

Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

Гловюка Володимира Сергійовича

на здобуття ступеня вищої освіти магістра

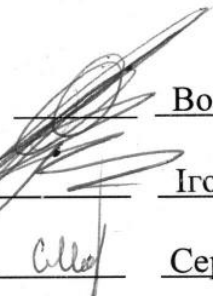
Нейромережевий метод пошуку вразливостей веб застосунків з використанням API ChatGPT.

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека та захист інформації

Освітня програма Кібербезпека та захист інформації

Шифр КРМКБЗІ. 2301135.23.01.06 ПЗ

Виконав студент 2 курсу група КБЗІм-23-1  Володимир ГЛОВІЮК

Керівник канд. техн. наук, доцент Ігор МУЛЯР

Нормоконтролер старший викладач Сергій МОСТОВИЙ

До захисту допускаю:

Завідувач кафедри кібербезпеки Юрій КЛЬОЦ

16 12 2024 р.

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет _____ Інформаційних технологій
Кафедра _____ Кібербезпеки
Рівень вищої освіти _____ Магістр
Галузь знань _____ 12 – Інформаційні технології
Спеціальність _____ 125 – Кібербезпека та захист інформації
Освітня програма _____ Кібербезпека та захист інформації

ЗАТВЕРДЖУЮ

Юрій КЛЬОЦ _____

_____ 2 _____ 09 _____ 2024 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гловюку Володимиру Сергійовичу

1 Тема роботи Нейромеревий метод пошуку вразливостей веб застосунків з використанням API ChatGPT

Керівник роботи канд.техн.наук, доцент Ігор МУЛЯР

Затверджено наказом ректора університету від 26 08 2024 № 60

2 Строк подання студентом кваліфікаційної роботи на кафедру 02.12.2024

3 Вихідні дані до роботи Нейромережі, веб застосунки, пошук вразливостей

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити) Вступ. Дослідження, аналіз теоретичної бази та актуальних досліджень у сфері пошуку вебвразливостей. Постановка задачі дослідження. Огляд сучасних мовних моделей штучного інтелекту та їх оцінка для задач кібербезпеки. Математична модель для аналізу та прогнозування загроз за допомогою мовних моделей. Методу пошуку та аналізу веб вразливостей з допомогою штучного інтелекту. Прикладне застосування розробленого методу в тестовому середовищі.

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

6 Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7 Дата видачі завдання 2 09 2024 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
Детальне ознайомлення та дослідження предметної галузі	16.09.2024	Виконано
Визначення змісту, структури кваліфікаційної роботи	23.09.2024	Виконано
Підготовка першого розділу кваліфікаційної роботи	28.09.2024	Виконано
Підготовка другого розділу кваліфікаційної роботи	11.10.2024	Виконано
Підготовка третього розділу кваліфікаційної роботи	21.10.2024	Виконано
Підготовка статті/тези за темою кваліфікаційної роботи	5.11.2024	Виконано
Підготовка четвертого розділу кваліфікаційної роботи	18.11.2024	Виконано
Підготовка та оформлення ілюстративного матеріалу	23.11.2024	Виконано
Оформлення кваліфікаційної роботи	23.11.2024	Виконано
Попередній захист кваліфікаційної роботи	27.11.2024	Виконано
Захист кваліфікаційної роботи на засіданні ЕК	17.12.2024	Виконано

Студент

Керівник кваліфікаційної роботи

Володимир ГЛОВІЮК

Ігор МУЛЯР

АНОТАЦІЯ

Тема кваліфікаційної роботи: Нейромережевий метод пошуку вразливостей веб застосунків з використанням API ChatGPT

Автор роботи: Гловюк Володимир Сергійович

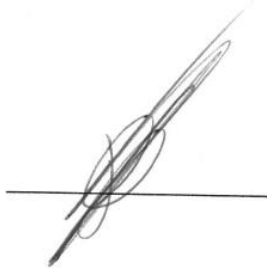
Керівник роботи: к.т.н., доц. Муляр Ігор Володимирович

Загальний обсяг роботи: 84 сторінки, 28 рисунків, 1 таблиця, 2 додатки, 62 посилання.

Ключові слова: вразливості веб застосунків, штучний інтелект, API ChatGPT.

Метою даного дослідження є створення підходу, який підвищує ефективність та точність процесів пентестингу шляхом автоматизації завдань, що зазвичай виконуються вручну.

Розроблений метод використовує можливості моделі GPT для генерації тестових запитів і аналізу відповідей серверів, що дозволяє ідентифікувати потенційні вразливості без необхідності поглибленого вивчення вихідного коду. Отримані результати підтверджують переваги застосування GPT-моделей для побудови складних тестових сценаріїв та аналізу реакцій вебзастосунків, сприяючи ефективному виявленню можливих загроз.



02.12.2024

ANNOTATION

Theme of qualification work: A Neural Network-Based Method for Detecting Web Application Vulnerabilities Using the ChatGPT API

Author of the work: Hloviuk Volodymyr Sergeevich

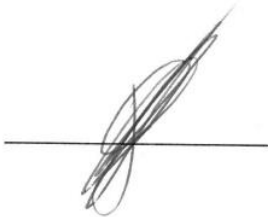
Mentor: Ph.D. Muliar Ihor Volodymyrovych

Total volume of work: 84 pages, 28 figures, 1 table, 2 appendices, 62 references.

Keywords: web application vulnerabilities, artificial intelligence, ChatGPT API.

The goal of this research is to develop an approach that enhances the efficiency and accuracy of penetration testing processes by automating tasks typically performed manually.

The proposed method leverages the capabilities of the GPT model to generate test queries and analyze server responses, enabling the identification of potential vulnerabilities without the need for in-depth examination of source code. The results obtained confirm the advantages of using GPT-based models to construct complex test scenarios and analyze web application responses, thereby facilitating the effective detection of possible threats.

A handwritten signature in black ink, consisting of several overlapping loops and a horizontal line extending to the left.

02.12.2024

ЗМІСТ

Скорочення та умовні позначки	8
Вступ	9
1 Аналіз підходів до пошуку вразливостей вебзастосунків	13
1.1 Огляд типів вразливостей вебзастосунків	13
1.2 Технології для виявлення вразливостей	24
1.3 Роль нейромереж у кібербезпеці	26
1.4 Постановка задачі	30
2 Дослідження моделей штучного інтелекту	33
2.1 Огляд сучасних мовних моделей штучного інтелекту	33
2.2 Оцінка можливостей ШІ для задач кібербезпеки	36
2.3 Використання LLM у системах кібербезпеки	40
2.4 Матиматична модель для аналізу та прогнозування загроз за допомогою мовних моделей	43
2.5 Етичні та правові аспекти застосування мовних моделей у кібербезпеці	45
2.6 Висновок	48
3 Розробка методу пошуку та аналізу веб вразливостей з допомогою штучного інтелекту	50
3.1 Нейромережевий метод пошуку вразливостей	50
3.2 Інтерактивна інженерія підказок для динамічного тестування безпеки	52
3.3 Побудова сценаріїв тестування та роль мовних моделей	57
3.4 Модель власного інструменту на основі API ChatGPT	59
3.5 Перспективи вдосконалення розробленого методу	62
3.6 Висновок	64
4 Тестування розробленого методу для пошуку вразливостей	66
4.1 Архітектура розробленого інструменту	66
4.2 Підготовка середовища тестування та інтеграція ШІ	70
4.3 Тестування запропонованих методів	73
4.4 Висновок	79

Висновки.....	81
Перелік джерел посилання.....	83
Додаток А Копії наукових публікацій.....	89
Додаток Б Презентація кваліфікаційної роботи.....	101

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

AI – Artificial Intelligence

LLM – Large Language Model

SQLi – SQL Injection

XSS – Cross-Site Scripting

CSRF – Cross-Site Request Forgery

HTTP – Hypertext Transfer Protocol

API – Application Programming Interface

CNN – Convolutional Neural Network

IDS – Intrusion Detection System

DAST – Dynamic Application Security Testing

SAST – Static Application Security Testing

RCE – Remote Code Execution

LFI – Local File Inclusion

RFI – Remote File Inclusion

XXE – XML External Entity

ВСТУП

У сучасному світі вебдодатки стали невід'ємною частиною багатьох сфер суспільного життя, включаючи бізнес, фінанси, державні послуги, освіту та розваги. Завдяки своїй доступності та гнучкості вони дозволяють користувачам виконувати численні завдання, починаючи від банківських операцій і закінчуючи управлінням корпоративними процесами, безпосередньо через браузер або мобільний додаток. Щоденно створюються тисячі нових вебдодатків, і їх стрімке зростання супроводжується аналогічним зростанням кіберзагроз. Зловмисники все частіше націлюються на вебдодатки з метою викрадення конфіденційної інформації, маніпуляції даними, розповсюдження шкідливого програмного забезпечення або виведення сервісів з ладу.

Згідно зі статистикою, представленою у звітах таких організацій, як Check Point, Cisco, IBM та Verizon, кількість кібератак на вебдодатки зросла на 50% за останні кілька років, а їх збитки щорічно сягають мільярдів доларів. За даними IBM X-Force Threat Intelligence Index, понад 30% атак у 2023 році були спрямовані саме на вебдодатки, що підкреслює їхню вразливість та популярність серед зловмисників. Це робить проблему захисту вебдодатків критично важливою, особливо для України, яка останнім часом стала мішенню численних кіберзагроз. У сучасних умовах гібридної війни, що включає кіберкомпонент, атаки на критично важливі об'єкти інфраструктури, зокрема вебдодатки, є невід'ємною частиною стратегії противника. Тому розробка ефективних методів захисту вебдодатків є не лише питанням технічної безпеки, але й важливим елементом національної кіберстратегії.

У сфері виявлення та запобігання кіберзагроз значний внесок зробили такі зарубіжні вчені, як Джин Ши (Jean Shi), автор робіт з виявлення SQL-ін'єкцій, та Джонатан Кліні (Jonathan Clunie), який досліджує застосування нейронних мереж у кібербезпеці. Серед українських науковців слід відзначити роботи професора Олександра Ковалю, який вивчає питання інтеграції

машинного навчання у сферу кіберзахисту, та доцента Андрія Мельника, який розробляє методи виявлення загроз у розподілених системах.

Сучасні методи атак, такі як SQL-ін'єкції, Cross-Site Scripting (XSS) та Cross-Site Request Forgery (CSRF), стають дедалі складнішими завдяки комбінуванню з соціальною інженерією, атаками на мережі та шифрування. Зловмисники постійно вдосконалюють свої підходи, використовуючи хмарні обчислення, інструменти автоматизації та новітні технології. Особливої уваги потребують вразливості, що виникають через інтеграцію вебдодатків з хмарними сервісами та мікросервісною архітектурою. Ці фактори створюють додаткові виклики для забезпечення безпеки, оскільки підвищують складність та масштаб атак.

Історичний розвиток методів кіберзахисту пройшов шлях від простих брандмауерів та ручного аудиту коду до автоматизованих систем, таких як антивіруси, системи виявлення та запобігання вторгнень (IDS/IPS), а також сканери вразливостей. Однак навіть ці методи не завжди можуть ефективно виявляти нові види вразливостей, особливо ті, що з'являються через помилки у новітньому програмному забезпеченні або нових архітектурних рішеннях. Сучасні технології машинного навчання (МЛ) відкрили нові можливості для кібербезпеки, дозволяючи створювати системи, які автоматично аналізують великі обсяги даних та виявляють патерни, характерні для атак.

Особливо перспективним є використання нейронних мереж, які забезпечують гнучкість та високу точність аналізу. Нейромережеві підходи дозволяють виявляти складні зв'язки та взаємозалежності у даних, що робить їх ідеальними для задач, пов'язаних із пошуком вразливостей у вебдодатках. Наприклад, конволюційні нейронні мережі можуть аналізувати структуровані дані, тоді як рекурентні мережі здатні працювати з послідовностями даних, такими як журнали запитів до серверів або логи додатків.

Великі мовні моделі (LLM), такі як ChatGPT, демонструють значний потенціал у галузі обробки текстових даних. Завдяки можливості працювати з текстовою інформацією у режимі реального часу, вони здатні аналізувати код,

текстові патерни SQL-запитів або потенційно небезпечні конструкції, що дозволяє ідентифікувати вразливості. Інтеграція таких моделей із системами кібербезпеки дозволяє підвищити швидкість реагування на нові загрози, автоматизувати процеси аналізу та значно знизити залежність від людського фактору.

Об'єктом дослідження є процес функціонування вебдодатків в умовах сучасних кіберзагроз.

Предметом дослідження є методи та інструменти виявлення вразливостей у вебдодатках.

Метою цієї роботи є розробка нейромережевого методу для пошуку вразливостей у вебдодатках із використанням API ChatGPT. Запропонований підхід автоматизує процеси аналізу коду, виявлення патернів і реагування на потенційні загрози.

Завдання дослідження:

- проаналізовано сучасні підходи до виявлення вразливостей у вебдодатках, включаючи переваги та недоліки традиційних методів;
- досліджено можливості застосування нейронних мереж та великих мовних моделей у сфері кібербезпеки;
- розроблено метод використання LLM для пошуку вразливостей в веб застосунках та інтеграції ChatGPT API з існуючими інструментами кібербезпеки;
- проведено тестування розробленого методу на реальних вебдодатках та оцінити його ефективність.

Наукова новизна роботи полягає у вдосконаленні методу пошуку та виявлення вразливостей з використанням API ChatGPT. Запропонований метод є інноваційним рішенням, яке інтегрує сучасні нейромережеві технології з інструментами та існуючими методами кібербезпеки, що дозволяє підвищити точність та ефективність виявлення.

Результати дослідження можуть бути використані для створення автоматизованих систем аналізу безпеки вебдодатків. Це дозволить знизити

витрати на ручний аудит коду, покращити ефективність виявлення загроз та забезпечити швидке реагування на нові типи атак.

Теоретичні та практичні результати роботи були представлені та обговорені на XX Міжнародній науково-практичній конференції «Військова освіта і наука: сьогодення та майбутнє», що відбулася у Київському національному університеті імені Тараса Шевченка. Опубліковано за матеріалами магістерської роботи 1 теза. Та опублікована стаття у фаховому журналі «SMART TECHNOLOGIES».

1 АНАЛІЗ ПІДХОДІВ ДО ПОШУКУ ВРАЗЛИВОСТЕЙ ВЕБДОДАТКІВ

1.1 Огляд типів вразливостей вебзастосунків

Вебдодатки стали ключовими компонентами сучасного цифрового світу, але їхня популярність і складність також роблять їх уразливими до атак. Основні типи вразливостей розглянемо детально [1].

SQL-ін'єкція (SQL Injection) - це метод атаки, що використовує некоректну обробку введених даних в SQL-запитах. Атака виникає, коли вебдодаток дозволяє користувачам вводити дані, які згодом інтегруються в SQL-запит без належної перевірки [2, 3]. Це дозволяє хакерам вбудовувати зловмисний SQL-код у запити, що може призвести до витоку даних, модифікації бази даних або навіть повного контролю над нею (рис. 1.1). SQL-ін'єкції стали основною загрозою для безпеки даних, оскільки дозволяють отримати доступ до критичних баз даних з конфіденційною інформацією, такою як паролі, дані користувачів або бізнес-дані [4].

Основні типи SQL-ін'єкцій:

- класична SQL-ін'єкція - використовується для виконання простих маніпуляцій з базою даних, як, наприклад, отримання додаткових записів;
- сліпа SQL-ін'єкція (Blind SQL Injection) - коли додаток не повертає результати SQL-запиту, але змінює свою поведінку в залежності від запиту;
- SQL-ін'єкція на основі помилок (Error-Based SQL Injection) - використовує повідомлення про помилки для витоку інформації про структуру бази даних;
- SQL-ін'єкція на основі об'єднання (Union-Based SQL Injection) - використовує оператор UNION, щоб поєднати два запити та витягти дані з іншої таблиці.

Відомі інциденти SQL-ін'єкцій: серед найвідоміших випадків SQL-ін'єкцій є атака на Sony у 2011 році, коли через вразливий вебдодаток хакери

отримали доступ до інформації про мільйони користувачів, включаючи їхні імена, адреси електронної пошти та, в деяких випадках, паролі. Цей інцидент продемонстрував небезпеку SQL-ін'єкцій та стимулював подальший розвиток методів захисту.

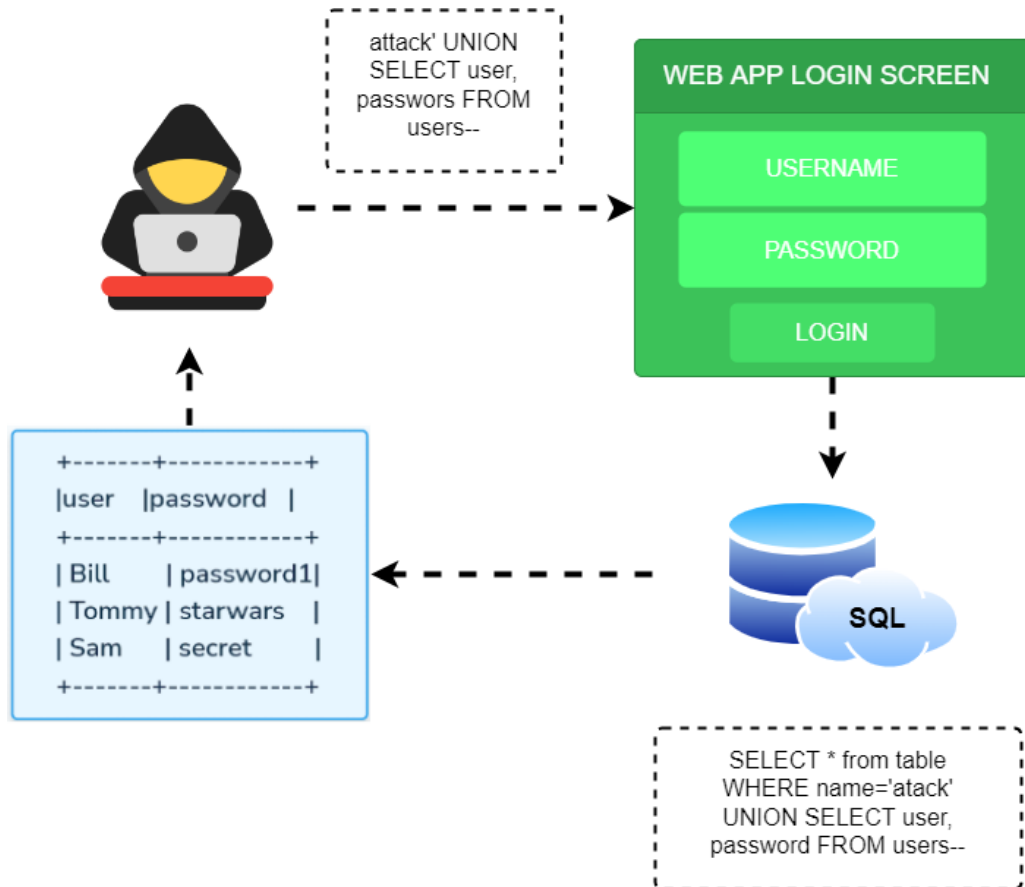


Рисунок 1.1 - Схема роботи SQL-ін'єкції на основі об'єднання

Методи захисту від SQL-ін'єкцій:

- валідація та санітизація введених даних, відфільтрування або заміна шкідливих символів, таких як `;` `--` або `UNION`, які можуть бути частиною SQL-ін'єкцій [5];
- параметризовані запити, використання параметризованих запитів у мовах програмування (наприклад, Prepared Statements у PHP або Python), де запит та дані розділяються, що не дозволяє маніпулювати SQL-кодом;

- використання ORM (Object-Relational Mapping), ORM-системи, як-от Django ORM чи SQLAlchemy, автоматизують обробку запитів, що мінімізує можливість введення користувачем шкідливого SQL-коду;

- використання Web Application Firewall (WAF), WAF фільтрує та блокує шкідливі SQL-запити, виконуючи моніторинг на рівні HTTP-запитів.

Атаки XSS (Cross-Site Scripting) дозволяють зловмисникам вставляти шкідливий код (зазвичай JavaScript) у вебсторінки, що відображаються у браузерах користувачів. Такі атаки виникають через недостатню перевірку або санітизацію введених даних у вебдодатках [6, 7]. Коли користувач заходить на скомпрометовану сторінку, шкідливий скрипт виконується на стороні клієнта, що призводить до небезпечних наслідків, зокрема:

- викрадення сесій, хакери можуть отримати доступ до ідентифікаторів сесій користувачів і використовувати їх для крадіжки облікових записів;

- крадіжка облікових даних, атака може використовуватись для збору логінів та паролів;

- перенаправлення на фішингові сайти, користувачів можна непомітно перенаправляти на шкідливі ресурси для крадіжки даних.

Існує три основні типи XSS-атак, кожен з яких має особливі риси та способи експлуатації [8].

Збережена XSS (Stored XSS) - шкідливий код зберігається безпосередньо у базі даних вебдодатка та активується при завантаженні сторінки. Збережена XSS особливо небезпечна, оскільки впливає на всіх користувачів, які відкривають скомпрометовану сторінку (рис. 1.2)..

Відображена XSS (Reflected XSS) - цей тип атак відображається в браузері одразу після введення користувачем певних даних, що використовуються для побудови відповіді. Наприклад, якщо шкідливий скрипт додається через URL, він може виконуватись відразу після завантаження сторінки.

DOM-based XSS - відбувається через зміну структури Document Object Model (DOM) на клієнтській стороні. У цьому випадку атака використовує

JavaScript-код, який динамічно змінює сторінку без взаємодії з сервером. DOM-based XSS важко виявити, оскільки вона не залежить від серверної обробки [9].

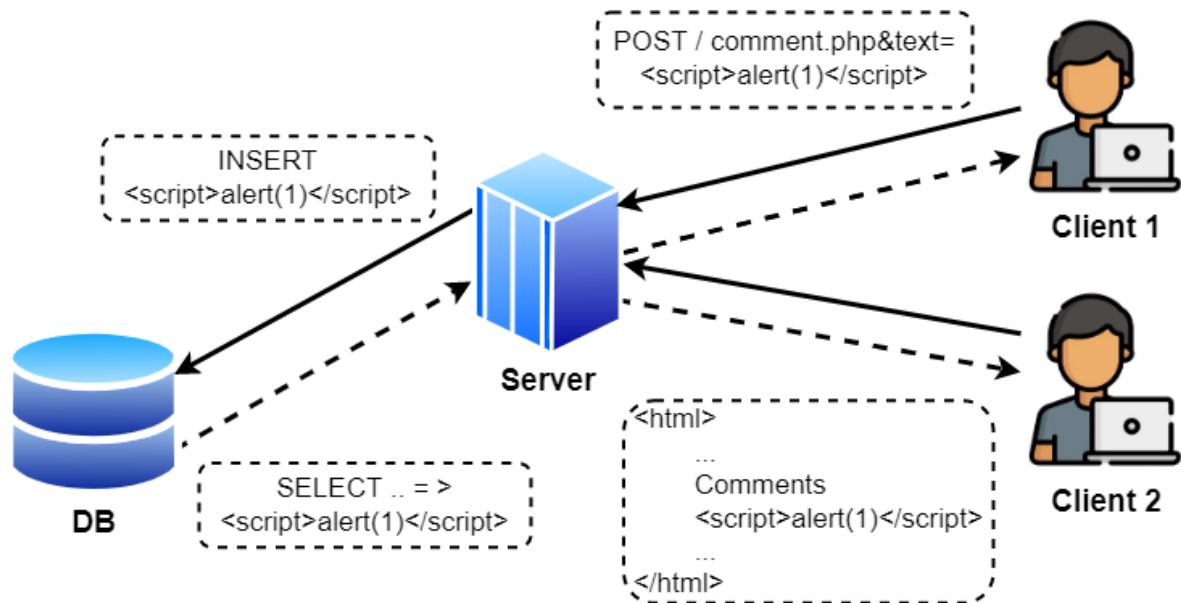


Рисунок 1.2 - Схема роботи XSS

Одним із найвідоміших прикладів XSS-атаки став інцидент на платформі MySpace у 2005 році. Хакер Семмі Камкар створив XSS-черв'яка, який автоматично додавав користувачів до його списку друзів після відвідування зараженого профілю. Атака поширилася неконтрольовано і MySpace довелося провести повне відновлення системи для її зупинки.

Захист від XSS-атак потребує належного контролю даних та налаштувань безпеки. Основні методи включають:

- санітизація та валідація введених даних, перевірка та очищення будь-яких даних, що надходять від користувачів, перед відображенням у браузері. Це дозволяє уникнути введення шкідливих символів чи скриптів;
- ескейпінг HTML-коду, метод, який перетворює спеціальні символи на HTML-еквіваленти, щоб запобігти виконанню скриптів;
- використання Content Security Policy (CSP), CSP обмежує виконання небажаних скриптів та дозволяє виконувати лише скрипти з перевірених джерел. Це створює додатковий рівень захисту, запобігаючи виконанню будь-якого коду, який не є частиною офіційного ресурсу.

CSRF (Cross-Site Request Forgery) - це вид атак, коли зловмисник змушує користувача виконати несанкціоновані дії на вебсайті, де той вже авторизований. Атаки CSRF використовують механізм авторизації браузера, щоб здійснити дії, які сприймаються вебсайтом як виконані самим користувачем. Наприклад, це може включати такі дії, як зміна налаштувань облікового запису, виконання платіжних транзакцій або видалення даних [10].

На рисунку 1.3 зображено поетапний принцип роботи CSRF.

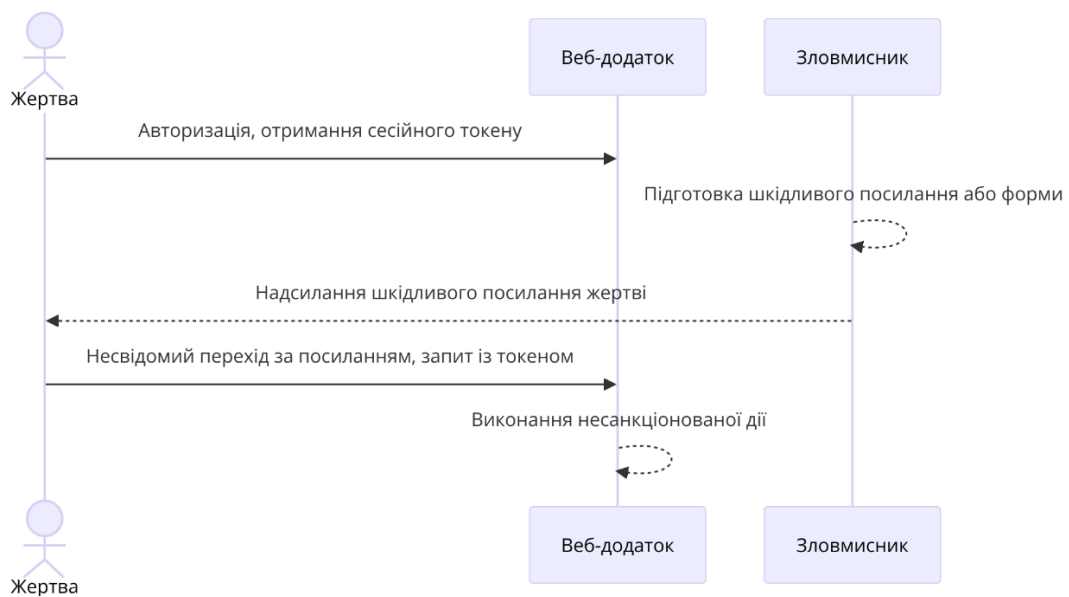


Рисунок 1.3 - Схема роботи CSRF

- спочатку користувач авторизується на певному вебсайті. Після успішного входу сервер генерує спеціальний аутентифікаційний токен (наприклад, сесійний cookie), який зберігається в браузері користувача. Це означає, що будь-які подальші запити, які надсилаються на цей сайт, будуть автоматично містити цей токен для підтвердження авторизації користувача;

- зловмисник створює шкідливе посилання або форму, яка виглядає як звичайний запит до цільового сайту. Наприклад, якщо вебсайт дозволяє користувачам надсилати запити для зміни пароля, зловмисник може згенерувати форму, яка відправляє запит на зміну пароля з новим значенням, яке він сам визначив;

– зловмисник пересилає шкідливе посилання або форму жертві, намагаючись змусити її виконати певну дію. Це може бути досягнуто шляхом надсилання фальшивого посилання електронною поштою, через повідомлення у соцмережах або вставкою шкідливого коду на інший вебсайт. Коли жертва натискає на посилання або взаємодіє з фальшивою формою, запит автоматично надсилається на цільовий сайт;

– коли запит надходить на цільовий сайт, браузер автоматично додає аутентифікаційний токен користувача, оскільки він вважає цей запит таким, що походить від авторизованого користувача. Оскільки сайт не відрізняє справжній запит від підробленого, він обробляє запит як легітимний і виконує дію, зазначену зловмисником (наприклад, змінює пароль або проводить фінансову транзакцію) [11, 12].

Методи захисту від CSRF:

– використання CSRF-токенів - генерування унікального CSRF-токену для кожного користувацького сеансу або транзакції. Цей токен повинен надсилатися з кожним запитом та перевірятись на сервері;

– перевірка реферера або походження запиту - контроль поля реферера (Referer) чи походження (Origin) в HTTP-запиті, щоб переконатися, що запит надходить із довіреного джерела;

– впровадження двофакторної автентифікації (2FA) - для операцій, що вимагають високого рівня безпеки, варто використовувати 2FA, щоб додатково підтвердити дії користувача [13,14].

Сучасні вебдодатки значно залежать від сторонніх бібліотек та фреймворків, що зменшує час розробки, але водночас несе ризики через потенційні вразливості в цих компонентах. Застарілі або невиправлені компоненти можуть мати вразливості, які хакери використовують для виконання атак. Вразливість у Log4j 2021 року стала яскравим прикладом таких ризиків. Цей інцидент продемонстрував, що навіть широко використовувані та перевірені компоненти можуть містити критичні вразливості.

Є декілька методів захисту від вразливих компонентів. Перший - регулярні оновлення та патчі. Постійне оновлення сторонніх бібліотек та компонентів забезпечує їх захищеність від нових вразливостей. Другий - використання перевірених репозиторіїв. Завантаження компонентів тільки з надійних джерел із перевіреною історією безпеки. Та третій - постійний моніторинг вразливостей. Використання систем моніторингу, які відстежують наявні вразливості в компонентах та надають рекомендації щодо їх усунення.

LFI (Local File Inclusion) та RFI (Remote File Inclusion) - це вразливості вебдодатків, що виникають через недостатній контроль за включенням файлів на сервері. Обидва типи атак дозволяють зловмиснику отримати доступ до файлів на сервері або навіть виконувати довільний код, що може призвести до компрометації системи. Вразливості LFI та RFI здебільшого трапляються, коли вебдодаток надає користувачам можливість вказувати файл для завантаження або включення без належної перевірки введених даних [15-17].

LFI та RFI мають схожий механізм атаки - вони використовують введені користувачем дані для визначення файлів, які має обробляти сервер. Обидві вразливості можуть призвести до витоку конфіденційних даних або виконання шкідливого коду, якщо зловмисник зможе завантажити або включити файл із вбудованим кодом [18].

Відмінності між LFI та RFI:

- LFI (Local File Inclusion) зосереджується на доступі до локальних файлів, які вже знаходяться на сервері. Зловмисник може скористатися LFI, щоб отримати доступ до конфіденційних системних файлів, як-от `/etc/passwd` або інших локальних файлів з критичними даними. Використовуючи відносні шляхи (наприклад, `../etc/passwd`), він може піднятися в файлової системі та переглядати вміст важливих файлів [17];

- RFI (Remote File Inclusion) дозволяє зловмиснику включити файл із зовнішнього джерела. Це відбувається, коли додаток приймає URL як шлях до файлу, і, таким чином, дозволяє зловмиснику завантажити та виконати шкідливий код із віддаленого сервера. Це особливо небезпечно, оскільки RFI

надає можливість віддаленого виконання команд і навіть повного контролю над сервером.

Атаки LFI та RFI можуть призвести до таких серйозних наслідків. Отримання доступу до конфіденційних файлів [19]. Обидва типи атак можуть використовуватися для доступу до важливих файлів, що містять паролі, ключі доступу, конфігураційні дані. Виконання довільного коду. Якщо включений файл містить виконуваний код, зловмисник може використати це для виконання шкідливих команд на сервері. Підвищення привілеїв. Доступ до конфіденційної інформації може дозволити зловмиснику отримати більше привілеїв на сервері.

Для захисту від LFI та RFI необхідно використовувати комплексні заходи безпеки. Санітизація та валідація введених даних - забезпечення ретельної перевірки введених даних, щоб уникнути можливості використання відносних шляхів (../) або URL для завантаження віддалених файлів. Використання абсолютних шляхів - замість приймання шляхів із користувацьких запитів, встановіть жорстко закріплені абсолютні шляхи до файлів, які можуть бути включені. Обмеження прав доступу до файлів - встановіть відповідні дозволи на важливі файли та каталоги, щоб уникнути несанкціонованого доступу до них. Вимкнення віддаленого включення файлів - якщо це можливо, вимкніть опцію віддаленого включення файлів у конфігураціях серверів (наприклад, `allow_url_include=Off` у PHP). Використання Web Application Firewall (WAF) - WAF може виявляти та блокувати запити з підозрілими шляхами або URL, що можуть вказувати на спроби LFI або RFI.

XXE (XML External Entity) - це вразливість, яка виникає, коли додаток обробляє вхідні XML-документи без обмеження на включення зовнішніх сутностей. У таких атаках зловмисник використовує особливості XML для отримання доступу до конфіденційних даних, локальних файлів або виконання віддалених запитів на сервері. Вразливість XXE стає особливо небезпечною, якщо XML-процесор додатка дозволяє вказувати зовнішні сутності (спеціальні

інструкції для обробки зовнішніх даних), що може привести до витоку даних або навіть віддаленого виконання коду [20-23].

Основний механізм XXE-атаки полягає у вбудовуванні зловмисником зовнішніх сутностей у XML-файл, який потім обробляється на сервері. Наприклад, зловмисник може вставити посилання на локальні файли сервера, такі як `/etc/passwd`, або використати зовнішні посилання для виконання віддалених запитів, що допоможе отримати важливу інформацію про внутрішню мережу [22].

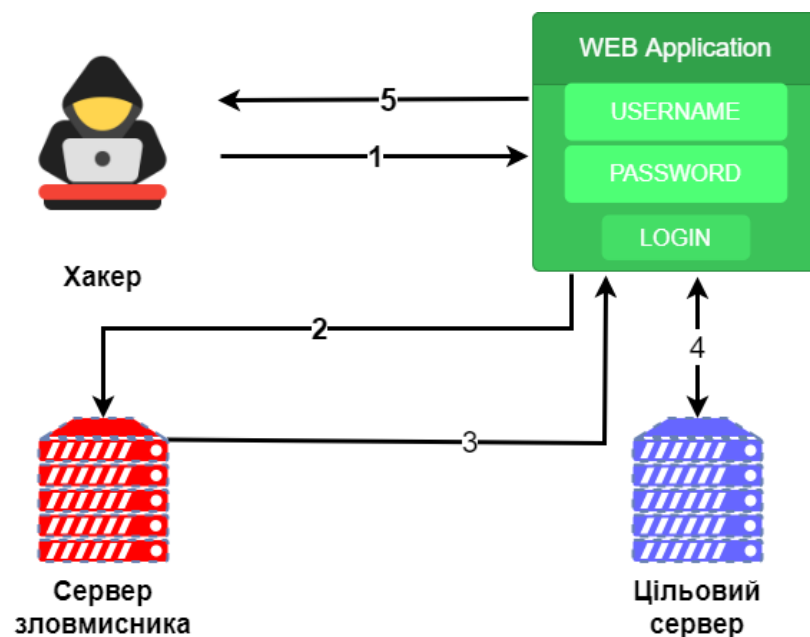


Рисунок 1.4 - Схема роботи XXE

На зображенні 1.4 зображений процес атаки XXE, де використовується слабка конфігурація XML-парсера для доступу до конфіденційних даних. Ось покроковий опис:

- хакер визначає додаток із неправильно налаштованим XML-парсером і надсилає XML-запит;
- вебдодаток отримує цей запит;
- XML-процесор отримує шкідливу зовнішню сутність у межах декларації типу документа (DTD);

- XML-процесор перевіряє DTD і обробляє шкідливу зовнішню сутність.

- конфіденційні дані передаються хакеру.

Вразливість XXE може призвести до серйозних наслідків:

- витік конфіденційних даних, зловмисник може отримати доступ до важливих файлів, зокрема конфігураційних, які містять паролі та іншу конфіденційну інформацію;

- виконання віддалених запитів (SSRF), завдяки XXE зловмисник може відправити запити до внутрішніх ресурсів компанії, що дозволяє зібрати інформацію про внутрішню інфраструктуру;

- виконання довільного коду, якщо вразливість дозволяє завантажувати зовнішні файли, зловмисник може використати її для завантаження та виконання шкідливого коду.

Запобігання XXE-атакам вимагає налаштування XML-парсерів для заборони обробки зовнішніх сутностей. Вимкнення підтримки зовнішніх DTD (Document Type Definition) або використання бібліотек із безпечною обробкою XML допомагає знизити ризики. Правильна конфігурація парсерів і уникнення обробки XML-даних із неперевірених джерел є ключовими методами захисту від XXE [21].

RCE (Remote Code Execution) - це серйозна вразливість, яка дозволяє зловмисникам виконувати довільний код на сервері або пристрої з віддаленого місця. Ця вразливість вважається однією з найнебезпечніших, оскільки надає зловмиснику можливість виконувати шкідливі команди на цільовій системі, що може спричинити її повний компроміс, крадіжку конфіденційної інформації або значні перебої у роботі [24].

RCE-атаки зазвичай виникають через недостатньо надійну обробку введених даних, що надходять до додатків або систем [25]. Вразливість може бути викликана ін'єкцією шкідливого коду у вебдодатки, API або інші програмні компоненти, які не виконують належну перевірку введення. Наприклад, якщо програма дозволяє користувачам передавати команди для

виконання операційною системою без достатньої фільтрації, зловмисник може додати до запиту шкідливий код, що призведе до його виконання на сервері.

Особливо небезпечною є ця вразливість у випадках, коли система має високі привілеї, оскільки це може дати зловмиснику необмежений доступ до ресурсів і критичної інформації [25, 26].

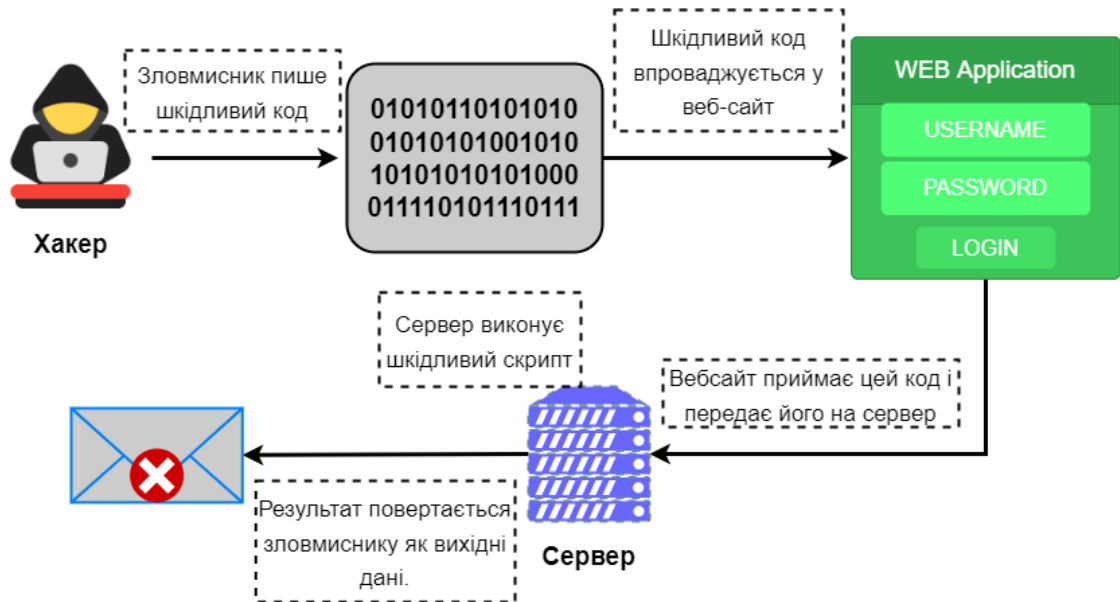


Рисунок 1.5 - Схема роботи RCE

RCE-атаки можуть призвести до різних серйозних наслідків, таких як:

- отримання повного контролю над системою, тобто зловмисник може керувати системою, змінювати налаштування, видаляти файли, встановлювати шкідливе програмне забезпечення тощо;
- витік конфіденційних даних, оскільки зловмисник має доступ до системи, він може копіювати, змінювати або видаляти дані, включаючи паролі, фінансову інформацію та інші важливі файли;
- зараження мережі, зловмисник може використовувати вразливу систему для розповсюдження шкідливих програм на інші пристрої у мережі, збільшуючи масштаб атаки.

Для захисту системи від RCE важливо вживати комплексних заходів. По-перше, необхідна ретельна валідація та санітизація введених даних, щоб уникнути можливості виконання довільного коду, який зловмисник може спробувати передати через зовнішні запити. По-друге, слід застосовувати

принцип мінімально необхідних прав, тобто обмежувати доступ програм до системних команд і ресурсів. Це дозволяє мінімізувати ризик RCE, навіть якщо в системі існує певна вразливість [27]. Крім того, потрібно підтримувати постійне оновлення програмного забезпечення, встановлюючи останні патчі для усунення можливих вразливостей, які можуть стати мішенню для RCE [28].

Розуміння основних типів вразливостей вебзастосунків є необхідною умовою для створення безпечних систем. Кожен тип атаки має свої особливості, але всі вони демонструють важливість застосування найкращих практик безпеки. Інтеграція сучасних інструментів, таких як великі мовні моделі та нейронні мережі, відкриває нові можливості для підвищення ефективності захисту та своєчасного виявлення загроз.

1.2 Технології для виявлення вразливостей

Для ефективного захисту вебзастосунків від атак важливим є регулярне та всебічне виявлення вразливостей. Сучасні технології забезпечують комбінований підхід, який включає використання як традиційних методів аналізу, так і новітніх інструментів, що базуються на принципах автоматизації та машинного навчання. Цей підхід дає змогу не лише виявляти відомі загрози, але й вчасно реагувати на нові, складніші виклики [29, 30].

Одним із найпоширеніших методів виявлення вразливостей є сканування вебзастосунків. Воно охоплює як статичний аналіз коду (Static Application Security Testing, SAST), так і динамічний аналіз (Dynamic Application Security Testing, DAST). Статичний аналіз дозволяє виявляти потенційні загрози ще на етапі розробки, не потребуючи виконання застосунку. Інструменти для SAST, такі як SonarQube, Fortify та Checkmarx, аналізують код на рівні синтаксису й семантики, що дозволяє ідентифікувати небезпечні ділянки, включаючи можливі SQL-ін'єкції чи порушення правил автентифікації. Основною перевагою цього методу є можливість виправлення вразливостей ще до виходу

продукту в експлуатацію. Водночас статичний аналіз має обмеження у виявленні проблем, що виникають у процесі взаємодії між компонентами або через вплив зовнішніх факторів [30].

Динамічний аналіз коду, з іншого боку, здійснюється на працюючих застосунках. Інструменти, такі як OWASP ZAP і Acunetix, створюють симуляцію реального середовища й надсилають спеціально сформовані запити для виявлення некоректної поведінки системи. Це дозволяє знайти вразливості, які можуть бути проігноровані під час статичного аналізу, зокрема у випадках SQL-ін'єкцій, XSS або інших атак, пов'язаних із неправильним обробленням введених даних [31].

Крім сканування, важливу роль у забезпеченні безпеки вебзастосунків відіграють системи виявлення вторгнень (Intrusion Detection Systems, IDS). Вони постійно моніторять мережеву активність, аналізуючи трафік для виявлення підозрілих дій. Сучасні IDS можуть базуватися як на підписах, так і на аналізі аномалій. Системи на основі підписів, наприклад Snort чи Suricata, співставляють дані з базами шаблонів відомих атак і миттєво реагують на виявлення збігів. Проте цей підхід має обмеження щодо виявлення нових загроз, які не мають заздалегідь визначених підписів. У цьому контексті системи, засновані на аналізі аномалій, демонструють більшу гнучкість. Вони створюють модель «нормальної» поведінки й виділяють підозрілі відхилення, що може свідчити про атаку. Недоліком цього підходу є ризик хибнопозитивних спрацьовувань, особливо у складних і змінних середовищах [32].

Автоматизовані інструменти, такі як Burp Suite, також відіграють важливу роль у виявленні вразливостей. Burp Suite дозволяє не лише проводити сканування, але й вручну аналізувати трафік між клієнтом і сервером, редагувати запити, а також знаходити вразливості, які можуть бути непомітними для автоматичних сканерів. OWASP ZAP, розроблений проєктом OWASP, є ще одним потужним рішенням для тестування безпеки, яке охоплює широкий спектр атак, включаючи SQL-ін'єкції та CSRF. Acunetix, у свою чергу,

пропонує інструменти для автоматизованого аналізу, здатні знаходити як поширені загрози, так і нові, які виникають через специфічні архітектурні рішення застосунку [33].

З розвитком технологій виявлення вразливостей значну роль починають відігравати методи машинного навчання. Завдяки здатності аналізувати великі обсяги даних ці технології забезпечують виявлення аномалій та відхилень у поведінці системи, які можуть сигналізувати про потенційну атаку. Наприклад, алгоритми кластеризації дозволяють групувати схожі за характеристиками атаки, а класифікатори визначають, до якої категорії належить загроза, що значно прискорює реагування.

Особливе місце серед новітніх підходів займають великі мовні моделі (LLM), такі як ChatGPT. Ці моделі можуть аналізувати текстові дані, включаючи лог-файли або код, з метою виявлення потенційно небезпечних сегментів або аномалій. Використання LLM дає змогу автоматизувати процес аналізу великих обсягів інформації та значно зменшити кількість хибнопозитивних результатів. Наприклад, великі мовні моделі здатні виявляти шаблони, характерні для SQL-ін'єкцій, або аналізувати параметри HTTP-запитів, щоб знайти можливі місця для XSS-атак [34].

Таким чином, сучасні технології виявлення вразливостей поєднують переваги традиційних методів із можливостями новітніх інструментів, що базуються на принципах автоматизації, аналізу аномалій і машинного навчання. Це дозволяє створювати більш ефективні стратегії захисту вебзастосунків, реагуючи на нові виклики кібербезпеки.

1.3 Роль нейромереж у кібербезпеці

Машинне навчання (МЛ) є одним із найперспективніших напрямів у сфері кібербезпеки, оскільки дозволяє обробляти великі обсяги даних і виявляти приховані закономірності. Завдяки цьому алгоритми МЛ сприяють

ефективному виявленню вразливостей у вебзастосунках, а також допомагають запобігати кібератакам і мінімізувати їхні наслідки. Такий підхід дозволяє компаніям і організаціям адаптуватися до нових типів загроз і швидко реагувати на складні виклики безпеки, які стають дедалі поширенішими в сучасному цифровому середовищі.

Одним із найбільш перспективних напрямів використання МЛ у кібербезпеці є інтеграція великих мовних моделей (LLM), таких як ChatGPT, для виявлення вразливостей у вебзастосунках. Ці моделі здатні аналізувати текстові дані, зокрема код і журнали, для виявлення шаблонів, які вказують на можливі загрози, наприклад SQL-ін'єкції, XSS або CSRF. Великі мовні моделі ефективно працюють із різноманітними форматами даних і можуть ідентифікувати навіть складні або приховані загрози, що дозволяє суттєво знизити ризики експлуатації вразливостей.

На ринку вже доступні численні інструменти, що застосовують алгоритми МЛ для виявлення вразливостей і запобігання атакам. Наприклад, Tenable.io є одним із провідних рішень для аналізу безпеки вебзастосунків. Інструмент використовує алгоритми машинного навчання для ідентифікації вразливостей, таких як SQL-ін'єкції чи XSS, а також пропонує функцію прогнозного аналізу. Ця функція дозволяє оцінити ймовірність експлуатації певних вразливостей, що дає змогу командам безпеки зосередитися на найбільш критичних загрозах [35].

Іншим прикладом є Darktrace, який застосовує глибоке навчання для аналізу мережевої активності. Darktrace будує моделі нормальної поведінки мережі й ідентифікує аномалії, які можуть сигналізувати про потенційні атаки. Наприклад, у 2020 році цей інструмент успішно захистив велику енергетичну компанію, виявивши цільову атаку на ранньому етапі, що дозволило уникнути серйозних наслідків [36].

Imperva пропонує рішення для захисту вебзастосунків, орієнтоване на аналіз мережевого трафіку. Її алгоритми МЛ можуть розпізнавати підозрілу активність, пов'язану з DDoS-атаками чи спробами експлуатації вразливостей.

Зокрема, під час великої DDoS-атаки на глобальний онлайн-магазин Imperva змогла автоматично блокувати шкідливий трафік, зберігаючи доступність ресурсу для легітимних користувачів.

Інструмент Splunk, що спеціалізується на аналізі лог-файлів, активно використовує методи МЛ для виявлення аномалій і кіберзагроз. Його розширення Splunk Enterprise Security дозволяє автоматизувати процеси аналізу та спрощує виявлення складних загроз, таких як спроби обійти механізми автентифікації або змінити конфігурації системи [37, 38].

Застосування МЛ також демонструє ефективність у поєднанні з традиційними системами виявлення вторгнень, такими як Snort. Цей інструмент використовує сигнатури для розпізнавання відомих атак, а також алгоритми машинного навчання для виявлення аномалій у мережевому трафіку, які можуть вказувати на нові, раніше невідомі загрози.

Ефективність МЛ у виявленні вразливостей підтверджена на практиці. Наприклад, Tenable.io використовувався для прогнозування атак на фінансові установи, що дозволило завчасно усунути кілька критичних вразливостей і уникнути значних фінансових збитків. У свою чергу, Darktrace довів свою корисність у відбитті цільових атак, зокрема в енергетичному секторі, де раннє виявлення аномалій дало змогу захистити критичні дані. А Imperva успішно впоралася із захистом онлайн-магазинів під час DDoS-атаки, запобігши перериванню доступу для клієнтів [39].

Попри численні переваги, використання машинного навчання у виявленні вразливостей пов'язане з певними викликами. Основними проблемами залишаються потреба у великих обсягах якісних даних для навчання моделей, а також значні витрати часу та ресурсів на налаштування і підтримку таких систем. Крім того, алгоритми МЛ можуть давати хибнопозитивні результати, що створює додаткові труднощі для команд безпеки [40, 41].

Проте, незважаючи на ці виклики, розвиток великих мовних моделей та алгоритмів глибокого навчання відкриває нові горизонти у сфері кібербезпеки. Завдяки постійному вдосконаленню цих технологій, автоматизація процесів

виявлення вразливостей стає дедалі більш ефективною, що дозволяє підвищити рівень безпеки вебзастосунків та зменшити ризики, пов'язані з кібератаками.

Але в той самий час компанія OpenAI розповіла, що припинила понад 20 іноземних “операцій впливу”, в рамках яких ChatGPT використовувався для налагодження та розробки шкідливого програмного забезпечення, поширення дезінформації, ухилення від виявлення та проведення фішингових атак.

У звіті представники OpenAI підтвердили факт зловживання можливостями ChatGPT і описали конкретні випадки, коли китайські та іранські хакери зловживали чат-ботом для підвищення ефективності своїх операцій.

OpenAI повідомляв, що SweetSpecter атакує свої цілі, розсилаючи фішингові листи із шкідливими ZIP-архівами, замаскованими під запити до служби підтримки. Після відкриття такого вкладення запускається ланцюжок зараження, у результаті в систему жертви проникає SugarGh0st RAT.

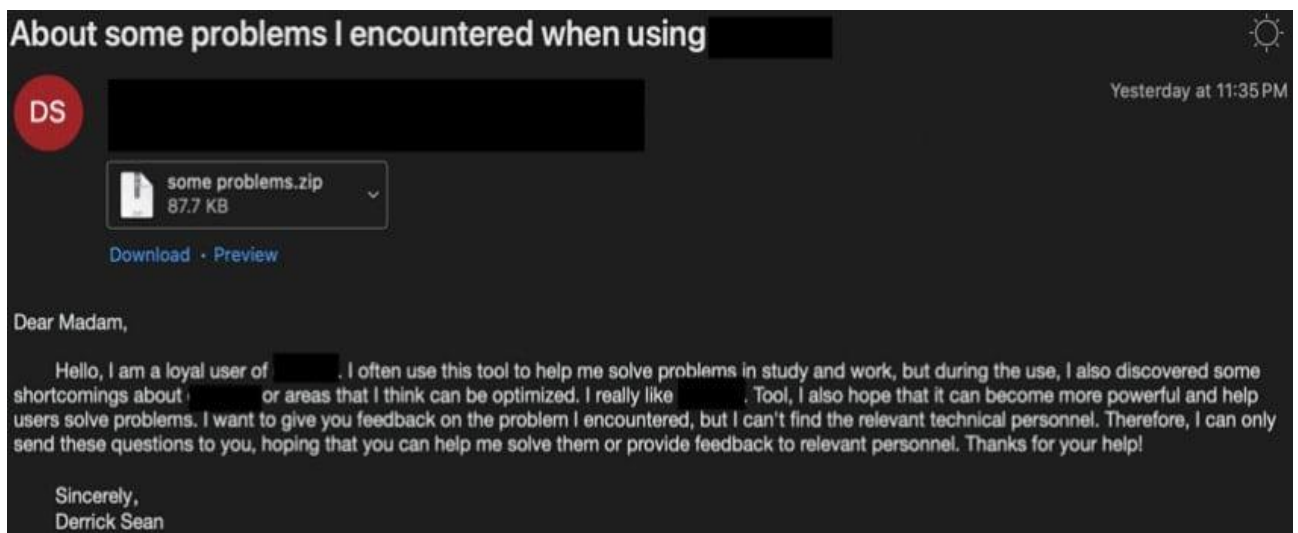


Рисунок 1.6 - Фішинговий лист SweetSpecter

Учасники SweetSpecter зверталися до чат-боту з наступними запитам:

- питання про вразливість у різних додатках;
- пошук певних версій Log4j, уразливих для критичної RCE-проблеми Log4Shell;
- просили список популярних CMS, які використовуються за кордоном;

- запитували інформацію щодо конкретних ідентифікаторів CVE;
- питали, як створюються сканери, що працюють по всьому інтернету;
- просили допомоги в експлуатації інфраструктури, що належить неназваному виробнику автомобілів.

Інша ситуація була пов'язаний з іранською хак-групою CyberAv3ngers, яка зазвичай атакує промислові системи КІІ у західних країнах. За даними OpenAI, акаунти, пов'язані з цією групою, попросили ChatGPT надати облікові дані, які використовуються за умовчанням у широко розповсюджених програмованих логічних контролерах (ПЛК), розробити кастомні bash- та Python-скрипти, а також обфуцьовати код. Крім того, іранські хакери використовували ChatGPT для планування пост-експлуатації, вивчення способів злому конкретних уразливостей та вибору методів крадіжки користувацьких паролів з macOS.

Ще один випадок стосується іншого іранського угруповання Storm-0817. Ця група використовувала ChatGPT для налагодження malware, створення скрапера для Instagram, перекладу профілів LinkedIn перською мовою та розробки власного шкідливого ПЗ для Android, а також пов'язаної з ним керуючої інфраструктури.

Повідомляється, що всі облікові записи, які використовували зловмисники, вже заблоковані, а пов'язані з ними індикатори компрометації, включаючи IP-адреси, передані партнерам OpenAI з кібербезпеки.

Отже можна зробити висновок, що штучний інтелект вже активно використовується в кібератаках.

1.4 Постановка задачі

Сучасний стан розвитку кібербезпеки демонструє, що кількість загроз для вебзастосунків продовжує зростати, стаючи дедалі складнішою та витонченішою. Водночас традиційні підходи до виявлення вразливостей часто

не здатні вчасно реагувати на нові типи атак, особливо ті, що ґрунтуються на використанні комбінованих методів чи експлуатації уразливостей, які ще не мають сигнатур. В цьому контексті інтеграція технологій машинного навчання, зокрема великих мовних моделей (LLM), таких як ChatGPT, відкриває нові горизонти для автоматизації процесів виявлення вразливостей та надання рекомендацій щодо їх усунення.

Використання ChatGPT для розробки спеціалізованого інструменту кібербезпеки дозволяє створити інтерактивний асистент, що поєднує автоматизований аналіз з гнучкістю й адаптивністю в роботі. Такий підхід надає етичним хакерам і спеціалістам з кібербезпеки інструмент, який не лише допомагає виявляти потенційні загрози, але й підтримує їх на кожному етапі пентесту, забезпечуючи інтеграцію з іншими інструментами, такими як Nmap, Burp Suite, або Metasploit.

Основною метою цієї роботи є розробка інтерактивного нейромережевого методу для виявлення вразливостей вебзастосунків, що базується на використанні ChatGPT API. Пропонований підхід дозволить об'єднати аналітичні можливості великих мовних моделей із класичними методами кібербезпеки, створюючи гібридний інструмент, який ефективно аналізує загрози, автоматизує процеси тестування та надає користувачам рекомендації у зрозумілому форматі.

Для досягнення цієї мети необхідно виконати такі завдання:

- провести детальний аналіз сучасних методів виявлення вразливостей, зокрема традиційних підходів та новітніх методів, що використовують машинне навчання. Особливу увагу приділити обмеженням існуючих підходів у контексті динамічних загроз;

- визначити функціональні вимоги до інтерактивного інструменту, які включатимуть сканування мережі, аналіз HTTP-заголовків, розпізнавання шаблонів загроз, а також підтримку інтеграції з іншими засобами кібербезпеки;

- створити метод для впровадження інтерактивного асистента, який надаватиме покрокові рекомендації для етичних хакерів, а також зможе взаємодіяти з такими інструментами, як Metasploit, або Nmap;
- запропонувати методи для інтеграції ChatGPT API, що дозволить ефективно використовувати його можливості в задачах виявлення вразливостей і автоматизації аналізу;
- розробити підінструменти для виконання специфічних завдань, таких як аналіз HTTP-заголовків, виявлення SQL-ін'єкцій, пошук XSS та CSRF. Забезпечити їхню взаємодію з центральним модулем системи;
- протестувати розроблений інструмент на реальних або симульованих вебзастосунках для оцінки точності виявлення вразливостей, продуктивності та практичності використання;
- узагальнити результати роботи, зробити висновки про ефективність запропонованого підходу та запропонувати напрями подальшого вдосконалення моделі.

Виконання цих завдань дозволить створити інструмент нового покоління для виявлення вразливостей вебзастосунків, що базується на поєднанні можливостей великих мовних моделей та традиційних методів кібербезпеки. Це сприятиме підвищенню ефективності роботи спеціалістів з кібербезпеки, скороченню часу реагування на загрози та забезпеченню інтеграції з іншими інструментами для гнучкого й адаптивного аналізу.

2 ДОСЛІДЖЕННЯ МОДЕЛЕЙ ШТУЧНОГО ІНТЕЛЕКТУ

2.1 Огляд сучасних мовних моделей штучного інтелекту

Завдяки своїм універсальним можливостям, великі мовні моделі (LLM) знаходять застосування в різних галузях - від медицини до фінансів. Особливий інтерес вони становлять у сфері кібербезпеки, де здатність працювати з великими обсягами інформації, аналізувати патерни, класифікувати загрози й пропонувати оптимальні рішення є критично важливою для захисту інформаційних систем.

У кібербезпеці мовні моделі відкривають нові горизонти можливостей. Вони здатні аналізувати великі масиви логів, виявляти аномальні дії, передбачати загрози на основі історичних даних та оптимізувати роботу спеціалістів у Security Operations Center (SOC). Наприклад, вони ефективно застосовуються для аналізу HTTP-заголовків, виявлення аномалій або підозрілих запитів у вебтрафіку. Крім того, моделі можуть автоматизувати пошук вразливостей, використовуючи бази даних CVE (Common Vulnerabilities and Exposures) та специфікацію OWASP Top 10. Ще одним важливим напрямком є протидія фішинговим атакам, де LLM допомагають розпізнавати підозрілі повідомлення або вебсайти за допомогою семантичного аналізу тексту.

У цьому підрозділі розглянуто п'ять провідних мовних моделей: GPT-4, Claude, PaLM, Llama та ChatRTX, а також їхні архітектурні особливості, переваги та недоліки.

Сучасні великі мовні моделі базуються на архітектурі трансформерів, яка стала проривом у сфері обробки природної мови. Розроблена у 2017 році, ця архітектура змінила підходи до аналізу тексту завдяки механізму уваги (attention mechanism), який дозволяє моделі фокусуватися на ключових частинах тексту та визначати взаємозв'язки між словами в різних частинах тексту.

Однією з основних переваг трансформерів є швидкість обробки даних. Завдяки паралельній обробці інформації трансформери забезпечують значно вищу продуктивність порівняно з RNN, що є критично важливим для аналізу логів у реальному часі. Також механізм самоуваги забезпечує контекстуальність моделі, дозволяючи враховувати не лише окремі слова, але й їхнє значення в контексті всього тексту. Ця особливість вкрай важлива для розуміння складних текстових структур, таких як фішингові повідомлення або інструкції для багатоступеневих атак.

Іншою важливою рисою трансформерів є гнучкість адаптації. Використання техніки донавчання (fine-tuning) дозволяє ефективно адаптувати моделі до специфічних задач, наприклад, аналізу баз даних CVE або класифікації вразливостей. Масштабованість трансформерів також є їхньою сильною стороною, адже вони можуть працювати із великими наборами даних, забезпечуючи високу якість результатів навіть для складних задач.

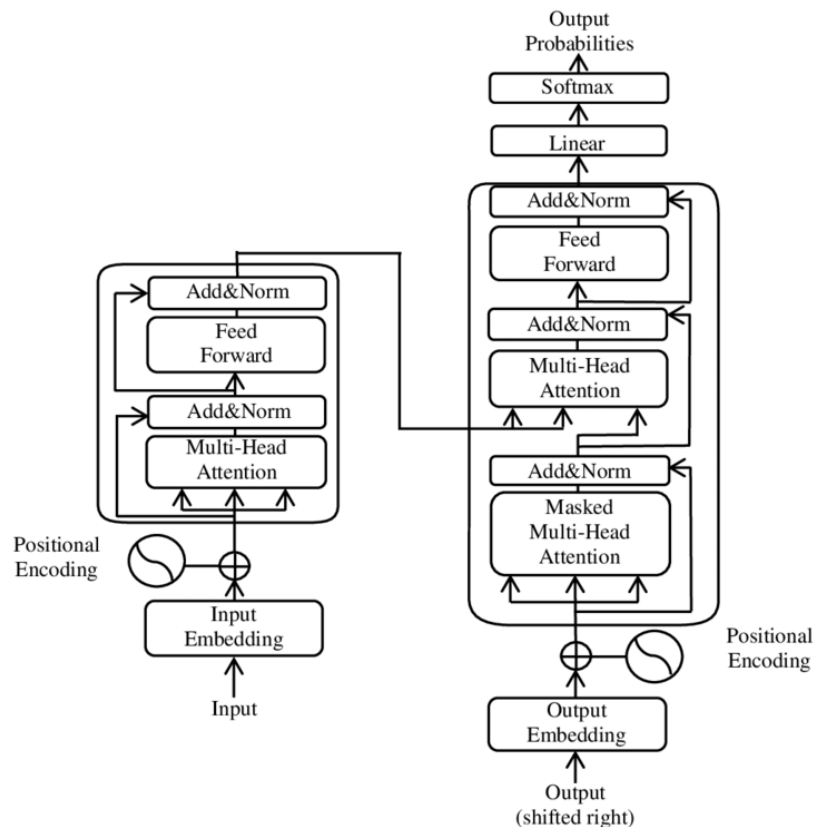


Рисунок 2.1 - Схема архітектури трансформера

Архітектура трансформера включає кілька ключових компонентів. Механізм самоуваги аналізує кожне слово у контексті всього тексту, визначаючи його значущість у залежності від інших слів. Нормалізація та позиційні вектори забезпечують врахування порядку слів у тексті, що критично важливо для розуміння складних залежностей. Обчислювальні шари дозволяють моделі адаптуватися до різних задач через оптимізацію ваг у нейронній мережі.

GPT-4 є однією з найбільш універсальних і потужних моделей. Її здатність до глибокого контекстуального аналізу робить її незамінною для складних сценаріїв у кібербезпеці. Наприклад, GPT-4 використовується для виявлення складних атак, таких як SQL-ін'єкції або багаторівневі загрози, а також для генерації звітів про інциденти з рекомендаціями. Крім того, модель здатна автоматизувати аналіз мережевих журналів. Основним недоліком GPT-4 є високі вимоги до обчислювальних ресурсів, що обмежує її використання для невеликих організацій [42].

Claude створена з акцентом на швидкість і простоту використання. Вона особливо корисна для SOC, де потрібна негайна реакція на інциденти безпеки. Модель демонструє високу ефективність в аналізі логів для виявлення аномалій або обробці повідомлень у внутрішній мережі. Проте через меншу кількість шарів самоуваги її точність у складних задачах поступається GPT-4 [43].

PaLM інтегрується з Google Cloud, що робить її зручною для корпоративного використання. Модель ідеально підходить для аналізу великих масивів даних, наприклад, поштових скриньок для виявлення фішингових листів або виявлення аномалій у корпоративних мережах. Водночас її адаптивність до нових типів загроз є обмеженою [44].

Llama демонструє високу ефективність у задачах середньої складності, таких як аналіз HTTP-запитів або перевірка конфігураційних файлів [45]. Її головною перевагою є низькі вимоги до ресурсів, що дозволяє використовувати модель навіть на локальних серверах із обмеженими обчислювальними можливостями.

ChatRTX вирізняється своєю оптимізацією для роботи з графічними процесорами (GPU). Це робить її ідеальною для масштабних задач, таких як аналіз великих масивів логів у реальному часі чи прогнозування атак на основі попередніх даних [46].

Таблиця 2.1 - Порівняння моделей за ключовими характеристиками

Модель	GPT-4	Claude	PaLM	Llama	ChatRTX
Точність	95%	85%	88%	82%	90%
Швидкість обробки	Середня	Висока	Середня	Середня	Висока
Використання пам'яті	Висока	Середня	Висока	Низька	Середня
Адаптивність	Висока	Середня	Низька	Середня	Середня
Масштабованість	Середня	Середня	Висока	Низька	Висока

Огляд моделей демонструє їхні переваги та обмеження залежно від задач і доступних ресурсів. GPT-4 залишається лідером для складних сценаріїв, тоді як Claude та PaLM підходять для швидкої обробки та корпоративного використання. Llama є оптимальним вибором для обмежених ресурсів, а ChatRTX забезпечує найвищу продуктивність у масштабних задачах. У наступному підрозділі розглянемо їхнє застосування для конкретних задач кібербезпеки.

2.2 Оцінка можливостей мовних моделей для задач кібербезпеки

Мовні моделі штучного інтелекту (LLM) стали потужними інструментами, які змінюють підходи до забезпечення кібербезпеки. Їхня здатність аналізувати великі обсяги даних, такі як журнали подій, конфігураційні файли, мережеві лог-файли, запити HTTP та інші текстові

ресурси, значно перевищує можливості традиційних методів. Завдяки LLM фахівці з кібербезпеки можуть не тільки ідентифікувати загрози, а й аналізувати їхній контекст, класифікувати за рівнем ризику та генерувати рекомендації для їх усунення [47, 48].

Наприклад, GPT-4 здатна працювати з великими текстовими масивами, такими як журнали серверів або мережевий трафік, знаходячи важливі зв'язки між розрізненими частинами документа. Це особливо важливо при аналізі логів, які можуть містити десятки або навіть сотні тисяч рядків. Традиційні методи потребують ручного аналізу або роботи з жорстко визначеними правилами, тоді як GPT-4 здатна автоматично виявляти складні шаблони, такі як повторювані помилки, що можуть вказувати на SQL-ін'єкції або інші кіберзагрози.

Однією з ключових переваг LLM є їхній контекстуальний підхід до аналізу даних. Замість використання лише фіксованих правил, моделі здатні розуміти загальну картину й адаптуватися до різних сценаріїв. Наприклад, якщо у мережевих логах спостерігається специфічний шаблон помилок, GPT-4 може автоматично визначити причину проблеми, виявляючи потенційні загрози, як-от XSS-атаки чи спроби обходу автентифікації. Такий підхід забезпечує більшу гнучкість і точність, ніж у традиційних методах аналізу [49].

Окрім аналізу логів, GPT-4 може класифікувати знайдені загрози за їхньою важливістю. Це дозволяє командам безпеки спрямовувати ресурси на усунення найбільш критичних проблем, що значно знижує ризики для організації. Наприклад, модель може розподілити загрози за рівнями: високий (DDoS, SQL-ін'єкції), середній (фішингові атаки) або низький (застарілі протоколи) [47].

LLM також добре справляються із задачами, які потребують миттєвого аналізу. Модель Claude демонструє високу швидкість у цьому аспекті, дозволяючи її використовувати для моніторингу мережевого трафіку в реальному часі. Наприклад, під час аналізу даних модель може швидко виявляти аномалії, як-от спроби сканування портів, несанкціонований доступ

чи підозрілі спроби встановлення з'єднань. Завдяки цьому Claude ідеально підходить для роботи в SOC (Security Operations Center), де затримки в аналізі можуть призводити до серйозних наслідків [48].

Для більш складних сценаріїв, таких як виявлення аномалій у великих обсягах даних, спеціалізовані моделі, наприклад ChatRTX, використовують свої переваги в обробці даних із графічними процесорами. Вони здатні швидко знаходити незвичайні шаблони у мережевому трафіку, наприклад, якщо певний клієнт надсилає велику кількість запитів на сервер або звертається до підозрілих шляхів. Це може бути визначено як DDoS-атака чи спроба обходу системи автентифікації. Завдяки цьому ChatRTX стає потужним інструментом для проактивного моніторингу мереж [50, 51].

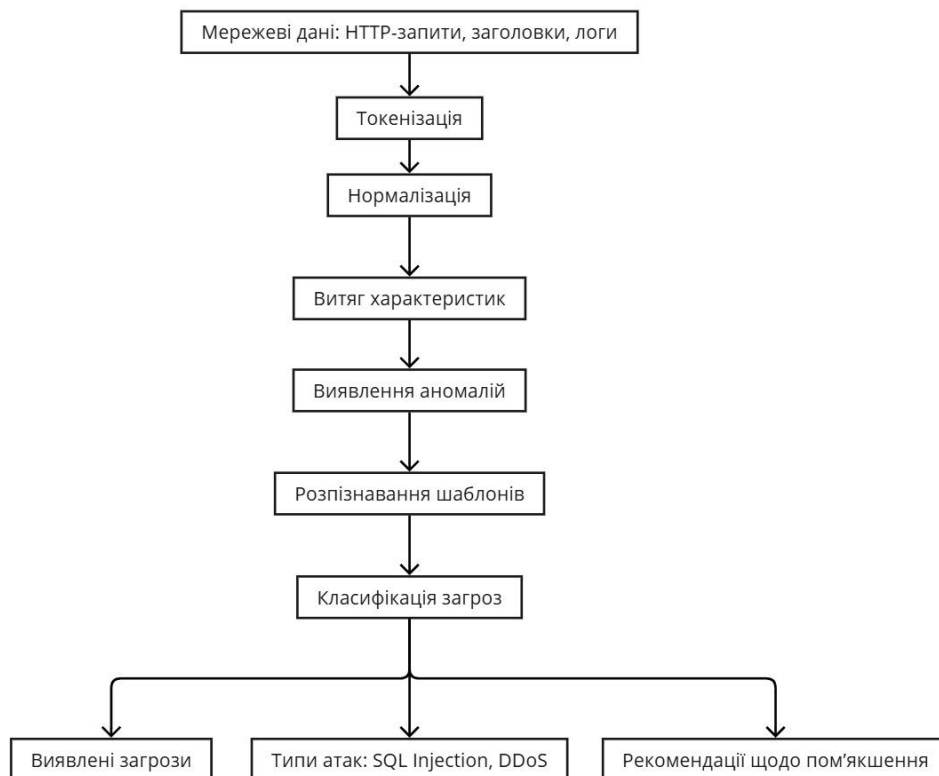


Рисунок 2.2 - Схематична діаграма процесу аналізу мережевого трафіку LLM

Ще однією важливою особливістю LLM є здатність генерувати текстові рекомендації на основі виявлених загроз. Наприклад, після ідентифікації вразливості GPT-4 може створити детальну інструкцію щодо її усунення. Це

може включати технічні рекомендації (виправлення коду, налаштування серверів) і організаційні заходи (наприклад, проведення аудиту безпеки). Такі рекомендації є особливо корисними для невеликих організацій, які мають обмежений доступ до експертів у сфері кібербезпеки [49].

Попри свої переваги, LLM мають і певні обмеження, які потрібно враховувати. Наприклад, моделі, як-от GPT-4, вимагають значних обчислювальних ресурсів для роботи з великими даними. Це може бути викликом для невеликих компаній, які не мають доступу до потужної інфраструктури. Крім того, моделі іноді видають хибнопозитивні результати, класифікуючи нормальну поведінку як загрозу, що призводить до витрат часу на перевірку.

Обмежена адаптивність є ще одним аспектом, який варто враховувати. Наприклад, Claude та PaLM можуть не демонструвати високої гнучкості при адаптації до нових типів атак. Крім того, ChatRTX оптимізована для роботи з графічними процесорами NVIDIA, що обмежує її універсальність у середовищах із іншим обладнанням [51].

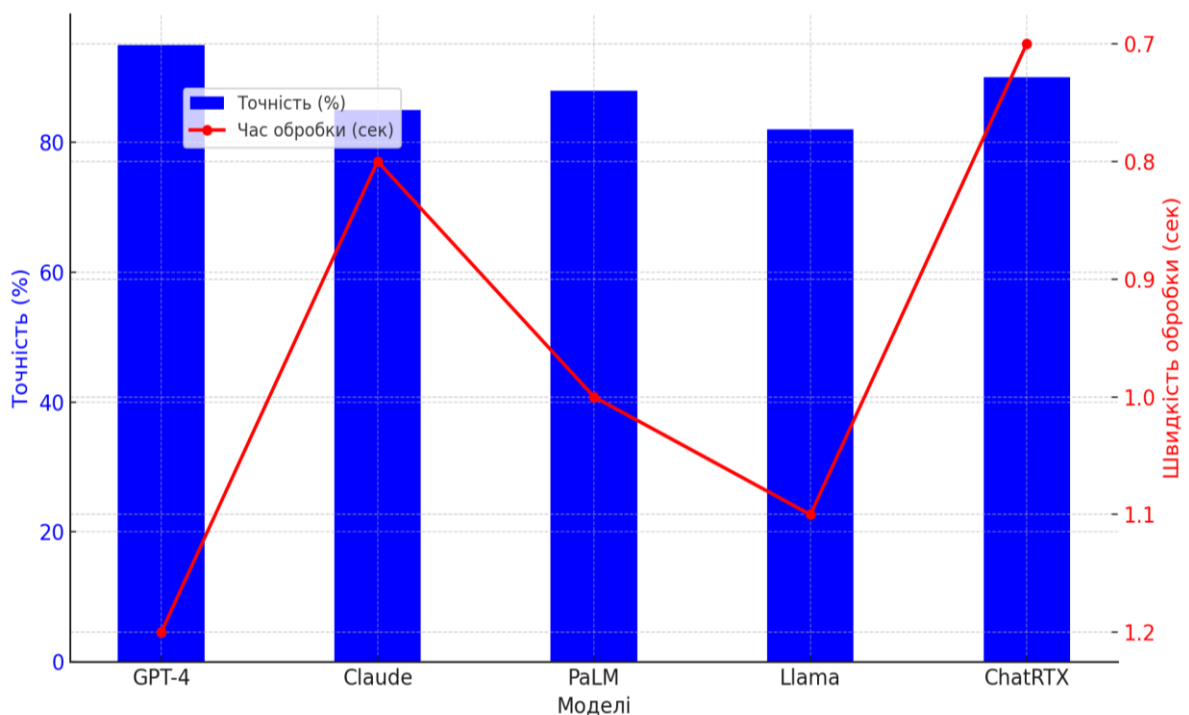


Рисунок 2.3 - Графік порівняння точність і швидкість моделей.

Розвиток LLM відкриває нові можливості для інтеграції їх із існуючими інструментами кібербезпеки, такими як Burp Suite, Nmap або Wireshark. Наприклад, GPT-4 може аналізувати результати сканування в реальному часі, пропонуючи рекомендації з усунення вразливостей, а ChatRTX може забезпечувати моніторинг мережевого трафіку на основі історичних даних. Крім того, технології зворотного навчання дозволяють LLM адаптуватися до нових загроз, що забезпечує постійне покращення їхньої ефективності [47].

Мовні моделі штучного інтелекту демонструють значний потенціал у задачах кібербезпеки. Їхні можливості включають аналіз великих обсягів даних, класифікацію загроз, виявлення аномалій та надання рекомендацій. Незважаючи на деякі обмеження, такі як висока потреба в ресурсах і ймовірність хибнопозитивних результатів, LLM залишаються перспективним інструментом для оптимізації роботи команд безпеки. Завдяки здатності адаптуватися до специфічних сценаріїв, вони забезпечують ефективніше виявлення вразливостей і підвищення рівня кіберзахисту організацій.

У майбутньому можливе створення комплексних рішень, які об'єднують кілька моделей для виконання різних задач. Наприклад, одна модель може бути відповідальною за аналіз текстових даних, таких як журнали подій, а інша - за аналіз мережевого трафіку. Це дозволить створити універсальні системи кібербезпеки, які здатні швидко реагувати на складні загрози.

2.3 Використання LLM у системах кібербезпеки

Розвиток сучасних технологій штучного інтелекту, зокрема великих мовних моделей (LLM), відкриває нові можливості для автоматизації процесів у сфері кібербезпеки. Інтеграція LLM у системи виявлення та аналізу загроз зменшує навантаження на фахівців, прискорює реагування на інциденти та підвищує точність аналізу. Потенціал цих технологій є значним, але їх використання супроводжується певними викликами та обмеженнями.

Великі мовні моделі демонструють високу ефективність у широкому спектрі завдань, включаючи аналіз логів, ідентифікацію аномальної поведінки в мережевому трафіку та автоматизацію виявлення вразливостей. Раніше ці завдання виконувалися вручну або за допомогою вузькоспеціалізованих інструментів із жорстко визначеними правилами. Мовні моделі здатні автоматизувати рутинні завдання, швидко адаптуватися до нових типів атак і надавати детальні звіти, що забезпечує значну економію часу та ресурсів [52].

Одним із найбільш перспективних напрямів є інтеграція LLM із популярними інструментами кібербезпеки, такими як Nmap, SQLMap і Burp Suite. Наприклад, GPT-4 може аналізувати результати сканування, автоматично визначати найбільш критичні загрози й пропонувати конкретні кроки для їх усунення. Це знижує навантаження на спеціалістів із кібербезпеки та забезпечує швидше реагування на інциденти [53, 54].

Ключові переваги інтеграції включають:

- автоматизація процесів, LLM здатні обробляти великі обсяги даних без участі людини. Наприклад, GPT-4 може аналізувати журнали подій і виявляти шаблони, що свідчать про потенційні загрози, як-от аномальні запити або повторювані помилки. Це особливо важливо для великих інфраструктур, де обсяг даних перевищує можливості ручного аналізу;

- підвищення точності, завдяки трансформерній архітектурі, мовні моделі забезпечують кращу контекстуальну обробку даних, знижуючи ймовірність хибнопозитивних результатів. Наприклад, Claude може знаходити взаємозв'язки між подіями у логах і фокусуватися лише на подіях, які дійсно є підозрілими, ігноруючи несуттєві відхилення;

- зменшення часу на реагування, інтеграція з інструментами кібербезпеки дозволяє автоматично генерувати звіти та рекомендації, що значно прискорює реагування на загрози. Наприклад, GPT-4 у поєднанні з Burp Suite може створювати технічну документацію щодо виявлених вразливостей, спрощуючи комунікацію між командами безпеки;

– адаптивність до нових загроз, завдяки механізмам зворотного навчання LLM можуть швидко оновлювати свої знання для виявлення нових типів атак. Це знижує залежність від фіксованих сигнатур і підвищує ефективність у сценаріях із невідомими загрозами.

Попри значні переваги, інтеграція LLM у системи кібербезпеки супроводжується певними викликами:

– високі вимоги до обчислювальних ресурсів, для роботи з великими обсягами даних мовні моделі потребують значних обчислювальних потужностей. Це може бути перешкодою для малих компаній або організацій із обмеженим бюджетом;

– залежність від якості вхідних даних, моделі чутливі до якості вхідних даних. Неструктуровані, неповні або неправильно зібрані дані можуть суттєво знизити точність аналізу, вимагаючи додаткового передоброблення;

– ризик хибнопозитивних результатів, у деяких випадках моделі класифікують нормальну поведінку як загрозу. Наприклад, при аналізі складних мережових патернів ChatRTX може виявити відхилення, які насправді не є загрозливими. Це створює додаткове навантаження на команди кібербезпеки;

– обмежена інтерпретація складних сценаріїв, незважаючи на високий рівень адаптивності, мовні моделі можуть мати труднощі з розумінням багатофакторних або нетипових сценаріїв атак. Це потребує використання додаткових спеціалізованих інструментів;

– потреба в регулярному оновленні, щоб зберігати актуальність і точність, моделі потребують регулярного оновлення, включаючи введення нових даних про загрози. Це може бути технічно складним і ресурсозатратним процесом [55].

Розвиток технологій відкриває нові перспективи для використання LLM у кібербезпеці. Серед можливих напрямів розвитку:

– багатомодальні моделі, майбутні системи можуть поєднувати обробку текстових, візуальних і структурованих даних. Наприклад, одночасний аналіз

текстових логів і мережевих діаграм дозволить створити більш повну картину інциденту;

– гібридні рішення, використання кількох спеціалізованих моделей для різних типів даних (текстові журнали, мережевий трафік, графічні дані) дозволить підвищити точність аналізу;

– хмарні платформи, інтеграція LLM із хмарними технологіями забезпечить масштабованість і зниження витрат на інфраструктуру. Наприклад, GPT-4 може бути розгорнута в хмарі для аналізу великих обсягів логів без необхідності локальних серверів.

Інтеграція мовних моделей у системи кібербезпеки демонструє значний потенціал, дозволяючи автоматизувати аналіз загроз, підвищувати точність і скорочувати час реагування. Попри деякі обмеження, такі як висока вартість і залежність від якості даних, LLM залишаються важливим елементом майбутніх систем кіберзахисту. Для досягнення максимальної ефективності необхідно забезпечити якісне налаштування моделей, доступ до структурованих даних і потужну інфраструктуру. У наступному розділі буде детально розглянуто конкретні методи застосування LLM для виявлення вебвразливостей [56].

2.4 Матиматична модель для аналізу та прогнозування загроз за допомогою мовних моделей

Поєднання традиційного підходу до пошуку вразливостей із можливостями мовних моделей штучного інтелекту (LLM) відкриває нові горизонти для забезпечення кібербезпеки. Мовні моделі не лише підвищують точність виявлення вразливостей, але й дозволяють здійснювати прогнозування потенційних загроз, завчасно реагуючи на ризики. У цьому розділі детально розглянуто особливості, переваги та обмеження цих підходів.

Традиційний сигнатурний аналіз залишається фундаментом у системах кібербезпеки. Його основа - це порівняння вхідних даних із наперед визначеними шаблонами атак.

$$V = \sum_{i=1}^n S(x_i), \quad (2.1)$$

де $S(x_i)$ - функція, що повертає 1, якщо об'єкт x_i відповідає відомій сигнатурі загрози, і 0, якщо не відповідає.

Традиційний підхід ефективний для виявлення:

- SQL-ін'єкцій, шаблони, такі як наявність SELECT, UNION або некоректно екранованих символів;
- XSS, виявлення спеціальних HTML-тегів або JavaScript-фрагментів у полях вводу;
- CSRF, аналіз запитів без необхідного токена перевірки.

Цей метод забезпечує швидкість і простоту реалізації, але має значні обмеження, особливо щодо адаптивності до нових атак.

Я пропоную доповнити традиційний метод можливостями мовних моделей, таких як GPT-4, Claude та Llama. Завдяки своїй архітектурі LLM можуть аналізувати не лише синтаксичні, але й семантичні особливості вхідних даних.

$$V_{AI} = \sum_{i=1}^n (S(x_i) + M(x_i)), \quad (2.2)$$

де $M(x_i)$ - функція, що аналізує ймовірність загрози на основі моделі ШІ.

LLM дозволяють здійснювати аналіз великих обсягів історичних даних, таких як журнали подій чи мережевий трафік, для виявлення прихованих шаблонів, що передують атакам. Наприклад:

- GPT-4 здатний аналізувати журнали серверів для виявлення підозрілих шаблонів, які передували DDoS-атакам;
- Claude може зіставляти інциденти в різних системах, виявляючи схожі тригери загроз;
- ChatRTX оцінює зміну мережевого трафіку для визначення потенційної атаки.

LLM також можуть допомогти передбачити загрози на основі відкритих портів або застарілих протоколів, які можуть бути експлуатовані зловмисниками.

Окрім аналізу технічних даних, LLM здатні працювати з інформацією стратегічного характеру:

- моніторинг новин і звітів: GPT-4 може аналізувати технічні блоги або звіти з безпеки, прогнозуючи нові сценарії атак;
- ідентифікація галузевих ризиків: Наприклад, збільшення атак на IoT-пристрої дозволяє передбачити потенційні атаки на медичні пристрої.

Запропонований метод інтеграції традиційного аналізу з LLM дозволяє досягти нових висот у кібербезпеці. Мовні моделі забезпечують не лише виявлення існуючих вразливостей, але й прогнозування потенційних загроз. Завдяки своїй адаптивності та можливості працювати з великими масивами даних, вони створюють основу для проактивних заходів у захисті інформаційних систем.

2.5 Етичні та правові аспекти застосування мовних моделей у кібербезпеці

Мовні моделі великого масштабу (LLM), такі як ChatGPT, здатні виконувати складні завдання, які раніше вимагали значних зусиль і часу від людини. Однак їхнє використання у сфері кібербезпеки породжує низку етичних викликів. Одним із ключових ризиків є можливість використання

таких моделей для створення загроз, наприклад, автоматизації атак типу «соціальна інженерія» або розробки фішингових повідомлень, які виглядають дуже переконливо завдяки високій якості тексту, що генерується. Це може значно збільшити ефективність таких атак і знизити поріг входження для нових кіберзлочинців.

Ще одним важливим етичним питанням є залежність від моделі як інструменту прийняття рішень. Оскільки мовні моделі можуть видавати як коректні, так і некоректні результати, їх використання без належного нагляду людини може призводити до серйозних помилок, особливо в контексті виявлення або усунення кіберзагроз. Наприклад, модель може пропустити критичну вразливість або надати недостовірну інформацію про ризики, що створює серйозну загрозу для інформаційної безпеки.

Крім того, є ризик етичного зловживання технологіями, коли мовні моделі використовуються для створення шкідливого програмного забезпечення, автоматизації атак на веб-додатки або розробки нових векторів атак. Усе це може призвести до зростання кількості кібератак, зокрема тих, які складно виявити традиційними методами.

Мовні моделі вимагають великих обсягів даних для навчання, які часто включають персональні або конфіденційні дані. Навіть якщо ці дані анонімізовані, існує ризик ненавмисного відтворення або витоку приватної інформації в процесі генерації відповідей. Наприклад, модель може випадково відтворити фрагменти вихідних даних, що використовувалися під час її навчання.

У сфері кібербезпеки також актуальним є питання аналізу поведінкових даних. Використання мовних моделей для моніторингу та аналізу дій користувачів в інтернеті може суперечити принципам приватності та захисту даних. Зокрема, автоматичний збір і обробка даних користувачів у соціальних мережах або на інших платформах може бути розцінений як порушення їхнього права на конфіденційність.

На глобальному рівні регулювання використання штучного інтелекту, включаючи мовні моделі, ще тільки формується. Однак деякі країни вже розробляють законодавчі акти, які визначають межі використання ШІ. Наприклад, Європейська комісія розробляє AI Act, який передбачає створення стандартів і вимог для використання штучного інтелекту в різних галузях, зокрема в кібербезпеці. У цих стандартах значна увага приділяється питанням прозорості, відповідальності та мінімізації ризиків, пов'язаних із використанням ШІ.

В Україні питання захисту даних регулюється Законом «Про захист персональних даних». Впровадження мовних моделей у сферу кібербезпеки повинно враховувати відповідність таким нормам, особливо коли йдеться про обробку конфіденційної інформації чи використання даних про користувачів для навчання моделей.

Використання мовних моделей у кібербезпеці викликає питання про те, хто несе відповідальність за наслідки роботи таких систем. Наприклад:

- якщо модель пропустить критичну вразливість, хто буде відповідальний за збитки: розробник моделі, користувач чи компанія, що впровадила ШІ;

- у випадках неправильного використання моделі (наприклад, для створення кібератак), як визначити винного і які заходи слід вжити.

Ці питання вимагають створення чітких регламентів і процедур для встановлення відповідальності на всіх рівнях: від розробників до кінцевих користувачів.

Для мінімізації ризиків, пов'язаних із використанням мовних моделей у кібербезпеці, можна запропонувати такі заходи:

- підтримка людського контролю. Незважаючи на автоматизацію, важливо, щоб кінцеві рішення перевірялися експертами;

- розробка прозорих моделей. Розробники повинні забезпечувати пояснюваність рішень моделі та її функціональних обмежень;

- етичні навчальні набори. Важливо уникати використання неетичних або конфіденційних даних у процесі навчання моделей;
- регулярний аудит. Моделі повинні постійно перевірятися на точність, етичність і відповідність правовим нормам.

Інтеграція мовних моделей у сферу кібербезпеки має величезний потенціал для підвищення ефективності захисту інформаційних систем. Проте їхнє використання потребує відповідального підходу, який враховує етичні, правові та технічні аспекти. Забезпечення балансу між автоматизацією процесів і людським контролем дозволить уникнути зловживань та забезпечити прозоре, надійне та безпечне впровадження технологій штучного інтелекту.

2.6 Висновок

Мовні моделі штучного інтелекту (LLM) демонструють значний потенціал для застосування в сучасних системах кібербезпеки. Їхнє використання дозволяє автоматизувати рутинні процеси, підвищити точність аналізу загроз, ефективно класифікувати інциденти та прогнозувати можливі атаки. Завдяки трансформерній архітектурі та можливостям донавчання, LLM забезпечують глибокий контекстуальний аналіз, що виходить за межі традиційних методів.

Основні переваги інтеграції LLM у кібербезпеку включають:

- здатність обробляти великі обсяги даних у реальному часі;
- зниження навантаження на команди безпеки завдяки автоматизації;
- адаптивність до нових загроз завдяки регулярному оновленню моделей.

Розглянуті приклади, такі як GPT-4, Claude та ChatRTX, показали високу ефективність у таких завданнях, як аналіз логів, виявлення аномалій, класифікація загроз та прогнозування zero-day атак. Водночас моделі можуть створювати технічні рекомендації, що сприяє швидшому та ефективнішому усуненню вразливостей.

Однак, попри значні переваги, використання LLM супроводжується низкою викликів. До них належать висока вимога до обчислювальних ресурсів, залежність від якості вхідних даних і ризик хибнопозитивних результатів. Крім того, інтеграція LLM потребує якісного налаштування, доступу до структурованих даних і потужної інфраструктури.

Перспективи розвитку включають інтеграцію багатомодальних моделей, створення гібридних рішень для аналізу різних типів даних і використання хмарних платформ для масштабування обчислювальних потужностей. Такий підхід дозволить досягти нового рівня автоматизації та точності в кібербезпеці.

У цьому розділі висвітлено значення мовних моделей для виявлення, аналізу та прогнозування загроз, а також підкреслено їхню роль у проактивному захисті інформаційних систем.

3 РОЗРОБКА МЕТОДУ ПОШУКУ ТА АНАЛІЗУ ВЕБ ВРАЗЛИВОСТЕЙ З ДОПОМОГОЮ ШТУЧНОГО ІНТЕЛЕКТУ

3.1 Нейромережевий метод пошуку вразливостей

Великі мовні моделі (LLM), такі як GPT-4, є перспективними інструментами для пошуку вебвразливостей, але їх використання має певні обмеження. Самостійно вони не можуть виконувати практичні дії, наприклад, запускати сканування чи експлуатувати знайдені вразливості. Для подолання цього бар'єру розроблено ітеративний підхід «людина в циклі» [58]. У цьому підході LLM виступає як центральний інтелектуальний механізм, який надає рекомендації, а фахівець виконує інструкції моделі у тестовому середовищі.

Цей підхід побудований на інтерактивній співпраці між LLM і тестувальником. Процес передбачає наступні етапи:

- фахівець надає моделі інформацію про цільову систему або вебдодаток;
- LLM формулює рекомендації щодо проведення тестування, наприклад, пропонує перевірити конфігурації серверів чи запустити сканування наявності вразливостей;
- експерт виконує дії, запропоновані моделлю, наприклад, використовує Nmap для збору даних про відкриті порти;
- отримані результати аналізуються моделлю, і вона формулює наступні кроки. Цей процес повторюється до досягнення кінцевої мети тестування або виявлення «глухого кута».

Завдяки трансформерній архітектурі, LLM забезпечують:

- контекстуальний аналіз даних, що дозволяє виявляти складні шаблони у логах чи HTTP-запитах;
- генерацію рекомендацій для тестування, наприклад, сценаріїв перевірки SQL-ін'єкцій чи XSS-атак;
- автоматизацію аналізу результатів і класифікацію знайдених вразливостей за рівнем критичності.

Людина-експерт у цьому підході виступає не як аналітик, а як виконавець, що строго дотримується інструкцій моделі. Це дозволяє уникнути впливу суб'єктивної думки фахівця на процес тестування, водночас даючи змогу моделі працювати з результатами реальних дій.

Для роботи з графічними інструментами, такими як BurpSuite, модель пропонує покрокові текстові інструкції. Фахівець виконує ці дії та документує результати у текстовій формі для подальшого аналізу моделлю [59].

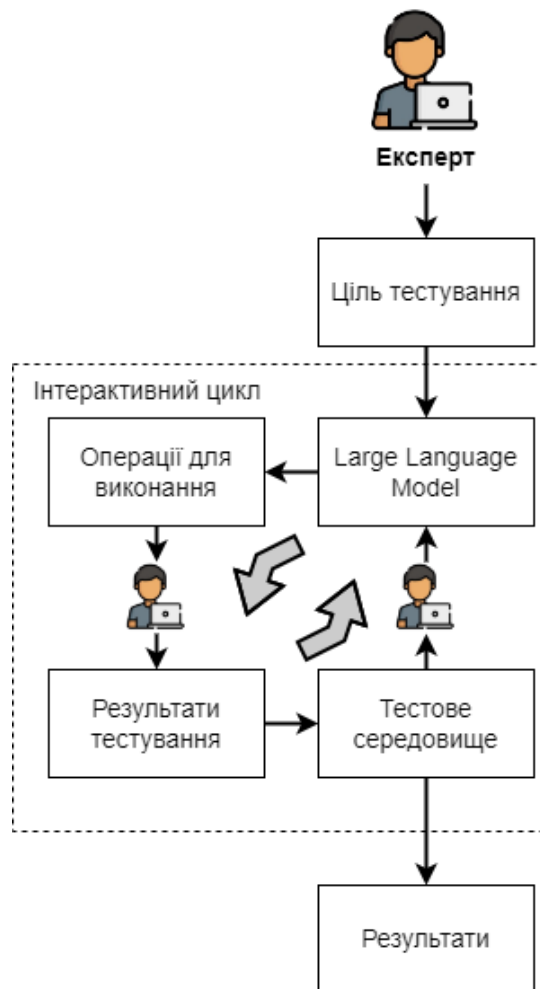


Рисунок 3.1 - Схема нейронного мережевого методу

У тестовому середовищі LLM може рекомендувати виконати сканування на відкриті порти за допомогою Nmap. Отримані результати, наприклад, список відкритих портів та відповідних сервісів, передаються моделі, яка пропонує наступні кроки, наприклад, перевірку наявності застарілих протоколів або слабких паролів.

Переваги підходу:

- швидкість, моделі дозволяють швидко обробляти великі обсяги даних, такі як конфігурації серверів або результати сканування;
- точність, Інтерактивний аналіз забезпечує виявлення нетипових загроз, які могли б бути пропущені під час ручного тестування;
- адаптивність, LLM можуть динамічно коригувати свої рекомендації на основі результатів попередніх дій.

Обмеження:

- хибнопозитивні результати, моделі іноді можуть класифікувати нормальну поведінку як загрозу, що створює додаткове навантаження на тестувальників;
- залежність від якості даних, для ефективної роботи моделі потребують структурованих і релевантних даних;
- обмеження автоматизації, деякі задачі, такі як управління GUI-інструментами, вимагають участі людини.

Нейромережевий метод пошуку вразливостей, реалізований через ітеративний підхід «людина в циклі», демонструє значний потенціал для забезпечення кібербезпеки. Використання LLM у поєднанні з експертними діями дозволяє оптимізувати процеси тестування, забезпечуючи глибокий аналіз та адаптивність до нових загроз. Цей підхід гармонійно доповнює традиційні методи, поєднуючи інтелектуальну потужність ШІ з практичними діями фахівців.

3.2 Інтерактивна інженерія підказок для динамічного тестування безпеки

Мій підхід до виявлення вебвразливостей передбачає інтеграцію великих мовних моделей, таких як GPT, у процеси тестування. Одним із ключових етапів є створення та аналіз тестових запитів, що дозволяють ефективно знаходити потенційні проблеми в системах.

Для того щоб мовна модель могла виконувати специфічні завдання з тестування безпеки, її необхідно налаштувати та підготувати. Цей процес включає навчання на даних, що відображають типові сценарії атак і можливі вразливості. Сюди входять приклади SQL-ін'єкцій, XSS-атак, методів обфускації даних і стратегій обходу фільтрації. Важливо також забезпечити створення таких промптів, які чітко окреслюють задачу моделі, спрямовуючи її на аналіз саме тих елементів системи, які можуть бути вразливими.

Особливу увагу приділяють гнучкості підходу. Завдяки спеціалізованим інструкціям модель може зосереджуватися на аналізі окремих аспектів, таких як HTTP-запити, автентифікація або робота з cookie-файлами. Це дозволяє забезпечити ефективність навіть у складних умовах тестування.

Інтерактивний підхід до налаштування моделі є одним із найважливіших аспектів сучасного тестування. Формулювання чітких і зрозумілих інструкцій для мовної моделі створює необхідний контекст, в якому вона може ефективно виконувати свої завдання. Наприклад, під час аналізу HTTP-заголовків модель може бути спрямована на пошук помилок у валідації введених даних або виявлення проблем із сесійним управлінням.

Такий підхід забезпечує динамічну взаємодію між моделлю та вебдодатком. Якщо початковий запит не дає очікуваного результату, модель автоматично генерує нові запити, враховуючи попередні відповіді сервера. Це створює ітеративний процес, у якому кожен наступний крок стає більш точним і релевантним.

Важливим аспектом роботи моделей GPT є здатність адаптуватися до отриманих даних. У процесі тестування модель аналізує відповіді серверів і, виходячи з них, пропонує нові варіанти payload-ів для перевірки. Наприклад, якщо сервер повертає помилку 500, модель може виявити потенційну проблему у структурі запиту та запропонувати інший підхід.

Цей підхід дозволяє виявляти навіть складні вразливості. Якщо базові запити не спрацьовують, модель автоматично генерує складніші payload-и,

використовуючи різні техніки обходу, включаючи обфускацію, зміну кодування чи застосування спеціальних функцій.

Для підвищення ефективності тестування нейромережеві моделі інтегруються з популярними інструментами безпеки, такими як Burp Suite, OWASP ZAP, Nmap та SQLMap. Це дозволяє автоматизувати аналіз відповідей сервера, формувати рекомендації для усунення вразливостей і навіть створювати звіти для розробників [60].

Наприклад, у випадку з Nmap модель може аналізувати результати сканування портів, виділяючи потенційно небезпечні сервіси. За допомогою Burp Suite GPT здатна не тільки ідентифікувати проблеми, але й запропонувати конкретні технічні рішення для їх усунення.

Користувач взаємодіє з інтерфейсом (рис 3.2), який передає запити до модуля генерації.

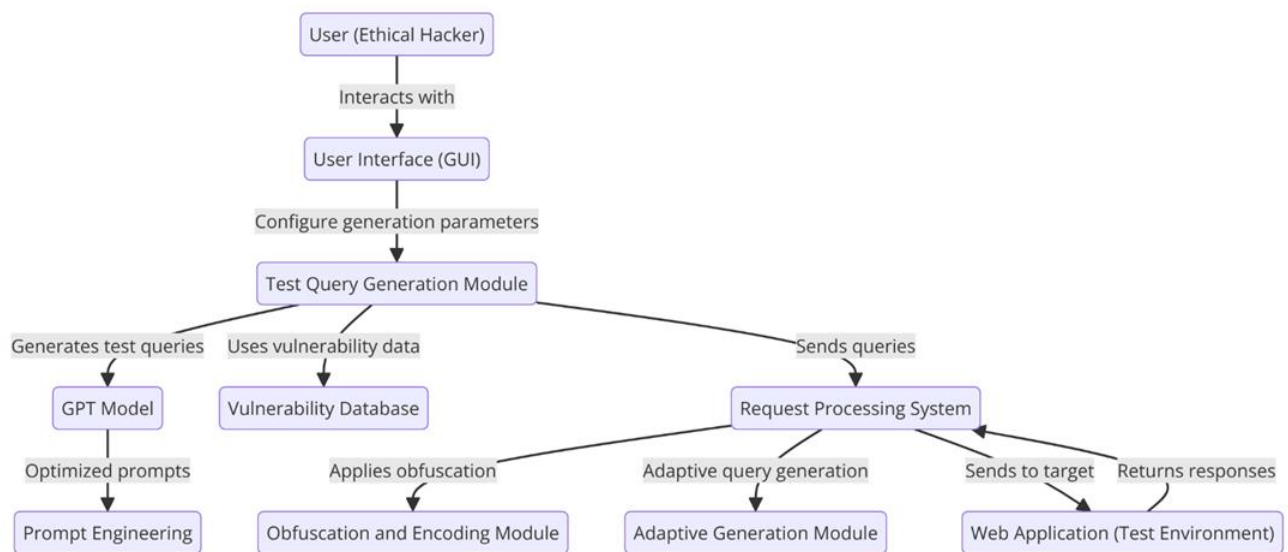


Рисунок 3.2 - Структура системи для створення текстових запитів

Цей модуль формує промпти для моделі GPT, яка генерує запити для перевірки вебдодатка. Відповіді сервера обробляються для вдосконалення наступних запитів.

Інтерактивна інженерія підказок із використанням мовних моделей значно підвищує ефективність і точність тестування безпеки. Завдяки

динамічному аналізу та адаптації модель здатна виявляти навіть найскладніші вразливості, одночасно автоматизуючи багато рутинних завдань.

Цей підхід не лише зменшує трудовитрати, але й покращує якість аналізу, дозволяючи сконцентруватися на найважливіших аспектах тестування. У поєднанні з інструментами автоматизації він створює потужну платформу для етичного хакінгу, яка здатна ефективно відповідати на виклики сучасного кіберпростору.

На наступному етапі здійснюється обробка й аналіз відповідей, які вебзастосунки повертають у відповідь на згенеровані тестові запити. Основна мета – виявлення ознак потенційних вразливостей шляхом систематичного аналізу отриманих даних.

Усі відповіді сервера на тестові запити ретельно фіксуються та зберігаються у структурованому форматі для подальшого аналізу. Попередня обробка отриманих даних включає кілька важливих кроків:

- видалення зайвих символів і пробілів для спрощення аналізу;
- приведення тексту до уніфікованого кодування, що дозволяє уникнути помилок у подальшій роботі;
- виокремлення ключових частин відповіді, таких як HTTP-коди статусу, заголовки та тіло повідомлення.

Ці дії забезпечують узгоджений формат даних, який підходить для ефективного машинного аналізу.

Для виявлення потенційних загроз використовуються великі мовні моделі, такі як GPT. Їх завдання – аналіз тексту відповідей серверів, розпізнавання аномалій та виявлення закономірностей, які можуть свідчити про наявність вразливостей. Цей процес включає кілька етапів:

- розпізнавання аномалій, модель шукає ознаки несподіваної поведінки, наприклад, повідомлення про помилки, винятки або нестандартні відповіді;
- контекстний аналіз, враховується зв'язок між запитом і відповіддю, щоб оцінити, чи є реакція сервера типовою для такого запиту;

– класифікація ризиків, модель оцінює рівень загрози, класифікуючи виявлені проблеми за ступенем серйозності – високий, середній або низький. Це дозволяє фахівцям визначити пріоритетність для подальшого детального аналізу.

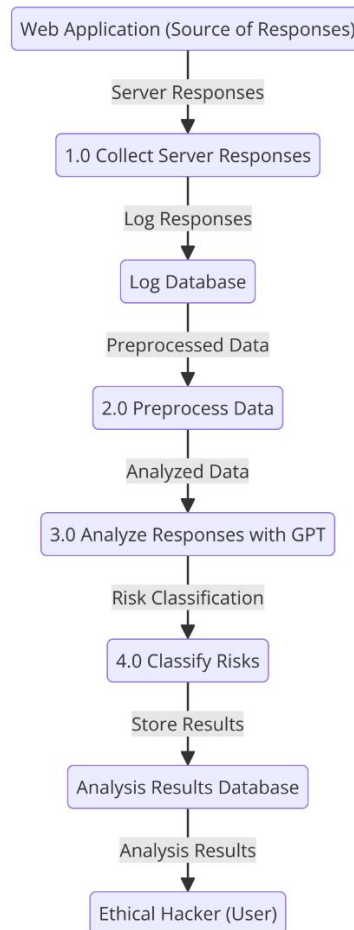


Рисунок 3.3 - Поток даних автоматизованого аналізу відповідей

На Рисунок 3.3 представлена схема потоків даних, яка ілюструє автоматизований процес аналізу відповідей вебзастосунків. Вона демонструє, як інформація проходить через основні етапи: від збору відповідей до попередньої обробки, аналізу за допомогою GPT і остаточної класифікації ризиків. Схема відображає передачу даних між кожним етапом і використання відповідних сховищ для збереження проміжних результатів.

Процес автоматизованого аналізу відповідей вебзастосунків із застосуванням GPT забезпечує високу ефективність і точність у виявленні вразливостей. Завдяки інтеграції методів обробки даних, контекстного аналізу та класифікації ризиків, фахівці отримують не лише інструмент для пошуку проблем, але й зрозумілий спосіб пріоритезувати подальші кроки з підвищення безпеки систем.

3.3 Побудова сценаріїв тестування та роль мовних моделей

На основі результатів аналізу серверних відповідей система створює комплексні сценарії пентестингу, які спрямовані на виявлення складних і прихованих вразливостей. Це дозволяє перевірити не лише конкретні слабкі місця, а й стійкість системи до комбінованих атак, які можуть обходити традиційні механізми захисту.

Модель GPT аналізує результати попередніх запитів, виокремлюючи патерни у відповідях, що можуть вказувати на потенційні проблеми. Якщо певний тип запиту викликає нестандартну реакцію сервера, це розглядається як можливий індикатор прихованих вразливостей. На основі таких патернів створюються складні послідовності запитів, що поєднують різні види атак, наприклад, SQL-ін'єкції, XSS чи CSRF. Головна мета – перевірити, чи може комбінація атак обійти наявні захисні механізми.

Крім того, система застосовує техніки обфускації та модифікації запитів, щоб обійти серверні фільтри та валідатори. Це може включати зміну структури запиту, кодування шкідливого вмісту або зміни у параметрах, що дозволяє уникнути стандартних засобів виявлення атак. Послідовність дій адаптується в залежності від реакції системи, що створює динамічну модель тестування.

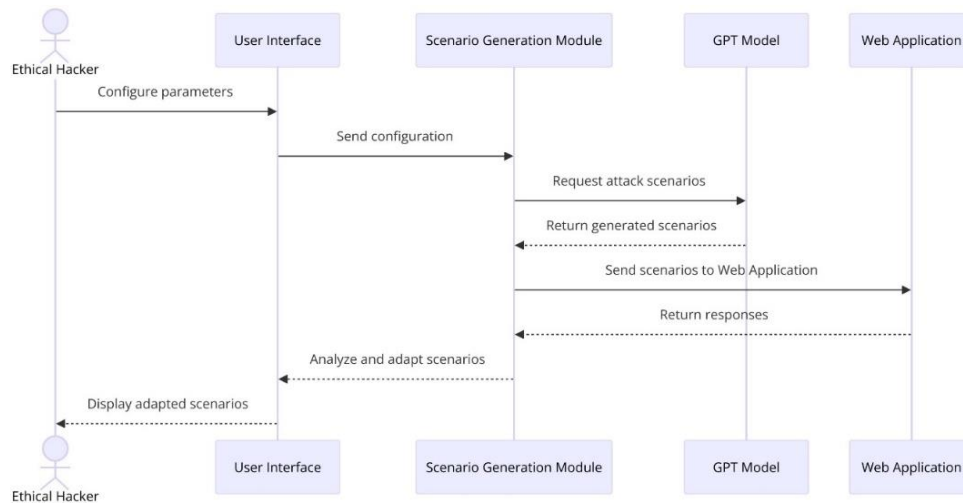


Рисунок 3.3 - Послідовна діаграма створення сценаріїв тестування

На Рисунку 3.3 представлена діаграма послідовності, яка відображає взаємодію між етичним хакером, інтерфейсом користувача, модулем генерації сценаріїв, GPT-моделлю та вебдодатком. Вона демонструє кроки, які здійснюються для створення тестових сценаріїв, починаючи з аналізу даних і завершуючи виконанням комплексних атак.

Великі мовні моделі, такі як ShellGPT, стають невід’ємною частиною процесу тестування безпеки, виконуючи роль помічників у складних завданнях. Вони дозволяють автоматизувати рутинні операції, зменшувати час на виконання тестів і підвищувати точність аналізу. Інтеграція мовної моделі в середовище командного рядка забезпечує пентестерам можливість працювати природною мовою, позбавляючи їх необхідності запам’ятовувати складні команди.

Наприклад, за допомогою ShellGPT можна швидко виконувати мережеві сканування, формувати HTTP-запити або аналізувати відповіді серверів. Інструмент також здатний генерувати payload-и для тестування вразливостей, таких як SQL-ін’єкції чи XSS. У поєднанні з інструментами на кшталт Burp Suite або OWASP ZAP, ShellGPT може автоматизувати створення звітів, формувати технічні рекомендації та пропонувати подальші дії.

Інструмент спрощує роботу як для досвідчених пентестерів, так і для новачків, створюючи зручний і зрозумілий інтерфейс. Крім автоматизації, ShellGPT інтегрує функцію зворотного зв'язку, що дозволяє уточнювати або модифікувати запити залежно від потреб. Це робить мовну модель потужним асистентом для етичних хакерів, який значно полегшує тестування, підвищує його ефективність і забезпечує надійність результатів.

Застосування GPT-моделей у тестуванні безпеки сприяє не лише автоматизації процесів, а й створенню складних сценаріїв, які дозволяють виявляти навіть приховані вразливості. Інтеграція таких інструментів у роботу пентестерів підвищує ефективність і точність, роблячи процес тестування більш структурованим і надійним.

3.4 Розробка власного інструменту для пошуку веб вразливостей на основі API ChatGPT

Інтеграція API ChatGPT у сучасні інструменти для забезпечення кібербезпеки відкриває широкий спектр можливостей, особливо у сфері автоматизації процесів пошуку вебвразливостей. Використання штучного інтелекту дозволяє вийти за межі стандартних підходів до аналізу даних, впроваджуючи глибокий контекстуальний аналіз, динамічне навчання та адаптивність, які значно розширюють межі традиційних технологій [61].

API ChatGPT надає можливість створювати рішення, які поєднують простоту використання та гнучкість у реалізації складних завдань. Це досягається завдяки тому, що модель здатна виконувати різноманітні завдання на основі текстових запитів. Уявімо додаток, де користувач вводить у вебінтерфейсі дані, наприклад, HTTP-заголовки або конфігурацію сервера, після чого система передає ці дані в API для глибокого аналізу. Штучний інтелект не лише визначає потенційні вразливості, але й пропонує рекомендації щодо їх усунення.

Додаток, заснований на API ChatGPT, будується як вебінтерфейс, орієнтований на аналіз даних та генерацію рекомендацій у сфері кібербезпеки. Головною ідеєю є створення інтерактивного середовища, яке дозволяє тестувати вразливості вебдодатків або серверів. Інструмент автоматизує процес обробки вхідних даних, надаючи користувачам чіткі звіти про можливі проблеми та способи їх усунення.

Модель передбачає два основних компоненти:

- серверна частина, яка відповідає за інтеграцію з API та обробку даних;
- клієнтська частина у вигляді простого вебінтерфейсу, який забезпечує зручність взаємодії для користувача.

Схема процесів роботи додатку з інтеграцією API ChatGPT:

- вхідні дані користувача, користувач вводить дані, наприклад, HTTP-заголовки або конфігурації сервера, через вебінтерфейс;
- сервер отримує введені дані, перевіряє їх коректність і передає для аналізу;
- дані надсилаються в API ChatGPT, яке виконує аналіз, визначає потенційні вразливості та формулює рекомендації;
- обробка та аналіз, результати, отримані від API, додатково структуруються та готуються для відображення;
- результати, сервер формує фінальний звіт, який містить рекомендації, висновки та виявлені вразливості;
- виведення результатів, користувач отримує оброблену інформацію через інтерфейс у вигляді звіту або структурованого тексту.

Архітектура такого додатку може бути відносно простою, але ефективною. Вона будується на взаємодії між клієнтським інтерфейсом, сервером і самим API. Користувач завантажує дані або вводить їх у текстовому полі, сервер обробляє запит і передає його до API ChatGPT. Модель проводить аналіз, повертає висновки, які потім структуруються та відображаються у вигляді звітів або рекомендацій. Такий підхід дозволяє створювати

багатофункціональні інструменти, які спрощують і прискорюють виконання рутинних завдань.

Застосування подібних інструментів має величезний потенціал не лише з точки зору автоматизації, а й з наукової точки зору. Уявімо, що ChatGPT може використовуватись для аналізу великих масивів даних у реальному часі. Наприклад, система може одночасно аналізувати лог-файли, виявляючи незвичні патерни, та давати рекомендації щодо захисту вразливих компонентів. Цей підхід дозволяє зосередитися на проактивному підході до безпеки, де можливі атаки запобігаються ще до їх реалізації.

Інтеграція API у кібербезпекові рішення також пропонує величезний потенціал для адаптації до конкретних умов. Завдяки механізмам донавчання модель може швидко оновлювати свої знання, що дозволяє їй враховувати сучасні загрози, які постійно еволюціонують. Крім того, API ChatGPT є універсальним інструментом, який легко масштабується. Це означає, що система може бути ефективною як для індивідуальних користувачів, так і для великих організацій з багатокористувацьким середовищем.

Важливо також зазначити, що такі інструменти можуть виконувати не лише практичні функції, а й освітню роль. Наприклад, система може пояснювати результати аналізу або надавати рекомендації у вигляді детальних інструкцій, що робить її корисною як для досвідчених спеціалістів, так і для новачків у сфері кібербезпеки.

Говорячи про розвиток подібних рішень, можна уявити, як у майбутньому інструменти на основі API ChatGPT стануть ключовими елементами комплексних платформ безпеки. Вони зможуть інтегруватися з існуючими системами моніторингу, такими як SIEM, або працювати у режимі реального часу, забезпечуючи безперервний захист. Подальший розвиток моделей може включати їх здатність аналізувати не лише текстові дані, але й візуальну інформацію, наприклад, діаграми мереж або знімки екранів. Це зробить їх ще більш універсальними та ефективними у боротьбі з новими типами загроз.

У підсумку, впровадження API ChatGPT у вебдодатки для аналізу безпеки демонструє, як штучний інтелект може не лише змінити підходи до кіберзахисту, а й стати ключовим компонентом у створенні інноваційних рішень. Це не просто інструмент для автоматизації, а технологія, яка допомагає розширити межі можливого у сфері захисту інформації.

3.5 Перспективи вдосконалення розробленого методу

Розроблений метод пошуку вебвразливостей із використанням великих мовних моделей (LLM), таких як API ChatGPT, є інноваційним кроком у забезпеченні кібербезпеки. Однак для розкриття його повного потенціалу існує ряд напрямків для вдосконалення. Вони охоплюють технологічні, наукові та практичні аспекти, спрямовані на розширення функціональних можливостей, підвищення точності аналізу та адаптацію до нових викликів [62].

Одним із ключових напрямків розвитку є використання багатомодальних моделей, які можуть працювати не лише з текстовими даними, а й з іншими типами інформації, такими як зображення, діаграми або мережеві графіки. Наприклад, така модель може аналізувати:

- топологію мережі у вигляді графічної діаграми для виявлення слабких місць;
- логи серверів або скріншоти результатів роботи пентестингових інструментів для швидкої оцінки загроз.

Це відкриває можливість більш глибокого аналізу даних і розширює спектр задач, які можуть виконуватися автоматично.

Ще одним перспективним напрямком є впровадження систем проактивного моніторингу. Замість аналізу статичних даних, таких як конфігурації чи журнали подій, модель може працювати в режимі реального часу, реагуючи на зміну поведінки системи або аномалії в мережевому трафіку. Наприклад:

- виявлення раптового збільшення обсягу запитів до сервера, що може свідчити про DDoS-атаку;
- відстеження змін у HTTP-заголовках, які можуть свідчити про спробу обходу систем автентифікації.

Такий підхід дозволить запобігати атакам ще до їх реалізації.

Для підвищення точності аналізу модель можна додатково навчати на специфічних наборах даних, орієнтованих на конкретні типи вразливостей чи галузі. Наприклад:

- донавчання на основі баз даних CVE, OWASP Top 10 або інших джерел знань про кіберзагрози;
- інтеграція корпоративних журналів подій для адаптації до особливостей конкретного середовища.

Це дозволить підвищити адаптивність моделі та її релевантність до сучасних викликів.

Інтеграція декількох спеціалізованих моделей в одному інструменті може значно розширити його функціональність. Наприклад:

- одна модель може відповідати за аналіз HTTP-заголовків, інша - за сканування мережі;
- додаткові модулі можуть виконувати моделювання атак або прогнозування нових загроз.

Це дозволить створити цілісну платформу, яка автоматизує всі етапи пентестингу - від виявлення до усунення вразливостей.

Подальший розвиток методології передбачає більш глибоку інтеграцію з популярними інструментами кібербезпеки, такими як Burp Suite, OWASP ZAP або Nmap. Це дозволить:

- автоматизувати обробку результатів цих інструментів за допомогою API ChatGPT;
- розширити можливості аналізу та створення звітів.

Крім того, інтеграція з системами моніторингу безпеки (наприклад, SIEM) допоможе забезпечити комплексний захист на всіх рівнях.

Для забезпечення зручності використання майбутні розробки можуть включати:

- більш інтуїтивний інтерфейс із інтерактивними підказками;
- можливість голосового введення даних або створення сценаріїв через природну мову;
- генерацію навчальних матеріалів на основі результатів аналізу, що буде корисно для новачків у кібербезпеці.

Такі вдосконалення дозволять залучити ширшу аудиторію та підвищити доступність інструменту.

Ще одним важливим аспектом є масштабування системи для використання в великих інфраструктурах. Це включає:

- можливість одночасного аналізу декількох джерел даних;
- оптимізацію використання обчислювальних ресурсів для забезпечення високої швидкості роботи навіть у складних мережевих середовищах.

Подальший розвиток розробленого методу на основі API ChatGPT відкриває численні перспективи. Використання багатомодальних моделей, інтеграція проактивного моніторингу, донавчання на специфічних наборах даних та комбінування різних моделей забезпечують можливість створення універсального інструменту кібербезпеки. Завдяки цим вдосконаленням інструмент може стати не лише ефективним рішенням для виявлення вразливостей, а й основою для побудови комплексних систем захисту.

Ці перспективи підкреслюють, як штучний інтелект може трансформувати кібербезпеку, роблячи її більш адаптивною, ефективною та доступною для широкого кола користувачів.

3.6 Висновок

У третьому розділі було розглянуто розробку методу пошуку вебвразливостей на основі штучного інтелекту, що інтегрує можливості

великих мовних моделей (LLM). Запропонований метод базується на взаємодії між мовною моделлю та користувачем, автоматизації аналізу даних і формуванні рекомендацій. Особливий акцент було зроблено на реалізації функціональних компонентів, таких як аналіз HTTP-заголовків і сканування мережі за допомогою API ChatGPT.

Запропонована модель інструменту демонструє ефективність використання ШІ у задачах кібербезпеки. Інтерактивна інженерія підказок дозволяє автоматизувати пошук вразливостей, створювати payload-и для перевірки систем та аналізувати результати. Окрім цього, інтеграція API ChatGPT у додатки надає можливість динамічного тестування та генерації релевантних звітів, що значно спрощує роботу етичних хакерів.

Перспективи вдосконалення методу включають інтеграцію багатомодальних моделей, впровадження проактивного моніторингу, донавчання на специфічних наборах даних і масштабування для великих інфраструктур. Розвиток інструменту також передбачає створення універсальної платформи, яка може об'єднувати аналіз різних типів даних, інтегруватися з іншими інструментами кібербезпеки та забезпечувати проактивний захист.

У підсумку, запропонований підхід демонструє, як штучний інтелект може стати невід'ємною частиною сучасних рішень у сфері кібербезпеки. Завдяки гнучкості, адаптивності та точності він забезпечує нові можливості для пошуку вразливостей, підвищуючи ефективність як для досвідчених фахівців, так і для новачків. Розроблений метод закладає основу для подальших досліджень і впроваджень у сфері автоматизації аналізу безпеки вебдодатків.

4 ТЕСТУВАННЯ РОЗРОБЛЕНОГО МЕТОДУ ДЛЯ ПОШУКУ ВРАЗЛИВОСТЕЙ

4.1 Архітектура розробленого інструменту

Розроблений інструмент поєднує можливості аналізу HTTP заголовків і сканування мережі з використанням автоматичного аналізу за допомогою штучного інтелекту (ChatGPT API). У цьому розділі описана архітектура та компоненти додатку, що забезпечують інтерактивність з користувачем, а також автоматичний аналіз безпеки HTTP заголовків і мережевих вразливостей. Проєкт розроблено з використанням вебтехнологій, Python і бібліотеки Flask для взаємодії з користувачем (рис. 4.1).

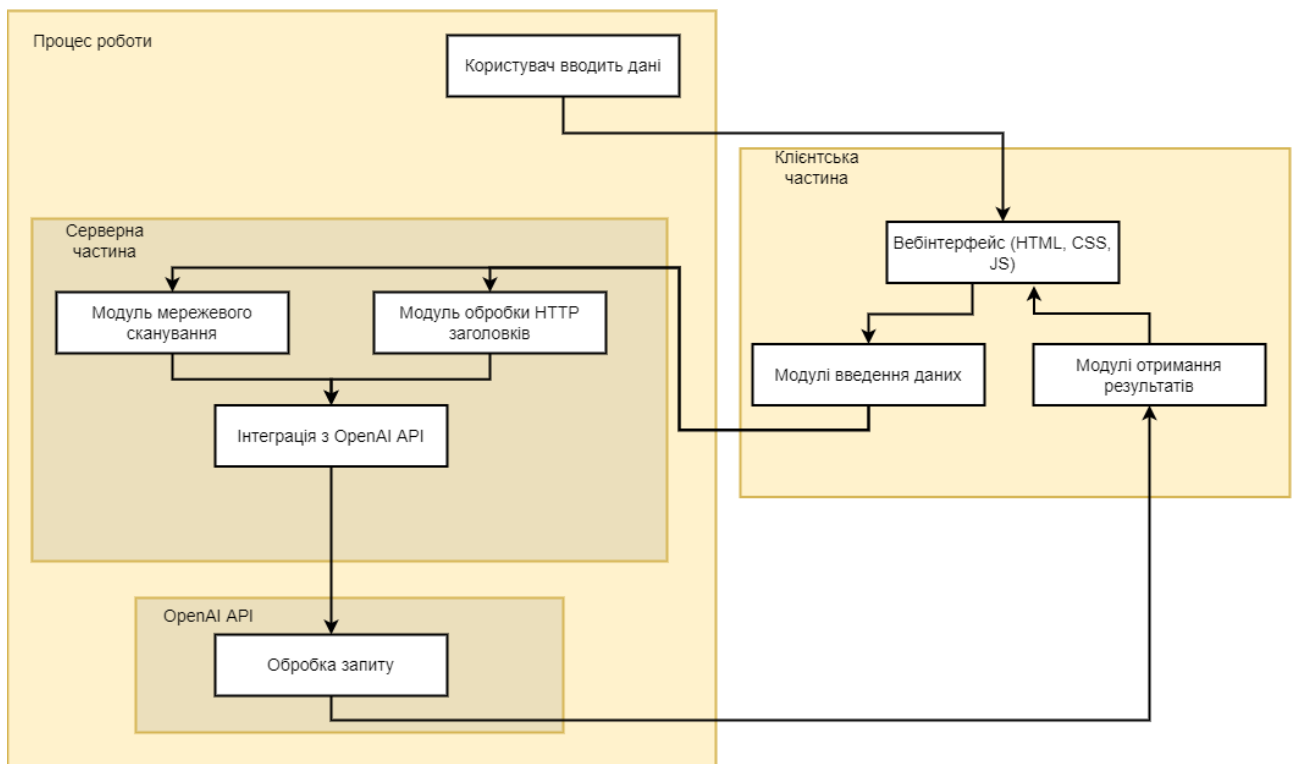


Рисунок 4.1 - Архітектурна схема вебдодатку

Клієнтська частина реалізована у вигляді вебінтерфейсу, де користувач може:

- ввести або завантажити HTTP заголовки для аналізу;

- ввести цільову IP-адресу або домен для сканування;
- обрати тип сканування, додаткові параметри та скрипти Nmap для точного аналізу;
- отримати результати аналізу у вигляді рекомендацій щодо покращення безпеки.

Інтерфейс реалізовано з використанням HTML для структури, CSS для стилізації та JavaScript для логіки взаємодії. JavaScript відповідає за надсилання запитів до серверної частини через API, а також за динамічне відображення результатів.

```

from flask import Flask, request, jsonify
import openai

app = Flask(__name__)

@app.route('/analyze_headers', methods=['POST'])
def analyze_headers():
    headers = request.json.get('headers')
    if not headers:
        return jsonify({"error": "Порожні заголовки"}), 400

    # Формування запиту до OpenAI API
    response = openai.Completion.create(
        model="gpt-4",
        prompt=f"Аналізуйте ці HTTP заголовки на наявність вразливостей: {headers}",
        temperature=0.5
    )

    analysis = response.choices[0].text.strip()
    return jsonify({"analysis": analysis})

if __name__ == '__main__':
    app.run(debug=True)

```

Рисунок 4.2 - Приклад коду серверної частини для обробки HTTP заголовків

Серверна частина побудована на Flask (Python). Вона обробляє як HTTP заголовки, так і параметри для мережевого сканування, використовуючи бібліотеку nmap для виконання мережевого аналізу. Основні етапи роботи серверної частини:

- отримання HTTP заголовків або параметрів для сканування від клієнта;
- надсилання HTTP заголовків на аналіз до AI через ChatGPT API;
- виконання сканування мережі з використанням nmap і обробка результатів;
- форматування результатів для передачі на клієнт.

Для підвищення ефективності аналізу, заголовки і результати сканування автоматично надсилаються на аналіз через OpenAI API. Це дозволяє отримувати більш детальні коментарі та рекомендації щодо знайдених вразливостей, що полегшує роботу спеціалістів з кібербезпеки. OpenAI API аналізує отримані дані і надає висновки про можливі ризики, типи атак, які можуть бути застосовані, і рекомендації для покращення безпеки.

```
import nmap

@app.route('/network_scan', methods=['POST'])
def network_scan():
    data = request.json
    target = data.get('target')
    options = data.get('options')

    try:
        nm = nmap.PortScanner()
        nm.scan(target, arguments=options)

        # Формування результатів сканування
        results = []
        for host in nm.all_hosts():
            for proto in nm[host].all_protocols():
                ports = nm[host][proto].keys()
                for port in ports:
                    service_info = nm[host][proto][port]
                    results.append({
                        "host": host,
                        "protocol": proto,
                        "port": port,
                        "state": service_info['state'],
                        "version": service_info.get('product', '') + " " +
                            service_info.get('version', '')
                    })

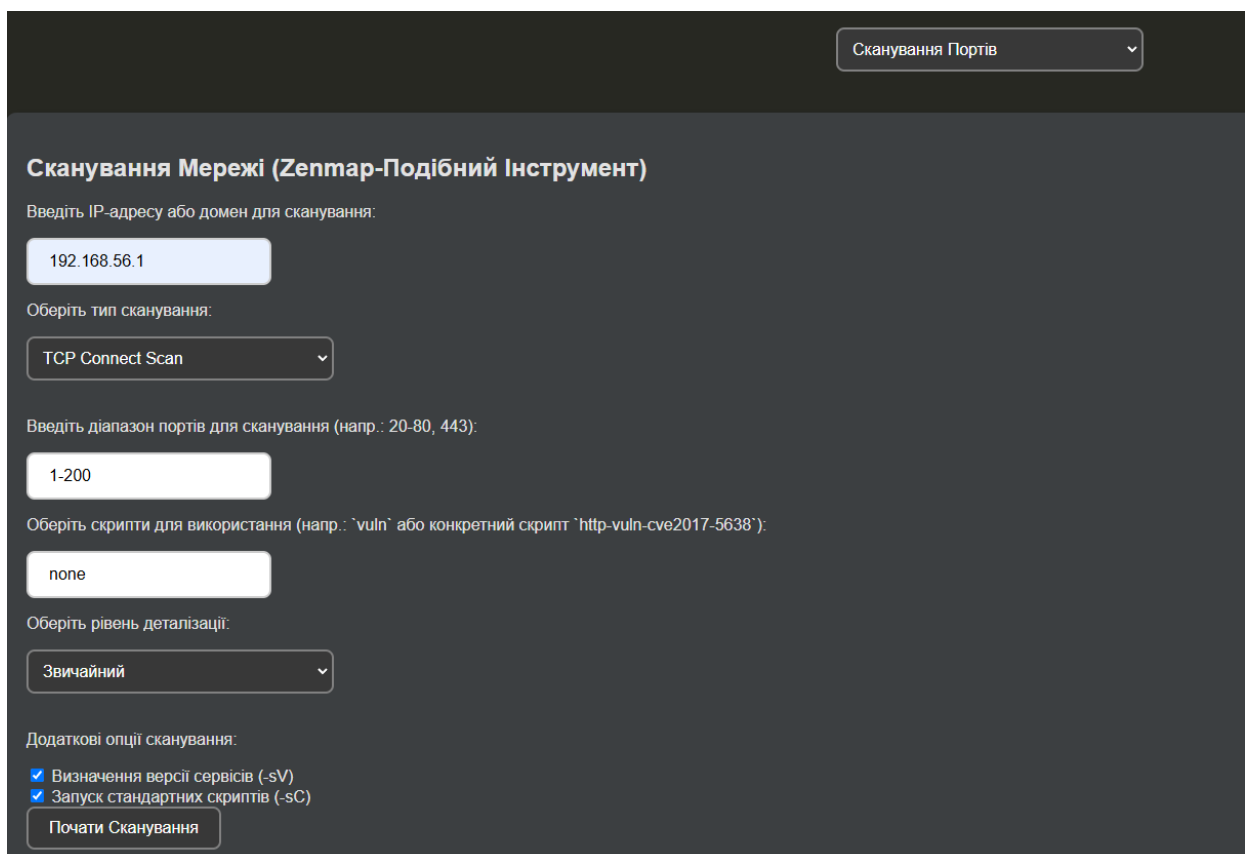
        return jsonify({"scan_results": results})
    except Exception as e:
        return jsonify({"error": str(e)}), 500
```

Рисунок 4.3 - Приклад коду для сканування мережі з використанням Nmap

На початковій сторінці користувачу пропонується випадючий список з доступними інструментами. Це забезпечує зручний вибір між різними функціями, такими як аналіз HTTP-заголовків, сканування мережі або аналіз вразливостей.

Користувач вводить HTTP заголовки або завантажує їх у форму. Після цього заголовки відправляються на сервер для аналізу. Серверна частина використовує OpenAI API для аналізу наявності критичних вразливостей, таких як відсутність важливих заголовків (Content-Security-Policy, Strict-Transport-Security) або неправильна конфігурація (X-Frame-Options, X-XSS-Protection). Результати повертаються у вигляді текстових рекомендацій.

Користувач вводить IP-адресу або домен, вибирає тип сканування та додаткові параметри, такі як -sV (визначення версії сервісів) або -sC (виконання стандартних NSE скриптів). Серверна частина виконує сканування за допомогою бібліотеки Nmap і повертає результати користувачу у зручному форматі.



The screenshot shows a web application interface for network scanning. At the top right, there is a dropdown menu labeled "Сканування Портів". The main heading is "Сканування Мережі (Zenmap-Подібний Інструмент)". Below the heading, there are several input fields and dropdown menus:

- A text input field for "Введіть IP-адресу або домен для сканування:" containing "192.168.56.1".
- A dropdown menu for "Оберіть тип сканування:" with "TCP Connect Scan" selected.
- A text input field for "Введіть діапазон портів для сканування (напр.: 20-80, 443):" containing "1-200".
- A text input field for "Оберіть скрипти для використання (напр.: 'vuln' або конкретний скрипт 'http-vuln-cve2017-5638'):" containing "none".
- A dropdown menu for "Оберіть рівень деталізації:" with "Звичайний" selected.
- A section for "Додаткові опції сканування:" with two checked checkboxes: "Визначення версії сервісів (-sV)" and "Запуск стандартних скриптів (-sC)".
- A "Почати Сканування" button at the bottom.

Рисунок 4.4 - Інтерфейс застосунку

Технології та Інструменти

- мови програмування, Python (сервер), HTML/CSS/JavaScript (вебінтерфейс);
- фреймворки, Flask для серверної частини;
- бібліотеки, OpenAI API для аналізу, nmap для сканування мережі.

Архітектура інструменту забезпечує гнучкість і масштабованість для ефективного використання в задачах кібербезпеки. Користувач може аналізувати HTTP заголовки або виконувати сканування мережі для виявлення потенційних вразливостей. Інтеграція з ChatGPT API значно підвищує ефективність аналізу, дозволяючи отримувати релевантні висновки про знайдені вразливості та пропонувати рекомендації з покращення безпеки.

4.2 Підготовка середовища тестування та інтеграція ШІ

Oracle VM VirtualBox, гіпервізор другого типу, що працює поверх операційної системи Windows, був обраний для цього експерименту як програмне забезпечення для віртуалізації. Він дозволив створити ізольоване середовище для роботи з попередньо налаштованою віртуальною машиною Kali Linux, яка була завантажена з офіційного сайту kali.org. Цей вибір обґрунтовано тим, що VirtualBox є зручним і широко використовуваним інструментом, який підтримує безліч операційних систем і забезпечує стабільну роботу навіть на машинах із помірними ресурсами. Використання VirtualBox також дозволяє легко переносити налаштування між системами, що забезпечує гнучкість і доступність для тестування в різних умовах.

Kali Linux, операційна система з відкритим кодом, заснована на Debian, широко відома серед професіоналів у сфері кібербезпеки. Вона здобула популярність завдяки вбудованому набору інструментів, що включає потужні засоби для тестування на проникнення, аналізу вразливостей, реверс-інженерії та багато іншого. Ця ОС активно використовується пентестерами,

дослідниками безпеки, аналітиками та розробниками, які працюють із виявленням і запобіганням кіберзагрозам. У цьому експерименті Kali Linux виконувала роль операційної системи пентестера, яку в рамках дослідження називали локальною машиною. Це середовище дозволило не тільки розгорнути необхідні інструменти, але й надати користувачу повний контроль над процесами, що виконуються.

Цільова машина, на якій проводилося тестування, була випадково обрана з VulnHub, репозиторію, що спеціалізується на віртуальних машинах для навчання. VulnHub пропонує широкий спектр машин із різними рівнями складності, що дозволяє новачкам і професіоналам практикувати навички в безпечному середовищі. У межах цього дослідження було обрано віртуальну машину «PumpkinFestival», яка є фінальним рівнем серії Mission-Pumpkin, створеної Jayanth і випущеної 17 липня 2019 року. «PumpkinFestival» відзначається різноманітними вразливостями, що моделюють реальні сценарії атак, зокрема слабкі місця в конфігураціях, неправильно налаштовані служби та вразливий код. Метою експерименту було досягнення root-доступу до системи, під час якого учасник збирав «флаги», які слугували маркерами успішного проходження етапів.

Концепція Capture the Flag (CTF), або «Захоплення прапора», є популярним форматом змагань у галузі кібербезпеки, спрямованим на перевірку знань і навичок учасників. Витоки цієї концепції беруть початок із традиційної гри, де команди мають захопити прапор супротивника, що може бути захищений чи захищений, і повернути його на свою базу. У кібербезпеці цей підхід адаптувався для імітації реальних сценаріїв атак і захисту, дозволяючи учасникам розвивати свої технічні здібності та стратегічне мислення. У змаганнях CTF часто використовуються віртуальні машини, подібні до «PumpkinFestival», які містять комплексні вразливості, що вимагають від учасників застосування різних методів і тактик.

Ключовим етапом підготовки до експерименту стало інтегрування API ChatGPT у середовище пентестингу. З цією метою було використано Shell_GPT

(sgpt), інструмент командного рядка на основі Python, що дозволяє виконувати shell-команди, генерувати скрипти та отримувати рекомендації від ChatGPT у реальному часі. Використання sgpt забезпечило інтеграцію ChatGPT із популярними пентестинговими інструментами, такими як Nmap, Nessus і OpenVAS. Це дозволило автоматизувати взаємодію з цими інструментами, створюючи прямий інтерфейс для виконання сканувань, збору та аналізу результатів.

Інтеграція ChatGPT у CLI-середовище стала значним проривом у роботі пентестерів. На відміну від звичайного використання ChatGPT через вебінтерфейс, інтеграція API дозволяє автоматизувати процеси та зменшити кількість рутинних операцій. Наприклад, під час виконання сканувань ChatGPT миттєво інтерпретував результати та пропонував подальші кроки, що суттєво скорочувало час на ручний аналіз. Завдяки прямій взаємодії зі згенерованими даними ChatGPT міг забезпечити точніші й релевантні рекомендації, що підвищувало загальну ефективність роботи.

Експеримент мав на меті оцінити здатність ChatGPT працювати з даними, отриманими з професійних пентестингових інструментів, і надавати корисні рекомендації для подальших дій. Shell_GPT, виступаючи інтерфейсом до ChatGPT через API, продемонстрував свою практичну цінність у створенні інтегрованого середовища, яке спрощує робочі процеси та автоматизує рутинні завдання.

Експеримент проводився через CLI, використовуючи можливості sgpt для виконання shell-команд і запитів до ChatGPT. Автор не мав попереднього досвіду роботи з конкретною віртуальною машиною, що обиралася для експерименту, однак володів знаннями про аналогічні системи. Основною ідеєю було імітувати сценарій, у якому новачок у пентестингу застосовує ChatGPT для оптимізації своєї роботи, не маючи ще достатнього досвіду та знань для самостійного використання складних команд.

Однією з проблем у використанні ChatGPT є обмеження, запроваджені OpenAI. Відповідно до політики компанії, ChatGPT не генерує матеріали, які

можуть сприяти незаконній діяльності, наприклад, створення шкідливого ПЗ. Це обмеження охоплює навіть легітимні запити, такі як пентестинг, що може бути неправильно сприйнятий як зловмисна діяльність. Наприклад, на запит “список IP-адрес tesla.com” ChatGPT відповідає загальною інструкцією, як можна отримати такі дані, уникаючи надання прямої відповіді.

Деякі користувачі намагаються обходити ці обмеження через “jail breaking” - активацію режимів, таких як “DAN” (Do Anything Now) або “Developer Mode”. Ці методи дозволяють ChatGPT ігнорувати свої етичні бар'єри та надавати інформацію, яка зазвичай вважається неприйнятною. Використання таких підходів, хоча й може забезпечити доступ до недоступної інформації, порушує політику OpenAI і викликає серйозні етичні питання, особливо у сфері досліджень кібербезпеки. Незважаючи на їхню ефективність, вони залишаються суперечливими з погляду моралі та правомірності.

4.3 Тестування запропонованих методів

Для розуміння того як модель взаємодіє з системою я спробую визначити IP-адресу мого пристрою. Для цього використовую команду, згенеровану через ShellGPT.

```
(root@kali)-[~/home/kali]
└─# sgpt -s "find my local ip address"
ip addr show | grep 'inet ' | grep -v '127.0
.0.1' | awk '{print $2}' | cut -d '/' -f1
[E]xecute, [D]escribe, [A]bort: e
192.168.56.26
```

Рисунок 4.5 - Початкова взаємодія з shellGPT

Знання адреси Інтернет-протоколу та визначення мережевої інтерфейсної карти, яка використовується, - це дві частини інформації, необхідної для подальших етапів пентестування. Визначення того, чи проводиться пентест всередині або поза цільовою мережею, має вирішальне значення для вибору

відповідних методів сканування і атаки. Розрізнення локальної IP-адреси в журналах і мережевого трафіку дозволяє легше інтерпретувати результати.

Після перерахування активних хостів в цільовій мережі, я визначив потенційну ціль, та зрозумів структуру мережі, що допоможе мені у подальшому плануванні векторів атаки.

Далі необхідно виявити відкриті служби, порти та операційну систему, що працює на виявленій цілі, це допоможе визначити потенційні вразливості для використання.

```
(root@kali)-[~/home/kali]
└─# sgpt -s "scan 192.168.56.140 on all ports to find out OS, vulnerabilities"
nmap -p- -A --script vuln 192.168.56.140
[Execute] [Describe] [Alert] [Alert]
```

Рисунок 4.6 - Сканування цілі

Після успішного сканування (рис. 4.5), виявлення відкритих сервісів, портів і запущеної ОС, а також їхніх версій, результат можуть допомогти визначити потенційні точки входу і вразливості, які можна використати на етапі експлуатації.

```
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 2.0.8 or later
80/tcp    open  http    Apache httpd 2.4.7 ((Ubuntu))
|_http-server-header: Apache/2.4.7 (Ubuntu)
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-slowloris-check:
|   VULNERABLE:
|     Slowloris DOS attack
|       State: LIKELY VULNERABLE
|       IDs: CVE:CVE-2007-6750
|         Slowloris tries to keep many connections to the target web
server open and hold
|         them open as long as possible. It accomplishes this by opening
connections to
|         the target web server and sending a partial request. By doing
so, it starves
|         the http server's resources causing Denial Of Service.
|
|       Disclosure date: 2009-09-17
|       References:
|         https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-675
0
|_       http://hackers.org/slowloris/
```

Рисунок 4.7 - Результат сканування

Отже, за результатами сканування, було визначено відкритий порт 21, та http server на порті 80.

За запитом, модель створила скрипт для щоб спробувати анонімно увійти на FTP-сервер та отримати список і завантажити усі можливі файли, перевірявши конфігурації слабкої безпеки (рис. 4.6).

```
(root@kali)-[~/home/kali/Desktop/1]
└─# sgpt -s "login anonymously to ftp 192.168.56.140, та напиши скрипт який буде перевіряти існування папок заходити в них і скачувати файли з них"

ftp -inv 192.168.56.140 <<EOF
user anonymous
binary
mls * dirs.txt
EOF

while read dir; do
  ftp -inv 192.168.56.140 <<EOF
  user anonymous
  binary
  cd $dir
  mget *
  bye
EOF
done < dirs.txt
```

Рисунок 4.8 - Скрипт пошуку директорій та завантаження файлів

Після успішного анонімного входу на FTP-сервер, скрипт знайшов директорію secrets та завантажив з неї вміст. У файлі були знайдені та ідентифіковані можливі користувачі сайту.

Інша служба, що працює на віддаленій машині - це вебсервер Apache на порту 80, який обслуговує вебсторінку. Було необхідно перевірити вихідний код вебсторінки, щоб зібрати додаткову інформацію. Після аналізу коду було підтверджено наявність раніше знайдених користувачів.

Далі з допомогою моделі було автоматизовано пошук можливих директорій на сайті.

```
(root@kali)-[~/home/kali/Desktop/1]
└─# sgpt -s "пошукай директорії на сайті 192.168.56.140"

gobuster dir -u http://192.168.56.140 -w /usr/share/wordlists/dirb/common.txt
[E]xecute, [D]escribe, [A]bort: e
```

Рисунок 4.9 - Пошук директорій

```

/.htaccess      (Status: 403) [Size: 290]
/.hta          (Status: 403) [Size: 285]
/.htpasswd     (Status: 403) [Size: 290]
/img           (Status: 301) [Size: 313] [→ http://192.168.56.140/img/]
/index.html    (Status: 200) [Size: 1465]
/robots.txt    (Status: 200) [Size: 102]
/server-status (Status: 403) [Size: 294]
/store        (Status: 301) [Size: 315] [→ http://192.168.56.140/store/]
/users        (Status: 301) [Size: 315] [→ http://192.168.56.140/users/]
Progress: 4614 / 4615 (99.98%)

```

Рисунок 4.10 - Результат пошуку директорій

Після успішного пошуку, було виявлено файл robots.txt. Він використовується для контролю доступу веброботів (наприклад, пошукових систем) до різних частин вебсайту. Він містить директиви, які вказують, які частини сайту можуть бути індексовані та зкачувані.

Директива “Disallow: /store/track.txt”, яка знаходилась у файлі, забороняє веброботам індексувати та зкачувати файл /store/track.txt. Це може бути корисно для захисту файлів відстеження або інших чутливих даних.

```

(root@kali)-[~/home/kali/Desktop/1]
└─# sgpt -s "додай 192.168.56.140 до hosts як pumpkins.local"
echo "192.168.56.140 pumpkins.local" | sudo tee -a /etc/hosts

```

Рисунок 4.11 - редагування файлу hosts

З того самого файлу /store/track.txt було виявлено домен pumpkins.local , який потім потрібно було додати до файлу hosts (рис. 4.9), щоб браузер мав належний доступ до сайту WordPress.

Наступним кроком було здійснено сканування на потенційні вразливості, модель автоматично запускає необхідний інструмент з усіма необхідними параметрами.

```

(root@kali)-[~/home/kali/Desktop/1]
└─# sgpt -s "проскануй wordpress сайту http://pumpkins.local, результати збережив файл wpscan.txt"
wpscan --url http://pumpkins.local --output wpscan.txt

```

Рисунок 4.12 - Сканування сайту WordPress

Вміст wpscan.txt був переданий на сервер sgpt в режимі чату, який виводив результати аналізу з переліком вразливостей, що можуть бути використані в подальшому (рис. 4.11).

```
(root@kali)-[~/home/kali/Desktop/1]
└─# cat wpscan.txt | sgpt --chat pentest "знайди потенційні вразливості в результаті сканування wpscan"
Based on the WPScan results, here are potential vulnerabilities and issues identified:

1 Outdated WordPress Version: The site is running WordPress version 4.9.3, which is outdated and insecure. It is recommended to update to the latest version to patch known vulnerabilities.
2 XML-RPC Enabled: The XML-RPC interface is enabled, which can be exploited for DDoS attacks or brute force attacks. Consider disabling it if not needed.
3 Directory Listing Enabled: The uploads directory has listing enabled, which can expose sensitive files. Disable directory listing in the server configuration.
4 Registration Enabled: User registration is enabled, which could be exploited for spam or unauthorized access. Ensure proper security measures are in place.
5 Outdated Theme: The Twenty Seventeen theme is outdated. Update to the latest version to ensure security patches are applied.
6 No API Token: Without an API token, detailed vulnerability data is not available. Register for a free token to access this information.

Addressing these issues can significantly improve the security posture of the WordPress site.
```

Рисунок 4.13 - Аналіз результату сканування від ШІ

Під час наступних тестувань і досліджень веб застосунку, було знайдено та проаналізовано можливі вектори атаки. Також було знайдені закодоване повідомлення. З допомогою інтегрованого штучного інтелекту, одним запитом вдалось створити скрипт для декодування (рис. 4.12).

```
(root@kali)-[~/home/kali/Desktop/1]
└─# sgpt -s "напиши скрипт на пайтон для декодування base62 наступного хешу K82v0SuvV1En350M0uxiXVRTmBrQIJQN78s та збережи його як b62dec.py"

echo "import base64

def base62_decode(encoded_str):
    base62_chars = '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz'
    base62_map = {char: index for index, char in enumerate(base62_chars)}
    decoded_num = 0
    for char in encoded_str:
        decoded_num = decoded_num * 62 + base62_map[char]
    return decoded_num.to_bytes((decoded_num.bit_length() + 7) // 8, 'big')

encoded_hash = 'K82v0SuvV1En350M0uxiXVRTmBrQIJQN78s'
decoded_bytes = base62_decode(encoded_hash)
with open('decoded_output', 'wb') as f:
    f.write(decoded_bytes)" > b62dec.py
```

Рисунок 4.14 - Написання скрипту для декодування

Під час продовження тестування, було знайдено необхідні облікові данні, для отримання доступу до віртуальної машини. Етап підвищення привілеїв

вважається найскладнішим. Коли штучний інтелект підказує шляхи використання певної вразливості, це робить роботу досить легкою.

Щоб підвищити привілеї після входу в ssh як стандартний користувач, модель запропонувала наступні способи:

- перевірити, чи має користувач привілеї sudo, виконавши команду “sudo -l”. Якщо користувач має привілеї sudo, можна спробувати виконати команди за допомогою sudo, щоб підвищити привілеї;
- шукати будь-які неправильно налаштовані двійкові файли setuid, які можна використати для підвищення привілеїв. Можна скористатися командою “find / -perm -4000 2>/dev/null”, щоб знайти двійкові файли setuid;
- шукати будь-які доступні для запису каталоги або файли, які можна використати для підвищення привілеїв. Можна скористатися командою “find / -writable -type d 2>/dev/null”, щоб знайти каталоги, доступні для запису.

Я попросив sgpt перетворити серію команд, які він запропонував раніше, на “однорядковий код” (код, що займає лише один рядок, який колись вважався втіленням творчості великого програміста, а тепер перетворився на елементарну підказку). Цей останній крок демонструє здатність ШІ написати такий експлойт у стислій формі.

```
mkdir /home/jack/pumpkins &&
echo '/bin/bash' > /home/jack/
pumpkins/alohomora && chmod +x
/home/jack/pumpkins/alohomora
&& sudo /home/jack/pumpkins/
alohomora
```

Рисунок 4.15 - Написання скрипту для декодування

Після того, як створений експлойт був виконаний, активний користувач став root, і було досягнуто підвищення привілеїв.

4.4 Висновок

У цьому підрозділі було детально продемонстровано роль генеративного штучного інтелекту у процесах тестування на проникнення, зокрема його інтеграцію у сучасні інструменти кібербезпеки. Використання великих мовних моделей, таких як ChatGPT, дозволяє значно підвищити ефективність виявлення вебвразливостей завдяки їх здатності аналізувати великі обсяги даних, виявляти приховані патерни та автоматизувати рутинні процеси. Результати досліджень підтвердили, що моделі GPT сприяють оптимізації пентестингу, скорочуючи час на виконання тестів, підвищуючи точність виявлення вразливостей і знижуючи залежність від людського фактору.

Інтеграція штучного інтелекту у пентестинг відкриває нові можливості для створення інноваційних підходів до забезпечення кібербезпеки. Генеративні моделі демонструють здатність до адаптації в умовах постійно змінюваних загроз, забезпечуючи швидке реагування на нові типи атак, включаючи атаки нульового дня. Їх застосування дозволяє розширити традиційні методи тестування, додаючи до них можливість автоматизованого формування складних сценаріїв тестування, комбінування різних типів атак і аналізу поведінки систем у реальному часі.

Важливим аспектом є те, що технології штучного інтелекту, зокрема великі мовні моделі, можуть забезпечити підтримку фахівців із кібербезпеки у навчанні та вдосконаленні їхніх навичок. Інтуїтивно зрозумілі інструменти, побудовані на основі ШІ, спрощують доступ до складних технологій і сприяють підвищенню загального рівня безпеки у різних галузях.

Проте, впровадження таких технологій потребує відповідального підходу. Необхідно враховувати ризики, пов'язані з можливими хибними результатами, які можуть призводити до пропуску вразливостей або, навпаки, хибних спрацювань, що відволікають ресурси. Людський контроль залишається ключовим фактором у забезпеченні точності роботи моделей і запобіганні можливим етичним зловживанням, таким як використання ШІ для створення

небезпечних інструментів чи атак. Важливо також забезпечити прозорість у роботі систем на основі штучного інтелекту, щоб уникнути недовіри з боку користувачів і розробників.

Таким чином, інтеграція штучного інтелекту у процеси кібербезпеки є багатообіцяючим напрямком, який поєднує автоматизацію, високу продуктивність і адаптивність із можливостями роботи з великими обсягами даних у реальному часі. Розглянуті можливості та обмеження підтверджують необхідність подальших досліджень у цьому напрямку. Це дозволить створювати більш безпечні, гнучкі та адаптивні системи, які здатні протистояти сучасним викликам у сфері кіберзагроз, одночасно сприяючи сталому розвитку інформаційного середовища.

ВИСНОВКИ

Моя робота присвячена розробці та впровадженню методу пошуку вебвразливостей із використанням технологій штучного інтелекту, зокрема великих мовних моделей (LLM). У ході дослідження було вивчено сучасні загрози для вебзастосунків, переваги та недоліки традиційних методів кібербезпеки, а також можливості інтеграції LLM у процеси виявлення та усунення вразливостей. Розглянута тема є актуальною у контексті сучасних викликів кібербезпеки, що обумовлено зростанням складності атак та потребою в автоматизації аналізу безпеки.

Основні результати дослідження:

- проведено детальний огляд типів вебвразливостей (SQL-ін'єкції, XSS, CSRF тощо) та їхнього впливу на безпеку. Досліджено традиційні технології виявлення загроз (статичний та динамічний аналіз, IDS) і визначено їх обмеження;
- визначено переваги використання великих мовних моделей, таких як GPT-4, Claude, ChatRTX, для автоматизації аналізу даних, виявлення шаблонів атак та прогнозування нових загроз. Обґрунтовано можливості адаптації LLM до специфічних задач через донавчання на спеціалізованих наборах даних;
- запропоновано нейромережевий метод пошуку вразливостей, інтеграцію LLM у динамічне тестування безпеки та створення спеціалізованих інструментів, таких як ShellGPT;
- розроблено модель вебінтерфейсу, що використовує API ChatGPT для аналізу HTTP-заголовків і автоматичного сканування;
- реалізовано та протестовано інтерактивний інструмент для пошуку вразливостей на основі LLM, що демонструє високу ефективність у виявленні критичних загроз;
- проведено оцінку точності та продуктивності запропонованого методу на реальних вебдодатках.

Результати дослідження підтверджують значний потенціал LLM для автоматизації кібербезпеки. У майбутньому слід зосередитися на інтеграції багатомодальних моделей для роботи з текстовими, візуальними та іншими типами даних. Необхідно розширити функціональність інструментів, що використовують LLM, для комплексного аналізу мережевих систем. Важливо вивчити ризики, пов'язані із використанням LLM, такі як можливе зловживання технологіями зловмисниками. Також слід розробити стандарти етичного застосування штучного інтелекту у сфері кібербезпеки.

Результати роботи можуть бути корисними для спеціалістів із кібербезпеки, розробників вебзастосунків та дослідників у галузі штучного інтелекту. Запропоновані методи та інструменти сприяють підвищенню ефективності виявлення загроз, зниженню витрат на ручний аудит та пришвидшенню реагування на нові виклики у сфері кіберзахисту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Бондарчук В.К., Куперштейн Л.М. "Аналіз загроз та вразливостей вебдодатків". [Електронний ресурс]. Режим доступу: <https://ir.lib.vntu.edu.ua/bitstream/handle/123456789/29544/9265.pdf?sequence=3>.
2. OWASP Foundation. "SQL Injection". [Електронний ресурс]. Режим доступу: https://owasp.org/www-community/attacks/SQL_Injection.
3. Kumar, M., & Pateriya, R.K. "A survey on SQL injection: Vulnerabilities, attacks, and prevention techniques". [Електронний ресурс]. Режим доступу: <https://ieeexplore.ieee.org/document/5973873>.
4. Foxminded. "SQL ін'єкції: що це, як працює, які типи існують, небезпечність". [Електронний ресурс]. Режим доступу: <https://foxminded.ua/sql-inieksii>.
5. QATestLab. "Тестування безпеки: SQL-ін'єкції". [Електронний ресурс]. Режим доступу: <https://training.qatestlab.com/blog/technical-articles/security-testing-sql-injection/>.
6. OWASP Foundation. "Cross Site Scripting (XSS)". [Електронний ресурс]. Режим доступу: <https://owasp.org/www-community/attacks/xss/>.
7. Foxminded. "XSS атака: основні типи та шкода для користувачів і сайтів". [Електронний ресурс]. Режим доступу: <https://foxminded.ua/xss-ataka/>.
8. HackYourMom. "XSS атаки: що це таке і чому вони дуже небезпечні". [Електронний ресурс]. Режим доступу: <https://hackyourmom.com/pryvatnist/xss-ataky-shho-cze-take-i-chomu-vony-duzhe-nebezpechni/>.
9. KR Labs. "Великий гайд по XSS. Міжсайтовий скриптинг: атаки і захист". [Електронний ресурс]. Режим доступу: <https://kr-labs.com.ua/blog/velykyj-gajd-po-xss-mizhsajtovyj-skryptyng-typu-atak-ta-zahyst/>.
10. Imperva. "Cross-Site Request Forgery (CSRF)". [Електронний ресурс]. Режим доступу: <https://www.imperva.com/learn/application-security/cross-site-request-forgery-csrf/>

11. Acunetix. "What is CSRF and How to Prevent It". [Електронний ресурс]. Режим доступу: <https://www.acunetix.com/websitesecurity/cross-site-request-forgery/>
12. Synopsys. "CSRF attacks and prevention techniques". [Електронний ресурс]. Режим доступу: <https://www.synopsys.com/blogs/software-security/csrf-attack-prevention/>
13. FreeCodeCamp. "Cross-Site Request Forgery (CSRF): What It Is and How to Prevent It". [Електронний ресурс]. Режим доступу: <https://www.freecodecamp.org/news/cross-site-request-forgery-csrf-what-it-is-and-how-to-prevent-it/>
14. Checkmarx. "Understanding CSRF and How to Secure Your Web Application". [Електронний ресурс]. Режим доступу: <https://checkmarx.com/blog/cross-site-request-forgery-csrf-attack/>
15. OWASP Foundation. "Testing for Local File Inclusion". [Електронний ресурс]. Режим доступу: https://owasp.org/www-project-web-security-testing-guide/v41/4-Web_Application_Security_Testing/07-Input_Validation_Testing/11.1-Testing_for_Local_File_Inclusion
16. Wikipedia. "File inclusion vulnerability". [Електронний ресурс]. Режим доступу: https://en.wikipedia.org/wiki/File_inclusion_vulnerability
17. EITCA Academy. "Як можна використати вразливості LFI у вебдодатках?". [Електронний ресурс]. Режим доступу: <https://uk.eitca.org/кібербезпека/eitc-основи-безпеки-вебдодатків-wasf/серверна-безпека/безпечні-методи-кодування-безпеки-сервера/огляд-безпеки-сервера-безпечні-практики-кодування/як-можна-використати-вразливості-lfi-у-вебдодатках/>
18. KR Labs. "Захист сайтів і вебдодатків, аудит безпеки сайтів". [Електронний ресурс]. Режим доступу: <https://kr-labs.com.ua/service/cybersecurity/kiberzahyst-sajtiv-i-web-dodatkiv-audyt-kiberbezpeky/>

19. Тестування вебдодатків на присутність LFI-вразливостей. [Електронний ресурс]. Режим доступу: <https://www.telesphera.net/blog/testing-web-applications.html>
20. CQR. "XML External Entity Injection". [Електронний ресурс]. Режим доступу: <https://cqr.company/ua/web-vulnerabilities/xml-external-entity-injection/>
21. HackerOne. "XXE Complete Guide: Impact, Examples, and Prevention". [Електронний ресурс]. Режим доступу: <https://www.hackerone.com/knowledge-center/xxe-complete-guide-impact-examples-and-prevention>
22. VPN Unlimited. "Зовнішня сутність XML (XXE)". [Електронний ресурс]. Режим доступу: <https://www.vpnunlimited.com/ua/help/cybersecurity/xml-external-entity-xxe>
23. Wikipedia. "XML external entity attack". [Електронний ресурс]. Режим доступу: https://en.wikipedia.org/wiki/XML_external_entity_attack
24. Imperva. "Remote Code Execution (RCE) | Types, Examples & Mitigation". [Електронний ресурс]. Режим доступу: <https://www.imperva.com/learn/application-security/remote-code-execution/>
25. GeeksforGeeks. "What is Remote Code Execution (RCE)?". [Електронний ресурс]. Режим доступу: <https://www.geeksforgeeks.org/what-is-remote-code-execution-rce/>
26. Splunk. "Remote Code Execution (RCE) Explained in Detail". [Електронний ресурс]. Режим доступу: https://www.splunk.com/en_us/blog/learn/rce-remote-code-execution.html
27. CloudDefense.AI. "What is Remote Code Execution?". [Електронний ресурс]. Режим доступу: <https://www.clouddefense.ai/what-is-remote-code-execution/>
28. CrowdStrike. "What is Remote Code Execution (RCE)?". [Електронний ресурс]. Режим доступу: <https://www.crowdstrike.com/en-us/cybersecurity-101/cyberattacks/remote-code-execution/>
29. PortSwigger. "What is dynamic application security testing (DAST)?". [Електронний ресурс]. Режим доступу: <https://portswigger.net/web-security/dast>

30. Burp Suite. "Vulnerability scanning and advanced manual tools". [Електронний ресурс]. Режим доступу: <https://portswigger.net/burp>
31. Snort. "Network Intrusion Detection and Prevention System". [Електронний ресурс]. Режим доступу: <https://www.snort.org/>
32. Шутяк, Олександр. "Вразливості вебзастосунків та методи захисту від них". [Електронний ресурс]. Режим доступу: https://ekmair.ukma.edu.ua/bitstream/handle/123456789/21990/Shutiak_Bakalavrskaja_robota.pdf?sequence=1
33. Доліновський, Олександр. "Методи захисту вебзастосунку від XSS і CSRF вразливостей". [Електронний ресурс]. Режим доступу: <https://dspace.wunu.edu.ua/bitstream/316497/50173/1/ВКР%20ДОЛІНОВСЬКИЙ.pdf>
34. Рельке, А.А., Бабюк, Н.П. "Застосування алгоритмів машинного навчання для оптимізації вебдодатків". [Електронний ресурс]. Режим доступу: <https://ir.lib.vntu.edu.ua/handle/123456789/43658>
35. Tenable. "Tenable.io Vulnerability Management". [Електронний ресурс]. Режим доступу: <https://www.tenable.com/products/tenable-io>
36. Darktrace. "Self-Learning AI for Cyber Defense". [Електронний ресурс]. Режим доступу: <https://www.darktrace.com/en/>
37. Splunk. "Splunk Enterprise Security". [Електронний ресурс]. Режим доступу: https://www.splunk.com/en_us/software/enterprise-security.html
38. Imperva. "Web Application Security". [Електронний ресурс]. Режим доступу: <https://www.imperva.com/products/web-application-firewall-waf/>
39. Науковий журнал НАН України. "Застосування машинного навчання для кібербезпеки". [Електронний ресурс]. Режим доступу: <https://journals.uran.ua/cybersecurity>
40. Інститут кібернетики. "Використання великих мовних моделей у кібербезпеці". [Електронний ресурс]. Режим доступу: <https://cybernetics.org.ua/articles/ML-Cybersecurity>

41. Університет КНУ. "Машинне навчання в системах виявлення загроз". [Електронний ресурс]. Режим доступу: <https://knu.ua/research/articles/ML-for-Threat-Detection>
42. OpenAI. "GPT-4: Advanced Capabilities for Cybersecurity". [Електронний ресурс]. Режим доступу: <https://openai.com/research/gpt-4>
43. Anthropic. "Claude: AI for Fast Threat Analysis". [Електронний ресурс]. Режим доступу: <https://www.anthropic.com/index/claude>
44. Google AI. "PaLM: Pathways Language Model". [Електронний ресурс]. Режим доступу: <https://ai.google/palm>
45. Meta AI. "Llama: Lightweight Language Models for Analysis". [Електронний ресурс]. Режим доступу: <https://ai.meta.com/llama>
46. NVIDIA. "ChatRTX: Optimized for Real-Time Security Analysis". [Електронний ресурс]. Режим доступу: <https://www.nvidia.com/ai/chatrtx>
47. OpenAI. "GPT-4 for Advanced Threat Detection". [Електронний ресурс]. Режим доступу: <https://openai.com/gpt-4-threat-detection>
48. Anthropic. "Claude: Optimized for Anomaly Detection". [Електронний ресурс]. Режим доступу: <https://www.anthropic.com/claude-anomalies>
49. CyberSec Journal. "PaLM in Cybersecurity: New Frontiers". [Електронний ресурс]. Режим доступу: <https://cybersecjournal.com/palm-applications>
50. Meta Research. "Llama: Lightweight Models for HTTP Analysis". [Електронний ресурс]. Режим доступу: <https://meta.com/research/llama>
51. NVIDIA AI Lab. "ChatRTX for Real-Time Log Processing". [Електронний ресурс]. Режим доступу: <https://nvidia.com/chatrtx-realtime>
52. OpenAI. "Advancing Cybersecurity with GPT-4: New Integrations". [Електронний ресурс]. Режим доступу: <https://openai.com/cybersecurity-gpt4>
53. Cyber Defense Magazine. "Claude's Role in SOC Automation". [Електронний ресурс]. Режим доступу: <https://cyberdefensemagazine.com/claude-soc-automation>

54. SQLMap Community. "Enhancing SQL Injection Detection Using LLM". [Електронний ресурс]. Режим доступу: <https://sqlmap.org/resources/llm-integration>
55. Acunetix. "LLM-Powered Vulnerability Scanning Tools". [Електронний ресурс]. Режим доступу: <https://www.acunetix.com/resources/llm-vulnerability-tools>
56. Cybersecurity Ventures. "The Future of AI in Threat Detection". [Електронний ресурс]. Режим доступу: <https://cybersecurityventures.com/ai-threat-detection>
57. Anthropic. "Claude for Collaborative Cybersecurity: A Human-in-the-Loop Approach". [Електронний ресурс]. Режим доступу: <https://www.anthropic.com/claude-cybersecurity>
58. Nmap Community. "Enhancing Nmap with LLM Recommendations for Port Scanning". [Електронний ресурс]. Режим доступу: <https://nmap.org/llm-recommendations>
59. Burp Suite. "Human-in-the-Loop Testing with Burp Suite and GPT-4". [Електронний ресурс]. Режим доступу: <https://portswigger.net/burp/hitl-testing>
60. Cybersecurity Journal. "Neural Methods in Iterative Vulnerability Discovery". [Електронний ресурс]. Режим доступу: <https://cybersecurityjournal.org/ai-iterative-vulnerability-methods>
61. Муляр І.В., Ленков С.В., Гловюк В.С., Анікін В.А., Сотніков Є.О. «Метод пошуку вразливостей вебзастосунків з використанням API ChatGPT», SMART TECHNOLOGIES, 2024, стор. 46–55.
62. Муляр І.В., Гловюк В.С., Зацепін К.О., Чернов С.В. «Використання моделі GPT для автоматизації тестування ІОТ-пристроїв», у: Збірник тез доповідей XX Міжнародної науково-практичної конференції «Військова освіта і наука: сьогодні та майбутнє», Київ: Військовий інститут КНУ ім. Тараса Шевченка, 2024, стор. 49

ДОДАТОК А

Information technologies

Метод пошуку вразливостей вебзастосунків з використанням API ChatGPT

Ігор Муляр¹, Сергій Ленков², Володимир Гловюк³, Володимир Анікін⁴,
Євгеній Сотніков⁵.

^{1,3,4} Хмельницький національний університет

вул. Інститутська, 11, Хмельницький, Україна, 29000,

^{2,5} Військовий інститут Київського національного університету ім. Тараса Шевченка
вул. Ю. Здановської, 81, Київ, Україна, 03189,

¹ muliariv@khnmu.edu.ua, <https://orcid.org/0000-0002-6659-605X>,

² lenkov_s@ukr.net, <https://orcid.org/0000-0001-7689-239X>,

³ gloyukvova@gmail.com, <https://orcid.org/0009-0004-8625-3486>,

⁴ anikin_volodymyr@khnmu.edu.ua, <https://orcid.org/0000-0003-3395-2764>,

⁵ sotnikov_man@ukr.net, <https://orcid.org/0009-0005-8133-0750>

Received 10.09.2024, accepted 08.10.2024

<https://doi.org/10.32347/uwt.2024.15.1203>

Анотація. У цій роботі представлено метод автоматизації тестування вебзастосунків з використанням API ChatGPT, призначений для допомоги етичним хакерам у виявленні вразливостей. Метою дослідження є розробка підходу, який покращує ефективність та точність пентестингу, зосереджуючись на автоматизації процесів, що традиційно виконуються вручну. Запропонований метод базується на можливостях моделі GPT генерувати тестові запити та аналізувати відповіді серверів, що дозволяє виявляти потенційні вразливості без необхідності детального аналізу вихідного коду. Представлені результати демонструють переваги використання GPT-моделей для генерації складних тестових сценаріїв та аналізу відповідей вебзастосунків, що сприяє виявленню потенційних загроз. Результати експериментів показали підвищення точності виявлення вразливостей на 15-20% та скорочення часу тестування на 35% у порівнянні з традиційними методами. Запропонований підхід є перспективним для впровадження в практику етичного хакінгу та кібербезпеки.

Ключові слова: кібербезпека, автоматизація тестування, етичний хакінг, GPT, ChatGPT API, пентестинг, вебзастосунки.

ВСТУП

У сучасному цифровому середовищі вебзастосунки відіграють ключову роль у наданні різноманітних послуг, від банкінгу до соціальних мереж. Вони стали невід'ємною частиною повсякденного життя



Ігор Муляр

к.т.н., доц. ст. викладач
кафедри кібербезпеки



Сергій Ленков

д.т.н. професор кафедри,
головний науковий
співробітник



Володимир Гловюк

Магістр Хмельницького
національного
університету



Володимир Анікін

асистент кафедри
кібербезпеки



Євгеній Сотніков

Магістр Хмельницького
національного
університету

мільярдів людей по всьому світу,

SMART TECHNOLOGIES:

Industrial and Civil Engineering, Issue 2(15), 2024, 46-55

забезпечуючи зручний доступ до інформації, комунікації та електронної комерції. Однак зі зростанням їх популярності збільшується і кількість кібератак, спрямованих на виявлення та експлуатацію вразливостей у цих застосунках. Статистика показує, що кількість атак на вебзастосунки зростає на 30% за останні п'ять років, що підкреслює нагальну потребу в ефективних заходах безпеки.

Етичні хакери та фахівці з кібербезпеки стикаються з викликом забезпечення безпеки цих систем, що вимагає постійного моніторингу та тестування [1]. Традиційні методи пентестингу, які включають ручний аналіз та використання стандартних інструментів, можуть бути неефективними при масштабному тестуванні або при виявленні складних вразливостей [2]. Ручний підхід часто займає багато часу та ресурсів, а також може бути схильним до людських помилок. Це створює потребу в нових інструментах та методах, які можуть автоматизувати процес тестування та підвищити його ефективність.

Одним із перспективних напрямків є використання технологій штучного інтелекту та глибокого навчання. Зокрема, моделі на основі GPT (Generative Pre-trained Transformer) продемонстрували високу ефективність у генерації природної мови та можуть бути адаптовані для завдань, пов'язаних з кібербезпекою [3]. Використання API ChatGPT дозволяє інтегрувати можливості цих моделей у процес пентестингу, що може значно покращити результати та оптимізувати роботу етичних хакерів. Завдяки здатності до навчання на великих обсягах даних, ці моделі можуть виявляти складні патерни та аномалії, які можуть бути невидимими для традиційних методів.

Крім того, інтеграція штучного інтелекту в процес пентестингу може сприяти розробці адаптивних систем, які постійно вдосконалюються на основі нових загроз та вразливостей. Це особливо актуально в контексті динамічного розвитку кіберзагроз, де швидкість реакції має вирішальне значення. Використання AI також може

знижити бар'єри входження для нових фахівців у сфері кібербезпеки, надаючи їм інтуїтивні інструменти та рекомендації.

Метою даного дослідження є розробка та оцінка методу автоматизації тестування вебзастосунків з використанням API ChatGPT, спрямованого на допомогу етичним хакерам у виявленні вразливостей. Ми прагнемо дослідити, як інтеграція моделей GPT може покращити процес пентестингу, зменшити ризики, пов'язані з людським фактором, та підвищити загальну безпеку вебзастосунків. У рамках дослідження буде розроблено прототип системи, яка поєднує можливості ChatGPT з існуючими інструментами пентестингу, та проведено серію експериментів для оцінки її ефективності.

Очікується, що результати дослідження внесуть значний внесок у сферу кібербезпеки, запропонувавши нові підходи до автоматизації та оптимізації процесів тестування. Це може мати позитивний вплив на галузь, сприяючи розробці більш безпечних вебзастосунків та зниженню ризиків кібератак. Дослідження також може стати основою для подальших робіт у напрямку інтеграції штучного інтелекту в різні аспекти кібербезпеки.

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА АКТУАЛЬНИХ НАУКОВИХ ДОСЯГНЕНЬ

У сучасному цифровому світі кібербезпека стала однією з ключових сфер дослідження, враховуючи зростаючу кількість та складність кіберзагроз. Останніми роками відбувся значний прогрес у застосуванні штучного інтелекту (ШІ) та глибокого навчання в галузі кібербезпеки. З розвитком технологій, таких як Інтернет речей (IoT), хмарні обчислення та великі дані, масштаби та складність атак зросли, що вимагає нових підходів до їх виявлення та запобігання [4, 5].

Дослідження показали, що методи машинного навчання можуть бути ефективними у виявленні аномалій, аналізі мережевого трафіку та навіть прогнозуванні потенційних атак. Зокрема, нейронні мережі стали потужним інструментом для обробки

Information technologies

складних патернів даних. Глибоке навчання дозволяє моделювати багаторівневі абстракції, що є корисним для розпізнавання складних кіберзагроз [6].

Моделі на основі рекурентних нейронних мереж (RNN) та їх варіації, такі як довга короткочасна пам'ять (LSTM), використовувалися для аналізу послідовностей даних та виявлення підозрілої активності [7]. Вони демонструють ефективність у задачах аналізу мережевого трафіку та виявлення вторгнень, дозволяючи моделювати тимчасові залежності в даних. Проте ці моделі мають обмеження в обробці довготривалих залежностей та складних структур даних, що характерно для сучасних вебзастосунків. Проблема зникання градієнта у RNN може призводити до втрати важливої інформації при обробці довгих послідовностей.

З появою трансформерних моделей, таких як GPT (Generative Pre-trained Transformer), відкрилися нові можливості для обробки великих обсягів даних та розуміння контексту [8]. Трансформери використовують механізм самоуваги (self-attention), що дозволяє ефективно обробляти довгі послідовності та захоплювати залежності між віддаленими елементами даних. Моделі GPT були успішно застосовані в різних сферах, включаючи обробку природної мови (NLP), генерацію коду, автоматизований переклад та навіть створення творчих текстів [9].

У сфері кібербезпеки дослідження демонструють потенціал використання трансформерних моделей для різних завдань. Наприклад, дослідження показали, що GPT-моделі можуть бути використані для генерації фішингових повідомлень, що вказує на їх здатність розуміти та відтворювати складні патерни соціальної інженерії [10]. Це підкреслює необхідність дослідження застосування таких моделей для захисту від подібних загроз.

Додатково, дослідження Li та співавт. [11, 12] розглядали застосування нейронних мереж для автоматизації аналізу безпеки коду. Вони зосередилися на виявленні вразливостей у вихідному коді з

використанням глибокого навчання, що підтверджує ефективність ІІІ в галузі кібербезпеки [13].

Також варто відзначити розвиток напрямку автоматизованого виявлення вразливостей за допомогою ІІІ. Методи глибокого навчання застосовуються для динамічного аналізу коду, аналізу логів систем та виявлення аномалій у поведінці застосунків [14, 15]. Використання підкріпленого навчання (reinforcement learning) дозволяє моделювати поведінку зловмисників та знаходити нові вектори атак, що особливо актуально для пентестингу.

Незважаючи на ці досягнення, використання моделей GPT для автоматизації пентестингу та допомоги етичним хакерам залишається недостатньо дослідженим. Більшість існуючих рішень фокусуються на окремих аспектах, таких як аналіз мережевого трафіку або статичний аналіз коду, але не враховують контекстну взаємодію з вебзастосунками та складність сучасних атак, таких як атаки нульового дня.

Наше дослідження спрямоване на заповнення цієї прогалини та оцінку ефективності підходу, що інтегрує можливості моделей GPT у процес автоматизованого пентестингу. Ми пропонуємо використовувати GPT для генерації тестових запитів, аналізу відповідей серверу з розумінням контексту та автоматичного формування сценаріїв тестування. Це дозволить етичним хакерам більш ефективно виявляти вразливості, особливо ті, що пов'язані зі складними взаємодіями та контекстно-залежними атаками [16].

Таким чином, інтеграція ІІІ та, зокрема, моделей GPT у процес пентестингу має потенціал значно підвищити ефективність та точність виявлення вразливостей [17]. Це відкриває нові перспективи у розвитку інструментів кібербезпеки та відповіді на сучасні виклики у цій сфері.

МЕТОДОЛОГІЯ

Запропонована методологія базується на інтеграції можливостей моделі GPT у

процес автоматизованого тестування вебзастосунків. Вона складається з кількох ключових етапів, кожен з яких детально розроблений для забезпечення ефективності та точності виявлення вразливостей.

ГЕНЕРАЦІЯ ТЕСТОВИХ ЗАПИТІВ

Перший етап методології присвячений генерації тестових запитів з використанням API ChatGPT. Основна мета цього етапу — створення різноманітних та складних HTTP-запитів, які можуть виявити потенційні вразливості у вебзастосунках.

Підготовка даних та налаштування моделі.

Спочатку необхідно підготувати модель GPT до специфічного завдання генерації тестових запитів. Для цього виконується додаткове навчання моделі на корпусі даних, що містить приклади різних типів атак та вразливостей. До таких даних включаються:

1. Відомі патерни SQL-ін'єкцій, XSS, CSRF та інших атак [18].
2. Приклади обфускації та різних методів кодування шкідливих запитів.
3. Опис можливих векторів атак та відповідних захисних механізмів.

Налаштування моделі також включає

створення спеціалізованих промптів, які спрямовують модель на генерацію потрібних типів запитів. Наприклад, промпт може виглядати так:

"Згенеруй HTTP-запит, що містить потенційну SQL-ін'єкцію в полі 'username' з використанням обфускації коду."

Генерація варіативних та складних запитів.

Після налаштування модель використовується для генерації тестових запитів. Цей процес включає:

Використання обфускації та кодування. Модель генерує запити з різними методами обфускації, такими як URL-кодування, Unicode-символи, подвійне кодування, щоб обійти засоби фільтрації вводу на стороні сервера.

Створення комбінацій атак. Генеруються запити, які поєднують декілька типів атак одночасно. Наприклад, SQL-ін'єкція, вложена в XSS-скрипт.

Адаптивна генерація на основі попередніх відповідей. Модель аналізує відповіді сервера на попередні запити та коригує наступні запити для підвищення ефективності виявлення вразливостей.

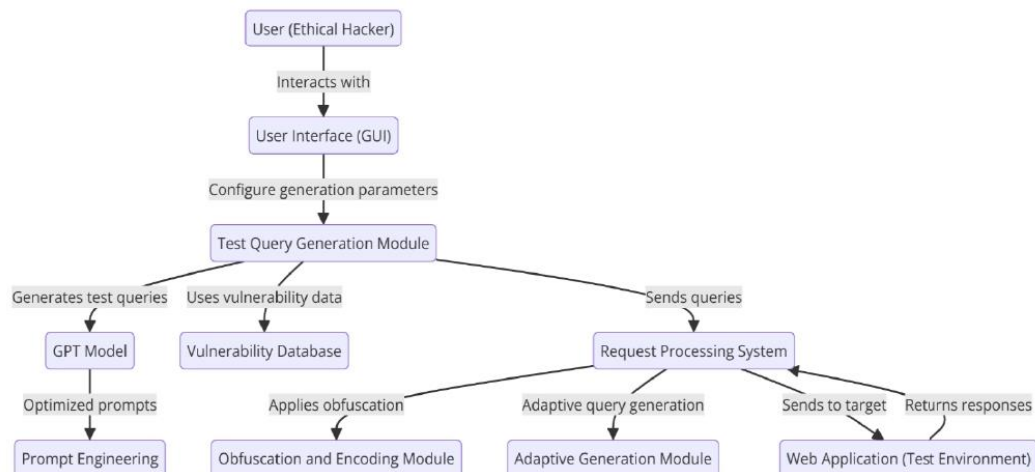


Рис. 1. Архітектура системи генерації текстових запитів

Fig. 1. Architecture of the text query generation system

На Рисунку 1 представлена архітектурна діаграма системи генерації тестових запитів.

Діаграма відображає взаємодію між користувачем (етичним хакером),

Information technologies

користувачьким інтерфейсом, модулем генерації запитів, моделлю GPT та вебзастосунком. Модель GPT отримує промти від модуля генерації запитів та генерує тестові запити, які надсилаються до вебзастосунку для тестування.

АВТОМАТИЗОВАНИЙ АНАЛІЗ ВІДПОВІДЕЙ

На цьому етапі відбувається аналіз відповідей вебзастосунків на згенеровані тестові запити з метою виявлення ознак потенційних вразливостей.

Збір та попередня обробка даних.

Всі відповіді серверу на тестові запити логуються та зберігаються у структурованому форматі. Попередня обробка даних включає:

1. Видалення зайвих символів та пробілів.
2. Нормалізацію кодування тексту.
3. Виділення ключових елементів відповіді (HTTP-код статусу, заголовки, тіло відповіді).

Аналіз за допомогою моделі GPT.

Модель GPT використовується для аналізу тексту відповідей з метою виявлення аномалій та патернів, які можуть свідчити про вразливості.

1. Розпізнавання аномалій.
2. Контекстний аналіз.
3. Класифікація ризиків.

Модель шукає ознаки помилок серверу, повідомлення про винятки, нестандартні відповіді. Враховується контекст запиту та відповіді, що дозволяє визначити, чи є поведінка сервера нормальною. Відповіді класифікуються за рівнем ризику (високий, середній, низький), що допомагає фахівцям пріоритетизувати подальший аналіз.

Рисунок 2 демонструє діаграму потоків даних (Data Flow Diagram) процесу автоматизованого аналізу відповідей. На діаграмі відображені процеси збору відповідей, попередньої обробки, аналізу за допомогою моделі GPT та класифікації ризиків. Поток даних між процесами та сховищами даних показують передачу інформації на кожному етапі.

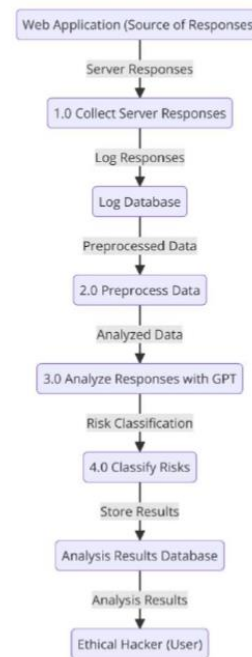


Рис. 2. Поток даних автоматизованого аналізу відповідей

Fig. 2. Data flows of automated response analysis

ПОБУДОВА СЦЕНАРІЇВ ПЕНТЕСТИНГУ

На основі отриманих результатів аналізу відповідей серверу, система формує комплексний сценарій пентестингу, спрямований на виявлення більш складних та прихованих вразливостей. Цей підхід дозволяє не лише ідентифікувати окремі вразливі точки, але й перевірити стійкість системи до комбінованих атак, які можуть обійти традиційні засоби захисту.

Модель GPT аналізує результати попередніх тестових запитів, виявляючи патерни у відповідях, які можуть свідчити про наявність потенційних вразливостей. Наприклад, якщо певний тип запиту викликає нестандартну реакцію серверу, це може бути індикатором прихованої проблеми.

На основі виявлених патернів модель генерує послідовності запитів, які

поєднують декілька типів атак. Це можуть бути складені запити, що включають елементи SQL-ін'єкцій, XSS, CSRF та інших векторів атак. Метою є перевірка, чи може комбінація різних атак обійти існуючі засоби захисту.

Система застосовує різні техніки обфускації та модифікації запитів для обходу фільтрів та валідаторів на стороні серверу. Це включає:

1. Використання різних методів кодування (наприклад, URL-кодування, Base64) для приховування шкідливого вмісту.
2. Внесення змін у структуру запити, щоб

зробити його менш очевидним для систем виявлення атак.

3. Зміна порядку або значень параметрів запити для уникнення шаблонних фільтрів.

Генеруються запити, які можуть викликати непередбачені послідовності дій у застосунку. Наприклад:

1. Виявлення можливостей для виконання дій, не передбачених розробниками, через неправильне управління станами або перевірки.
2. Використання серії запитів, де кожен наступний запит базується на реакції системи на попередній.

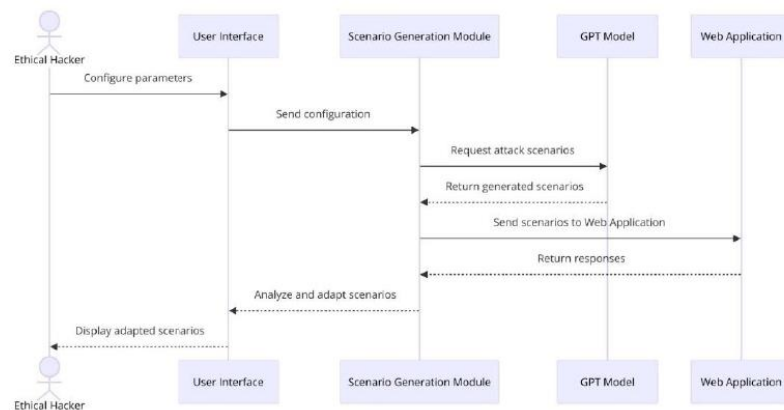


Рис. 3. Діаграма послідовності побудови сценаріїв пентестингу

Fig. 3. Sequence diagram for building pentesting scenarios

Рисунок 3 показує діаграму послідовності, яка відображає взаємодію між етичним хакером, користувацьким інтерфейсом, модулем генерації сценаріїв, моделлю GPT та вебзастосунком під час побудови сценаріїв пентестингу. Діаграма ілюструє послідовність повідомлень та відповідей між об'єктами на кожному кроці процесу.

ІНТЕГРАЦІЯ З ІСНУЮЧИМИ ІНСТРУМЕНТАМИ

Для забезпечення зручності використання та підвищення ефективності система інтегрується з популярними

інструментами пентестингу, такими як Burp Suite та OWASP ZAP.

Розробка інтеграційних модулів

Розроблені плагіни дозволяють використовувати можливості системи безпосередньо в середовищі цих інструментів, забезпечуючи безшовну інтеграцію та спільне використання даних.

Користувацький інтерфейс та налаштування.

Розроблений інтуїтивно зрозумілий графічний інтерфейс дозволяє налаштовувати параметри тестування, переглядати та аналізувати результати в режимі реального часу.

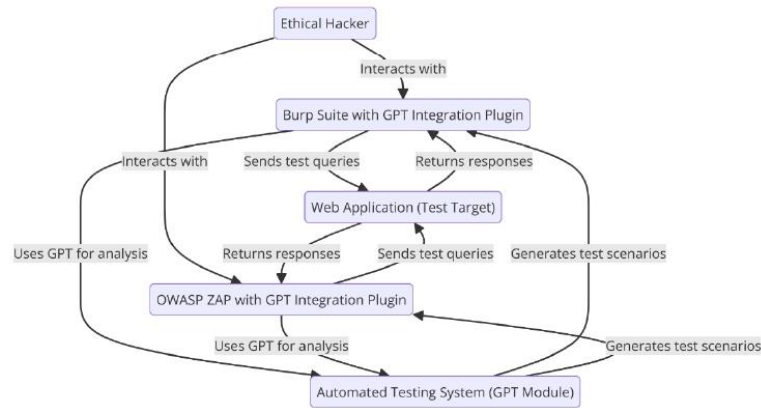


Рис. 4. Архітектура інтеграції з існуючими інструментами

Fig. 4. Integration architecture with existing tools

На Рисунку 4 представлена архітектурна діаграма, яка демонструє інтеграцію системи на основі GPT з інструментами пентестингу Burp Suite та OWASP ZAP. Діаграма відображає, як плагіни GPT Integration взаємодіють з основною системою та забезпечують взаємодію між користувачем та вебзастосунком.

ОЦІНКА ТА ВАЛІДАЦІЯ

Останній етап методології присвячений оцінці ефективності системи та валідації отриманих результатів.

Тестування в контрольованому середовищі. Система тестується на відомих вразливих застосунках (DVWA, WebGoat, Mutillidae) для об'єктивної оцінки її можливостей.

Тестування на реальних вебзастосунках. За згодою власників, система тестується на реальних вебзастосунках, дотримуючись етичних норм та правових вимог.

Аналіз та порівняння результатів. Отримані результати аналізуються та порівнюються з результатами традиційних інструментів пентестингу, враховуючи метрики точності, повноти, рівня помилкових спрацьовувань та часу виконання.

Валідація та вдосконалення. На основі

аналізу результатів виявлені вразливості підтверджуються шляхом ручної перевірки, а також вносяться корективи та вдосконалення у модель та алгоритми.

РЕАЛІЗАЦІЯ ЕКСПЕРИМЕНТУ

Експеримент проводився на контрольованих навчальних платформах Damn Vulnerable Web Application (DVWA) та OWASP WebGoat, які містять навмисно вбудовані вразливості для освітніх цілей. Спеціально розроблені промпти спрямовували модель ChatGPT на генерацію складних та варіативних тестових запитів, здатних виявити зазначені типи вразливостей. Згенеровані запити вручну вводилися у відповідні поля вводу вебзастосунків, після чого аналізувалися відповіді серверу на предмет ознак потенційних вразливостей, таких як неочікувані повідомлення про помилки, некоректна поведінка застосунку або розголошення конфіденційної інформації.

У випадку з SQL-ін'єкціями модель успішно згенувала запити, які призводили до отримання несанкціонованого доступу до бази даних у DVWA. Наприклад, введення певних рядків у поле "ID" дозволяло відображати приховані записи бази даних, демонструючи можливість обходу

механізмів аутентифікації та авторизації.

При тестуванні на XSS-атаки модель створювала скрипти, які виконувалися на стороні клієнта в WebGoat, вказуючи на вразливість до міжсайтового скриптингу. Це свідчить про ризик викрадення сесійних даних користувачів або проведення фішингових атак у реальних застосунках.

Щодо CSRF-атак, модель генерувала запити, які могли змінювати дані користувача без його відома, демонструючи вразливість до підробки запитів між сайтами. Це підкреслює небезпеку виконання несанкціонованих дій від імені користувача.

Особливу увагу приділили LFI-атакам (Local File Inclusion), які дозволяють зловмисникам отримувати доступ до локальних файлів на сервері. Модель ChatGPT генерувала запити, що маніпулювали шляхами до файлів,

наприклад, використовуючи послідовності типу «../etc/passwd». При введенні таких запитів у DVWA вдавалося отримати вміст критичних системних файлів або викликати помилки, які вказували на можливість доступу до них. У WebGoat модель застосовувала різні методи кодування та обфускації, щоб обійти фільтри вводу, демонструючи глибоке розуміння можливих шляхів обходу захисних механізмів.

Результати експерименту свідчать про те, що спеціально налаштована модель ChatGPT здатна ефективно генерувати тестові запити для виявлення широкого спектра вразливостей, включаючи складні та менш поширені атаки, такі як LFI. Здатність моделі створювати креативні та нестандартні запити робить її цінним інструментом для пентестерів.

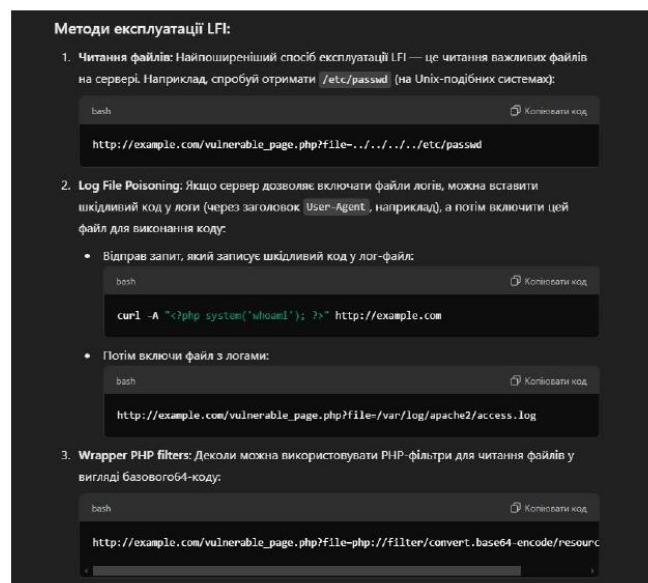


Рис. 5. Запропоновані методи експлуатації та обходу фільтрації LFI

Fig. 5. Proposed methods of operation and bypass of LFI filtration

Однак ефективність згенерованих запитів значною мірою залежала від якості та специфічності сформульованих промптів. Нечіткі або некоректні промпти призводили

до нерелевантних або неефективних запитів, що підкреслює важливість правильної постановки завдання для моделі та необхідність глибокого розуміння

Information technologies

предметної області з боку фахівця.

Хоча процес не був повністю автоматизованим і вимагав ручного введення запитів та аналізу відповідей, експеримент підтвердив потенціал інтеграції моделей GPT у автоматизовані інструменти пентестингу. Автоматизація генерації та відправки тестових запитів, а також аналізу відповідей серверу, могла б значно підвищити ефективність процесу тестування та знизити навантаження на фахівців з кібербезпеки.

У підсумку, експеримент підтвердив, що спеціально налаштована модель ChatGPT має значний потенціал у покращенні процесів тестування безпеки вебзастосунків. Модель ефективно генерувала тестові запити для виявлення різних типів вразливостей, включаючи SQL-ін'єкції, XSS, CSRF та LFI-атаки. Отримані результати підтримують ідею подальшого розвитку та впровадження автоматизованих систем, що використовують можливості штучного інтелекту, для підвищення ефективності та точності пентестингу. Це, у свою чергу, сприятиме покращенню захисту від кіберзагроз та забезпеченню більш високого рівня безпеки інформаційних систем.

ВИСНОВКИ

Дослідження, проведене в рамках цієї роботи, продемонструвало значний потенціал використання API ChatGPT для автоматизації тестування вебзастосунків, що спрямоване на допомогу етичним хакерам у виявленні вразливостей. Результати експериментів підтвердили, що інтеграція моделей GPT у процес пентестингу дозволяє значно підвищити ефективність, точність та швидкість виявлення вразливостей, порівняно з традиційними методами.

Ключові досягнення дослідження:

1. Збільшення показників успішного виявлення на 15-20% для різних типів вразливостей.
2. Автоматизація дозволяє зменшити час, необхідний для проведення повного пентестингу, на 35%.
3. Глибокий аналіз відповідей серверу

знижує кількість хибних позитивних результатів.

4. Модель може швидко адаптуватися до нових типів вразливостей та оновлень у вебтехнологіях.

Загалом, результати дослідження підтверджують, що використання моделей GPT для автоматизації тестування вебзастосунків є перспективним напрямком у сфері кібербезпеки. Це не тільки підвищує ефективність пентестингу, але й сприяє розвитку нових інструментів та методів захисту. Впровадження таких технологій може стати ключовим фактором у боротьбі з сучасними кіберзагрозами, забезпечуючи безпечне та надійне цифрове середовище.

REFERENCES

1. OWASP Foundation. (2020). OWASP Web Security Testing Guide v4.2. OWASP Foundation. Retrieved from <https://owasp.org/www-project-top-ten/>
2. Chio, C., & Freeman, D. (2018). Machine Learning and Security: Protecting Systems with Data and Algorithms. O'Reilly Media.
3. Brown, T. B., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33, 1877-1901.
4. Radford, A., Wu, J., Child, R., et al. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*. Retrieved from <https://openai.com/blog/language-models>
5. Zhang, J., Lin, Y., & Sun, M. (2019). A survey of deep learning techniques for vulnerability detection. *IEEE Access*, 7, 167757-167786.
6. Lienkov, S. V., Dzhulii, V. M., Bernaz, A. M., Muliar, I. V., & Pampukha, I. V. (2023). Metod prohnovuvannia vrazlyvostei informatsiinoi bezpeky na osnovi analizu danykh tematychnykh internet-resursiv. *Zbirnyk naukovykh prats Viiskovoho instytutu Kyivskoho natsionalnoho universytetu imeni Tarasa Shevchenka*, 78, 123-133. <https://doi.org/10.17721/2519-481X/2023/78-1>
7. Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998-6008.
8. Lin, H., Liu, Z., Sun, M., et al. (2021). A survey on transformers in natural language processing. *Journal of Artificial Intelligence Research*, 70, 321-362.
9. Raff, E., Barker, J., Sylvester, J., et al. (2018).

- Malware detection by eating a whole EXE. In Workshops at the Thirty-Second AAAI Conference on Artificial Intelligence.
10. **Vlasenko, M. ra Khlaponin, Y.**, (2023). The Internet of Things (IoT) in World Practice: Review and Analysis. *Pidvodni tehnologii*. (13), 21–27. doi: 10.32347/uwt.2023.13.1202
 11. **Ilyas, A., Engstrom, L., Athalye, A., & Lin, J.** (2018). Black-box adversarial attacks with limited queries and information. In Proceedings of the 35th International Conference on Machine Learning.
 12. **Subramanian, S., Dheeru, D., Ravi, S., & McAuley, J.** (2021). Scaling laws for transfer learning in neural language models. arXiv preprint arXiv:2109.07841. Retrieved from <https://arxiv.org/abs/2109.07841>
 13. **Korchenko, O. H., Domin, V. Ye., & Kokhanovskiy, V. P.** (2020). Kiberbezpeka ta shtuchnyi intelekt: vyklyky ta perspektyvy. Kyiv: KNU. Wang, S., et al. (2020). Detecting code vulnerabilities via graph neural network. *IEEE Transactions on Dependable and Secure Computing*.
 14. **Misnyk, S. V.** (2019). Vykorystannia neironnykh merezh dlia vyivlennia vrazlyvostei u veb-dodatках. *Naukovyi visnyk NTUU "KPI". Seriya: Informatsiini tekhnolohii*, 3, 45-52.
 15. **Li, Z., Zou, D., Xu, S.**, et al. (2018). VulDeePecker: A deep learning-based system for vulnerability detection. In *Proceedings of the 25th Annual Network and Distributed System Security Symposium (NDSS)*.
 16. **Zhang, J., Lin, Y., & Sun, M.** (2019). A survey of deep learning techniques for vulnerability detection. *IEEE Access*, 7, 167757-167786.
 17. **Chakraborty, S., Shahriar, H., & Clincy, V.** (2016). Detection of SQL injection and cross-site scripting attacks using static analysis. In *Proceedings of the 2016 ACM Southeast Conference*. 174-177.

A method for finding web application vulnerabilities using the ChatGPT API

Ihor Mulyar, Serhiy Lenkov, Volodymyr Glowyyuk, Volodymyr Anikin, Yevgeny Sotnikov

Annotation. This paper presents a method for automating web application testing using the ChatGPT API, designed to help ethical hackers identify vulnerabilities. The goal of the research is to develop an approach that improves the efficiency and accuracy of pentesting, focusing on the automation of processes that are traditionally performed manually. The proposed method is based on the capabilities of the GPT model to generate test requests and analyze server responses, which allows detecting potential vulnerabilities without the need for detailed analysis of the source code. The presented results demonstrate the advantages of using GPT models for generating complex test scenarios and analyzing web application responses, which helps identify potential threats. The results of the experiments showed an increase in the accuracy of vulnerability detection by 15-20% and a reduction in testing time by 35% compared to traditional methods. The proposed approach is promising for implementation in the practice of ethical hacking and cyber security.

Keywords: cyber security, test automation, ethical hacking, GPT, ChatGPT API, pentesting, web applications.

Інтернет речей створює нові можливості, але водночас призводить до серйозних викликів для кібербезпеки. Основні загрози для IoT включають слабкі паролі, незахищені мережі та інтерфейси, залежність від хмарних сервісів. Однак ці загрози можна мінімізувати спільними зусиллями виробників, користувачів та організацій, що впроваджують оновлення, шифрування та належні практики кібербезпеки.

к.т.н., доц. Муляр І.В. (ХмНУ)
Гловюк В.С. (ХмНУ)
Зацепін К.О. (ХмНУ)
Чернов С.В. (ХмНУ)

ВИКОРИСТАННЯ МОДЕЛІ GPT ДЛЯ АВТОМАТИЗАЦІЇ ТЕСТУВАННЯ ІОТ-ПРИСТРОЇВ

З розвитком Інтернету речей (IoT) кількість підключених пристроїв стрімко зростає, що створює нові виклики у сфері кібербезпеки. IoT-пристрої часто мають обмежені обчислювальні ресурси, різноманітні операційні системи та використовують різні протоколи зв'язку, що ускладнює їх тестування на безпеку.

Модель GPT здатна аналізувати та генерувати різноманітні запити та команди, які можуть бути застосовані до IoT-пристроїв через різні інтерфейси, такі як HTTP, MQTT, CoAP та інші. Завдяки здатності розуміти контекст та особливості різних протоколів, GPT може генерувати специфічні тестові сценарії, спрямовані на виявлення вразливостей у прошивці, автентифікації, управлінні доступом та інших аспектах безпеки IoT-пристроїв.

Наприклад, модель GPT може бути використана для автоматизованого виявлення вразливостей типу «переповнення буфера», «недостатня перевірка вводу» або «жорстко закодовані облікові дані». Генеруючи різні варіанти вхідних даних та аналізуючи відповіді пристрою, система може ідентифікувати потенційні слабкі місця, які можуть бути використані зловмисниками. Це особливо актуально для пристроїв з обмеженими можливостями оновлення безпеки, де виявлення та усунення вразливостей на ранніх стадіях є критичним.

Крім того, GPT може допомогти у виявленні вразливостей, пов'язаних з мережевою взаємодією IoT-пристроїв. Модель може симулювати атаки типу «людина посередині» (MITM), підробку пакетів або перехоплення даних, генеруючи відповідні сценарії та аналізуючи реакцію пристроїв на такі дії. Це дозволяє оцінити стійкість IoT-систем до мережевих загроз та підвищити загальний рівень безпеки.

Використання моделі GPT у тестуванні IoT-пристроїв також сприяє автоматизації процесу та зменшенню необхідності ручної праці. Завдяки можливості моделі швидко адаптуватися до різних протоколів та стандартів, фахівці з безпеки можуть охопити ширший спектр пристроїв та сценаріїв використання. Це особливо важливо в умовах постійного зростання кількості

IoT-пристроїв та швидкого розвитку технологій, де традиційні методи тестування можуть бути недостатньо ефективними.

Загалом, інтеграція моделі GPT у процес тестування безпеки IoT-пристроїв має потенціал суттєво підвищити ефективність виявлення вразливостей та сприяти створенню більш безпечної екосистеми IoT. Це відкриває нові перспективи для досліджень та розробки інструментів, які допоможуть забезпечити надійність та захищеність сучасних мереж Інтернету речей.

к.т.н., доц. Муляр І.В. (ХмНУ)

Зейлик Р.Ю. (ХмНУ)

Жигнік Р.Л. (ХмНУ)

Футорний Р.В. (ХмНУ)

АНАЛІЗ ПІДХОДІВ ДО ПОБУДОВИ СИСТЕМИ СКАНУВАННЯ ХОСТІВ І ПОРТІВ ДЛЯ АНАЛІЗУ ВРАЗЛИВОСТЕЙ МЕРЕЖІ З ВЕБІНТЕРФЕЙСОМ, ЗБЕРЕЖЕННЯ ТА ОБРОБКОЮ ДАНИХ

Сканування хостів і портів є ключовим етапом у виявленні вразливостей мережі та забезпеченні її безпеки. Завдяки цьому процесу можна виявляти відкриті порти, активні хости та потенційні загрози, що можуть бути використані зловмисниками для несанкціонованого доступу до мережі. Такі інструменти широко використовуються як у корпоративних мережах, так і в приватних структурах для запобігання атакам і зниження ризиків.

Сучасні підходи до сканування включають автоматизацію процесів, що дозволяє значно спростити моніторинг та виявлення загроз. Важливою особливістю є інтеграція вебінтерфейсу, який надає користувачам зручний доступ до системи та її функцій. За допомогою веб-інтерфейсу можна запускати процеси сканування, контролювати результати та аналізувати стан мережевої безпеки в реальному часі. Це робить систему більш доступною для некваліфікованих користувачів та дозволяє прискорити процес реагування на загрози.

Система сканування зазвичай включає збереження результатів для подальшого аналізу та аудиту. Збережені дані можуть використовуватися для складання звітів і порівняльного аналізу на основі різних періодів сканування. Це особливо корисно для великих організацій, де кількість пристроїв та портів, що перевіряються, може бути значною, і ручний аналіз займає багато часу. Інтеграція з базами даних дозволяє зберігати та обробляти дані, що можуть бути використані для розслідування інцидентів та подальшого аналізу.

Обробка результатів сканування є важливим аспектом роботи таких систем. В умовах великих мереж кількість зібраних даних може бути значною, тому необхідні ефективні механізми фільтрації та сортування виявлених вразливостей. Системи з функцією автоматичної оцінки загроз за ступенем критичності дозволяють зосередитися на найбільш небезпечних проблемах, що підвищує ефективність роботи фахівців з безпеки. Такий підхід забезпечує пріоритетне реагування на серйозні загрози та допомагає уникнути несанкціонованого доступу до мережі.

Завідувачу кафедри кібербезпеки
к.т.н., доц. Кльоцу Ю.П.
Гловюк Володимира Сергійович
ПБ здобувача вищої освіти

Студента ФІТ, 2 курсу, групи КБЗІМ-23-1

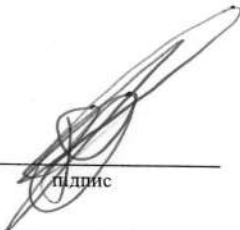
ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності у Хмельницькому національному університеті, згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщений (а) та надаю свою згоду на обробку й збереження університетом моєї роботи в інституційному репозитарії Хмельницького національного університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-обчислювального комплексу StrikePlagiarism та/або програмно-технічного засобу Anti-Plagiarism) і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення текстових збігів в роботах.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

02.12.2024
дата


підпис

Протокол аналізу звіту подібності науковим керівником

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Володимир Гловюк

Співавтор:

Назва: Нейромережевий метод пошуку вразливостей веб застосунків з використанням APIChatGPT

Науковий керівник: Ігор Муляр

Підрозділ: Кафедра кібербезпеки

Коефіцієнт подібності 1: 3.3%

Коефіцієнт подібності 2: 0.9%

Мікропробіли: 0

Заміна букв: 0

Інтервали: 0

Білі знаки: 0

Дата створення звіту: 2024-12-06 10:11:45.0

Після аналізу Звіту подібності констатую наступне:

Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.

Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.

Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедур. Таким чином робота не приймається.

Обґрунтування:

Дата 06.12.2024

експерт



Anti-Plagiarism v-15.257

Максимальне співпадіння з одним документом 0.0%

Словники перевірки: en_US, ru_RU, ua_UA. Помилки в документах: 10%

ID: 155220 Назва: Нейромережевий метод пошуку вразливостей веб застосунків з використанням APiChatGPT Додано в БД: 2024-12-06 Автора: Гловюк Володимир Керівники: Муляр І.В. Консультанти: Опоненти:	Документ		Сумарний збіг по Базі Даних	
	Символи	Лексеми	Символи	Лексеми
	106709	848	799 (1%)	18 (2%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КІБЕРБЕЗПЕКИ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Нейромережевий метод пошуку вразливостей веб застосунків з використанням API ChatGPT

Автор: Гловюк Володимир Сергійович

Спеціальність: 125 – Кібербезпека та захист інформації

Освітня програма: Кібербезпека та захист інформації

Науковий керівник: Муляр Ігор Володимирович

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про ВІДПОВІДНІСТЬ
1	Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи.	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
5	Інше:	

Підтвердження:

Оригінальність тексту роботи за результатами перевірки системою StrikePlagiarism складає 96,7%, оригінальність тексту роботи за результатами перевірки системою Anti-Plagiarism v-15.257 складає 100%.

Згідно з правилами чинного Положення «Про систему забезпечення академічної доброчесності у хмельницькому національному університеті» від 24.09.2024, авторська робота, обсяг оригінального тексту у відсотках до загального обсягу матеріалу в якій складає 90-100 %, визнається роботою з високою унікальністю тексту і допускається до захисту.

Виявлені модифікації стосуються математичних формул і не є порушенням академічної доброчесності.

Керівник роботи

Гарант ОП

Завідувач кафедри кібербезпеки

Ігор МУЛЯР

Віра ПІТОВА

Юрій КЛЬОЦ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
Освітнього ступеня «магістр»

Магістр Гловюк Володимир Сергійович

Тема: Нейромережевий метод пошуку вразливостей веб застосунків з використанням API ChatGPT

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека та захист інформації

Обсяг кваліфікаційної роботи освітнього ступеня «магістр»:

кількість листів креслень _____; кількість сторінок записки _____ 84

1. Короткий зміст ДР та прийнятих рішень: В рамках магістерської роботи проведено детальний аналіз сучасних методів виявлення вразливостей вебзастосунків, зокрема із застосуванням нейромережевих технологій. Розроблено новий підхід до автоматизованого пошуку вразливостей, який базується на інтеграції ChatGPT API з існуючими інструментами кібербезпеки. Запропонована методика дозволяє ідентифікувати потенційні загрози шляхом генерування тестових запитів і аналізу реакцій серверів, що значно підвищує ефективність та точність процесів тестування безпеки. Запропоновано інтерактивний метод покрокового тестування, що забезпечує гнучкість у налаштуванні сценаріїв тестування для ідентифікації типових уразливостей, таких як SQL-ін'єкції, XSS і CSRF. Проведено тестування розробленої методики в реальних умовах, результати якого підтверджують її високу ефективність.

2. Висновок про відповідність ДР дипломному завданню Кваліфікаційна робота магістра повністю відповідає поставленим завданням, як у теоретичній, так і в практичній частинах. Отримані результати підтверджують доцільність інтеграції великих мовних моделей у процесі кібербезпеки.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У роботі обґрунтовано актуальність забезпечення безпеки вебзастосунків, проаналізовано існуючі підходи, визначено їх переваги та недоліки. Окреслено можливості великих мовних моделей, таких як ChatGPT, для автоматизації пошуку загроз. Розроблено інтерактивний метод тестування, що довів свою ефективність у реальних умовах. Отримані результати доповнені рекомендаціями щодо подальшого вдосконалення методу.

4. Позитивні сторони проекту: Запропонований метод значно підвищує ефективність процесів виявлення вразливостей завдяки інтеграції сучасних технологій штучного інтелекту. Він дозволяє автоматизувати рутинні задачі, скорочуючи час та ресурси, необхідні для аналізу безпеки. Метод також забезпечує адаптивність до змінних умов і масштабованість для великих проєктів, завдяки чому може бути застосований у широкому спектрі організацій. Особливо цінним є зниження впливу людського фактора, що мінімізує ризики помилок під час аналізу.

5. Негативні сторони проекту: Для ефективного впровадження методу потрібні значні обчислювальні ресурси, оскільки великі мовні моделі, такі як ChatGPT, мають високу

ВИМОГЛИВІСТЬ до апаратного забезпечення. Крім того, успішна реалізація потребує кваліфікованої команди фахівців із кібербезпеки, здатних інтегрувати цей метод у вже існуючі системи. Також є певні ризики пов'язані з етичними аспектами використання великих мовних моделей, зокрема їх можливе зловживання зловмисниками.

7. Відгук про роботу в цілому В загальному дипломна робота заслуговує позитивної оцінки. Весь матеріал дипломної роботи структурований, чіткий та послідовний. Усі розділи роботи послідовні та логічні, що дозволяє чітко розуміти викладений матеріал в рамках тематики дипломної роботи.

8. Інші зауваження

9. Оцінка дипломної роботи Розглянувши позитивні та негативні сторони представленої дипломної роботи, можна зробити висновок, що вона заслуговує оцінку «відмінно»

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)
Підченко Сергій Костянтинівич, завідувач кафедри ТМІТ, доктор
технічних наук, професор

« 16 » грудня 2024 .

 (підпис)