DOI 10.15276/imms.v14.no3.249
УДК 004.457

Informatics and Mathematical Methods in Simulation
Vol.14 (2024), No. 3, pp. 249-258

ЗАСТОСУВАННЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ВИЯВЛЕННЯ РОБОТИ КЕЙЛОГЕРІВ В ОПЕРАЦІЙНІЙ СИСТЕМІ

Г.Д. Шибаєв[1], Л.Ю. Гальчинський[2]

Київський політехнічний інститут імені Ігоря Сікорського
37, Берестейський пр., м. Київ, 03056, Україна
Emails: K233@ukr.net[1]; hleonid@gmail.com[2]

Представлено методологію виявлення кейлогерів за допомогою методів машинного навчання. Кейлогери – це поширена форма зловмисного програмного забезпечення, яке фіксує натискання клавіш, щоб викрасти конфіденційні дані, створюючи серйозну загрозу безпеці для користувачів і організацій. Традиційні методи виявлення часто покладаються на підходи на основі сигнатур, які можна обійти розширеними, поліморфними або без файловими кейлогерами. Це дослідження використовує машинне навчання (ML) для виявлення аномальної поведінки та шаблонів, що вказують на дії кейлогера.

Автори досліджують різні моделі машинного навчання, включаючи дерева рішень, опорні векторні машини (SVM), випадкові ліси та нейронні мережі, щоб класифікувати звичайну та шкідливу поведінку систем. Було проведено експерименти для оцінки точності, точності, запам'ятовування та оцінки F1 різних моделей ML. Результати показують, що методи, засновані на ML, можуть значно підвищити рівень виявлення порівняно з традиційними методами. У дослідженні також обговорюються потенційні проблеми, такі як помилкові спрацьовування, узагальнення моделі для нових варіантів кейлогерів і необхідність постійного навчання для адаптації до зловмисного програмного забезпечення, що розвивається. Отримані дані свідчать про те, що система виявлення на основі ML може запропонувати надійне та адаптивне рішення для боротьби з кейлогерами, підкреслюючи важливість інтеграції штучного інтелекту в інфраструктуру кібербезпеки.

Ключові слова: машинне навчання, кейлогер, API-функції, штучний інтелект

Вступ. Виявлення кейлогерів залишається надзвичайно актуальним у сучасному інформаційному. Незважаючи на розвиток технологій безпеки, кейлогери продовжують бути ефективними засобами для порушення конфіденційності, цілісності та доступності даних у операційній системі. Виявлення клавіатурних шпигунів є критично важливим, оскільки вони становлять серйозну загрозу окремим особам, організаціям і навіть національній безпеці, тихо фіксуючи та записуючи кожне натискання клавіш на комп'ютері чи мобільному пристрої. Кейлогери можуть записувати паролі, дані кредитних карток, банківську інформацію та персональні ідентифікаційні номери (PIN). Якщо цю інформацію перехоплять зловмисники, це може призвести до крадіжки особистих даних, фінансових втрат і несанкціонованого доступу до приватних облікових записів. Кейлогери можна використовувати для викрадення конфіденційної ділової інформації, включаючи комерційні таємниці, інтелектуальну власність, дані клієнтів і внутрішні комунікації. Це може призвести до значних фінансових втрат, шкоди репутації та юридичних наслідків. Кейлогери підривають особисту конфіденційність, відстежуючи не лише паролі та конфіденційні фінансові дані, але й особисті розмови, електронні листи та інші типи спілкування.

Враховуючи ці значні ризики, виявлення та нейтралізація клавіатурних шпигунів є першочерговим завданням у сфері кібербезпеки, спрямованим на захист особистої конфіденційності, захист організаційних активів і запобігання масштабній шкоді в

різних секторах.

Мета роботи полягає у визначення можливості виявлення роботи кейлогера на основі використання методів штучного інтелекту в режимі реального часу, що включає детальний аналіз існуючих методів захисту на основі методів машинного навчання.

Аналіз та особливості кейлогерів у операційній системі. Залежно від способу роботи та структури кейлогери можна в цілому класифікувати на програмні та апаратні кейлогери. В даній роботі ми розглядаємо тільки програмні кейлогери. Програмні кейлогери - це програми, які приховано відстежують і записують натискання клавіш і часто вбудовані в інше шкідливе програмне забезпечення. Програмні кейлогери додатково класифікуються в залежності від рівня привілеїв, які необхідні для функціонування. З повними привілеями працюють кейлогери, що написані на рівні ядра. У свою чергу програмні кейлогери рівня користувача можна класифікувати на такі типи:

Кейлогер на основі API. Це тип програмного кейлогера, який перехоплює та записує натискання клавіш шляхом підключення до API операційної системи (інтерфейс прикладного програмування)[1]. API надають набір інструментів і функцій, які розробники використовують для взаємодії з основною системою, дозволяючи програмам виконувати такі завдання, як обробка введення з клавіатури, керування файлами або доступ до обладнання. Кейлогер на основі API використовують ці легітимні API для захоплення подій клавіатури під час їх обробки операційною системою, що робить їх дуже ефективними та небезпечними. Кейлогер на основі API працює шляхом підключення до системних API, які обробляють введення з клавіатури. Ці клавіатурні шпигуни використовують законні функції операційної системи, які використовують програми для виявлення натискань клавіш користувача та відповіді на них. Кейлогери на основі API часто використовуються, оскільки вони відносно прості у реалізації та можуть бути приховано розгорнуті як частина шкідливої програми, такої як троян або сценарій на основі браузера. Оскільки кейлогер на основі API використовують легітимні системні функції, їм не потрібні привілеї на системному рівні або доступ до ядра, що зменшує ймовірність їх виявлення традиційними антивірусними програмами або програмами для захисту від шкідливих програм.

Кейлогери на основі API покладаються на підключення або перехоплення викликів API, які керують введенням з клавіатури. Наведемо типовий сценарій роботи кейлогера цього типу:

1. Генерація ключової події. Коли користувач натискає клавішу на клавіатурі, апаратне забезпечення клавіатури генерує сигнал, який надсилається в операційну систему. Цей сигнал відповідає певному коду ключа (наприклад, клавіша «Б» пов’язана з певним кодом).

2. Перехоплення API. Перш ніж натискання клавіші досягне цільової програми, воно проходить через певні системні API, відповідальні за керування введенням, наприклад GetAsyncKeyState, GetKeyState або TranslateMessage у операційній системі Windows. Кейлогер на основі API підключається до цих функцій, вставляючи себе в процес, який обробляє ці виклики API.

3. Перехоплення натискань клавіш. Після підключення кейлогер фіксує дані про натискання клавіш під час проходження через систему. Ці дані зазвичай включають код віртуальної клавіші (вказує, яку клавішу було натиснуто) та іншу інформацію, таку як модифікатори (наприклад, Shift, Ctrl), щоб визначити, чи була введена велика літера чи символ.

4. Реєстрація даних. Потім перехоплені натискання клавіш реєструються кейлогером шляхом запису їх у локальний файл у системі або надсилання даних на віддалений сервер, контрольований зловмисником.

Кейлогер для захоплення форм. Це дуже небезпечний і складний тип зловмисного програмного забезпечення, призначене для перехоплення та захоплення даних, введених у веб-форми, перед тим, як вони будуть безпечно передані через Інтернет [11].

Наведемо типовий сценарій роботи кейлогера для захоплення форм.

1. Зараження та встановлення. Захоплювачі форм зазвичай доставляються через фішингові електронні листи, миттєві завантаження (шкідливе програмне забезпечення.

2. Підключення до процесу надсилання форми у браузері. Після інсталяції програма захоплення форм підключається до процесу браузера для обробки надсилання форм. Це робиться за допомогою методу, відомого як перехоплення API, який дозволяє кейлогеру перехоплювати зв'язок між браузером і основною операційною системою.

3. Перехоплення даних форми, Коли користувач надсилає форму, захоплювач форм перехоплює дані безпосередньо перед тим, як вони будуть зашифровані для передачі через Інтернет.

4. Відправлення на віддалений сервер. Після захоплення даних форми програма захоплення форм передає викрадену інформацію на віддалений сервер, контрольований зловмисником.

5. Націлювання на конкретні веб-сайти або форми, Розширені засоби захоплення форм часто налаштовані на конкретні веб-сайти або типи форм.

Кейлогери ін'єкції пам'яті: кейлогери ін'єкції пам'яті представляють складний клас зловмисного програмного забезпечення, яке працює шляхом ін'єкції шкідливого коду безпосередньо в пам'ять запущених процесів. Наведемо типовий сценарій роботи кейлогера ін'єкції пам'яті [8].

1. Початкове зараження. Клавіатурні шпигуни з ін'єкцією пам'яті зазвичай отримують початковий доступ до системи за допомогою традиційних механізмів доставки зловмисного програмного забезпечення, таких як фішингові електронні листи, шкідливі завантаження або використання вразливостей у програмному забезпеченні.

2. Введення коду в пам'ять. Після зараження системи кейлогер впроваджує свій шкідливий код у пам'ять існуючого законного процесу. Це часто робиться за допомогою таких методів, як впровадження DLL або видалення процесу.

3. Перехоплення пам'яті. Після введення кейлогер підключається до ключових системних API або функцій, які використовуються цільовим процесом для захоплення вхідних даних. Наприклад, у системах Windows кейлогер може підключатися до таких функцій, як GetAsyncKeyState() або GetKeyboardState(), які відповідають за обробку введення з клавіатури.

4. Збір даних. Коли користувач вводить текст, клавіатурний шпигун реєструє кожне натискання клавіші в реальному часі. Оскільки кейлогер працює в пам'яті "законного" процесу, натискання клавіш фіксуються в точці, де вони все ще знаходяться у вигляді відкритого тексту та ще не зашифровані чи захищені заходами безпеки.

5. Викрадення даних. Після того як кейлогер зафіксує натискання клавіш, дані зберігаються в пам'яті або передаються безпосередньо на віддалений сервер, яким керує зловмисник.

Небезпека присутності програм-шпигунів у пам'яті компютерів дуже велика, а виявлення їх присутності суттєво ускладнено. Анти-кейлогери на основі сигнатур та евристичного аналізу можуть захистити ваш комп'ютер від клавіатурних шпигунів. Анти-кейлогери на основі сигнатур виправдовують свою назву, тим що шукають сигнатури клавіатурних шпигунів, тоді як анти-кейлогери з евристичним аналізом працюють, тестуючи сумнівні програми в контрольованому середовищі. З цих двох типів анти-кейлогери на основі сигнатур є найпоширенішими. Причому анти-кейлогерів на основі сигнатур набагато більше, ніж анти-кейлогерів з евристичним аналізом. Але недоліком використання анти-кейлогера на основі сигнатур є те, що він захищає лише від відомих і записаних кейлогерів. Недоліком евристичного підходу є те, що анти-кейлогери на основі евристичного аналізу схильні до помилкових спрацьовувань. Вони можуть помилково ідентифікувати законну програму як кейлогер. Відтак, розглянемо можливості протидії загрозам кейлогерам на основі алгоритмів машинного навчання. Зауважимо, що незважаючи на різні способи реалізації кейлогерів режиму користувача,

усіх їх об’єднує спільний механізм – використання специфічних API-функцій.

Алгоритми машинного навчання для виявлення роботи кейлогера. Класифікація є фундаментальним завданням у машинному навчанні (ML)[2], мета якої полягає в тому, щоб передбачити мітку або категорію для даного вхідного матеріалу на основі його характеристик. Це тип навчання під наглядом, що означає, що модель вивчає дані з мітками — дані, які попередньо класифіковані за категоріями.

У машинному навчанні класифікація відноситься до процесу прогнозування класу або категорії спостереження на основі вхідних даних (ознак)[3]. Результатом є окрема мітка або категорія, наприклад:

- Бінарна класифікація. Якщо існує лише два можливих класи (наприклад, спам /не спам, здоровий/хворий, шахрайство/не шахрайство).
- Мультикласова класифікація. Коли існує більше двох класів (наприклад, спам/не спам).

Процес навчання алгоритму класифікації зазвичай включає кілька ключових кроків.

1. Збір даних. По-перше, потрібен набір даних з мітками. Цей набір даних складається з функцій (вхідних змінних) і міток (цільового виходу). Наприклад, під час виявлення спаму функції можуть включати слова, використані в електронному листі, а мітка вказуватиме, чи є електронний лист спамом чи ні.
2. Розробка функцій. Функції представляють характеристики даних, які використовуються для прогнозування. У деяких випадках необроблені дані потрібно перетворити на корисні функції за допомогою таких процесів, як нормалізація, кодування або зменшення розмірності.
3. Вибір моделі. Відповідний алгоритм класифікації вибирається на основі характеру даних і поточної проблеми. Різні алгоритми мають сильні та слабкі сторони залежно від розміру, складності та структури набору даних.
4. Навчання моделі. Модель навчається шляхом надання їй навчальних даних, де вона вивчає зв’язок між функціями та відповідними мітками.
5. Тестування та перевірка моделі. Після навчання модель оцінюється на тестовому наборі (дані, яких вона раніше не бачила), щоб виміряти її продуктивність. Такі методи, як перехресна перевірка, можна використовувати, щоб переконатися, що модель добре узагальнює нові дані.
6. Прогноз. Після навчання модель може передбачити мітку для нових, небачених прикладів. Модель призначає найімовірнішу мітку класу кожній точці вхідних даних на основі її вивчених меж прийняття рішень.

Існує багато різних типів алгоритмів класифікації, кожен із яких має свої сильні сторони та застосування [9]. Нижче наведено основні алгоритми, що будуть використовуватись у цій роботі:

1. Дерева рішень. Є одними з найбільш широко використовуваних алгоритмів машинного навчання, відомих своєю простотою, можливістю інтерпретації та ефективністю в задачах класифікації та регресії. У контексті кібербезпеки дерева рішень особливо корисні для виявлення зловмисного програмного забезпечення. зокрема клавіатурних шпигунів.

- Кореневий вузол. Початкова точка дерева, яке представляє весь набір даних.
- Внутрішні вузли. Ці вузли представляють точки прийняття рішень на основі функцій. Кожен внутрішній вузол задає запитання (наприклад, «Чи значення функції більше за X?»).
- Гілки. Зв’язки між вузлами, які представляють результати запитань, поставлених у внутрішніх вузлах.
- Листові вузли. Ці вузли представляють остаточну класифікацію або рішення. Для проблем класифікації кожен кінцевий вузол представляє мітку класу, наприклад «виявлено кейлогер» або «немає кейлогера».

Принцип роботи алгоритму.

- Поділ даних. Набір даних розділено на основі значень ознак, які найкраще поділяють дані на окремі класи. Для оцінки розподілів використовуються такі показники, як домішка Джіні або приріст інформації.

- Рекурсивне розбиття. Процес повторюється на кожному вузлі, далі розбиваючи дані, доки не буде виконано умову зупинки.

- Прогнозування. Нові дані класифікуються за правилами прийняття рішень від кореневого до кінцевого вузла, що дає остаточну класифікацію.

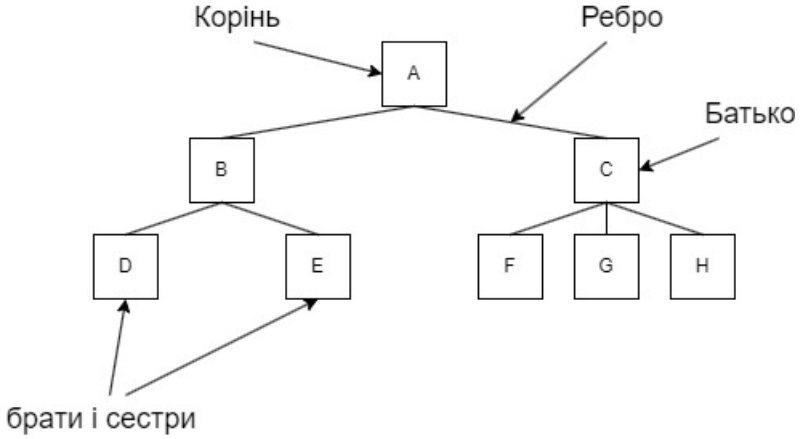

Рис 1. Структура алгоритму “дерево рішень”

2. Random Forest — це комплексний алгоритм машинного навчання, який поєднує кілька дерев рішень для створення більш надійної та точної моделі [3].

Принцип роботи алгоритму.

2.1. Вибірка даних. Random Forest створює кілька дерев рішень, і кожне дерево навчається на початковій вибірці навчальних даних. Початкова вибірка означає, що кожне дерево навчається на різній підмножині даних, де деякі точки даних можуть бути включені кілька разів, а інші можуть бути виключені взагалі. Це допомагає зменшити ризик переобладнання, яке може статися, якщо використовуватиметься лише одне дерево.

2.2. Випадковий вибір функції. Для кожного вузла прийняття рішень у дереві Random Forest вибирає випадкову підмножину функцій. Цей крок зменшує кореляцію між окремими деревами та гарантує, що кожне дерево відрізняється від інших.

2.3. Будівництво дерева. Кожне дерево рішень будується з використанням вибраної підмножини даних і функцій. Алгоритм розділяє вузли на основі таких показників, як домішка Джіні або приріст інформації, щоб створити правила прийняття рішень, які призводять до класифікації.

2.4. Голосування. Після створення всіх дерев вони використовуються для прогнозування нових даних. Наприклад, для завдань класифікації, таких як визначення того, чи є процес кейлогером, кожне дерево в лісі віддає «голос» за один із класів (наприклад, «Кейлогер» або «Не кейлогер»).Остаточний прогноз робиться на основі голосування більшістю: як результат вибирається клас, який отримує найбільшу кількість голосів з дерев рішень.

2.5. Вихід. Остаточний результат класифікації визначається шляхом узагальнення результатів усіх дерев у лісі. Оскільки передбачення базуються на кількох моделях, Random Forest зазвичай дає більш точні та надійні результати порівняно з одним деревом рішень.

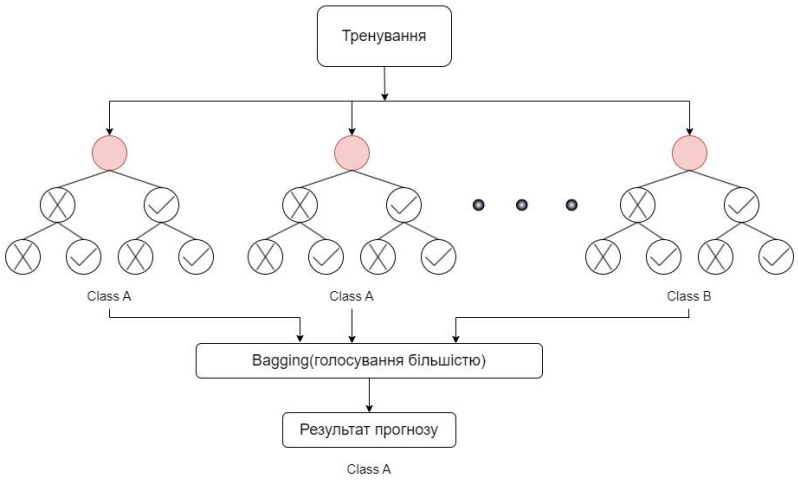

Рис 2. Принцип роботи алгоритму “Random Forest”

3. Support Vector Machines. Це алгоритм класифікації, метою якого є пошук оптимальної межі (також відомої як гіперплощина), яка найкраще розділяє точки даних із різних класів. Для виявлення кейлогерів ці класи зазвичай є «кейлогером» і «не-кейлогером», де метою є правильна класифікація процесу чи програми на основі його поведінкових особливостей.

Принцип роботи алгоритму.

- Навчання. Алгоритм аналізує позначений набір даних, щоб знайти гіперплощину, яка найкраще розділяє класи, максимізуючи маржу; класифікація: для нових, невідомих даних, SVM визначає, на якій стороні гіперплощини знаходиться точка даних, класифікуючи її відповідно;

- Класифікація. Для нових невидимих даних SVM визначає: з якого боку гіперплощини знаходиться точка даних і відповідно класифікуючи її.

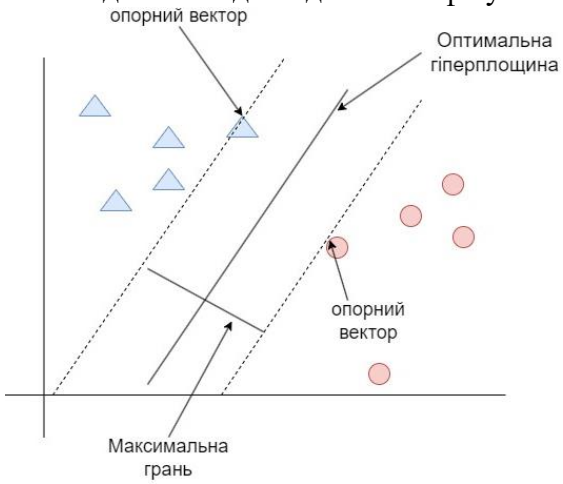

Рис 3. Принцип роботи алгоритму “Support vector machine”

Опис методу виявлення кейлогера за допомогою машинного навчання. Хибно позитивні та хибно негативні оцінки використовуються для розрахунку кількох корисних показників для оцінки моделей. Які метрики оцінки є найбільш значущими, залежить від конкретної моделі та конкретної задачі, вартості різних неправильних класифікацій та того, чи є набір даних збалансованим чи незбалансованим.

В нашому експерименті ми будемо використовувати дані з відкритого датасету ресурсу kaggle [12] та використовувати Scikit-learn бібліотеку машинного навчання з відкритим кодом, що містить у собі алгоритми машинного навчання, що були приведені у розділі 2.

Обраний датасет включає в себе як інформацію о використання мережі(як наприклад, колонки: “Total Backward Packets”, “Flow Bytes/s”, “Fwd Header Length”, “Average Packet Size”), так і роботу з API функціями(колонки “Source IP”, “min_seg_size_forward”, “act_data_pkt_fwd”, “Total Fwd Packets”). Саме ці ознаки будуть ключовими для виявлення наявності роботи кейлогера.

Блок-схема методу приведена на рисунку 4:

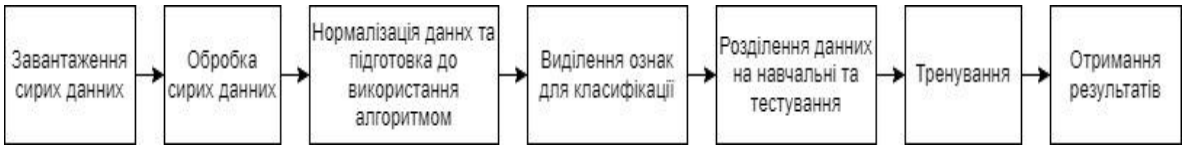

Рис 4. Схема експерименту

Перш за все датасет потрібно підготувати для використання перед тренуванням моделі. Для цього видалимо всі строки, що містят “nan”(ознаки, що не мають значень) та зайві признаки. Всі алгоритми, що використовуються в дослідженні є алгоритмами класифікації, а отже так званими “алгоритмами з вчителем”. “Вчителем” у датасеті є колонка “Class”, що вказує наявність(значення “Keylogger”), або відсутність роботи кейлогера(значення “Benign” у колонці). Розділивши датасет на два окремих: для навчання та тренування, ми також ділимо кожен з них ще на два: перший з датасет з всіма признаками без колонки “Class”, другий датасет окремо тільки колонку з “Class”, нашим вчителем.

На цьому етапі дані готові для використання алгоритмами машинного навчання, реалізацію яких ми взяли з відкритої бібліотеки Scikit-learn. Передавши датасеты з всіма признаками та “вчителем”, за допомогою бібліотеки ми отримали результати алгоритмів машинного навчання, а саме їх прогноз відносно кожного рядка у тренувальному датасеты. Саме ці вихідні бінарні показники стануть базою для визначення можливості використання машинного навчання у виявленні кейлогера.

Для визначення точності результату моделі ми будемо використовувати метрику Accuracy, що описується наступним рівнянням:

$$\text{Accuracy} = (TP + TN) \,/\, (TP + TN + FP + FN) * 100 \quad (1)$$

де

TP - true positive - правильно призначені позитивні класифікації

TN - true negative - правильно призначені негативні класифікації

FP - false positive - хибно призначені позитивні класифікації

FN - false negative - хибно призначені негативні класифікації

Accuracy - це частка всіх класифікацій, які були правильними як позитивними, так і негативними.

recision - це частка всіх позитивних класифікацій моделі, які насправді є позитивними. Математично описується наступним рівнянням:

$$\text{Precision} = (TP) \,/\, (TP + FP) \quad (2)$$

Recall - частка всіх фактичних позитивних результатів, які були правильно класифіковані як позитивні

$$\text{Recall} = (TP) \,/\, (TP + FN) \quad (3)$$

Результати дослідження та обговорення. В ході експерименту було використано наступні алгоритми машинного навчання: RandomForest, DecisionTree, SVM, KNN для для датасету з Kaggle [12] та отримали наступні результати:

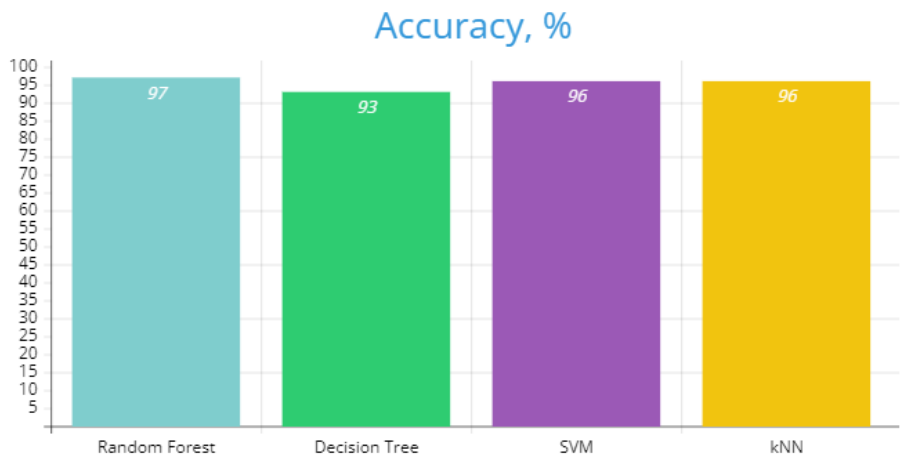

Рис 5. Результат визначення метрики точності алгоритмами машинного навчання

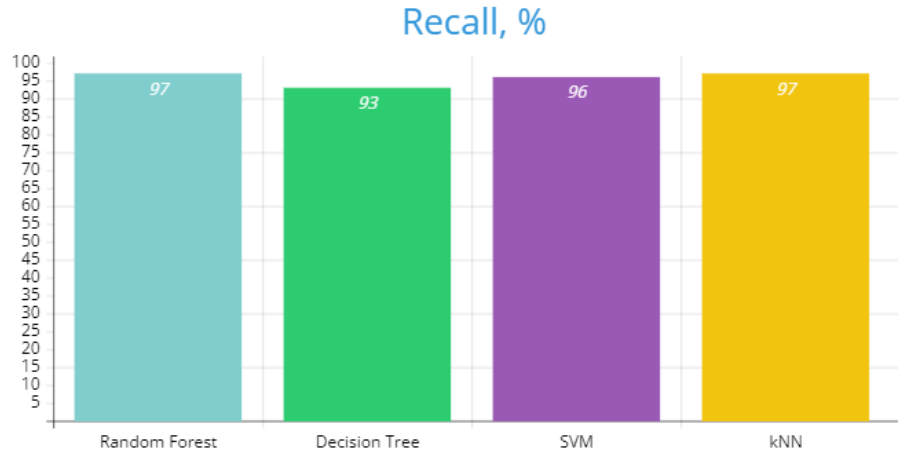

Рис 6. Результат визначення метрики повноти алгоритмами машинного навчання

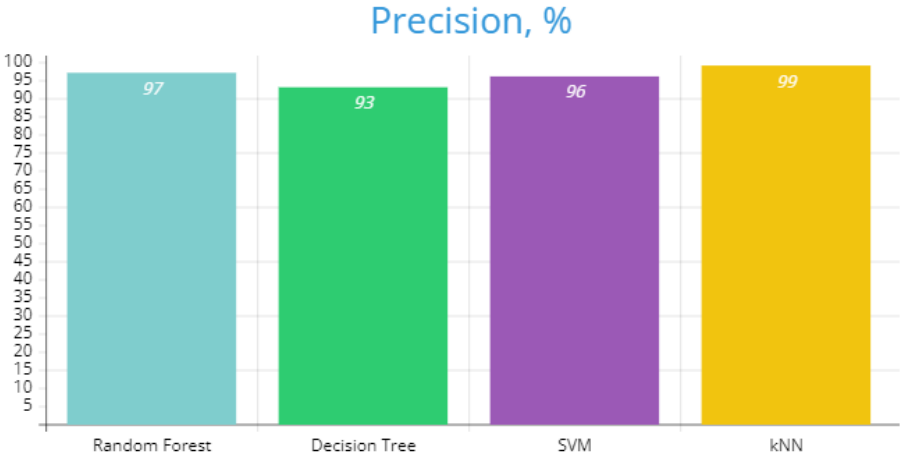

Рис 7. Результат визначення частки позитивних класифікацій алгоритмами машинного навчання

Результати дослідження зведено у підсумкову таблицю:

Таблиця 1.

Результати проведеного експерименту

Назва алгоритму	Accuracy	Recall	Precision
Random Forest	0,97	0,97	0,97
Decision Tree	0,93	0,93	0,93
SVM	0,96	0,96	0,96
KNN	0,96	0,97	0,99

Виходячи з отриманих результатів(див. табл. 1), ми отримали високі оцінки надійності виявлення кейлогерів практично для усіх обраних методів, в деяких випадках досягається 97% точність. RandomForest, KNN, SVM показали найкращі показники з точки зору точності, а DecisionTree – найгірші. Це показує, що дані алгоритми добре підходять для виявлення роботи кейлогера у системі і потребують більш глибокого вивчення.

Висновки. За допомогою підходу машинного навчання ми можемо виявляти різноманітні небезпечні дії, які виконують кейлогери в системі. У цьому дослідженні було доведено, що алгоритми машинного навчання можуть бути використовувані для виявлення роботи кейлогерів і шпигунського програмного забезпечення. Висновки грунтуються на численних показниках і надані на основі звіту про категоризацію для визначення продуктивності системи при виявленні шпигунського програмного забезпечення кейлогера. У запропонованому методі ми використали кілька алгоритмів машинного навчання для класифікації набору даних кейлогера: k-найближчих сусідів (KNN), RandomForest, Дерево ухвалення рішень (Decision Tree), Support Vector Machine класифікатори. RandomForest досяг найкращої точності 97%, тоді як Decision Tree має найнижчу точність 93%. Майбутня робота продовжуватиме вивчати проблему з використанням більш передових технологій у класифікації з використанням глибокого навчання, щоб підвищити безпеку користувача від роботи кейлогера.

Список літератури

1. Шибаєв Г., Гальчинський Л. Виявлення роботи кейлоггерів допомогою алгоритму дендритної клітинки з багаторазовою роздільною здатністю. Grail of Science, 2023. С. 173-176. URL: https://doi.org/10.36074/grail-of-science.
2. Moustafa N., Turnbull B., Choo K.-K.R. An ensemble intrusion detection technique based on proposed statistical flow features for protecting network traffic of internet of things, *IEEE Internet Things J*. URL: https://doi.org/10.1109/JIOT.2018.2871719
3. Jehad A., Rehanullah Kh., Nasir Ah., Imran M. Random forests and decision trees.*International Journal of Computer Science Issues* (IJCSI). 2012.
4. Chuvakin A. An overview of unix rootkits. iDEFENCE inc., 2003.
5. Kuncoro A. P., Kusuma B. A. Keylogger is a hacking technique that allows threatening information on mobile banking user. *International Conference on Information Technology Information System and Electrical Engineering (ICITISEE)*. 2018.P. 141, URL: https://doi.org/10.1109/ICITISEE.2018.8721028
6. Kruegel, C., Vigna, G., & Robertson, W. A multi-model approach to the detectionof web-based malware. Proceedings of the IEEE Symposium on Security and Privacy, 2009. https://doi.org/10.1016/j.comnet.2005.01.009
7. Raff E., Barker J., Sylvester J., Brandon R., Catanzaro B., Nicholas C. Malware detection by eating a whole EXE. *Proceedings of the AAAI Conference onArtificial Intelligence (AAAI)*. 2017. URL: https://doi.org/10.48550/arXiv.1710.09435
8. Sreenivas R.S., Anitha R. Detecting keyloggersbased on traffic analysis with periodic behaviour.Network Security, 2011. No.7. P.14-19. URL: https://doi.org/10.1016/S1353-4858(11)70076-9, 2011. https://doi.org/10.1016/S1353-4858(11)70076-9
9. Aslam M., Idrees R.N., Baig M.M., Arshad. M.A. Anti-hook shield against the software key loggers. *National Conference on Emerging Technologies.* 2004. P.189-191. URL: http://dx.doi.org/10.1109/DEST.2007.371990
10. Le D., Yue C., Smart T., Wang H. Detecting kernel level keyloggers through dynamic taint analysis. College of William & Mary, Department of ComputerScience, Williamsburg, VA, Tech. Rep. WM-CS-2008-05. 2008. URL: http://dx.doi.org/10.1088/1742-6596/2007/1/012005
11. Goring S.P., Rabaiotti J.R., Jones A.J. Anti keylogging measures for secure Internet login: An example of the law of unintended consequences. *Computers & Security*, V. 26. No 6. P. 421-426. URL: https://doi.org/10.1016/j.cose.2007.05.003, 2007.

https://doi.org/10.1016/j.cose.2007.05.003
12. Subhadeep Chakraborty. Detection of Keylogger using Machine Learning and Deep Learning. 2017. URL: https://doi.org/10.34740/kaggle/dsv/2625337

DETECTING THE WORK OF KEYLOGGERS IN THE OPERATING SYSTEM USING MACHINE LEARNING METHODS

H.D. Shybaiev[1], O.A. Halchynsky[2]

Ihor Sikorsky Kyiv Polytechnic Institute
37 Beresteyskyi Avenue, Kyiv, 03056, Ukraine
Emails: K233@ukr.net[1]; hleonid@gmail.com[2]

This article presents a methodology for detecting keyloggers using machine learning methods. Keyloggers are a common form of malware that captures keystrokes to steal sensitive data, posing a serious security threat to users and organizations. Traditional detection methods often rely on signature-based approaches that can be circumvented by advanced, polymorphic, or fileless keyloggers. This research uses machine learning (ML) to detect anomalous behavior and patterns indicative of keylogger activity.The authors explore various machine learning models, including decision trees, support vector machines (SVMs), random forests, and neural networks, to classify normal and malicious system behavior. Experiments were conducted to evaluate the precision, accuracy, recall, and F1 score of different ML models. The results show that ML-based methods can significantly improve the detection rate compared to traditional methods. The study also discusses potential challenges such as false positives, generalizing the model to new variants of keyloggers, and the need for ongoing training to adapt to evolving malware. The findings suggest that an ML-based detection system can offer a robust and adaptive solution to combat keyloggers, highlighting the importance of integrating artificial intelligence into cybersecurity infrastructure.

Keywords: machine learning, keylogger, artificial intelligence