https://doi.org/10.31891/2307-5732-2025-349-20
УДК 004.9

ДЕМКІВСЬКА ТЕТЯНА
Київський національний університет технологій та дизайну
https://orcid.org/0000-0002-2176-163X
E-mail: demkivskiy@gmail.com
ЧУПРИНКА НАТАЛІЯ
Київський національний університет технологій та дизайну
https://orcid.org/0000-0002-8952-7567
E-mail: chuprinka.nv@knutd.com.ua
ЯХНО ВОЛОДИМИР
Київський національний університет технологій та дизайну
https://orcid.org/0000-0001-6129-4178
E-mail yaxno.vm@knutd.com.ua
ЖЕНЖЕРА МАКСИМ
Київський національний університет технологій та дизайну

РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ ДЛЯ СТВОРЕННЯ СИГНАТУР З МЕТОЮ ІДЕНТИФІКАЦІЇ ЗЛОВМИСНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У роботі вирішено завдання розробки автоматизованої системи генерації сигнатур для виявлення зловмисного програмного забезпечення.

Ключові слова: зловмисне програмне забезпечення, кіберзагрози, сигнатурний метод виявлення, сигнатури, автоматизована система

DEMKIVSKA TETIANA, CHUPRYNKA NANALIYA,
YAKHNO VOLODYMYR, ZHENZHERA MAKSYM
Kyiv National University of Technologies and Design

DEVELOPMENT OF AN AUTOMATED SYSTEM FOR CREATING SIGNATURES TO IDENTIFY MALWARE

The work solves the problem of developing an automated signature generation system for detecting malicious software (MSW). The system simplifies the process of analyzing, creating and managing signatures, allowing users to quickly receive results in adaptive formats.

The general characteristics of malicious software are studied, the main approaches to its analysis are presented, in particular static and dynamic methods, and various types of methods for detecting MSW are described. Particular attention is paid to the signature method as one of the most effective in modern practice.

The requirements for an automated signature generation system are determined. The system implementation process is considered, which includes the selection of appropriate technologies and the implementation of components in accordance with the established requirements. A comparison of methods and tools available on the market is carried out, and the uniqueness of the proposed approach, which combines efficiency, integration via API and ease of use, is substantiated.

MongoDB and Python technologies were selected for development, which provide flexibility and productivity. This approach contributes to effective integration with other software products to increase the level of cybersecurity.

Integration with the MongoDB database for data storage and processing has been implemented, and a web interface has been created for convenient visual representation of signatures. An automated system has been developed that generates signatures, stores them in the database, provides access via API, and displays the results in the web interface. Examples of generated signatures are provided to confirm the functionality of the system.

As part of the system implementation, key modules have been developed, such as signature generation, API for data access, and integration of the web interface for convenient presentation of results.

Keywords: malware, cyber threats, signature-based detection method, signatures, automated system

Постановка проблеми

В роботі вирішується питання розробки автоматизованої системи генерації сигнатур для виявлення шкідливого програмного забезпечення. Система спрощує процес аналізу, створення та управління сигнатурами, дозволяючи користувачам швидко отримувати результати у адаптивних форматах.

Проблема ефективного виявлення зловмисного програмного забезпечення (ЗПЗ) залишається однією з ключових у сфері кібербезпеки. Попри існування численних інструментів, аналоги автоматизованої системи генерації сигнатур із інтеграцією через веб-інтерфейси та API відсутні. Це створює потребу у розробці нових підходів, які забезпечують високу точність, адаптивність і зручність використання.

Більшість існуючих систем для аналізу та управління кіберзагрозами, такі як MISP і OpenCTI, мають вузькоспеціалізовані завдання, орієнтовані на колекціонування, структурування, та інтерактивне відображення інформації про загрози. Основний акцент у цих платформах робиться на:

- роботу з інцидентами: відстеження, реєстрація, аналіз і зв'язок між інцидентами;
- графічний інтерфейс: спрощення аналізу загроз за допомогою інтерактивних візуалізацій і дашбордів;

- інтеграцію з різними джерелами: збирання та агрегування даних із зовнішніх та внутрішніх джерел.

Проте вони не спрямовані на глибокий аналіз окремих файлів і не враховують особливості файлів, які представлені у звітах динамічного аналізу (sandbox). Зокрема, такі системи не фокусуються на детальному статичному аналізі файлів (метадані, секції, рядки, хеші), не обробляють інформацію з кожного окремого файлу в sandbox-звітах, не надають інструментів для генерації сигнатур, які можуть бути використані для автоматичного виявлення зловмисного ПЗ в реальних системах.

На сьогоднішній день відсутні open-source системи, які б автоматично генерували сигнатури для виявлення зловмисного ПЗ на основі статичного аналізу файлів. Існуючі рішення, такі як MISP і OpenCTI, є найближчими аналогами, але їхня функціональність сфокусована на інших аспектах.

Ці системи не підтримують автоматизовану роботу зі статичним аналізом окремих файлів і не створюють сигнатури для реального використання в детектуванні.

Проєкт, спрямований на створення таких сигнатур із врахуванням метаданих, рядків, хешів та інших характеристик, заповнює цю прогалину, пропонуючи унікальну функціональність, недоступну у відкритих рішеннях.

Аналіз останніх досліджень

Аналіз досліджень підтверджує, що боротьба зі шкідливим програмним забезпеченням залишається критично важливою проблемою. Це зумовлено постійним зростанням кількості та складності кіберзагроз, значними потенційними збитками від шкідливого ПЗ та вразливістю сучасних інформаційних систем. Робота [1] є комплексним посібником, який охоплює всі аспекти кібербезпеки, від аналізу шкідливого програмного забезпечення до оволодіння необхідними інструментами та навичками. Вона надає читачам можливість стати професіоналами в цій галузі, забезпечуючи їх необхідними знаннями та практичними порадами. В роботі [2] представлено детальний огляд підходів до виявлення зловмисного програмного забезпечення та останніх методів виявлення, які використовують ці підходи. Мета статті полягає в тому, щоб допомогти дослідникам мати загальне уявлення про підходи до виявлення зловмисного програмного забезпечення, плюси та мінуси кожного підходу до виявлення та методи, які використовуються в цих підходах. В [3] пропонуються підходи до виявлення фішингу для уникнення для зменшення ризиків від фішингових атак, серед яких алгоритми глибинного навчання. В [4] розглянуто проблеми боротьби із загрозами від зловмисного програмного забезпечення, яке шифрує конфіденційну інформацію і не розкриває файли, поки не отримає грошове винагородження. Для боротьби із зазначеними загрозами запропоновано використання моделі, на основі машинного навчання. Здійснено порівняння запропонованої моделі з уже існуючими та відмічено переваги розробленої моделі.

Формулювання цілей статті

Метою дослідження є доведення ефективності автоматизованої системи генерації сигнатур для виявлення та аналізу зловмисного програмного забезпечення, яка відповідає сучасним вимогам кібербезпеки.

Виклад основного матеріалу

Програмне забезпечення, що розроблене з метою завдати шкоди користувачеві комп’ютера або комп’ютерній системі носить назву зловмисне програмне забезпечення (або malware, що є скороченням від malicious software).

Зловмисне програмне забезпечення потрапляє на комп'ютер різними шляхами: через заражені файли, завантажені з інтернету; через електронну пошту або через підключення заражених носіїв інформації.

Розглянемо деякі типи зловмисного програмного забезпечення: це в першу чергу віруси, що поширюються, копіюючи себе та впроваджуючи свій код в інші програми або файли; черв'яки, які можуть самостійно поширюватися по мережі, використовуючи вразливості в системі безпеки; троянські програми, які маскуються під корисні програми, але при виконанні завдають шкоди, наприклад, крадуть дані або відкривають доступ зловмисникам; програми-вимагачі, що шифрують файли на комп'ютері та вимагають викуп за їх відновлення; шпигунські коди, які таємно збирають інформацію про користувача, наприклад, паролі, дані кредитних карт або історію відвідувань; рекламне програмне забезпечення, що відображає небажану рекламу на комп'ютері; боти, які використовуються для створення ботнетів - мереж заражених комп'ютерів, які можуть використовуватися для здійснення DDoS-атак, розсилки спаму або інших зловживань.

Розглянемо методи виявлення і аналізу зловмисного програмного забезпечення. В чому між ними різниця? Виявлення - це процес ідентифікації потенційно зловмисного програмного забезпечення. Його мета - швидко та ефективно визначити, чи є програма шкідливою. Аналіз - це більш глибоке дослідження шкідливого програмного забезпечення для розуміння його функцій, поведінки та потенційного впливу. Його мета - отримати детальну інформацію про загрозу.

Методами автоматизованого аналізу зловмисного програмного забезпечення є: динамічний, статичний та гібридний метод. Домінують динамічний та статичний аналіз. Статичний аналіз полягає в дослідженні коду без його запуску. Перевагою статичного методу є безпечність. Це дозволяє глибоко дослідити структуру програми, Недоліком цього підходу є те, що не завжди виявляється шкідлива поведінка, що залежить від виконання коду. Динамічний аналіз дозволяє побачити, як зловмисне ПЗ діє

в реальних умовах. Запускаючи програму в контрольованому середовищі, ми можемо спостерігати за її поведінкою та чітко зрозуміти її зловмисний вплив.

Методами виявлення зловмисного програмного забезпечення є сигнатурний аналіз, який дозволяє порівнювати код програми з відомими сигнатурами шкідливого ПЗ; евристичний аналіз – аналіз поведінки програми ті її коду на наявність підозрілих дій та аналіз на основі методів машинного навчання, що передбачає використання алгоритмів машинного навчання для виявлення зловмисного програмного забезпечення на основі аналізу великих обсягів даних.

Метод виявлення зловмисного програмного забезпечення на основі сигнатур є одним з найпростіших методів виявлення зловмисного програмного забезпечення. Він базується на порівнянні коду програми з відомими сигнатурами шкідливого ПЗ.

Сигнатура - це унікальний фрагмент коду або послідовність байтів, яка ідентифікує конкретний зразок зловмисного програмного забезпечення. Сигнатури зберігаються в базі даних, яка постійно оновлюється.

Антивірусне програмне забезпечення сканує файли на комп'ютері та порівнює їх код з сигнатурами в базі даних. Якщо знайдено збіг, програма вважається шкідливою, і антивірусне ПЗ може її заблокувати, видалити або помістити в карантин.

Сигнатурний аналіз є відносно простим та швидким методом, що дозволяє швидко сканувати велику кількість файлів. Метод ефективний проти відомих шкідливих програм, сигнатури яких вже є в базі даних. Але неефективність проти нових загроз - не здатний виявляти нові або модифіковані шкідливі програми, сигнатури яких ще не додано до бази даних. База даних сигнатур може містити лише обмежену кількість записів, що може призвести до того, що деякі види шкідливого ПЗ залишаться непоміченими.

Розроблений проєкт складається з таких основних компонентів: збагачення бази даних, генерація сигнатур на основі підготовленої бази даних, експорт якісних згенерованих сигнатур у потрібному форматі. Загальна схема роботи алгоритму представлена на Рис. 1.

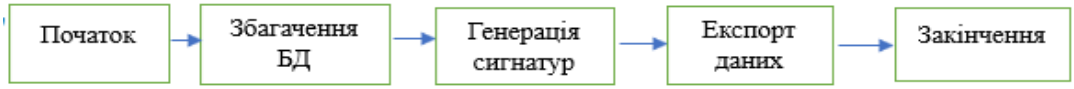

Рис.1 Загальна схема роботи алгоритму

Важливим принципом у розробленні цього проєкту є забезпечення універсальності та гнучкості. Система має дозволяти легке налаштування та заміну джерел, форматів і додаткових правил. Це гарантує адаптивність до нових вимог і зручність інтеграції з різними типами даних та інструментів.

Розглянемо кожен компонент більш детально.

Збагачення БД. Основною моделлю даного проєкту є шкідливі виконувані файли, а також інші пов'язані артефакти, такі як мережева активність, ключі реєстру, конфігураційні файли тощо. Ключовою метою є створення універсального інструменту для зібрання, оброблення та аналізу даних, що дозволяє виявляти характерні ознаки та формувати якісні сигнатури для ідентифікації загроз.

Головним джерелом даних є сервіси динамічного аналізу програмного забезпечення, які працюють із використанням віртуальних машин (sandbox). Сервіси динамічного аналізу програмного забезпечення, які працюють із використанням віртуальних машин (sandbox), є важливими інструментами для дослідження та виявлення шкідливого програмного забезпечення. Вони дозволяють безпечно запускати підозрілі файли в ізольованому середовищі та спостерігати за їх поведінкою, не завдаючи шкоди реальній системі.. Однак архітектура проєкту передбачає можливість розширення та інтеграції з іншими джерелами, такими як пропрієтарні дані, зокрема журнали роботи систем (логи) та звіти в альтернативних форматах, наприклад, JSON, XML або специфічні формати, що використовуються внутрішніми системами організацій. Це забезпечує гнучкість системи та її здатність адаптуватися до різних умов і джерел даних.

Резюмучи, етап збагачення даних включає такі ключові кроки.

Крок 1. Data collection (збір даних). Дані збираються з різних джерел, таких як sandbox, журнали мережевої активності або інші зовнішні системи.

Крок 2. Data normalization (нормалізація даних). Зібрані дані перетворюються у стандартизований формат, що дозволяє уніфікувати їх подальшу обробку. Наприклад, поля, які описують IP-адреси, часові мітки, хеші файлів, нормалізуються відповідно до єдиного стандарту.

Крок 3. Data storage (збереження даних). Нормалізовані дані зберігаються в оптимізованій базі даних, яка забезпечує швидкий доступ і можливість масштабування для обробки великих обсягів інформації.

Генерація сигнатур. Генерація сигнатур є критично важливим етапом у процесі виявлення шкідливого програмного забезпечення. Її метою є створення точних та ефективних сигнатур, які дозволять ідентифікувати та блокувати шкідливе ПЗ.

Під час цього етапу, структуровані дані, отримані на попередньому етапі збагачення, перетворюються на готові до використання сигнатури. Ці сигнатури потім інтегруються в системи

виявлення загроз, де вони використовуються для сканування файлів та дій на предмет наявності шкідливого коду.

Особлива увага приділяється якості сигнатур. Важливо забезпечити, щоб вони були достатньо точними, щоб мінімізувати кількість помилкових спрацювань (false positives), коли доброякісні файли або дії помилково ідентифікуються як шкідливі. Такі помилкові спрацювання можуть призвести до блокування важливих ресурсів та порушення роботи системи.

Водночас, сигнатури повинні бути достатньо чутливими, щоб виявляти навіть найновіші та найскладніші загрози. Помилкові негативні результати (false negatives), коли шкідливе ПЗ залишається непоміченим, можуть мати серйозні наслідки для безпеки системи.

Процес генерації складається з таких основних кроків.

Крок 1. Аналіз сигнатур

На цьому етапі аналізуються дані з бази, щоб визначити основні характеристики, які можна використовувати для створення сигнатур. Ідентифікація унікальних ознак: пошук властивостей, які чітко відрізняють шкідливий файл або поведінку від доброякісних. Це можуть бути статичні артефакти (наприклад, специфічні хеш-значення, унікальні строки, шаблони коду) або поведінкові ознаки (наприклад, спроби підключення до відомих шкідливих доменів, створення файлів у системних каталогах). Кореляція між даними: виявлення закономірностей і взаємозв'язків між різними артефактами, наприклад, мережевою активністю та змінами у реєстрі. Класифікація: визначення категорій загроз, наприклад, трояни, шифрувальники, експлойти, що допомагає оптимізувати наступний етап генерації.

Крок 2. Створення сигнатур

Цей етап відповідає за безпосереднє формування сигнатур на основі проведеного аналізу.

Автоматична генерація: система автоматично створює сигнатури на основі виділених унікальних ознак. Наприклад: метадані виконуваних файлів; YARA-правила для виявлення файлів за їх вмістом (рядками, паттернами); Snort/Suricata сигнатури: для аналізу мережевого трафіку; індикатори компрометації (IOCs): хеші файлів, IP-адреси, домени.

Підтримка гнучкості: можливість налаштовувати генерацію сигнатур залежно від потреб (детектування певного виду загроз або всіх підозрілих об'єктів).

Оптимізація: виключення надлишкових або дубльованих сигнатур, щоб зменшити навантаження на системи виявлення загроз.

Крок 3. Перевірка сигнатур

Перевірка є важливим кроком, що гарантує якість і надійність згенерованих сигнатур. Цей крок включає:

- тестування на вибірках: перевірка сигнатур на великій базі файлів, яка включає як доброякісні, так і шкідливі зразки. Сюди входить аналіз помилкових спрацювань: наприклад, якщо сигнатура виявляє доброякісні файли, вона потребує уточнення; аналіз пропущених загроз: якщо сигнатура не детектує відомі загрози, її потрібно доопрацювати;
- динамічне тестування: сигнатури перевіряються у реальному середовищі (наприклад, у sandbox або системах моніторингу);
- ручна ревізія: важливі сигнатури проходять додатковий аналіз аналітиками, щоб підтвердити їхню якість;
- документування: збереження результатів перевірки, що дозволяє швидко вносити корективи у разіц необхідності

Експорт даних

Етап експорту забезпечує перетворення згенерованих сигнатур у необхідний формат для їх подальшого використання у системах виявлення загроз. Основна мета цього етапу — надати користувачу можливість швидко й ефективно отримати лише релевантні сигнатури у потрібному форматі, забезпечуючи високу якість результату.

Процес експорту складається з наступних кроків.

Крок 1. Конвертація формату

Сигнатури, створені в системі, повинні бути адаптовані до вимог цільового середовища.

Підтримка різних форматів:

- YARA-правила для аналізу вмісту файлів;
- Snort/Suricata сигнатури для систем виявлення мережевих загроз;
- CSV/JSON/XML для інтеграції у зовнішні аналітичні інструменти.

Уніфікація структури: перетворення даних у стандартизовану форму, яка відповідає формату призначення. Наприклад, додавання заголовків, метаінформації чи опису сигнатури.

Конфігурація на вимогу: підтримка специфічних налаштувань формату залежно від вимог користувача чи цільової системи.

Крок 2. Перевірка якості

Перед експортом сигнатур важливо провести їх перевірку на відповідність заданим критеріям.

Перевірка коректності формату: сигнатури тестуються на синтаксичну правильність у цільових системах (наприклад, верифікація YARA-правил за допомогою YARA CLI).

Перевірка актуальності: відбір тільки перевірених і затверджених сигнатур, які успішно пройшли стадії аналізу та тестування.

Верифікація метаінформації: перевірка описів, категорій та інших полів, які можуть впливати на використання сигнатур у продуктивному середовищі.

Крок 3. Експорт сигнатур

На завершальному етапі система надає користувачу зручні засоби для отримання потрібних даних.

Фільтрація: можливість експортувати лише релевантні сигнатури за критеріями:

- тип загрози (троян, шифрувальник, шпигунське ПЗ тощо);
- джерело даних або метод створення (поведінковий, статичний аналіз);
- актуальність (дата створення або останньої перевірки).

Масштабованість: підтримка експорту великих обсягів сигнатур або окремих груп залежно від запиту.

Різні способи експорту:

- файловий експорт: формування файлів для завантаження;
- API: можливість інтеграції з іншими системами для передачі сигнатур у реальному часі;
- хмарне сховище: збереження сигнатур у централізованому репозиторії для віддаленого доступу.

У практичній реалізації системи генерації сигнатур розглянуто розроблення MVP (Minimum Viable Product) проєкту системи генерації сигнатур. MVP - це мінімально життєздатний продукт, який володіє базовою функціональністю для демонстрації працездатності системи та отримання зворотного зв'язку від користувачів.

Процес реалізації системи включає вибір відповідних технологій і імплементацію компонентів відповідно до сформованих вимог. Для збереження даних була обрана база даних MongoDB. Це рішення обумовлено тим, що MongoDB забезпечує гнучкість та ефективність при роботі як зі структурованою, так і з напівструктурованою інформацією. Саме така різноманітність даних є типовою для аналізу шкідливого програмного забезпечення, де часто зустрічаються дані різних форматів та рівнів організації.

Використання MongoDB надає можливість збереження даних різних типів, гнучко змінювати структуру даних, швидко отримувати доступ до даних, масштабувати систему. MongoDB добре працює з великими обсягами неструктурованої інформації, що важливо для проєкту, де зберігаються різноманітні дані про файли. В цілому, вибір MongoDB для збереження даних при аналізі шкідливого програмного забезпечення є обґрунтованим та ефективним рішенням, яке забезпечує необхідну гнучкість, продуктивність та масштабованість системи. Використання цієї бази даних дозволило ефективно зберігати, обробляти та масштабувати інформацію, необхідну для генерації сигнатур.

Як основну мову програмування використано Python, що дозволило ефективно реалізувати алгоритми аналізу, обробки даних та інтеграції компонентів системи. Цей вибір є ідеальним для реалізації проєкту генерації сигнатур. Її популярність, багата екосистема бібліотек і простота у використанні забезпечують ефективну реалізацію ключових завдань.

На Рис.2 представлено приклад реальної генерованої сигнатури в JSON форматі.

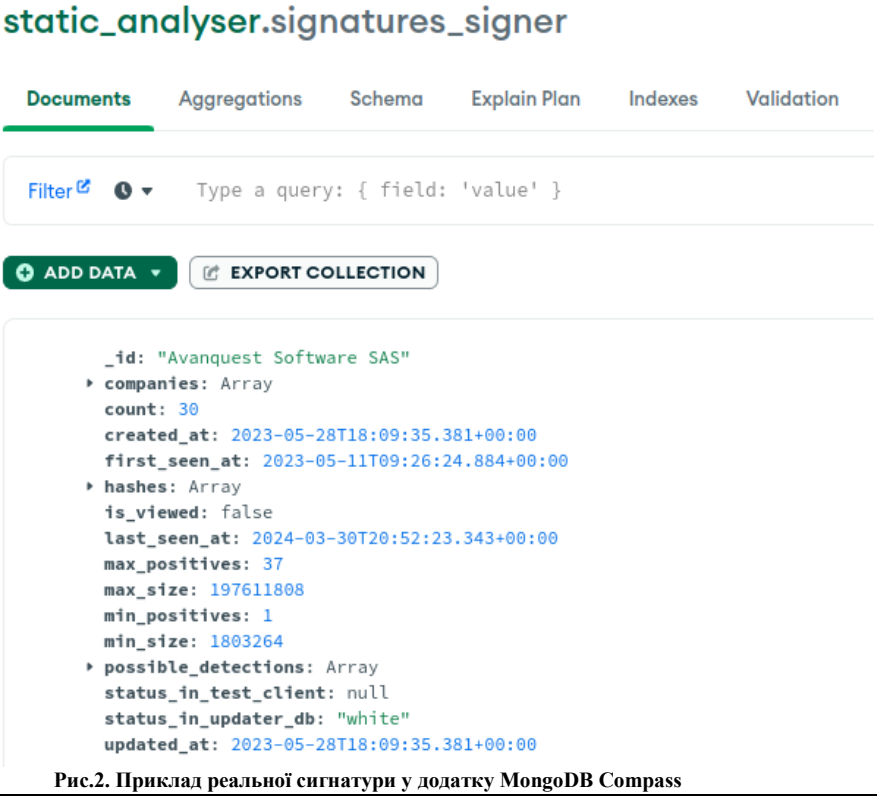

Рис.2. Приклад реальної сигнатури у додатку MongoDB Compass

Скриншот зроблено у додатку MongoDB Compass, що дозволяє зручно маніпулювати даними в MongoDB через GUI (Gaphical User Interface).

У рамках реалізації системи виконано розробку ключових модулів, таких як генерація сигнатур, API (Application Programming Interface) для доступу до даних, а також інтеграція веб-інтерфейсу для зручного представлення результатів. Імплементація базується на, сформованих вимогах, що забезпечило відповідність системи поставленим завданням і високий рівень функціональності.

Висновки

Досліджено загальну характеристику шкідливого програмного забезпечення, наведено основні підходи до його аналізу, зокрема статичний і динамічний методи, та описано різні типи методів виявлення ЗПЗ. Особливу увагу приділено сигнатурному методу як одному з найефективніших у сучасній практиці.

Визначенню вимоги до автоматизованої системи генерації сигнатур, а також проведено аналіз існуючих рішень у цій галузі. Проведено порівняння методів і інструментів, доступних на ринку, та обґрунтовано унікальність запропонованого підходу, який поєднує ефективність, інтеграцію через API та зручність використання.

Здійснена практична реалізація системи. Для розробки обрано технології MongoDB і Python, які забезпечують гнучкість та продуктивність. Реалізовано автоматизовану систему, що генерує сигнатури, зберігає їх у базі даних, надає доступ через API та демонструє результати у веб-інтерфейсі. Наведено приклад згенерованої сигнатури, який підтверджують функціональність системи.

Запропонована автоматизована система генерації сигнатур є ефективним інструментом для виявлення шкідливого програмного забезпечення. Вона дозволяє автоматизувати рутинні процеси аналізу та виявлення, покращує швидкість і точність виявлення ШПЗ, а також забезпечує можливість інтеграції з іншими системами кібербезпеки.

У сучасному світі кіберзагроз, що постійно змінюється, система повинна бути адаптивною. Для цього в перспективі пропонується використовувати алгоритми машинного навчання, які допоможуть виявляти аномалії та аналізувати нові види зловмисного програмного забезпечення. Це забезпечить постійне оновлення системи та її здатність протистояти новим загрозам.

Література

1. Saxe J. Malware Data Science: Attack Detection and Attribution / J. Saxe, H. Sanders – 2018. – P.272 https://unidel.edu.ng/focelibrary/books/Malware%20Data%20Science%20Attack%20Detection%20and%20Attribution%20by%20Joshua%20Saxe%20Hillary%20Sanders%20(z-lib.org).pdf

2. Aslan O. Comprehensive Review on Malware Detection Approaches / O. Aslan, R. Samet // IEEE Access. – 2020. – Volume 8. – P. 6249-6271. doi: 10.1109/ACCESS.2019.2963724

3. Catal C. Applications of deep learning for phishing detection: a systematic literature review / C. Catal, G. Giray, B. Tekinerdogan, S. Kumar, S.Shukla // Knowledge and Information Systems. – 2022. – Volume 64. – P. 1457–1500. https://doi.org/10.1007/s10115-022-01672-x

4. Ogundokun R. Application of machine learning for ransomware detection in IoT devices / R. Ogundokum, A.Bamidele, S.Misra, O. Abikoye // Artificial intelligence for cyber security: methods, issues and possible horizons or opportunities. Cham: Springer International Publishing. – 2021. – P. 393-420. https://doi.org/10.1007/978-3-030-72236-4_16

References

1. Saxe J. Malware Data Science: Attack Detection and Attribution / J. Saxe, H. Sanders – 2018. – P.272 https://unidel.edu.ng/focelibrary/books/Malware%20Data%20Science%20Attack%20Detection%20and%20Attribution%20by%20Joshua%20Saxe%20Hillary%20Sanders%20(z-lib.org).pdf

2. Aslan O. Comprehensive Review on Malware Detection Approaches / O. Aslan, R. Samet // IEEE Access. – 2020. – Volume 8. – P. 6249-6271. doi: 10.1109/ACCESS.2019.2963724

3. Catal C. Applications of deep learning for phishing detection: a systematic literature review / C. Catal, G. Giray, B. Tekinerdogan, S. Kumar, S.Shukla // Knowledge and Information Systems. – 2022. – Volume 64. – P. 1457–1500. https://doi.org/10.1007/s10115-022-01672-x

4. Ogundokun R. Application of machine learning for ransomware detection in IoT devices / R. Ogundokum, A.Bamidele, S.Misra, O. Abikoye // Artificial intelligence for cyber security: methods, issues and possible horizons or opportunities. Cham: Springer International Publishing. – 2021. – P. 393-420. https://doi.org/10.1007/978-3-030-72236-4_16