05.05.2025	*виконано*
9	Розробка механізму виявлення і класифікації вразливостей системного реєстру методами ML	06.05.2025 – 26.05.2025	*виконано*
10	Оформлення пояснювальної записки	27.05.2025 – 03.06.2025	*виконано*
11	Підготовка до захисту	04.06.2025 – 15.06.2025	*виконано*

Завдання видав _____ (підпис) Юрій ЩЕБЛАНІН (ініціали, прізвище)

Завдання прийняв до виконання _____ (підпис) Дмитро ОНАЦЬКИЙ (ініціали, прізвище)

Термін подання кваліфікаційної роботи до ЕК « 13» червня 2025 року

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи складається зі вступу, трьох розділів, загальних висновків, списку використаних джерел та додатків: 80 сторінок, 15 рисунків, 13 таблиць, 4 додатків.

Мета кваліфікаційної роботи: підвищити захищеність реєстру операційної системи Windows шляхом використання методів машинного навчання в ході аналізу активних процесів.

Об'єкт дослідження: процес виявлення вразливостей реєстру операційної системи Windows з застосуванням методів машинного навчання.

Предметом дослідження є методи виявлення вразливостей реєстру операційної системи Windows.

Методи дослідження: аналіз літературних джерел, вивчення стандартів з кібербезпеки, структурування отриманих даних, системний підхід, оцінка методів захисту, методи порівняння.

Практичною цінністю отриманих результатів є підвищення захищеності реєстру операційної системи Windows за рахунок розробки механізму, виявлення вразливостей системного реєстру та класифікації виявлених вразливостей методами машинного навчання.

Пропозиції щодо продовження досліджень включають розширення функціональних можливостей механізму виявлення вразливостей системного реєстру методами машинного навчання за рахунок інтеграції з традиційними засобами аналізу захищеності системи.

Ключові слова: база даних вразливостей, вразливість, реєстр операційної системи, машинне навчання, Isolation Forest, Random Forest.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

API	–	Application Programming Interface
ІКС	–	Інформаційно-комунікаційні системи
ML	–	Машинне навчання
DL	–	Глибинне навчання
ІС	–	Інформаційна система
ОС	–	Операційна система
ПЗ	–	Програмне забезпечення
КЗЗ	–	Комплекс засобів захисту
НСД	–	Несанкціонований доступ
БД	–	База даних
SIEM	–	Security information and event management
СУБД	–	Система управління базами даних

ВСТУП

Сучасною об’єктивною реальністю є стрімкий розвиток інформаційних технологій, який супроводжується одночасним зростанням кількості кіберзагроз. Проблема захисту інформації стає однією з найактуальніших, зокрема через значне збільшення обсягів оброблюваних даних та вразливостей інформаційних ресурсів, що вимагає вдосконалення існуючих методів виявлення вразливостей та розробки нових підходів.

Зручними засобами своєчасного виявлення вразливостей в інформаційних системах є інструменти статичного та динамічного аналізу, які можуть ідентифікувати відомі вразливості. Однак, постійне збільшення кількості нових вразливостей складових інформаційно-комунікаційних систем (ІКС), які складно виявити традиційними методами та засобами, вимагає створення більш ефективних рішень для реалізації автоматизованого пошуку вразливостей з використанням таких сучасних технологій, як машинне навчання.

Пріоритетним є виявлення тих вразливостей, які з найбільшою вірогідністю будуть використані зловмисниками, щоб негативно вплинути на ІКС. Тому виникає необхідність розгляду нових методів і засобів захисту критично важливих для функціонування системи компонентів, одним із яких є системний реєстр операційної системи Windows.

Визначальним для правильної роботи системи, її збереження, а також забезпечення цілісності даних є своєчасне виявлення аномалій у роботі операційної системи (ОС), найпоширенішою з яких є ОС Windows. Для стабільного функціонування ОС і захисту програмного забезпечення (ПЗ) від несанкціонованого доступу критично важливим є системний реєстр операційної системи Windows. Він є централізованою базою даних, що містить налаштування для апаратного забезпечення, параметрів ОС, додатків і ПЗ, інформації, на яку постійно здійснюють посилання інші застосунки, тощо. У реєстрі зберігається уся інформація про налаштування безпеки, які контролюють доступ до ресурсів

системи. Неправильні налаштування можуть призвести до вразливостей, використання яких зловмисниками нанесуть збитки користувачам, тому виявлення таких вразливостей є важливим завданням для забезпечення безпеки системи. Величезний обсяг даних у реєстрі і динамічність записів реєстру, еволюція версій Windows і складність сучасних кібератак створюють значні проблеми для його аналізу і реагування на інциденти.

Для вирішення зазначеної проблеми виявлення вразливостей реєстру в роботі розглянуто методи та засоби захисту на основі аналізу реєстру у комплексі з новими напрямками безпеки, одним із яких є машинне навчання.

Актуальність теми полягає у необхідності вдосконалення та підвищення результативності системних розслідувань, пов'язаних з реєстром операційної системи Windows. Дослідження можливостей пошуку вразливостей на основі аналізу системного реєстру методами машинного навчання підвищить ефективність раннього виявлення та усунення помилок в процесі захисту операційної системи.

Об'єкт дослідження: процес виявлення вразливостей реєстру операційної системи Windows з застосуванням методів машинного навчання.

Предметом дослідження є методи виявлення вразливостей реєстру операційної системи Windows.

Мета кваліфікаційної роботи – підвищити захищеність реєстру операційної системи Windows шляхом використання методів машинного навчання в ході аналізу активних процесів.

Для досягнення зазначеної мети кваліфікаційної роботи поставлені наступні завдання:

- провести аналіз типів вразливостей та їхнього впливу на стан безпеки інформаційної системи;
- розглянути існуючі засоби аналізу захищеності інформаційних систем;
- дослідити структуру, особливості і методи аналізу реєстру операційної системи Windows;

- проаналізувати методи машинного навчання, алгоритми та архітектури, які використовують для побудови моделей для виявлення вразливостей системного реєстру;
- розробити механізм для створення моделі машинного навчання, яка буде здійснювати аналіз реєстру ОС і класифікувати виявлені вразливості за критеріями інформаційної безпеки.

При вирішенні поставлених завдань у кваліфікаційній роботі використані: методи аналізу (при розкритті основних особливостей даних реєстру), структурування отриманих даних, системний підхід, методи порівняння (при дослідженні методів і моделей машинного навчання).

Галузь застосування. Розроблений механізм може бути покладений в основу створення засобу виявлення вразливостей реєстру ОС Windows методами машинного навчання і використовуватися у галузі захисту інформації.

Новизна. Запропоновано сучасний механізм виявлення і класифікації вразливостей системного реєстру, який передбачає комплексну архітектуру структури виявлення вразливостей методами машинного навчання, що може значно підвищити точність та збільшити швидкість виявлення вразливостей.

Практичною цінністю є можливість підвищити захищеність реєстру операційної системи Windows за рахунок розробки механізму виявлення вразливостей системного реєстру та їхньої класифікації методами машинного навчання. Отримані результати можуть бути впроваджені в систему аудиту, що спростить завдання виявлення вразливостей системним адміністраторам.

Майбутнім покращенням у процесі практичної реалізації механізму може бути впровадження інтеграції з системою SIEM і традиційними засобами аналізу захищеності ОС.

Апробація роботи. Основні результати роботи представлені на конференції «XI International Conference. Information Technology and Implementation (Satellite)» від 21 листопада 2024 року.

РОЗДІЛ 1
ВРАЗЛИВОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ І СУЧАСНІ МЕТОДИ ЇХ ВИЯВЛЕННЯ

1.1 Типи вразливостей інформаційних систем та їх класифікація

Будь-якій інформаційній системі (ІС) властиві певні вразливості. Кількість їх постійно збільшується. За даними Microsoft щодо проаналізованих вразливостей за 2024 рік, загальна кількість вразливостей у 2024 році досягла рекордного показника 1360 [1].

Вразливості ІС обумовлені недоліками, що виникають у процесі функціонування кожної системи. Оскільки виникнення потенційних загроз безпеки пов’язано з наявністю вразливостей, їх виявлення та усунення стає одним із найважливіших елементів забезпечення безпеки інформаційних систем.

«Вразливість системи (system vulnerability) – нездатність системи протистояти реалізації певної загрози або сукупності загроз» [2, п. 4.2.11].

У [3] визначено, що «вразливість системи – властивість системи, через використання якої створюється загроза для її безпеки, порушується сталий, надійний та штатний режим функціонування системи, здійснюється несанкціоноване втручання в її роботу, створюється загроза для безпеки (захищеності) електронних інформаційних ресурсів, конфіденційності, цілісності, доступності таких ресурсів».

Успішна реалізація атаки на інформацію за допомогою вразливостей приводить до несанкціонованого розкриття, модифікації, пошкодження, знищення, недоступності або втрати інформації.

Основними причинами наявності вразливостей та виникнення шкідливих процесів функціонування інформаційних систем організацій є; властивості функціональних компонентів даної інформаційної системи; властивості впроваджених зловмисником функціональних компонентів в дану інформаційну

систему; переходи інформаційної системи в небезпечний стан внаслідок ненавмисних дій користувачів [4].

Відповідно до [5] вразливості різних типів виникають в інформаційних системах через недоліки політики безпеки; помилки адміністративного керування; недоліки алгоритмів захисту; помилки реалізації алгоритмів захисту.

Порівнюють вразливості на основі їхніх характеристик. Опис вразливості – це інформація, що розкриває її особливості. Вразливості поділяють на: приховані або невідомі вразливості (наприклад, 0-day вразливості, які ще не виявлені або не оприлюднені); відомі вразливості (вже ідентифіковані й описані); загальновідомі (оприлюднені у відкритих джерелах, зокрема в базах CVE), які активно використовуються зловмисниками [6].

Вразливість нульового дня можна також визначити як підмножину вразливостей, для яких не існує патчів або інших виправлень [7]. Через нерегулярність внесення виправлень загальновідомі, але невиправлені помилки, є одними з найпоширеніших типів вразливостей.

Стандарт ДСТУ ISO/IEC 15408-3 [8] поділяє вразливості за ступенем ризику: вразливості з високим рівнем ризику (зловмисник може отримати безпосередній доступ до вузла в обхід міжмережних екранів або інших засобів захисту); вразливості із середнім рівнем ризику (зловмисник може заволодіти інформацією, яка вірогідно дозволить отримати доступ до вузла); вразливості з низьким рівнем ризику (експлуатація вразливості не завдає значної шкоди).

У дослідженні [9] поширеними типами вразливостей ОС, які можуть бути результатом помилок програмування, застарілого ПЗ або небезпечної інтеграції ПЗ, визнанні:

- переповнення буферу (відбувається, коли програма записує в буфер більше даних, ніж може вмістити, що потенційно дозволяє зловмисникам перезаписувати області пам'яті);
- помилки підвищення привілеїв (експлойт, що дозволяє користувачеві отримати більш високий рівень доступу, ніж передбачалося, що призводить до несанкціонованого контролю за функціями системи);

- невиправлені вразливості ПЗ (вразливості, що існують у застарілому ПЗ або системах, до яких не було застосовано виправлення безпеки);
- вразливості впровадження (вони виникають, коли зловмисник може впровадити шкідливий код у програму, що часто спостерігається при впровадженні SQL, NoSQL, ОС та LDAP);
- експлойти нульового дня (це вразливості, які використовуються зловмисниками до того, як постачальник програмного забезпечення випустить виправлення або навіть дізнається про вразливість).

За результатами проведеного аналізу інформації, викладеної у [5], [9], [6], [10] стосовно існуючих класифікацій вразливостей, сформована табл. А.1.1, у якій узагальнено різні чинники класифікації і відповідні їм види вразливостей.

Оскільки вразливості інформаційних систем і причини їх виникнення надзвичайно різноманітні, потенційна небезпека для системи різна, необхідно здійснювати оцінку вразливостей і для нейтралізації їхнього впливу вживати ефективні методи і засоби. З цією метою створені бази даних вразливостей.

База даних (БД) вразливостей – ресурс, який призначений для збору, зберігання і поширення інформації про вразливості, що впливають на комп’ютерну безпеку. База даних містить опис вразливості, надає оцінку можливого впливу на систему і шляхи, необхідні для зменшення проблеми [11].

Характеристики деяких проаналізованих БД вразливостей ([12], [13], [14], [15]), з метою їх порівняння за функціональними можливостями для визначення більш відповідних для застосування, наведені у табл. А.1.2.

На основі проведеного дослідження зроблено висновок, що доцільним у якості зручного і достовірного джерела є застосування бази даних NVD NIST, оскільки вона інтегрована з БД CVE, і у разі появи у цій базі даних нової вразливості, експерти проводять її аналіз і публікують детальну характеристику вразливості, включаючи види найбільш вірогідних атак.

Для оцінки критичності вразливості існують системи і методи, які створені різними організаціями, наприклад: система аналізу CERT/CC враховує передумови, необхідні для експлуатації вразливості; система аналізу

вразливостей SANS надає критичний рівень вразливостям, якщо для них існує загальнодоступна програма, яка використовує ці вразливості; система оцінки від Microsoft застосовує методику оцінки ризику, який пов'язаний із шкідливим ПЗ).

Відкритим стандартом для визначення оцінки рівня вразливості є CVSS **(**Common Vulnerability Scoring System), який містить класифікацію вразливостей за шкалою критичності від 0 до 10, що показують рівень впливу вразливості на критерії безпеки. CVSS використовує кілька метрик: базову, часову та контекстну, показники яких впливають на рейтинг вразливості [16].

Базові показники характеризують наскільки легко зловмисник може експлуатувати уразливість з метою порушення конфіденційності, цілісності та доступності інформації. Часові метрики додають поправки стосовно характеристик, які змінюються протягом життєвого циклу вразливості. Наприклад, зрілість коду експлойтів, якщо вони існують, і доступність виправлень. Контекстні метрики враховують характеристики вразливості, які залежать від конкретної реалізації або середовища.

Бази даних вразливостей надають засоби для аналізу вразливостей у комп'ютерних мережах. Їх поповнюють при виявленні нових вразливостей, що полегшує пошук і усунення вразливостей інформаційної системи фахівцями, а також створює величезний потенціал для зловмисників і хакерів.

1.2 Аналіз впливу вразливостей на стан безпеки інформаційних систем

Аналіз вразливостей – це процес, який спрямований на пошук будь-яких загроз, вразливих місць і ризиків потенційного несанкціонованого проникнення зловмисників у інформаційну систему.

Виявлення вразливостей – це процес визначення слабкостей у системі, які можуть бути використані зловмисниками для порушення її безпеки.

Оцінка вразливості включає розгляд усіх загроз, які можуть використовувати її у певних ситуаціях. Відповідно до [2] «загроза (threat) – будь-

які обставини чи події, що можуть спричинити порушення політики безпеки інформації та (або) нанесення збитку ІКС». Останнім часом однією з основних причин таких збитків стало зараження інформаційних систем через використання зловмисниками вразливостей [17].

У кожної загрози є певний перелік вразливостей, за допомогою яких зловмисник може здійснити свої плани. Під час реалізації загрози зловмисник може використати одну або декілька вразливостей. Джерела загрози можуть виходити як зсередини, так і ззовні організації.

Під загрозами конфіденційності розуміють неправомірний доступ до інформації; загрози доступності – це здійснення дій, що унеможливлюють чи затрудняють доступ до ресурсів інформаційної системи; загрози цілісності – це неправомірна зміна даних. До реалізації загрози призводить атака. «Атака (attack) – це спроба реалізації загрози» [2, п.4.2.9].

Джерела загроз викликають ризик інформаційної безпеки. «Ризик (risk) – функція ймовірності реалізації певної загрози, виду і величини завданих збитків» [2, п.4.3.3]. Загроза використовує уразливість системи, щоб завдати шкоди компонентам системи.

Дія зловмисника спрямована на використання певної вразливості з метою впливу на інформаційний об'єкт та комплексний захист ПЗ, який теж може мати вразливості. Вплив на властивості інформаційного об’єкта робить можливим виникнення загрози і здійснення зловмисником її реалізації з метою завдання шкоди і нанесення збитків власнику (рис. 1.1).

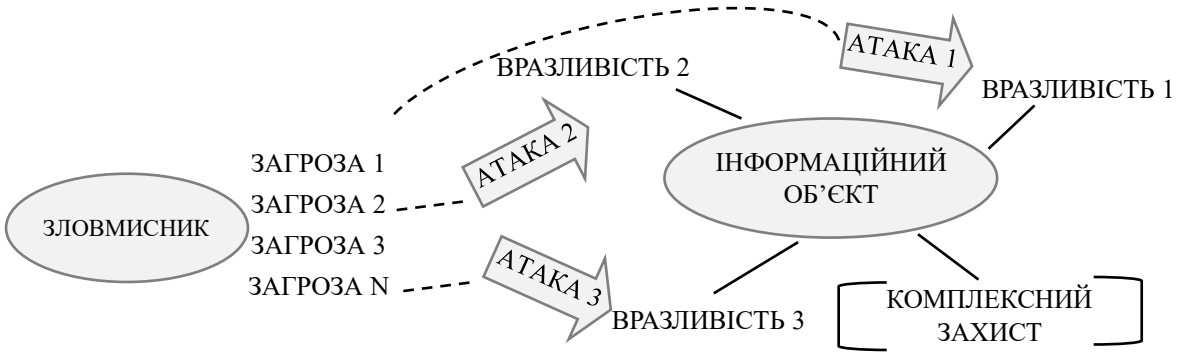

Рисунок 1.1 – Взаємодія загроз та вразливостей

Схема взаємодії між загрозами, вразливостями і ризиком показує, що реалізація загрози безпеці ІС прямо залежать від дій зловмисника з виявлення і використання вразливостей, що наявні у системі, тобто, якщо відсутня вразливість, то неможлива й атака, що її використовує; натомість головною протидією зловмисникові є процес аналізу і виявлення вразливостей, здійснюваний фахівцями з кібербезпеки (рис. А.1.2).

Вразливості можуть впливати на стан безпеки ІС на різних її рівнях, і кожному з рівнів співвідносять різні типи вразливостей [5]:

– до рівня мережі, який відповідає за взаємодію вузлів ІС, відносять вразливості мережних протоколів – стека TCP/IP, протоколів NetBEUI, IPX / SPX;

– рівень ОС що відповідає за обслуговування системи управління базами даних (СУБД) та прикладного ПЗ охоплює вразливості конкретної операційної системи Windows, UNIX, Novell;

– рівень СУБД, що відповідає за зберігання та обробку даних ІС, охоплює вразливості конкретних СУБД – Oracle, MSSQL, Sybase;

– до рівня прикладного ПЗ, що відповідає за взаємодію з користувачем, відносять вразливості програмного забезпечення WEB, SMTP серверів і т. п.

Вразливості системного ПЗ та вразливості прикладного програмного забезпечення різняться за можливостями впливу на систему.

Призначенням системного ПЗ є управління роботою компонентів комп’ютера та взаємодія між ними, виявлення та усунення недоліків у роботі комп’ютера, забезпечення роботи інших програм, підтримки взаємодії з користувачами, надання їм консультацій щодо експлуатації комп’ютера.

Керування ресурсами (фізичними та логічними) і процесами обчислювальних систем є функцією операційної системи. Вразливості ОС автоматично спричиняють кардинальні недоліки у захисті всієї інформаційної системи. У [10] виокремлені підсистеми і можливі їх вразливості:

– засоби ініціалізації (завантаження) системи можуть спричинити некоректну роботу елементів системи (наприклад, виявитись недоступними у зв’язку з некоректним наданням повноважень доступу до цих ресурсів);

– підсистеми керування процесами, пам'яттю та пристроями, підсистема керування файловою системою створюють основу для розмежування доступу в системі (використання помилок в цих підсистемах зловмисником може викликати перевищення повноважень і несанкціонований доступ до елементів системи або отримання повного доступу до інформації);

– підсистема ідентифікації й автентифікації є основою роботи інших підсистем захисту (помилки можуть стати причиною некоректного виконання процедур ідентифікації й автентифікації, розкриття ідентифікаторів і паролів).

До прикладного програмного забезпечення відносяться прикладні програми загального користування та спеціальні прикладні програми, які використовують користувачі. В прикладному ПЗ виникнення вразливостей не може завдати значної шкоди роботі системного ПЗ. Такі вразливості призводять до збою процесу, який містить ці помилки [6]. Вразливостями у прикладному програмному забезпеченні можуть бути:

– функції навмисно впроваджених шкідливих програм, які використовують відомі вразливості програмного забезпечення, що дозволяє їм діяти з більшими повноваженнями в порівнянні з повноваженнями прикладного процесу та користувача, який його ініціював;

– частини коду програм («дірки», «люки»), які були зроблені розробником, що дають можливість обходити процеси ідентифікації, автентифікації, перевірки цілісності та ін., передбачені в операційній системі;

– відсутність необхідних засобів захисту (автентифікації, перевірки цілісності, перевірки форматів повідомлень, блокування функцій, які модифіковані без дозволу, тощо).

Причиною широкого застосування зловмисниками вразливостей є те, що проведення таких атак зазвичай вимагає небагато ресурсів, їх можна автоматизувати та масштабувати для охоплення найрізноманітніших ІС, вони можуть завдати дуже значних фінансових або репутаційних збитків підприємству.

1.3 Огляд існуючих методів і засобів аналізу захищеності ІС

Стандарти і нормативні документи України, міжнародні стандарти містять вимоги до системи управління інформаційною безпекою щодо забезпечення збереження конфіденційності, цілісності й доступності інформації і вимагають проведення регулярних перевірок інформаційних систем на наявність вразливостей для запобігання ризиків.

Стандарт ISO/IEC 27001 приділяє значну увагу питанням превентивного виявлення та усунення вразливостей. Для впровадження найбільш ефективних заходів захисту необхідними є: достатня інформація щодо вразливостей і загроз, викликаних ними; аналіз ризиків. У пункті А.8.8 [18] підкреслено, що управління вразливістю є оптимальним захистом інфраструктури.

Захист інформації від загроз і підтримка стану інформаційної безпеки в організації здійснюються у порядку, який регламентує політика безпеки інформації. Реалізують політику безпеки інформації за допомогою комплексу засобів захисту (КЗЗ). Підсистеми КЗЗ ОС виконують певні функції захисту (табл. А.1.3). Вони мають бути реалізовані відповідно до декларованої політики безпеки і вимог гарантій [19].

КЗЗ ОС є сукупністю значної множини програмних модулів, серед яких частина реалізується у режимі ядра, інша частина – у режимі користувача, як прикладні програми, які можуть мати вразливості. Тому актуальним завданням для своєчасної нейтралізації загроз безпеки інформації є пошук ефективних рішень і засобів виявлення тих неприпустимих подій у процесах функціонування інформаційної системи, які можуть бути результатом її вразливостей.

Необхідним елементом системи безпеки комп’ютерної мережі є аналіз її захищеності. Системи аналізу захищеності є частиною адаптивної безпеки комп'ютерної мережі, яка включає в себе: аналіз захищеності (Security Assessment) і виявлення атак (Intrusion Detection) [5].

Засоби аналізу захищеності тестують мережу або хост і можуть вже на початковому етапі здійснення атаки виявляти і своєчасно усувати вразливості.

Вони запобігають самій можливості реалізації атаки, надаючи додаткову інформацію для систем виявлення вторгнень.

Після того, як система аналізу захищеності виявляє вразливості, які використовують зловмисники, системи виявлення атак в процесі реалізації виявляють атаку в режимі реального часу, використовуючи обманні системи, які імітують важливі для зловмисника ресурси, і в момент атаки збирають інформацію про нападника з метою його викриття.

Системи, що функціонують у процесі наступної фази атаки, шукають зміни контрольних пакетів і аналізують реєстраційні журнали.

Таким чином, адаптивний підхід до різних етапів атаки дозволяє розпізнавати загрозу ще в процесі її підготовки і формування, а не під час її реалізації. Пошук вразливостей дозволяє оцінити можливість отримання зловмисником НСД до системи, зрозуміти можливості компрометації її зловмисниками і здійснення зловмисних дій.

Однією з технологій аналізу захищеності інформації є проведення аудиту безпеки, який передбачає проведення детальної перевірки існуючих механізмів захисту та виявлення можливих недоліків. У табл. А.1.4 наведені основні методи, які застосовують при проведенні аудиту.

Важливою складовою засобів виявлення вразливостей є підготовка звітів, що дозволить не тільки виявляти вразливості, а і надавати системним адміністраторам пропозиції щодо практичних шляхів з їх усунення. Результати аудиту формують базу для ідентифікації та оцінки ризиків.

Виявлення вразливостей в ОС Windows є ключовим етапом у забезпеченні інформаційної безпеки. Основними методами пошуку вразливостей є:

– статистичний аналіз, який полягає в аналізі інформації щодо конфігурацій, системних налаштувань, файлів і реєстру без виконання програм. Його використовують для перевірки: невірних налаштувань політик безпеки; відкритих портів і служб; слабких паролів або облікових записів без паролів; наявності старих або вразливих версій програмного забезпечення;

– динамічний аналіз, який передбачає спостереження за поведінкою системи під час її роботи: аналіз журналів подій (Event Viewer); відстеження змін у файловій системі та реєстрі; моніторинг запущених процесів та мережевої активності; виявлення аномалій або підозрілих дій у реальному часі.

Механізмом статистичного аналізу є сканування. Програмний засіб сканує систему на предмет відомих вразливостей і виявляє потенційні вразливості за непрямими свідченнями без фактичного підтвердження. Метод характеризується високою швидкістю та простотою реалізації.

Засобом динамічного аналізу є зондування або імітація атак. Він забезпечує виявлення вразливостей на хості, що перевіряється, і передбачає симуляцію атак на систему, тобто експлуатування вразливостей для проникнення в систему і пошук додаткових слабких місць з метою виявлення вразливостей: експлойтів, недостатнього налаштування або інших проблем безпеки [20].

Аналіз коду доповнює автоматизовані методи аналізу і передбачає здійснення перевірки з використанням спеціальної програми дизасемблера і переведення його в код асемблера для простішого сприйняття людиною [21]. Перегляд коду людиною дозволяє виявити вразливості, які, через їхню складність або специфіку контексту, можуть бути не зафіксовані іншими інструментами. Наприклад, недостатній контроль доступу, неправильне управління сесіями, вразливості, які ще не включені до БД автоматизованих інструментів, можуть бути виявлені тільки за допомогою аудиту коду, тому він є важливим у забезпеченні безпеки інформаційних систем.

Можливим є спостереження за внутрішніми взаємодіями програмного забезпечення з одночасним аналізом поведінки зовнішніх запитів [22]. Кожен з цих методів має свої особливі функції, тому їх використання є необхідним для забезпечення повноцінного захисту.

Сканування на основі методу сигнатур передбачає пошук сигнатури – відомої характерної ознаки присутності того чи іншого шкідливого ПЗ. База сигнатур постійно оновлюються, щоб включати нові загрози. Сканування виконується досить швидко, оскільки полягає у порівнянні ланцюжків байтів із

бібліотеки сканера з ланцюжками байтів, отриманих від ресурсу, проте передбачає лише порівняння без перевірки правильності припущень [23].

Евристичний аналіз – це метод, який вивчає дії та поведінку файлів, ПЗ або процесів для виявлення будь-яких відхилень або потенційних показників, які можуть вказувати на наявність загрози. Теоретично він здатен знайти навіть загрозу нульового дня, проте витрачає набагато більше ресурсів, ніж сканування на основі сигнатур; потрібує більше часу, щоб знайти результати [24].

Фазинг – автоматизоване тестування ПЗ, яке використовує багатоетапну перевірку з різними інструментами на різних стадіях життєвого циклу системи для ідентифікації та нейтралізації вразливостей шляхом відправлення неочікуваних, випадкових або недійсних даних ПЗ на основі спостереження за тим, як ПЗ поводиться та реагує на введення [25].

Розглянуті методи є необхідними механізмами виявлення вразливостей у складі інформаційних систем для оптимізації та підвищення їх захищеності.

Пошук вразливостей, ідентифікацію і оцінку вразливостей ІС, формування звітів і усунення або нейтралізацію вразливостей здійснюють засоби аналізу захищеності, які поступово досліджують інформаційну систему на усіх її рівнях.

Актуальність застосування засобів аналізу захищеності виникає з можливості завчасного визначення значного переліку типів вразливостей, що присутні в інформаційній системі. Це має сприяти застосуванню необхідних заходів, що забезпечать мінімізацію або усунення можливості використання виявлених вразливостей зловмисником.

Сканери вразливостей – це інструменти (програмні або апаратні), які призначені для здійснення діагностики та автоматизованого сканування різних елементів мережі, комп'ютерів та програмного забезпечення з метою виявлення, аналізу та усунення потенційних вразливостей у системі інформаційної безпеки.

Сканери поділяють за типами:

– мережеві сканери, які здійснюють дистанційні перевірки станів контрольованих хостів на мережевому рівні та ідентифікують уразливості, які вимагають негайного реагування;

– сканери рівня вузла, які встановлюють безпосередньо на сканований вузол з метою виконання локального аналізу станів контрольованих хостів. Їхніми особливими рисами є більш ретельний пошук вразливостей і більша вірогідність результату, оскільки вони працюють на сканованому вузлі і використовують обліковий запис з максимальними привілеями (root, SYSTEM). Вони можуть здійснювати перевірки, які неможливо або важко виконати мережевими сканерами [11].

Основним завданням сканерів вразливостей є оцінка безпеки процесів, тестування та аналіз систем і служб з метою ідентифікації відомих вразливостей відповідно до встановлених стандартів БД вразливостей, наприклад, CVE або NVD, для порівняння з поточним станом системи. Засіб аналізу вразливостей забезпечує сканування мережевих протоколів та програмних компонентів з метою виявлення стороннього проникнення в систему. Результатом перевірки є звіт, який містить пріоритетний перелік виявлених вразливостей, їх детальний опис, оцінку ризику, а також заходи щодо усунення або мінімізації впливу загроз.

Сканери безпеки представлені різними моделями, більшість з яких орієнтована на пошук вразливостей у конкретній області – сканери портів, сканери топології комп’ютерної мережі, сканери вразливостей мережевих сервісів. Засобами сканування є, наприклад, сканери: Nessus, OpenVAS, Qualys.

Nessus/NeWT – ПЗ, яке здатне виявити найпоширеніші типи вразливостей, зокрема ідентифікувати застарілі або вразливі версії служб або доменів, а також помилки в конфігурації (наприклад, відсутність механізмів автентифікації на SMTP-сервері). Такий аналіз дозволяє своєчасно виявляти критичні недоліки в налаштуваннях системи безпеки. Nessus для ОС Unix і NeWT (Nessus on Windows Technology) аналог для платформи Windows [26].

OpenVAS – потужний інструмент для виявлення вразливостей, робота якого базується на колекції NVT тестів безпеки і базі вразливостей CVE і підтримує регулярні оновлення бази даних вразливостей [27].

Qualys – сервіс, який використовують для виявлення і кількісної оцінки вразливості безпеки ПЗ, моніторингу відповідності та управління активами [28].

Кожен звіт, згенерований сканером, потребує проведення аналізу експертом з кібербезпеки, що дозволить уникнути помилкових висновків. Проблемою сканування є те, що інструменти аналізу захищеності ОС спрямовані насамперед на перевірку параметрів конфігурації, характерних для конкретного сімейства однієї ОС без урахування змін, які вносить у ОС кожний виробник, і це ускладнює аналіз специфічних параметрів. Пошук вразливостей в ОС також став важливою необхідністю, оскільки складність ПЗ постійно зростає. Сучасні ОС володіють тисячами двійкових файлів, проте інструментів, що здатні тестувати програми у масштабах ОС за короткий проміжок часу відсутні.

У загальному випадку процес автоматизованого пошуку вразливостей зводиться до повного перебору вразливостей, відомості про які присутні у базі даних сканера. Це кропіткий процес, тому створення інших ефективних засобів автоматизованого пошуку вразливостей з використанням таких сучасних технологій, як машинне навчання, є актуальним завданням.

Практичним застосуванням технік, що імітують атаки на систему для виявлення вразливих точок, є популярний інструмент виконання тесту на проникнення Metasploit Project. Він створений для надання інформації про вразливість, допомоги у створенні сигнатур для IDS та тестування експлойтів. Найбільш відомий Metasploit Framework, який застосовують для розробки та виконання експлойтів і використовують для тестування вразливостей та автоматизації атак [29].

Перевагою тестування на проникнення є його здатність моделювати найгірший можливий сценарій, оскільки тестування імітує атаку кваліфікованого зловмисника, який експлуатує вразливості для проникнення в систему і виявлення слабких місць без завдання шкоди системі. Автоматичне сканування, навпаки, просто повідомляє про знайдені проблеми без їх подальшої перевірки.

Однак, враховуючи те, що будь-яка вразливість у ОС може поставити під загрозу безпеку додатків, і те, що ОС є важкими для фазингу через складність компонентів і проблеми із продуктивністю після збоїв під час тестування, сканування є дуже важливим.

Узагальнена схема виявлення вразливостей ІС, ідентифікації і оцінки, наведена на рис. А.1.4.

Згідно з [3] на основі результатів аналізу звіту власник ІС здійснює оцінку можливих наслідків використання вразливості системи для цілісності, конфіденційності та доступності даних. Якщо загрози є критичними для захищеності ІС, приймають рішення про внесення відповідних змін до системи або її конфігурації, і повідомляють CERT UA або галузеву команду реагування.

Процес оцінки вразливості завершується розробкою рекомендацій з її усунення або виключення можливості її використання. На вперше виявлену вразливість формують відповідний паспорт і після виправлення недоліків, вразливість отримує статус відомої вразливості. Послідовність кроків життєвого циклу вразливості (рис. А.1.5) свідчить про важливість проведення ретельних перевірок і аудитів безпеки з метою виявлення потенційних вразливостей і наявності відомих вразливостей для перевірки захисту системи.

Основними цілями для зловмисників є ОС Microsoft Windows, які є найпоширенішими операційними системами (75% використовуваних настільних комп’ютерів). Серед настільної версії найбільшою є частка Win10 і Win11 - відповідно 61,82 % і 34,94 % станом на листопад 2024 року. Тому у роботі буде розглянуто реєстр Win10 [30].

Операційні системи є найбільш схильними до вразливостей, оскільки в них акумулюється основна частина механізмів захисту: засоби розмежування доступу до ресурсів, аутентифікація користувачів, аудит подій та ін., що робить їх одним із головних елементів інформаційної безпеки, і, одночасно, основними цілями зловмисників для використання відомих і невідомих вразливостей, щоб отримати НСД або здійснити інші порушення.

Традиційні засоби перевірки для виявлення помилок через складність і великий об’єм даних у ОС не гарантують, що системи ОС будуть повністю вільні від помилок, які можуть використати зловмисники. За таких обставин особливої уваги потребує ідентифікація, оцінка та управління ризиками, які пов’язані із використанням зловмисниками локальних вразливостей, присутність яких може

бути використана для встановлення зловмисного ПЗ. Важливим джерелом даних для проведення аудиту є системний реєстр, оскільки ПЗ, яке використовують зловмисники, у багатьох випадках залишають у реєстрі відбитки, тож зібрана в реєстрі інформація може довести наявність вразливостей і активність шкідливого ПЗ. Щоб визначити критично важливі області реєстру для виявлення вразливостей і атак, необхідно провести аналіз його структури і функцій.

1.4 Структура і функції реєстру операційної системи Windows

Реєстр ОС Windows є ієрархічною базою даних, у якій зберігається інформація про налаштування системи. Він містить інформацію щодо: параметрів пристроїв, що встановлені в комп'ютері; з'єднань віддаленого доступу; профілів кожного користувача; інформації про зв'язки файлів з програмами та типів документів, які кожна з них може створювати; параметрів властивостей папок та значків додатків; використовуваних портів [31].

Windows організовує реєстр у вигляді кущів, які мають логічно структуровану деревоподібну ієрархію розділів, підрозділів і параметрів. Структура має багаторівневу організацію [31].

Кореневі розділи найвищого рівня: два основні розділи (HKEY_LOCAL_MACHINE, HKEY_USERS) і три додаткові розділи (HKEY_CLASSES_ROOT, HKEY_CURRENT_USER, HKEY_CURRENT_CONFIG), розташовані на першому рівні. Кожний кореневий розділ відповідає певному типу інформації (табл. А.1.5). Взаємозв'язок, що існує між кореневими розділами реєстру наведено на рис. А.1.6 [32].

Ключі реєстру (Registry Keys), які розташовані на другому рівні, поділяються на дві категорії: перша, що визначається системою (зміна імен ключів може зробити Windows повністю непрацездатною); друга, що визначається користувачем (імена ключів може змінювати адміністратор комп'ютера, що не матиме фатальних наслідків для системи).

На третьому рівні розташовані підрозділи реєстру (Subkeys), які формують файли (містять певні значення конфігурації системи / програми). Компоненти реєстру Параметри (Values) розташовані на четвертому рівні (містять фактичну інформацію, збережену у реєстрі, яка зумовлює роботу ОС і всього комп'ютера).

Кожне із дерев системного реєстру у розділі «My Computer» є ключем. Максимальний розмір імені ключа складає 255 символів.

Файли системного реєстру Windows фізично зберігаються у різних підкаталогах системного каталогу (табл. А.1.6) [33]. Кожен системний розділ реєстру має відповідний файл, резервні копії яких розміщені за тим же шляхом).

Реєстр підтримує різні типи даних для параметрів, які він містить (табл. А.1.7) та інші, менш поширені типи. Параметр реєстру має ім'я, тип даних і значення.

Основними функціями, які необхідні для роботи з реєстром, є [34]:

- RegCreateKeyEx()– створення ключів
- RegOpenKeyEx() – відкриває вказаний розділ реєстру;
- RegCloseKey() – закриває дескриптор вказаного розділу реєстру;
- RegFlushKey() – збереження змін;
- RegDeleteValue() – видаляє іменоване значення із зазначеного розділу реєстру;
- RegSetValueEx() – встановлює дані та тип зазначеного значення у розділі реєстру (після генерації ключа, виникає потреба внести до ключа додаткові дані, які використовуються програмою);
- RegQueryInfoKey() – витягує інформацію про вказаний розділ реєстру (за допомогою цієї функції можна отримати повну інформацію про підключі).

Функції, які може виконувати Реєстр для безпечної роботи системи, дозволяють вирішувати проблеми, що виникають у процесі роботи програм, та ефективно налаштовувати режими їхньої роботи, а також усувати несправності в роботі обладнання, пов’язані з некоректним застосуванням ресурсів ОС та перерозподіляти ресурси ОС так, як це потрібно адміністратору.

Реєстр Windows є головним елементом, який можна використати для відстеження змін і виявлення зловмисних дій у реєстрі. Але об'єм даних, що містить реєстр, становить серйозну проблему для аналізу, особливо у режимі реального часу. Аналіз усього реєстру вручну потребує багато часу і ресурсів, що створює проблеми для ефективного вилучення відповідних даних з реєстру.

Вирішення таких проблем потребує нових стратегій захисту і викликає необхідність використання машинного навчання (ML), щоб процеси аналізу реєстру і відстеження кіберзагроз, що можуть виникнути стали ефективнішими.

1.5 Можливості використання методів машинного навчання для аналізу реєстру і управління вразливостями

Технології ML успішно застосовують у вирішенні завдань захисту даних.

Методи ML працюють в одному з таких режимів**:** навчання з учителем, навчання з частковим залученням вчителя, навчання без вчителя, навчання з підкріпленням.

Навчання з учителем потребує навчальної вибірки, яка повною мірою представляє особливості системи і включає приклади даних як нормального, так і аномального класів. Застосовують навчання з учителем, зокрема для вирішення типових задач класифікації і регресії [35].

Алгоритми ML з вчителем класифікують записи реєстру, наприклад, на безпечні та шкідливі, нормальні та аномальні. Метод класифікації може, зокрема, значно спростити виявлення загроз конфіденційності даних, відокремивши користувачів, що ідентифікувались, від анонімних і неідентифікованих.

Моделі регресійного аналізу застосовують для демонстрації або прогнозування залежності між певним процесом та його потенційними наслідками. Прикладами використання методів регресійного аналізу є: прогнозування; аналітика дієвості політики безпеки; аналітика ризиків [36].

Навчання без учителя використовують, коли набір достовірних даних недоступний. Цей метод вирішує, зокрема задачі кластеризації для виявлення

закономірностей у даних [37]. Це особливо корисно для виявлення невідомих патернів у даних.

Алгоритми ML без вчителя можуть виявляти аномалії, атаки нульового дня, виконувати дослідження даних у кібербезпеці, зокрема групувати записи реєстру за схожістю, що дозволяє виявити неочевидні зв'язки між різними параметрами.

Навчання з частковим залученням вчителя поєднує в собі ознаки навчання з вчителем та навчання без вчителя. Модель, навчена на прикладах одного класу, здатна ідентифікувати нові об’єкти як такі, що належать до цього класу або відхиляються від нього, фактично визначаючи належність до аномального або протилежного класу. Алгоритми, що функціонують у режимі розпізнавання частково з учителем, не потребують попередньої інформації про аномальний клас примірників. Це розширює сферу їхнього застосування та дозволяє розпізнавати відхилення за відсутності заздалегідь визначених відомостей про них [38].

Для прогнозування поведінки системи, яка ґрунтується на значних обсягах історичних даних або для виявлення складних патернів використовують нейронні мережі. У переважній більшості нейромереж застосовують контрольоване навчання, в процесі якого фактичний вихід постійно порівнюється з цільовим значенням для коригування моделі [39].

Можливості використання методів ML для аналізу реєстру і виявлення вразливостей спираються на властивості технології ML і полягають у тому, що машинне навчання може вирішувати певні завдання в аналізі даних системного реєстру Windows і застосовуватися процесах:

– виявлення аномалій у системному реєстрі, таких як незвичайні зміни параметрів, які можуть вказувати на шкідливу діяльність або збої в системі. Моделі ML можуть бути навчені на нормальних шаблонах поведінки реєстру і виявляти відхилення від цього шаблону;

– аналізу поведінки програм, які встановлені в системі, за допомогою алгоритмів ML, що дозволить автоматично класифікувати програми та оцінити їх вплив на систему або навіть передбачити їхню майбутню поведінку,

наприклад, можна прогнозувати ймовірність того, що певна програма може бути шкідливою на основі історичних даних;

– оптимізації продуктивності операційної системи, наприклад, використовувати моделі для виявлення неефективних параметрів реєстру, які можуть уповільнювати роботу системи, та рекомендувати коригування.

Таким чином, використання моделі ML дозволить підвищити захищеність реєстру ОС Windows.

Висновок за розділом 1

У першому розділі розглянуто основні типи вразливостей і їхній вплив на стан безпеки ІС на різних її рівнях і визначено, що найбільш часто зловмисники здійснюють атаки на ОС, оскільки в них зібрані основні механізми захисту, і здійснення атаки на них може завдати найбільшої шкоди. Важливим інструментом реалізації політики безпеки є аналіз захищеності ІС.

Методами аналізу захищеності є сканування (забезпечує регулярний моніторинг і виявлення відомих загроз) і зондування (допомагає виявити нові та складні загрози і оцінити реальні сценарії атак). Були розглянуті існуючі засоби аналізу захищеності інформаційних систем.

У ході дослідження визначено, що системний реєстр є найважливішою частиною Windows для функціонування операційної системи, оскільки кожна програма, що працює в ОС, обов’язково звертається до реєстру під час роботи. Тому виявлення вразливостей реєстру ОС має надзвичайне значення для підтримки цілісності системи, управляння конфігураціями обладнання і ПЗ, забезпечення стабільної роботи системи. Встановлено, що аналіз реєстру ОС викликає труднощі через великий обсяг даних, що зумовлює проблеми для ефективного вилучення даних реєстру при виконанні ручного аналізу.

Важливо використовувати сучасні стратегії аналізу реєстру, які передбачають впровадження методів машинного навчання, що дозволить підвищити захищеність реєстру операційної системи Windows.

РОЗДІЛ 2
МЕТОДИ АНАЛІЗУ ТА ОЦІНКИ ЗАХИЩЕНОСТІ РЕЄСТРУ ОПЕРАЦІЙНОЇ СИСТЕМИ WINDOWS

2.1 Особливості даних системного реєстру

Аналіз реєстру – це процес, що включає перевірку реєстру Windows для виявлення та аналізу потенційних загроз безпеки. Вибір методів аналізу реєстру вимагає проведення дослідження його структури і функцій для визначення критичних розділів реєстру.

Реєстр є важливим джерелом інформації для проведення аналізу, оскільки лані, які містить реєстр, надають уявлення щодо системних дій, дій користувача і потенційних порушеннях безпеки:

- реєстр зберігає значну кількість складних, необроблених даних про конфігурації, які дуже важливі для системи, і які вплинуть на стабільність Windows, якщо зникнуть або будуть неправильно налаштовані;
- користувачі з правами адміністратора мають практично повний контроль для редагування ключів реєстру;
- деякі несанкціоновані програми для свого автоматичного запуску в процесі завантаження ОС використовують різні елементи реєстру. Несанкціоновані зміни неавторизованими користувачами або програмами, можуть змінити поведінку або безпеку Windows і додатків (наприклад, вимкнути певні функції або можливості системи, розкрити конфіденційну інформацію і облікові дані, що зберігаються у реєстрі, або надати підвищені привілеї іншим користувачам чи програмам);
- реєстр Windows є також базою даних, яку можуть використовувати розробники різних типів шкідливих програм для їх поширення по всій ОС. Шкідливі програми виконують зловмисні дії у фоновому режимі, візуально залишаючись прихованими, і створюють певні вразливості, уникаючи виявлення

традиційним антивірусним ПЗ, тобто закріплюються у системі, що надає їм змогу зберігати контроль і продовжувати зловмисну діяльність;

- впроваджена вірусна програма з правами адміністратора може стати причиною модифікації реєстру, що негативно вплине на стабільність системи. Вона може використовувати власні інструменти Windows для виконання своїх команд, тому її неможливо виявити за допомогою сигнатурного ПЗ безпеки, наприклад, антивірусу;
- до неможливості нормальної роботи системи можуть призвести пошкодження реєстру (збої живлення, помилки диску, помилки користувача). Повна непрацездатність комп'ютера може стати наслідком дій зловмисника, який видалить файли реєстру. Він також може скопіювати інформацію, що дасть йому можливість підібрати паролі до облікових записів користувачів.

Особливостями функціонування реєстру, важливими для процесів виявлення підозрілої активності, є: отримання доступу до певного набору ключів реєстру під час звичайного виконання більшістю програм Windows; використання більшістю користувачів певного набору програм під час запуску своїх машин; регулярність діяльності реєстру за часом.

Кущі розділів реєстру є єдиною структурою, що містить значення часу Last Write (час останнього запису). Ці дані особливо корисні для аналізу реєстру, оскільки вони вказують не тільки на тривалість, протягом якої зафіксовано певні дії користувача в системі, але також можуть показати, коли окремий параметр було додано до розділу або змінено. Такі особливості визначають системний реєстр як оптимальне місце для пошуку нерегулярних, аномальних дій, оскільки вразливість, шкідлива програма може суттєво відхилятися від звичайної діяльності і може бути виявлена.

Про несанкціоновані зміни реєстру ОС може свідчити невідоме ПЗ, скрипти або файли, що виконуються, наприклад, виявлення розділу реєстру, який вказує на підозрілий exe-файл без чіткого призначення або аномальний розмір ключа реєстру, наприклад, розділ реєстру, обсяг якого значно збільшився з невідомої причини.

Також у реєстрі ОС Windows зберігається інформація, яка стосуються безпеки: у реєстрі фіксується більшість змін у адміністративних програмах, наприклад, Панелі Керування (Control Panel); реєстр містить усю інформацію про безпеку (системні політики, імена користувачів та паролі); реєстр зберігає велику частину важливої інформації про час виконання e-rat guratimr, яка необхідна для запуску програм; реєстр містить повний перелік апаратних пристроїв (операційна система (Configuration manager) додає до реєстру інформацію про устаткування). Інформація зберігається у двійкових файлах, що дозволяє не лише записувати до реєстру великі обсяги різних даних, а й суттєво підвищує швидкість роботи з ними, але вимагає використання спеціалізованих програм.

Такі особливості даних реєстру і його функцій зробили його найбільш вразливим компонентом ОС і, одночасно, найважливішим джерелом даних і основою для реконструкції подій і виявлення зловмисних дій. Здійснення аналізу даних реєстру дозволить отримати інформацію щодо: програм, які запускаються автоматично (сліди шкідливих програм); втручання у політику безпеки (наприклад, відключення UAC); змін у механізмах аутентифікації; спроб приховування слідів присутності (очищення журналів).

У ході аналізу реєстру фахівці стикаються з певними проблемами: до складу реєстру входять взаємопов’язані ключі і дані, які реєструють нещодавно отримані файли, дані щодо активності користувача; записи реєстру часто змінюються під час роботи системи або встановлення ПЗ; дані реєстру можуть бути розбиті на частини по кущах та ключах; різні версії Windows вносять зміни у структури реєстру; записи реєстру не завжди мають явний контекст, що потребує експертної інтерпретації; постійна еволюція кіберзагроз створює нові зловмисні методи, які здатні навіть оминати традиційні процеси аналізу реєстру; традиційні інструменти потребують значних витрат часу і ресурсів і не мають необхідних функцій для виявлення аномалій.

Проте, активність реєстру ОС Windows, регулярність його роботи і те, що майже всі системи взаємодіють з реєстром, робить його важливим елементом для вирішення завдань забезпечення безпеки операційної системи. Водночас зміни,

викликані несанкціонованим доступом до реєстру, можуть призвести до нестабільності системи або виведення її з ладу, тому важливо забезпечити безпеку реєстру, щоб зробити безпечною усю систему і попередити наслідки можливих помилок. Відповідно, необхідним є аналіз змін, що відбуваються у реєстрі, і на його основі визначення вразливостей.

2.2 Ідентифікація критичних розділів і виявлення ненадійних або небезпечних налаштувань реєстру

Реєстр Windows містить значну кількість ключів, доступ до яких повинен бути обмеженим, щоб неправильні користувачі або програми не мали доступу до чутливих розділів реєстру, оскільки це може бути небезпечно.

Важливо правильно ідентифікувати ті частини реєстру Windows, які є ключовими для стабільності та безпеки ОС, оскільки зазвичай такі розділи використовуються для зберігання налаштувань, які контролюють поведінку системи, програм і користувачів, і неправильне або небезпечне налаштування може стати вразливістю.

Найбільш критичною для ушкоджень є інформація, що зберігається у кущі HKEY_LOCAL_MACHINE. Вона дозволяє ідентифікувати все встановлене ПЗ та включені функції ОС. Дані, що містяться в цьому розділі, відносяться відразу до всіх профілів користувачів, зареєстрованих у системі. Наприклад, серед іншого, в цьому кущі є багато об'єктів, які можуть бути включені за допомогою групової політики Windows. Цей кущ також містить ключі SAM та Security, пов’язані з доступом до облікових даних.

Більшість атак, впроваджуваних зловмисниками, пов’язана із діями шкідливих програм, які для забезпечення своєї безперервної роботи можуть створювати або змінювати ключі і значення для виконання зловмисних дій (наприклад, створювати записи автозапуску у реєстрі, щоб мати можливість запускатися автоматично при запуску Windows або впроваджуватись у записи інших програм з метою зміни їхньої поведінки, запитувати частини реєстру, щоб

отримати інформацію про вразливості, або впровадити нові ключі, які допоможуть створити вразливості на машині). Приміром, ключ Run може дозволити зловмисним файлам автоматично виконуватися під час перезавантаження скомпрометованої системи; зміни ключа HKEY_CLASSES_ROOT\exefile\shell\open\command можуть призвести до того, що небезпечні програми будуть виконуватися замість звичайних.

Деякі приклади дій шкідливих програм та способів створення ними аномальної діяльності реєстру [40], [41], [42], [43] наведені у табл. Б.2.1.

Поширеним методом є створення служб для отримання стійкості та виконання ескалації привілеїв. Якщо такі служби існують, суб'єкт може змінити шлях до файлу, щоб він вказував на шкідливий двійковий файл, і при наступному запуску служби буде виконано шкідливий код.

Увімкнення вбудованого локального облікового запису адміністратора, який за замовчуванням відключено, є ще одним прийомом, що використовується зловмисниками для досягнення стійкості. Основною тактикою зловмисників є підвищення привілеїв.

Приклади критичних для роботи системи ключів і файлів системного реєстру наведені у табл. Б.2.2. Критичною є також інформація, що міститься у файлі NTUser.dat, який являє собою файл детальних налаштувань користувацького облікового запису.

Важливими даними для виявлення змін у реєстрі, моніторингу активних процесів і безпеки у реальному часі є: UserAssist – ключ частини реєстру, що містить запис програм, які часто запускалися користувачем; MRU – останні використовувані файли; ShellBags – навігація папками і зовнішні підключені пристрої (USB-накопичувачі) [36].

У ключі UserAssist можна знайти такі значенні: ім'я файлу, час останнього запуску і їхня кількість. Дані UserAssist розміщені у Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist.

Ключі ShellBags можуть містити інформацію щодо минулих дій користувача, найменування і шляхи до відкритих папок (навіть якщо папка буда

видалена), докладні відомості щодо міток часу, часу створення, часу модифікація, часу доступу. Тобто, ShellBags дають уявлення щодо порушення конфіденційності. Інформація, що міститься у цих ключах є дуже важливою для здійснення аналізу потенційних наслідків порушень безпеки.

На основі проведеного аналізу структури і функцій системного реєстру, його особливостей, впливу шкідливих програм на його роботу визначені типові вразливості, які пов’язані з неправильними налаштуваннями системного реєстру Windows і здійснений їхній розподіл за критеріями безпеки (табл. Б.2.3).

Виявлення вразливостей відбувається постійно. У якості прикладу можна навести відомості щодо деяких вразливостей**,** які стосуються роботи реєстру:

– вразливість CVE-2024-43452 – критична вразливість реєстру Windows, що приводить до підвищення привілеїв. Експлуатація вразливості не потребує взаємодії з користувачем, а спирається на сприятливі для експлуатації умови. Вразливими, серед інших продуктів, є деякі версії Windows 10 [44];

– вразливість CVE-2022-21907 [45] дозволяє зловмисникам здійснити віддалене виконання коду на вразливому хості. Вона стосується файлу HTTP Protocol Stack (http.sys), який використовується для прослуховування HTTP-запитів на серверах IIS (Internet Information Services). Надсилання спеціально створеного пакета дозволяє віддалено виконувати код користувачам, які не здійснили автентифікацію.

Розкриття несанкціонованих чи шкідливих змін у реєстрі Windows є найважливішим аспектом захисту від вразливостей. Виявити несподівані дії, що приховані у системних налаштуваннях та можливі вразливості допоможе аналіз активних процесів – інформації стосовно нещодавно виконаних програм; файлів, які були використані останнім часом; навігації папками.

Для виявлення ненадійних або небезпечних налаштувань реєстру і вразливостей, потрібно регулярно перевіряти важливі ключі реєстру, слідкувати за змінами, які можуть свідчити про вторгнення шкідливого програмного забезпечення, використовувати програмні рішення для запису і порівняння змін у реєстрі.

2.3 Журнали подій операційної системи Windows

Журнал подій Windows (англ. Windows Event log) – це докладний запис подій, пов’язаних із системою, безпекою та застосунками, який зберігається в ОС Windows. Журнали подій можна використовувати для діагностики проблем системи та деяких додатків, а також прогнозування майбутніх проблем [46].

Операційна система С Windows відслідковує у своїх файлах журналу інформацію про апаратні та програмні події.

Записи журналу подій локалізовані у певному ключі реєстру HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog.

Кожне джерело подій у журналі має відповідний, індивідуально створений підключ. Події від кожного джерела можуть бути класифіковані в окрему для кожного джерела категорію (рис. Б.2.1).

Файли журналу подій (підключі) структуровано у три основні категорії:

– файл журналу застосунків – для застосунків і служб (інформація про встановлення ПЗ, оновлення та збої, про події, специфічні для застосунків);

– файл журналу безпеки призначений для реєстрації системних подій аудиту (інформація про спроби входу до системи, модифікації облікових записів, невдалі спроби доступу та інші інциденти, що стосуються безпеки);

– файл системного журналу, у якому фіксують події, що стосуються драйверів пристроїв (інформація про збої обладнання, проблеми з драйверами, запуск і завершення роботи системи та інші події на системному рівні).

Події повинні бути класифіковані за одним з п'яти визначених типів (табл. Б.2.4). Елементами журналу подій Windows є:

– ім’я журналу подій (до якого будуть записуватись події з різних компонентів, у яких виникла подія, що реєструються);

– джерело події (програма або процес, який її ініціює, наприклад, усі події входу та виходу з системи походять із джерела безпеки);

– ідентифікатор події (унікальний ідентифікаційний номер Windows, який допомагає адміністраторам однозначно ідентифікувати певну зареєстровану

подію, наприклад, події, що стосуються блокування облікових записів Windows фіксуються в журналі безпеки із джерелом захисту та ідентифікатором події 644);

– тип події (описує рівень серйозності події, що відбулася, і може сприяти класифікації виду діяльності, що її спричинив);

– категорія події (класифікує події на групи залежно від типу події);

– час/дата (час і дата події є ключовими для визначення частоти виникнення конкретної події в певний момент доби та в визначений день тижня;

– ідентифікатор сервера (унікальний ідентифікатор кожного сервера, спрощує адміністратору зв’язати сповіщення з конкретним хостом, що дозволяє швидко локалізувати джерело проблеми);

– ідентифікатор користувача (логін користувача, який увійшов у систему на комп'ютері Windows у момент, коли відбулася подія).

Основні функції роботи з подіями:

– OpenEventLog – відкриття журналу на певному комп’ютері для адміністративних операцій;

– ReadEventLog – зчитування частини журналу в буфер;

– GetOldestEventLogRecord – отримати номер найстарішого запису;

– GetNumberOfEventLogRecords – отримати кількість записів в зазначеному журналі;

– NotifyChangeEventLog – отримувати повідомлення при записі в зазначений журнал;

– BackupEventLog – запис журналу в архів;

– ClearEventLog – очищення журналу з можливістю запису в архів;

– OpenBackupEventLog – відкриття архівної копії журналу;

– CloseEventLog – закриття журналу;

– RegisterEventSource – відкриття журналу для запису подій від вказаного джерела;

– ReportEvent – запис події в журнал;

– DeregisterEventSource – закриття журналу, відкритого для запису.

Журнал подій може містити певні вразливості, пов’язані, наприклад, із:

– неможливістю розділити права на доступ до інформації та очищення журналу адміністраторами і використанням ними утиліти Winzapper для видалення записів з журналу подій, що створює значну загрозу. Якщо обліковий запис адміністратора був зламаний, цілісність історії подій може бути порушена. Для запобігання вразливості доцільно створити віддалений сервер для зберігання журналу, доступ до якого буде здійснюватися виключно через консоль;

– створенням порушником великої кількості нових подій, щоб заповнити журнал з метою його переповнення, що приведе до замін старих подій або зупинить запис, і зробить журнал сприйнятливим до атак. Збільшення максимального розміру журналу частково може допомогти вирішити проблему.

– реєстрацією зловмисника під обліковим записом адміністратора з метою змінити політику аудиту – зупинити запис у журналі несанкціонованої активності (в залежності від налаштувань політики аудиту). Запис про подію можна очистити за допомогою Winzapper.

Отже, важливість журналів подій у забезпечені безпеки сучасних IT-систем обумовлена наступним:

– журнали подій допомагають мережним адміністраторам відстежувати потенційні загрози і проблеми, які можуть погіршувати продуктивність;

– дані лог-файлів надають інформацію, яка допомагає пов’язати облікові записи з певними подіями і визначити, де саме необхідно застосувати термінові заходи безпеки;

– регулярна синхронізація дати / часу допомагає реконструювати хронологію події;

– інформацію, що містять лог-файли, використовують для виявлення проблем та ідентифікації інцидентів безпеки, таких як вразливості системи.

– дані журналу подій можна використовувати для виявлення кібератак, водночас необхідно враховувати різноманіття незвичних подій (невдалі спроби автентифікації, блокування облікових записів, аномалії у використанні пам'яті,

несанкціонований доступ до критично важливих файлів / даних тощо), щоб не прийняти хибне рішення.

Важливо своєчасно переглядати зміни, що відбуваються в реєстрі, оскільки саме зміни в реєстрі використовують зловмисники, щоб послабити цільове середовище та мати більший контроль над системою.

2.4 Інструменти моніторингу реєстру ОС Windows

Забезпечення ефективного захисту вимагає наявності упорядкованої системи, здатної отримувати інформацію із відповідних джерел, обробляти і аналізувати її, перетворювати на основі правил у корисну інформацію, яка у подальшому може використовуватися іншими програмами і базами даних.

Для роботи з реєстром фахівці використовують інструменти, функції яких призначені для резервного копіювання, відновлення, очистки і оптимізації реєстру, оскільки вірогідність та непошкодженість інформації, її репрезентативність у результаті фільтрації і структурування значно впливають на оснований результат аналізу оброблюваних даних.

Для збору даних з системного реєстру розробниками Windows передбачено кілька інструментів: REG EDIT.EXE, REG QUERY, EVENT VIEWER та інші.

Редактор системного реєстру REG EDIT.EXE – вбудований інструмент для перегляду та редагування даних реєстру дозволяє вручну переглядати ключі та значення реєстру, здійснювати пошук та експорт даних для подальшого аналізу.

Інструментом для отримання даних з реєстру через командний рядок (Command Prompt) є REG QUERY – команда, яку застосовують з метою перегляду вмісту ключів реєстру, підрозділів або параметрів. Наприклад, команда REG QUERY HKLM\Software\Microsoft\ResKit / Version відображає всі значення параметру Version в реєстрі [47].

Застосунок Event Log Explorer надає користувачам можливість переглядати, аналізувати та контролювати записи журналів подій Windows. Функціональність Log Explorer, зокрема в частині роботи з табличними даними,

перевершує можливості вбудованого засобу перегляду журналів подій від Microsoft. Event Log Explorer дає змогу користувачам аналізувати різноманітні журнали подій: безпеку, додаток, систему, налаштування, службу каталогів, DNS та інше. Інші функції включають: миттєвий доступ до журналів подій (інструмент дозволяє працювати як з локальними, так і з віддаленими журналами подій, а також з файлами журналу подій у форматі EVT та EVTX); ефективну фільтрацію (система фільтрації за описами подій дозволяє здійснювати пошук за описами подій з використанням регулярних виразів, фільтрувати за параметрами подій безпеки та застосовувати користувацькі фільтри); експортування подій та генерацію звітів (експортувати та друкувати події) [48].

Використання інструменту EVENT VIEWER – компоненту, що включений у склад ОС Windows, дозволяє адміністраторам переглядати лог дані – систематичний моніторинг та фіксація подій, параметрів і станів системи під час її функціонування.

Автоматизовані інструменти моніторингу журналів подій, які вміють фільтрувати події та ідентифікувати критичну проблему у системі, допомагають фахівцям уникнути процесу переглядання журналів подій у ручному режимі.

Запуск застосунку Windows Event Log Viewer для перегляду журналу подій надає інформацію про такі категорії подій: події програм (англ. Application events) – звіти про перебіг виконання програми; події безпеки (англ. Security events) – звіти про результати дій безпеки; події налаштування (англ. Setup events) – в основному стосуються контролерів домену; події системи (англ. System events) – звіти про події, пов’язані з системою та її компонентами; перенесені події (англ. Forwarded events:) – звіти, що надсилаються іншими комп’ютерами.

Застосунок MyEventViewer дозволяє переглядати кілька журналів подій в одному списку, а також опис подій та дані. Для запуску цього програмного забезпечення потрібно запустити виконуваний файл. Застосунок має спрощений інтерфейс. Він дуже зручний для користувачів: дозволяє переглядати події з віддаленого комп’ютера; визначені події можна приховати від конкретних користувачів; події можна відфільтрувати, використовуючи низку критеріїв [49].

Існують також спеціальні утиліти та програмні рішення для аналізу реєстру:

– Regshot – утиліта для контролю за змінами у системному реєстрі Windows. За допомогою RegShot можна порівняти стан записів у реєстрі «до» та «після» проведення системних змін (встановлення програм і т. п.). Є можливість додавати коментарі до логів, що зберігаються. RegShot допоможе з'ясувати, у разі, якщо система почне працювати нестабільно, які зміни у реєстрі стали причиною цієї нестабільності [50].

– RegRipper – інструмент для автоматизованого збору інформації з реєстру Windows. Він дозволяє витягувати різноманітні дані з ключів реєстру (наприклад, історію запусків, дані про користувачів тощо) [51]. Це спрощує аналіз активності системи, поведінки користувачів і потенційних ознак дискредитації, доводячи таким чином, що кущі реєстру є важливішим джерелом доказів наявності вразливостей у системі. Збір енергозалежних даних RegRipper допомагає реконструювати події і виявляти порушення безпеки.

– Yara – це інструмент, який допомагає дослідникам зловмисного ПЗ ідентифікувати та класифікувати зразки шкідливих програм. YARA дозволяє розробляти описи сімейств шкідливого ПЗ або інших об’єктів, що потребують аналітичної характеристики на основі текстових або двійкових шаблонів. Кожен опис має складатися з набору рядків і булевого виразу, які визначають його логіку. Можна використати правила Yara щодо даних реєстру [52].

Для автоматизації завдання, яке включає оболонку командного рядка, скриптову мову та платформу управління конфігурацією, використовують кросплатформене рішення Windows PowerShell (WPS).

Файли, облікові записи користувачів, служби Windows, і т. п. – будь-який об’єкт, який викликає відповідна спрощена програма (Cmdlet), що використовується у середовищі PowerShell, система WPS перетворює у своїй консолі на зручний лістинг [39].

Таким чином, для вилучення даних з реєстру і їх аналізу звичайно використовують існуючі інструменти або спеціальні утиліти.

2.5 Методи аналізу системного реєстру для виявлення вразливостей

Аналіз реєстру Windows є важливою складовою частиною вивчення поведінки системи та виявлення можливих аномалій або вразливостей. Під час аналізу досліджують зміни, що відбулися у реєстрі, і які свідчать про активні процеси: дані, що вказують на нещодавно виконані програми, останні виконувані файли, інформацію щодо підключених зовнішніх пристроях.

Захист від вразливостей полягає у застосуванні інструментів моніторингу реєстру або програмного забезпечення безпеки для відстежування змін у реєстрі у режимі реального часу.

Одним із методів є використання вбудованих інструментів для аналізу реєстру, який здійснюють за розділами реєстру, підрозділами, тимчасовими мітками та значеннями, коли фахівець вручну переглядає ключі та значення реєстру. Такий аналіз забезпечує безпосередній доступ до всіх налаштувань і точність, проте він може бути трудомістким і займати багато часу та, зокрема, особливо трудомістким у випадку великомасштабних атак при реагуванні на інциденти. Такі методи стають неефективними у середовищах з високим навантаженням або в гнучких середовищах [53].

Величезний обсяг та складність реєстру, в якому реєструються зміни, згенеровані встановленням ПЗ, діями користувачів та системними операціями, лише посилюють проблеми ручного аналізу [54].

Інструменти, які були розглянуті у попередньому підрозділі, частково вирішують проблеми, проте вони мають властиві їм обмеження у своїй адаптивності і часто не встигають за новими векторами загроз, які можуть ховатися в змінах реєстру. Це обмежує ефективність їх використання. Наприклад, виявлення аномалій на основі даних журналу подій вручну стає важчим у міру збільшення розміру файлів журналів. Виникає необхідність у автоматизованих засобах виявлення аномалій, які можуть ідентифікувати шаблони у даних реєстру. Перехід до проведення аналізу змін реєстру у режимі реального часу зменшить залежність від помилок персоналу, поліпшить ідентифікацію НСД.

Аналіз активних процесів можна здійснити шляхом порівняння декількох версій реєстру. Для здійснення методу можна використовувати RegRipper и Regshot. Збір даних передбачає вилучення кущів реєстру NTUser.dat, SYSTEM, SOFTWARE і SAM, які зберігають критично важливу інформацію щодо системи і користувача. Процес складається з таких етапів: створення образу цільової системи для забезпечення цілісності даних і попередження випадкових змін; після вилучення кущів реєстру їх аналізують щодо наявності свідчень активності користувача, встановлення програмного забезпечення і підключень зовнішніх пристроїв; інформація спрямовується до часової шкали подій. Наступним етапом є впровадження змін у реєстрі і порівняння даних. Виявлені відмінності документують і консолідують у таблиці.

За результатами проведеного розгляду інструментів для проведення моніторингу реєстру з метою виявлення аномалій або вразливостей, сформована табл. Б.2.5, у якій основні методи аналізу системного реєстру розділені на типи відповідно до процесу виявлення змін.

Таким чином, методи аналізу реєстру можуть варіюватися в залежності від цілей дослідження та наявних інструментів. Для глибокого аналізу можливим є застосування кількох підходів та використання методів машинного навчання для ефективної автоматизації аналізу реєстру та виявлення вразливостей.

2.6 Огляд методів і моделей машинного навчання для аналізу даних системного реєстру і виявлення вразливостей

Основними методами машинного навчання, необхідними для аналізу даних системного реєстру і виявлення вразливостей, є класифікація (для виявлення аномалій на основі історичних даних) і кластеризація (для групування подій та виявлення відхилень від нормальних груп).

Виявлення аномалій – пошук непередбачених значень (патернів) в потоках даних [55]. Аномалія (викид, помилка, відхилення або виключення) – це відхилення поведінки системи від стандартної (очікуваної) [56].

Авторами [57] зазначено, що аномалії, пов’язані з безпекою, поділяють на три категорії**:** точкові, контекстні та колективні. Точкова аномалія виникає, коли окремий елемент даних виявляє відхилення від загальної сукупності, і може розглядатися як аномалія. Контекстуальна аномалія спостерігається, якщо екземпляр даних є аномальним лише в певному контексті. Колективна аномалія утворюється, якщо послідовність пов’язаних примірників даних (наприклад, ділянка часового ряду) є аномальною по відношенню до цілого набору даних. Окремий екземпляр даних в такій послідовності може не бути відхиленням, проте спільна поява таких екземплярів є колективною аномалією.

Класифікація передбачає розподіл поведінкових екземплярів у попередньо визначені класи в наборі даних. Екземпляр, який не може бути віднесений до жодного з визначених класів, вважається відхиленням. Процес виявлення аномалій відбувається в два етапи: навчання та розпізнавання. Класифікатор тренується на масиві маркованих даних (точкам даних присвоюються категорії відповідно до встановлених правил), далі визначається приналежність до одного з відомих класів. В іншому випадку екземпляр позначається, як аномалія.

Найбільш широко вживаними механізмами реалізації розпізнавання аномалій за допомогою класифікації є: SVM, One Class SVM, Random Forest, та Autoencoders [34].

Метод опорних векторів (SVM) – це найпопулярніший алгоритм класифікації, що базується на принципі розділу простору гіперплощиною. SVM ідеально підходить для виявлення аномалій, де потрібно класифікувати між нормальними і аномальними класами. Алгоритм має велику масштабованість через свою простоту і може виконувати завдання, такі як виявлення аномалій в режимі реального часу [58].

One Class SVM – є однією з форм класичного алгоритму, для навчання якого достатньо мати всього один клас. Його використання доцільне у випадку, коли для тренування моделі доступні тільки спостереження, що не мають аномалій, і тоді можливо скористатися цим алгоритмом. Щоб навчити модель для кожного нового спостереження визначати, чи є воно аномальним.

Для виявлення аномалій шляхом відновлення даних і порівняння відновлених даних з оригінальними можуть бути використані автокодувальники (Autoencoders) [32].

Глибокі нейронні мережі, зокрема рекурентні нейронні мережі (RNN), можуть бути використані для аналізу послідовностей змін у реєстрі і прогнозування майбутніх змін або класифікації відмінностей між версіями, класифікації вразливостей. Алгоритми LSTM (Long Short-Term Memory) або GRU (Gated Recurrent Units) можуть бути використані для моделювання та аналізу послідовностей даних реєстру.

Порівняння знімків реєстру також можна розглядати як задачу кластеризації, де версії реєстру класифікуються в групи в залежності від подібності змін. Для цього можуть бути використані методи, що оцінюють схожість між реєстрами за допомогою векторизації даних. Можливість групувати записи реєстру за схожістю дозволяє виявити неочевидні зв'язки між різними параметрами, наприклад для порівняння версій реєстру.

Кластеризація у виявленні аномалій – це спосіб групування набору екземплярів мережевих даних таким чином, що в одному пакеті – кластері будуть всі зловмисні екземпляри, які досягають більшої схожості порівняно з іншими групами (кластерами, які можуть бути нормальними екземплярами).

Для задач кластеризації використовують алгоритми K-means, DBSCAN (знаходить скупчення точок і будує навколо кластери, виявляє аномалії) або Hierarchical Clustering (метод кластерного аналізу, завдання якого – створити ієрархію кластерів – ієрархічна кластеризація).

Метод k- means – один з найпоширеніших методів кластерного аналізу, він має на меті розподілити «n» об’єктів даних на «k» кластерів, намагаючись мінімізувати середню квадратичну відстань від усіх точок кластеру до його центру. Процес триває ітеративно до тих пір, поки внесення жодних змін у кластери стане неможливим. Поширеність методу k-середніх зумовлена його простотою, гнучкістю, швидкою збіжністю. Функціонування алгоритму значно

сповільнюється при кластеризації великих масивів даних. Крім того, існує ймовірність збіжності алгоритму до локального мінімуму цільової функції [59].

Ще один метод, який технічно близький до концепції кластеризації, – це викиди, які визначають деякі екземпляри даних як такі, що більше відхиляються від звичайних груп у просторі даних. Виявлення відхилень, які вказують на наявність шкідливих програм, можна здійснити за допомогою алгоритму Isolation Forest. Його завдання – ізолювати викиди.

Виявлення аномалій на основі кластеризації мають кілька переваг порівняно з класифікацією: дані класифікуються без нагляду та не вимагають позначок класів для всіх ознак; техніки кластеризації ефективні у великих наборах даних, завдяки тому, що вони зменшують обчислювальну складність та досягають кращої ефективності порівняно з іншими методами класифікації. Недоліками виявлення аномалій на основі кластеризації є те, що: методи сильно пов’язані з ефективністю створення профілю для нормальних екземплярів; кластеризація часто потребує багато часу для динамічного оновлення профілю для законних мережевих екземплярів [60].

Для реалізації методу обробки тексту використовують алгоритми на основі нейронних мереж Word2Vec, TF-IDF, або BERT, для перетворення реєстру в числові вектори, що дозволяють порівнювати записи між собою.

Вивчаючи зміни в реєстрі, можна використовувати методи ML для автоматичного виявлення шкідливого ПЗ, яке змінює ключові налаштування системи. Також моделі машинного навчання можуть застосовуватися для аудиту системних налаштувань і пошуку вразливостей на основі історичних даних.

Щоб розробити архітектуру механізму виявлення вразливостей, важливо дослідити існуючі роботи, у яких розглянуті можливості різних моделей ML. Виходячи з цих міркувань, був проведений огляд релевантних наукових робіт стосовно методу і задачі машинного навчання, у яких розглядалися методи та моделі ML для аналізу даних системного реєстру і виявлення вразливостей.

У дослідженні від компанії FireEye [61], використовували моделі ML, побудовані із застосуванням алгоритму градієнтний бустинг дерев рішень

(XGBoost)), для виявлення шкідливих програм через аналіз реєстру. Класифікація змін у реєстрі як шкідливих або легітимних та автоматичне виявлення шкідливих ключів дозволили визначити відхилення, які вказують на наявність rootkits або backdoors з точністю 94%.

Для виявлення аномалій у змінах реєстру компанія CrowdStrike впровадила модель ML на основі алгоритму One-Class SVM для класифікації рідкісних подій. Модель аналізувала час, частоту і природу змін у реєстрі, щоб автоматично сигналізувати про підозрілі дії до їх розгортання в системі, що дозволило ідентифікувати рідкісні або незвичні зміни з точністю 92% [62].

Дослідницька команда з Carnegie Mellon University (CMU) [63] створила систему на основі рекурентної нейронної мережі (RNN) для аналізу часових рядів змін у реєстрі і прогнозування можливих загроз. Модель ML навчалася на історичних даних про атаки та пов’язані з ними зміни в реєстрі, аналізувала дані реєстру і передбачала можливі атаки до їхнього завершення. Дослідники використали неконтрольоване навчання для виявлення аномалій і контрольовані методи для класифікації відомих атак. Система показала високу точність (96%) у прогнозуванні атак, що використовували незвичайні зміни конфігурацій реєстру для отримання доступу до системи.

Антивірусна компанія Symantec використовувала моделі глибинного навчання CNN для аналізу послідовностей змін у реєстрі, пов’язаних з програмами-шпигунами [64]. Використання моделі на основі CNN дозволяє виявляти приховані шаблони змін реєстру, що допомогло компанії швидко ідентифікувати й ізолювати підозрілі програми, навіть якщо вони були створені спеціально для обходу традиційних методів виявлення, з точністю до 91,04%.

Автори роботи [65] розглянули стан сучасних систем виявлення аномалій і підкреслили, що для отримання та підготовки інформаційних ознак з наборів даних важливими є методи попередньої обробки даних, включаючи створення ознак, стиснення, перетворення та нормалізацію, щоб надати методам ML відповідні ознаки, які зможуть забезпечити ефективність роботи моделі. У

дослідженні проаналізовані можливості алгоритмів і методів машинного навчання і глибинного навчання для застосування у кібербезпеці.

Алгоритми навчання на основі дерева рішень DTs є механізмом прийняття рішень, який використовується для поділу набору даних, що складається з багатьох записів, на менші набори за допомогою набору правил. Їх можна використовувати як у контексті регресії (результат – прогнозоване значення цільової функції), так і в контексті класифікації (в цьому випадку передбачається результат – клас, якому належать дані). Дерева рішень знаходять застосування в різних галузях зокрема, оцінки вразливостей [66].

Можливості різних алгоритмів машинного навчання, зокрема KNN, DT, SVM, PCA, K-Means, SVD, NB, ANN, LR и RF були розглянуті у роботі [67]. Автори зробили висновок про те, що необхідно використовувати переваги методів машинного навчання для аналізу реєстру операційної системи Windows.

Дослідження [68] пропонує багатомодульну структуру машинного навчання, яка включає некероване ML, масштабовану збірку ознак та відбір ознак. Для вчасного виявлення загроз використовується нова схема виявлення викидів, IsolationForest (ML-IF). Експериментально перевірено на наборі даних UNSW-NB15, що запропонована структура перевершує існуючі методи стосовно точності та рівня виявлення, зменшуючи обчислювальну складність.

Автори [69] зазначають, що ансамблі моделей, зокрема ансамблі дерев рішень: багінг і бустінг, та нейронні мережі та їх ансамблі є найбільш ефективними та популярними. Дерева рішень ефективно виявляють закономірності в даних, пропонуючи значні можливості для їх візуалізації та відбору ознак за критерієм важливості. Зокрема, IsolationForest – добре виявляє аномальні значення під час побудови лісу дерев.

Дослідники [70] зосередились на можливостях використання навчання з підкріпленням RL для ефективного розслідування зловмисних програм під час реагування на кіберінциденти. Моделі MDP, які засновані на RL системах, можуть класифікувати і визначати пріоритети зловмисним записам реєстру, що підвищує як швидкість, так і точність аналізу.

Автори [71] у своєму аналітичному дослідженні розглянули виявлення програмних вразливостей із застосуванням методів ML: контрольованих алгоритмів (NB, SVM, випадковий ліс, CNN, BLSTM) і неконтрольованих методів ML (пошук вразливостей за шаблонами, які створені експертним способом, для чого застосовувалися зниження розмірності, кластеризація API викликів, виявлення аномалій та аналіз викривлень). У висновках зазначено, що методи пошуку вразливостей на основі ML мають високий потенціал: наведено можливі підходи, зокрема, засновані на прихованих моделях Маркова і RNN.

З метою вивчення потенціалу методів ML для покращення традиційних підходів фазингу, автори дослідження [72] оцінюють продуктивність різних технік машинного навчання, включаючи класичні методи (ML) і глибоке навчання (DL), і зазначають, що алгоритми SVM частіше інших використовують в якості класифікаторів у фазингу. Вони є корисними у випадках, коли типів даних (ознак) більше, ніж фактичних точок даних (вибірок), також в порівнянні з іншими класифікаторами SVM потребують менше пам’яті, оскільки використовують лише деякі точки даних (опорні вектори) для прийняття рішень. Стосовно моделей глибокого навчання (DL) найбільш популярними визнані архітектури GAN і LSTM. Перевага LSTM полягає в його здатності обробляти послідовні дані.

Отже, найбільш часто застосовуваним методом пошуку вразливостей з використанням ML є класифікація, а найбільш часто використовуваними алгоритмами є SVM, GAN і LSTM.

З табл. Б.2.6, у якій систематизовані результати проведеного огляду релевантних наукових робіт, слідує, що різні моделі машинного навчання продукують хороші результати в завданнях кібербезпеки, зокрема, аналізі реєстру Windows. Кожен з цих методів може пришвидшити аналіз великих обсягів даних у реєстрі і підвищити точність виявлення вразливостей.

Також визначено, що необхідно застосовувати алгоритм, який найкраще відповідає наданому набору вхідних даних. а також від вимог до роботи саме алгоритму – швидкості, якості, необхідних ресурсів.

Проведений аналіз досліджень стосовно можливостей алгоритмів машинного навчання у процесах аналізу системного реєстру і виявлення вразливостей дозволив, з урахуванням відомостей, наведених у [69] розробити схему, яка містить методи і алгоритми машинного навчання, архітектури глибинного навчання, застосовування яких можливе для аналізу системного реєстру і виявлення вразливостей (рис. Б.2.2).

Методи, засновані на штучному інтелекті для виявлення та захисту від кібератак і загроз, проаналізовані у дослідженні [73]. Автори зробили висновок, що хоча методи ШІ є більш ефективними в порівнянні з традиційними стратегіями кібербезпеки, заснованими на сигнатурах та правилах, більшість методів, заснованих на машинному навчанні та методах глибокого навчання, розгортаються за принципом «чорного ящика», що означає, що експерти з безпеки не можуть пояснити, як такі процедури надають конкретні висновки. Тому при розробці засобів кібербезпеки, необхідно створювати моделі, які простіші для пояснення, але зберігають при цьому високу точність і дозволяють розуміти механізми кіберзахисту наступного покоління, а також керувати ними.

На основі проведеного аналізу методів машинного навчання і типів алгоритмів ML, які застосовують у кібербезпеці для аналізу системного реєстру і виявлення вразливостей, визначено, що вирішити завдання підвищення захищеності операційної системи у ході аналізу активних процесів можна впровадженням низки рішень, які передбачають використання моделей ML:

– аналіз великої кількості даних алгоритмами ML має допомогти у виявленні складних, невидимих на перший погляд проблем. Тому необхідною є інтеграція різних джерел даних, таких як результати аналізу реєстру, логів виконання, інформації про вразливості;

- використання Марківського процесу прийняття рішень (МППР), оскільки він подібний до процесу пошуку вразливостей у реєстрі. Поточні дані щодо системи можна визначити як стан системи, а застосування утиліти – як дію. МППР автоматизує і оптимізує процеси, щоб замінити втручання людини у послідовні етапи прийняття рішень;

- впровадження моделі ML з підкріпленням. У контексті завдання підвищення захищеності ОС і виявлення вразливостей, середовище для RL може бути симуляцією дій реєстру операційної системи Windows у поєднанні з реальними журналами даних. Це має дозволити агенту взаємодіяти з різними станами реєстру та отримувати зворотний зв'язок. Алгоритм самостійно визначає найбільш корисні дії для підвищення продуктивності в заданій ситуації. Це дозволить експертам приймати статистично обґрунтовані рішення в процесі дослідження складного реєстру Windows, аналізу тимчасової шкали і мінливих даних в ході аналізу активних процесів;
- застосування методів машинного навчання у завданнях кібербезпеки з метою виявлення вразливостей реєстру, які передбачають розробку моделі на основі алгоритмів Random Forest, IsolationForest, SVM, Neural Network. Наприклад, вони автоматизують виявлення аномальних змін у процесі порівняння двох знімків реєстру (модель ML навчиться розпізнавати нормальні та анормальні зміни в реєстрі на основі попереднього досвіду та даних і класифікувати виявлені зміни за критеріями інформаційної безпеки);
- аналіз опису вразливостей з бази даних CVE здійснювати за допомогою методу обробки природної мови (NLP). Для перетворення опису вразливостей у числові вектори можна застосувати алгоритм TF-IDF, а для класифікації, наприклад, логістичну регресію);
- інтеграція методів машинного навчання для виявлення вразливостей реєстру ОС Windows з традиційними методами аналізу, такими як статичний і динамічний аналізи. Це може удосконалити модель ML (наприклад, статичний аналіз можна використовувати для базового порівняння значень реєстру, а машинне навчання може допомогти автоматизувати процес виявлення аномальних змін у реєстрі, які не відповідають еталонним зразкам).

Такі рішення можуть покращити точність і швидкість виявлення вразливостей, допомогти у розумінні контексту вразливостей, що сприятиме розробці більш надійних захисних стратегій.

Машинне навчання дозволяє автоматизувати процеси моніторингу і виявлення аномалій, що дає можливість аналітикам сконцентруватися на більш складних завданнях, покращити продуктивність системи.

Алгоритми ML можуть масштабуватися для аналізу великих обсягів даних та сприяють забезпеченню безпеки реєстру операційної системи.

Висновок за розділом 2

За результатами проведеного у розділі аналізу, важливо зазначити, що для запобігання непоправної шкоди для роботи ОС необхідно аналізувати дані реєстру ОС Windows і виявляти можливі вразливості для вирішення задачі підвищення захищеності ОС. Насамперед, підлягають аналізу для виявлення змін і вразливостей системного реєстру найбільш критичні розділи: HKEY_CURRENT_USER і HKEY_LOCAL_MACHINE. Журнал подій ОС, що розташований у журналі подій Windows, є важливим джерелом даних для аналізу.

Типові вразливості реєстру ОС Windows впливають на конфіденційність, цілісність, доступність інформації. Для виявлення вразливостей е важливим і необхідним процесом є аналіз реєстру Windows. Існують різні застосунки та утиліти, які застосовують для вилучення даних з реєстру і їх аналізу, що допомагає контролювати роботу системи.

За результатами проведеного розгляду інструментів для проведення моніторингу реєстру з метою виявлення аномалій або вразливостей, визначені основні методи аналізу системного реєстру відповідно до процесу виявлення змін. Встановлено, що для підвищення ефективності захисту, важливо автоматизувати процес аналізу. Це вимагає використання моделей ML.

Огляд методів і моделей ML для аналізу даних реєстру ОС і виявлення вразливостей, здійснений на основі актуальних наукових досліджень, дозволив розробити схему, яка містить методи і алгоритми ML, , застосовування яких є можливим для аналізу системного реєстру і виявлення вразливостей, і визначити можливості застосування певних алгоритмів.

РОЗДІЛ 3
ПРОЄКТУВАННЯ МОДЕЛІ ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ

3.1 Загальні вимоги до механізму виявлення вразливостей методами машинного навчання

У попередньому розділі було показано, що алгоритми ML ефективні у виявленні вразливостей, і, таким чином, зосередження уваги на використанні таких методів для виявлення вразливостей у системному реєстрі, може підвищити захищеність операційної системи і стати ефективним механізмом захисту.

Метою розробки моделі машинного навчання є створення механізму, який має аналізувати реєстр операційної системи, знаходити потенційно небезпечні або вразливі ключі, повідомляти користувача про знайдені ризики.

Для досягнення мети встановлені конкретні вимоги до механізму. Модель машинного навчання повинна:

- працювати локально в межах операційної системи Windows;
- зчитувати певні гілки реєстру, які є найбільш відповідними для аналізу (наприклад, HKLM, HKCU);
- порівнювати знайдені ключі зі списком відомих вразливостей (наприклад, небезпечних автозапусків, слабких політик безпеки тощо);
- класифікувати виявлені вразливості за критеріями інформаційної безпеки (конфіденційність, цілісність, доступність);
- виводити звіт роботи у формі таблиці, наприклад, розділ реєстру, дата виявлення, тип вразливості, підсумок;
- у процесі роботи зберігати інформацію, яку використовує і накопичує система стосовно вразливостей, у локальній базі даних і синхронізувати її із зовнішніми БД, актуалізувати метрики і іншу інформацію.

Механізм виявлення вразливостей у середовищі реєстру Windows на основі методів машинного навчання має передбачати застосування таких алгоритмів ML, які зможуть навчитися розпізнавати нормальні та аномальні зміни в реєстрі на основі попереднього досвіду та даних, що дозволить визначити аномальні зміни у реєстрі, і бути зрозумілими для фахівців.

3.2 Процес машинного навчання моделі виявлення вразливостей

Процес машинного навчання моделі для аналізу реєстру операційної системи Windows і виявлення вразливостей включає кілька ключових етапів, послідовність яких наведена на рис. 3.1.

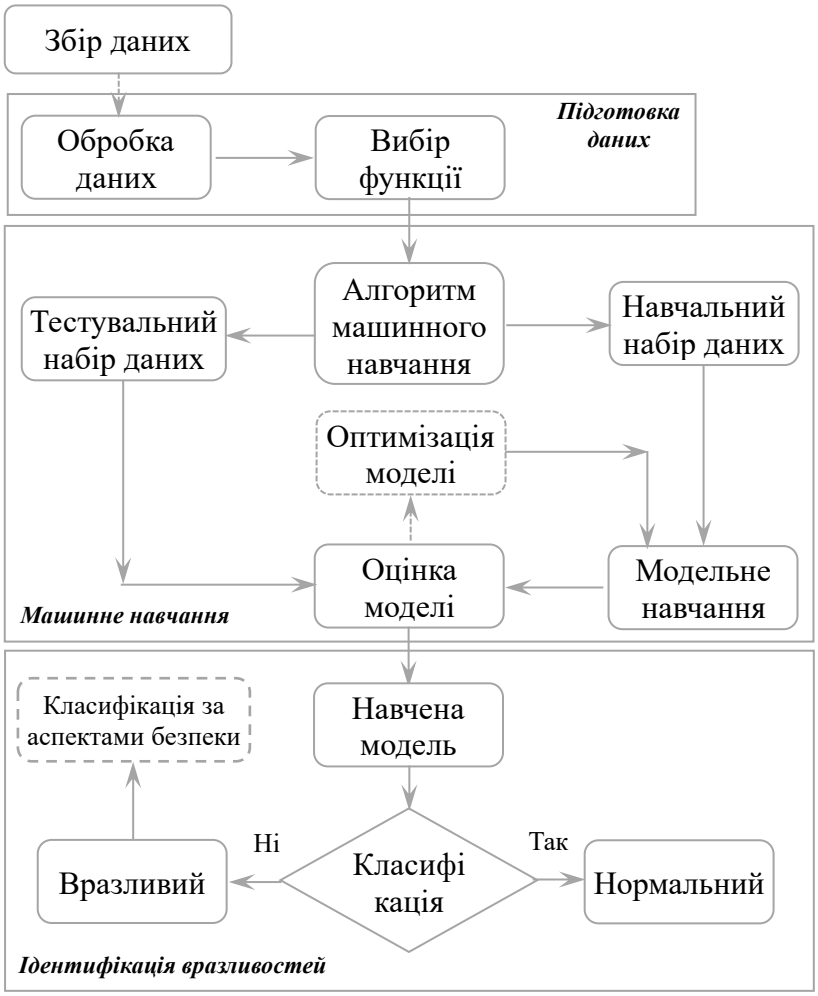

Рисунок 3.1 - Процес машинного навчання для пошуку і ідентифікації вразливостей у системному реєстрі

Машинне навчання має слідувати основним крокам, які складаються із необхідних для тренування моделі ML процесів, і містити деякий додатковий процес для виявлення і класифікації вразливостей:

1. Збір та загрузка даних. Для створення ефективної моделі машинного навчання, здатної аналізувати реєстр і класифікувати виявленні аномальні дані за аспектами інформаційної безпеки, важливо вибрати правильні типи даних для тренування, оскільки від цього залежить ефективність роботи моделі ML. Ці фактичні дані повинні бути найбільш відповідними для аналізу і містити інформацію, важливу для виявлення змін у реєстрі, моніторингу активних процесів і допомагати ідентифікувати потенційні вразливості.

2. Процес обробки. Складається з: попереднього аналізу та підготовки даних, який полягає у очищенні, нормалізації та трансформації даних, щоб забезпечити їх придатність для аналізу. Це особливо важливий етап у випадках, коли дані збираються з різних джерел. Проблемами з даним можуть бути, наприклад, порожні стовбці і дублювання рядків або різні типи даних. Очистка даних фокусується на виявленні і виправленні помилок у наборі даних.

3. Вилучення ознак. За допомогою процесу, який використовується для зменшення розмірності набору даних, приймаючи багато функцій і представляючи їх у вигляді меншої кількості функцій, зменшують складність даних, щоб витягти основні ознаки, які спрямовані на пошук найбільш важливих вихідних даних, що покращить обчислювальну ефективність. Потім ознаки, які працюють для використання існуючих даних, проєктуються для отримання нових змінних.

4. Навчання моделі. На цьому етапі необхідно вибрати тип моделі машинного навчання (алгоритми виявлення аномалій або моделі класифікації). Далі модель тренують, використовуючи марковані або немарковані набори даних (обрані алгоритми виконують тренування на великій кількості даних, модель аналізує дані, виявляє закономірності та навчається на основі цих закономірностей).

5. Оцінка моделі. Після навчання моделі проводиться ретельне тестування для оцінки продуктивності і точності моделі у виявленні вразливостей або класифікації подій за допомогою набору даних, які не використовувалися під час тренування, що дозволяє перевірити її точність і надійність. Отримані результати тестування надають уявлення про ефективність моделі та є необхідними для визначення необхідності коригування або доопрацювання моделі на основі отриманих результатів тестування для досягнення кращих результатів або поліпшення швидкості роботи.

6. Виявлення вразливостей. На етапі виявлення вразливостей використовують перевірені дані з основного набору даних, оскільки результат залежить від набора даних і контрольованості використовуваних алгоритмів. Для підтвердження продуктивності моделі виявлення вразливостей можна використовувати інший набір даних. Вихідні дані цього етапу спрямовуються на етап класифікації для визначення типу виявлених даних.

7. Класифікація. Завданням останнього етапу є класифікація і аналіз даних з метою визначення їх нормальності з використанням різних типів алгоритмів машинного навчання та класифікаторів. Іноді їх класифікують за типами анормальних даних.

Виявлення вразливостей реєстру Windows з метою підвищення захищеності операційної системи може бути здійснено шляхом навчання моделі машинного навчання, яка буде аналізувати критичні ключі реєстру для вилучення основних ознак і аналізу їх для виявлення вразливостей і повідомлення про них.

3.3 Розробка архітектури механізму виявлення вразливостей

Щоб забезпечити виконання поставлених вимог і правильне функціонування, модель виявлення вразливостей має складатися з таких ключових компонентів (рис. В.3.2):

- модуль підготовки даних, який виконує функції збору і обробки даних, вилучення ознак:

- модуль машинного навчання, який включає вибір алгоритму ML, навчання моделі і оцінку;
- модуль ідентифікації, який здійснює виявлення вразливостей, класифікацію вразливостей відносно впливу на систему за аспектами безпеки інформації, передавання результатів фахівцям;
- модуль управління або пом’якшення вразливостей для розгляду результатів аналізу фахівцями і інтеграції з системою SIEM для забезпечення безпеки системи.

Побудована структура механізму може забезпечити комплексний підхід до виявлення вразливостей, інтегруючи різні технології для забезпечення високої точності та ефективності системи.

Важливо вибрати правильні типи даних для аналізу, оскільки неповні дані можуть спотворити результат виявлення. Основними типами даних, які варто використати для тренування моделі, є наступні:

- дані з файлів кущів реєстру для отримання критично важливої інформації для аналізу (SYSTEM, SAM, SOFTWARE, SECURITY та NTUSER.DAT), які містять відомості, що допомагають у реконструкції подій і виявлення порушень безпеки, і можуть полегшити аналіз активності системи, поведінки користувача і потенціальні ознаки порушень;
- дані лог-файлів Журналу подій, які використовують для розпізнавання проблем та ідентифікації подій безпеки, зокрема, вразливостей системи. Записи журналу подій містяться в ключі реєстру HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\EventLog. Ці журнали можуть зберігати залишки реєстру навіть після видалення, що робить їх цінними для аналізу;
- дані про вразливості, які допомагають навчити моделі машинного навчання розпізнавати відомі вразливості та їхні характеристики. Доцільно використовувати загальновідомі бази даних, наприклад, Common Vulnerabilities and Exposures (CVE).

Використовуючи інтеграцію різних типів даних, наведених вище, можна отримувати, досліджувати та інтерпретувати дані реєстру для виявлення дій користувачів, змін у системі та потенційних інцидентів безпеки, зокрема, виявляти вразливості.

Попередня обробка даних передбачає приведення всіх вихідних даних до одного формату, зручного для подальшого аналізу і машинного навчання, оскільки реєстр Windows складається з численних ключів і значень. Це може бути .csv, .json або таблиця з ключами – стовпцями і їх значеннями – значеннями в рядках.

Попередня обробка даних передбачає автоматизований збір даних з двох знімків реєстру, виділяючи ключі, підключі, значення та їх типи. Для обробки можна використовувати бібліотеки pandas, difflib (Python) або інші інструменти. Наступною дією є видалення дублікатних або незначущих змін (наприклад, часові мітки).

При формуванні ознак важливо врахувати, що відсутність певних деталей може ускладнити вивчення класифікатором. Формування ознак (feature engineering) полягає у створенні векторного представлення реєстру: категоріальні ознаки (назви ключів, шляхи); числові ознаки (довжина значень, числові параметри); бінарні ознаки (було змінено чи ні).

З кожного ключа реєстру можна зібрати, наприклад, такі ознаки: num_values (скільки значень містить ключ), num_subkeys (кількість підключів), depth (глибина в ієрархії ключа), value_type_count (кількість різних типів ключів), has_run_in_name (бінарна ознака, яка позначає чиє "Run"/"Service" у шляху), is_remote_key (вказує, чи стосується ключ віддаленого доступу (RemoteRegistry). Можна застосовувати регулярні вирази (regex) для пошуку певних шаблонів.

Для ефективного аналізу і порівняння знімків реєстру необхідно виконати наступне:

- створити знімок (snapshot) реєстру у вигляді структурованого файлу (наприклад, .json або .csv) для забезпечення цілісності даних і попередження випадкових змін. Після вилучення кушів реєстру їх аналізують на наявність

активності користувача, встановлення ПЗ, підключення зовнішніх пристроїв. Початковий знімок кущів реєстру з цільової системи було зроблено перед введенням будь-яких тестових змін. Було зазначено певний каталог, що містить відповідні файли кущів;

– потім, після введення тестових змін у реєстрі для створення другого знімку кущів реєстру, повторити сканування;

– порівняти знімки реєстру для виявлення аномальних змін. У результаті порівняння двох знімків: отримуємо додані ключі, видалені ключі, модифіковані ключі.

– Для кожного зміненого ключа формуємо ознаки для процесу машинного навчання: глибина (depth), кількість змін, типи значень, ознаки назви ключа (наприклад, has_run, is_service, is_remote).

Завданням модуля машинного навчання є використання алгоритмів ML для аналізу змінених ключів, виявлених у результаті порівняння двох знімків реєстру. Результатом має бути автоматичне виявлення аномальних змін у реєстрі, що можуть бути пов’язані з: шкідливими автозапусками, змінами служб, підозрілими ключами тощо.

Робота модуля складається з наступних етапів:

1. Вибір алгоритму. Для виконання завдання, у випадку аналізу реєстру операційної системи, доцільно використовувати тип навчання без вчителя, оскільки і алгоритм машинного навчання, який призначений для аналізу реєстру за допомогою методу виявлення аномалій. Аномальні події можуть стати цінними факторами для фахівців з інформаційної безпеки, оскільки сигналізують про необхідність негайних дій стосовно захисту інформації. З урахуванням необхідних функціональних можливостей алгоритму, було прийнято рішення, що для реалізації методу виявлення аномалій можна використовувати алгоритм Isolation Forest, оскільки він добре справляється з великими наборами даних і ефективно виявляє аномалії [70].

Алгоритм Isolation Forest має лінійну тимчасову складність і низьке використання пам’яті, що важливо при аналізі великих об’ємів даних [74].

2. Навчання моделі. Модель Isolation Forest навчається шляхом надання їй зібраних даних, де вона вивчає зв’язок між функціями та відповідними мітками. Записи, які модель позначила як "Anomaly", мають властивості, що відрізняються від більшості даних, і можуть бути виявлені як незвичні або підозрілі зміни в реєстрі. Записи, позначені як "Normal", є стандартними і не мають виявлених аномалій. Такий підхід можна використовувати для виявлення підозрілих змін у реєстрі Windows, наприклад, для моніторингу системи на наявність зловмисних змін чи для порівняння версій реєстру між різними станами системи. Можна налаштувати параметр contamination для конкретних задач і аналізувати різні типи аномалій залежно від специфіки системи.

3. Оцінка моделі. Після навчання моделі відбувається тестування її роботи, що передбачає оцінювання точності моделі на тестовій вибірці (дані, яких вона раніше не бачила), щоб виміряти її продуктивність.

Ключові метрики, які використовують для вимірювання точності навченої моделі. наступні:

- точність (Accuracy): визначається як відношення правильних результатів до загальної кількості випадків, використовується в задачах класифікації, де визначається відсоток правильних передбачень за формулою:

$$\text{Точність} = \text{TP} + \frac{\text{TN}}{\text{TP}} + \text{TN} + \text{FP} + \text{FN} \qquad (3.1)$$

де, TP (True Positives) – кількість випадків, коли модель правильно ідентифікувала вразливість.

TN (True Negatives) – кількість випадків, коли модель правильно ідентифікувала відсутність вразливості.

FP (False Positives) – кількість випадків, коли модель помилково ідентифікувала вразливість.

FN (False Negatives) – кількість випадків, коли модель не змогла ідентифікувати наявну вразливість;

- повнота (Recall) вимірює здатність моделі виявляти всі реальні випадки вразливостей. Це відсоток істинно позитивних результатів з усіх реальних позитивних випадків.

$$\text{Повнота} = \text{TP}/(\text{TP} + \text{FN}) \tag{3.2}$$

- точність (Precision) вимірює, наскільки багато з тих випадків, які модель класифікувала як вразливі, дійсно є вразливими. Це відсоток істинно позитивних результатів з усіх позитивних прогнозів, зроблених моделлю:

$$\text{Точність} = \text{TP} \,/\, (\text{TP} + \text{FP}) \tag{3.3}$$

- F1–показник (F1-score) є середньозважене значення точності та запам1ятовування від 0 до 1, де 1 – оптимальний показник F. Використовується як міра точності в задачах, де є дисбаланс між класами;

$$F_1 = 2 \cdot (\text{Точність} \cdot \text{Повнота})/(\text{Точність} + \text{Повнота}) \tag{3.4}$$

де, Точність (Precision) – відсоток правильно ідентифікованих вразливостей з усіх випадків, які модель класифікувала як вразливі.

Повнота (Recall) – відсоток правильно ідентифікованих вразливостей з усіх реальних випадків вразливостей.

Ці метрики часто використовуються разом для отримання повної картини ефективності моделі машинного навчання. Вибір метрик може залежати від конкретної задачі та контексту застосування моделі.

Вимоги до системи можуть передбачати важливість зосередження на певних показниках, наприклад, максимізувати показник Повнота для забезпечення виявлення якомога більшої кількості вразливостей, навіть за рахунок деякого збільшення помилкових позитивних результатів.

Такий підхід до оцінювання допомагає визначити, який алгоритм найкраще відповідає наданому набору даних для вирішення певної проблеми. Вибір алгоритму залежить від поставлених задач, об’єму та якості вхідних даних, а також згідно з вимогами до самої роботи алгоритму – швидкості, якості, необхідних ресурсів. Робота моделі на цьому етапі спрямована на виявлення вразливостей реєстру операційної системи. Натренована і перевірена модель класифікує дані на нормальні і аномальні з використанням алгоритму Isolation Forest.

Isolation Forest (IF) – це сучасний метод для виявлення аномалій, робота якого полягає в тому, що він використовує принцип ізоляції точок даних, а не їх порівняння з іншими точками або шаблонами. Ізоляція – це процес відділення аналізованого екземпляра від інших екземплярів вибірки.

Алгоритм Isolation Forest будує випадкові дерева прийняття рішень, де кожне дерево має на меті ізолювати точки даних. Передбачається, що аномалій мало і вони різні, і тому вони більш схильні до ізоляції.

Головною ідеєю роботи алгоритму є те, що аномальні точки можуть бути ізольовані від решти даних за допомогою мінімальної кількості розділів (операцій розподілу), тоді як звичайні (нормальні) точки потребують значно більше таких операцій для їх відокремлення. Оскільки аномальні точки, як правило, відокремлюються від решти даних швидше, їх глибина в дереві буде значно меншою.

У ході роботи алгоритму простір ознак поділяється випадковим чином. Його дії полягають у побудові випадкового бінарного дерева, в якому корінь – весь простір ознак, а вузол представлений випадковою ознакою та порогом розбиття.

Поріг розбиття вибирається із рівномірного розподілу на відрізку від мінімального до максимального значення обраної ознаки. При тотожному збігу об'єктів у вузлі завершуємо роботу алгоритму. Глибина листа в дереві буде відповідати значенню алгоритму anomaly_score.

Оцінку аномалії виконують за формулою:

$$S(x,n) = 2^{-\frac{E[h(x)]}{c(n)}}, \quad (3.5)$$

де h (x) – довжина шляху до спостереження x (довжина шляху об'єкта x в дереві ізоляції);

c (n) – середня довжина шляху безуспішного пошуку у двійковому дереві пошуку (нормувальне значення, яке залежить від розміру вибірки), а n - кількість точок даних. Ця нормалізація необхідна, оскільки, якщо кількість об'єктів у дереві збільшується (тобто розмір вибірки більший), довжина шляхів для всіх об'єктів також зростає;

E (h(x)) – це середня довжина шляху об'єкта в усіх деревах моделі.

Оскільки аномалій небагато і вони відрізняються від інших даних, вони матимуть коротший шлях до кореня дерева порівняно з нормальними об’єктами даних і тому оцінка аномалії буде обернено пропорційна довжині шляху.

Робота алгоритму складається з кількох етапів (рис. В.3.3):

1. Побудова випадкових дерев. Алгоритм генерує безліч випадкових дерев (зазвичай кілька сотень або тисяч), які намагаються ізолювати точки даних. Розподіли для кожного дерева будуються випадковим чином, і кожне дерево містить різні варіанти розподілу даних. Чим менше розділень необхідно для ізоляції точки, тим ймовірніше, що ця точка є аномальною.

2. Ізоляція точок. Під час побудови дерев кожна точка відокремлюється від інших точок за допомогою випадкових розподілів. Точки, які легко піддаються ізоляції, мають коротку глибину в дереві, тоді як для нормальних точок потрібні додаткові кроки для їх ізоляції. Таким чином, аномалії відокремлюються значно швидше, ніж нормальні точки.

3. Обчислення коефіцієнта аномальності. Кожна точка отримує коефіцієнт аномальності на основі глибини, на якій вона була ізольована. Точки, які відокремлюються на ранніх етапах (з меншою глибиною дерева), отримують

вищий коефіцієнт аномальності. Якщо цей коефіцієнт перевищує певний поріг, точка класифікується як аномалія.

4. Результати виявлення аномалій. Після того як для кожної точки обчислено коефіцієнт аномальності, алгоритм визначає, чи є точка аномальною, порівнюючи її коефіцієнт з пороговим значенням. Якщо коефіцієнт аномальності значно більший за 1, точка вважається аномальною. Перевагою Isolation Forest є його швидкість та ефективність при обробці великих наборів даних.

Алгоритм здатний працювати з величезними обсягами даних, при цьому має лінійну складність, що дозволяє використовувати його для масштабованих задач без значних витрат часу чи пам'яті. Окрім того, цей метод не вимагає попереднього розподілу даних, що робить його універсальним для роботи з різноманітними типами даних. Оскільки Isolation Forest є методом навчання без вчителя, він не потребує попередньо помічених аномалій і може застосовуватися до нових наборів даних без маркування [71].

Одним із основних недоліків Isolation Forest є насамперед чутливість до параметрів, зокрема до вибору кількості дерев і глибини дерев. Невірно налаштовані параметри можуть значно знизити точність виявлення аномалій.

Алгоритм Isolation Forest активно застосовують для обробки великих даних, де він ефективно виявляє аномальні події в реальному часі, такі як відхилення в поведінці користувачів або систем. Isolation Forest є одним з найшвидших і найефективніших методів для виявлення аномалій, особливо в ситуаціях, коли важлива швидкість обробки великих даних.

Дані, отримані на цьому етапі, модель спрямовує на етап класифікації для визначення типу виявлених даних. Класифікація вразливостей за впливом на систему здійснюється згідно з основними принципами безпеки інформації: конфіденційності, доступності та цілісності. Для того, щоб визначити, до якого критерію відноситься конкретна виявлена вразливість, можна застосувати аналіз вразливостей на основі метрик методами машинного навчання. Це може бути корисно для класифікації системних вразливостей, які мають числові або метадані (наприклад, відомі вразливості з CVE).

Наступним кроком є створення класифікаційної моделі на тих прикладах, які були віднесені до аномалій, і її тренування на цих аномаліях, щоб передбачити їх тип. Для класифікації обрано алгоритм Random Forest – ансамблевий метод машинного навчання, який працює за допомогою побудови численних дерев прийняття рішень під час тренування моделі й продукує значення випадкової величини, що трапляється найчастіше в сукупності спостережень, для класів (класифікацій). Метод Random Forest може бути адаптований до Isolation Forest, дозволяючи об'єднати результати кількох моделей і таким чином отримати більш стабільні та точні прогнози.

В алгоритмі Random Forest ознака для розподілу визначається з рандомної підмножини m. Прийнято для вирішення задачі класифікації обирати $m = \sqrt{|d|}$, де d – кількість ознак в датасеті.

У задачах класифікації побудова дерева рішень має тривати доти, доки кожен лист не міститиме не більше одного об’єкта. Це є передумовою більш точного визначення класу, проте не є обов’язковою вимогою. Для більшості задач Random Forest дає одні з найкращих результатів, випадково вибирає sqrt (кількість ознак), ефективний для середніх та великих наборів даних [69].

Алгоритм процесу виявлення і класифікації вразливостей системного реєстру методами машинного навчання наведено на рис. В.3.4. Результатом роботи алгоритму є автоматична класифікація нових вразливостей, які потрапляють у систему, в одну з визначених категорій.

Розроблений механізм виявлення вразливостей реєстру операційної системи методами машинного навчання програмно реалізований в частині виявлення і класифікації вразливостей за критеріями безпеки. Вихідний програмний код рішення міститься у Додатку Г.

Отримані результати роботи модуля виявлення вразливостей мають бути передані до модуля управління для прийняття рішень щодо управляння вразливістю.

Модуль управління складається з двох підсистем: підсистеми усунення вразливостей, яка використовує машинне навчання для ідентифікації і

класифікації виявлених вразливостей з використанням наборів даних CVE і CWE і виявлення невідомих вразливостей шляхом вивчення шаблонів існуючих вразливостей, наприклад, вивчення шаблонів з попередніх сценаріїв, і підсистеми генерації звітів, що генерує звіти і висновки, які мають містити інформацію, таку як: скановані розділи реєстру, виявлені вразливості, рівень загрози вразливості за аспектами інформаційної безпеки.

Визначений рівень загрози вразливості у результаті класифікації є основою для оцінки рівня ризику і вибору стратегії захисту (виправлення, пом'якшення, прийняття), що надає можливість обрати необхідні засоби захисту (рис. В.3.5).

Вибір стратегії «Усунення» передбачає повне усунення проблеми, наприклад встановленням виправлення. Для виправлення вразливостей важливо: використовувати патчі або оновлення для програмного забезпечення; переглянути політики безпеки та налаштування доступу; проводити додаткове тестування після виправлення вразливостей. Якщо повністю усунути проблему неможливо, запроваджують стратегію «Зниження», що містить заходи зі зниження ризику використання вразливості, наприклад, шляхом оновлення системи та переналаштування контролю безпеки. Стратегія «Прийняття» може бути застосована, коли вразливість не пов'язана з особливими ризиками. Наступним кроком такої стратегії може бути внесення оновлень у контроль безпеки та конфігурацію безпеки.

Найбільш важливою процедурою безпеки системного реєстру є підтвердження того, що тільки авторизовані користувачі можуть з ним взаємодіяти.

Шляхами усунення експлуатації вразливості можуть бути надійні методи автентифікації і авторизації, контроль доступу. Виявляти нові загрози безпеки системи дозволяє постійний моніторинг.

Згенеровані звіти прийнятих рішень обов'язково перевіряються експертом, щоб гарантувати, що прийняті рішення правильні і доречні.

Механізмом, який забезпечить відправлення звітів експертам, може бути електронна пошта. Цей механізм буде генерувати з бази даних усі звіти щодо

вразливостей, аналізувати результати у словник, що трансформується у CSV-файл, який буде прикріплено до електронного листа для відправлення експерту (рис. В.3.6).

Щоб забезпечити фахівцям з безпеки ІБ доступ до детальної інформації про виявлені вразливості може бути створена панель управління для візуалізації результатів аналізу (підсистема перевірки експертом), що допоможе визначити пріоритети з усунення: з’ясувати, наскільки критичною є кожна вразливість для конкретної системи, наприклад, для банківської системи надзвичайне значення має порушення конфіденційності.

Модуль управляння може бути інтегрований з системою SIEM, що підвищить захищеність системи і продемонструє цілісний підхід до захисту: проактивна ідентифікація (виявлення потенційних вразливостей методами машинного навчання, що дозволяє вчасно їх усунути); неперервний моніторинг (SIEM-системи контролюють середовище на наявність аномальної поведінки); поліпшене реагування на інциденти (інформація, одержана в результаті роботи моделі ML, може бути використана в правилах кореляції SIEM, що підвищує точність виявлення).

Отже, підхід, який ґрунтується на інтеграції з SIEM-системами для моніторингу безпеки, дозволить не лише виявити зміни, а й зрозуміти їхній вплив та знайти потенційні загрози. За допомогою SIEM системи можна забезпечити постійний моніторинг і аналіз подій в комп'ютерній системі (рис. В.3.7).

Одним з підходів інтеграції з SIEM може бути Syslog – надсилання подій як syslog-повідомлень на порт SIEM. Наведена на рис. В.3.8 схема надає уявлення про потік даних і функціональні задачі кожного з модулів.

Механізм виявлення вразливостей має створити умови для підвищення точності та ефективності системи виявлення вразливостей.

Важливою особливістю проєктованого механізму є те, що він включає в себе рішення, яке демонструє принцип об’єднання технологій, орієнтованих на безпеку, відомий як Security Orchestration, Automation, and Response (SOAR).

3.4 Виявлення вразливостей системного реєстру та тренди у сфері машинного навчання

Застосування машинного навчання для пошуку вразливостей реєстру ОС дозволяє виявляти приховані вразливості та передбачати загрози. Це важливо, оскільки вразливості постійно змінюються і їх кількість зростає.

Важливими аспектами є: підвищення точності виявлення вразливостей, оскільки машинне навчання мінімізує можливість людських помилок; можливість виявлення вразливостей, які могли б залишитися непоміченим при ручному аналізі; швидкість оцінки, так як моделі машинного навчання здатні провести сканування своєчасно, що критично для таких ресурсів, як операційна система; здатність прогнозувати потенційні вразливості шляхом аналізу історичних даних, що надає можливості для вжиття превентивних заходів; адаптивність моделей ML дає змогу системам безперервно вдосконалюватися, еволюціонувати, адаптуючись до новітніх технік атак та вразливостей, що надає перевагу над зловмисником.

Одночасно машинне навчання стикається з проблемами. Ефективність роботи механізму виявлення вразливостей у системному реєстрі з використанням моделей ML залежить від якості даних, на яких навчені моделі. Якщо навчальні дані є неповними, алгоритм може не виявити певні типи вразливостей, що може призвести до переоцінки або до недооцінки ризиків безпеки операційної системи. Тому можливим удосконаленням механізму виявлення вразливостей у реєстрі ОС для підвищення її захищеності може бути розширення бази ключів, які підлягають аналізу, застосування інтеграції моделі машинного навчання з іншими елементами системи аналізу захищеності, наприклад, інтеграція з встановленим антивірусним сканером та сканером безпеки, що дозволить не лише виявити зміни, а й зрозуміти їхній вплив на систему.

Проблемою є помилкові спрацьовування, тому модуль перевірки прийнятих моделлю ML рішень є обов’язковим елементом механізму виявлення вразливостей. Еволюція вразливостей вимагає оновлень і перенавчання моделей.

Ймовірно, у майбутньому ключові напрямки, які будуть використовувати для розробки механізмів виявлення вразливостей у реєстрі методами машинного навчання, передбачатимуть: моделі ML розширюватимуть можливості експертів, але не можуть повністю замінити; алгоритми ML будуть адаптуватися до нових загроз завдяки постійному навчанню на реальних інцидентах і зможуть виявляти вразливості нульового дня, виявляючи аномалії, що можуть вказувати на них, надаючи додатковий рівень захисту; інтеграція моделей ML з квантовими обчисленнями значно пришвидшить аналіз і оцінку вразливостей.

Механізм виявлення вразливостей системного реєстру методами ML може підвищити захищеність операційної системи, що вкрай необхідно в умовах постійно змінюваних кіберзагроз. Оптимізувати реагування на інциденти та покращити загальне управління безпекою допоможе інтеграція з існуючими платформами кібербезпеки: встановленими засобами аналізу захищеності, встановленими сканерами, SIEM системами.

Висновок за розділом 3

У третьому розділі окреслені вимоги до механізму виявлення вразливостей у реєстрі операційної системи методами машинного навчання; розглянуто ключові етапи розробки методу автоматизованого виявлення вразливостей з використанням машинного навчання; визначена важливість вибору ефективних методів машинного навчання для аналізу виявлених у реєстрі аномалій, необхідність розробки комплексної архітектури системи виявлення вразливостей, оцінки ризиків, а також інтеграція цих методів з системою SIEM.

Розроблено та проаналізовано структуру механізму, який включає модулі збору даних, попередньої обробки даних, машинного навчання, оцінки, що забезпечує ефективне та систематичне виявлення вразливостей.

Здійснено вибір алгоритмів машинного навчання для класифікації аномалій і класифікації виявлених вразливостей за критеріями безпеки,

розроблено комплексну структурну схему механізму і програмно реалізовано виявлення і класифікацію вразливостей системного реєстру методами ML.

Розроблений механізм виявлення вразливостей системного реєстру методами машинного навчання є корисним для підвищення захищеності операційної системи. Використання машинного навчання вдосконалює процес виявлення і усунення вразливостей, робить систему безпеки більш адаптованою в умовах постійно змінюваних кіберзагроз.

Також проаналізовані переваги, виклики і тенденції використання механізму виявлення вразливостей у реєстрі ОС, що підкреслює важливість інтеграції технологій машинного навчання з традиційними методами для створення ефективних та надійних систем виявлення вразливостей.

ВИСНОВКИ

В рамках кваліфікаційної роботи було:

– проведено аналіз типів вразливостей та їхнього впливу на стан безпеки інформаційної системи;

- розглянуто існуючі засоби аналізу захищеності інформаційних систем;

- досліджено структуру, особливості і методи аналізу реєстру операційної системи Windows;

- проаналізовано методи машинного навчання, алгоритми та архітектури, які використовують для побудови моделей для виявлення вразливостей системного реєстру;

- розроблено механізм для створення моделі машинного навчання, який буде здійснювати аналіз реєстру ОС і класифікувати виявлені вразливості за критеріями інформаційної безпеки.

У ході дослідження розглянуто потенційні вразливості на рівні операційної системи, зокрема, реєстру ОС. Наслідками впливу типових вразливостей реєстру ОС на безпеку ІС є порушення конфіденційності, цілісності, доступності.

Визначено, що виявлення вразливостей реєстру операційної системи до того, як вони будуть використані, стає все більш необхідним завданням і є однією з важливих технологій захисту ОС. Розглянуто структуру, функції і особливості даних системного реєстру, які роблять його важливим ресурсом для проведення аналізу, методи та інструменти для збору даних і виявлення вразливостей.

У кваліфікаційній роботі проведено дослідження останніх наукових напрацювань з питань використання методів ML для виявлення і класифікації вразливостей реєстру ОС Windows з метою визначення рішень, що можна застосувати для підвищення захищеності операційної системи у ході активних процесів. Результатом аналізу є узагальнення останніх тенденцій досліджень стосовно виявлення вразливостей на основі машинного навчання.

Доведена необхідність розробки механізму виявлення вразливостей у середовищі системного реєстру саме на основі ML, оскільки воно володіє певними можливостями для створення ефективних рішень вирішення проблеми підвищення захищеності реєстру операційної системи.

Кваліфікаційна робота зосереджена на розробці механізму виявлення вразливостей на основі моделей ML, який зможе автоматично виявляти вразливості реєстру ОС. За результатами досліджень запропоновано модель ML, яка програмно реалізована в частині виявлення вразливостей і їхньої класифікації за критеріями безпеки.

Розроблена архітектура механізму виявлення вразливостей реєстру ОС враховує: переваги використання машинного навчання у завданнях кібербезпеки; можливі рішення підвищення захищеності операційної системи Windows; важливість інтеграції моделі ML з іншими системами безпеки, щоб досягти кращих результатів у виявленні вразливостей реєстру операційної системи.

Майбутнім покращенням у процесі подальшої практичної реалізації механізму може бути інтеграція з іншими системами для автоматичного виправлення виявлених вразливостей, що дозволить виконати більш точну оцінку вразливостей і підвищити захищеність операційної системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Microsoft Vulnerabilities Report 2025. The latest Microsoft Vulnerabilities data, analyzed. URL: https://www.beyondtrust.com/resources/whitepapers/microsoft-vulnerability-report.

2. Термінологія в галузі захисту інформації в комп'ютерних системах від несанкціонованого доступу. НД ТЗІ 1.1-003-99 Наказ Департаменту спеціальних телекомунікаційних систем та захисту інформації Служби безпеки України від "28" квітня 1999 р. № 22. URL: https://tzi.ua/assets/files/1.1_003_99.pdf.

3. Про затвердження Порядку пошуку та виявлення потенційної вразливості інформаційних (автоматизованих), електронних комунікаційних, інформаційно-комунікаційних систем, електронних комунікаційних мереж. Постанова Кабінету Міністрів України від 16 травня 2023 р. № 497. URL: https://zakon.rada.gov.ua/laws/show/497-2023-%D0%BF#Text.

4. Гахов С. О. Застосування положень імунології в теорії захищених інформаційних систем. Сучасний захист інформації. 2018. № 2. С. 59 – 64. URL: https://journals.dut.edu.ua/index.php/dataprotect/article/view/1902/1805.

5. Грайворонській М.В. Безпека інформаційно-комунікаційних систем: підручник. 2009. URL: https://ela.kpi.ua/handle/123456789/44867.

6. Микитишин А.Г., Митник М.М., Голотенко О.С., Карташов В.В.. Комплексна безпека інформаційних мережевих систем: навчальний посібник, 2023.. URL: https://elartu.tntu.edu.ua/handle/lib/42625.

7. Zero-day vulnerability. URL: https://en.wikipedia.org/wiki/Zero-day_vulnerability.

8. ISO/IEC 15408-1 Information technology — Security techniques — Evaluation criteria for IT security Part 1: Introduction and general model. URL: https://www.iso.org /standard/72891.html.

9. Vulnerability in Cyber Security: A Comprehensive Guide. URL: https://trainings.internshala.com/blog/vulnerability-in-cyber-security/.

10. Яцків В.В., Якименко І.З Опорний конспект лекцій з дисципліни «Моніторинг мережевої безпеки» для студентів напряму підготовки кібербезпека‖ освітнього рівня магістр» / Тернопільський національний економічний університет, 2019. URL: http://dspace.wunu.edu.ua/bitstream/316497/36006/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%97_%D0%9C%D0%9C%D0%91.pdf.

11. Vulnerability database. URL: https://en.wikipedia.org/wiki/Vulnerability_database.

12. Known Exploited Vulnerabilities Catalog. URL: https://www.cisa.gov/known-exploited-vulnerabilities-catalog.

13. CVE Abstraction Content Decisions: Rationale and Application URL: https://cve.mitre.org/cve/editorial_policies/cd_abstraction.html.

14. Common Weakness Enumeration A community-developed list of SW & HW weaknesses that can become vulnerabilities. URL: https://cwe.mitre.org/data/index.html.

15. Security Bulletins. URL: https://learn.microsoft.com/en-us/security-updates/securitybulletins/securitybulletins.

16. Common Vulnerability Scoring System v3.1: Specification Document. URL: https://www.first.org/cvss/specification-document.

17. 2024 Cybersecurity Almanac: 100 Facts, Figures, Predictions And Statistics. URL: https://cybersecurityventures.com/cybersecurity-almanac-2024/.

18. ISO/IEC 27001 «Information technologies. Protection methods. Information security management system. Requirements». URL: https://www.iso.org/standard/27001.

19. Загальні положення щодо захисту інформації в комп'ютерних системах від несанкціонованого доступу. НД ТЗІ 1.1-002-99. Наказ Департаменту спеціальних телекомунікаційних систем та захисту інформації Служби безпеки

України від 28 квітня 1999 р. № 22. URL: https://tzi.com.ua/downloads/1.1-002-99.pdf.

20. Merkow, Mark S. "Testing Part 2: Penetration Testing/Dynamic Analysis/IAST/RASP." Auerbach Publications (2021). URL: URL:http://dx.doi.org/10.1201/9781003265566-9.

21. Dis Assembly. URL: https://www.program-transformation.org/Transform/DisAssembly.

22. Setiawan, Hermawan, et al. "Vulnerability Analysis Using The Interactive Application Security Testing (IAST) Approach For Government X Website Applications." IEEE (2020). URL: http://dx.doi.org/10.1109/icoiact50329.2020.9332116.

23. Setiawan, Hermawan, et al. "Vulnerability Analysis Using The Interactive Application Security Testing (IAST) Approach For Government X Website Applications." IEEE (2020). URL: http://dx.doi.org/10.1109/icoiact50329.2020.9332116.

24. Software composition analysis (SCA). URL: https://www.invicti.com/learn/software-composition-analysis-sca/.

25. Фазинг. URL: https://uk.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B7%D0%B8%D0%BD%D0%B3.

26. Nessus Vulnerability Scanner. URL: https://www.tenable.com.

27. OpenVAS - Open Vulnerability Assessment Scanner. URL: https://www.openvas.org.

28. Qualys. URL: https://www.qualys.com.

29. Metasploit Framework. URL: https://docs.rapid7.com/metasploit/msf-overview/.

30. StatCounter. 2024. Desktop Windows Version Market Share Worldwide for the period Oct 2023 - Oct 2024. Statcounter GlobalStats Report. URL: https://gs.statcounter.com/os-version-marketshare/windows/desktop/worldwide.

31. Windows registry information for advanced users 15/01/2025. URL: https://learn.microsoft.com/en-us/troubleshoot/windows-server/performance/windows-registry-advanced-users.

32. Баган Т. Г. Методи машинного навчання при проєктуванні автоматизованих систем керування: навч. посіб. для аспірантів спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» /; КПІ ім. Ігоря Сікорського. Електронні текстові дані (1 файл: 313 кБайт). Київ: КПІ ім. Ігоря Сікорського, 2021. 28 с. URL: https://ela.kpi.ua/items/8915cb38-efe2-4618-a218-f631365f2746.

33. Стьопочкіна І.В., Новіков О.М. Методи штучного інтелекту в кібербезпеці: навч. посіб. для здобувачів спец. 125 «Кібербезпека» / КПІ ім. Ігоря Сікорського; уклад: – Електронні текстові дані (1 файл: 19,9 Мбайт). – Київ: КПІ ім. Ігоря Сікорського, 2022 – 82 с. URL: https://ela.kpi.ua/items/bf50435a-bfc2-48ef-a400-d31ca2e99b06.

34. Савченко В. А., Шаповаленко О.Д. Основні напрями застосування технологій штучного інтелекту у кібербезпеці Сучасний захист інформації №4(44), 2020. URL: https://journals.dut.edu.ua/index.php/dataprotect/article/view/2456.

35. Синєглазов В, Чумаченко О. Методи та технології напівкерованого навчання: Курс лекцій Навчальний посібник. URL: https://ela.kpi.ua/server/api/core/bitstreams/214edd4c-0556-4c8b-b56b-1fd246c9def5/content.

36. T. Zhang, C. Xu, J. Shen, X. Kuang and L. A. Grieco, How to Disturb Network Reconnaissance: A Moving Target Defense Approach Based on Deep Reinforcement Learning,in IEEE Transactions on Information Forensics and Security, vol. 18, pp. 5735-5748, 2023. URL: https://doi.org/10.1109/TIFS.2023.3314219.

37. Amit Hariyani Forensic Evidence Collection From Windows Host Using Python Based Tool. 2022. URL: https://www.academia.edu/95237112/Forensic_Evidence_Collection_From_Windows_Host_Using_Python_Based_Tool.

38. Булатецький В., Булатецька Л., Гришанович Т. Аналіз файлових об’єктів операційної системи Windows 10 для очищення й оптимізації простору системного розділу, 2022. URL: https://www.csecurity.kubg.edu.ua/index.php/journal/article/view/336/279.

39. Registry Functions 05/04/2022. URL: https://learn.microsoft.com/en-us/windows/win32/sysinfo/registry-functions.

40. Trojan:Win32/Jorik.C Sep 15, 2017. URL: https://www.microsoft.com/en-us/wdsi/threats/malware-encyclopedia-description?Name=Trojan:Win32/Jorik.C.

41. L0phtCrack. URL: https://uk.wikipedia.org/wiki/L0phtCrack.

42. What Is AimRecover.exe? How To Repair It? [SOLVED]. URL: https://www.solvusoft.com/en/files/error-virus-removal/exe/windows/dark-bay-ltd/hacker-s-handbook/aimrecover-exe/.

43. From Registry With Love: Malware Registry Abuses . January 19, 2023. URL: https://www.splunk.com/en_us/blog/security/from-registry-with-love-malware-registry-abuses.html.

44. Soc Team. Detailed Analysis of CVE-2024-43452: Windows Registry Elevation of Privilege Vulnerability.12.11.2024. URL: https://ogma.in/detailed-analysis-of-cve-2024-43452-windows-registry-elevation-of-privilege-vulnerability.

45. Інформація щодо вразливості в продуктах Microsoft CVE-2022-21907 18.01.2022. URL: https://cert.gov.ua/article/17961. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-21907.

46. What is the Windows Event Log?. URL: https://www.solarwinds.com/resources/it-glossary/windows-event-log.

47. reg query. URL: https://learn.microsoft.com/ru-ru/windows-server/administration/windows-commands/reg-query.

48. Event Log Explorer. URL: https://eventlogxp.com/.

49. MyEventViewer. URL: https://soft.mydiv.net/win/download-MyEventViewer.html.

50. Regshot. URL: https://github.com/Seabreg/Regshot.

51. RegRipper4.0. URL: https://github.com/keydet89/RegRipper4.0.

52. YARA in a nutshell. URL: https://virustotal.github.io/yara/.

53. Mahdi Rabbani, Yongli Wang, Reza Khoshkangini, Hamed Jelodar, Ruxin Zhao, Sajjad Bagheri Baba Ahmadi and Seyedvalyallah Ayobi. A Review on Machine Learning Approaches for Network Malicious Behavior Detection in Emerging Technologies 2021. // – Resource access mode. URL: https://www.mdpi.com/1099-4300/23/5/529.

54. Choi, J., Park, J. and Lee, S., 2021. Forensic exploration on Windows File History. Forensic Science International: Digital Investigation, 36, p.301134. URL: https://www.sciencedirect.com/science/article/abs/pii/S2666281721000329?via%3Dihub.

55. Hodge, V. J. and Austin, J. A Survey of Outlier Detection Methodologies. October 2004 DOI:10.1023/B:AIRE.0000045502.10941.a9. URL: https://www.researchgate.net/publication/220638052_A_Survey_of_Outlier_Detection_Methodologies.

56. JPCERT-CC. Detecting Lateral Movement through Tracking Event Logs. Technical Report Version 2, Japan Computer Emergency Response Team Coordination Center, December 2017. URL: https://www.jpcert.or.jp/english/pub/sr/DetectingLateralMovementThroughTrackingEventLogs_version2.pdf.

57. Kayacik H. G., Zincir-Heywood A. N., Heywood M. I. Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD 99 Intrusion Detection Datasets. Proc. of the 3rd Annual Conference on Privacy, Security and Trust. 2005. URL: https://www.researchgate.net/publication/220919984_Selecting_Features_for_Intrusion_Detection_A_Feature_Relevance_Analysis_on_KDD_99.

58. Liu, Y.; Pi, D. A novel kernel svm algorithm with game theory for network intrusion detection. KSII Trans. Internet Inf. Syst. 2017, URL: https://itiis.org/digital-library/manuscript/1774.

59. Javed Asharf, Nour Moustaf, Hasnat Khurshid, Essam Debie, Waqas Haider and Abdul Wahab A Review of Intrusion Detection Systems Using Machine and Deep Learning in Internet of Things: Challenges, Solutions and Future Directions July 2020. URL: https://www.mdpi.com/2079-9292/9/7/1177.

60. Mahdi Rabbani, Yongli Wang, Reza Khoshkangini , Hamed Jelodar, Ruxin Zhao, Sajjad Bagheri Baba Ahmadi and Seyedvalyallah Ayobi. A Review on Machine Learning Approaches for Network Malicious Behavior Detection in Emerging Technologies. URL: https://www.mdpi.com/1099-4300/23/5/.

61. Eduard Kovacs FireEye MalwareGuard Uses Machine Learning to Detect Malware. // – Resource access mode: 2018. URL: https://www.securityweek.com/fireeye-malwareguard-uses-machine-learning-detect-malware/.

62. CrowdStrike Falcon FileVantage Empowers Teams to Pinpoint Potential Adversary Activity Through Central Visibility and Scalable File Integrity Monitoring October 12, 2021 at 3:15 PM EDT. URL: https://ir.crowdstrike.com/news-releases/news-release-details/crowdstrike-falcon-filevantage-empowers-teams-pinpoint-potential/.

63. Carnegie Mellon University CyLab. URL: https://www.linkedin.com/showcase/carnegiemelloncylab/.

64. Qiaokun Wen*, K.P. Chow CNN based zero-day malware detection using small binary segments DFRWS 2021. URL: https://www.sciencedirect.com/science/article/pii/S2666281721000263.

65. M Rabbani, Yongli Wang, Reza Khoshkangini A Review on Machine Learning Approaches for Network Malicious Behavior Detection in Emerging Technologies. 2021. URL: https://www.mdpi.com/1099-4300/23/5/529.

66. Beginner's Guide to Decision Trees for Supervised Machine Learning. URL: https://www.quantstart.com/articles/Beginners-Guide-to-Decision-Trees-for-Supervised-Machine-Learning/.

67. Dunsin, D., Ghanem, M.C., Ouazzane, K. and Vassilev, V., 2024. Reinforcement learning for an efficient and effective malware investigation during

cyber Incident response. High-Confidence Computing. URL: https://arxiv.org/abs/2408.01999.

68. Z. A. El Houda, A. S. Hafid, and L. Khoukhi, ‘‘A novel machine learning framework for advanced attack detection using SDN,’’ in Proc. IEEE Global Commun. Conf. (GLOBECOM), Dec. 2021, pp. 1–6. URL: https://www.researchgate.net/publication/354020737_A_Novel_Machine_Learning_Framework_for_advanced_Attack_Detection_Using_SDN.

69. Мокін В. Б., Дратований М. В. Наука про дані: машинне навчання та інтелектуальний аналіз даних – Електронний навчальний посібник /– Вінниця: ВНТУ, 2024. URL: https://pdf.lib.vntu.edu.ua/books/2024/Mokin_2024_263.pdf.

70. Dunsin, D., Ghanem, M.C., Ouazzane, K., Vassilev, V., 2024. A comprehensive analysis of the role of artificial intelligence and machine learning in modern digital forensics and incident response. Forensic Science International. Digital Investigation 48, 301675. URL: https://doi.org/10.1016/j.fsidi.2023.301675 https://www.sciencedirect.com/science/article/pii/S2666281723001944?via%3Dihub.

71. Jin Z., Yu Y. Current and Future Research of Machine Learning Based Vulnerability Detection. Conference: 2018. URL: https://www.researchgate.net/publication/340231684_Current_and_Future_Research_of_Machine_Learning_Based_Vulnerability_Detection.

72. Sadegh Bamohabbat Chafjiri, Phil Legg, Jun Hong, Michail-Antisthenis Tsompanas. Vulnerability detection through machine learning-based fuzzing: A systematic review. doi.org/10.1016/j.cose.2024.103903 August 2024. URL: https://www.sciencedirect.com/science/article/pii/S0167404824002050.

73. Z. Zhang, H. Al Hamadi, E. Damiani, F. Taher. Explainable Artificial Intelligence Applications in Cyber Security: State-of-the-Art in Research. 2022. URL: https://www.researchgate.net/publication/363313465_Explainable_Artificial_Intelligence_Applications_in_Cyber_Security_State-of-the-Art_in_Research.

74. Liu, F., Ting, K., Zhou, Z. Isolation-Based Anomaly Detection 01 March 2012. URL: https://www.lamda.nju.edu.cn/publication/tkdd11.pdf

ДОДАТОК А

ВРАЗЛИВОСТІ ІНФОРМАЦІЙНИХ СИСТЕМ І СУЧАСНІ МЕТОДИ ЇХ ВИЯВЛЕННЯ

Таблиця А.1.1 – Класифікація вразливостей інформаційних систем

Чинник класифікації	Вид вразливості
За типом середовища	- вразливості програмного забезпечення; - вразливості обладнання (недоліки у фізичних компонентах системи); - вразливість мережі; - вразливості через поведінку користувача
За типом ПЗ	- вразливості системного ПЗ; - вразливості прикладного ПЗ
За типом життєвого циклу ПЗ	- вразливості, що виникли на етапі проєктування; - вразливості, що виникли на етапі реалізації; - вразливості, які є наслідком процесу експлуатації ІС
За причиною виникнення вразливості	- недоліки механізму аутентифікації; - недоліки захисту облікових даних; - наявність функцій для деструктивних дій; - відсутність перевірки достовірності даних; - особливості побудови програмних і апаратних засобів захисту
За характером наслідків від реалізації атаки	- вразливості, що використовують з метою переповнення буферу; - вразливості, що використовують для підбору пароля, ідентифікатора; - вразливості, що використовують з метою зміни прав доступу; - вразливості, що використовують для атаки «відмова в обслуговуванні»

Продовження додатку А

Таблиця А.1.2 – Функціональні можливості баз даних вразливостей

База даних	Функціональні можливості для використання
CVE (Common Vulnerabilities and Exposures)	Стандартизована БД загальновідомих вразливостей ІБ, система ідентифікації, підтримувана MITRE; полегшує обмін даними між окремими базами вразливостей та інструментами ІБ; кожній вразливості присвоюється ідентифікаційний номер, що містить рік виявлення і номер вразливості, опис і ряд загальнодоступних посилань, коментарі розробників.
NVD (National Vulnerability Database)	Інформаційна БД вразливостей Національного інституту стандартів і технологій США (NIST); містить додаткову інформацію до кожної CVE, яка допомагає визначити категорію вразливості; інтегрує всі бази вразливостей, доступні в США (відомості з бази даних CWE, дані щодо оцінки ризиків з бази CVSS, інформацію CPE для ідентифікації вразливого ПЗ і відомості про випуск постачальником ПЗ виправлення для вразливості).
CWE (Common Weakness Enumeration)	Класифікатор недоліків безпеки програмного забезпечення; підтримує організація MITRE; БД використовують для оцінки інструментів перевірки безпеки ПЗ і як загальний базовий стандарт для ідентифікації, попередження та пом’якшення негативних наслідків.
Microsoft Security bulletin	База даних містить інформацію тільки про вразливості Microsoft продуктів. Розглядаються уразливості безпеки в ПЗ Microsoft, описуються способи їх усунення та надаються посилання на застосовні оновлення для порушеного ПЗ

Таблиця А.1.3 – Підсистеми захисту ОС

Функції підсистем КЗЗ	Функціональні можливості
Керування політикою безпеки ОС	Підтримує відповідні політики безпеки (надає інтерфейси для налаштування підсистем розмежування доступу, ідентифікації й автентифікації, аудиту).
Ідентифікація й автентифікація	Полягає у процедурі встановлення належності користувачеві ідентифікатора, наприклад, облікових даних для отримання доступу до системи.
Розмежування доступу	Надає кожному користувачеві доступ лише до тих захищених об’єктів, до яких цей доступ дозволено на підставі ПРД, які є частиною політики безпеки.
Реєстрація й облік (аудит)	Реєструє усі потенційно небезпечні події, перевіряє логи, процедури та записи систем на їх достовірність та відповідність критеріям та показникам безпеки.
Криптографічні функції	Здійснюють захист конфіденційності та цілісності інформації при реалізації послуги ідентифікації та автентифікації і як самостійні засоби захисту або як допоміжні механізми в інших засобах.
Забезпечення цілісності	Містить додаткові засоби для захисту цілісності даних від НСД, випадкових помилок, аварійних ситуацій і збоїв у системі: здійснення функції відкату, автоматизації процесу створення резервних копій і відновлювання.
Антивірусний захист	Здійснює сканування системи для знаходження небезпечних файлів, шкідливих програм. Може виявляти і знищувати інфіковані файли.
Апаратні засоби	Заборона несанкціонованого (неавторизованого) зовнішнього доступу віддаленого користувача; заборона несанкціонованого (неавторизованого) внутрішнього доступу до баз даних в результаті випадкових чи умисних дій персоналу; захист цілісності програмного забезпечення.

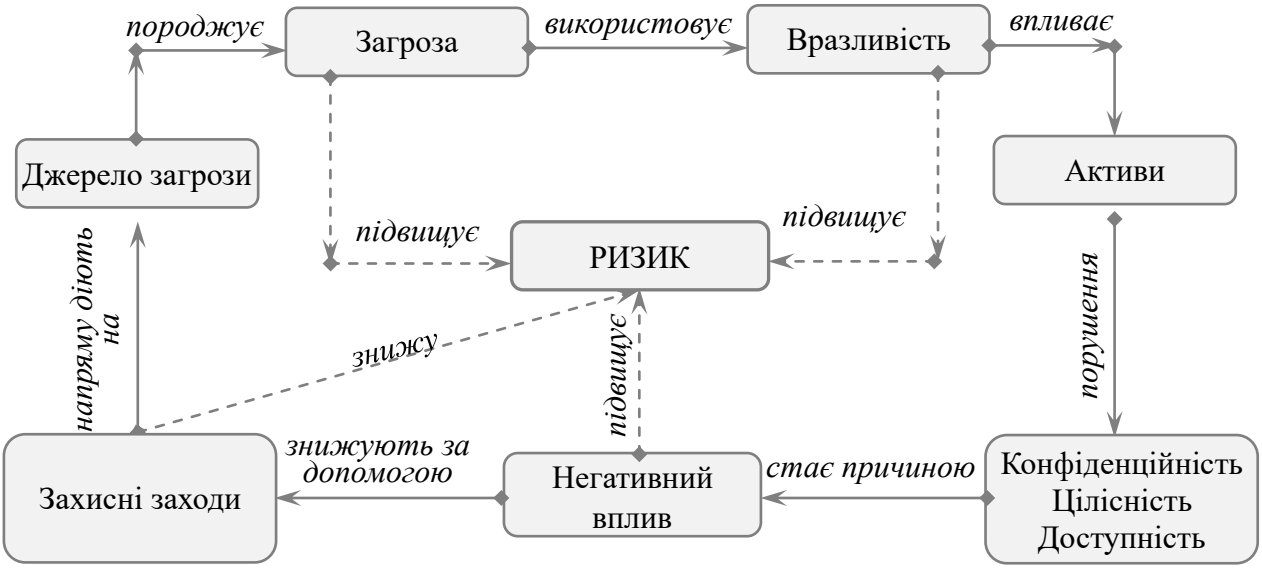

Рисунок А.3.2 – Взаємодія між загрозами, вразливостями і ризиком

Таблиця А.1.4 – Основні методи, які застосовують при проведенні аудиту

Метод	Характеристика
Аналіз конфігурацій (Configuration Auditing)	Є важливим процесом у виявленні та усуненні потенційних проблем безпеки Помилки в конфігурації можуть стати вразливістю, що піддає систему атакам.
Аналіз журналів (Log Analysis)	Дозволяє виявити потенційні аномалії, що можуть свідчити про спроби атаки чи інші проблеми безпеки. Наприклад, багаторазово повторювані неуспішні спроби входу або підозрілі мережеві запити).
Аудит прав доступу (Access Control Auditing)	Дозволяє здійснити перевірку налаштувань прав доступу користувачів до різних ресурсів. Аудит прав доступу допомагає у виявленні надмірних або неправильних привілеїв.
Аналіз мережевого трафіку (Network Traffic Analysis)	Є інструментом для моніторингу мережі, який може допомогти виявити аномальні патерни трафіку. Це може свідчити про експлуатацію вразливостей або атакуючі дії, такі як SQL-ін’єкції, DDoS-атаки тощо.

Продовження додатку А

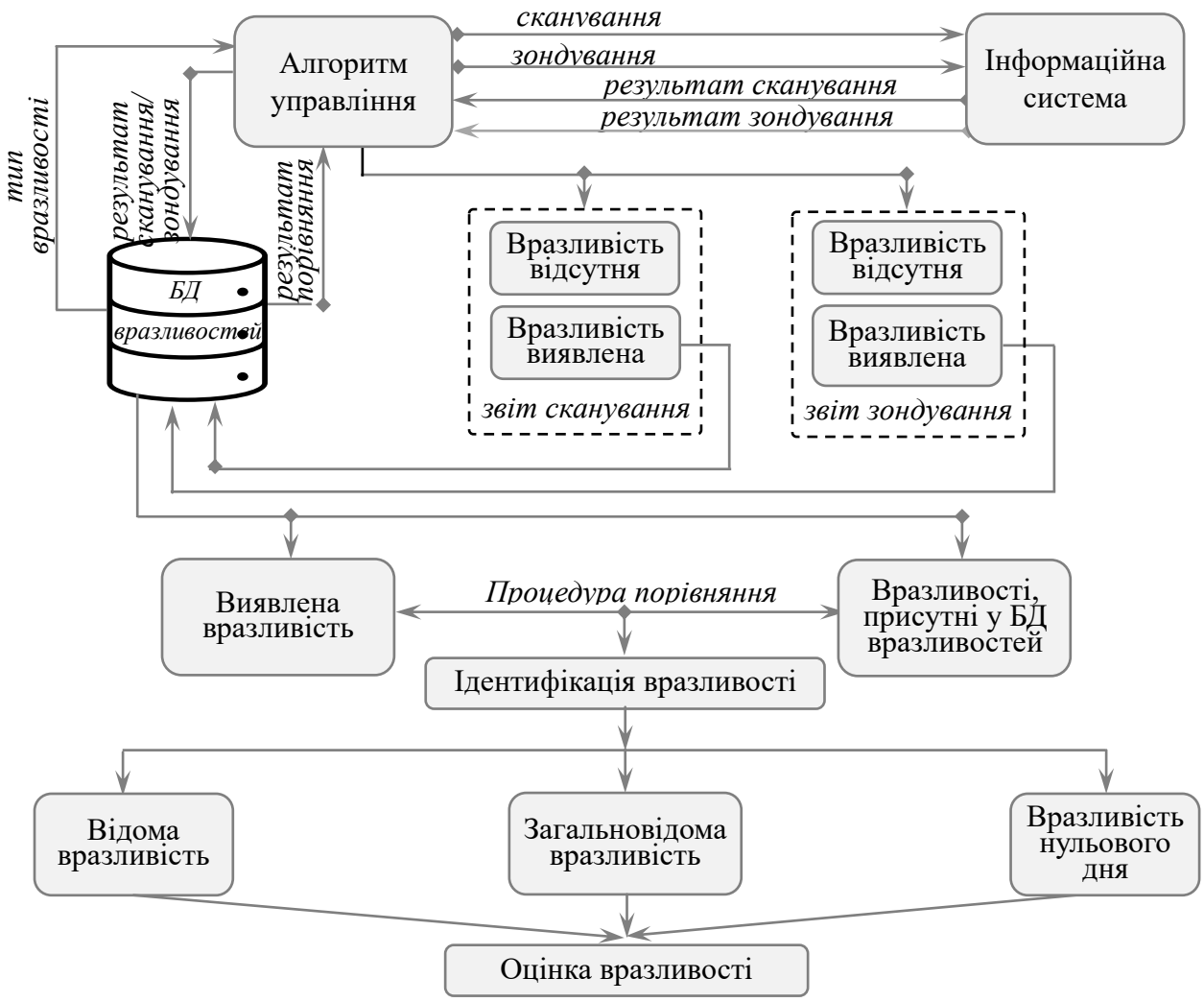

Рисунок А.3.3 – Узагальнена схема виявлення, ідентифікації і оцінки вразливостей

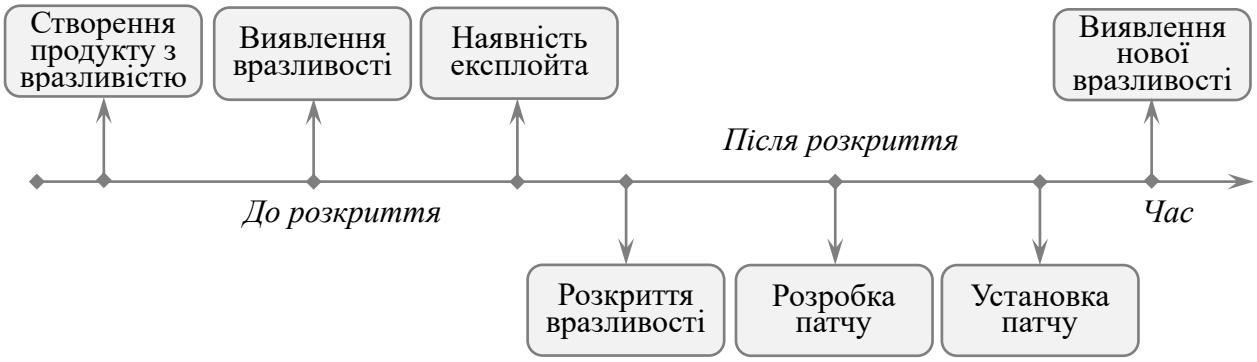

Рисунок А.3.4 – Життєвий цикл вразливостей

Таблиця.А.1.5 – Опис кореневих розділів Реєстру Windows

Ім’я кореневого розділу	Опис
HKEY_CURRENT_USER (HKCU)	Розділ містить налаштування, специфічні для користувача, який увійшов у систему локально: параметри конфігурації; персоналізація інтерфейсу і налаштування програм; які використовуються тільки цим користувачем.
HKEY_USERS (HKU)	Розділ містить відомості щодо завантажених профілів користувачів та профіль, використовуваний за замовчуванням. Це дозволяє зберігати конфігурацію для різних облікових записів користувачів, навіть якщо вони не активні в даний момент. Віддалені користувачі не мають профілів в цьому розділі сервера; їх профілі знаходяться в реєстрах власних комп’ютерів.
HKEY_LOCAL_MACHINE (HKLM)	Розділ зберігає відомості про локальний комп’ютер (включно з даними про обладнання і операційну систему, такі як тип шини, загальний обсяг доступної пам’яті, список драйверів пристроїв і параметри завантаження Windows).
HKEY_CLASSES_ROOT (HKCR)	Розділ містить відомості, які використовуються різними технологіями OLE, і дані про зіставлення типів файлів.
HKEY_CURRENT_CONFIG (HKCC)	Розділ містить відомості про конфігураційні дані для апаратного забезпечення, використовуваного локальним комп’ютером під час запуску системи. Ці параметри актуальні лише для поточної конфігурації апаратного забезпечення.

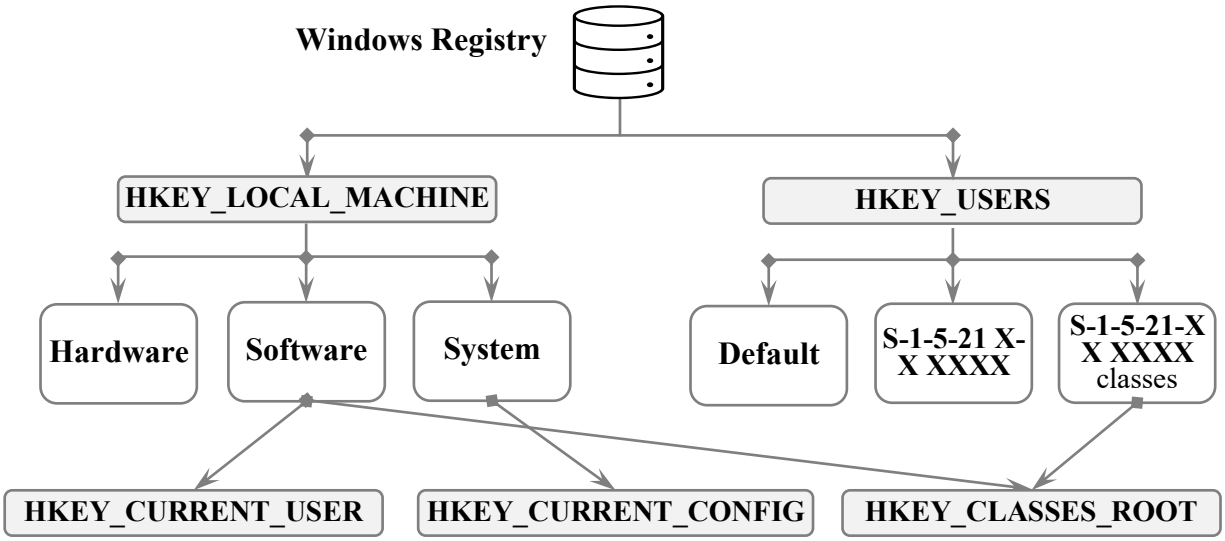

Рисунок А.3.5 – Зв’язок між кореневими ключами реєстру Windows

Таблиця А.1.6 – Зв’язок стандартних кущів реєстру з відповідними їм файлами

Назва куща реєстру	Імена файлів
HKEY_LOCAL_MACHINE \SYSTEM	%systemroot%\system32\config\system %systemroot%\sysWoW64\config\system
HKEY_LOCAL_MACHINE \SAM	%systemroot%\system32\config\sam %systemroot%\ sysWoW64\config\sam
HKEY_LOCAL_MACHINE \SECURITY	%systemroot%\system32\config\security %systemroot%\ sysWoW64\config\security
HKEY_LOCAL_MACHINE \SOFTWARE	%systemroot%\system32\config\software %systemroot%\ sysWoW64\config\software
HKEY_USERS\UserProfile	%systemdrive%\Users\%username%
HKEY_USERS.DEFAULT	%systemroot%\system32\config\default %systemroot%\ sysWoW64\config\default

Таблиця А.1.7 – Типи даних реєстру

Типи даних	Опис
REG_SZ	Рядковий (текстовий рядок фіксованої довжини)
REG_EXPAND_SZ	Розширений рядковий (рядок даних змінної довжини). Цей тип даних включає змінні, які обчислюються при використанні програми.
REG_BINARY	Двійковий, містить необроблені двійкові дані. Більшість відомостей про устаткування зберігається у вигляді двійкових даних і виводиться в редакторі реєстру в шістнадцятковому форматі.
REG_DWORD	Числовий, містить ціле число (4 байта), зазвичай служить як перемикач, де 0 – вимкнено, а 1 – увімкнено. Багато параметрів служб і драйверів пристроїв мають цей тип і відображаються в двійковому, шістнадцятковому або десятковому форматах
REG_LINK	рядковий, вказує шлях до файлу, це символьне посилання у форматі Unicode
REG_MULTI_SZ	Багаторядковий, визначає масив рядків. Цей тип, як правило, мають списки і інші записи в форматі, зручному для читання. Окремі значення розділяються пробілами, комами або іншими символами.
REG_QWORD	Дані, подані у вигляді 64-розрядного цілого числа (довжина – 8 байт)

ДОДАТОК Б

МЕТОДИ АНАЛІЗУ ТА ОЦІНКИ ЗАХИЩЕНОСТІ РЕЄСТРУ ОПЕРАЦІЙНОЇ СИСТЕМИ WINDOWS

Таблиця Б.2.1 – Приклади дій шкідливих програм

Шкідлива програма	Дія і спосіб створення аномальної діяльності реєстру
Програма Trojan: Win32/Jorik.C (за допомогою маскування приховує від користувача справжню мету своїх дій і таємно виконує додаткову шкідливу діяльність)	При запуску копіює себе в папку Temp під випадковим ім'ям; повідомляє на віддалений сервер про успішне зараження комп'ютера; для автоматичного запуску при перезавантаженні системи, він додає до реєстру Windows шлях до свого файлу в гілку: HKLM \Software\Microsoft\Windows\CurrentVersion\Run, що забезпечує виникнення вразливості; використовує її для створення нових ключів для власного використання програмою; отримує доступ до HKLM\Security\ Provider, що дозволяє йому знайти інформацію щодо безпеки машини для визначення вразливих місць.
Програма Hacker's Handbook	Під час запуску програми виникають помилки AimRecover.exe, що можуть бути результатом зараження зловмисним ПЗ. У результаті виникають недопустимі записи реєстру Windows - недійсні посилання на шляху до файлів, що створені зловмисною програмою. Відстеження того, коли і де сталася помилка AimRecover.exe, є важливою частиною інформації для усунення вразливостей
Програма LOpht Crack (застосунок для перевірки стійкості та відновлення паролів)	Інструмент злому паролів для машин Windows: отримує хешований файл SAM, що містить паролі для всіх користувачів; використовує словник або підхід «грубої сили» для пошуку паролів - застосовує вразливості схеми шифрування Windows, що дозволяє їй виявляти деякі символи в паролі; програма створює у реєстрі власний розділ, який має складатися з багатьох запитів створення ключів та встановлення значень, що будуть розміщені на ключах. Виявити таку вразливість можливо: раніше ключі не існували на хост-машині, тому раніше не були ідентифікованими, їхня поява є аномальністю.

Продовження таблиці Б.2.1

Шкідлива програма	Дія і спосіб створення аномальної діяльності реєстру
Програма Hacker's Handbook	Під час запуску програми виникають помилки AimRecover.exe, що можуть бути результатом зараження зловмисним ПЗ. У результаті виникають недопустимі записи реєстру Windows – недійсні посилання на шляху до файлів, що створені зловмисною програмою. Відстеження того, коли і де сталася помилка AimRecover.exe, є важливою частиною інформації для усунення вразливостей
Програма Remcos RAT (поширений троян віддаленого доступу)	Remcos має багато вбудованих функцій, які використовували для збереження доступу та ідентифікації інформації про продукт.
Програма Trickbot	Носій зловмисного ПО, який дозволяє зловмисникам доставляти декілька типів корисних навантажень. Модуль ПЗ Trickbot збирає усі встановлені додатки на скомпрометованому хості, запитуючи реєстр .

Таблиця Б.2.2 – Ключі реєстру, важливі для безпечної роботи ОС

Ключі реєстру	Опис
HKLM \Software\Microsoft\Windows\ Currentversion\Run	Створює автозапуск програми
HKLM\SYSTEM\CurrentControlSet\ Services	Міститься інформація щодо автозавантаження драйверів і служб
HKCU\Software\Microsoft\Windows\ CurrentVersion\Policies	Міститься інформація щодо політики безпеки користувача
HKLM\SYSTEM\CurrentControlSet\ Control\Lsa	Міститься інформація щодо налаштування локальної безпеки
HKEY_LOCAL_MACHINE\SAM	Міститься інформація щодо імен локальних користувачів та їхніх зашифрованих паролів (дані доступні лише з підвищеними правами);
HKEY_LOCAL_MACHINE\Security	Міститься інформація, що відноситься до захисту ОС

Таблиця Б.2.3 – Приклади типових вразливостей реєстру ОС Windows

Тип вразливості	Наслідки для безпеки ОС	Аспект безпеки
Неправильні права доступу до ключів реєстру	Може дозволити неавторизованим користувачам читати або змінювати чутливі дані (змінювати критичні налаштування або виконувати шкідливі програми)	Конфіденційність
Відсутність автентифікації	Доступ до реєстру без перевірки особи – ризик розкриття або модифікації конфіденційних записів	Конфіденційність
Залишені вразливості від видалених програм	Старі ключі програм, якщо не будуть очищені, можуть бути використані для компрометації системи або обману служб безпеки	Цілісність
Неправильні налаштування політики безпеки в реєстрі	Політики безпеки можуть бути змінені зловмисником, що порушує цілісність системної конфігурації (користувачі можуть запускати небезпечні програми, вимикати віддалений доступ та контролювання доступу до певних функцій)	Цілісність
Вразливості в компонентах сторонніх виробників	Сторонні компоненти можуть містити шляхи доступу до конфіденційних даних без належного захисту	Конфіденційність
Помилкові конфігурації для автоматичного запуску програм	Дозволяє запуск шкідливого ПЗ при старті системи, порушуючи очікувану поведінку ОС	Цілісність
Вразливості, пов’язані з неправильними налаштуваннями системних служб Windows	Зловмисники можуть отримати доступ до системи, експлуатуючи недоліки в системних службах Windows, що може спричинити збої служб або відмову у наданні критичних функцій	Доступність

Продовження таблиці Б.2.3

Тип вразливості	Наслідки для безпеки ОС	Аспект безпеки
Неправильні записи, пов’язані з оновленнями Windows	Перешкоджання нормальному функціонуванню системи або дозвіл зловмиснику уникнути патчів безпеки. Некоректні оновлення можуть зламати систему або створити вразливості.	Цілісність
Вразливості через залишені старі версії драйверів або вразливі драйвери	Можливість експлуатації вразливостей, які вже були виправлені в нових версіях. Небезпечні драйвери можуть призвести до відмов або дестабілізації системи.	Доступність

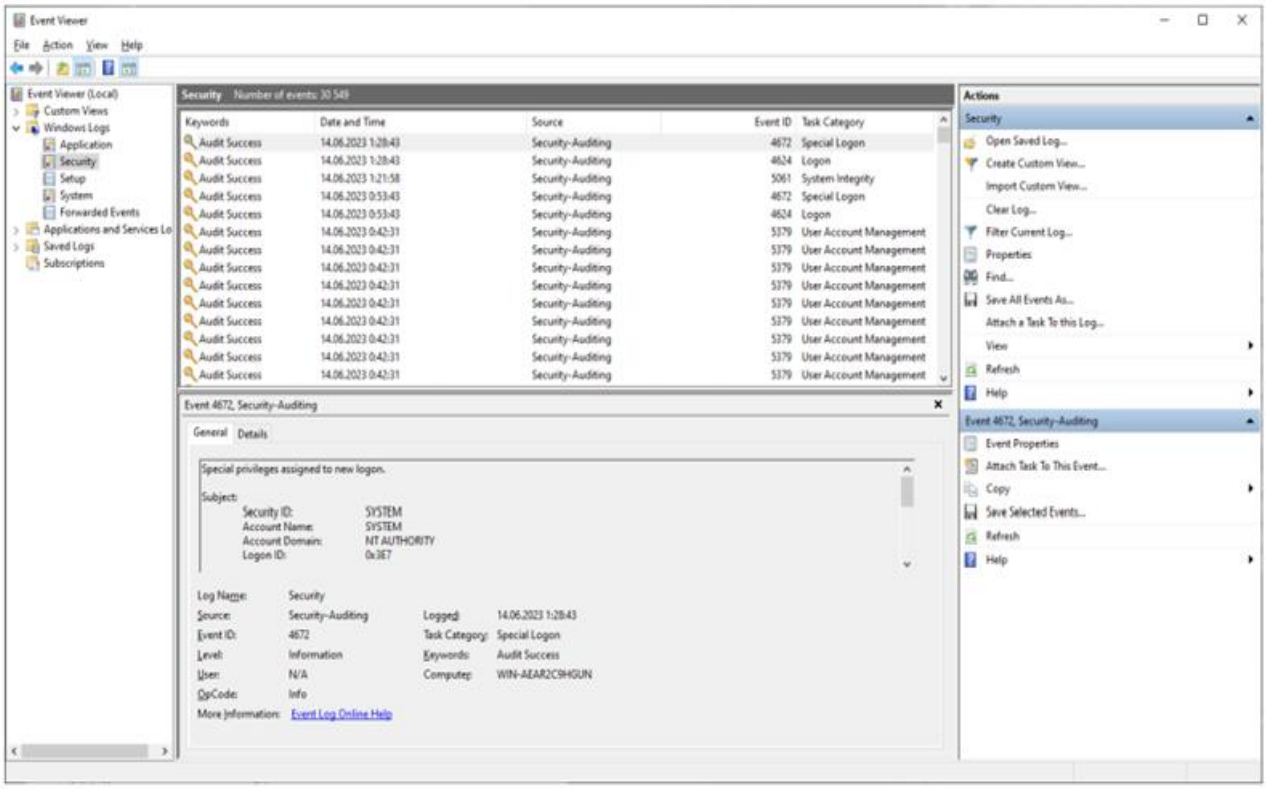

Рисунок Б.3.6 – Система журналювання подій Windows 10

Таблиця Б.2.4 – Типи подій

Тип подій	Опис подій
Повідомлення	Події, що описують вдале завершення дії застосуванням, драйвером або службою. Приклад події: Windows завантажує драйвер мережі, інцидент буде зареєстрований як повідомлення.
Попередження	Події вказують на проблеми, які не вимагають негайного втручання (наприклад, вичерпання ресурсів), але можуть призвести до складнощів в майбутньому. Приклад події: реєстрація такої проблеми, як брак місця на диску.
Помилка	Події вказують на серйозні труднощі, які зазвичай призводять до втрати функціональності або даних. Приклад події: неможливість запуску служби при завантаженні.
Успішний аудит	Події, які відповідають успішно завершеним діям, пов’язаними із підтримкою безпеки системи. Приклад події: успішний вхід користувача в систему.
Аудит відмови	Події, які відповідають невдало завершеній дії при зверненні до ресурсів, що проходять аудит. Приклад події: спроба відкрити файл, не маючи відповідних прав доступу, або невдала спроба доступу користувача до мережного диска.

Таблиця Б.2.5 – Методи аналізу системного реєстру

Тип	Характеристика
Ручний аналіз реєстру	Класичний підхід, коли фахівець вручну переглядає ключі та значення реєстру за допомогою вбудованого інструменту «Редактор реєстру» (regedit). Переваги: безпосередній доступ до всіх налаштувань, точність. Недоліки: це може бути дуже трудомістким, якщо реєстр великий і містить багато ключів
Використання інструментів для автоматизації аналізу реєстру	Спеціальні утиліти та програмні рішення для аналізу реєстру (Event Log Viewer , MyEventViewer, Regshot) можуть автоматично виділяти важливі зміни та надавати детальну інформацію для аналізу

Продовження таблиці Б.2.5

Тип	Характеристика
Аналіз змін реєстру на основі поведінкових ознак	Метод, при якому здійснюється моніторинг змін реєстру в реальному часі для виявлення підозрілих або аномальних змін, які можуть бути ознаками шкідливого ПЗ. Важливо налаштування аудиту змін у реєстрі через політики безпеки, використання інструментів, наприклад, Windows Event Viewer, для фіксації змін
Аналіз за допомогою сценаріїв PowerShell	Можна створювати власні сценарії PowerShell для моніторингу та аналізу реєстру. Наприклад, скрипти для пошуку змін у певних ключах реєстру або для збору інформації про конфігурацію системи
Порівняння версій реєстру	Метод, при якому використовують інструменти, які можуть допомогти порівняти різні версії реєстру, наприклад, після інсталяції нового ПЗ або зміни налаштувань. Порівнюючи версії, можна виявити ключові зміни в реєстрі, які можуть вказувати на можливі вразливості чи атаки
Інтеграція з іншими джерелами даних	Порівняння даних з реєстру з іншими джерелами, такими як файлові системи, мережевий трафік або процеси в пам’яті, для комплексного розслідування безпеки
Аналіз зловмисних змін у реєстрі	За допомогою методу аналізують шкідливі зміни у реєстрі шляхом: виявлення несанкціонованих автозапусків програм через ключі автозапуску (наприклад, HKCU\Software\Microsoft\Windows\CurrentVersion\Run) в та пошук підозрілих записів, які можуть свідчити про присутність шкідливого ПЗ, зміну політик безпеки або підміни налаштувань системи

Таблиця Б.2.6 – Результати систематизації огляду публікацій

Посилання	Рік	Метод ML	Результат
FireEye [64]	2018	Градієнтний бустинг дерев рішень (XGBoost)	Виявлення відхилень, які вказують на наявність rootkits або backdoors з точністю 94%
CrowdStrike [65]	2021	One-Class SVM	Ідентифікація рідкісних або незвичних змін з точністю 92%
CMU [66]	2021	RNN	Прогнозування атак, що використовували незвичайні зміни конфігурацій реєстру, з точністю 96%
Chow CNN based zero-day malware detection using small binary segments DFRWS [67]	2021	CNN	Ідентифікація й ізолювання підозрілих програм з точністю до 91,04%
A Review on Machine Learning Approaches for Network Malicious Behavior Detection in Emerging Technologies [68]	2021	Алгоритми ML Архітектури DL	Важливість попередньої обробки даних для отримання відповідних ознак, щоб забезпечити ефективність роботи моделі ML. Аналіз можливостей методів ML і DL у кібербезпеці
Beginner's Guide to Decision Trees for Supervised Machine Learning [69]	2021	Дерева рішень DTs	Використовують для завдань регресії і класифікації
Reinforcement learning for an efficient and effective malware investigation during cyber Incident response [70]	2024	KNN, DT, SVM, PCA, K-Means, SVD, NB, ANN, LR и RF	Необхідно використовувати переваги методів машинного навчання для аналізу реєстру операційної системи Windows

Продовження таблиці Б.2.6

Посилання	Рік	Метод ML	Результат
A novel machine learning framework for advanced attack detection using SDN [71]	2021	IsolationForest	Розроблена багатомодульна структура ML. Використовується нова схема виявлення викидів
Наука про дані: машинне навчання та інтелектуальний аналіз даних [72]	2024	Багінг і бустінг, IsolationForest	Ансамблі дерев рішень є найбільш ефективними. IsolationForest добре виявляє аномальні значення
A comprehensive analysis of the role of artificial intelligence and machine learning in modern digital forensics and incident response [73]	2024	Моделі MDP, які засновані на RL системах	Методи можуть класифікувати і визначати пріоритети зловмисним записам реєстру, що підвищує як швидкість, так і точність аналізу
Current and Future Research of Machine Learning Based Vulnerability Detection [74]	2018	Наївний Байєс, SVM, випадковий ліс, CNN, BLSTM, RNN	Методи пошуку вразливостей на основі ML мають високий потенціал (95%). Нейронні мережі перевершують результативність усіх інших методів.
Vulnerability detection through machine learning-based fuzzing: A systematic review [75]	2024	SVM, GAN, LSTM	Алгоритми SVM частіше інших використовують в якості класифікаторів у фазингу. Найбільш популярними є архітектури GAN і LSTM

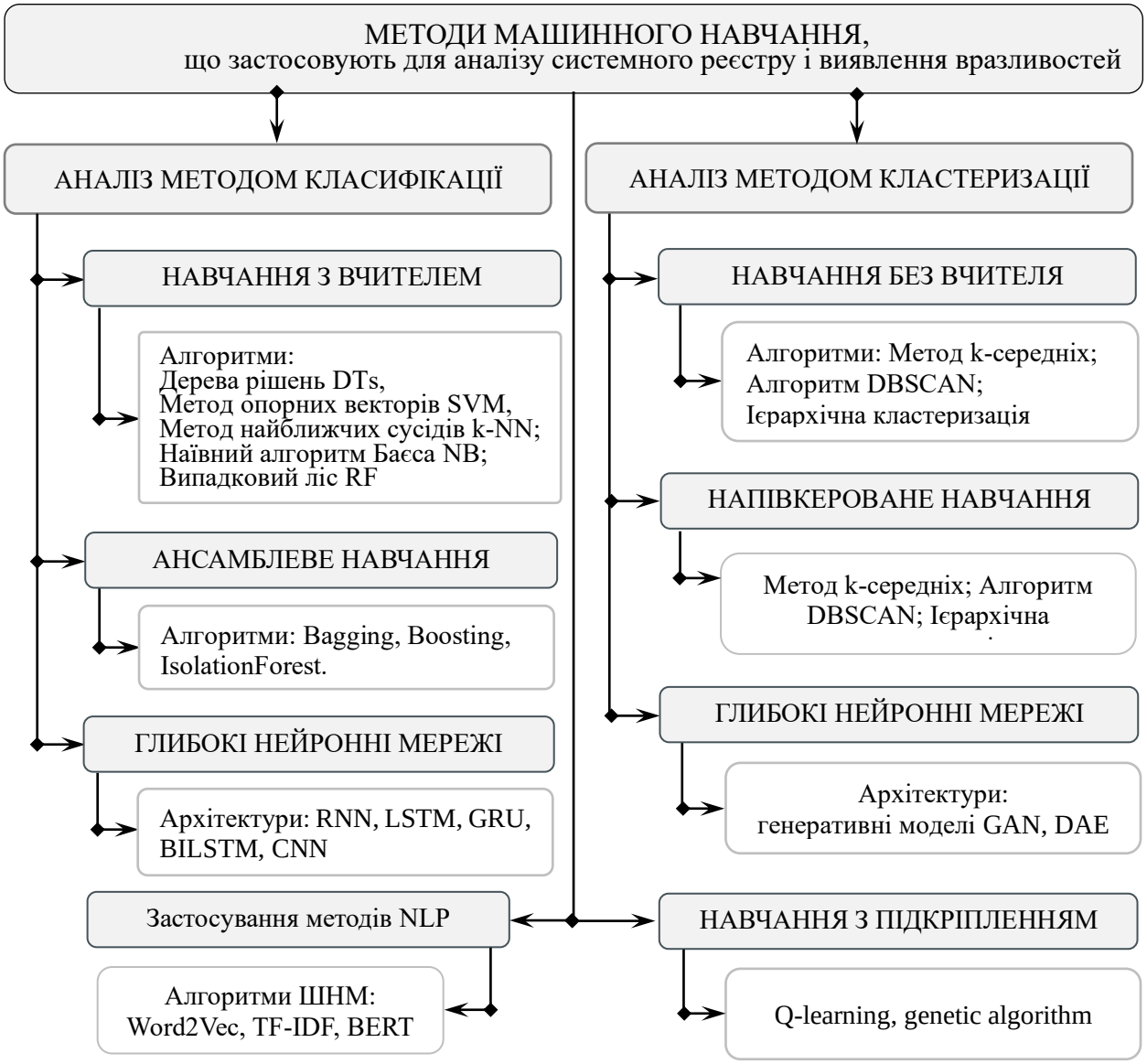

Рисунок Б.3.2 – Методи ML для аналізу реєстру і виявлення вразливостей

ДОДАТОК В

ПРОЄКТУВАННЯ МОДЕЛІ ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ

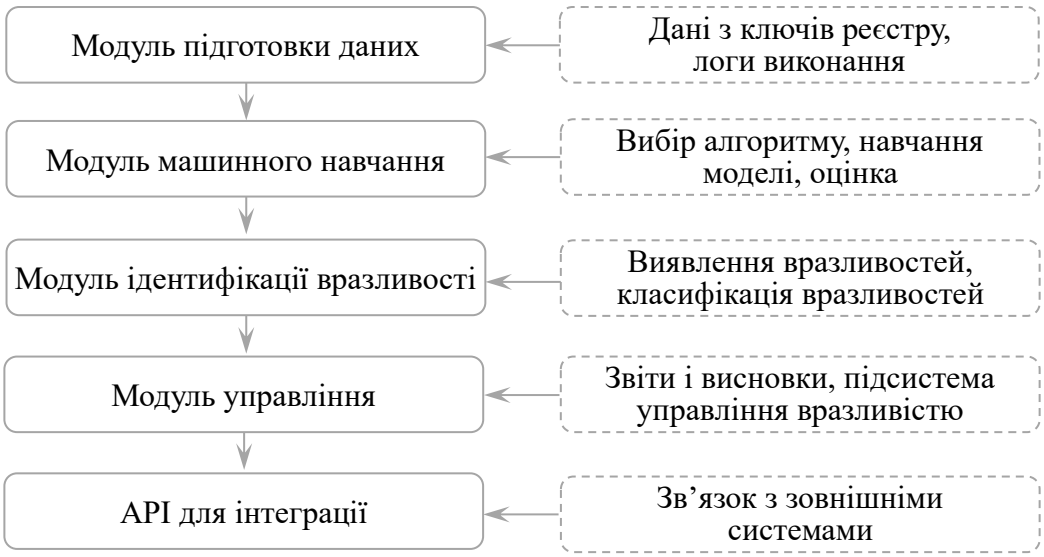

Рисунок В.3.7 – Ключові компоненти механізму виявлення вразливостей

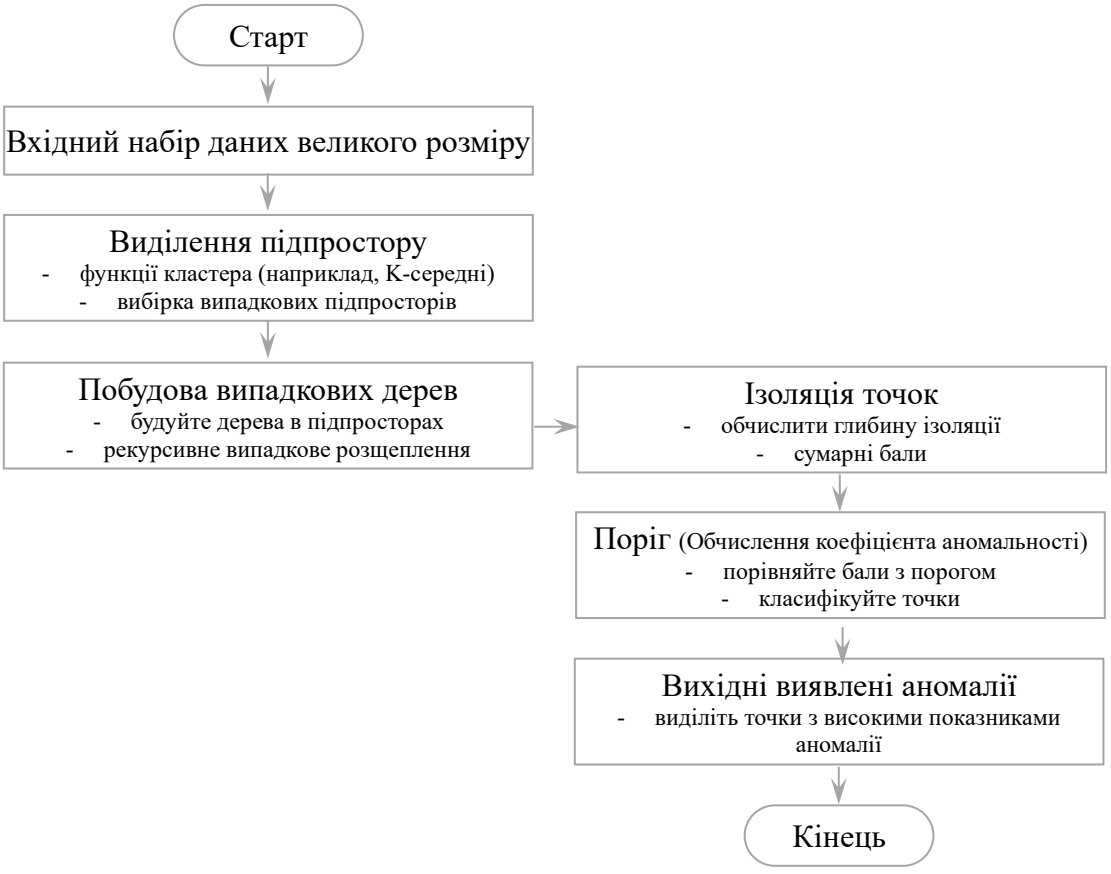

Рисунок В.3.3 – Процес роботи алгоритму Isolation Forest

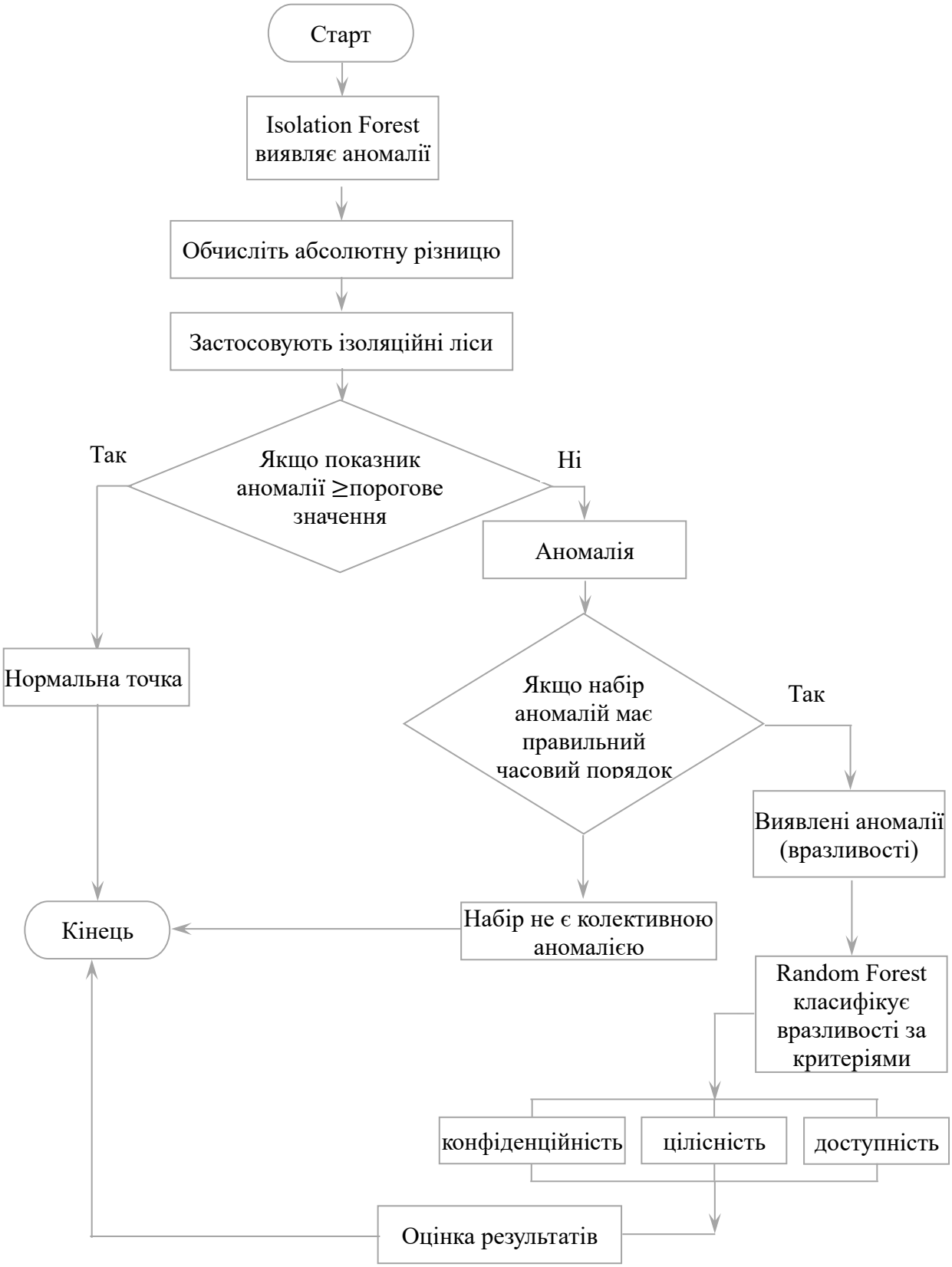

Рисунок В.3.4 – Алгоритм процесу виявлення і класифікації вразливостей системного реєстру методами машинного навчання

Рисунок В.3.5 – Механізм усунення вразливостей у системному реєстрі

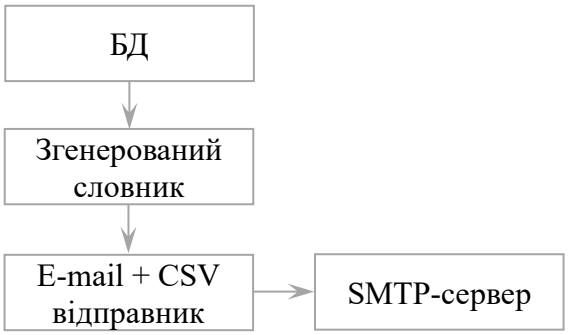

Рисунок В.3.6 – Графічне представлення підсистеми генерації звітів

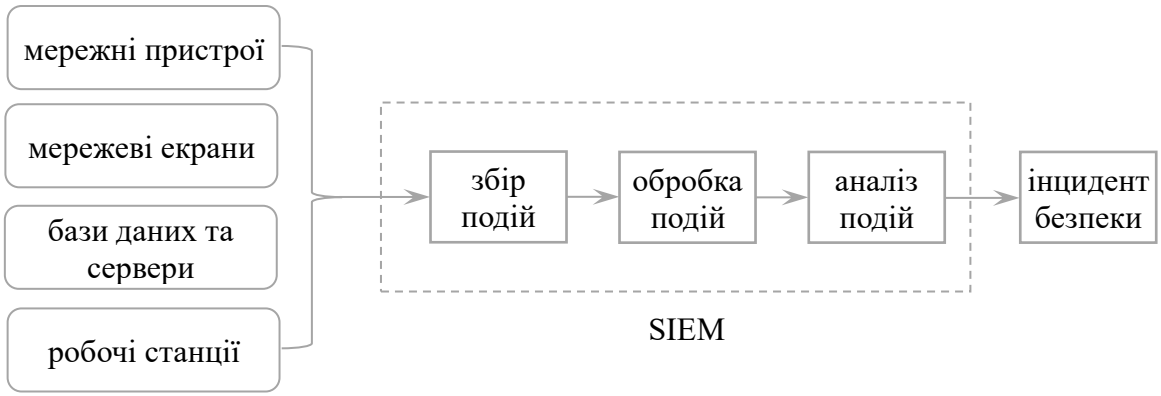

Рисунок В.3.7 – Схема роботи SIEM системи

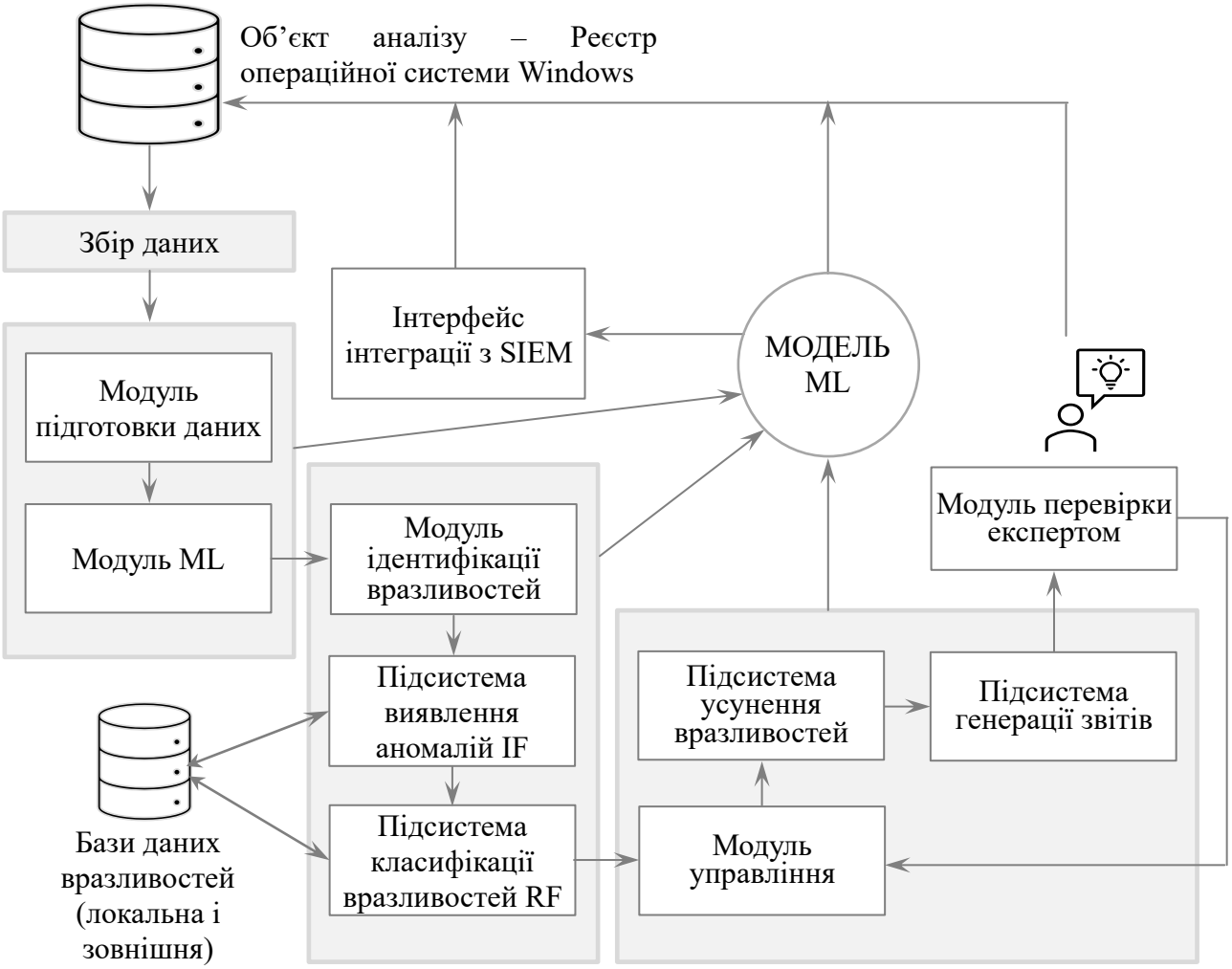

Рисунок В.3.8 – Графічне представлення взаємозв’язку між модулями механізму виявлення вразливостей у системному реєстрі методами ML

ДОДАТОК Г

ФРАГМЕНТИ ВИХІДНОГО КОДУ РОЗРОБЛЕНОГО МЕХАНІЗМУ ВИЯВЛЕННЯ ВРАЗЛИВОСТЕЙ СИСТЕМНОГО РЕЄСТРУ

```
import pandas as pd
from sklearn.ensemble import IsolationForest, RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, accuracy_score
import json
import numpy as np

# --- 1. Підготовка даних ---

# 1.1. "Ідеальний" образ реєстру
ideal_registry_df = pd.DataFrame({
   'Registry_Path': [

r'HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer',

r'HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run',
       r'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services'
   ],
   'Value_Name': ['DisableMSI', 'MaliciousApp', 'MyMaliciousService'],
   'Expected_Type': ['REG_DWORD', 'REG_SZ', 'REG_SZ'],
   'Expected_Value': [1, '', ''],
   'Expected_Existence': [1, 0, 0],
   'Security_Context': ['System Hardening', 'Autostart Management', 'Service
Hardening']
})

print("Ideal Registry Data:")
print(ideal_registry_df)
print("-" * 50)

# 1.2. Симуляція даних зі знімка реєстру (звіт Regshot після парсингу)
current_registry_snapshot_data = pd.DataFrame({
   'Registry_Path': [

r'HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer',

r'HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run',
```

```
        r'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows
NT\CurrentVersion',
        r'HKEY_CURRENT_USER\Control Panel\Desktop',

r'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\MyMaliciousS
ervice',

r'HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Ru
n'
    ],
    'Value_Name': ['DisableMSI', 'NewMaliciousApp', 'BuildLabEx', 'Wallpaper',
'ImagePath', 'AnotherApp'],
    'Current_Value': [0, 'C:\\malware.exe', '19045.3448',
'C:\\Windows\\Web\\Wallpaper\\Theme1\\img1.jpg',
'C:\\Windows\\system32\\malicious.exe', 'C:\\ProgramFiles\\some_app.exe'],
    'Change_Type': ['Modified', 'Added', 'Modified', 'Modified', 'Added', 'Added']
})

print("\nSimulated Current Registry Snapshot (after parsing Regshot):")
print(current_registry_snapshot_data)
print("-" * 50)

# --- 2. Виявлення відхилень від "ідеального" образу та Feature Engineering ---

def get_registry_path_features(path):
    if
r'HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\Installer' in
path:
        return 1
    elif
r'HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Run'
in path:
        return 2
    elif r'HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services' in path:
        return 3
    else:
        return 0

anomalies_data = []
for index, row in current_registry_snapshot_data.iterrows():
    path_feature = get_registry_path_features(row['Registry_Path'])

    change_type_numeric = 0
    if row['Change_Type'] == 'Modified':
        change_type_numeric = 1
```

```
    elif row['Change_Type'] == 'Added':
        change_type_numeric = 2
    elif row['Change_Type'] == 'Deleted':
        change_type_numeric = 3

    is_suspicious_value = 1 if 'malware' in str(row['Current_Value']).lower() or 'virus'
in str(row['Current_Value']).lower() or '.exe' in str(row['Current_Value']).lower() else
0

    is_deviated_from_ideal = 0
    ideal_row = ideal_registry_df[
        (ideal_registry_df['Registry_Path'] == row['Registry_Path']) &
        (ideal_registry_df['Value_Name'] == row['Value_Name'])
    ]
    if not ideal_row.empty:
        # Check if value deviates
        if ideal_row.iloc[0]['Expected_Value'] != row['Current_Value'] and
row['Change_Type'] == 'Modified':
            is_deviated_from_ideal = 1
        # Check if existence deviates (e.g., expected not to exist but was added)
        if ideal_row.iloc[0]['Expected_Existence'] == 0 and row['Change_Type'] ==
'Added':
            is_deviated_from_ideal = 1
    elif row['Change_Type'] == 'Added':
        is_deviated_from_ideal = 1


    anomalies_data.append([
        path_feature,
        change_type_numeric,
        is_suspicious_value,
        is_deviated_from_ideal
    ])

features_df = pd.DataFrame(anomalies_data, columns=['Path_Feature',
'Change_Type_Numeric', 'Is_Suspicious_Value', 'Is_Deviated_From_Ideal'])

print("\nFeatures for Anomaly Detection (after initial comparison logic):")
print(features_df)
print("-" * 50)

# --- 3. Виявлення аномалій за допомогою Isolation Forest ---

iso_forest = IsolationForest(contamination=0.1, random_state=42)
```

```
# Визначаємо список ознак, на яких навчається модель
feature_columns = ['Path_Feature', 'Change_Type_Numeric', 'Is_Suspicious_Value',
'Is_Deviated_From_Ideal']

# Навчаємо Isolation Forest тільки на колонках з ознаками
iso_forest.fit(features_df[feature_columns])

features_df['anomaly_score'] =
iso_forest.decision_function(features_df[feature_columns])
features_df['is_anomaly'] = iso_forest.predict(features_df[feature_columns])

print("\nIsolation Forest Results:")
print(features_df)
print("-" * 50)

# --- 4. Підготовка даних для навчання Random Forest Classifier ---

training_data_for_rf = pd.DataFrame({
   'Path_Feature': [1, 2, 0, 3, 0, 1, 2, 3, 0, 0],
   'Change_Type_Numeric': [1, 2, 1, 2, 2, 1, 2, 2, 1, 2],
   'Is_Suspicious_Value': [0, 1, 0, 0, 0, 0, 1, 0, 0, 0],
   'Is_Deviated_From_Ideal': [1, 1, 0, 1, 1, 1, 1, 1, 0, 1],
   'CVE_Label': [
      'CVE-2021-9012', # DisableMSI=0
      'CVE-2023-1234', # MaliciousApp added (Run key)
      'Unknown',       # Normal change
      'CVE-2022-5678', # MaliciousService added
      'Unknown',       # Another app added, but maybe not vulnerability
      'CVE-2021-9012', # More DisableMSI=0 examples
      'CVE-2023-1234', # Another malicious app in Run key
      'CVE-2022-5678', # Another service added
      'Unknown',       # More normal changes
      'Unknown'
   ]
})

# Розділяємо дані на тренувальний та тестовий набори для Random Forest
X_train_rf = training_data_for_rf[['Path_Feature', 'Change_Type_Numeric',
'Is_Suspicious_Value', 'Is_Deviated_From_Ideal']]
y_train_rf = training_data_for_rf['CVE_Label']

# Навчаємо Random Forest Classifier
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42)
rf_classifier.fit(X_train_rf, y_train_rf)
```

```
print("\nRandom Forest Classifier (trained for CVE Classification):")
print(f"RF Classifier trained with {len(X_train_rf)} examples.")
print("-" * 50)


# --- 5. Класифікація виявлених аномалій за допомогою Random Forest та CVE ---

# Об'єднуємо початкові дані зі знімка реєстру з результатами Isolation Forest
anomalies_with_raw_data = current_registry_snapshot_data.copy()
anomalies_with_raw_data['anomaly_score'] = features_df['anomaly_score']
anomalies_with_raw_data['is_anomaly'] = features_df['is_anomaly']
anomalies_with_raw_data['Path_Feature'] = features_df['Path_Feature']
anomalies_with_raw_data['Change_Type_Numeric'] =
features_df['Change_Type_Numeric']
anomalies_with_raw_data['Is_Suspicious_Value'] =
features_df['Is_Suspicious_Value']
anomalies_with_raw_data['Is_Deviated_From_Ideal'] =
features_df['Is_Deviated_From_Ideal']


identified_vulnerabilities = []

# CVE дані для вилучення C-I-A (розширені для прикладу)
cve_details_extended = {
    'CVE-2023-1234': {'description': 'Vulnerability allows arbitrary code execution via
registry key manipulation in
HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run.', 'Confidentiality':
'High', 'Integrity': 'High', 'Availability': 'High'},
    'CVE-2022-5678': {'description': 'Privilege escalation vulnerability through
insecure service path.', 'Confidentiality': 'Low', 'Integrity': 'High', 'Availability':
'High'},
    'CVE-2021-9012': {'description': 'Security bypass due to incorrect registry
permissions related to Windows Installer.', 'Confidentiality': 'Medium', 'Integrity':
'Medium', 'Availability': 'Low'},
    'Unknown': {'description': 'Anomaly detected, but no specific CVE match found.',
'Confidentiality': 'N/A', 'Integrity': 'N/A', 'Availability': 'N/A'}
}


for index, row in anomalies_with_raw_data.iterrows():
    if row['is_anomaly'] == -1: # Якщо Isolation Forest вважає це аномалією
        # Готуємо ознаки для Random Forest класифікатора
        features_for_rf = pd.DataFrame([[
            row['Path_Feature'],
```

```
            row['Change_Type_Numeric'],
            row['Is_Suspicious_Value'],
            row['Is_Deviated_From_Ideal']
        ]], columns=X_train_rf.columns)

        # Прогнозуємо CVE ID за допомогою Random Forest
        predicted_cve = rf_classifier.predict(features_for_rf)[0]

        # Отримуємо деталі CVE
        cve_info = cve_details_extended.get(predicted_cve,
cve_details_extended['Unknown'])

        potential_vulnerability = {
            'Registry_Path': row['Registry_Path'],
            'Value_Name': row['Value_Name'],
            'Current_Value': row['Current_Value'],
            'Change_Type': row['Change_Type'],
            'Anomaly_Score': row['anomaly_score'],
            'CVE_ID': predicted_cve,
            'Description': cve_info['description'],
            'Confidentiality_Impact': cve_info['Confidentiality'],
            'Integrity_Impact': cve_info['Integrity'],
            'Availability_Impact': cve_info['Availability']
        }
        identified_vulnerabilities.append(potential_vulnerability)

# --- 6. Формування звіту ---

print("\n--- Identified Vulnerabilities Report ---")
if identified_vulnerabilities:
    report_df = pd.DataFrame(identified_vulnerabilities)
    print(report_df.to_string())
else:
    print("No significant vulnerabilities identified based on current criteria.")

# Розділення за критеріями безпеки C-I-A
print("\n--- Vulnerabilities by CIA Impact ---")
cia_categorized = {
    'Confidentiality': [],
    'Integrity': [],
    'Availability': []
}

for vuln in identified_vulnerabilities:
    if vuln['Confidentiality_Impact'] == 'High':
```

```
        cia_categorized['Confidentiality'].append(vuln['CVE_ID'] if vuln['CVE_ID'] !=
'N/A' else vuln['Registry_Path'])
    if vuln['Integrity_Impact'] == 'High':
        cia_categorized['Integrity'].append(vuln['CVE_ID'] if vuln['CVE_ID'] != 'N/A'
else vuln['Registry_Path'])
    if vuln['Availability_Impact'] == 'High':
        cia_categorized['Availability'].append(vuln['CVE_ID'] if vuln['CVE_ID'] !=
'N/A' else vuln['Registry_Path'])

for impact, cve_list in cia_categorized.items():
    print(f"{impact} Impact (High): {', '.join(cve_list) if cve_list else 'None'}")
```