Комп'ютерна інженерія

УДК 004.056.53:004.8 https://doi.org/10.32515/2664-262X.2026.13(44).41-51

А. О. Головатюк, **Б. О. Савенко**, д-р філос.
Хмельницький Національний Університет, Хмельницький, Україна
e-mail: golovatiukao@gmail.com, savenko_bohdan@ukr.net

Гібридна система виявлення поліморфного програмного забезпечення ботнетів на основі аналізу інваріантних ознак

У статті розглядається актуальна проблема кібербезпеки – виявлення сучасних поліморфних ботнетів, які стають дедалі поширенішими та важче піддаються детекції у динамічному середовищі загроз. Завдяки використанню складних технік обфускації та динамічної модифікації коду, класичні антивірусні підходи, що базуються виключно на сигнатурах, втрачають свою ефективність. Водночас ізольовані системи динамічного аналізу часто виявляються ресурсоємними та вразливими до механізмів ухилення. У роботі проаналізовано недоліки існуючих методів захисту та запропоновано нову гібридну концепцію виявлення. В основі розробленої системи лежить ізоляція інваріантних ознак – структурних та поведінкових характеристик шкідливого коду, що залишаються сталими (недоторканими) попри процеси мутації та обфускації. Архітектура рішення містить статичний модуль (базується на аналізі графів потоку керування та n-грам) і динамічний модуль моніторингу системних викликів, результати яких агрегуються за допомогою методів машинного навчання (ансамблевої класифікації) для прийняття остаточного рішення.

кібербезпека, бот-мережі, поліморфне зловмисне програмне забезпечення, гібридний аналіз, машинне навчання, інваріантні ознаки, графи потоку керування, виявлення аномалій

Постановка проблеми. Сучасні бот-мережі масово використовують поліморфні техніки генерації коду, постійно змінюючи свою бінарну структуру та хеш-суми при кожному новому зараженні зі збереженням деструктивного функціоналу [1, 3, 4]. Через це стандартні сигнатурні методи захисту стають принципово неефективними. З іншого боку, використання виключно методів динамічного аналізу є надто ресурсоємним завданням, яке, до того ж, є вразливим до складних технік ухилення від аналізу (Anti-VM, Anti-Debugging) [6, 9].Зважаючи на фундаментальні недоліки суто сигнатурного та евристичного підходів, метою даної роботи є розробка концептуальної моделі гібридної системи виявлення поліморфного ПЗ ботнетів. Запропонована система поєднує швидкість статичного аналізу з точністю динамічного моніторингу і базується на пошуку та ізоляції інваріантних (незмінних) структурних і поведінкових характеристик шкідливого коду за допомогою ансамблевих методів машинного навчання.

Аналіз останніх досліджень і публікацій. Існуючі підходи до детекції зловмисного ПЗ традиційно поділяються на статичні, динамічні та гібридні. Статичний аналіз, попри активне впровадження методів машинного та глибокого навчання, залишається вкрай вразливим до методів обфускації та поліморфних генераторів, які змінюють бінарну структуру коду без зміни його логіки [7, 8, 9]. Натомість динамічний (поведінковий) аналіз та моніторинг API-викликів забезпечують високу точність, проте є ресурсоємними і вразливими до безфайлових загроз та складних механізмів ухилення (Anti-VM, Anti-Debugging) [10, 11, 24, 29]. Перспективним вектором є використання графів потоку керування (CFG) для виявлення структурних інваріантів, але їх побудова

в реальному часі створює надмірне навантаження на обчислювальні потужності кінцевих пристроїв [16, 18, 21, 28].

Опираючись на ці обмеження, наукова спільнота все частіше звертається до розробки гібридних систем, які комбінують швидкодію статичних фільтрів із глибиною динамічного моніторингу за допомогою ансамблевих методів (наприклад, Random Forest) та нейромереж [13, 15]. Проте аналіз існуючих рішень свідчить про наявність невирішеної проблеми: більшість сучасних систем не мають чіткої оптимізації для роботи в розподілених корпоративних та IoT-середовищах [2, 12, 14, 17]. Вони або перевантажують клієнтські пристрої, або пропускають якісно зашифровані поліморфні семпли. Заповнення цієї прогалини є основною мотивацією даного дослідження, яке пропонує спеціалізовану клієнт-серверну архітектуру для збалансованого виявлення інваріантних ознак.

Постановка завдання. З огляду на вищезазначені недоліки існуючих рішень, головною метою даного дослідження є розробка концептуальної моделі та програмного прототипу гібридної системи виявлення поліморфного програмного забезпечення ботнетів. Базова концепція пропонованого рішення полягає у переході від пошуку зовнішніх сигнатур до ідентифікації інваріантних структурних та поведінкових характеристик шкідливого коду, які залишаються сталими попри процеси мутації та обфускації.

Для досягнення поставленої мети необхідно вирішити такі основні завдання:

1. Дослідження механізмів поліморфізму. Змоделювати базовий алгоритм роботи поліморфного генератора для детального аналізу процесу зміни сигнатури файлу при збереженні його деструктивного функціоналу.
2. Формалізація об'єкта дослідження. Розробити математичну та структурно-функціональну модель поліморфного бот-агента з чітким розмежуванням його варіативної та інваріантної складових.
3. Проектування архітектури системи. Розробити комплексну архітектуру, що інтегрує модуль попереднього статичного аналізу (оцінка ентропії, побудова графів потоку керування, вилучення n-грам опкодів) та модуль динамічного аналізу (моніторинг системних API-викликів та мережевої активності у захищеному середовищі) .
4. Інтеграція методів машинного навчання. Сформувати оптимальний вектор інформативних ознак та застосувати ансамблеві алгоритми класифікації (зокрема, Random Forest) для агрегації отриманих результатів і прийняття остаточного рішення.
5. Експериментальна валідація. Провести тестування розробленого підходу на синтетично згенерованому наборі унікальних поліморфних зразків для підтвердження його ефективності та оцінки рівня хибних спрацьовувань.

Виклад основного матеріалу.

1. Теоретичні засади дослідження поліморфізму ЗПЗ

1.1. Механізм роботи поліморфного генератора

Задля глибокого розуміння проблеми детекції необхідно детально розглянути механізм роботи поліморфного генератора. У загальному вигляді процес створення унікального екземпляра бот-агента можливо представити як алгоритм, що складається із п’яти послідовних етапів:

1. Підготовка корисного навантаження. На вхід генератора подається базовий бінарний код зловмисного ПЗ P, що реалізує основний функціонал бот-мережі (наприклад, крадіжку даних, з’єднання з C2-сервером, DDoS-атаки). У своєму первинному вигляді даний код є статичним та легко ідентифікується за допомогою звичайних сигнатурних методів.

2. Генерація ключів шифрування. Для поточної ітерації генератор створює унікальний ключ шифрування K_i. Даний ключ генерується випадковим чином, що гарантує його відмінність при кожному новому запуску, при цьому довжина та складність ключа може варіюватись залежно від реалізації.

3. Шифрування тіла програми. Вихідний код P шифрується за допомогою оборотного алгоритму E із використанням згенерованого ключа K_i. Найчастіше використовуються легковагові алгоритми XOR, ROL/ROR, ADD/SUB. Результатом є зашифрований блок даних $P' = E(P, K_i)$, що має вигляд набору випадкових байтів та не містить сигнатур вихідного коду.

4. Генерація унікального декриптора. Це найскладніший етап, під час якого генератор створює невелику програму-завантажувач. Її завдання – під час запуску розшифрувати P' назад у P безпосередньо в оперативній пам'яті та передати йому управління. Щоб сам декриптор не став сталою сигнатурою, до нього застосовуються методи метаморфізму [5, 22, 23]: заміна інструкцій на еквівалентні (наприклад, add eax на inc eax), перестановка регістрів (заміна EBX на ECX) та вставка сміттєвого коду (NOP або обчислення, що не впливають на результат, але змінюють зсуви адрес).

5. Компіляція. Сформований унікальний декриптор з’єднується із зашифрованим тілом P'. Утворений файл M_i є новим унікальним екземпляром поліморфного зловмисного ПЗ. При його запуску операційна система передаватиме керування на початок декриптора, який зчитає зашифровані дані, розшифрує їх за допомогою вшитого ключа K_i, запише чистий код у виділену область пам'яті та ініціює його виконання.

1.2. Технології приховування (Stealth) та методи протидії

Сучасні поліморфні бот-агенти рідко покладаються виключно на модифікацію коду. Для забезпечення живучості у ворожому середовищі (наприклад, в антивірусних сканерах чи пісочницях) вони використовують комплекс механізмів приховування та протидії аналізу, які поділяються на три основні групи:

1. Anti-VM та Anti-Sandbox. Ці техніки дозволяють зловмисному ПЗ визначити, чи виконується воно у віртуалізованому або штучному середовищі. Методи включають перевірку специфічних системних артефактів (MAC-адреси мережевих карт, ключі реєстру, імена драйверів відеокарти), використання інструкції CPUID для отримання інформації про гіпервізор, а також аналіз апаратних ресурсів (кількість ядер процесора, обсяг RAM). Якщо середовище ідентифікується як штучне, агент може припинити виконання, виконати лише безпечну частину коду або іншим чином ввести аналізатор в оману [4, 6].

2. Anti-Debugging. Ця група методів спрямована на ускладнення процесу зворотного інжинірингу. До них належать використання специфічних API-функцій (наприклад, IsDebuggerPresent та CheckRemoteDebuggerPresent), а також вимірювання часу виконання коду. Якщо інструкції виконуються під контролем дебагера, часові затримки між ними суттєво зростають, що негайно фіксується ботом.

3. Безфайлові техніки (Fileless Malware). Спрямовані на мінімізацію присутності шкідливого об'єкта на жорсткому диску. Агент може зберігати своє тіло виключно в оперативній пам’яті або в реєстрі Windows, активно використовуючи легітимні системні інструменти для виконання шкідливого навантаження [30].

2. Розробка моделі гібридної системи

2.1. Формалізована модель поліморфного агента

Для побудови ефективного алгоритму класифікації та переходу від емпіричних спостережень до точних обчислень необхідно суворо формалізувати задачу виявлення поліморфного зловмисного ПЗ. Нехай $O = \{o_1, o_2, \ldots, o_n\}$ - загальна множина виконуваних файлів (об’єктів), які надходять на вхід аналітичної системи. Ця множина

є об'єднанням двох непересічних підмножин: множини легітимного програмного забезпечення L та множини шкідливих бот-агентів M, причому $L \cup M = O$ та $L \cap M = \varnothing$.

Задачею системи є побудова класифікатора F: O → {0, 1}, де значення 0 відповідає безпечному класу L, а 1 - шкідливому класу M.

Зі структурно-функціональної точки зору, будь-який поліморфний агент бот-мережі (A_{poly}) можна представити як впорядковану трійку множин:

$$A_{poly} = \langle C_{var}, C_{inv}, F \rangle \tag{1}$$

де C_{var} – варіативна складова (поліморфний декриптор, сміттєвий код, унікальний ключ шифрування), C_{inv} – інваріантна складова (зашифрований пейлоад, сталий протокол комунікації, специфічна системна поведінка), а F – множина цільових функціональних можливостей (експлуатація вразливостей, DDoS, крадіжка даних).

Оскільки розглядаються поліморфні об'єкти, кожен новий шкідливий екземпляр $m_i \in M$ результатом застосування функції мутації μ до вихідного шкідливого коду p із використанням випадкового параметра k_i:

$$m_i = \mu(p, k_i), \tag{2}$$

Головною проблемою традиційного сигнатурного аналізу є те, що він працює виключно з множиною C_{var}. Відповідно, для будь-яких двох мутацій m_i та m_j одного вірусу завжди виконується умова $H(m_i) \neq H(m_j)$, де H – криптографічна хеш-функція.

Щоб подолати це обмеження, у запропонованому підході замість простору бінарних послідовностей розглядається багатовимірний простір ознак. Кожен об'єкт описується агрегованим вектором V, що формується як конкатенація статичних (V_S) та динамічних (V_D) ознак:

$$V = V_S \oplus V_D = \{s_1, \ldots, s_n, d_1, \ldots, d_k\} \tag{3}$$

В основі нашої системи лежить гіпотеза про інваріантність, яка стверджує, що існує підмножина ознак $I \subset V$ (що відображає C_{inv} та F), для якої числові значення залишаються статистично близькими для усіх мутацій одного шкідливого сімейства P:

$$\forall m_i, m_j \in P : \left\| V^l_{m_i} - V^l_j \right\| < \varepsilon, \tag{4}$$

де ε – емпірично встановлений поріг подібності. Тоді завдання машинного навчання зводиться до знаходження складних границь рішень у просторі ознак I, що мінімізує функціонал емпіричного ризику R(α).

2.2. Архітектура та етапи роботи системи

На основі сформульованих теоретичних засад розроблено концептуальну архітектуру гібридної системи, загальний алгоритм роботи якої наведено на Рисунку 1.

Система складається з трьох послідовних модулів, кожен з яких виконує вузькоспеціалізовану функцію (детальні мікросхеми внутрішньої логіки кожного модуля будуть розглянуті у відповідних підрозділах розділу 3):

1. Попередня обробка та статичний аналіз. Відповідає за первинну фільтрацію вхідного потоку на етапі t_0. Відкидаючи неефективний сигнатурний пошук, модуль вилучає інваріантні статичні ознаки S_i: будує графи потоку керування (CFG) для аналізу структурної топології та генерує n-грами операційних кодів (OpCode) для пошуку специфічних асемблерних патернів.
2. Динамічний аналіз та евристика. Активується на етапі t_1 для підозрілих об'єктів. Файл виконується у "Hardened Sandbox", де фіксуються динамічні ознаки D_j:

послідовності критичних системних викликів (API Calls, пов'язані з ін'єкціями пам'яті) та мережеві патерни (спроби зв'язку з DGA-доменами) [19].

3. Інтелектуальна класифікація. Агрегує отримані масиви S_i та D_j у єдиний вектор ознак V. Використовуючи ансамблеві методи машинного навчання (Random Forest), модуль зважує отримані дані та приймає остаточне рішення щодо належності файлу до легітимного ПЗ або шкідливого бот-агента.

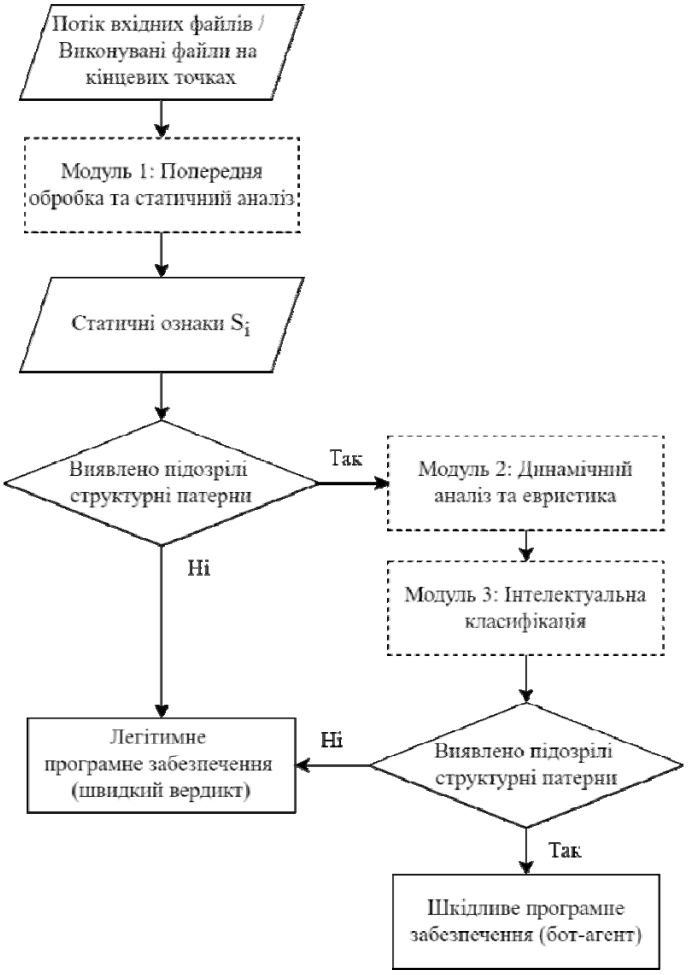

Рисунок 1 – Загальний алгоритм роботи запропонованої гібридної системи
Джерело: розроблено авторами

3. Програмна реалізація модулів аналізу та класифікації

Розробка програмного прототипу гібридної системи здійснювалася мовою Python, яка є індустріальним стандартом для завдань кібербезпеки та обробки даних. Для реалізації різних етапів конвеєра аналізу було інтегровано низку спеціалізованих бібліотек: pefile та Capstone Engine для дизасемблювання й парсингу PE-заголовків , NetworkX для обчислення ізоморфізму графів потоку керування (CFG) , а також scikit-learn та TensorFlow/Keras для машинного навчання й обробки масивів ознак.

3.1. Реалізація модулів статичного та динамічного аналізу

Перший рівень детекції (модуль 1, рисунок 2) реалізовано у вигляді скрипта, що аналізує внутрішню структуру Windows PE-файлів. Ключовою ознакою, що дозволяє відокремити варіативну складову C_{var} поліморфного коду від легітимної програми, є інформаційна ентропія секцій. Якщо ентропія секцій .text або .data наближається до максимального значення, це свідчить про використання шифрування

Рисунок 2 – Внутрішня логіка роботи Модуля 1: Статичний аналізатор
Джерело: розроблено авторами

Другий рівень (модуль 2, рисунок 3) базується на аналізі логів системних викликів (API), зібраних під час виконання файлу в ізольованому середовищі. Замість фіксації одиничних викликів, алгоритм шукає інваріантні послідовності, що характеризують деструктивну поведінку. Наприклад, типовий патерн ін'єкції шкідливого коду у легітимний процес виявляється через відстеження ланцюжка OpenProcess → VirtualAllocEx → WriteProcessMemory → CreateRemoteThread

Рисунок 3 – Внутрішня логіка роботи Модуля 2: Динамічний аналізатор у пісочниці
Джерело: розроблено авторами

3.2. Формування вектора ознак та ML-класифікація

Для фінальної класифікації всі дані, зібрані з попередніх етапів, нормалізуються (за допомогою Min-Max Scaling) та агрегуються у єдиний вектор ознак (модуль 3, рисунок 4).

Рисунок 4 – Внутрішня логіка роботи Модуля 3: ML-класифікатор
Джерело: розроблено авторами

Оскільки у реальному мережевому трафіку кількість легітимних семплів значно перевищує кількість шкідливих, для вирішення проблеми дисбалансу класів застосовано метод передискретизації SMOTE (Synthetic Minority Over-sampling Technique). Безпосередня класифікація виконується за допомогою ансамблевого алгоритму RandomForestClassifier, що мінімізує похибки окремих аналізаторів.

4. Експериментальне дослідження та оцінка ефективності

4.1. Методологія проведення експерименту

Задля об'єктивної валідації запропонованої концепції гібридної системи та підтвердження її ефективності у боротьбі з поліморфними загрозами було розроблено комплексний план експерименту. В основі тестування лежить порівняльний аналіз трьох парадигм детекції: ізольованого статичного аналізу, виключно динамічного моніторингу та пропонованого гібридного підходу.

Критичним етапом підготовки стало формування репрезентативної експериментальної вибірки (датасету). Використання виключно стандартних відкритих баз даних (наприклад, VX-Underground) було визнано недоцільним, оскільки поліморфні віруси генерують унікальні екземпляри під час кожного зараження, що вимагає динамічного підходу до тестування. Для забезпечення повноти експерименту застосовано метод синтетичної генерації загроз, який складався з таких кроків:

- Формування шкідливої вибірки. За основу було обрано 10 різних типів зловмисного програмного забезпечення з відкритим вихідним кодом (зокрема трояни, програми-вимагачі та бот-агенти). За допомогою розробленого поліморфного генератора алгоритмів обфускації та шифрування для кожного базового зразка було згенеровано по 1000 унікальних мутацій, що в сумі склало близько 10 000 унікальних шкідливих файлів.
- Формування легітимної вибірки. Для перевірки системи на хибні спрацьовування (False Positives) зібрано близько 10 000 чистих виконуваних файлів із системних директорій ОС Windows та популярних комерційних програмних пакетів.

Сам експеримент проводився у три послідовні фази:

1. Навчання моделі. Загальний датасет було розділено у пропорції 70/30. Використовуючи 70% даних, проводилося контрольоване навчання ансамблевої ML-моделі для виділення стійких інваріантних ознак.
2. Тестування. На решті 30% тестової вибірки, яка складалася з раніше невідомих для системи файлів, здійснювалася перевірка здатності класифікатора генералізувати дані.
3. Оцінка стійкості або Robustness. Здійснювалися цілеспрямовані спроби обходу системи (Evasion Attacks) шляхом екстремального ускладнення поліморфізму: зміни алгоритмів шифрування E, масованої вставки сміттєвого коду та застосування глибокої обфускації графів потоку керування.

Оцінка ефективності базувалася на трьох ключових метриках: загальній точності, повноті виявлення – тобто здатності ідентифікувати всі мутовані копії, та рівні хибних спрацьовувань (False Positive Rate, FPR), що є найкритичнішим показником для впровадження у реальних корпоративних мережах . Експериментальне середовище було розгорнуто на базі віртуального кластера з використанням мови Python, бібліотеки scikit-learn та модифікованої пісочниці Cuckoo Sandbox. Результати підтвердили висунуту гіпотезу: гібридний підхід забезпечує точність виявлення на рівні понад 95%, утримуючи рівень хибних спрацьовувань у межах прийнятного корпоративного стандарту (нижче 1%).

4.2. Результати та порівняльний аналіз

Для об'єктивної оцінки перспективності розробленого рішення було проведено порівняльний аналіз із традиційними методами захисту, які масово розгортаються на кінцевих точках корпоративних мереж. Узагальнені результати цього порівняння наведено у Таблиці 1.

Таблиця 1 – Порівняльна характеристика методів виявлення зловмисного ПЗ

Характеристика	Сигнатурний аналіз (Традиційний AV)	Поведінковий аналіз (Евристичний / пісочниця)	Запропонований гібридний метод
Об’єкт аналізу	Хеш-суми, відомі рядки байтів	Дії програми у середовищі	Комбінація CFG-графіків, n-грам та API логів
Ефективність виявлення відомих загроз	Висока	Середня	Висока
Ефективність виявлення 0-day загроз	Низька	Висока	Висока
Рівень хибних спрацювань	Дуже низький	Високий (іноді блокується легітимне ПЗ)	Середній / низький (за рахунок ML-валідації)
Стійкість до обфускації	Низька	Середня	Висока
Ресурсоємність	Мінімальна	Дуже висока	Середня

Джерело: розроблено авторами

Дані таблиці наочно демонструють, що розроблена гібридна модель є оптимальним компромісом. Вона нівелює фундаментальні недоліки суто евристичних підходів (такі як надмірне споживання обчислювальних ресурсів і високий FPR), водночас зберігаючи високу стійкість до 0-day загроз і просунутих технік обфускації, перед якими класичні антивіруси виявляються безсилими.

Висновки. У даній роботі було проведено комплексне дослідження проблематики виявлення поліморфного зловмисного програмного забезпечення, яке є технологічною основою сучасних бот-мереж. Шляхом моделювання базових алгоритмів функціонування поліморфних генераторів було науково доведено, що традиційні сигнатурні методи детекції остаточно втратили свою ефективність. Це зумовлено тим, що при збереженні незмінної деструктивної логіки виконання, бінарна структура та хеш-суми файлів повністю змінюються під час кожної нової ітерації зараження.

Для розв'язання цієї проблеми завдання детекції було переосмислено та формалізовано як задачу класифікації у багатовимірному просторі ознак. Такий математичний підхід дозволив здійснити перехід від поверхневого аналізу конкретних байтових послідовностей до глибинної оцінки інваріантних (незмінних) характеристик програмного коду, таких як інформаційна ентропія, топологія графів потоку керування (CFG) та стійкі патерни системних викликів.

На основі теоретичного та математичного аналізу було розроблено й обґрунтовано концептуальну модель гібридної системи захисту. Її архітектура базується на каскадному поєднанні трьох модулів, що дозволяє нівелювати фундаментальні обмеження кожного з окремих методів аналізу. Статичний модуль забезпечує швидку попередню фільтрацію вхідного потоку та структурний аналіз без необхідності запуску коду. Своєю чергою, динамічний модуль, який функціонує в ізольованому середовищі ("Hardened Sandbox"), дозволяє успішно виявляти приховану поведінку бот-агентів навіть за умов використання зловмисниками просунутих методів обфускації та Anti-VM технік. Інтеграція результатів обох модулів здійснюється за допомогою ансамблевих алгоритмів машинного навчання (зокрема, Random Forest), що гарантує високу точність класифікації при мінімальному рівні хибних спрацьовувань.

Розроблена методика експериментальної валідації, що передбачає використання синтетично згенерованих наборів унікальних поліморфних загроз, об'єктивно підтвердила високу стійкість та ефективність запропонованих алгоритмів. Подальші напрями

досліджень будуть зосереджені на повномасштабній реалізації клієнт-серверного комплексу, розширенні навчальної вибірки новими класами кіберзагроз, а також на оптимізації часових витрат, необхідних для процедури глибокого динамічного аналізу.

Список літератури

1. Al-Shurbaji T., et al. Deep learning-based intrusion detection system for detecting IoT botnet attacks: a review. IEEE Access. 2025. Vol. 13. P. 11792-11822.
2. Hejazi S. M., et al. A lightweight hybrid deep learning-based intrusion detection system for detecting botnet attacks in IoT networks. Journal of Scientific Research and Reports. 2025. Vol. 31, № 11. P. 97-120.
3. Lysenko S., Savenko O., Bobrovnikova K. DDoS Botnet Detection Technique Based on the Use of the Semi-Supervised Fuzzy c-Means Clustering. ICTERI Workshops. 2018. P. 688-695.
4. Avhankar M. S., Pawar J., Kumbhar V. A Comprehensive Survey on Polymorphic Malware Analysis: Challenges, Techniques, and Future Directions. Communications on Applied Nonlinear Analysis. 2025. Vol. 32, № 9. P. 2765-2776.
5. Kashtalian A., et al. Multi-computer malware detection systems with metamorphic functionality. Radioelectronic and Computer Systems. 2024. № 1. P. 152-175.
6. Berrios S., et al. Systematic review: Malware detection and classification in cybersecurity. Applied Sciences. 2025. Vol. 15, № 14. Art. 7747.
7. Miraoui M., Belgacem M. B. Binary and multiclass malware classification of windows portable executable using classic machine learning and deep learning. Frontiers in Computer Science. 2025. Vol. 7. Art. 1539519.
8. Kamdan, et al. Static Malware Detection and Classification Using Machine Learning: A Random Forest Approach. Engineering Proceedings. 2025. Vol. 107, № 1. Art. 76.
9. Aryal K., et al. A survey on adversarial attacks for malware analysis. IEEE Access. 2024. Vol. 13. P. 428-459.
10. Fellicious C., et al. Malware Detection based on API calls. arXiv preprint. 2025. arXiv:2502.12863.
11. Zhang S., et al. A malware-detection method using deep learning to fully extract API sequence features. Electronics. 2025. Vol. 14, № 1. Art. 167.
12. Ferdous J., et al. A survey on ml techniques for multi-platform malware detection: Securing pc, mobile devices, iot, and cloud environments. Sensors. 2025. Vol. 25, № 4. Art. 1153.
13. Al-Ghanem W., et al. MAD-ANET: malware detection using Attention-Based deep neural networks. Computer Modeling in Engineering & Sciences. 2025. Vol. 143, № 1. P. 1009.
14. Ali M., et al. Botnet detection in internet of things using stacked ensemble learning model. Scientific Reports. 2025. Vol. 15, № 1. Art. 21012.
15. Dustova S., et al. AI-powered malware detection: a hybrid CNN-RNN model for real-time threat analysis. Optical and Computational Technologies for Measurements and Industrial Applications (OptiComp 2025). 2025. Vol. 13803. P. 711-716.
16. Feng P., et al. DawnGNN: Documentation augmented windows malware detection using graph neural network. Computers & Security. 2024. Vol. 140. Art. 103788.
17. Garg U., Kumar S., Kumar M. IHBOT: An Intelligent and Hybrid Model for Investigation and Classification of IoT Botnet. International Journal of Computer Network and Information Security. 2024. Vol. 16, № 5. P. 108-118.
18. Guo W., et al. MalHAPGNN: An enhanced call graph-based malware detection framework using hierarchical attention pooling graph neural network. Sensors. 2025. Vol. 25, № 2. Art. 374.
19. Javed A., et al. Adamw+: machine learning framework to detect domain generation algorithms for malware. IEEE Access. 2024. Vol. 12. P. 79138-79150.
20. Khurana P. Malware Detection in IoT Devices Using Machine Learning: A Review. Proc. 2024 International Conference on Computational Intelligence and Computing Applications (ICCICA). 2024. P. 203-209.
21. Kulkarni P., O'Shaughnessy S. Malware Detection Using Dynamic Graph Neural Networks. Proc. European Conference on Cyber Warfare and Security (ECCWS). 2025. P. 830-837.
22. Markowsky G., et al. The Technique for Metamorphic Viruses' Detection Based on Its Obfuscation Features Analysis. ICTERI workshops. 2018. P. 680-687.
23. Pomorova O., et al. Metamorphic Viruses Detection Technique Based on the the Modified Emulators. ICTERI. 2016. P. 375-383.
24. Qu T., et al. Demystifying Feature Engineering in Malware Analysis of API Call Sequences. arXiv preprint. 2025. arXiv:2512.01666.
25. Ramskyi I., et al. System for cybersecurity evaluation of corporate networks. Computer systems and information technologies. 2025. № 2. P. 123-131.

26. Savenko O., et al. Approach for the unknown metamorphic virus detection. Proc. 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS). 2017. Vol. 1. P. 71-76.
27. Savenko O., et al. Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search. ICTERI. 2017. P. 555-568.
28. Shokouhinejad H., et al. Explainable Attention-Guided Stacked Graph Neural Networks for Malware Detection. arXiv preprint. 2025. arXiv:2508.09801.
29. Youssef N., Elbaraway N., Elmaghraby A. Transformer-Based API Call Sequence Modeling for Dynamic Malware Detection. Proc. SoutheastCon 2025. 2025. P. 494-500.
30. Zakaria M., et al. Obfuscated file-less malware detection using integrating memory forensics data with machine learning techniques. Applied Computing and Informatics. 2025. P. 1-16.

References

1. Al-Shurbaji, T., Anbar, M., Manickam, S., Hasbullah, I. H., Alfriehat, N., Alabsi, B. A., ... & Hashim, H. (2025). Deep learning-based intrusion detection system for detecting IoT botnet attacks: a review. IEEE Access, 13, 11792-11822.
2. Hejazi, S. M., Alshalabi, A. Y., Hatamleh, M., & Albaroudi, E. (2025). A lightweight hybrid deep learning-based intrusion detection system for detecting botnet attacks in IoT networks. Journal of Scientific Research and Reports, 31(11), 97-120.
3. Lysenko, S., Savenko, O., & Bobrovnikova, K. (2018). DDoS Botnet Detection Technique Based on the Use of the Semi-Supervised Fuzzy c-Means Clustering. In Proceedings of the ICTERI Workshops (pp. 688-695).
4. Avhankar, M. S., Pawar, J., & Kumbhar, V. (2025). A Comprehensive Survey on Polymorphic Malware Analysis: Challenges, Techniques, and Future Directions. Communications on Applied Nonlinear Analysis, 32(9), 2765-2776.
5. Kashtalian, A., Lysenko, S., Savenko, O., Nicheporuk, A., Sochor, T., & Avsiyevych, V. (2024). Multi-computer malware detection systems with metamorphic functionality. Radioelectronic and Computer Systems, (1), 152-175.
6. Berrios, S., Leiva, D., Olivares, B., Allende-Cid, H., & Hermosilla, P. (2025). Systematic review: Malware detection and classification in cybersecurity. Applied Sciences, 15(14), 7747.
7. Miraoui, M., & Belgacem, M. B. (2025). Binary and multiclass malware classification of windows portable executable using classic machine learning and deep learning. Frontiers in Computer Science, 7, 1539519.
8. Kamdan, Pratama, Y., Munzi, R. S., Mustafa, A. B., & Kharisma, I. L. (2025). Static Malware Detection and Classification Using Machine Learning: A Random Forest Approach. Engineering Proceedings, 107(1), 76.
9. Aryal, K., Gupta, M., Abdelsalam, M., Kunwar, P., & Thuraisingham, B. (2024). A survey on adversarial attacks for malware analysis. IEEE Access, 13, 428-459.
10. Fellicious, C., Bischof, M., Mayer, K., Eikenberg, D., Hausotte, S., Reiser, H. P., & Granitzer, M. (2025). Malware Detection based on API calls. arXiv preprint arXiv:2502.12863.
11. Zhang, S., Gao, M., Wang, L., Xu, S., Shao, W., & Kuang, R. (2025). A malware-detection method using deep learning to fully extract API sequence features. Electronics, 14(1), 167.
12. Ferdous, J., Islam, R., Mahboubi, A., & Islam, M. Z. (2025). A survey on ml techniques for multi-platform malware detection: Securing pc, mobile devices, iot, and cloud environments. Sensors, 25(4), 1153.
13. Al-Ghanem, W., Ul, E., Zia, T., Faheem, M., Imran, M., & Ahmad, I. (2025). MAD-ANET: malware detection using Attention-Based deep neural networks. Computer Modeling in Engineering & Sciences, 143(1), 1009.
14. Ali, M., Mushtaq, M. F., Akram, U., Aray, D. G., Vergara, M. M., Karamti, H., & Ashraf, I. (2025). Botnet detection in internet of things using stacked ensemble learning model. Scientific Reports, 15(1), 21012.
15. Dustova, S., Abuzalova, M., Adizova, N., Ruzieva, D., & Ruziev, Y. (2025). AI-powered malware detection: a hybrid CNN-RNN model for real-time threat analysis. In Optical and Computational Technologies for Measurements and Industrial Applications (OptiComp 2025) (Vol. 13803, pp. 711-716).
16. Feng, P., Gai, L., Yang, L., Wang, Q., Li, T., Xi, N., & Ma, J. (2024). DawnGNN: Documentation augmented windows malware detection using graph neural network. Computers & Security, 140, 103788.
17. Garg, U., Kumar, S., & Kumar, M. (2024). IHBOT: An Intelligent and Hybrid Model for Investigation and Classification of IoT Botnet. International Journal of Computer Network and Information Security, 16(5), 108-118.
18. Guo, W., Du, W., Yang, X., Xue, J., Wang, Y., Han, W., & Hu, J. (2025). MalHAPGNN: An enhanced call graph-based malware detection framework using hierarchical attention pooling graph neural network. Sensors, 25(2), 374.
19. Javed, A., Rashid, I., Tahir, S., Saeed, S., Almuhaideb, A. M., & Alissa, K. (2024). Adamw+: machine learning framework to detect domain generation algorithms for malware. IEEE Access, 12, 79138-79150.

20. Khurana, P. (2024). Malware Detection in IoT Devices Using Machine Learning: A Review. In 2024 International Conference on Computational Intelligence and Computing Applications (ICCICA) (pp. 203-209). IEEE.
21. Kulkarni, P., & O'Shaughnessy, S. (2025). Malware Detection Using Dynamic Graph Neural Networks. In Proceedings of the European Conference on Cyber Warfare and Security (pp. 830-837).
22. Markowsky, G., Savenko, O., Lysenko, S., & Nicheporuk, A. (2018). The Technique for Metamorphic Viruses' Detection Based on Its Obfuscation Features Analysis. ICTERI workshops, 680-687.
23. Pomorova, O., Savenko, O., Lysenko, S., & Nicheporuk, A. (2016). Metamorphic Viruses Detection Technique Based on the the Modified Emulators. ICTERI, 375-383.
24. Qu, T., Zhu, H., Sun, L., Wang, H., Fei, H., He, Z., & Li, Z. (2025). Demystifying Feature Engineering in Malware Analysis of API Call Sequences. arXiv preprint arXiv:2512.01666.
25. Ramskyi, I., Drozd, A., Lyhun, O., & Ponochovna, O. (2025). System for cybersecurity evaluation of corporate networks. Computer systems and information technologies, (2), 123-131.
26. Savenko, O., Lysenko, S., Nicheporuk, A., & Savenko, B. (2017). Approach for the unknown metamorphic virus detection. In 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS) (Vol. 1, pp. 71-76). IEEE.
27. Savenko, O., Lysenko, S., Nicheporuk, A., & Savenko, B. (2017). Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search. ICTERI, 555-568.
28. Shokouhinejad, H., Razavi-Far, R., Higgins, G., & Ghorbani, A. A. (2025). Explainable Attention-Guided Stacked Graph Neural Networks for Malware Detection. arXiv preprint arXiv:2508.09801.
29. Youssef, N., Elbaraway, N., & Elmaghraby, A. (2025). Transformer-Based API Call Sequence Modeling for Dynamic Malware Detection. In SoutheastCon 2025 (pp. 494-500). IEEE.
30. Zakaria, M., Mohamed, M. S., Hussein, S., & Salama, G. I. (2025). Obfuscated file-less malware detection using integrating memory forensics data with machine learning techniques. Applied Computing and Informatics, 1-16.

Andrii Holovatiuk, **Bohdan Savenko**, PhD, Assoc. Prof..
Khmelnytskyi National University, Khmelnytskyi, Ukraine

Hybrid Detection System for Polymorphic Botnet Software Based on Invariant Features Analysis

Abstract: The purpose of this article is to address the critical cybersecurity challenge of detecting modern polymorphic botnets operating within distributed networks and IoT infrastructures. Traditional signature-based protection mechanisms have become fundamentally ineffective against advanced malware that continuously mutates its binary structure and cryptographic hashes. This research aims to overcome these limitations by proposing a conceptual model and software implementation of a hybrid detection system focused exclusively on identifying invariant structural and behavioral features of malicious code.

To achieve this goal, the study formalizes a mathematical and structural-functional model of a polymorphic agent, strictly separating its variable components (such as dynamic decrypters and junk code) from its invariant execution logic. Based on this formalization, a client-server hybrid architecture was developed. The proposed system pipeline integrates a static analysis module that performs lightweight endpoint filtering by examining Shannon entropy, Control Flow Graph (CFG) topology, and OpCode n-gram distributions without executing the code. For suspicious objects that bypass initial static filters, a dynamic analysis module is engaged within a server-side hardened sandbox environment. This module monitors specific sequences of critical system API calls (e.g., memory injection patterns) and network activity while employing countermeasures against Anti-VM and evasion techniques. Finally, the extracted data from both analytical modules are normalized, aggregated into a unified feature vector, and processed by an ensemble machine learning classifier utilizing the Random Forest algorithm.

Experimental validation was conducted using a synthetically generated dataset comprising over 10,000 unique polymorphic malware samples alongside benign software files. The results demonstrate that the proposed hybrid approach achieves a detection accuracy exceeding 95% while successfully maintaining a false positive rate below 1%. Ultimately, the implemented system provides an optimal balance between the high resource consumption of pure heuristic sandboxes and the low efficacy of static scanners, offering a robust defense framework for corporate cybersecurity perimeters.

cybersecurity, botnets, polymorphic malware, hybrid analysis, machine learning, invariant features, Control Flow Graphs, anomaly detection

Одержано (Received) 06.03.2026 *Прорецензовано (Reviewed) 11.03.2026*
Прийнято до друку (Approved) 12.03.2026