НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

До захисту допущено
Завідувач кафедри

_____ **Дмитро ЛАНДЕ**
(підпис)

«_____»_____2024 р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи, технології та математичні методи кібербезпеки»
спеціальності 125 «Кібербезпека»

на тему: Розробка та впровадження OSINT-фреймворку для збору та аналізу даних з відкритих джерел

Виконав (-ла): здобувач вищої освіти **IV** курсу, групи ФБ-01
(шифр групи)

Пітель Богдан Вікторович
(прізвище, ім'я, по батькові) (підпис)

Керівник д.т.н., професор, завідувач кафедри ІБ Ланде Дмитро Володимирович
(посада, науковий ступінь, вчене звання, прізвище, ім'я, по батькові) (підпис)

Рецензент
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище, ім'я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Здобувач вищої освіти _____
(підпис)

Київ – 2024 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
НАВЧАЛЬНО-НАУКОВИЙ ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – перший
(бакалаврський)Спеціальність – 125
«Кібербезпека»
Освітньо-професійна програма «Системи, технології та математичні методи кібербезпеки»

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____Дмитро ЛАНДЕ
(підпис)

«___»_____2024 р.

ЗАВДАННЯ

на дипломну роботу здобувачу вищої освіти

Пітелю Богдану Вікторовичу

1. Тема роботи: Розробка та впровадження OSINT-фреймворку для збору та аналізу даних з відкритих джерел, керівник роботи Ланде Дмитро Володимирович, д.т.н., професор, завідувач кафедри інформаційної безпеки. затверджені наказом по університету від «__»_____2024 р. №
2. Термін подання здобувачем вищої освіти роботи «__»_____2024 р.
3. Вихідні дані до роботи: Наукова література з розробки та впровадження OSINT-фреймворків, методів збору та аналізу даних з відкритих джерел, а також бібліотек для обробки інформації (networkx, json).
4. Зміст роботи: Робота містить створення Osint-фреймворку, завданням якого є пошук, аналіз та обробка інформації з подальшою візуалізацією зв’язків між різними об’єктами та суб’єктами.
5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо):Презентація до захисту дипломної роботи.
6. Дата видачі завдання: 30.10.2023

Календарний план

№з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів дипломної роботи	Примітка
1	Отримання завдання та узгодження теми роботи	30.10.2023	Виконано
2	Визначення структури дипломної роботи	20.02.2024 - 17.03.2024	Виконано
3	Дослідження методів збору даних з відкритих джерел (OSINT)	18.03.2024 - 24.03.2024	Виконано
4	Вивчення існуючих OSINT-інструментів та їх можливостей	25.03.2024 - 21.04.2024	Виконано
5	Програмна реалізація OSINT-фреймворку	22.04.2024 - 28.04.2024	Виконано
6	Тестування розробленого фреймворку	29.04.2024 - 19.05.2024	Виконано
7	Здійснення збору та аналізу даних з відкритих джерел за допомогою фреймворку	20.05.2024 - 26.05.2024	Виконано
8	Аналіз результатів	27.05.2024 - 09.06.2024	Виконано

Здобувач вищої освіти _____ Богдан ПІТЕЛЬ

(підпис) (**Власне ім'я, ПРІЗВИЩЕ**)

Керівник роботи _____ Дмитро ЛАНДЕ

(підпис) (**Власне ім'я, ПРІЗВИЩЕ**)

РЕФЕРАТ

Обсяг дипломної роботи 48 сторінок, 15 ілюстрацій, 1 додаток і 20 джерел літератури.

Об'єкт дослідження: Процеси збору та аналізу даних з відкритих джерел.

Методи дослідження: За допомогою ознайомлення з науковою літературою було знайдено інструменти, які я використовував у дослідженні.

Предмет дослідження: Методи та інструменти автоматизації збору та аналізу даних з відкритих джерел, а також розроблено програму, яка виконує візуалізації зв’язків між об’єктами.

Методи дослідження: У роботі використовуються методи аналізу наукової літератури, порівняльний аналіз існуючих інструментів OSINT, методи соціальної інженерії, програмної інженерії для розробки фреймворку, а також методи візуалізації даних для представлення результатів.

Наукова новизна одержаних результатів: Розроблений OSINT-фреймворк забезпечує автоматизований збір, обробку та аналіз даних з відкритих джерел, з можливістю візуалізації зв’язків між різними об’єктами та суб’єктами, що дозволяє значно підвищити ефективність роботи аналітиків та спеціалістів з інформаційної безпеки.

Практичне значення одержаних результатів: Впровадження розробленого OSINT-фреймворку дозволяє значно скоротити час на збір та аналіз даних з відкритих джерел, підвищити точність та достовірність отриманих результатів, зробити більш зручним поширення готових результатів, а також забезпечити візуалізацію зв’язків між об’єктами та суб’єктами.

Отримані результати: У роботі представлено розробку та впровадження OSINT-фреймворку, який дозволяє автоматизувати процес збору, обробки та аналізу даних з відкритих джерел. Проведено тестування та показано практичне застосування фреймворку, результати показали його ефективність та надійність. Розроблена програма може бути використаний в різних галузях для підвищення ефективності роботи з відкритими даними.

Ключові слова: OSINT, збір даних, аналіз даних, відкриті джерела, фреймворк, візуалізація даних.

ABSTRACT

Volume of the thesis: 48 pages, 15 illustrations, 1 appendice, and 20 literature sources.

Object of research: Processes of collecting and analyzing data from open sources. Research methods: By reviewing scientific literature, tools were found that I used in the research.

Subject of research: Methods and tools for automating the collection and analysis of data from open sources, as well as the development of a program that performs visualizations of connections between objects.

Research methods: The work uses methods of analyzing scientific literature, comparative analysis of existing OSINT tools, social engineering methods, software engineering for the framework development, and data visualization methods for presenting the results.

Scientific novelty of the obtained results: The developed OSINT framework provides automated collection, processing, and analysis of data from open sources, with the capability to visualize connections between different objects and subjects, which significantly enhances the efficiency of analysts and information security specialists.

Practical significance of the obtained results: The implementation of the developed OSINT framework significantly reduces the time for collecting and analyzing data from open sources, increases the accuracy and reliability of the results, makes it easier to share the ready results, and ensures visualization of connections between objects and subjects.

Obtained results: The work presents the development and implementation of an OSINT framework that automates the process of collecting, processing, and analyzing data from open sources. Testing and practical application of the framework were conducted, showing its effectiveness and reliability. The developed program can be used in various fields to improve the efficiency of working with open data.

Keywords: OSINT, data collection, data analysis, open sources, framework, data visualization.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

OSINT - Open Source Intelligence (розвідка на основі відкритих джерел)

GUI - Graphical User Interface (графічний інтерфейс користувача)

API - Application Programming Interface (інтерфейс прикладного програмування)

JSON - JavaScript Object Notation (стандартний формат обміну даними)

IP - Internet Protocol (інтернет-протокол)

NLP - Natural Language Processing (обробка природної мови)

PDF - Portable Document Format (портативний формат документа)

SAS - Statistical Analysis System (система статистичного аналізу)

SPSS - Statistical Package for the Social Sciences (пакет статистичного аналізу для соціальних наук)

ВСТУП

В умовах сучасного інформаційного суспільства обсяги відкритих даних постійно зростають, і їх ефективне використання може забезпечити значну конкурентну перевагу в різних галузях, таких як інформаційна безпека, аналітика ринків, журналістика та багато інших. Збір та аналіз даних з відкритих джерел (Open Source Intelligence, OSINT) стає все більш важливим елементом у процесах прийняття рішень. Однак, зростання обсягу даних та їх різноманітність ставлять перед дослідниками нові виклики, які потребують розробки ефективних інструментів для автоматизації цих процесів. Тема розробки та впровадження OSINT-фреймворку є надзвичайно актуальною, оскільки дозволяє вирішити ці завдання шляхом створення системи, здатної автоматично збирати, обробляти та аналізувати дані з різних відкритих джерел.

Актуальність теми роботи: Сучасний світ характеризується великою кількістю доступної інформації, яка міститься у відкритих джерелах, таких як інтернет-ресурси, соціальні мережі, новинні сайти тощо. Використання цієї інформації може надати вагомі переваги в різних сферах діяльності. Наприклад, у сфері інформаційної безпеки OSINT дозволяє виявляти потенційні загрози та аналізувати поведінку зловмисників, у бізнесі - відстежувати конкурентів та аналізувати ринкові тенденції, а в журналістиці - збирати фактичну інформацію для проведення розслідувань. Однак, обсяг даних, що підлягають аналізу, може бути величезним, і їх ручне опрацювання стає надзвичайно трудомістким та малоефективним. Це зумовлює необхідність розробки спеціалізованих інструментів, які б автоматизували процес збору та аналізу інформації, роблячи його швидшим та точнішим.

Загальний аналіз стану проблеми: Станом на сьогодні, існує велика кількість інструментів та методів для збору даних з відкритих джерел, таких як Maltego, Shodan, TheHarvester та інші. Кожен з цих інструментів має свої особливості та обмеження. Наприклад, деякі з них орієнтовані на збір технічної інформації (IP-адреси, домени), інші - на аналіз соціальних мереж. Проте, часто виникає потреба у використанні декількох інструментів одночасно, що ускладнює

процес збору та аналізу даних. Крім того, більшість існуючих інструментів не надають можливості зручної візуалізації зв’язків між об’єктами, що ускладнює інтерпретацію отриманих результатів.

Основні задачі, з вирішенням яких було пов'язане виконання роботи: Метою даної роботи є розробка та впровадження OSINT-фреймворку, який забезпечить автоматизований збір, обробку та аналіз даних з відкритих джерел, з можливістю візуалізації зв’язків між різними об’єктами та суб’єктами. Для досягнення цієї мети були поставлені наступні задачі:

1. Дослідження існуючих методів та інструментів збору даних з відкритих джерел.
2. Розробка архітектури OSINT-фреймворку, яка дозволить інтегрувати різні методи збору та обробки даних.
3. Програмна реалізація основних компонентів фреймворку, включаючи модулі для збору, обробки та аналізу даних, а також інтерфейс користувача.
4. Тестування розробленого фреймворку для забезпечення його коректної роботи в різних сценаріях.
5. Проведення збору та аналізу даних з використанням розробленого фреймворку для оцінки його ефективності та надійності.
6. Візуалізація результатів аналізу даних для полегшення інтерпретації та прийняття рішень на основі отриманих даних.

Таким чином, дана робота спрямована на розробку інноваційного інструменту для збору та аналізу даних з відкритих джерел, який дозволить значно підвищити ефективність роботи аналітиків та спеціалістів з інформаційної безпеки, забезпечивши їм зручні засоби для отримання та інтерпретації необхідної інформації.

1 ТЕОРЕТИЧНІ ОСНОВИ OSINT

1.1 Вступ до OSINT

Open Source Intelligence (OSINT), або розвідка на основі відкритих джерел, є процесом збору та аналізу інформації з відкритих, публічно доступних джерел для отримання корисних розвідувальних даних. Цей термін охоплює широкий спектр джерел, включаючи Інтернет, соціальні мережі, новинні сайти, урядові звіти, наукові публікації та багато іншого.

1.1.1 Історія OSINT

Історично OSINT виник як підхід до збору інформації в рамках військової розвідки, але з розвитком Інтернету та інформаційних технологій він набув значного поширення в цивільних сферах. Перші спроби систематизації та використання відкритих джерел у розвідувальній діяльності були зроблені під час Першої та Другої світових війн. Проте справжній прорив у використанні OSINT стався в епоху цифрової революції, коли доступ до інформації значно спростився, а її кількість зросла в геометричній прогресії.

1.1.2 Значення та актуальність використання OSINT у сучасному світі

У сучасному світі OSINT має велике значення завдяки доступності та багатству інформації, яку можна отримати з відкритих джерел. Це дозволяє організаціям та індивідам отримувати актуальні дані, необхідні для прийняття стратегічних рішень. Основні переваги OSINT включають низьку вартість, швидкість доступу до інформації та можливість отримання даних з різних джерел для всебічного аналізу.

1.1.3 Застосування OSINT у розвідці

OSINT надає розвідувальним органам, як державним, так і приватним, необхідну інформацію для розуміння діяльності конкурентів та аналізу об'єктивних і суб'єктивних факторів, що впливають на цю діяльність. Відкрита інформація використовується у поєднанні з іншими ресурсами, такими як

агентурні дані, для підвищення ефективності розвідувальної діяльності. Як зазначається у книзі “Конкурентна розвідка” А.Г. Додонова, Д.В. Ланде, В.В. Прищепи та В.Г. Путятіна: "Відкриті джерела містять величезну кількість інформації, яка потрібна і задовольняє вимогам розвідувальних органів, як державних, так і приватних, комерційних, що забезпечує розуміння об'єктивних і суб'єктивних факторів, пов'язаних, наприклад, з діяльністю конкурентів".

1.1.4 Застосування OSINT у силах безпеки та оборони України

У сучасних умовах OSINT відіграє важливу роль у забезпеченні безпеки та оборони України. Відкрита інформація з різних джерел дозволяє українським силам оборони оперативно реагувати на загрози та приймати стратегічні рішення.

Одним із яскравих прикладів ефективного використання OSINT в для оборони України є діяльність волонтерських Osint організацій. Вони активно використовують методи OSINT для збору та аналізу інформації про переміщення військ противника, його дії та інші важливі події.

1.1.4 Застосування OSINT у бізнесі

У бізнесі використовується комерційна розвідка, інтелектуальна розвідка та бізнес-аналітика. Компаніям потрібен OSINT для збору та аналізу інформації з відкритих джерел, що допомагає їм розуміти ринкові тенденції, аналізувати конкурентів та приймати обґрунтовані рішення. Відкрита інформація також використовується для моніторингу репутації компанії. Завдяки OSINT, бізнес може своєчасно адаптуватися до змін ринкового середовища і залишатися конкурентоспроможним.

1.2 Методи збору даних з відкритих джерел

1.2.1 Класифікація джерел даних

Дані з відкритих джерел можуть бути класифіковані за різними критеріями, залежно від типу джерела та його доступності. Основні типи джерел даних включають:

- Онлайн-джерела: веб-сайти, блоги, форуми, соціальні мережі, новинні портали.
- Офіційні документи: урядові звіти, публічні записи, патенти, наукові публікації.
- Мультимедійні джерела: відео, аудіо, зображення.
- Бази даних: комерційні бази даних, публічні реєстри.
- Спеціалізовані джерела: професійні журнали, галузеві дослідження.

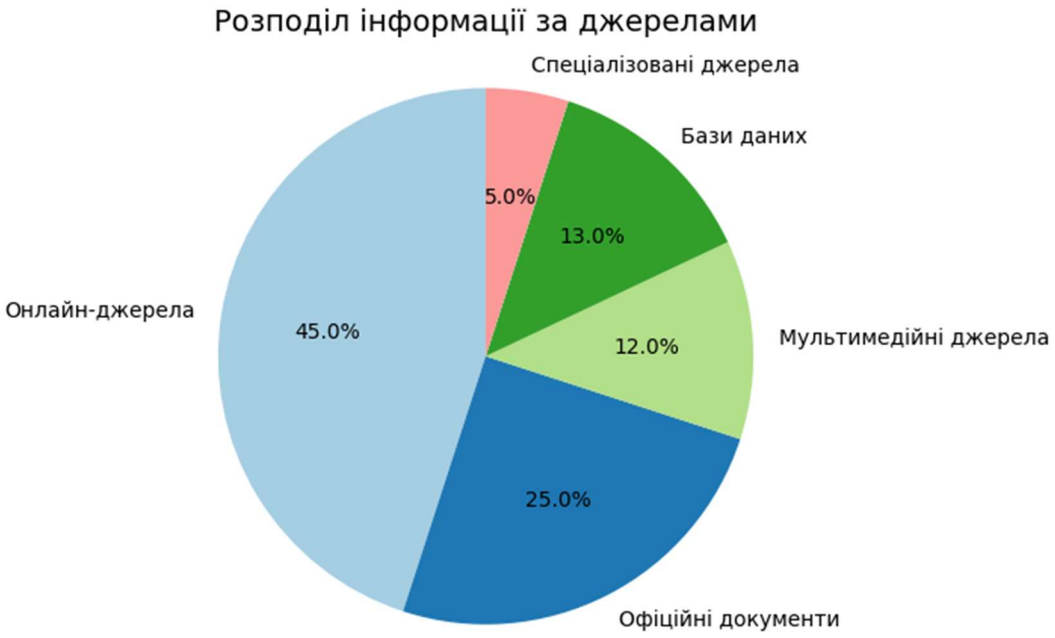

Рисунок 1.1 – Розподіл інформації за джерелами

1.2.2 Основні інструменти та технології для збору даних

Для збору даних з відкритих джерел застосовуються різні інструменти та технології, що забезпечують ефективність та точність отриманої інформації:

1. Пошукові системи: Google, Bing, Yahoo, які дозволяють знаходити інформацію за ключовими словами та фразами.
2. Соціальні медіа інструменти: спеціалізовані програми для моніторингу та аналізу соціальних мереж, такі як Hootsuite, TweetDeck.
3. Парсери: автоматизовані програми для збору даних з веб-сайтів (наприклад, BeautifulSoup для Python).
4. Аналіз тексту: технології Text Mining та Natural Language Processing (NLP) для обробки та аналізу текстової інформації.
5. Візуалізація даних: інструменти для представлення даних у зрозумілому форматі, такі як Tableau, Power BI.

Кількість інструментів для збору та аналізу даних постійно зростає, що дозволяє підвищити ефективність роботи з відкритими джерелами. Нові технології та програми полегшують процес збору, обробки та візуалізації даних, роблячи його більш доступним для дослідників та аналітиків.

1.2.3 Особливості збору даних з різних типів джерел

Збір даних з різних типів джерел має свої особливості, які варто враховувати для досягнення максимальних результатів. При роботі з веб-сайтами важливо враховувати частоту оновлення інформації, достовірність джерела та можливість автоматичного збору даних. Соціальні мережі потребують особливої уваги до контексту повідомлень, настроїв користувачів та потенційного впливу фейкових новин. Офіційні документи зазвичай є найбільш достовірними джерелами, але можуть вимагати додаткової перевірки та аналізу.

Мультимедійні дані, такі як відео, аудіо та зображення, вимагають спеціальних інструментів для обробки та аналізу контенту. Збір даних з баз даних забезпечує великий обсяг структурованої інформації, але може бути обмежений у доступі або вимагати підписки. Спеціалізовані джерела, такі як професійні журнали та галузеві дослідження, надають високоякісну інформацію, але доступ до них може бути обмежений через платний доступ або підписку.

Таким чином, кожен тип джерела даних має свої унікальні характеристики та виклики, які слід враховувати при зборі та аналізі інформації. Використання відповідних інструментів та методів дозволяє ефективно збирати та аналізувати дані з різних джерел, забезпечуючи обґрунтовані висновки та рекомендації.

1.3 Соціальна інженерія як компонент OSINT

1.3.1 Визначення соціальної інженерії

Соціальна інженерія – це методи маніпулювання людьми з метою отримання конфіденційної або чутливої інформації. Цей підхід базується на психологічних маніпуляціях, які використовуються для обману людей, змушуючи їх розкривати секрети або надавати доступ до систем і ресурсів, що зазвичай залишаються захищеними. Соціальна інженерія може включати різні форми обману, такі як фішинг, вішинг, та інші методи, які використовуються для отримання доступу до інформації або ресурсів.

1.3.2 Методи соціальної інженерії

Соціальна інженерія використовує різні методи для досягнення своїх цілей. Основні методи включають:

1. Фішинг: надсилання підроблених електронних листів або повідомлень, які імітують офіційні запити від відомих організацій з метою отримання особистих даних або доступу до систем.

2. Вішинг: телефонні дзвінки, під час яких зловмисник видає себе за довірену особу (наприклад, співробітника банку) для отримання конфіденційної інформації.
3. Смишинг: надсилання фальшивих SMS-повідомлень з метою отримання особистих даних або доступу до мобільних пристроїв.
4. Pretexting (Пре-текстинг): створення фальшивого сценарію або історії, яка змушує цільову особу розкрити конфіденційну інформацію.
5. Baiting (Наживка): залишення фізичних пристроїв (наприклад, флеш-накопичувачів) у загальнодоступних місцях з метою, щоб хтось знайшов і використав їх, надавши доступ до системи зловмисникам.
6. Tailgating (Підслідування): фізичне проникнення в захищену зону шляхом слідування за уповноваженою особою.

Фішинг продовжує бути однією з найпоширеніших форм кіберзлочинності. За даними експертів, кожного дня зловмисники розсилають близько 3,4 мільярда шкідливих електронних листів, при цьому Google блокує приблизно 100 мільйонів з них. Згідно зі статистикою Cisco Umbrella, фішингові атаки становлять 46% від усіх атак на організації, і 96% цих атак орієнтовані на корпоративні електронні адреси.

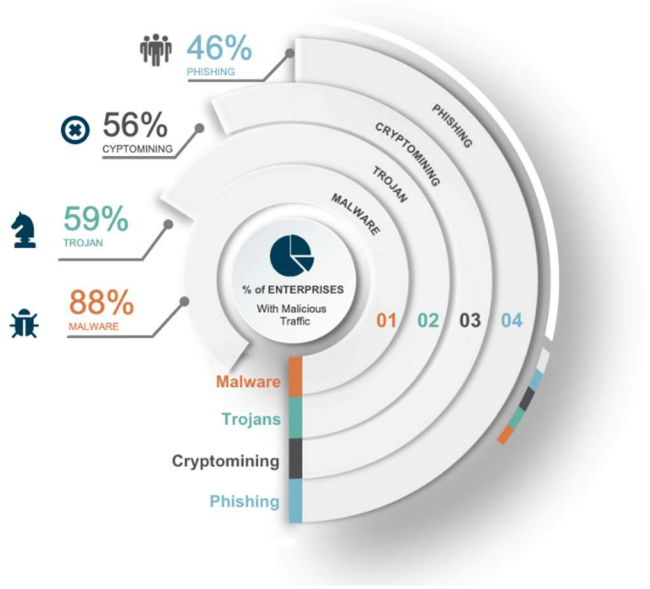

Рисунок 1.2 – Популярні форми кіберзлочинності

1.3.3 Взаємозв'язок соціальної інженерії та OSINT

Соціальна інженерія та OSINT тісно взаємопов'язані, оскільки інформація, зібрана за допомогою OSINT, може використовуватися для підготовки та проведення атак соціальної інженерії. Використовуючи відкриті джерела інформації, зловмисники можуть отримати детальну інформацію про особу або організацію, що дозволяє їм створювати більш переконливі атаки. Наприклад, вивчивши профілі в соціальних мережах, зловмисник може дізнатися про звички, інтереси та коло спілкування жертви, що допоможе йому створити фішингове повідомлення, яке буде виглядати більш правдоподібно.

Таким чином, соціальна інженерія є важливим компонентом OSINT, оскільки інформація, зібрана з відкритих джерел, дозволяє зловмисникам ефективніше маніпулювати людьми та отримувати доступ до конфіденційної інформації.

1.4 Процеси обробки та аналізу зібраної інформації

1.4.1 Методи обробки зібраних даних

Методи обробки зібраних даних включають різні техніки, такі як фільтрація, сортування, агрегація та класифікація. Фільтрація допомагає видалити нерелевантні дані, сортування організовує дані в зручному порядку, агрегація об'єднує дані для отримання узагальненої інформації, а класифікація категоризує дані для подальшого аналізу.

1.4.2 Інструменти для аналізу даних

Інструменти для аналізу даних варіюються від простих програмних засобів до комплексних платформ, які забезпечують глибокий аналіз великих обсягів даних. Деякі з найбільш популярних інструментів включають:

1. Excel: використовується для базового аналізу даних, побудови графіків і таблиць.
2. R і Python: мови програмування, які пропонують широкі можливості для статистичного аналізу та візуалізації даних.
3. SPSS і SAS: програми для статистичного аналізу, які використовуються в академічних та комерційних дослідженнях.
4. Tableau і Power BI: платформи для візуалізації даних, що дозволяють створювати інтерактивні графіки та панелі.

1.4.3 Візуалізація результатів аналізу

Візуалізація результатів аналізу є важливою частиною процесу обробки даних, оскільки вона дозволяє зрозуміло представити складні дані та результати аналізу. Використання графіків, діаграм, карт і панелей допомагає перетворити числові дані на візуальні елементи, які легко зрозуміти і інтерпретувати. Популярні інструменти для візуалізації даних, такі як Tableau, Power BI, і Python бібліотеки (наприклад, Matplotlib, Seaborn), забезпечують широкий спектр можливостей для створення ефективних візуальних представлень даних.

1.5 Виклики та обмеження в OSINT

1.5.1 Проблеми точності та достовірності даних

Однією з основних проблем в OSINT є питання точності та достовірності зібраних даних. Відкриті джерела інформації часто містять велику кількість шуму, дезінформації та фейкових новин, що ускладнює процес відбору надійних даних. Також існує ризик отримання застарілої інформації, яка більше не є актуальною або точною. Використання методів валідації даних, таких як перевірка джерел та перехресний аналіз, допомагає мінімізувати ці ризики, але повністю уникнути їх неможливо.

1.5.2 Етичні та правові аспекти використання OSINT

Етичні та правові аспекти є важливими факторами, які необхідно враховувати при використанні OSINT. Збір і аналіз інформації з відкритих джерел може порушувати конфіденційність і права на приватність окремих осіб та організацій. Важливо дотримуватися законів і нормативних актів, що регулюють захист даних, авторські права та права на конфіденційність. Етичні питання включають відповідальне використання інформації, уникнення маніпуляцій та дезінформації, а також повагу до прав та інтересів суб'єктів інформації.

1.5.3 Майбутні тенденції та перспективи розвитку OSINT

Майбутні тенденції в розвитку OSINT передбачають подальше вдосконалення технологій збору та аналізу даних. Зокрема, розвиток штучного інтелекту та машинного навчання дозволить автоматизувати процеси обробки великих обсягів даних і підвищити точність аналізу. Зростатиме використання нових джерел інформації, таких як Інтернет речей (IoT) та дані з соціальних мереж, що дозволить отримувати ще більш детальну та актуальну інформацію.

Перспективи розвитку OSINT також включають інтеграцію з іншими видами розвідки, такими як сигнальна розвідка (SIGINT) та розвідка на основі зображень (IMINT), що дозволить створювати більш комплексні та всебічні розвідувальні продукти. Однак, з розвитком технологій та збільшенням обсягів даних зростатимуть і виклики, пов'язані з безпекою та конфіденційністю інформації, що вимагатиме постійного вдосконалення методів захисту даних та етичних стандартів.

Висновки до розділу 1

У розділі 1 було розглянуто ключові теоретичні поняття та аспекти OSINT (Open Source Intelligence), або розвідки на основі відкритих джерел.

OSINT визначено як процес збору та аналізу інформації з відкритих, публічно доступних джерел для отримання корисних розвідувальних даних. Історичний розвиток OSINT показує його походження у військовій розвідці та значне поширення у цивільних сферах з розвитком Інтернету та інформаційних технологій.

Значення та актуальність OSINT у сучасному світі пояснюється доступністю та багатством інформації, яка дозволяє організаціям та індивідам приймати стратегічні рішення на основі актуальних даних. OSINT використовується в різних сферах, включаючи розвідку, безпеку та оборону, а також бізнес, де він допомагає у зборі та аналізі інформації про ринкові тенденції, конкуренті та інші важливі аспекти.

Методи збору даних з відкритих джерел включають використання різних інструментів та технологій, таких як пошукові системи, соціальні медіа інструменти, парсери, технології аналізу тексту та інструменти для візуалізації даних. Особливості збору даних з різних типів джерел підкреслюють важливість врахування специфічних характеристик кожного джерела для досягнення максимальних результатів.

Соціальна інженерія як компонент OSINT показує, як інформація, зібрана з відкритих джерел, може бути використана для підготовки та проведення атак, маніпулюючи людьми для отримання конфіденційної інформації.

Процеси обробки та аналізу зібраної інформації включають різні методи, такі як фільтрація, сортування, агрегація та класифікація, що допомагають організувати та проаналізувати дані для отримання обґрунтованих висновків.

Виклики та обмеження в OSINT включають проблеми точності та достовірності даних, етичні та правові аспекти, а також майбутні тенденції та перспективи розвитку. Зокрема, розвиток штучного інтелекту та машинного

навчання обіцяє підвищити ефективність та точність аналізу даних, що відкриває нові можливості для використання OSINT у різних сферах.

OSINT є важливим інструментом у сучасному світі, який дозволяє отримувати цінну інформацію з відкритих джерел та використовувати її для прийняття обґрунтованих рішень у різних галузях. Це підкреслює необхідність постійного вдосконалення методів збору та аналізу даних, а також дотримання етичних та правових норм у цій сфері.

2 РОЗРОБКА OSINT-ФРЕЙМВОРКУ

2.1 Вибір інструментів та технологій

Для розробки OSINT-фреймворку використано кілька ключових інструментів, які забезпечують ефективну роботу з даними з відкритих джерел, а також зручність і продуктивність розробки. Основні критерії для вибору інструментів включали їх функціональність, надійність, продуктивність та підтримку в спільноті розробників.

Python вибрано як основну мову програмування через його простоту, величезну кількість бібліотек для обробки даних та розробки інтерфейсу користувача, а також через можливість запуску на різних операційних системах.

Для створення графічного інтерфейсу користувача (GUI) обрано бібліотеку Tkinter, яка є стандартною для Python. Tkinter надає широкий набір функцій для розробки зручних інтерфейсів, включаючи можливості для створення інтерактивних вікон, форм, кнопок та інших елементів.

Для побудови та візуалізації графів обрано бібліотеку NetworkX, яка забезпечує зручні інструменти для створення, маніпуляції та аналізу складних графів. NetworkX підтримує різні формати збереження графів, що дозволяє легко зберігати та завантажувати дані, і надає функції для візуалізації зв’язків між об’єктами.

Для візуалізації даних на графіках та діаграмах обрано бібліотеку Matplotlib, яка надає багатий набір інструментів для створення різноманітних типів графіків та діаграм, що дозволяє ефективно візуалізувати дані, зібрані з відкритих джерел.

2.2 Проектування архітектури фреймворку

2.2.1 Визначення основних компонентів фреймворку

Архітектура OSINT-фреймворку складається з кількох основних компонентів, кожен з яких виконує певні функції та забезпечує загальну функціональність системи. Основні компоненти фреймворку включають:

1. **Модуль збору даних**: Відповідає за автоматизований збір даних з різних відкритих джерел, таких як веб-сайти, соціальні мережі, бази даних та інші публічні ресурси.

```python
def search_google():
    query = f'"{node}"'
    webbrowser.open(f"https://www.google.com/search?q={query}")

def search_pdf():
    query = f'"{node}" filetype:pdf'
    webbrowser.open(f"https://www.google.com/search?q={query}")

def search_youtube():
    query = f'"{node}"'
    webbrowser.open(f"https://www.youtube.com/results?search_query={query}")

def search_telegram(phone):
    webbrowser.open(f"tg://resolve?phone={phone}")

def search_viber(phone):
    webbrowser.open(f"https://viber.click/{phone}")

def search_whatsapp(phone):
    webbrowser.open(f"https://wa.me/{phone}")

def search_archive(site):
    if site:
        archive_url = f"https://wayback-api.archive.org/web/20240000000000*/{site}"
        webbrowser.open(archive_url)

def search_hibp():
    hibp_url = "https://haveibeenpwned.com/"
    webbrowser.open(hibp_url)

def search_google_maps(coordinates):
    webbrowser.open(f"https://www.google.com/maps/search/?api=1&query={coordinates}")

def search_yandex_maps(coordinates):
    webbrowser.open(f"https://yandex.com/maps/?text={coordinates}")

def search_bing_maps(coordinates):
    lat, lon = coordinates.split(',')
    webbrowser.open(f"https://www.bing.com/maps?q={lat},{lon}")
```

Рисунок 2.1 – Модуль збору даних з відкритих джерел, реалізовані у фреймворку

2. **Модуль обробки даних**: Здійснює первинну обробку зібраних даних, включаючи фільтрацію, нормалізацію та структурування інформації для подальшого аналізу.

```
edit_window = tk.Toplevel(root)
edit_window.title("Edit Target Info")
edit_window.geometry("800x550")

ttk.Label(edit_window, text="Target Info:").pack(pady=5)
entry_info = ttk.Entry(edit_window, width=50)
entry_info.insert(0, node_info)
entry_info.pack(pady=5)

contact_label_text = "Phone Number\nInternational Format:" if target_type == 'Person' else "Coordinates:"
ttk.Label(edit_window, text=contact_label_text).pack(pady=5)
vcmd = (edit_window.register(validate_contact_input), '%d', '%i', '%P', '%s', '%S', '%v', '%V', '%W')
entry_contact = ttk.Entry(edit_window, width=50, validate='key', validatecommand=vcmd)
entry_contact.insert(0, node_contact)
entry_contact.pack(pady=5)

ttk.Label(edit_window, text="Email:").pack(pady=5)
entry_email = ttk.Entry(edit_window, width=50)
entry_email.insert(0, node_email)
entry_email.pack(pady=5)

ttk.Label(edit_window, text="Site:").pack(pady=5)
entry_site = ttk.Entry(edit_window, width=50)
entry_site.insert(0, node_site)
entry_site.pack(pady=5)
```

Рисунок 2.2 – Частина модулю, який відповідає за обробку даних

3. **Модуль візуалізації даних**: Забезпечує візуалізацію результатів аналізу у вигляді графіків, діаграм та інтерактивних візуалізацій.

```
def visualize_graph():
    plt.clf()
    pos = nx.spring_layout(graph)

    person_nodes = [node for node in graph.nodes if graph.nodes[node].get('type') == 'Person']
    object_nodes = [node for node in graph.nodes if graph.nodes[node].get('type') == 'Object']

    person_colors = [graph.nodes[node].get('color', 'lightblue') for node in person_nodes]
    object_colors = [graph.nodes[node].get('color', 'lightblue') for node in object_nodes]

    nx.draw(graph, pos, nodelist=person_nodes, node_color=person_colors, node_size=3000, edge_color='gray', font_size=10, font_weight='bold', arr
    nx.draw(graph, pos, nodelist=object_nodes, node_color=object_colors, node_shape='s', node_size=3000, edge_color='gray', font_size=10, font_we

    # Додавання підписів до вузлів
    nx.draw_networkx_labels(graph, pos, font_size=10, font_weight='bold')

    visualize_graph.pos = pos
    canvas.draw()
```

Рисунок 2.3 – Частина модулю візуалізації даних

4. **Графічний інтерфейс користувача (GUI)**: Надає зручний інтерфейс для взаємодії користувача з фреймворком, управління процесами збору та аналізу даних, а також перегляду результатів.

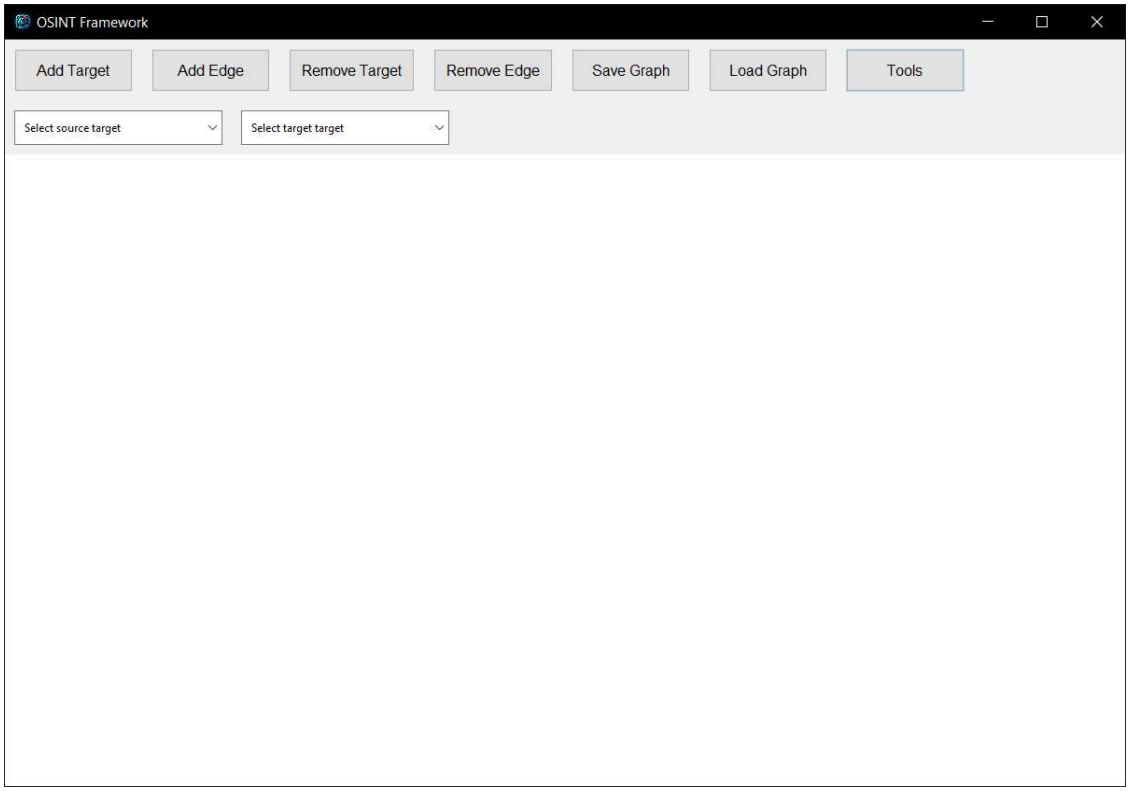

Рисунок 2.4 – Графічний інтерфейс Osint-фреймворку

2.3 Інтерфейс користувача

Інтерфейс користувача (GUI) є ключовим компонентом OSINT-фреймворку, що забезпечує зручність і ефективність взаємодії користувача з системою. Для розробки GUI було обрано бібліотеку Tkinter, яка є стандартною для створення графічних інтерфейсів у Python. Вибір Tkinter зумовлений її простотою використання, широкими можливостями для створення різноманітних елементів інтерфейсу та активною підтримкою спільноти розробників.

Основне вікно програми складається з кількох ключових елементів, які забезпечують повний функціонал для управління процесами збору, обробки, аналізу та візуалізації даних. Панель кнопок розташована у верхній частині вікна і включає кнопки для додавання цілей, додавання зв'язків, видалення цілей, видалення зв'язків, збереження та завантаження графу, а також відкриття вікна з інструментами. Кожна кнопка прив'язана до відповідної функції, що дозволяє користувачу виконувати необхідні дії натисканням однієї кнопки.

Полотно для візуалізації графу інтегроване з бібліотекою Matplotlib і забезпечує відображення графу у вигляді зрозумілих графічних структур. Цілі та зв'язки графу відображаються з різними кольорами та формами залежно від їх типу, що дозволяє користувачу легко розрізняти різні типи об'єктів. Візуалізація оновлюється автоматично при додаванні або видаленні цілей та зв'язків, що забезпечує динамічне відображення змін у графі.

Інтеграція функціональних модулів у GUI здійснюється за допомогою функцій-обробників, які викликаються при взаємодії користувача з елементами інтерфейсу. Наприклад, при натисканні на кнопку додавання цілі відкривається діалогове вікно, де користувач може ввести інформацію про нову ціль, включаючи назву, тип, контактну інформацію, електронну пошту, сайт та колір цілі. Після підтвердження введеної інформації нова ціль додається до графу, і візуалізація графу оновлюється для відображення нової цілі.

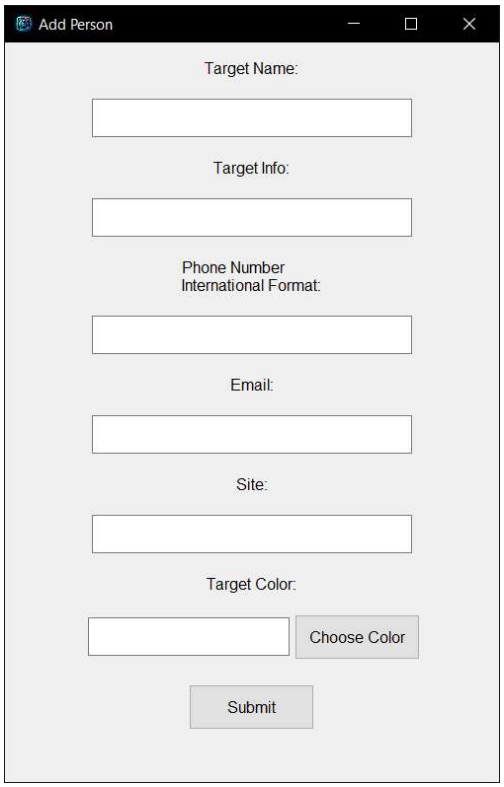

Рисунок 2.5 – Графічний інтерфейс вікна додавання нової цілі

Додавання зв'язків здійснюється через вибір джерела та цільового об'єкта з комбобоксів і натискання кнопки додавання зв'язку. Це створює зв’язок між вибраними цілями у графі та оновлює візуалізацію для відображення нового зв'язку.

Функції збереження та завантаження графу дозволяють користувачу зберігати поточний стан графу у файл JSON та завантажувати граф з файлу. Це забезпечує можливість зберігати результати роботи та продовжувати роботу з графом у майбутньому.

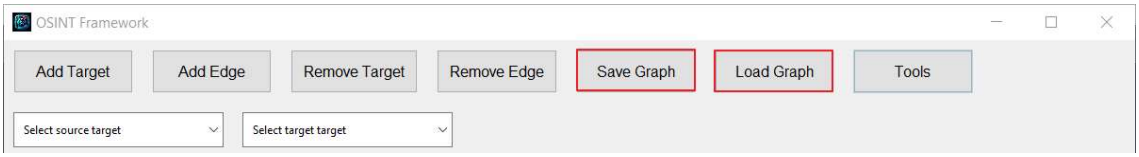

Рисунок 2.6 – Функції збереження та завантаження графу

Зручність та інтуїтивність інтерфейсу забезпечуються через декілька ключових принципів. Елементи управління розташовані у логічному порядку, що дозволяє користувачу легко знаходити необхідні функції. Використання діалогових вікон для введення та редагування даних дозволяє користувачу фокусуватися на поточному завданні, не відволікаючись на інші елементи інтерфейсу.

Валідація введених даних забезпечує коректність введеної інформації, запобігаючи введенню недопустимих значень. Наприклад, при введенні контактної інформації або координат здійснюється перевірка на допустимість символів. Контекстне меню для полів введення дозволяє користувачу використовувати функції копіювання, вставки та вирізання, що підвищує зручність введення та редагування інформації.

Інтерфейс підтримує інтерактивність, що дозволяє користувачу взаємодіяти з графом, натискаючи на об’єкти для перегляду та редагування їх інформації. Це робить процес роботи з графом більш наочним та зручним. Інтерфейс також надає

зворотний зв’язок користувачу у вигляді повідомлень про успішне виконання дій або помилок, що допомагає користувачу розуміти, що відбувається в системі та які дії потрібно виконати для вирішення проблем.

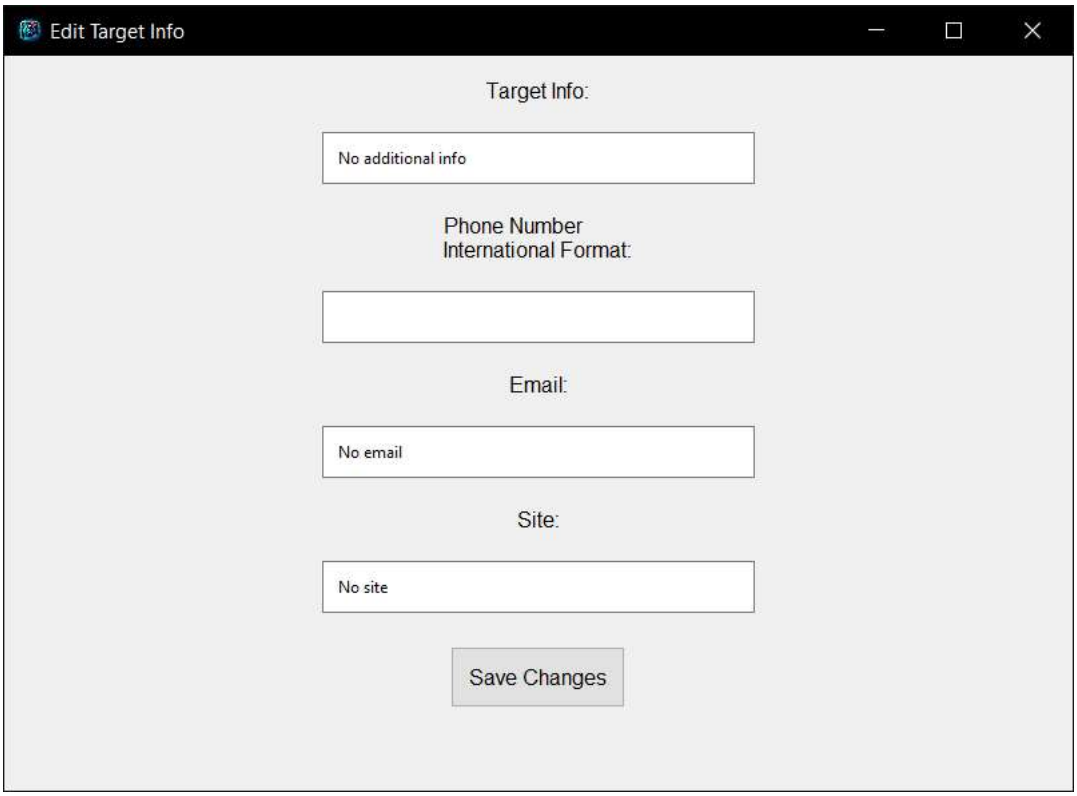

Рисунок 2.7 – Графічний інтерфейс вікна редагування інформації

У вікні інформації про об'єкт знаходиться корисний функціонал, що включає наступні кнопки:

1. Редагування інформації:
    - Натискання цієї кнопки відкриває діалогове вікно для редагування інформації про об'єкт, включаючи контактні дані, електронну пошту, вебсайт та інші деталі. Після збереження змін, граф автоматично оновлюється.
2. Копіювання до буфера обміну:
    - Копіювати інформацію: Копіює додаткову інформацію про об'єкт до буфера обміну.

- Копіювати контактні дані: Копіює контактну інформацію (номер телефону або координати) до буфера обміну.
- Копіювати електронну пошту: Копіює електронну пошту до буфера обміну.
- Копіювати вебсайт: Копіює URL вебсайту до буфера обміну.

3. Онлайн пошук:
    - Пошук у Google: Відкриває нову вкладку у браузері з результатами пошуку у Google за назвою об'єкта.
    - Пошук PDF: Відкриває нову вкладку у браузері з результатами пошуку PDF-документів у Google за назвою об'єкта.
    - Пошук на YouTube: Відкриває нову вкладку у браузері з результатами пошуку на YouTube за назвою об'єкта.
4. Пошук у соціальних мережах:
    - Пошук у Telegram: Відкриває Telegram для пошуку за номером телефону об'єкта.
    - Пошук у Viber: Відкриває Viber для пошуку за номером телефону об'єкта.
    - Пошук у WhatsApp: Відкриває WhatsApp для пошуку за номером телефону об'єкта.
5. Пошук на карті:
    - Пошук у Google Maps: Відкриває Google Maps для пошуку за координатами об'єкта.
    - Пошук у Yandex Maps: Відкриває Yandex Maps для пошуку за координатами об'єкта.
    - Пошук у Bing Maps: Відкриває Bing Maps для пошуку за координатами об'єкта.
6. Пошук в архіві вебсайтів:
    - Пошук у Wayback Machine: Відкриває Wayback Machine для пошуку архівованих версій вебсайту об'єкта.
7. Перевірка безпеки:

- Перевірка у HIBP: Відкриває сайт "Have I Been Pwned?" для перевірки, чи був зламаний електронний адрес об'єкта.

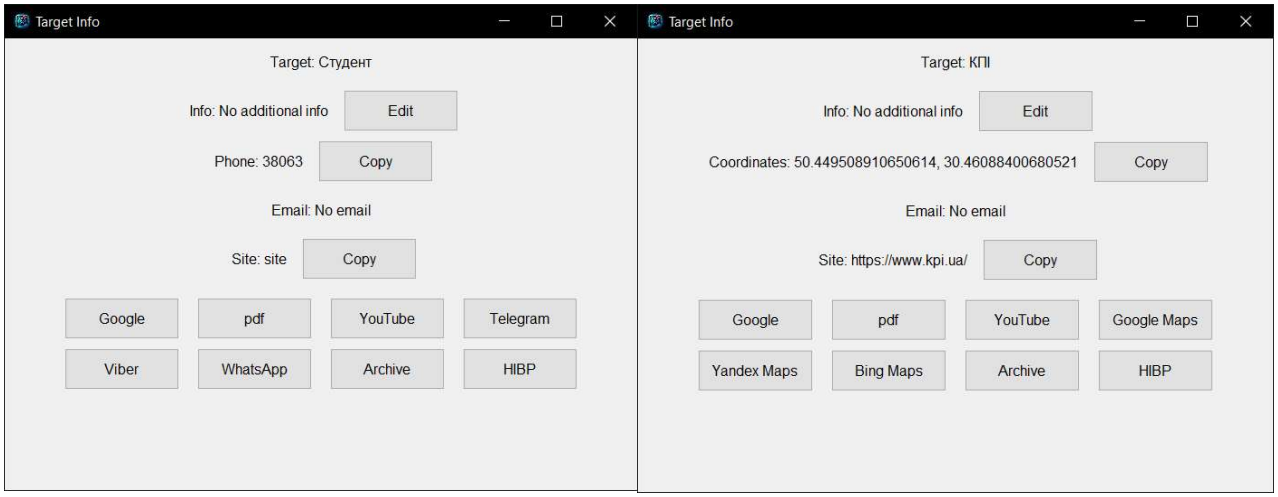

Рисунок 2.8 – Демонстрація функціоналу у вікні інформації про об’єкт дослідження

Користувач може додати новий інструмент, заповнивши форму з полями для назви інструменту, опису, посилання та вибору категорії. Після підтвердження новий інструмент додається до списку інструментів.

Функція управління категоріями дозволяє користувачу додавати та видаляти категорії інструментів, що забезпечує структурування та організацію інструментів за тематикою або призначенням. Це дозволяє легше орієнтуватися у великому наборі інструментів. Функції експорту та імпорту дозволяють зберігати список інструментів у файл JSON та завантажувати його з файлу. Це зручно для передачі налаштувань між різними системами або збереження резервних копій.

Рисунок 2.9 – Імпортовані інструменти

Таким чином, розробка та інтеграція графічного інтерфейсу користувача забезпечують зручну та ефективну взаємодію користувача з OSINT-фреймворком. Це дозволяє швидко та інтуїтивно виконувати всі необхідні дії для збору, обробки, аналізу та візуалізації даних з відкритих джерел, підвищуючи ефективність роботи користувача та загальну продуктивність системи.

Висновки до розділу 2

Розробка OSINT-фреймворку включала кілька ключових етапів, починаючи з вибору інструментів та технологій і закінчуючи створенням зручного та інтуїтивного графічного інтерфейсу користувача.

Для розробки OSINT-фреймворку було обрано Python як основну мову програмування завдяки його простоті, широкому набору бібліотек та кросплатформеності. Використання бібліотеки Tkinter для створення графічного

інтерфейсу користувача дозволило реалізувати зручний та функціональний GUI з багатими можливостями для створення інтерактивних елементів. Бібліотека NetworkX забезпечила інструменти для створення, маніпуляції та аналізу складних графів, тоді як Matplotlib використовувалася для візуалізації даних у вигляді різноманітних графіків та діаграм. Ці інструменти були обрані з урахуванням їх функціональності, надійності та підтримки в спільноті розробників.

Архітектура OSINT-фреймворку складається з кількох основних компонентів, кожен з яких виконує певні функції та забезпечує загальну функціональність системи. Основні компоненти включають модулі збору даних, обробки даних, візуалізації даних та графічного інтерфейсу користувача. Модуль збору даних автоматизує процес отримання інформації з відкритих джерел, таких як веб-сайти та соціальні мережі. Модуль обробки даних здійснює первинну обробку зібраної інформації, включаючи фільтрацію та структурування даних. Модуль візуалізації даних представляє результати аналізу у вигляді графіків та інтерактивних візуалізацій. Графічний інтерфейс користувача надає зручний спосіб взаємодії з фреймворком, дозволяючи управляти процесами збору та аналізу даних, а також переглядати результати.

Інтерфейс користувача є ключовим компонентом OSINT-фреймворку, що забезпечує зручність і ефективність взаємодії користувача з системою. Для створення GUI було обрано бібліотеку Tkinter, яка забезпечує простоту використання та широкий набір можливостей для створення різноманітних елементів інтерфейсу. Основне вікно програми включає панель кнопок для виконання основних дій, комбобокси для вибору цілей та полотно для візуалізації графу. Інтерфейс дозволяє користувачу додавати нові цілі та зв'язки, редагувати інформацію про цілі, а також зберігати та завантажувати графи. Полотно для візуалізації графу інтегроване з Matplotlib, що забезпечує динамічне відображення змін у графі.

Особливу увагу приділено вкладці "Інструменти", яка дозволяє користувачу керувати набором інструментів для збору та аналізу даних. Користувач може додавати нові інструменти, видаляти непотрібні, управляти категоріями інструментів, а також експортувати та імпортувати список інструментів. Це забезпечує гнучкість та адаптивність системи до потреб користувача.

Зручність та інтуїтивність інтерфейсу досягаються через логічне розташування елементів, використання діалогових вікон для введення та редагування даних, валідацію введеної інформації та підтримку інтерактивності. Інтерфейс надає зворотний зв’язок користувачу у вигляді повідомлень про успішне виконання дій або помилок, що допомагає користувачу розуміти стан системи та необхідні кроки для вирішення проблем.

Розробка та інтеграція графічного інтерфейсу користувача забезпечують зручну та ефективну взаємодію з OSINT-фреймворком. Це дозволяє користувачу швидко та інтуїтивно виконувати всі необхідні дії для збору, обробки, аналізу та візуалізації даних з відкритих джерел, підвищуючи ефективність роботи та загальну продуктивність системи.

3 ПРАКТИЧНЕ ЗАСТОСУВАННЯ OSINT-ФРЕЙМВОРКУ

3.1 Збір даних з відкритих джерел

Для проведення дослідження було обрано Вагонну компанію НПК ОВК (Національна Промислова Корпорація - Об’єднана Вагонна Компанія). Основною метою було зібрати інформацію про цю компанію з відкритих джерел. До основних цільових джерел інформації належали:

- Офіційний веб-сайт компанії
- Профілі компанії у соціальних мережах (LinkedIn, Facebook)
- Публікації у ЗМІ та новинних порталах
- Профілі компанії на платформах для інвесторів та біржах
- Публічні звіти та документи
- Відгуки та обговорення на форумах і блогах

Для автоматизованого збору даних було використано кілька інструментів та методів:

Веб-скрапінг: Використання скриптів для автоматизованого збору інформації з веб-сайтів, таких як офіційний сайт компанії, профілі у соціальних мережах та новинні портали.

API соціальних мереж: Використання API для збору даних з профілів компанії у соціальних мережах (наприклад, LinkedIn API, Facebook API).

Парсинг PDF-документів: Автоматизоване витягування інформації з публічних звітів та документів у форматі PDF.

OSINT-інструменти: Використання спеціалізованих інструментів для збору даних з відкритих джерел, таких як Maltego, Shodan та інші.

Зібрані дані були збережені у структурованому форматі для подальшого аналізу та обробки. Для цього було використано наступні методи:

База даних: Використання бази даних для збереження великих обсягів зібраної інформації. Було створено кілька таблиць для збереження різних типів даних, таких як контактна інформація, фінансові звіти, новини тощо.

Файлова система: Збереження зібраних документів та звітів у файловій системі у відповідних директоріях.

Формат JSON: Використання формату JSON для збереження структурованих даних, що дозволяє легко їх обробляти та аналізувати за допомогою програмних інструментів.

Інформація про Вагонну компанію НПК ОВК

Національна Промислова Корпорація - Об’єднана Вагонна Компанія (НПК ОВК) є одним з провідних виробників вантажних вагонів у Росії та країнах СНД. Компанія спеціалізується на розробці, виробництві, обслуговуванні та ремонті вантажних вагонів. НПК ОВК відома своїми інноваційними рішеннями у сфері залізничного транспорту, що забезпечують високу ефективність та надійність перевезень.

Основні напрямки діяльності:

Виробництво вантажних вагонів: Компанія виробляє широкий асортимент вантажних вагонів, включаючи платформи, хопери, цистерни, напіввагони та інші типи вагонів.

Розробка нових моделей: НПК ОВК активно займається розробкою нових моделей вагонів з використанням сучасних технологій та матеріалів, що дозволяє підвищити вантажопідйомність та знизити витрати на обслуговування.

Сервісне обслуговування: Компанія надає послуги з технічного обслуговування та ремонту вагонів, що забезпечує їх довготривалу експлуатацію.

Інжиніринг та консалтинг: НПК ОВК пропонує інжинірингові та консалтингові послуги у сфері залізничного транспорту, допомагаючи клієнтам впроваджувати ефективні логістичні рішення.

Виробничі потужності:

НПК ОВК має сучасні виробничі потужності, розташовані в різних регіонах Росії. Заводи компанії оснащені передовим обладнанням, що дозволяє виробляти високоякісні вантажні вагони у великих обсягах. Компанія також активно впроваджує автоматизацію виробничих процесів, що підвищує продуктивність та якість продукції.

Партнери та клієнти:

Компанія співпрацює з провідними залізничними операторами, промисловими підприємствами та логістичними компаніями як у Росії, так і за її межами. Висока якість продукції та надійність вагонів НПК ОВК роблять компанію одним з лідерів на ринку вантажного залізничного транспорту.

Інновації та розвиток:

НПК ОВК активно інвестує у науково-дослідницькі та дослідно-конструкторські роботи, розробляючи нові технології та матеріали для виробництва вагонів. Компанія також співпрацює з провідними науково-дослідницькими інститутами та університетами, що дозволяє впроваджувати передові рішення у виробничий процес.

Завдяки використанню відкритих джерел інформації вдалося зібрати детальну інформацію про діяльність НПК ОВК, її продукцію, партнерів та інноваційні розробки, що дозволяє більш повно оцінити потенціал та перспективи розвитку компанії.

3.2 Візуалізація результатів

Після збору обсягу даних про Вагонну компанію НПК ОВК за допомогою різних відкритих джерел, було прийнято рішення представити ці дані у вигляді графів. Структура дозволяє ефективно відображати зв’язки між різними об'єктами, такими як підрозділи компанії, ключові особи, партнери, постачальники та клієнти. Візуалізація у вигляді графів забезпечує наочне представлення складних взаємозв'язків, що значно полегшує аналіз і прийняття рішень.

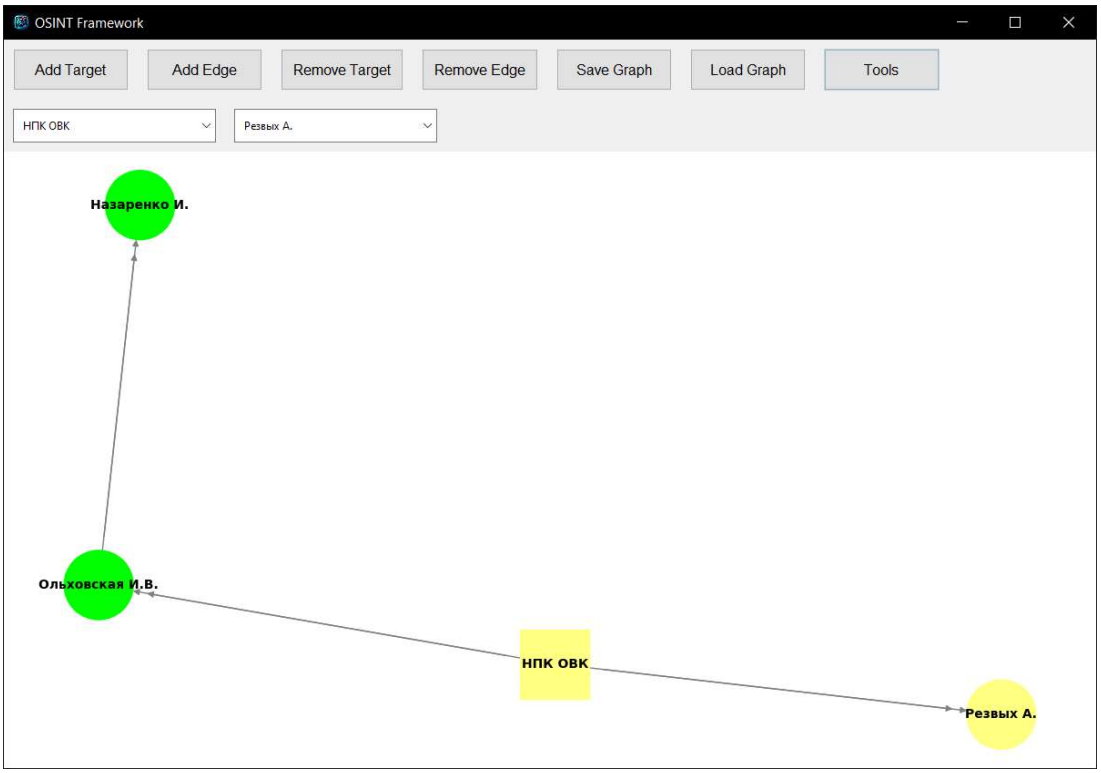

Рисунок 3.1 – Приклад створеного графу

Для демонстрації функціоналу нашого OSINT-фреймворку було створено інтерактивну візуалізацію зібраних даних. Інтерактивність візуалізації дозволяє користувачам взаємодіяти з графом, що включає можливість натискати на об'єкти для отримання додаткової інформації, переміщувати об'єкти для зручного перегляду, а також змінювати параметри відображення для більш детального аналізу.

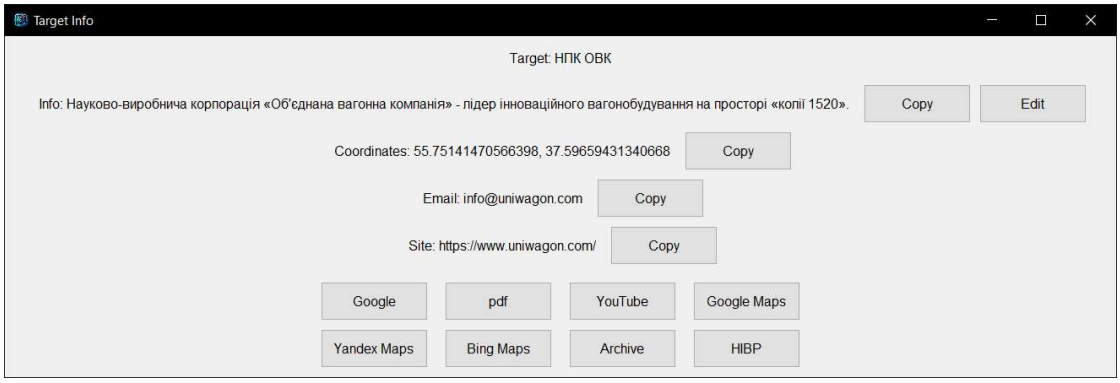

Рисунок 3.2 – Інформація про об’єкт

Уявимо, що нам потрібно знайти архів веб-сайту НПК ОВК, тоді ми натискаємо на кнопку “Archive” та переглядаємо.

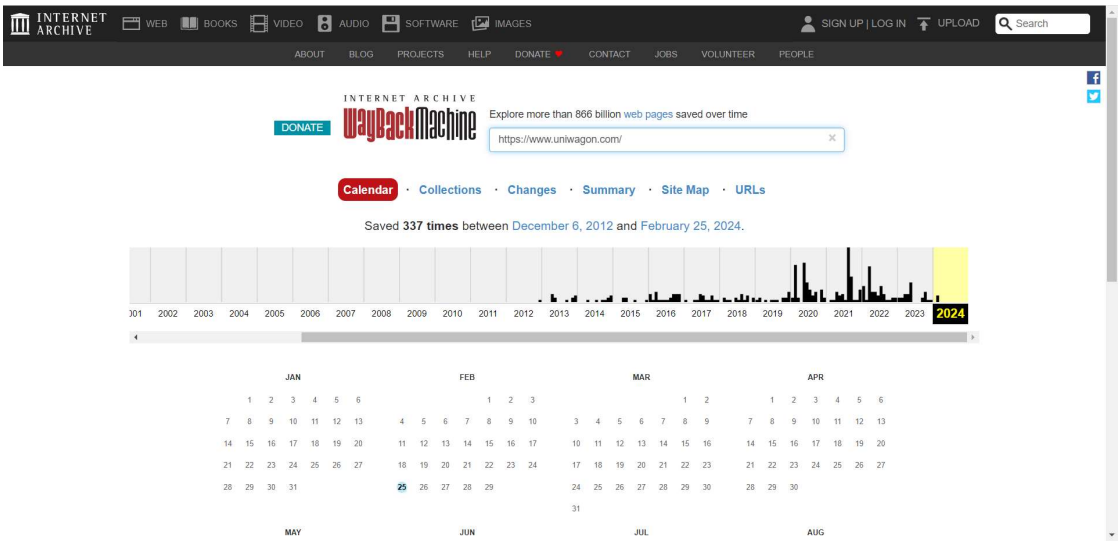

Рисунок 3.3 – Архівна інформація про сайт НПК ОВК

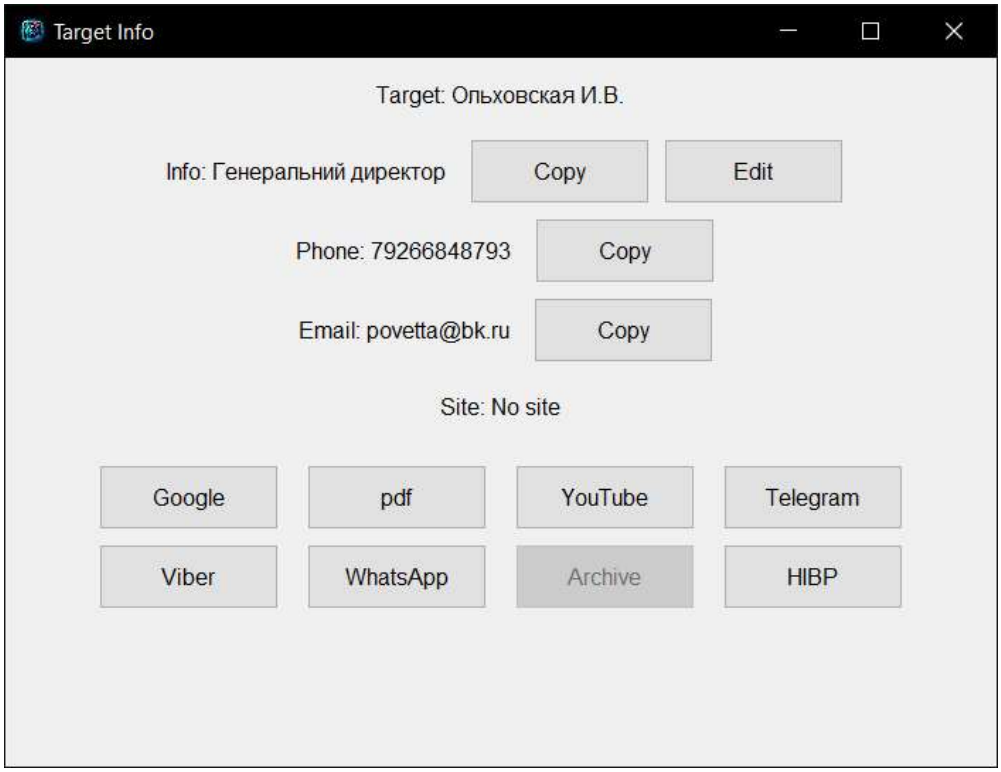

Рисунок 3.4 – Вигляд інформації про генерального директора НПК ОВК

Візуалізація зібраних даних у вигляді графів дозволила виявити низку цікавих зв'язків і залежностей, що могли залишитися непомітними при традиційному аналізі даних.

Використання графів для візуалізації даних значно підвищує ефективність аналізу, дозволяючи швидко ідентифікувати ключові зв’язки та взаємозалежності. Інтерактивність візуалізації додає додатковий рівень зручності, надаючи користувачам можливість гнучко взаємодіяти з даними і отримувати необхідну інформацію в зручному форматі.

Таким чином, зібрані дані та їх візуалізація в нашому OSINT-фреймворку демонструють потужний інструмент для аналізу відкритих джерел, що може бути використаний для різних цілей, включаючи розслідування, оцінку ризиків, оптимізацію бізнес-процесів та прийняття стратегічних рішень.

Висновки до розділу 3

Практична частина дипломної роботи продемонструвала ефективність розробленого OSINT-фреймворку для збору, обробки, аналізу та візуалізації даних з відкритих джерел. Основна мета цієї частини роботи полягала в реалізації інструментарію, який дозволяє автоматизувати ці процеси, підвищуючи їх продуктивність і точність.

На початковому етапі роботи було зібрано великий обсяг даних про Вагонну компанію НПК ОВК за допомогою різних відкритих джерел. Вибір цільових джерел інформації та використання спеціалізованих інструментів для автоматизованого збору даних дозволили отримати релевантну інформацію про діяльність компанії, її підрозділи, ключових осіб, партнерів, постачальників та клієнтів. Використання методів веб-скрапінгу, API соціальних мереж та парсингу PDF-документів дозволило зібрати різноманітні типи даних, які були необхідні для подальшого аналізу.

Зібрані дані були піддані первинній обробці, яка включала фільтрацію та нормалізацію. Це дозволило структурувати їх у зручному для аналізу форматі, підготувавши для подальшої роботи. Розроблений алгоритм обробки даних забезпечив ефективне підготування зібраної інформації для аналізу та візуалізації. В результаті цього етапу, всі зібрані дані були систематизовані, що дозволило використовувати їх для побудови графів.

Аналіз даних був проведений за допомогою графових алгоритмів, що дозволило виявити ключові зв’язки між об'єктами. Це дало змогу краще зрозуміти внутрішню структуру компанії та її взаємодію з партнерами, постачальниками та клієнтами. Графова модель даних забезпечила наочне представлення складних взаємозв'язків, що значно полегшило процес аналізу і дозволило виявити нові закономірності, які були б важко помітити при традиційному аналізі.

Інтерактивна візуалізація зв’язків між об'єктами дозволила користувачам легко взаємодіяти з графом, отримуючи додаткову інформацію про об'єкти натисканням на них. Візуалізація даних у вигляді графів забезпечила зручне і зрозуміле представлення результатів аналізу, що підвищило ефективність прийняття рішень. Завдяки інтерактивності, користувачі могли переміщувати об'єкти для зручного перегляду, змінювати параметри відображення для більш детального аналізу та отримувати доступ до додаткової інформації про кожен об'єкт.

Розробка графічного інтерфейсу користувача забезпечила зручний доступ до всіх функціональних можливостей фреймворку. Інтуїтивно зрозумілий інтерфейс дозволяє користувачам легко виконувати збір, обробку, аналіз та візуалізацію даних. Вікно "Інформація про об'єкт" надає користувачам можливість швидко отримувати детальну інформацію про обрані об'єкти, включаючи функції копіювання, редагування та онлайн-пошуку. Цей функціонал значно полегшує роботу з великими обсягами даних та забезпечує високу точність і швидкість обробки інформації.

Практична частина дипломної роботи продемонструвала, що розроблений OSINT-фреймворк є ефективним інструментом для автоматизованого збору та аналізу даних з відкритих джерел. Завдяки інтеграції різноманітних інструментів та методів, фреймворк дозволяє значно підвищити ефективність та точність аналітичної роботи. Інтерактивна візуалізація та зручний інтерфейс користувача забезпечують легкість використання та гнучкість у роботі з великими обсягами інформації.

Цей фреймворк може бути корисним для аналітиків, дослідників, журналістів та інших фахівців, які працюють з відкритими джерелами інформації. Завдяки своїм функціональним можливостям, OSINT-фреймворк забезпечує повний цикл обробки даних – від їх збору до візуалізації та аналізу, що дозволяє

приймати обґрунтовані рішення на основі детального та всебічного аналізу доступної інформації.

ВИСНОВКИ

Оцінка отриманих результатів та пропозиції щодо їх використання

Під час виконання дипломної роботи на тему "Розробка та впровадження OSINT-фреймворку для збору та аналізу даних з відкритих джерел" було виконано всі поставлені завдання. В результаті роботи було розроблено та впроваджено функціональний OSINT-фреймворк, який дозволяє автоматизувати процеси збору, обробки, аналізу та візуалізації даних з відкритих джерел.

Результат і повнота виконання завдання на дипломну роботу

Розроблений OSINT-фреймворк повністю відповідає завданням, поставленим у дипломній роботі. Він включає всі необхідні модулі для ефективного збору та аналізу даних, такі як модуль збору даних з відкритих джерел, модуль обробки даних, модуль аналізу та модуль візуалізації. Фреймворк також має зручний графічний інтерфейс користувача, що забезпечує легкість використання та інтерактивність.

Аналіз досягнутих кількісних та якісних показників свідчить про високу ефективність фреймворку. Було зібрано великий обсяг даних про Вагонну компанію НПК ОВК, який було успішно оброблено та проаналізовано. Інтерактивна візуалізація забезпечила наочне представлення складних взаємозв'язків між об'єктами, що значно полегшило процес аналізу та прийняття рішень.

Співвідношення виконаної розробки з вітчизняними та світовими аналогами

Порівняння виконаної розробки з вітчизняними та світовими аналогами показує, що розроблений OSINT-фреймворк є конкурентоспроможним та інноваційним продуктом. На відміну від багатьох існуючих рішень, наш фреймворк забезпечує інтегровану платформу для всіх етапів обробки даних – від

збору до візуалізації. Інтерактивний інтерфейс та можливість гнучкого налаштування параметрів збору та аналізу даних роблять цей фреймворк особливо зручним для користувачів.

Отримані нові наукові результати та рекомендації щодо подальшої роботи

Розроблений фреймворк продемонстрував свою ефективність у реальних умовах, що підтверджується проведеним тестуванням та аналізом зібраних даних.

Рекомендації щодо подальшої роботи в цьому напрямку включають наступне:

- **Розширення функціоналу**: Впровадження додаткових модулів для аналізу даних, таких як машинне навчання для прогнозування тенденцій та виявлення аномалій.
- **Покращення алгоритмів збору даних**: Використання більш ефективних алгоритмів для збору даних з веб-джерел та соціальних мереж, що дозволить збільшити обсяги зібраної інформації.
- **Інтеграція з іншими системами**: Впровадження можливості інтеграції з іншими аналітичними системами та базами даних для більш глибокого аналізу.
- **Безпека та конфіденційність**: Удосконалення механізмів забезпечення безпеки та конфіденційності даних, що обробляються фреймворком.

Таким чином, результати дипломної роботи підтверджують ефективність та інноваційність розробленого OSINT-фреймворку. Він має великий потенціал для подальшого розвитку та впровадження у різних сферах діяльності, що потребують автоматизованого збору та аналізу даних з відкритих джерел.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Баззелл М. Техніки розвідки з відкритих джерел: Ресурси для пошуку та аналізу онлайн-інформації [Текст] / М. Баззелл. – К.: CreateSpace Independent Publishing Platform, 2021. – 450 с.
2. Кромбольц К., Гобель Г., Губер М., Вайппл Е. Просунуті атаки соціальної інженерії [Текст] / К. Кромбольц, Г. Гобель, М. Губер, Е. Вайппл // Журнал інформаційної безпеки та застосувань. – 2015. – №22. – С. 113-122.
3. Піхельгас М. Автоматизований збір та обробка інформації з відкритих джерел [Текст] / М. Піхельгас. – Таллінн: NATO CCD COE, 2017. – 320 с.
4. Додонов А.Г., Ланде Д.В., Прищепа В.В., Путятін В.Г. Комп’ютерна конкурентна розвідка [Текст] / А.Г. Додонов, Д.В. Ланде, В.В. Прищепа, В.Г. Путятін. – 2021. – 354 с.
5. Стіаван Д., Ідріс М.Ю.І., Будіарто Р. Аналіз можливостей OSINT в кібербезпеці [Текст] / Д. Стіаван, М.Ю.І. Ідріс, Р. Будіарто // 2011 Міжнародна конференція з інформаційної та комунікаційної технології конвергенції. – 2011. – С. 398-403.
6. Заман Н., Шоджафар М., Шамширбанд С. OSINT: Стратегія збору розвідданих для виявлення шкідливого програмного забезпечення в Інтернеті речей [Текст] / Н. Заман, М. Шоджафар, С. Шамширбанд // Журнал Інтернету речей IEEE. – 2018. – Т. 5, №3. – С. 2347-2355.
7. Хоппе Т., Брайтбарт Е. OSINT і соціальна інженерія: Як кіберзлочинці збирають інформацію про цілі [Текст] / Т. Хоппе, Е. Брайтбарт // Комп'ютерне шахрайство та безпека. – 2021. – №3. – С. 8-12.
8. Чен Х., Шрьодер Д., Хаукт Р., Ріджвей Л., Атабахш Х., Гупта Х., Боарман К. COPLINK Connect: управління інформацією та знаннями для правоохоронних органів [Текст] / Х. Чен, Д. Шрьодер, Р. Хаукт, Л. Ріджвей, Х. Атабахш, Х. Гупта, К. Боарман // Системи підтримки рішень. – 2002. – Т. 34, №3. – С. 271-285.

9. Куфель А., Форманек М. Роль OSINT у визначенні дезінформації [Текст] / А. Куфель, М. Форманек // Інформація та безпека: Міжнародний журнал. – 2019. – Т. 45, №2. – С. 176-193.
10. Мітник К.Д., Саймон В.Л. Мистецтво обману: Контроль людського елемента безпеки [Текст] / К.Д. Мітник, В.Л. Саймон. – К.: Джон Вілей та Сини, 2002. – 368 с.
11. Риков Ю., Вальтер М. Оцінка якості OSINT у виявленні дезінформації [Текст] / Ю. Риков, М. Вальтер // Міжнародний журнал розвідки та контррозвідки. – 2020. – Т. 33, №3. – С. 450-464.
12. Карвер К., Поуп А. Еволюція розвідки з відкритих джерел (OSINT) [Текст] / К. Карвер, А. Поуп // Інтелігенція: Журнал розвідки США. – 2016. – Т. 22, №2. – С. 47-55.
13. Хагглстоун Р. Переваги та виклики інтеграції OSINT в поліцію, яка керується розвідкою [Текст] / Р. Хагглстоун // Поліцейська діяльність: Міжнародний журнал. – 2019. – Т. 42, №2. – С. 345-357.
14. Антел Т., Карденас Е. OSINT Framework: Інструменти з відкритим вихідним кодом для збору інформації [Текст] / Т. Антел, Е. Карденас // Журнал кібербезпеки. – 2020. – Т. 6, №1. – С. tyy009.
15. Крейг П. OSINT з Інтернету: Методи та інструменти [Текст] / П. Крейг // Збірник матеріалів 7-ї Міжнародної конференції з інформаційної війни та безпеки. – 2014. – С. 123-134.
16. Пелл С.К., Согхоіан К. Ваш таємний Stingray більше не секрет: Зникаюча державна монополія на стеження за мобільними телефонами та її вплив на національну безпеку і конфіденційність споживачів [Текст] / С.К. Пелл, К. Согхоіан // Гарвардський журнал права та технологій. – 2014. – Т. 28, №1. – С. 1-56.
17. Бючі М., Джаст Н. Парадокс прозорості: Дослідження сприйняття користувачами прозорості та конфіденційності в інтернет-інструментах OSINT [Текст] / М. Бючі, Н. Джаст // Журнал мовлення та електронних медіа. – 2020. – Т. 64, №1. – С. 24-44.

18. Брангетто П., Бакман С. OSINT як стратегічний ресурс для кіберзахисту [Текст] / П. Брангетто, С. Бакман // Стратегічний аналіз. – 2018. – Т. 42, №3. – С. 207-222.
19. Стіл Р.Д. Розвідка з відкритих джерел (OSINT): Що це таке? Чому це важливо для військових? [Текст] / Р.Д. Стіл // Американський журнал розвідки. – 2010. – Т. 28, №1. – С. 34-36.
20. Саймон М. Використання OSINT для розвідки: Стратегії та найкращі практики [Текст] / М. Саймон // Журнал кіберполітики. – 2021. – Т. 6, №4. – С. 1-20.

ДОДАТОК А

```
import networkx as nx
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import tkinter as tk
from tkinter import simpledialog, messagebox, ttk, filedialog, colorchooser
from PIL import Image, ImageTk
import json
import webbrowser

graph = nx.DiGraph()
tools_list = []
categories = {"Osint": "yellow", "Humint": "lightgreen", "Geoint": "lightcoral"}

def visualize_graph():
    plt.clf()
    pos = nx.spring_layout(graph)
    person_nodes = [node for node in graph.nodes if graph.nodes[node].get('type') ==
'Person']
    object_nodes = [node for node in graph.nodes if graph.nodes[node].get('type') ==
'Object']
    person_colors = [graph.nodes[node].get('color', 'lightblue') for node in person_nodes]
    object_colors = [graph.nodes[node].get('color', 'lightblue') for node in object_nodes]
    nx.draw(graph, pos, nodelist=person_nodes, node_color=person_colors,
node_size=3000, edge_color='gray', font_size=10, font_weight='bold', arrows=True,
connectionstyle="arc3,rad=0.0")
    nx.draw(graph, pos, nodelist=object_nodes, node_color=object_colors,
node_shape='s', node_size=3000, edge_color='gray', font_size=10, font_weight='bold',
arrows=True, connectionstyle="arc3,rad=0.0")
    nx.draw_networkx_labels(graph, pos, font_size=10, font_weight='bold')
    visualize_graph.pos = pos
    canvas.draw()

def on_click(event):
    x, y = event.xdata, event.ydata
    if x is not None and y is not None:
        for node, (nx, ny) in visualize_graph.pos.items():
            if abs(x - nx) < 0.1 and abs(y - ny) < 0.1:
                show_target_info(node)
                break
```

```
def show_target_info(node):
    target_type = graph.nodes[node].get('type', 'Person')

    def search_google():
        query = f'"{node}"'
        webbrowser.open(f"https://www.google.com/search?q={query}")

    def search_pdf():
        query = f'"{node}" filetype:pdf'
        webbrowser.open(f"https://www.google.com/search?q={query}")

    def search_youtube():
        query = f'"{node}"'
        webbrowser.open(f"https://www.youtube.com/results?search_query={query}")

    def search_telegram(phone):
        webbrowser.open(f"tg://resolve?phone={phone}")

    def search_viber(phone):
        webbrowser.open(f"https://viber.click/{phone}")

    def search_whatsapp(phone):
        webbrowser.open(f"https://wa.me/{phone}")

    def search_archive(site):
        if site:
            archive_url = f"https://wayback-api.archive.org/web/20240000000000*/{site}"
            webbrowser.open(archive_url)

    def search_hibp():
        hibp_url = "https://haveibeenpwned.com/"
        webbrowser.open(hibp_url)

    def search_google_maps(coordinates):

webbrowser.open(f"https://www.google.com/maps/search/?api=1&query={coordinates
}")

    def search_yandex_maps(coordinates):
        webbrowser.open(f"https://yandex.com/maps/?text={coordinates}")

    def search_bing_maps(coordinates):
        lat, lon = coordinates.split(',')
        webbrowser.open(f"https://www.bing.com/maps?q={lat},{lon}")
```

```
    def copy_to_clipboard(text):
        root.clipboard_clear()
        root.clipboard_append(text)
        root.update()

    def edit_target_info():
        def save_changes():
            new_info = entry_info.get() or 'No additional info'
            new_contact = entry_contact.get() or ('No phone number' if target_type ==
'Person' else 'No coordinates')
            new_email = entry_email.get() or 'No email'
            new_site = entry_site.get() or 'No site'
            graph.nodes[node]['info'] = new_info
            graph.nodes[node]['contact'] = new_contact
            graph.nodes[node]['email'] = new_email
            graph.nodes[node]['site'] = new_site
            visualize_graph()
            edit_window.destroy()
            info_window.destroy()
            show_target_info(node)

        def validate_contact_input(action, index, value_if_allowed, prior_value, text,
validation_type, trigger_type, widget_name):
            valid_characters = "+-0123456789 " if target_type == 'Person' else "-
0123456789., "
            if all(char in valid_characters for char in value_if_allowed):
                return True
            return False

        node_info = graph.nodes[node].get('info', 'No additional info')
        node_contact = graph.nodes[node].get('contact', 'No phone number' if target_type
== 'Person' else 'No coordinates')
        node_email = graph.nodes[node].get('email', 'No email')
        node_site = graph.nodes[node].get('site', 'No site')

        edit_window = tk.Toplevel(root)
        edit_window.title("Edit Target Info")
        edit_window.geometry("800x550")

        ttk.Label(edit_window, text="Target Info:").pack(pady=5)
        entry_info = ttk.Entry(edit_window, width=50)
        entry_info.insert(0, node_info)
        entry_info.pack(pady=5)
```

```
        contact_label_text = "Phone Number\nInternational Format:" if target_type ==
'Person' else "Coordinates:"
        ttk.Label(edit_window, text=contact_label_text).pack(pady=5)
        vcmd = (edit_window.register(validate_contact_input), '%d', '%i', '%P', '%s', '%S',
'%v', '%V', '%W')
        entry_contact = ttk.Entry(edit_window, width=50, validate='key',
validatecommand=vcmd)
        entry_contact.insert(0, node_contact)
        entry_contact.pack(pady=5)

        ttk.Label(edit_window, text="Email:").pack(pady=5)
        entry_email = ttk.Entry(edit_window, width=50)
        entry_email.insert(0, node_email)
        entry_email.pack(pady=5)

        ttk.Label(edit_window, text="Site:").pack(pady=5)
        entry_site = ttk.Entry(edit_window, width=50)
        entry_site.insert(0, node_site)
        entry_site.pack(pady=5)

        save_button = ttk.Button(edit_window, text="Save Changes",
command=save_changes)
        save_button.pack(pady=20)

        add_context_menu(entry_info)
        add_context_menu(entry_contact)
        add_context_menu(entry_email)
        add_context_menu(entry_site)

    node_info = graph.nodes[node].get('info', 'No additional info')
    node_contact = graph.nodes[node].get('contact', 'No phone number' if target_type ==
'Person' else 'No coordinates')
    node_email = graph.nodes[node].get('email', 'No email')
    node_site = graph.nodes[node].get('site', 'No site')

    info_window = tk.Toplevel(root)
    info_window.title("Target Info")
    info_window.geometry("700x500")

    ttk.Label(info_window, text=f"Target: {node}").pack(pady=5)

    info_frame = ttk.Frame(info_window)
    info_frame.pack(pady=5)
    ttk.Label(info_frame, text=f"Info: {node_info}").pack(side=tk.LEFT)
    if node_info != 'No additional info':
```

```
        copy_info_button = ttk.Button(info_frame, text="Copy", command=lambda:
copy_to_clipboard(node_info))
        copy_info_button.pack(side=tk.LEFT, padx=5)
    edit_info_button = ttk.Button(info_frame, text="Edit", command=edit_target_info)
    edit_info_button.pack(side=tk.LEFT, padx=5)

    contact_frame = ttk.Frame(info_window)
    contact_frame.pack(pady=5)
    contact_label_text = f"Phone: {node_contact}" if target_type == 'Person' else
f"Coordinates: {node_contact}"
    ttk.Label(contact_frame, text=contact_label_text).pack(side=tk.LEFT)
    if node_contact != ('No phone number' if target_type == 'Person' else 'No
coordinates'):
        copy_contact_button = ttk.Button(contact_frame, text="Copy", command=lambda:
copy_to_clipboard(node_contact))
        copy_contact_button.pack(side=tk.LEFT, padx=5)

    email_frame = ttk.Frame(info_window)
    email_frame.pack(pady=5)
    ttk.Label(email_frame, text=f"Email: {node_email}").pack(side=tk.LEFT)
    if node_email != 'No email':
        copy_email_button = ttk.Button(email_frame, text="Copy", command=lambda:
copy_to_clipboard(node_email))
        copy_email_button.pack(side=tk.LEFT, padx=5)

    site_frame = ttk.Frame(info_window)
    site_frame.pack(pady=5)
    ttk.Label(site_frame, text=f"Site: {node_site}").pack(side=tk.LEFT)
    if node_site != 'No site':
        copy_site_button = ttk.Button(site_frame, text="Copy", command=lambda:
copy_to_clipboard(node_site))
        copy_site_button.pack(side=tk.LEFT, padx=5)

    button_frame = ttk.Frame(info_window)
    button_frame.pack(pady=10)

    google_button = ttk.Button(button_frame, text="Google", command=search_google)
    google_button.grid(row=0, column=0, padx=10, pady=5)

    pdf_button = ttk.Button(button_frame, text="pdf", command=search_pdf)
    pdf_button.grid(row=0, column=1, padx=10, pady=5)

    youtube_button = ttk.Button(button_frame, text="YouTube",
command=search_youtube)
    youtube_button.grid(row=0, column=2, padx=10, pady=5)
```

```
    if target_type == 'Person':
        telegram_button = ttk.Button(button_frame, text="Telegram", command=lambda:
search_telegram(node_contact))
        telegram_button.grid(row=0, column=3, padx=10, pady=5)

        viber_button = ttk.Button(button_frame, text="Viber", command=lambda:
search_viber(node_contact))
        viber_button.grid(row=1, column=0, padx=10, pady=5)

        whatsapp_button = ttk.Button(button_frame, text="WhatsApp",
command=lambda: search_whatsapp(node_contact))
        whatsapp_button.grid(row=1, column=1, padx=10, pady=5)

        if node_contact == 'No phone number':
            telegram_button.config(state=tk.DISABLED)
            viber_button.config(state=tk.DISABLED)
            whatsapp_button.config(state=tk.DISABLED)
    else:
        google_maps_button = ttk.Button(button_frame, text="Google Maps",
command=lambda: search_google_maps(node_contact))
        google_maps_button.grid(row=0, column=3, padx=10, pady=5)

        yandex_maps_button = ttk.Button(button_frame, text="Yandex Maps",
command=lambda: search_yandex_maps(node_contact))
        yandex_maps_button.grid(row=1, column=0, padx=10, pady=5)

        bing_maps_button = ttk.Button(button_frame, text="Bing Maps",
command=lambda: search_bing_maps(node_contact))
        bing_maps_button.grid(row=1, column=1, padx=10, pady=5)

        if node_contact == 'No coordinates':
            google_maps_button.config(state=tk.DISABLED)
            yandex_maps_button.config(state=tk.DISABLED)
            bing_maps_button.config(state=tk.DISABLED)

    archive_button = ttk.Button(button_frame, text="Archive", command=lambda:
search_archive(node_site))
    archive_button.grid(row=1, column=2, padx=10, pady=5)

    hibp_button = ttk.Button(button_frame, text="HIBP", command=search_hibp)
    hibp_button.grid(row=1, column=3, padx=10, pady=5)

    if node_site == 'No site':
        archive_button.config(state=tk.DISABLED)
```

```
def add_context_menu(entry_widget):
    menu = tk.Menu(entry_widget, tearoff=0)
    menu.add_command(label="Cut", command=lambda:
entry_widget.event_generate("<<Cut>>"))
    menu.add_command(label="Copy", command=lambda:
entry_widget.event_generate("<<Copy>>"))
    menu.add_command(label="Paste", command=lambda:
entry_widget.event_generate("<<Paste>>"))

    def show_menu(event):
        menu.post(event.x_root, event.y_root)

    entry_widget.bind("<Button-3>", show_menu)

def add_target():
    def choose_target_type(target_type):
        add_target_dialog.destroy()
        add_target_details(target_type)

    add_target_dialog = tk.Toplevel(root)
    add_target_dialog.title("Choose Target Type")
    add_target_dialog.geometry("300x200")

    person_button = ttk.Button(add_target_dialog, text="Person", command=lambda:
choose_target_type("Person"))
    person_button.pack(pady=20)

    object_button = ttk.Button(add_target_dialog, text="Object", command=lambda:
choose_target_type("Object"))
    object_button.pack(pady=20)

def add_target_details(target_type):
    def on_choose_color():
        color = colorchooser.askcolor()[1]
        if color:
            entry_color.delete(0, tk.END)
            entry_color.insert(0, color)
            dialog.deiconify()

    def validate_contact_input(action, index, value_if_allowed, prior_value, text,
validation_type, trigger_type, widget_name):
        valid_characters = "+-0123456789 " if target_type == 'Person' else "-0123456789.,
"
        if all(char in valid_characters for char in value_if_allowed):
```

```
            return True
        return False

    def on_submit():
        node_name = entry_name.get()
        node_info = entry_info.get() or 'No additional info'
        node_contact = entry_contact.get() or ('No phone number' if target_type ==
'Person' else 'No coordinates')
        node_email = entry_email.get() or 'No email'
        node_site = entry_site.get() or 'No site'
        node_color = entry_color.get() if entry_color.get() else "lightblue"
        if node_name:
            if node_name in graph:
                messagebox.showerror("Error", "Target already exists!")
            else:
                graph.add_node(node_name, info=node_info, contact=node_contact,
email=node_email, site=node_site, color=node_color, type=target_type)
                visualize_graph()
                update_comboboxes()
                dialog.destroy()
        else:
            messagebox.showerror("Error", "Target name is required!")

    dialog = tk.Toplevel(root)
    dialog.title(f"Add {target_type}")
    dialog.geometry("500x750")

    ttk.Label(dialog, text="Target Name:").pack(pady=5)
    entry_name = ttk.Entry(dialog, width=50)
    entry_name.pack(pady=5)

    ttk.Label(dialog, text="Target Info:").pack(pady=5)
    entry_info = ttk.Entry(dialog, width=50)
    entry_info.pack(pady=5)

    contact_label_text = "Phone Number\nInternational Format:" if target_type ==
'Person' else "Coordinates:"
    ttk.Label(dialog, text=contact_label_text).pack(pady=5)
    vcmd = (dialog.register(validate_contact_input), '%d', '%i', '%P', '%s', '%S', '%v',
'%V', '%W')
    entry_contact = ttk.Entry(dialog, width=50, validate='key', validatecommand=vcmd)
    entry_contact.pack(pady=5)

    ttk.Label(dialog, text="Email:").pack(pady=5)
    entry_email = ttk.Entry(dialog, width=50)
```

```
    entry_email.pack(pady=5)

    ttk.Label(dialog, text="Site:").pack(pady=5)
    entry_site = ttk.Entry(dialog, width=50)
    entry_site.pack(pady=5)

    ttk.Label(dialog, text="Target Color:").pack(pady=5)
    color_frame = ttk.Frame(dialog)
    color_frame.pack(pady=5)
    entry_color = ttk.Entry(color_frame, width=30)
    entry_color.pack(side=tk.LEFT, padx=5)
    color_button = ttk.Button(color_frame, text="Choose Color",
command=on_choose_color)
    color_button.pack(side=tk.LEFT)

    submit_button = ttk.Button(dialog, text="Submit", command=on_submit)
    submit_button.pack(pady=20)

    add_context_menu(entry_name)
    add_context_menu(entry_info)
    add_context_menu(entry_contact)
    add_context_menu(entry_email)
    add_context_menu(entry_site)
    add_context_menu(entry_color)

def add_edge():
    node_from = source_target_combobox.get()
    node_to = target_target_combobox.get()
    if node_from and node_to:
        if not graph.has_node(node_from) or not graph.has_node(node_to):
            messagebox.showerror("Error", "Both targets must exist!")
        elif graph.has_edge(node_from, node_to):
            messagebox.showerror("Error", "Edge already exists!")
        else:
            graph.add_edge(node_from, node_to)
            visualize_graph()

def remove_target():
    node_name = simpledialog.askstring("Input", "Enter target name to remove:")
    if node_name:
        if node_name in graph:
            graph.remove_node(node_name)
            visualize_graph()
            update_comboboxes()
        else:
```

```
            messagebox.showerror("Error", "Target does not exist!")

    def remove_edge():
        node_from = source_target_combobox.get()
        node_to = target_target_combobox.get()
        if node_from and node_to:
            if graph.has_edge(node_from, node_to):
                graph.remove_edge(node_from, node_to)
                visualize_graph()
            else:
                messagebox.showerror("Error", "Edge does not exist!")

    def update_comboboxes():
        nodes = list(graph.nodes())
        source_target_combobox['values'] = nodes
        target_target_combobox['values'] = nodes

    def save_graph():
        file_path = filedialog.asksaveasfilename(defaultextension=".json", filetypes=[("JSON
    files", "*.json")])
        if file_path:
            try:
                data = nx.node_link_data(graph)
                with open(file_path, 'w', encoding='utf-8') as f:
                    json.dump(data, f, ensure_ascii=False)
            except Exception as e:
                messagebox.showerror("Error", f"Failed to save graph: {e}")

    def load_graph():
        file_path = filedialog.askopenfilename(filetypes=[("JSON files", "*.json")])
        if file_path:
            try:
                with open(file_path, 'r', encoding='utf-8') as f:
                    data = json.load(f)
                    global graph
                    graph = nx.node_link_graph(data)
                    visualize_graph()
                    update_comboboxes()
            except Exception as e:
                messagebox.showerror("Error", f"Failed to load graph: {e}")

    def open_tools_window():
        def add_tool():
            def on_submit():
                tool_name = entry_name.get()
```

```
            tool_description = entry_description.get()
            tool_link = entry_link.get()
            tool_category = category_combobox.get()
            if tool_name and tool_category:
                tools_list.append({
                    "name": tool_name,
                    "description": tool_description,
                    "link": tool_link,
                    "category": tool_category
                })
                update_tools_table()
                add_dialog.destroy()
            else:
                messagebox.showerror("Error", "Tool name and category are required!")

        add_dialog = tk.Toplevel(tools_window)
        add_dialog.transient(tools_window)
        add_dialog.title("Add Tool")
        add_dialog.geometry("400x500")

        ttk.Label(add_dialog, text="Tool Name:").pack(pady=5)
        entry_name = ttk.Entry(add_dialog, width=50)
        entry_name.pack(pady=5)

        ttk.Label(add_dialog, text="Tool Description:").pack(pady=5)
        entry_description = ttk.Entry(add_dialog, width=50)
        entry_description.pack(pady=5)

        ttk.Label(add_dialog, text="Tool Link:").pack(pady=5)
        entry_link = ttk.Entry(add_dialog, width=50)
        entry_link.pack(pady=5)

        ttk.Label(add_dialog, text="Category:").pack(pady=5)
        category_combobox = ttk.Combobox(add_dialog, values=list(categories.keys()),
state="readonly")
        category_combobox.pack(pady=5)

        submit_button = ttk.Button(add_dialog, text="Submit", command=on_submit)
        submit_button.pack(pady=20)

        add_context_menu(entry_name)
        add_context_menu(entry_description)
        add_context_menu(entry_link)

    def delete_tool():
```

```
    selected_item = tools_tree.selection()
    if selected_item:
        tool_index = tools_tree.index(selected_item[0])
        del tools_list[tool_index]
        update_tools_table()
    else:
        messagebox.showerror("Error", "No tool selected!")

def manage_categories():
    def add_category():
        def on_submit():
            category_name = entry_name.get()
            category_color = colorchooser.askcolor()[1]
            if category_name and category_color:
                categories[category_name] = category_color
                update_category_table()
                add_dialog.destroy()
            else:
                messagebox.showerror("Error", "Both name and color are required!")

        add_dialog = tk.Toplevel(categories_window)
        add_dialog.transient(categories_window)
        add_dialog.title("Add Category")
        add_dialog.geometry("400x200")

        ttk.Label(add_dialog, text="Category Name:").pack(pady=5)
        entry_name = ttk.Entry(add_dialog, width=50)
        entry_name.pack(pady=5)

        submit_button = ttk.Button(add_dialog, text="Submit", command=on_submit)
        submit_button.pack(pady=20)

    def delete_category():
        selected_item = category_tree.selection()
        if selected_item:
            category_name = category_tree.item(selected_item, "values")[0]
            del categories[category_name]
            update_category_table()
        else:
            messagebox.showerror("Error", "No category selected!")

    def update_category_table():
        for i in category_tree.get_children():
            category_tree.delete(i)
        for category, color in categories.items():
```

```
            category_tree.insert("", tk.END, values=(category,), tags=(category,))
            category_tree.tag_configure(category, background=color)

        categories_window = tk.Toplevel(tools_window)
        categories_window.transient(tools_window)
        categories_window.title("Manage Categories")
        categories_window.geometry("400x400")

        category_tree = ttk.Treeview(categories_window, columns=("Category",),
show='headings')
        category_tree.heading("Category", text="Category")
        category_tree.column("Category", anchor=tk.W, width=150)
        category_tree.pack(fill=tk.BOTH, expand=True, pady=10)

        button_frame = ttk.Frame(categories_window)
        button_frame.pack(fill=tk.X, pady=10)

        btn_add_category = ttk.Button(button_frame, text="Add Category",
command=add_category)
        btn_add_category.pack(side=tk.LEFT, padx=10)

        btn_delete_category = ttk.Button(button_frame, text="Delete Category",
command=delete_category)
        btn_delete_category.pack(side=tk.LEFT, padx=10)

        update_category_table()

    def update_tools_table():
        for i in tools_tree.get_children():
            tools_tree.delete(i)
        for tool in sorted(tools_list, key=lambda x: x["category"]):
            tools_tree.insert("", tk.END, values=(tool["name"], tool["description"],
tool["link"]))
            category_color = categories.get(tool["category"], "white")
            tools_tree.item(tools_tree.get_children()[-1], tags=(tool["category"],))
            tools_tree.tag_configure(tool["category"], background=category_color)

    def export_tools():
        file_path = filedialog.asksaveasfilename(defaultextension=".json",
filetypes=[("JSON files", "*.json")])
        if file_path:
            try:
                with open(file_path, 'w', encoding='utf-8') as f:
                    json.dump(tools_list, f, ensure_ascii=False, indent=4)
                messagebox.showinfo("Export Tools", "Tools exported successfully!")
```

```
            except Exception as e:
                messagebox.showerror("Error", f"Failed to export tools: {e}")

    def import_tools():
        file_path = filedialog.askopenfilename(filetypes=[("JSON files", "*.json")])
        if file_path:
            try:
                with open(file_path, 'r', encoding='utf-8') as f:
                    imported_tools = json.load(f)
                    tools_list.extend(imported_tools)
                    update_tools_table()
                messagebox.showinfo("Import Tools", "Tools imported successfully!")
            except Exception as e:
                messagebox.showerror("Error", f"Failed to import tools: {e}")

    tools_window = tk.Toplevel(root)
    tools_window.title("Tools")
    tools_window.geometry("800x400")

    tools_tree = ttk.Treeview(tools_window, columns=("Name", "Description", "Link"),
show='headings')
    tools_tree.heading("Name", text="Name")
    tools_tree.heading("Description", text="Description")
    tools_tree.heading("Link", text="Link")
    tools_tree.column("Name", anchor=tk.W, width=150)
    tools_tree.column("Description", anchor=tk.W, width=300)
    tools_tree.column("Link", anchor=tk.W, width=150)

    def on_item_click(event):
        item = tools_tree.selection()[0]
        link = tools_tree.item(item, "values")[2]
        webbrowser.open(link)

    tools_tree.bind("<Double-1>", on_item_click)
    tools_tree.pack(fill=tk.BOTH, expand=True)

    button_frame = ttk.Frame(tools_window)
    button_frame.pack(fill=tk.X, pady=10)

    btn_add_tool = ttk.Button(button_frame, text="Add Tool", command=add_tool)
    btn_add_tool.pack(side=tk.LEFT, padx=10)

    btn_delete_tool = ttk.Button(button_frame, text="Delete Tool",
command=delete_tool)
    btn_delete_tool.pack(side=tk.LEFT, padx=10)
```

```
    btn_manage_categories = ttk.Button(button_frame, text="Manage Categories",
command=manage_categories)
    btn_manage_categories.pack(side=tk.LEFT, padx=10)

    btn_export_tools = ttk.Button(button_frame, text="Export Tools",
command=export_tools)
    btn_export_tools.pack(side=tk.LEFT, padx=10)

    btn_import_tools = ttk.Button(button_frame, text="Import Tools",
command=import_tools)
    btn_import_tools.pack(side=tk.LEFT, padx=10)

    update_tools_table()

root = tk.Tk()
root.title("OSINT Framework")
root.geometry("1200x800")

style = ttk.Style()
style.configure('TButton', font=('Helvetica', 12), padding=10)
style.configure('TLabel', font=('Helvetica', 12), padding=10)
style.configure('TEntry', font=('Helvetica', 12), padding=10)
style.configure('TCombobox', font=('Helvetica', 12), padding=10)
style.map('Tools.TButton', background=[('active', 'lightblue'), ('!disabled', 'lightblue')])

logo_img = Image.open("logo.jpg")
logo_photo = ImageTk.PhotoImage(logo_img)
root.wm_iconphoto(True, logo_photo)

button_frame = ttk.Frame(root)
button_frame.pack(side=tk.TOP, fill=tk.X, pady=10)

btn_add_target = ttk.Button(button_frame, text="Add Target", command=add_target)
btn_add_target.pack(side=tk.LEFT, padx=10)

btn_add_edge = ttk.Button(button_frame, text="Add Edge", command=add_edge)
btn_add_edge.pack(side=tk.LEFT, padx=10)

btn_remove_target = ttk.Button(button_frame, text="Remove Target",
command=remove_target)
btn_remove_target.pack(side=tk.LEFT, padx=10)

btn_remove_edge = ttk.Button(button_frame, text="Remove Edge",
command=remove_edge)
```

```
btn_remove_edge.pack(side=tk.LEFT, padx=10)

btn_save_graph = ttk.Button(button_frame, text="Save Graph", command=save_graph)
btn_save_graph.pack(side=tk.LEFT, padx=10)

btn_load_graph = ttk.Button(button_frame, text="Load Graph", command=load_graph)
btn_load_graph.pack(side=tk.LEFT, padx=10)

btn_tools = ttk.Button(button_frame, text="Tools", command=open_tools_window,
style='Tools.TButton')
btn_tools.pack(side=tk.LEFT, padx=10)

target_selection_frame = ttk.Frame(root)
target_selection_frame.pack(side=tk.TOP, fill=tk.X, pady=10)

source_target_combobox = ttk.Combobox(target_selection_frame, state="readonly",
width=30)
source_target_combobox.pack(side=tk.LEFT, padx=10)
source_target_combobox.set("Select source target")

target_target_combobox = ttk.Combobox(target_selection_frame, state="readonly",
width=30)
target_target_combobox.pack(side=tk.LEFT, padx=10)
target_target_combobox.set("Select target target")

fig, ax = plt.subplots(figsize=(10, 6))
canvas = FigureCanvasTkAgg(fig, master=root)
canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=1)

canvas.mpl_connect('button_press_event', on_click)

visualize_graph()

root.mainloop()
```