Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Автоматизація збору корпоративної та особистої інформації з відкритих джерел

Виконала: студентка IV курсу, групи СБс-42
спеціальності 125 Кібербезпека
(шифр і назва спеціальності)

(підпис) Васюк К. В. (прізвище та ініціали)

Керівник (підпис) Марценюк В. П. (прізвище та ініціали)

Нормоконтроль (підпис) Кареліна О.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Загородна Н.В. (прізвище та ініціали)

Рецензент (підпис) (прізвище та ініціали)

Тернопіль-2021

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
Кафедра кібербезпеки

ЗАТВЕРДЖУЮ
Завідувач кафедри Загородна Н. В.
(підпис)
«_____» _____ 202__ р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)
за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)
Студентці Васюк Катерині Володимирівні
(прізвище, ім'я, по батькові)
1. Тема роботи Автоматизація збору корпоративної та особистої інформації з відкритих джерел
Керівник роботи Марценюк Василь Петрович, доктор технічних наук, професор кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом по університету від «_____» _____ 202__ року №_____
2. Термін подання студентом роботи
3. Вихідні дані до роботи

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Вступ. 1 Аналіз поняття та методик OSINT. 1.1 OSINT як частина системи кіберзахисту. 1.2 OSINT інструменти та методики. 1.3 Аналіз відомих технічних рішень та програмних продуктів. 1.4 Аналіз вимог до програмного забезпечення OSINT. 1.4.1 Розробка функціональних вимог. 1.4.2 Розробка нефункціональних вимог. 2 Математичне моделювання та програмна автоматизація OSINT. 2.1 Моделювання Математичного рішення. 2.2 Прогнозування загроз. 2.3 Архітектура API. 3 Розробка Автоматизованої системи OSINT. 3.1 Реалізація та моделювання проектних рішень. 3.2 Тестування функціональних можливостей. 3.3 Конфігурація автоматизованої системи 4 Безпека життєдіяльності, осонови охорони праці. 4.1 Значення адаптації в трудовому процесі 4.1.1 Фактори трудової адаптації. 4.2 Підбирання оптимальних параметрів мікроклімату на робочих місцях. 4.3 Показники ефективності та заходи щодо покращенню умов та охорони праці. Висновок.
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності,	Гурик О. Я. к.т.н., доцент		
основи охорони праці			

7. Дата видачі завдання

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка

Студент _____ Васюк К. В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Марценюк В. П.
(підпис) (прізвище та ініціали)

Анотація

Автоматизація збору корпоративної та особистої інформації з відкритих джерел. // Кваліфікаційна робота освітнього рівня «Бакалавр» // Васюк Катерина Володимирівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБс-42 // Тернопіль, 2021 // С. 73, рис. – 32 , табл. – 3, кресл. – 0, додат. – 4, бібліогр. – 26.

Ключові слова: Python, API, Telegram, бот, витік даних, OSINT, розвідка.

У кваліфікаційній роботі розроблено інструмент для збору доступної корпоративної та особистої інформації з використанням OSINT методик.

Інструмент для збору інформації надає можливість своїм користувачам проводити перевірку причетності електронних адрес до витоку даних та оцінити стан їх захищеності. Окрім цього, користувачі можуть знайти інформацію про домен будь якої організації та дізнатися, які сервіси використовуються доменом і на основі цього провести аналіз на наявність вразливостей. Сервіс реалізовано за допомогою інтеграцій в боті месенджера Telegram.

Програмна реалізація виконана мовою програмування Python, з використанням API ефективних та прогресивних OSINT інструментів таких як IntelligenceX та BuildWith.

Annotation

Automation of open sources corporative and personal information collecting // Qualification work of Bachelor educational degree // Vasiuk Kateryna // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Cybersecurity Department, SBs-42 group // Ternopil, 2021 // Pages – 73, figures – 32, tables – 3, sketches – 0, addendums – 4, references – 26.

Keywords: Python, API, Telegram, bot, data leak, OSINT, intelligence.

In the qualification work developed a tool for collecting available corporate and personal information using OSINT methods.

The information collection tool allows users to check the involvement of email addresses in data leakage and assess their security state. In addition, users can find information about any organization`s domain and find out what services are used by the domain and based on this information – analyze for vulnerabilities. The service is implemented through integrations in the Telegram messenger bot.

The tool is written on Python programming language with using APIs of effective and progressive OSINT services such as IntelligenceX and BuildWith.

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

OSINT (Open Source Intelligence) – розвідка на основі відкритих джерел.

POI (Point of Interest) – об’єкти або точки інтересу.

SaaS (Software as a Service) – програмне забезпечення як послуга.

MAU (Monthly Active Users) – це термін, який позначає кількість унікальних клієнтів, які протягом місяця взаємодіяли з товаром чи послугою компанії.

QR code (Quick Response) – двомірний штрих-код, що надає інформацію

CVE (Common Vulnerabilities and Exposures) – база даних вразливостей.

CVSS (Common Vulnerability Scoring System) – числова оцінка вразливості.

ІП – інформаційний пошук.

SQL (Structured query language) – мова програмування структурованих запитів.

KBPS – кілобіти за секунду, базова одиниця вимірювання швидкості передачі інформації.

IDD (Intelligence Driven Defense) – оборонна розвідка.

API (Application Programming Interface) – набір визначень підпрограм, протоколів взаємодії та засобів для створення програмного забезпечення.

XML (Extensible Markup Language) – стандарт побудови мов розмітки ієрархічно структурованих даних для обміну між різними застосунками.

JSON (JavaScript Object Notation) – текстовий формат обміну даними між комп’ютерами.

PGP (Pretty Good Privacy) – комп’ютерна програма, також бібліотека функцій, що дозволяє виконувати операції шифрування і цифрового підпису повідомлень, файлів та іншої інформації.

ВСТУП

Сьогоднішній ландшафт загроз, збір та консолідація зовнішньої інформації є причиною для підтримки ситуаційної обізнаності, зменшення експозиції та часу реагування на інциденти.

Для захисту від сучасних загроз потрібно бути на крок попереду. Активне розуміння організації та цифрових слідів її персоналу є частиною процесу, відомого як оборонна розвідка (IDD). Знання наявної інформації з точки зору зловмисника допоможе команді зрозуміти, де лежать потенційні слабкі сторони та бути готовим у випадку, якщо реальний зловмисник намагається використати ці слабкі сторони.

OSINT – це швидкозростаюча багатогранна техніка пошуку інформації, і все більша кількість організацій, навіть за межами фінансових корпорацій, федеральних та правоохоронних органів, інвестує в інструменти, які можуть полегшити роботу аналітиків та пришвидшити час вирішення питань. Питання, що стосуються автоматизації методик OSINT є актуальними.

Актуальність теми дипломної роботи пов’язана зі значним поширенням явища витоку даних і полягає в необхідності розробки автоматизованої системи збору інформації для скорочення часу роботи аналітиків з кібербезпеки.

Об’єктом дослідження є захищеність корпоративних та особистих даних.

Предметом дослідження є методики та інструменти OSINT для визначення стану захищеності корпоративних та особистих даних.

Завдання дослідження:

- проаналізувати методики і способи пошуку інформації за допомогою існуючих інструментів та технічних рішень;
- розглянути принцип роботи автоматизованих систем для пошуку інформації;
- описати явище витоку корпоративної та особистої інформації, ландшафт загроз у сучасному кібер просторі;
- визначити ефективність автоматизації процесу пошуку інформації;
- розробити інструмент для автоматизації пошуку інформації використовуючи інтеграції готових засобів.

Метою роботи є огляд існуючих програмних рішень для автоматизації OSINT, виявлення актуальних проблем в області кібербезпеки для працівників та компаній, а також розробка власного продукту для створення механізму проведення збору інформації з відкритих джерел для оцінки безпеки даних.

Методологічну основу дослідження склали методи аналізу, спостереження та порівняння для літературних джерел; математичне моделювання та синтез для проектування інженерно-технічного рішення.

Особливість дослідження полягає у нестандартному підході проведення OSINT за допомогою інтеграцій ефективних сервісів з пошуку інформації у месенджер Telegram, використання альтернативних способів виявлення витоків даних та аналіз їх впливу на організації. На сьогоднішній день область безпеки корпоративних даних мало досліджена, тому спостерігається стрімке зростання кіберзлочинності яке тягне за собою великі збитки для організацій.

Пояснювальна записка обсягом 73 сторінки містить 3 таблиці, 32 рисунків, 6 формул, 2 лістинги, 4 додатки. Список використаних джерел розміщується на 3 сторінках і містить 26 джерел.

Розділ 1 АНАЛІЗ ПОНЯТТЯ ТА МЕТОДИК OSINT

1.1 OSINT як частина системи кіберзахисту

Сьогодні так званий “Open Source INTelligence” (OSINT) є одним з найважливіших інструментів кібербезпеки. OSINT – це одна з розвідувальних областей, включаючи пошук, відбір та збір розвідувальної інформації, доступної з загальнодоступних джерел, а також аналіз цієї інформації.

OSINT, як правило, виконується за допомогою моніторингу, аналізу та дослідження інформації, що надходить з Інтернету. Матеріали, зібрані на основі інформації з відкритих джерел, підтримують усі розвідувальні методи та діяльність шляхом накопичення розвідувальних знань, їх аналізу та розповсюдження

Міжнародне співтовариство використовує дедалі більше інформації з відкритих джерел для вирішення широкого спектру проблем [1, 5]. Зокрема, роль OSINT під час здійснення інформаційних операцій визначається сукупністю аспектів, включаючи ефективність потоку інформації, обсяг, чіткість, простоту подальшого використання, вартість отримання тощо.

На процес планування та підготовки заходів OSINT впливають фактори, наведені у таблиці 1.1:

Таблиця 1.1
Фактори, що визначають планування заходів OSINT [2, 12]

Фактор	**Вплив фактору на планування заходів OSINT**
Ефективна інформаційна підтримка	Більшість необхідних довідкових матеріалів щодо об’єктів інформаційної експлуатації зібрано з відкритих джерел. Ця база побудована шляхом збору інформації з ЗМІ. Накопичення даних із відкритих джерел є ключовою функцією OSINT.
Актуальність	Доступність, глибина та масштаб загальнодоступної інформації дозволяють нам знаходити необхідну інформацію без залучення спеціалізованих засобів людської та технічної розвідки.
Спрощення процесів збору даних	OSINT надає необхідну інформацію, усуваючи необхідність залучати зайві технічні та людські методи розвідки.
Глибина аналізу даних	Будучи частиною розвідувального процесу, OSINT дозволяє менеджерам проводити поглиблений аналіз загальнодоступної інформації для прийняття відповідних рішень.

Продовження таблиці 1.1

Фактор	**Вплив фактору на планування заходів OSINT**
Ефективність	Різке скорочення часу доступу до інформації в Інтернеті. Скорочення кількості людських годин, витрачених на пошук інформації, людей та їх взаємозв'язків на основі відкритих джерел. Швидке отримання цінної відповідної інформації.
Обсяг	Можливість масового моніторингу певних джерел інформації, призначених для пошуку необхідного контенту, людей та подій. Досвід показує, що кваліфіковано зібрані фрагменти інформації з відкритих джерел, якщо взяти їх у цілому, можуть виявитися еквівалентними або навіть більш значними, ніж звіти професійної розвідки.
Якість	У порівнянні зі звітами спеціальних агентів, інформація з відкритих джерел виявляється більш правдивою.
Ясність	Хоча у випадках, коли використовується OSINT, надійність відкритих джерел може бути як чіткою, так і незрозумілою, у випадку таємно отриманих даних їх довіра завжди викликає сумніви.
Корисність	Будь-яка таємниця повинна захищатися бар'єрами "класифікацій", дозволами, обмеженим доступом тощо. Що стосується даних OSINT, вони можуть бути легко передані будь-яким зацікавленим організаційним органам. Є можливість провести комплексне дослідження на основі даних з Інтернету.
Вартість	Вартість отримання даних через OSINT мінімальна; це визначається лише ціною використовуваної послуги.

Сьогодні розробка спеціальних методів та засобів пошуку та аналітичного узагальнення мережевої інформації є особливо актуальною. Однак якісних рішень проблеми швидкої аналітичної обробки інформації, пошуку необхідних фактичних даних, виявлення тенденцій розвитку предметних областей та прогнозування досі немає. Проблеми розмірності та динаміки багатомовних інформаційних ресурсів у глобальних мережах вимагають фундаментальних досліджень у галузі математики (теорія графів, складні мережі), розпізнавання образів (класифікація, кластерний аналіз, нейронні мережі), комп'ютерна лінгвістика, обробка цифрових сигналів, нелінійний аналіз, тощо.

В даний час обсяг інформаційних ресурсів, розміщених у глобальних мережах, перевищує сотні трильйонів документів. Тільки соціальна мережа Facebook генерує більше 4 петабайт даних на день.

Соціальні мережі містять велику кількість цінних, з точки зору розвідки, даних, а аккаунти в них часто компрометуються і опиняються в пошкоджених

даних. Нижче наведена статистика кількості активних користувачів в найбільш популярних соціальних мережах:

Facebook (2020) – 1.8 млрд активних користувачів на місяць (monthly active users – MAU)

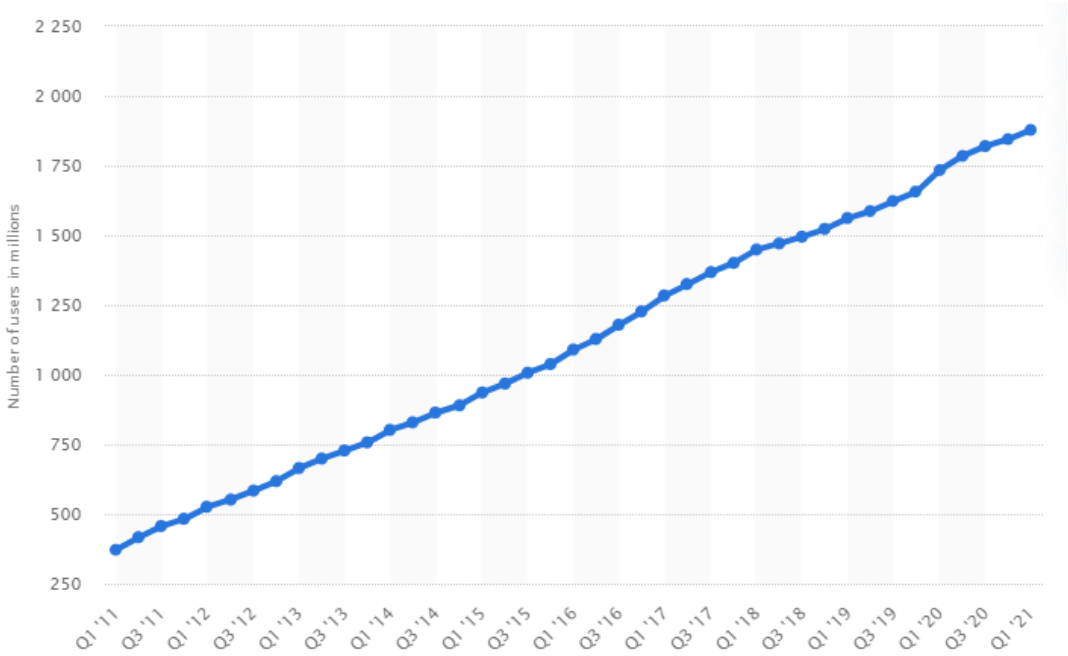


Рисунок 1.1 – Кількість активних користувачів Facebook згідно з statista.com [3]

Instagram (2018) – 1 млрд MAU.

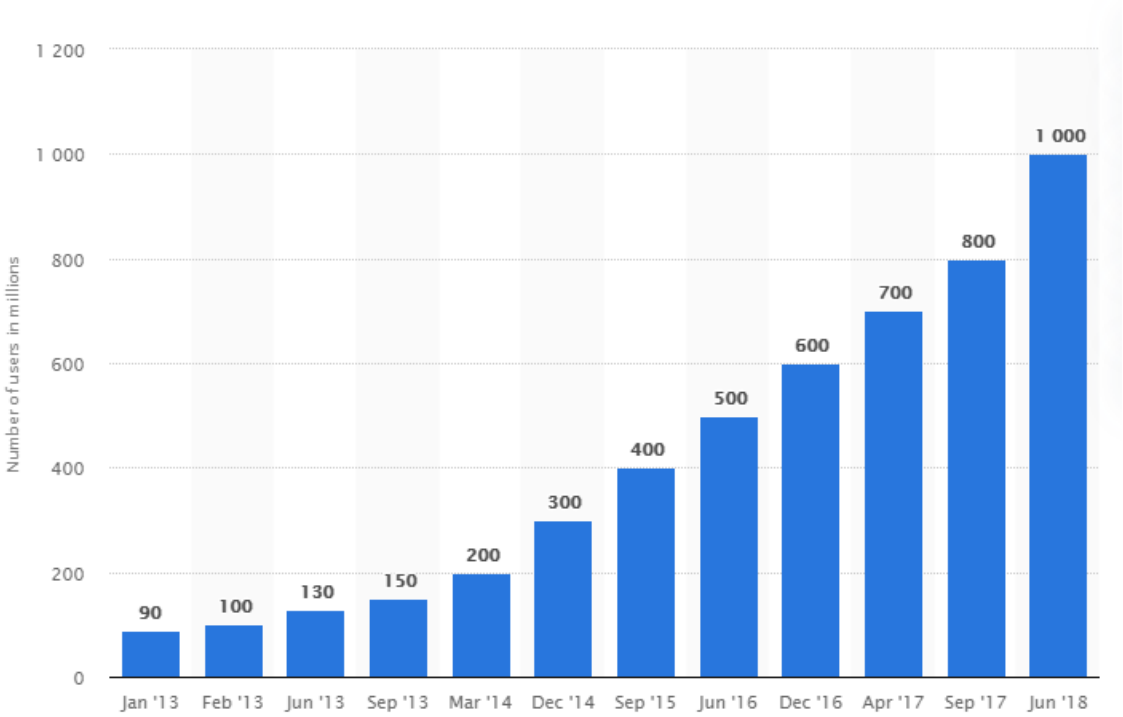


Рисунок 1.2 – Кількість активних користувачів Instagram згідно з statista.com [4]

Мережа професійних контактів Linkedin (2021) – 774.6 млн MAU.

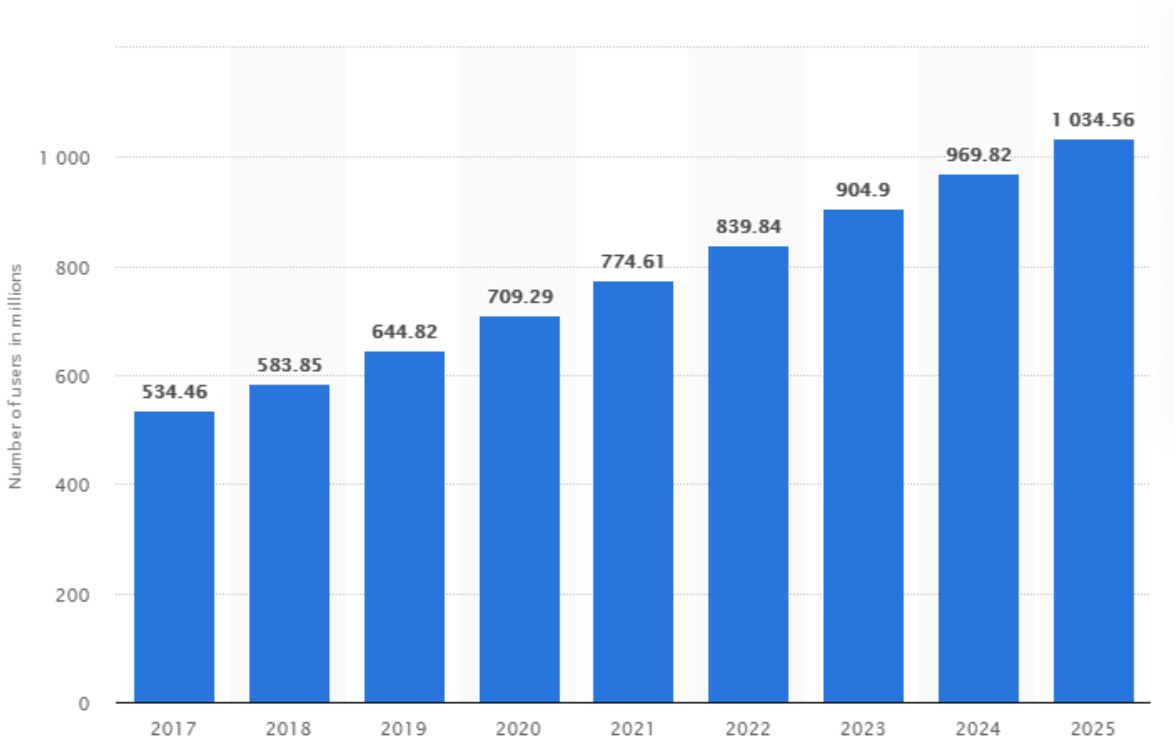


Рисунок 1.3 – Кількість активних користувачів LinkedIn згідно з statista.com [5]

Не так давно порушення, яке скомпрометувало дані кількох мільйонів людей, було б великою новиною. Зараз порушення, які зачіпають сотні мільйонів або навіть мільярди людей, є надто поширеними. На відміну від порушення даних, витік даних – це несанкціонована передача інформації з організації до зовнішніх одержувачів. Цей термін також використовується, коли дані передаються як фізично, так і цифровим шляхом. Часто, такого роду дані знаходяться у колекціях в відкритих джерелах, і також є частиною розвідувальної операції.

Станом на 2021 рік до найбільш серйозних витоків даних відносяться наступні:

1. Compilation of Many Breaches (COMB)

Дата – 2 лютого 2021 року

Кількість записів – 3.2 мільярди

Незаконно отримано доступ до таких даних – електронні адреси, паролі, облікові дані [6].

2. Facebook Data Breach

Дата – 3 квітня 2021 року

Кількість записів – 533 мільйони

Незаконно отримано доступ до таких даних – номери телефонів, електронні адреси, повні імена, адреси, дати народження [7].

3. LinkedIn Data Breach

Дата – 6 квітня 2021 року

Кількість записів – 500 мільйонів

Незаконно отримано доступ до таких даних – номери телефонів, електронні адреси, повні імена, підключені посилання на профілі інших соціальних мереж та інші персональні дані [8].

Факт витоку персональних або корпоративних даних є дуже важливим під час аналізу вразливих місць в системі, і на основі доступних даних можна складати план щодо запобігання можливих сценаріїв розвитку інцидентів.

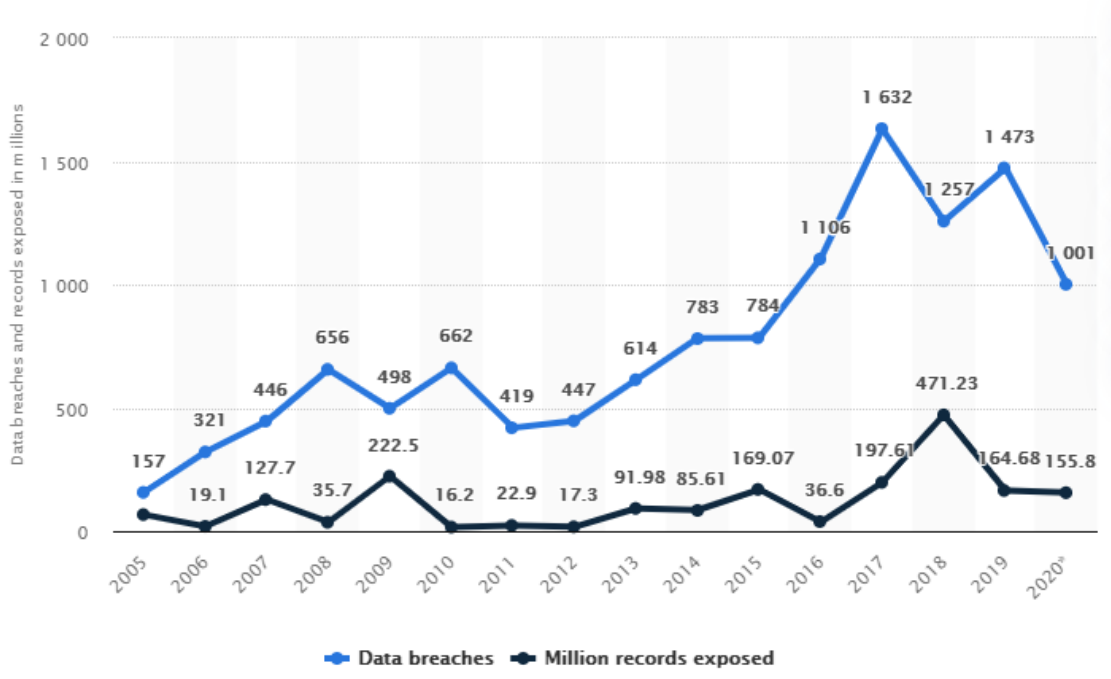


Рисунок 1.4 – Щорічна кількість порушень даних та викритих записів у США з 2005 по 2020 рік згідно з statista.com [9]

У 2020 році кількість випадків порушення даних, згідно з джерелом tdwi.org, у США склала 1001 випадок [10]. Тим часом, упродовж того ж року більше 155,8 мільйонів осіб зазнали витоку даних – тобто випадкового розкриття конфіденційної інформації через менш ніж адекватну інформаційну безпеку.

1.2 OSINT інструменти та методики

Головною умовою проведення OSINT є пошук лише на основі відкритих джерел, це означає, що даний метод розвідки включає в себе лише пасивне сканування та інформацію, яка знаходиться у відкритому доступі.

Пасивне сканування – це метод виявлення вразливостей, який спирається на інформацію, отриману з мережевих даних з цільового комп'ютера без прямої взаємодії.

Програми для сніфінгу пакетів можна використовувати для пасивного сканування для виявлення такої інформації, як операційна система, відомі протоколи, що працюють на нестандартних портах, та активні мережеві програми з відомими помилками. Пасивне сканування може проводитись адміністратором мережі, який перевіряє вразливі місця безпеки або зловмисником як попередню активну атаку.

Пасивне сканування має обмеження. Воно не настільки детальне, як активне сканування вразливостей, і не може виявити програми, які в даний час не розсилають трафік; цим типом сканування також не можна розрізнити неправдиву інформацію, видану для затуманення.

У сучасному швидкому темпі світової економіки одноразової оцінки недостатньо. Натомість необхідний постійний моніторинг для виявлення ризиків для репутації (постачальників, які мають злочинні зв'язки або з корумпованим управлінням) та / або виробництва (постачальники, які не платять своїм працівникам, екологічні проблеми, які можуть скомпрометувати надання послуг тощо). Методи OSINT, засновані на семантичних технологіях, підтримують аналітиків шляхом отримання та моніторингу тисяч фрагментів даних, що містяться у відкритих джерелах, у соціальних мережах та в межах конкретних інформаційних потоків, важливих для ланцюга поставок.

Об'єкти, які становлять інтерес для OSINT розслідування корпоративних даних, можна поділити на 3 категорії:

- люди;
- організації;

– домени.

OSINT техніки для розслідування людей беруть початок з виявлення реальних імен. Якщо справжня особа людини вже відома, а також її ім’я та прізвище, можна використовувати їх як ключові слова для початку пошуку. Потрібно мати на увазі, що пошук людини з використанням її справжнього імені може призвести до багатьох помилкових спрацьовувань. Тому можна поєднати справжнє ім’я людини з додатковою інформацією, наприклад, місцем розташування, щоб скоротити нерелевантні результати, які не збігатимуться.

Наступним кроком є ідентифікація нікнейму. Найпростіший спосіб виявлення імені користувача – це пошук за інформацією, необхідною для пошуку імені користувача. Це може бути комбінація імені та прізвища, можливо, це відповідає доменному імені, яким люди володіють. Тип імені користувача, яким користується людина, може варіюватися залежно від того, наскільки легко вони хочуть бути визначеними.

Як і справжнє ім’я, пошук за іменем користувача може викликати багато помилкових спрацьовувань. На рисунку 1.5 зображено основні використовувані методики в робочому процесі OSINT. Як видно з рисунку, перший етап включає збирання даних. На цьому етапі відбувається накопичення будь якої можливої інформації яка відноситься до об’єкту розслідування. Наступним етапом є аналіз отриманих даних, на основі яких можна розділити інформацію на особисту, корпоративну та мережеву. Впродовж проведення аналізу перевіряється достовірність отриманих даних, знаходження взаємозв’язків та отримання більш глибоких деталей. Заключним етапом є вилучення точних фактів з наявної інформації, формування оцінки ризиків, виявлення слабких сторін для об’єкту розслідування і створення стратегічних рішень для попередження можливості використання чутливих даних проти їх власника.

Пошук буде більш вдалим, якщо присутня електронна адреса для пошуку. На відміну від справжнього імені людини та її імен користувачів, електронна адреса буде унікальною для цієї особи і не матиме однакових помилкових спрацьовувань. Набагато більше шансів отримати потрібну інформацію, якщо доступна електронна

адреса. Як і ім’я користувача, людина може мати більше однієї адреси електронної пошти.

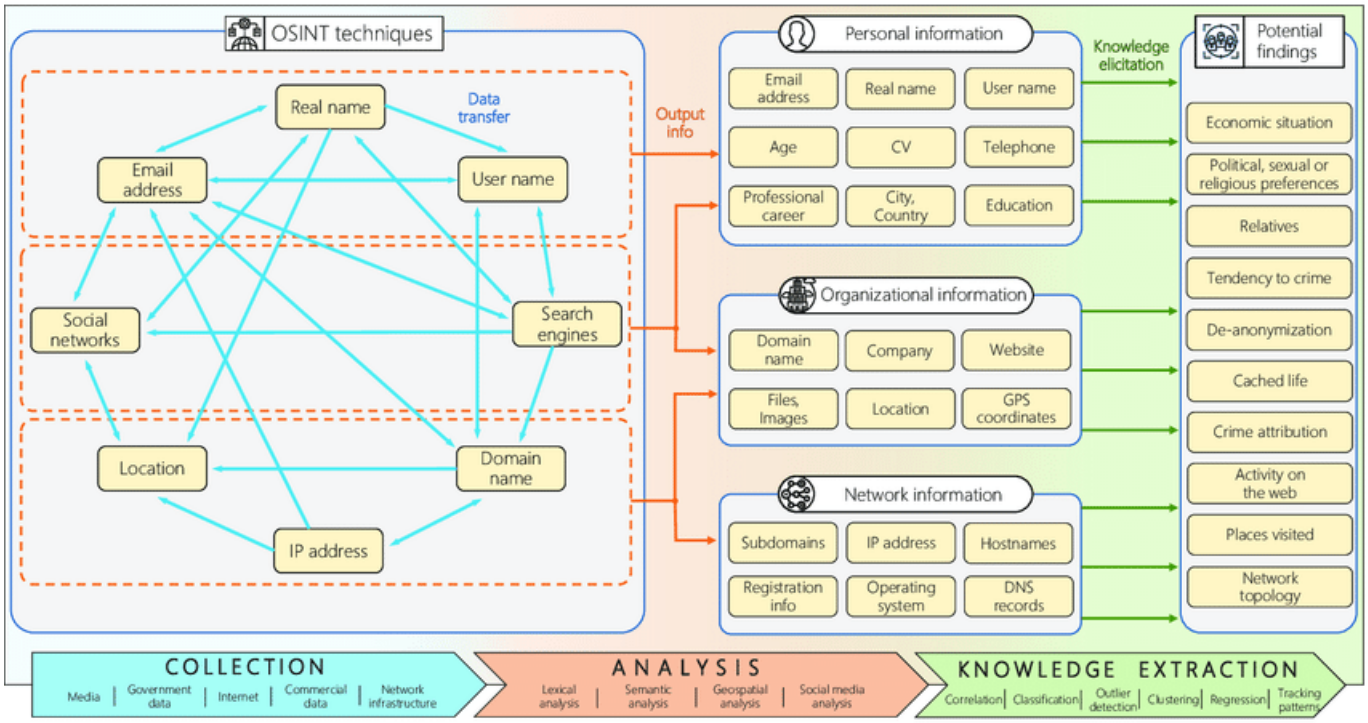


Рисунок 1.5 – Схема застосування методик OSINT [11, 2]

Для збору даних із відкритим кодом існує незліченна кількість юридичних та публічних джерел даних, і пошук правильних даних із потрібних джерел може зайняти багато часу і є в основному ручним процесом.

Загальна пошукова система може бути використана для початку розслідування, виявлення даних або нових джерел. Google – найпопулярніший, який приходить на думку. Розширені пошукові запити Google називають “Google dorks”, і вони використовуються для пошуку точних результатів у назві збору даних. Ціла база даних опцій була каталогізована в базі даних хакерської діяльності Google, і вони використовуються для пошуку OSINT в операціях безпеки та операціях злому.

До прикладу, нижче наведений пошук конфігураційних файлів з паролями. Файли конфігурації не повинні бути загальнодоступними, і файли .ENV – чудовий приклад цього. Якщо ми шукаємо файли .ENV, які містять рядок для пароля бази даних, миттєво знаходимо пароль до цієї бази даних, яку ми виявили.

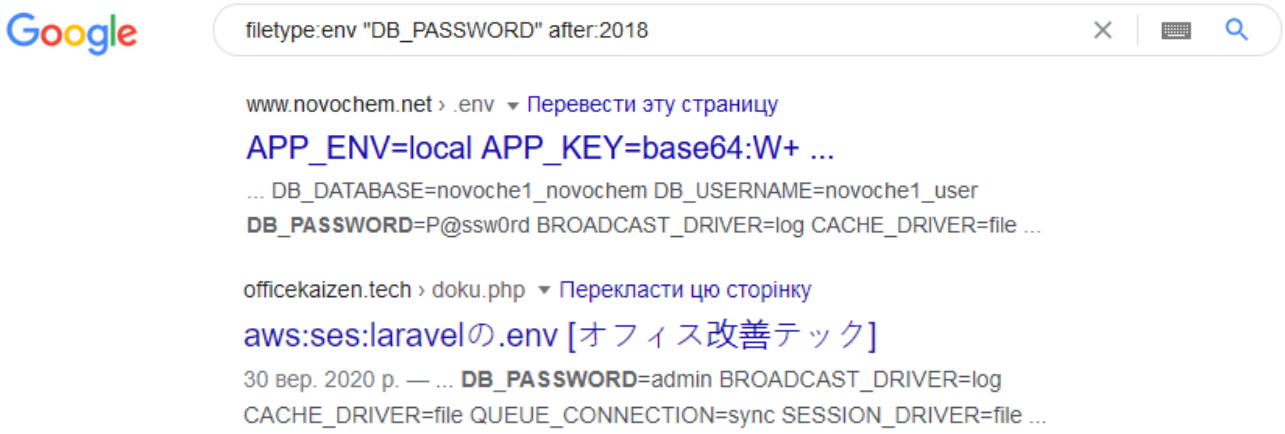


Рисунок 1.6 – Результат пошуку файлів формату .env

Списки електронної пошти – це чудовий спосіб знайти адреси електронних пошт та спробувати вилучити з них інформацію про корпоративні подробиці. Ці списки часто виставляються компаніями чи школами, які намагаються організувати списки розсилки для своїх членів.

Щоб їх знайти, потрібно шукати електронну таблицю типу файлу .XLS із рядком “email.xls” у URL-адресі.

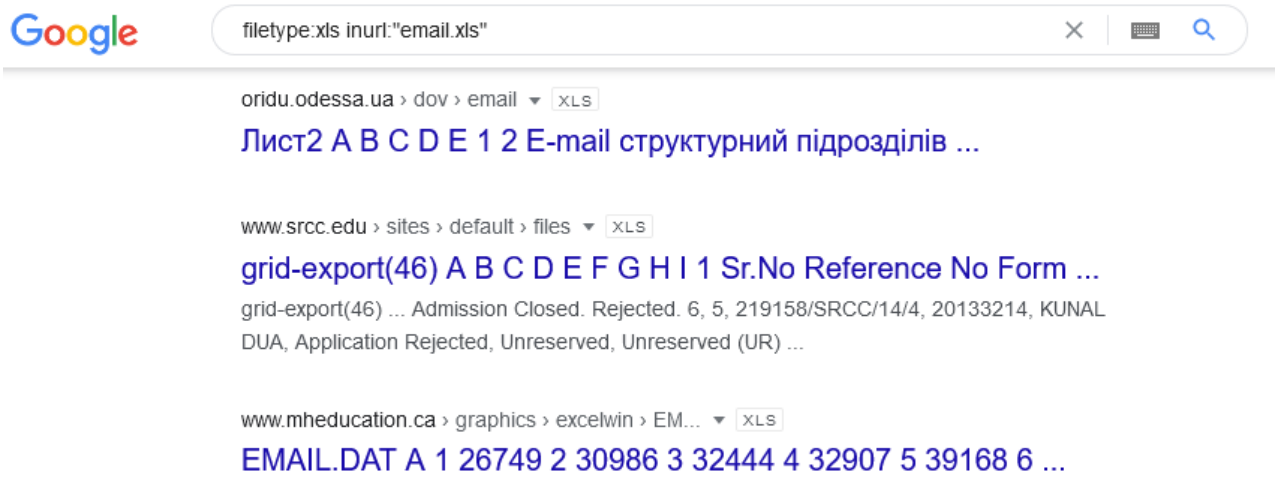


Рисунок 1.7 – Результат пошуку файлів які містять email адреси

Окрім Google, можна знайти різні результати в інших пошукових системах. Популярний загальний пошуковий механізм у спільноті OSINT, DuckDuckGo, також може бути використаний для отримання тих самих результатів.

На основі порушення даних, відкрита інформація, звичайно, також може бути джерелом збору розвідувальних даних. Існують такі ресурси, як haveibeenpwned.com, які надають інформацію про причетність електронних пошт до витоків, в які вони були втягнуті. Ця інформація дозволяє аналітику кібербезпеки або слідчому, можливо, визначити, скільки разів облікові дані користувача електронної пошти було пошкоджено, а також види послуг та платформ, якими користується або користувалась людина в минулому. Це може бути важливим цифровим слідом, і якщо інформація про людину розміщується не вперше, її особа стає дуже вразливою.

Розуміння цифрового простору, де часто може перебувати об’єкт розслідування, є важливим, і спостереження за соціальною активністю в соціальних мережах корисне, оскільки воно може розповісти про точки зору та діяльність людини чи компанії.

Інтернет – це джерело інформації про людей. Перебирання Інтернету може зайняти багато часу, щоб ідентифікувати та вловити цифровий слід людини, але загальнодоступні Інтернет-дані все ще є найпростішим способом пасивного дослідження та перевірки третіх сторін, нових працівників та шахраїв. Не вся інформація буде особливо цінною залежно від мети розслідування. Як правило, до кінцевого звіту потрібно буде включити особисту інформацію, таку як електронну адресу об’єкту розслідування, справжнє ім’я, ім’я користувача, вік, резюме, номер телефону, місцезнаходження, освіту та кар’єру.

Запис цифрового сліду вручну, як описано в цьому розділі, займає багато часу і не завжди ефективно. Автоматизовані інструменти можуть допомогти встановити зв’язок між даними, які можуть бути пропущені. В таблиці 1.2 наведені аспекти, які відображають інтерес при дослідженні POI організації згідно з джерелом mediasonar.com [12].

Таблиця 1.2

Аспекти при дослідженні POI організації

Офіційна інформація про бренд	Це стосується офіційних активів бренду, таких як веб-сайт або офіційні матеріали бренду, такі як звіти інвесторів. Це також може стосуватися новин про організацію, повідомлених у прес-релізах або ЗМІ.
Людський фактор	Людський фактор важко визначити, бо він непередбачуваний. Це стосується діяльності агентів організації, таких як працівники або керівники. Людський фактор любить користуватися власними пристроями, можливо, не завжди має на увазі найкращий інтерес до організації та схильний до помилок.

Продовження таблиці 1.2

Технічні сліди	Саме тут інтелект з відкритим кодом перетинається з кібербезпекою. Це може включати перегляд відкритих портів, DNS та IP, мережевих служб, можливостей віддаленого доступу та вразливостей.

Цифровий слід організації включає будь-яку пов’язану з ними діяльність в Інтернеті та комунікації, незалежно від них або про них. Колись це означало веб-сайт і знання того, що люди говорять про них у соціальних мережах або на сайтах з відгуками, але хмара все це змінила. Сьогодні цифровий слід організації повинен враховувати всі пристрої, підключені до Інтернету, і всі сторонні SaaS, включені в їх IT-інфраструктуру. Він повинен охоплювати зростаючу кількість точок даних у дедалі більш розрізненій екосистемі Інтернету.

OSINT використовується для отримання інформації про домени, що допомагає визначати стратегії кібербезпеки, розуміти інциденти та навіть з’ясувати джерело загрози. Він також використовується для визначення того, чи є сенс співпрацювати чи інтегруватися зі сторонніми організаціями, не кажучи вже про аналіз, який здійснюється фінансовою установою як частина діяльності “Знай свого клієнта” (KYC).

Зображення нижче демонструє інформацію про типи порушення даних за адресою @tntu.edu.ua, а саме кількість документів, поштових скриньок та веб ресурсів стосовно яких був витік даних.

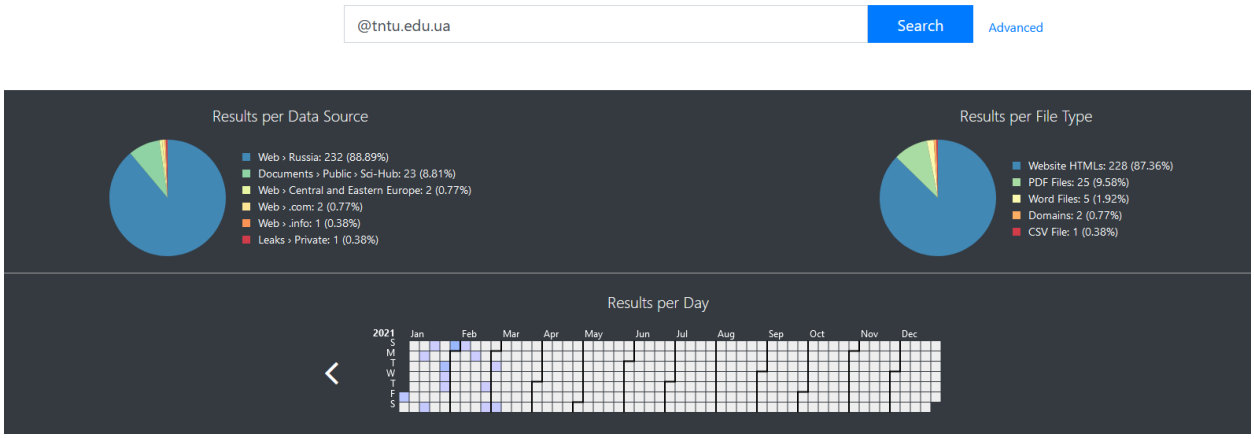


Рисунок 1.8 – Інформація про витік даних на домені tntu.edu.ua

Дещо з того, що буде в подальшому використовуватись, перелічено в OSINT Framework, визначному ресурсі для слідчих OSINT для вивчення інструментів та концепцій своїх досліджень. Він був створений Джастіном Нордіном, щоб допомогти будь-кому розпочати роботу, і став опорою та пусковим пунктом для багатьох людей та організацій. Цей фреймворк забезпечує достатньо інформативний огляд під час розслідування доменів.

Перша річ, яка має бути встановлена, це час коли був зареєстрований домен і ким саме. Потрібно мати на увазі, що інформація може бути прихована або реєстратори могли надати неправдиву інформацію – ці дані можуть вказувати на те, що домен прив’язаний до конкретної третьої сторони. Можна використовувати такі сервіси, як Who.is або ICANN Lookup, щоб знайти основну інформацію про реєстрацію домену. Якщо доступні не всі дані, це або означає, що інформація повністю прихована, або вона вам не доступна.

Інструменти пошуку, такі як Hunter.io, дозволяють виконувати пошук у домені, щоб знаходити пов’язані електронні пошти, і допомагають скласти всебічну картину того, хто пов’язаний з доменом або веб-сайтом. Це може забезпечити нові шляхи розслідування для вашого дослідження, і ви зможете зібрати профіль компанії у більш цілісному вигляді [13, 54].

У подальному дослідженні POI буде доречно дізнатись більше про те, які типи технологій підтримують домен. BuiltWith – це один з найпопулярніших і найпростіший у використанні інструментів для отримання інформації про

постачальників технологій, і навіть просто доступ до безкоштовних функцій пошуку може надати низку підказок про домен. Для більшості доменів існує безліч звичних інструментів: Google Analytics, платформа автоматизації маркетингу, система управління контентом та довгий перелік інших інструментів маркетингу, відстеження та звітування. Але серед усіх цих служб це може включати речі, які роблять організацію вразливою, або, що ще гірше, виявляють, що домен таємно діє зі зловмисними намірами. Якщо розглядати потенційного партнера або постачальника, особливо якщо домен пов'язаний з онлайновими службами та платформами, це може стати ключовим кроком та надати деяку інформацію.

Щоб отримати цифровий слід домену, потрібно знати не лише кореневий домен. Відображення всієї широти домену можна досягти кількома різними способами.

Можливо, найкращий спосіб зробити це – активне сканування, але є субдоменні сканери та інструменти, які можуть допомогти легалізувати роботу.

Компанії часто мають такі речі, як портали для співробітників, внутрішні субдомени або тестові сервери, до яких вони не хочуть, щоб усі мали доступ. Іноді служби на цих субдоменах вразливі, і це може сигналізувати про проблему у постачальника. Там, де домен є підозрілим, субдомени можуть допомогти підтвердити деякі з цих підозр або привести до розуміння масштабу проблеми.

Аналітики в просторі безпеки зазвичай використовують візуальний аналіз для відображення структури та зв'язків у своїх точках даних. Техніка візуалізації використовувалась давно, задовго до того, як були винайдені комп'ютери, і використовується у багатьох різних сценаріях. Це набагато корисніше, ніж карта сайту або список сторінок, намагаючись зрозуміти архітектуру веб-сайту.

1.3 Аналіз відомих технічних рішень та програмних продуктів

Збір даних – це перша частина процесу розслідування, яка включає багато елементів, включаючи знаходження, зберігання, обробку, аналіз та розподіл даних.

І звичайно, автоматизація означає, що ми будемо робити це автоматично, з мінімальним ручним введенням.

Проблемою такого рішення є те, що збір даних для розслідування може швидко призвести до перевантаження інформації. Хаотичний підхід збирати все, що може бути корисним, тут не працює. Останнє, що потрібно аналітикам – це витратити дорогоцінний час на сортування та фільтрування великих обсягів даних, поки вони намагаються активно зупинити розповсюдження кіберзагрози або швидко дійти до кореня проблеми. Отже, для збору даних автоматизація насправді передбачає неабияку кількість людської участі, принаймні на початку процесу.

До технологічних рішень, які б виконували функції пошуку на основі відкритих джерел нижче наведено деякі інструменти:

theHarvester – це інструмент, розроблений на Python. З його допомогою можна збирати таку інформацію, як електронні листи, субдомени, хости, імена співробітників, відкриті порти з різних загальнодоступних джерел, таких як пошукові системи, сервери ключів PGP та база даних SHODAN.

Maltego є інструментом OSINT та комп'ютерної криміналістики. Він забезпечує інтерактивний аналіз даних із розширеними візуалізаціями, що дозволяють ефективно аналізувати посилання [14].

Програмне забезпечення використовується для онлайн досліджень взаємозв'язку між даними з різних джерел в Інтернеті.

Він може виявити зв'язки між людьми та компаніями та знайти загальнодоступну інформацію.

Зображення нижче містить приклад робочого вікна програмного продукту та подання інформації у вигляді графа:

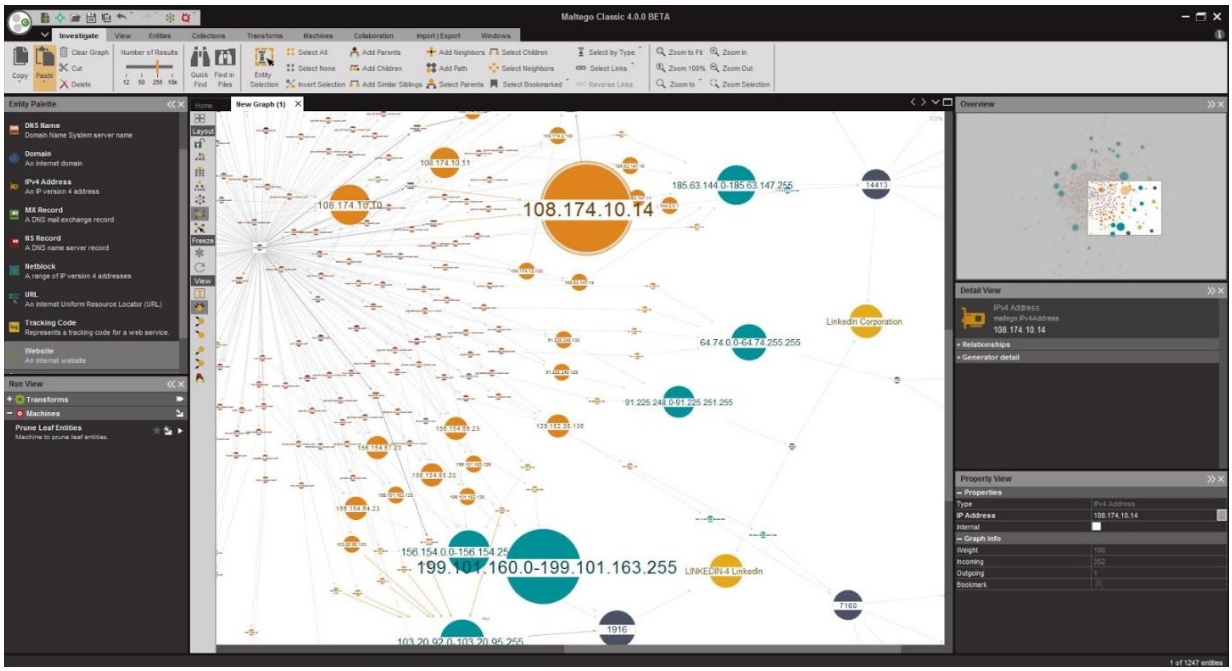


Рисунок 1.9 – Приклад роботи Maltego

SpiderFoot є інструментом розвідки з відкритим кодом. Його часто називають дактилоскопією з найширшою колекцією OSINT [15].

Інструмент може автоматично надсилати запити до понад 100 відкритих джерел та збирати інформацію про IP-адреси, доменні імена, веб-сервери, адреси електронної пошти тощо. Програмне забезпечення написане на Python.

Нижче зображена візуалізація знайдених результатів за доменом tntu.edu.ua:

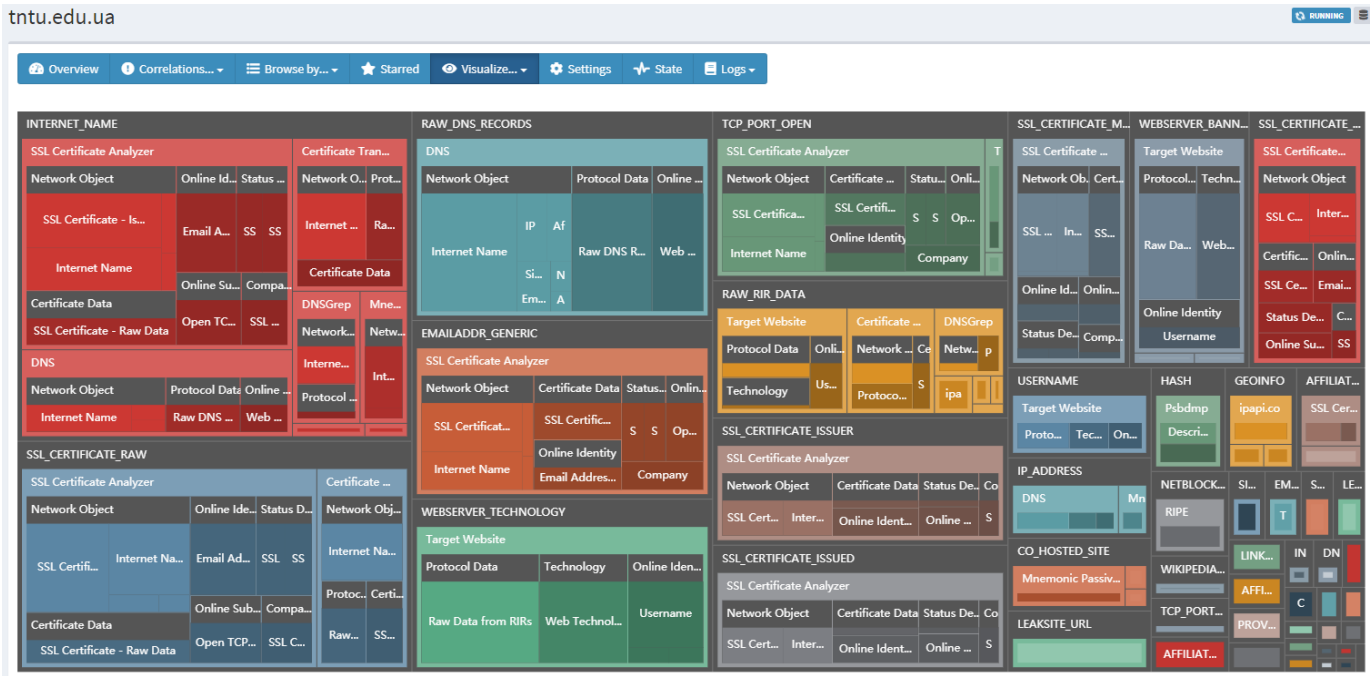


Рисунок 1.10 – Візуалізація даних в SpiderFoot

1.4 Аналіз вимог до програмного забезпечення OSINT

Функціональні вимоги описують, як повинен поводитися продукт, які його особливості та функції. Приклади використання описують взаємодію між системою та зовнішніми користувачами, яка веде до досягнення певних цілей.

Нефункціональні вимоги – це специфікація, яка описує можливості роботи системи та обмеження, що покращують її функціональність. Це можуть бути швидкість, безпека, надійність тощо.

На рисунку 1.12 зображено основні відмінності між функціональними та нефункціональними вимогами:

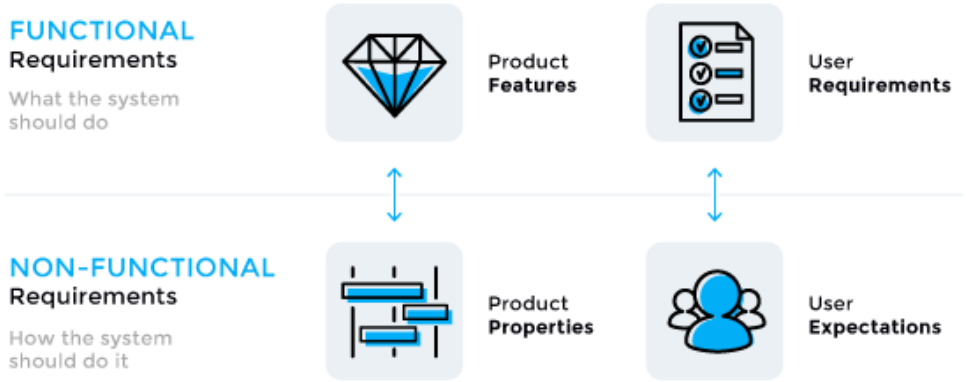


Рисунок 1.11 – Різниця між функціональними та нефункціональними вимогами

1.4.1 Розробка функціональних вимог

В даному випадку проект, що полягає в автоматизації OSINT складається з трьох основних завдань, які в свою чергу містять у собі ряд виконуваних функцій. Для того, щоб ці функції працювали коректно, потрібно створити вимоги, які б допомогли досягнути бажаного результату. Рисунок 1.12 демонструє способи використання продукту, а у таблиці 1.3 наведено детальний опис вимог. Формат таблиці включає в себе опис вимог відповідно до кожного можливого способу використання, що передбачений у схемі.

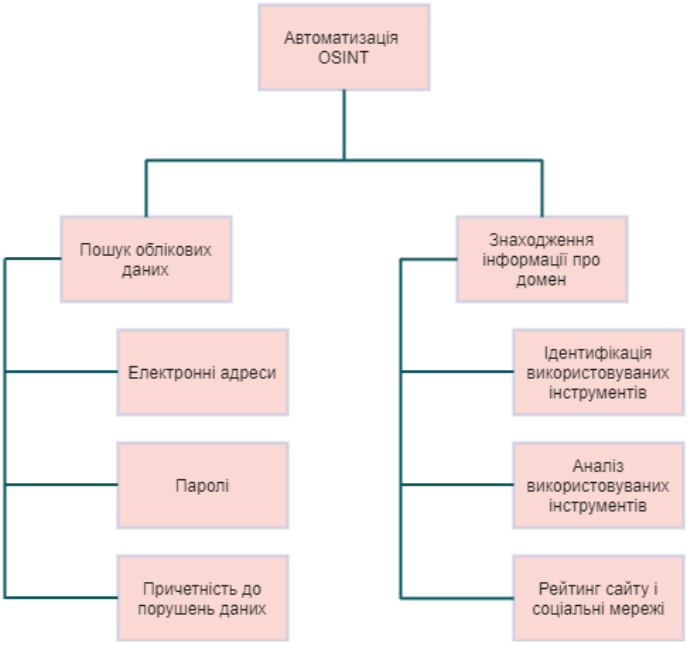


Рисунок 1.12 – Візуалізація способів використання

Таблиця 1.3

Опис вимог згідно зі способами використання

Спосіб використання	Вимоги
Пошук облікових даних	• Розпізнавання формату введених даних; • Пошук фактів причетності до порушень даних за допомогою відкритих джерел; • Виведення доступної інформації в форматі «email@example.com:password»
Знаходження інформації про домен	• Розпізнавання формату введених даних • Сканування об’єкту розслідування на основі відкритих джерел

1.4.2 Розробка нефункціональних вимог

Корисність можна розглядати як найважливішу нефункціональну вимогу. Основна увага при розробці боту полягає у створенні хорошого зрозумілого інтерфейсу для користувачів та вивченні того, як дизайн інтерфейсу та взаємодія може бути найкращим чином реалізована за допомогою даного середовища. Окрім цього варто розв’язати ряд наступних задач:

- впровадити спосіб виведення інформації у зручному для читання форматі;
- створити зрозумілий модуль реагування на некоректність введених даних та інформування про це користувача;
- забезпечити безперервну доступність сервісу.

Розділ 2 МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ПРОГРАМНА АВТОМАТИЗАЦІЯ OSINT

2.1 Моделювання математичного рішення

Інформаційний пошук – це процес отримання ресурсів інформаційної системи, які мають відношення до інформаційної потреби, із сукупності цих ресурсів. Пошуки можуть базуватися на повнотекстовому або іншому індексуванні на основі вмісту. Інформаційний пошук є наукою про пошук інформації в документі, пошук самих документів, а також пошук метаданих, що описують дані, та баз даних текстів, зображень чи звуків.

Автоматизовані системи пошуку інформації використовуються для зменшення того, що називалося перевантаженням інформації. ІП-система – це програмна система, що забезпечує доступ до книг, журналів та інших документів; зберігає та управляє цими документами. Веб-пошукові системи є найбільш видимими ІП-додатками.

Пошук інформації являє собою процес виявлення в деякій множині документів (текстів) всіх тих, які присвячені зазначеній темі (предмету), задовольняють заздалегідь визначеним умовам пошуку (запиту) або містять необхідні (відповідні інформаційної потреби) факти, відомості, дані.

Процес пошуку включає послідовність операцій, спрямованих на збір, обробку та надання інформації.

У загальному випадку пошук інформації складається з чотирьох етапів:

- визначення (уточнення) інформаційної потреби і формулювання інформаційного запиту;
- визначення сукупності можливих власників інформаційних масивів (джерел);
- вилучення інформації з виявлених інформаційних масивів;
- ознайомлення з отриманою інформацією і оцінка результатів пошуку.

Процес пошуку інформації починається, коли користувач вводить запит до системи. Запити – це формальні заяви про інформаційні потреби, наприклад рядок

пошуку в веб-пошукових системах. Під час отримання інформації запит не однозначно ідентифікує окремий об’єкт у колекції. Натомість декілька об’єктів можуть відповідати запиту, можливо, з різним ступенем релевантності.

Об’єкт – це сутність, яка представлена інформацією у колекції вмісту або базі даних. Запити користувачів узгоджуються з інформацією бази даних. Однак, на відміну від класичних запитів SQL до бази даних, при пошуку інформації результати, що повертаються, можуть збігатися з запитом, а може і не відповідати, тому результати зазвичай класифікуються. Цей рейтинг результатів є ключовою відмінністю пошуку інформації у порівнянні з пошуком у базі даних [16, p. 61].

Залежно від програми об’єктами даних можуть бути, наприклад, текстові документи, зображення, аудіо, розумові карти або відео. Часто самі документи не зберігаються і не зберігаються безпосередньо в ІП-системі, а натомість вони представлені в системі сурогатами документів або метаданими.

Класична задача ІП, з якої почався розвиток цієї галузі – це пошук документів, що задовольняють запиту, в рамках деякої статичної колекції документів. Але список завдань ІП постійно розширюється і тепер включає:

- питання моделювання;
- класифікація документів;
- фільтрація документів;
- кластеризація документів;
- проектування архітектури пошукових систем і призначених для користувача інтерфейсів;
- витяг інформації, зокрема анотування і реферування документів;
- мови запитів.

Також, перед інструментами ІП ставляться деякі завдання, що включають в себе морфологічний аналіз, дозвіл лексичної багатозначності і так далі.

Більшість ІП-систем обчислюють числову оцінку того, наскільки кожен об’єкт у базі даних відповідає запиту, і ранжують об’єкти відповідно до цього значення. Потім користувачеві відображаються об’єкти найвищого рейтингу. Пізніше процес може бути повторений, якщо користувач бажає уточнити запит.

Для ефективного отримання відповідних документів за допомогою ІП-стратегій документи, як правило, перетворюються у відповідне подання. Кожна стратегія пошуку включає певну модель для своїх цілей подання документів. Рисунок нижче ілюструє взаємозв'язок деяких типових моделей. На рисунку 2.1 моделі класифікуються за двома вимірами: математична основа та властивості моделі.

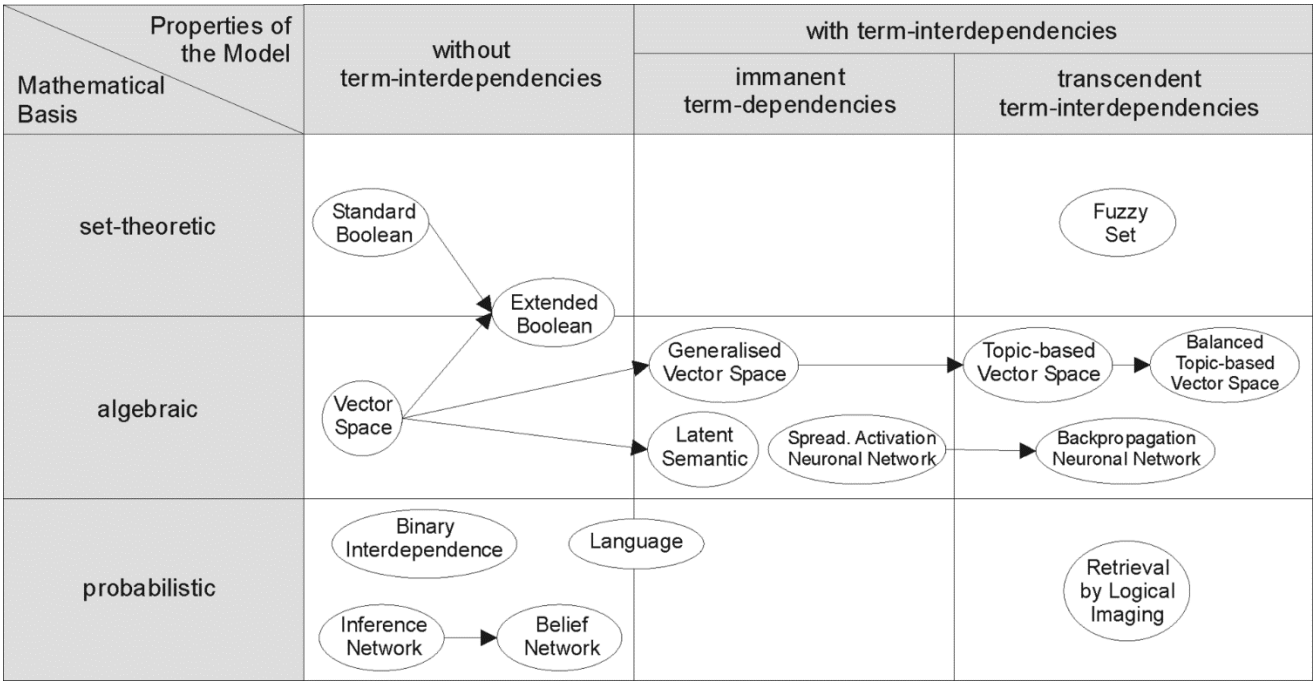


Рисунок 2.1 – Категоризація ІП-моделей [17]

Математична основа моделі пошуку включає в себе:

1. Теоретичні моделі множин представляють документи як набори слів або фраз. Подібність зазвичай випливає з теоретико-множинних операцій над цими множинами. Поширені моделі:

– стандартна булева модель;

– розширена булева модель;

– нечітке отримання.

2. Алгебраїчні моделі представляють документи та запити, як правило, у вигляді векторів, матриць або кортежів. Подібність вектору запиту та вектору документа представляється як скалярне значення:

– вектор космічної моделі;

– узагальнена векторна космічна модель;
– тематична векторна космічна модель;
– розширена булева модель;
– латентна семантична індексація, тобто латентний семантичний аналіз.

3. Імовірнісні моделі трактують процес пошуку документів як імовірнісний висновок. Подібності обчислюються як ймовірність того, що документ має відношення до даного запиту. У цих моделях часто використовуються ймовірнісні теореми, такі як теорема Байєса.

– бінарна модель незалежності;
– невизначений висновок;
– мовні моделі;
– модель дивергенції від випадковості;
– приховане виділення Діріхле.

4. Моделі пошуку на основі функцій розглядають документи як вектори значень функцій функції (або просто функцій) і шукають найкращий спосіб поєднати ці функції в єдиний бал релевантності, як правило, навчившись методам ранжування. Функціональні функції – це довільні функції документа та запиту, і як такі можуть легко включати майже будь-яку іншу модель пошуку як просто ще одну функцію.

Моделі без взаємозалежності термінів розглядають різні терміни / слова як незалежні. Цей факт зазвичай представляється у векторних просторових моделях припущенням ортогональності термінових векторів або в імовірнісних моделях припущенням про незалежність для змінних терміну.

Моделі з іманентними взаємозалежностями термінів дозволяють представити взаємозалежності між термінами. Однак ступінь взаємозалежності між двома термінами визначається самою моделлю. Зазвичай це прямо чи опосередковано (наприклад, шляхом зменшення розмірів) із спільного вживання цих термінів у цілому наборі документів.

Моделі із трансцендентною взаємозалежністю термінів дозволяють представити взаємозалежності між термінами, але вони не стверджують, як

визначається взаємозалежність між двома термінами. Вони покладаються на зовнішнє джерело для ступеня взаємозалежності двох термінів (наприклад, людина або складні алгоритми).

Існує багато способів оцінити наскільки добре документи, знайдені ІП, відповідають запиту. На жаль, поняття ступеня відповідності запиту є суб'єктивним поняттям, а ступінь відповідності залежить від конкретної людини, яка оцінює результати виконання запиту.

Точність

Визначається як відношення числа релевантних документів, знайдених ІП, до загальної кількості знайдених документів:

$$\text{Precision} = \frac{|D_{rel} \cap D_{retr}|}{|D_{retr}|} \tag{2.1}$$

де D_{rel} – це множина релевантних документів в базі, а D_{retr} – множина документів, знайдених системою.

Повнота

Відношення числа знайдених релевантних документів до загального числа релевантних документів в базі:

$$\text{Recall} = \frac{|D_{rel} \cap D_{retr}|}{|D_{rel}|} \tag{2.2}$$

де D_{rel} – це множина релевантних документів в базі, а D_{retr} – множина документів, знайдених системою.

Випадіння

Випадіння характеризує ймовірність знаходження нерелевантного ресурсу і визначається як відношення числа знайдених нерелевантних документів до загального числа нерелевантних документів в базі:

$$\text{Fall-out} = \frac{|D_{nrel} \cap D_{retr}|}{|D_{nrel}|} \tag{2.3}$$

де D_{nrel} – це множина нерелевантних документів в базі, а D_{retr} – множина документів, знайдених системою.

Міра Ван Різбергена

Іноді буває корисно об’єднати точність і повноту в одній усередненій величині. Для цієї мети середнє арифметичне не підходить, так як, наприклад, пошуковій системі досить повернути взагалі всі документи, щоб забезпечити рівну одиниці повноту при близькій до нуля точності, і середнє арифметичне точності і повноти буде не менше 1/2. Середнє гармонійне не володіє цим недоліком, оскільки при великій відмінності усереднених значень наближається до мінімального з них.

Тому хорошим заходом для спільної оцінки точності і повноти є *F*-міра, яка визначається як зважене гармонійне середнє точності *P* і повноти *R*:

$$F = \frac{1}{a\frac{1}{P}+(1-a)\frac{1}{R}}, \quad a \in [0,1]. \tag{2.4}$$

Зазвичай *F*-міру записують у вигляді

$$F = \frac{(\beta^2+1)PR}{\beta^2 P+R}, \beta^2 = \frac{(1-a)}{a}, \beta^2 \in [0, \infty]. \tag{2.5}$$

При $a = 1/2$ або $\beta = 1$ *F*-міра надає однакову вагу точності і повноті і називається збалансованою або F_1- мірою (в нижній індекс прийнято вказувати величину β), вираз для неї спрощується:

$$F_1 = \frac{2PR}{P+R} \tag{2.6}$$

Використання збалансованої *F*-міри не є обов’язковим: при $0 < \beta < 1$ віддається перевага точності, а при $\beta > 1$ більшу вагу отримує повнота. [18]

Центральне завдання ІП – допомогти користувачеві задовольнити його інформаційну потребу. Так як описати інформаційні потреби користувача технічно непросто, вони формулюються як деякий запит, який представляє з себе набір ключових слів, що характеризує те, що шукає користувач.

2.2 Прогнозування загроз

У цьому підрозділі висвітлюються використані джерела даних для прогнозування загроз, архітектура прогнозування загроз та деталі моделей машинного навчання, навчених моделювати загрози з використанням відкритої інформації з баз даних вразливостей з відкритим кодом, таких як ExploitDB, VulnDB

та CVE, а також анотовані дані з Інтернет-джерел, таких як соціальні мережі, форуми та блоги.

Рисунок 2.2 ілюструє, як уразливість інфраструктури організації призводить до експлуатування і використовуються акторами загроз для нападу на інфраструктуру організації. Він також ілюструє, як прогнозування загроз використовує наявні анотовані джерела даних (такі як VulnDB та ExploitDB), щоб навчитися передбачати загрози та застосовувати ці знання для виявлення загроз, знайдених в різних OSINT джерелах і в DarkWeb.

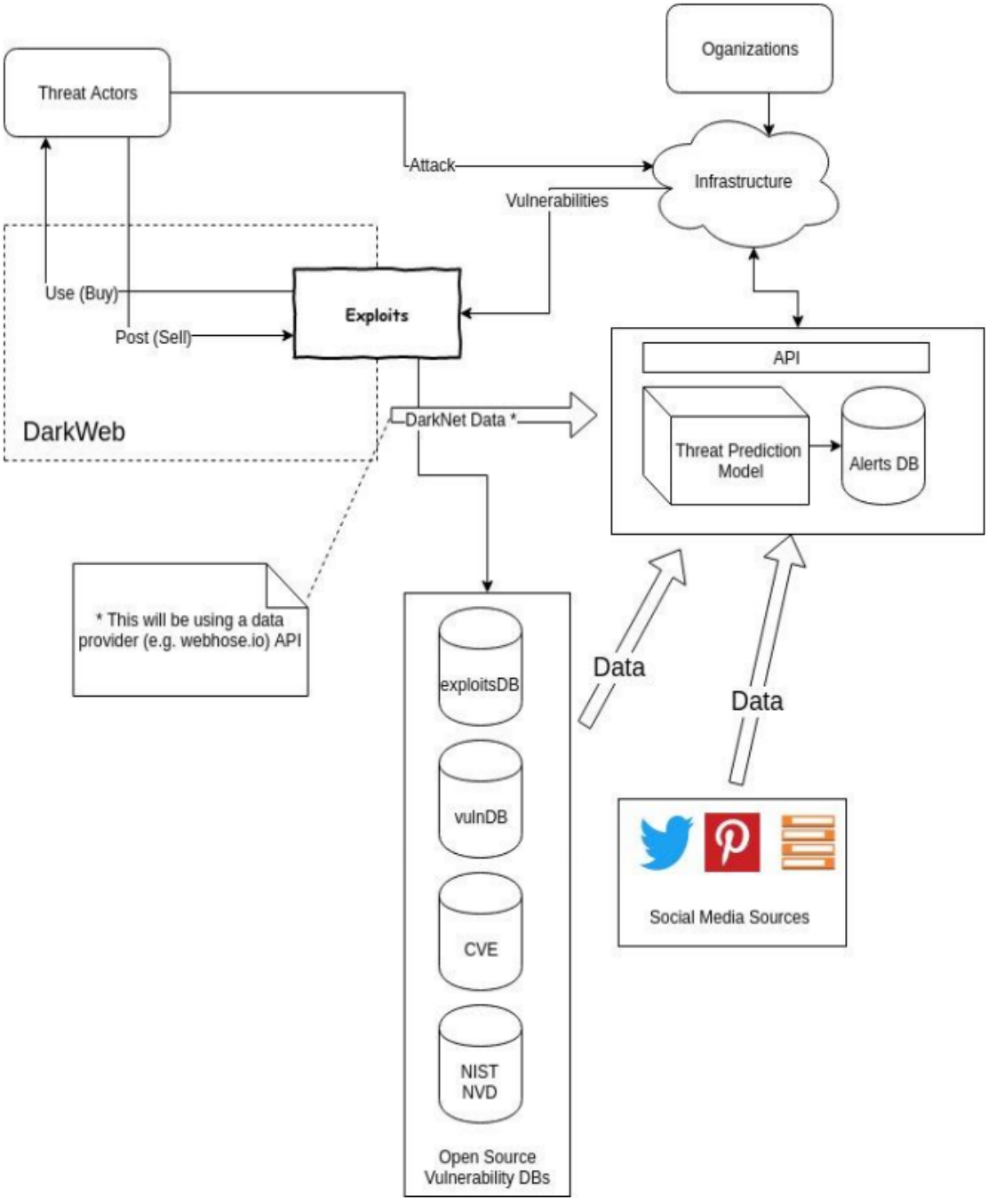


Рисунок 2.2 – Огляд суб’єктів та їх взаємозв’язків, а також роль прогнозатора загроз

Моделі прогнозування можуть бути розроблені для прогнозування різних результатів проекту та проміжних результатів за допомогою статистичних методів. Модель ефективності процесу приймає поняття ймовірності. Це також можна дослідити далі, будуючи моделювання. Результат можна вивчити як діапазон. Залежно від прогнозів можуть бути рекомендовані корекції. Модель може бути змодельована для прогнозування кінцевих результатів на основі запропонованих виправлень. Таким чином, це активна модель, яка допомагає технічному аналітику аналізувати дані та прогнозувати результати. Аналітики можуть змінювати дані та проводити аналіз. Потім вони можуть записати ці екземпляри та вибрати найкращий варіант. Модель допомагає аналітикам вирішити, який важіль регулювати для досягнення кінцевої мети проекту.

У поточній системі фокус на виправленні вразливостей не заснований на потенційному впливі вразливості. Крім того, виправленню всіх вразливих місць надається більше ніж належне значення через обмеження часу та витрат. Проактивна оцінка ризику до випуску IT-програми недоступна. Вплив будь-якої уразливості визначається за допомогою калькулятора CVSS, лише після аналізу того, як відбулася атака.

У запропонованій моделі, яка також була застосована в зразковій проектній організації, потенційний вплив вразливості прогнозується задовго до того, як IT-програму випустять для використання. Завдяки цій інформації про вплив, адекватні витрати та ресурси можуть бути розподілені на усунення вразливостей, тим самим зменшуючи вплив.

Для прогнозування впливу атак у цій пропозиції обраний метод множинної регресії. Багаторазові регресії мають певні припущення, такі як лінійність, відсутність мультиколінеарності, гомосцедастичності та нормальності.

Нижче наведено оперативне визначення розглянутих показників кібербезпеки:

– Y – загальний показник CVSS, залежний фактор. CVSS прогнозується на основі характеристик середовища та системи цільового додатка;

– $X1$ – кількість вразливостей, а саме загальна кількість вразливостей, виявлених статичними та динамічними інструментами виявлення вразливості для цільового додатка. Інструменти, встановлені та запущені проти цільової програми, можуть виявити кілька вразливих місць з допомогою такого алгоритму як тестування на проникнення;

– $X2$ – це середній вхідний мережевий трафік, зафіксований для програми протягом тижня атаки в KBPS.

Вразливості, про які повідомляють інструменти, можна класифікувати загалом за такими категоріями:

– зловживання API;
– вразливість автентифікації;
– вразливість авторизації;
– вразливість доступності;
– вразливість дозволу коду;
– вразливість якості коду;
– вразливість конфігурації;
– криптографічна вразливість;
– вразливість при обробці помилок;
– вразливість загальної логічної помилки;
– вразливість перевірки вхідних даних;
– вразливість реєстрації та аудиту;
– вразливість управління паролями;
– вразливість шляху;
– помилки протоколу;
– вразливість помилок діапазону та типу.

Точки даних в CVSS реєструються для кожної атаки. Результат роботи інструмента записуються для цільового додатка. За вказаний діапазон атаки також фіксується мережевий трафік. Дані з цих трьох джерел складаються у таблиці кожного разу, коли трапляється атака, як показано на рисунку 2.3. Визначення зразків визначені технічним аналітиком. У цій регресійній моделі оцінка CVSS (Y)

прогнозується за допомогою двох змінних – вразливості та мережевого трафіку. Розглянута нульова гіпотеза полягає в тому, що $X1$ і $X2$ не мають впливу на Y. Іншими словами, вразливість та мережевий трафік не впливають на оцінку CVSS.

На рисунку 2.3 висвітлено точки даних для кожного показника під час усіх випадків атаки.

Y	$X1$	$X2$
CVSS Score	Vulnerability	Network Traffic
2.1	20	324
5.3	53	623
1.0	15	235
8.0	85	932
2.9	28	438
3.0	25	498
3.8	38	391
1.0	18	132
1.2	16	177
5.9	63	823
4.3	39	579
2.8	30	455
1.1	14	231
4.2	35	725
5.4	51	740
1.9	21	345
2.0	25	432
4.1	37	467
6.2	58	845
1.1	15	111
2.3	22	191
1.2	16	182
2.8	30	292
6.9	68	952
4.8	55	600

Рисунок 2.3 – Точки даних проекту

Графік нормальної ймовірності, показаний на рисунку 2.4, є приблизно лінійним. З рисунка видно, що припущення про нормальність помилок не порушено.

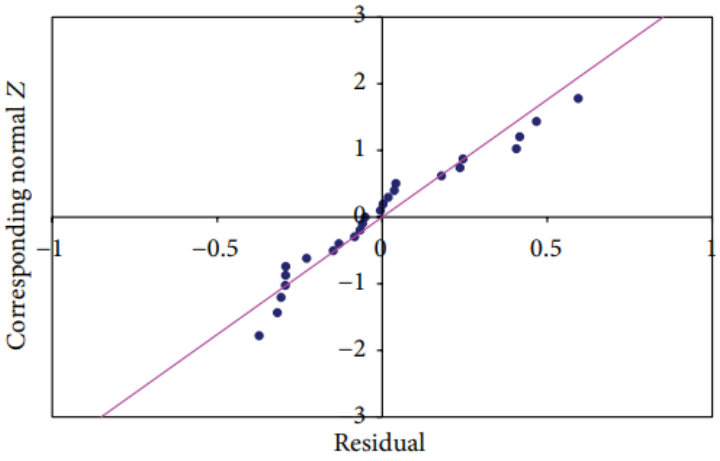


Рисунок 2.4 – Графік нормальної ймовірності

Як показано на рисунку 2.5, вразливість позитивно впливає на оцінку CVSS. Зі збільшенням вразливості рейтинг CVSS збільшується, а отже, вплив на IT-активи є високим. Вплив мережевого трафіку на CVSS позитивний. Це означає, що коли мережевий трафік і вплив вразливостей високий – оцінка CVSS також висока.

Intercept	Vulnerability	Network Traffic
−0.2983	0.07174	0.0025

Рисунок 2.5 – Рівняння регресії

Таким чином, на оцінку CVSS позитивно впливає як вразливість, так і мережевий трафік.

Прогнозований показник CVSS = −0,2893 + 0,07174 ∗ кількість вразливостей в IT-програмі, про які повідомляють інструменти + 0,0025 ∗ запропонований середній вхідний мережевий трафік для програми за тиждень, виміряний в KBPS.

Як показано на рисунку 2.6, b – коефіцієнт, що дає оцінки найменших квадратів, тоді як s (b) – стандартні похибки оцінок найменших квадратів для x змінних, а t – обчислювана t-статистика. Це коефіцієнт, поділений на стандартну помилку.

Значення P дає значення P для перевірки гіпотези. VIF кількісно визначає тяжкість мультиколінеарності в звичайному регресійному аналізі найменших квадратів. VIF для цих даних становить 6,41.

	Intercept	Vulnerability	Network Traffic
b	−0.296	0.0706	0.002
$s(b)$	0.121	0.007	0.0005
t	−2.442	9.359	4.545
P	0.0231	0.0000	0.0002

Рисунок 2.6 – Результати множинної регресії

Як показано на рисунку 2.7, SS – це сума квадратів, обумовлена регресією. Цей показник загальної варіації в Y можна пояснити регресією із змінною X. Df – це ступені свободи. MS є мірою суми квадратів. Середньоквадратична регресія (MSR) та середньоквадратична помилка (MSE) – це дві змінні, що визначають F: F = MSR / MSE. Статистика F використовується, щоб перевірити, чи змінні Y та X пов'язані.

Source	SS	Df	MS	F	P
Regression.	96.22	2	48.15	597	0.000
Error	1.77	22	0.080		
Total	98	24			

Рисунок 2.7 – Дисперсійний аналіз

Для цих даних MSR становить 48, а MSE – 0,08. Статистика F визначає, що значення P дорівнює нулю. Це підтверджує існування лінійного зв'язку між CVSS та двома змінними – мережевим трафіком та вразливостями. $R2$ надає інформацію про придатність моделі. У рівнянні регресії коефіцієнт детермінації $R2$ визначає, наскільки лінія регресії наближає реальні точки даних. Налаштований $R2$ – це модифікована версія, яка регулює кількість предикторів у моделі. Для цих даних $R2$ дорівнює 0,9819 та скоригований $R2$ дорівнює 0,9803.

Дані доводять, що на загальний бал CVSS впливають вразливості в мережі та мережевий трафік. Команда інфраструктури в організації регулярно передає вихідні дані щодо цих змінних команді з якості. Для кожного мережевого процесу, заснованого на типі мережі та розміщених додатках, може розглядатися логічне групування та базові значення організації. Потім технічні аналітики можуть звернутися до цих базових організаційних даних, коли вони починають процес

проектування мережі. Як частина процесу, вони також можуть використовувати ці контрольні значення для визначення верхньої та нижньої меж специфікації. Ці значення будуть доступні для кожного з параметрів підпроцесу. Потім технічний аналітик може визначити та проаналізувати, які вразливості потрібно контролювати, та вибирати порогові значення на основі цього.

На основі вибраних порогових значень проводиться аналіз “що якщо”. Враховуючи різні сценарії, значення вразливості та мережевого трафіку приймаються та надаються як вхідні дані для моделі. Потім прогнозований результат порівнюється з пороговими показниками. Потім прогнозований результат порівнюється з пороговими показниками. Важливо зазначити, що, змінюючи параметри, технічні аналітики повинні розуміти практичні наслідки проекту. Мова йде не лише про математичну модель, а про те, як її можна застосувати на практиці.

2.3 Архітектура API

API – це набір програмного коду, який забезпечує передачу даних між одним програмним продуктом та іншим. Він також містить умови цього обміну даними.

Інтерфейси програмування програм складаються з двох компонентів:

1. Технічна специфікація, що описує варіанти обміну даними між рішеннями, специфікація виконана у формі запиту на обробку та протоколи доставки даних.
2. Інтерфейс програмного забезпечення, написаний за специфікацією, яка його представляє.

Програмне забезпечення, якому потрібно отримати доступ до інформації або функціональних можливостей з іншого програмного забезпечення, викликає свій API, вказуючи вимоги щодо надання даних або функціональності. Інше програмне забезпечення повертає дані або функціональні можливості, що вимагаються попередньою програмою. А інтерфейс, за допомогою якого взаємодіють ці два додатки, визначає API.

Типи API наведено нижче.

Приватні API. Ці інтерфейси прикладного програмного забезпечення призначені для вдосконалення рішень та послуг в організації. Власні розробники або підрядники можуть використовувати ці API для інтеграції IT-систем або додатків компанії, створення нових систем або програм, спрямованих на клієнтів, використовуючи існуючі системи. Навіть якщо програми є загальнодоступними, сам інтерфейс залишається доступним лише для тих, хто працює безпосередньо з видавцем API. Приватна стратегія дозволяє компанії повністю контролювати використання API.

API партнерів. Партнерські API відкрито рекламуються, але діляться з діловими партнерами, які підписали угоду з видавцем. Поширеним варіантом використання партнерських API є інтеграція програмного забезпечення між двома сторонами. Компанія, яка надає партнерам доступ до даних або можливостей, отримує вигоди від додаткових потоків доходу. У той же час він може контролювати, як використовуються відкриті цифрові активи, переконатися, що сторонні рішення, що використовують їх API, забезпечують гідну взаємодію з користувачами, та підтримувати фірмовий стиль у своїх програмах.

Відкриті API. Ці API також відомі як зовнішні, і вони доступні для будь-яких сторонніх розробників. Публічна програма API дозволяє підвищити обізнаність про бренд та отримати додаткове джерело доходу при правильному виконанні.

Рисунок 2.1 зображує потік даних через архітектуру API. Шлюз API (пристрій або послуга) буде діяти як центральний вузол, де різні клієнти можуть отримувати інформацію з різних служб. Клієнти можуть відрізнятися за типом, наприклад: мобільні пристрої, сервери, дослідники або веб-програми. Послуги надаватимуться розробниками OSU або сторонніми програмами. Ця архітектура дозволить нам роз’єднати клієнтів та послуги, а також зосередити наші зусилля щодо стійкості на шлюзі API та послугах.

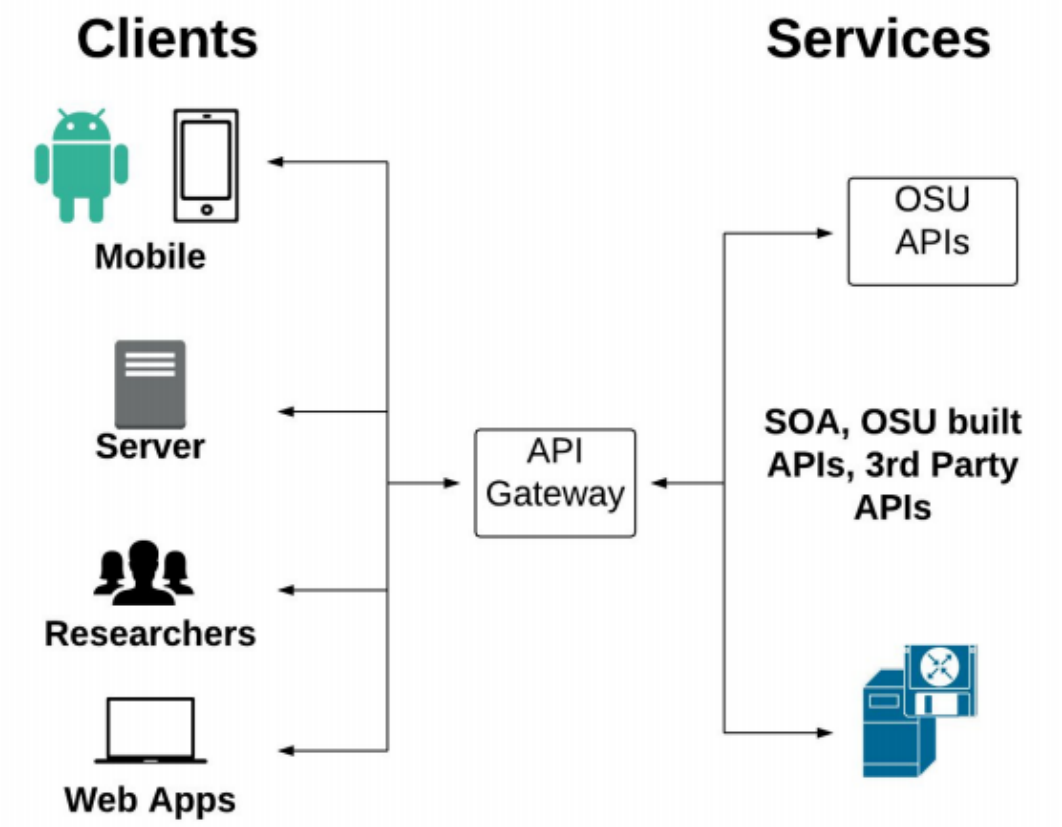


Рисунок 2.8 – Потік даних через API

Кожен API містить і реалізується за допомогою викликів функцій – мовних операторів, які вимагають програмне забезпечення для виконання певних дій та послуг. Виклики функцій – це фрази, що складаються з дієслів та іменників, наприклад:

– початок або закінчення сеансу;

– відновлення або отримання об’єктів із сервера.

Роль API значно більша, якщо ми дивимось на неї не лише з точки зору розробки програмного забезпечення, але й з точки зору ділової співпраці. Більше 60 відсотків учасників звіту “Поточний стан інтеграції API-2018” погодились, що інтеграція API є критично важливою для їхньої бізнес-стратегії. Дослідження також передбачає, що понад 50 відсотків усіх підприємств будуть співпрацювати через API.

У зв’язку з цим, двома основними завданнями для тих, хто приймає рішення та розробників, є вибір API, який відповідає конкретним бізнес-потребам компанії, та розуміння того, як ефективно його використовувати.

У даному проекті будуть використовуватись інтеграції наступних сервісів:

- IntelligenceX;
- BuildWith.

Розділ 3 РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ OSINT

3.1 Реалізація та моделювання проектних рішень

Реалізація автоматизованої системи буде відбуватись за допомогою бота в месенджері Telegram. Боти – це сторонні програми, які працюють всередині Telegram. Користувачі можуть взаємодіяти з ботами, надсилаючи їм повідомлення, команди та вбудовані запити. Керування ботами відбувається з допомогою використання запитів HTTPS до API для ботів.

Багато галузей перекладають обслуговування клієнтів на системи чат-ботів. Це пов’язано з величезним падінням вартості такого рішення, а також завдяки надійності та постійній доступності. Чат-боти забезпечують певний рівень підтримки користувачів без істотних додаткових витрат.

Незалежно від складності чат-ботів, структура програмного забезпечення, як правило, однакова. Чат-бот ускладнюється після додавання додаткових компонентів для більш природного спілкування. На рисунку 3.1 зображено схему, яка описує архітектуру чат-ботів.

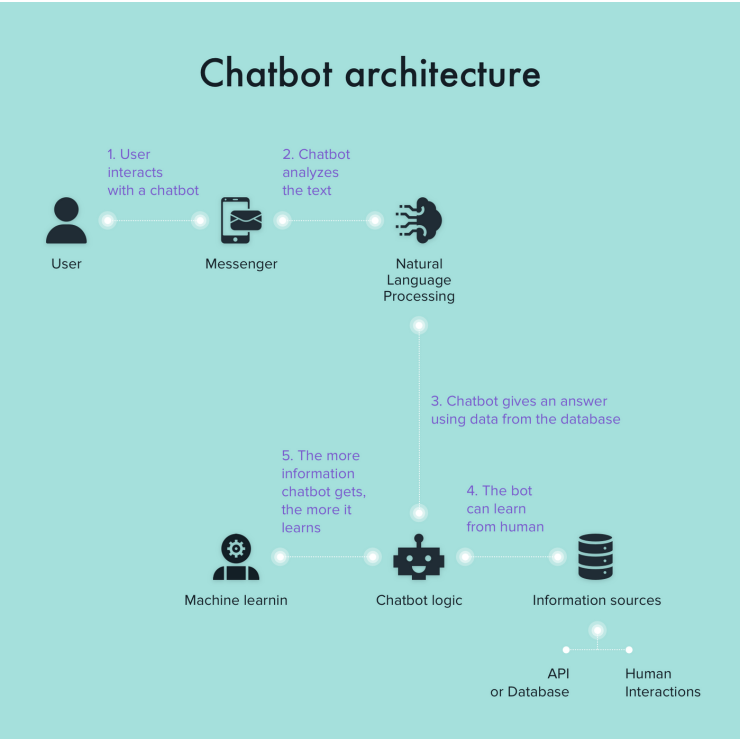


Рисунок 3.1 – Архітектура чат-ботів

Telegram – одна з найпопулярніших платформ обміну миттєвими повідомленнями сьогодні, оскільки вона дозволяє зберігати повідомлення в хмарі, а не лише на пристрої, і вона може виділитись хорошою підтримкою багатьох платформ, оскільки Telegram можна використовувати на Android, iOS, Windows і майже будь-якій іншій платформі, яка може підтримувати веб-версію.

Телеграм використовує власний протокол шифрування MTProto. MTProto API (він же Telegram API) – це API, через який додаток телеграм зв'язується з сервером. Telegram API повністю відкритий, так що будь-який розробник може написати свій клієнт месенджера.

Створення нового бота відбувається за допомогою використання такого ж бота.

Знайти його можна ввівши в пошуку назву BotFather, як зображено на рисунку 3.1.

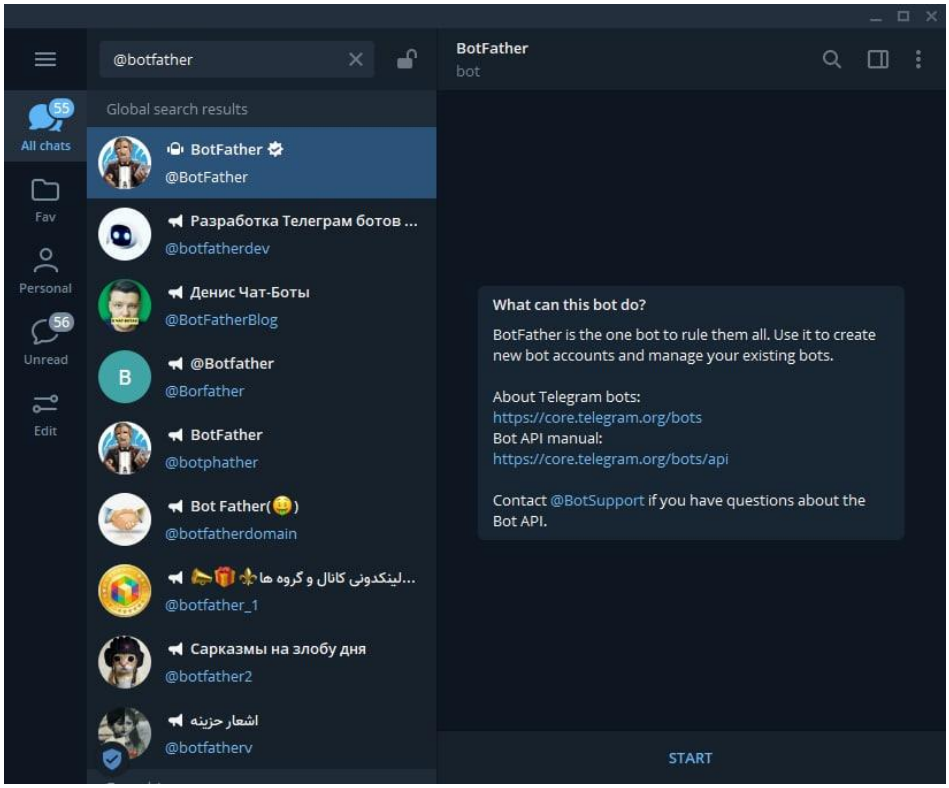


Рисунок 3.2 – Пошук BotFather

За допомогою команди /newbot потрібно створити нового бота, як зображено на рисунку 3.2. BotFather запитає назву та ім'я користувача, а потім створить маркер авторизації для нового бота.

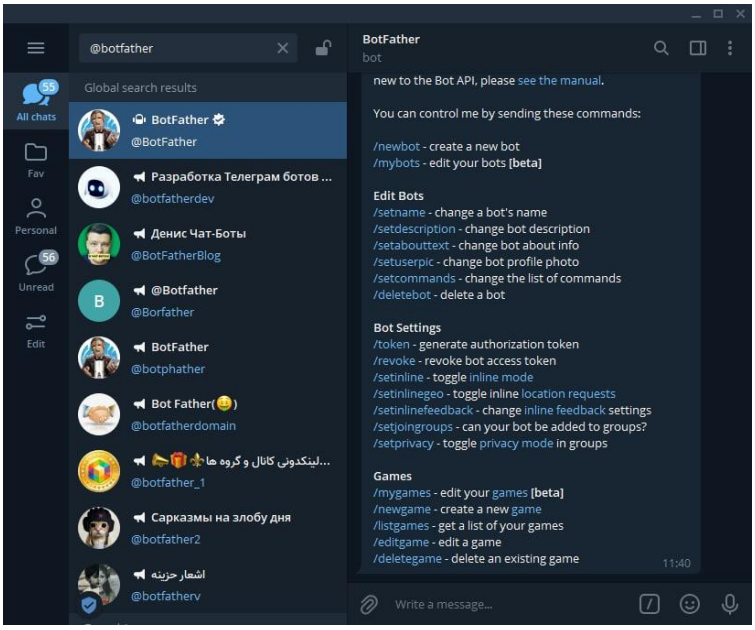


Рисунок 3.3 – Створення нового бота

Ім’я бота відображається в контактних даних та інших місцях. Ім’я користувача – це коротка назва, яка використовуватиметься у згадках та посиланнях t.me. Імена користувачів мають довжину 5-32 символи та не враховують регістр, але можуть містити лише латинські символи, цифри та підкреслення. Ім’я користувача бота має закінчуватися на “бот”, наприклад “tetris_bot” або “TetrisBot”. На рисунку 3.3 показано процес найменнування боту, у даному випадку називатися він буде OSINT Automation.

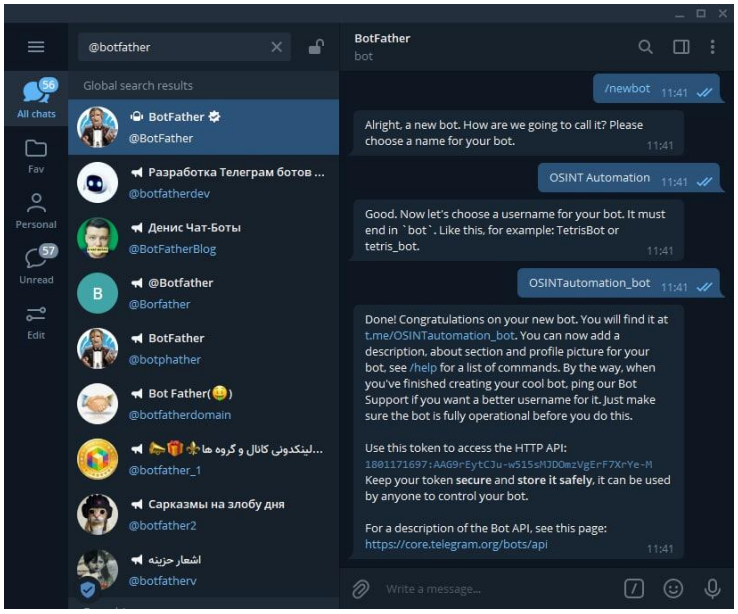


Рисунок 3.4 – Процес найменування бота

Після створення боту, BotFather надав токен. Токен виглядає приблизно так: 110201543: AAHdqTcvCH1vGWJxfSeofSAs0K5PALDsaw. Саме за допомогою токена можна керувати ботом.

Оформлення бота налаштовується в BotFather: меню / mybots → Edit Bot. Там можна змінити:

– ім’я бота;
– опис (Description) – це текст, який користувачі будуть бачити на початку діалогу з ботом під заголовком “Що може робити цей бот?”;
– інформація (About) – це текст, який буде видно в профілі бота;
– аватар – аватар ботів, на відміну від аватарів користувачів і чатів, не можуть бути анімованими. Тільки картинки.
– команди – тут маються на увазі підказки команд в боті. Детальніше про команди нижче.
– inline placeholder – інлайн режим

Коли користувач вперше відкриває бота, він бачить кнопку “Запустити” або “Почати” (залежить від платформи користувача), англійською – “Start”. Натискаючи на цю кнопку, він відправляє команду / start.

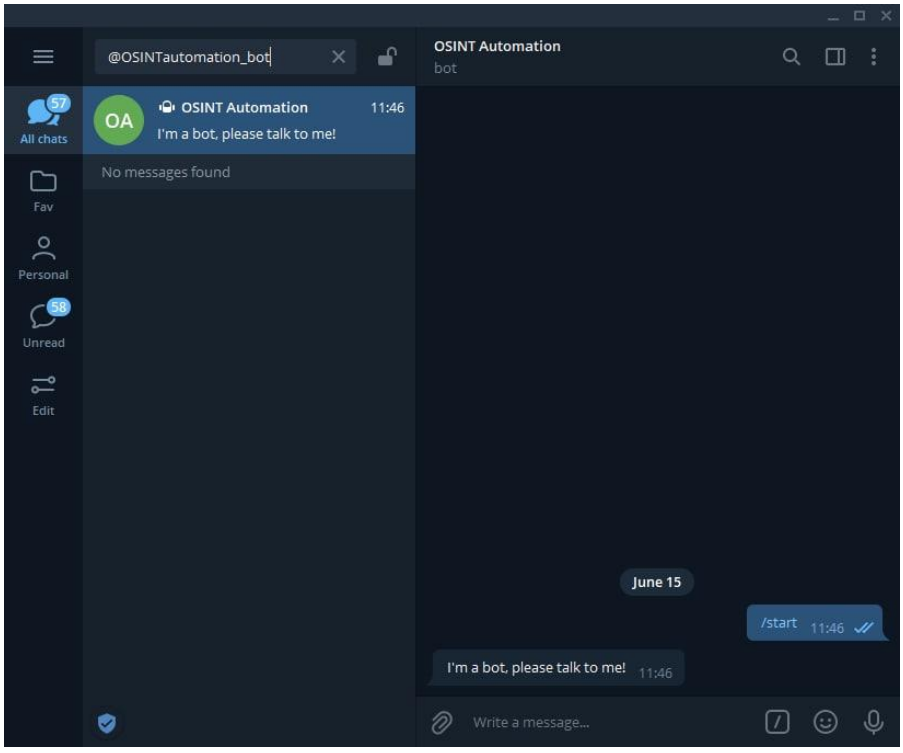


Рисунок 3.5 – Запуск першої команди /start

Є два основних способи працювати з Telegram в Python: за допомогою надсилання HTTPS запитів і за допомогою Webhook. У цьому проекті три ланки: комп'ютер з Python, сервер Telegram та telegram-клієнт.

На комп'ютері працює інтерпретатор Python, а всередині інтерпретатора працює програма на Python. Вона відповідає за весь контент: в неї закладені всі шаблони тексту, вся логіка і поведінка.

Всередині програми на Python працює бібліотека, яка відповідає за спілкування з сервером Telegram. У бібліотеку інтегровано секретний ключ, щоб сервер Telegram розумів, що програма пов'язана з певним конкретним ботом.

Коли клієнт з Telegram запитує у бота інформацію, запит приходить на сервер. Запит обробляється програмою на Python, відповідь йде на сервер Telegram, а сервер віддає відповідь клієнту.

Працювати бот буде тільки тоді, коли ввімкнений комп'ютер і на ньому запущена програма на Python. Якщо комп'ютер вимкнеться, пропаде інтернет або буде вимкнуто інтерпретатор, то бот працювати перестане: запити будуть приходити, але ніхто на них не відповість.

Реалізовувати логіку бота ми будемо за допомогою бібліотеки python-telegram-bot. Дана бібліотека використовується для швидкого і зручного створення ботів, так як вона містить в собі велику кількість реалізованих методів доступу до інформації Telegram.

Підмодуль telegram.ext побудований поверх чистої реалізації API, він надає простий у використанні інтерфейс. Підмодуль складається з декількох класів, але двома найважливішими є telegram.ext.Updater та telegram.ext.Dispatcher.

Клас Updater постійно отримує нові оновлення з Telegram та передає їх до класу Dispatcher. Потім додаються обробники різних команд та повідомлень у Dispatcher, які сортуватимуть оновлення, отримані Updater, і вже за зареєстрованими обробниками, інформацію передаватимуть у визначену функцію зворотного виклику.

Кожен обробник є екземпляром будь-якого підкласу класу telegram.ext.Handler. Бібліотека пропонує класи обробників майже для всіх випадків використання.

При створенні екземпляру класу Updater використовується токен доступу який раніше був отриманий при створенні боту. Приклад створення екземплярів класів Updater та Dispatcher, а також доданих обробників команд та повідомлень кінцевого користувача. Лістинг 3.1, що наведений нижче, є частиною додатку В.

Лістинг 3.1 – Ініціалізація боту

```
from telegram.ext import CommandHandler, Filters, MessageHandler, Updater
from config import BOT_API_KEY
from telegram_bot_handlers import TelegramBotHandlers

updater = Updater(token=BOT_API_KEY, use_context=True)
dispatcher = updater.dispatcher

start_handler = CommandHandler('start', TelegramBotHandlers.wellcome_message)
help_handler = CommandHandler('help', TelegramBotHandlers.wellcome_message)
unknown_message_handler = MessageHandler(Filters.text & (~Filters.command), TelegramBo
tHandlers.unknown_message)
scan_email_handler = CommandHandler('scan_email', TelegramBotHandlers.scan_email)
scan_domain_handler = CommandHandler('scan_domain', TelegramBotHadlers.scan_domain)

dispatcher.add_handler(start_handler)
dispatcher.add_handler(help_handler)
dispatcher.add_handler(unknown_message_handler)

dispatcher.add_handler(scan_email_handler)
dispatcher.add_handler(scan_domain_handler)

updater.start_polling()
```

Протягом виконання роботи було створено клас TelegramBotHadlers, який містить в собі усі функції обробників подій для виконання наступних дій:

– wellcome_message – виводить привітальне повідомлення користувачу, яке містить інформацію про інші доступні функції та їх опис;

– unknown_message – виводить інформаційне повідомлення про введення некоректних даних та підказку про команду /help яка також інформує про усі доступні в боті функції;

– scan_email – команда для сканування електронної адреси;

– scan_domain – команда для сканування домену.

В функції для сканування електронних адрес створено клас IntelxScanner(), який звертається до API сервісу IntelX для отримання релевантних даних про причетність електронної адреси до витоків даних і при успішних результатах пошуку надає інформацію про загальну кількість витоків даних та при наявності – облікові дані, включаючи пароль. Функція сканування електронної адреси продемонстрована в лістингу 3.2, що знаходиться в додатку А.

Лістинг 3.2 – Функція сканування електронної адреси

```
def search_email(self, email: str):
        """Method used to search email in intelx database and return results

        Args:
            mail (str): String formated email [example@domain.com]
        """
        result_str = f"No information found about {email}!"

        record_count, search = self.__search_email(email)

        if record_count == 0:
            return result_str

        result_str = f"Information found about {email}:\n\n"

        stats_str = self.__parse_email_stats(search=search)

        result_str = result_str + stats_str

        file_name = self.__download_first_file(search)

        email_data = self.__parse_downloaded_file(file_id=file_name, email=email)

        result_str = result_str + "\n\n" + email_data

        os.remove(f"downloads/intelx/{file_name}")

        return result_str
```

В свою чергу функція сканування доменів використовує BuildWith сканер. Він надає дані про використовувані доменом сервіси та інструменти, включаючи їх версії та опис, додаткову інформацію таку як рейтинг сайту та посилання на соціальні мережі, якщо такі було знайдено.

3.2 Тестування функціональних можливостей

Тестуваня програмного продукту з точки зору класифікації за програмними цілями ділиться на два класи:

- функціональне тестування;
- нефункціональне тестування.

Під функціональним тестуванням розуміється перевірка відповідності програмного продукту функціональним вимогам, зазначеним у технічному завданні на створення це продукту.

Нефункціональне тестування оцінює такі якості програмного продукту, як ергономіку або продуктивність.

Впродовж навантажувального тестування було виявлено, що існує ліміт запитів до сервера Telegram. У Bots FAQ на сайті Telegram названі такі:

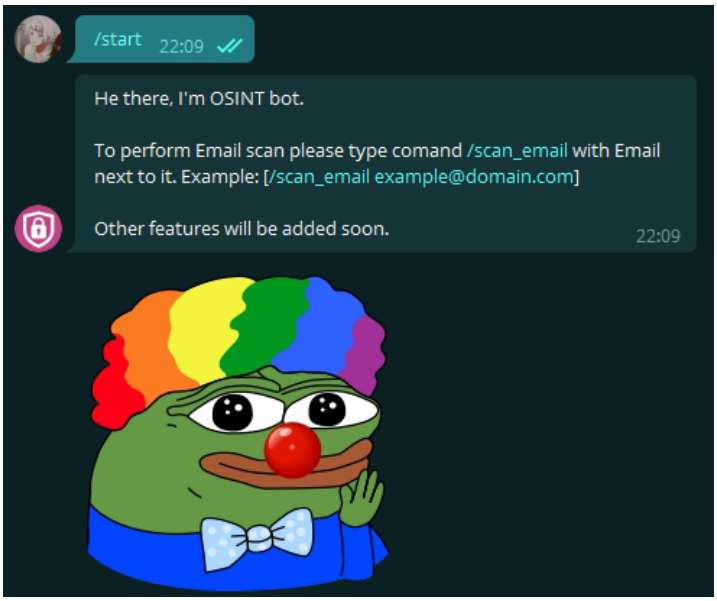
- не більше одного повідомлення в секунду в один чат;
- не більше 30 повідомлень в секунду взагалі;
- не більше 20 повідомлень за хвилину в одну групу.

Ці ліміти не суворі, а приблизні. Ліміти можуть бути збільшені для великих ботів через підтримку.

На рисунку 3.6 зображено вивід програми з інформацією про введені користувачами дані, та часом їх введення. Рисунок 3.7 демонструє вивід повідомлення з боку користувача.

```
[DEBUG  ][2021-06-17 22:11:03,744] telegram.bot: decorator: Entering: get_updates
[DEBUG  ][2021-06-17 22:11:03,747] telegram.ext.dispatcher: start: Processing Update: {'update_id': 222
654767, 'message': {'new_chat_members': [], 'caption_entities': [], 'group_chat_created': False, 'photo
': [], 'message_id': 93, 'date': 1623957063, 'supergroup_chat_created': False, 'chat': {'username': 'la
pka_kot1ka', 'first_name': 'lapka_kotika', 'type': 'private', 'id': 1280284278}, 'delete_chat_photo': F
alse, 'entities': [], 'sticker': {'emoji': '✋', 'thumb': {'file_unique_id': 'AQADgEmSpi4AA1M5AAI', 'wi
d
th': 128, 'height': 128, 'file_size': 7066, 'file_id': 'AAMCAgADGQEAA11gy55HbnPvjkLq2am_D-1xDDYoMgACowo
AAv-euEh_Tsl9m5evE4BJkqYuAAMBAAdtAANTOQACHwQ'}, 'is_animated': True, 'file_unique_id': 'AgADowoAAv-euEg
', 'width': 512, 'height': 512, 'set_name': 'honka_animated', 'file_size': 8896, 'file_id': 'CAACAgIAAx
kBAANdYMueR25z745C6tmpvw_tcQw2KDIAAqMKAAL_nrhIf07JfZuXrxMfBA'}, 'channel_chat_created': False, 'new_cha
t_photo': [], 'from': {'username': 'lapka_kot1ka', 'is_bot': False, 'first_name': 'lapka_kotika', 'id':
 1280284278, 'language_code': 'uk'}}}
```


Рисунок 3.6 – Вивід повідомлення користувача lapka_kot1ka



3.7 – Введення користувачем привітального повідомлення

Для демонстрації успішного пошуку факту витоку даних щодо електронної адреси було спеціально вибрано зразок, який підходить під задані умови – landry.todd@gmail.com. Це персональна електронна адреса віце президента компанії JMA Wireless, яка є світовим лідером у галузі мобільних бездротових підключень, включаючи зовнішні та внутрішні розподілені антенні системи. На рисунках 3.8 та 3.9 продемонстровано результат виводу інформації.

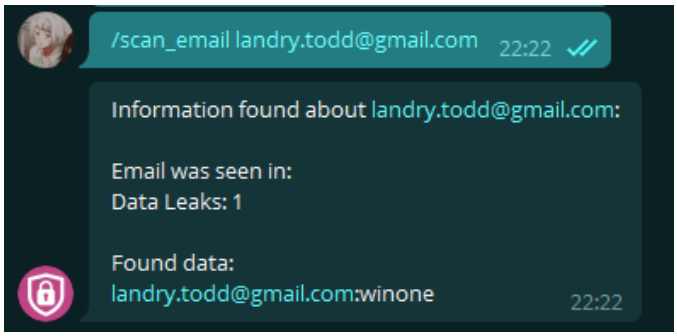


Рисунок 3.8 – Перевірка електронної адреси на витік даних з позитивним результатом

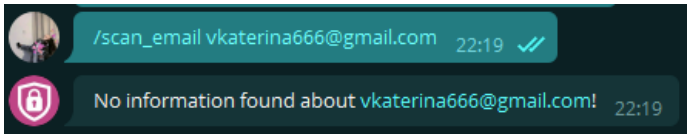


Рисунок 3.9 – Перевірка електронної адреси на витік даних з негативним результатом

В боті реалізовано пошук інформації за доменом, і за допомогою цієї інтеграції можна отримати розгорнуті дані про використовувані інструменти, рейтинг домену

та сторінки у соціальних мережах, приклад виводу зображено на рисунку 3.10.

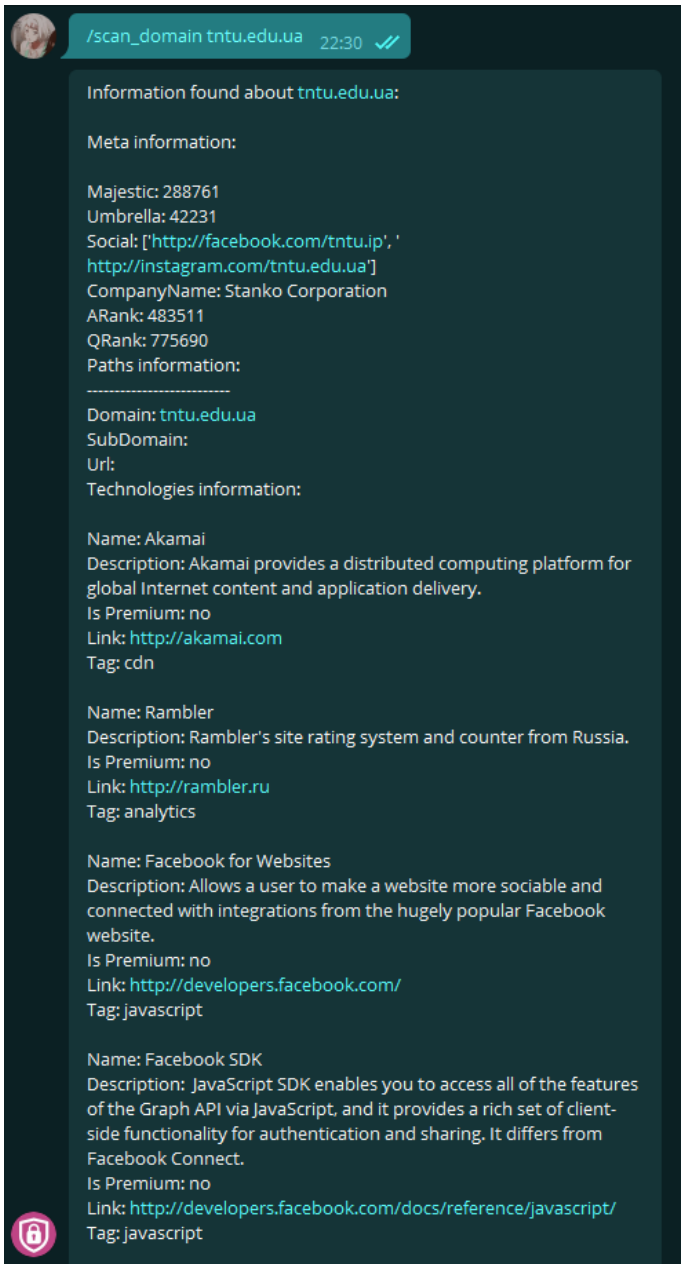


Рисунок 3.10 – Виведення інформації про домен tntu.edu.ua

3.3 Конфігурація автоматизованої системи

Першим кроком є отримання API ключів потрібних для інтеграцій сервісів. API BuiltWith Domain надає XML та JSON доступ до технологічної інформації веб-сайту, що включає всю технічну інформацію, яку можна знайти в детальних пошуках на веб-сайті builtwith.com та додаткові метадані, якщо такі є.

Для того щоб отримати API ключ потрібно ввійти в особистий кабінет або зареєструватися на сайті builtwith.com.

В особистому кабінеті IntelligenceX можна отримати свій API ключ, а також інформацію про кількість запитів доступних для використання в безкоштовній версії, як показано на рисунках 3.11 та 3.12.

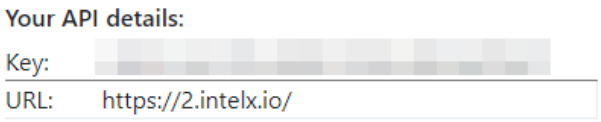
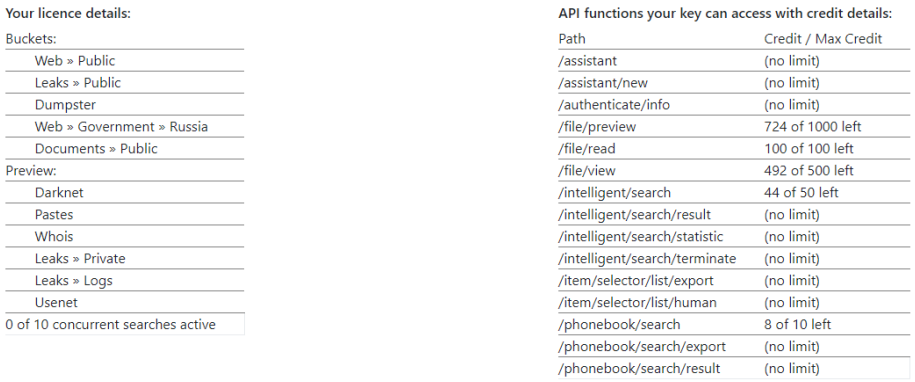


Рисунок 3.11 – Інформація про API



Your licence details:

Buckets:
Web » Public
Leaks » Public
Dumpster
Web » Government » Russia
Documents » Public
Preview:
Darknet
Pastes
Whois
Leaks » Private
Leaks » Logs
Usenet
0 of 10 concurrent searches active

API functions your key can access with credit details:

Path	Credit / Max Credit
/assistant	(no limit)
/assistant/new	(no limit)
/authenticate/info	(no limit)
/file/preview	724 of 1000 left
/file/read	100 of 100 left
/file/view	492 of 500 left
/intelligent/search	44 of 50 left
/intelligent/search/result	(no limit)
/intelligent/search/statistic	(no limit)
/intelligent/search/terminate	(no limit)
/item/selector/list/export	(no limit)
/item/selector/list/human	(no limit)
/phonebook/search	8 of 10 left
/phonebook/search/export	(no limit)
/phonebook/search/result	(no limit)

Рисунок 3.12 – Кількість доступних запитів

Усі використовувані API ключі доступні в конфігураційному файлі і продемонстровані на рисунку 3.13.

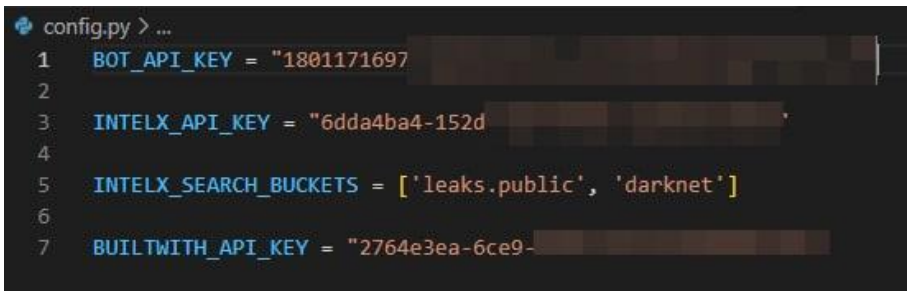


Рисунок 3.13 – API ключі

Процес конфігурації продовжується зі встановлення бібліотек, встановити бібліотеку потрібно за допомогою цієї команди:

```
pip3 install python-telegram-bot
```

Дана бібліотека є обов’язковою, так як вона не входить у стандартний набір пакетів Python. Ця бібліотека забезпечує чистий інтерфейс Python для API Telegram Bot і сумісна з версіями Python 3.6.2 і вище. PTB також може працювати на PyPy, хоча PyPy офіційно не підтримується.

В доповнення до чистої реалізації API, ця бібліотека має ряд класів високого рівня, що роблять розробку ботів простою та зрозумілою. Ці класи містяться в підмодулі telegram.ext.

Розділ 4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Значення адаптації в трудовому процесі

Праця людини безпосередньо пов’язана із виробничим середовищем. Працівник може нормально здійснювати трудову діяльність лише тоді, коли умови зовнішнього середовища відповідають оптимальним. Якщо вони змінюються, стають несприятливими, то на протидію їм організм людини включає спеціальний механізм, який зберігає постійність внутрішнього середовища, або змінює його в межах допустимого. Такий механізм називається адаптацією.

Адаптація є важливим засобом попередження травмування, виникнення нещасних випадків у трудовому процесі і відіграє значну роль в охороні праці.

Адаптація (від лат. adapto – пристосування) – це динамічний процес пристосування організму та його органів до мінливих умов зовнішнього середовища. Адаптація в трудовій діяльності поділяється на фізіологічну, психічну, соціальну та професійну.

Адаптацію у широкому змісті трактують як зміни, що супроводжують на рівні психічної регуляції так і процес активного пристосування індивіда до нових умов життєдіяльності. У науковій літературі визначаються біологічний, фізіологічний, операційний, інформаційний, комунікативний, особистісний і соціально-психологічний аспекти професійної адаптації. Таке розмаїття аспектів адаптації пояснюється тим, що людина розглядається як сплав властивостей індивіда й особистості. У такому випадку професійна адаптація являє собою єдність адаптації індивіда до фізичних умов фахового середовища (перший аспект, психофізіологічний), адаптації до професійних завдань, виконуваним операціям, професійної інформації (другий аспект, професійний) та адаптації особистості до соціальних компонентів професійного середовища (третій, соціально-психологічний аспект).

Трудова адаптація – це соціальний процес освоєння особистістю нової трудової ситуації, у якій особистість і трудове середовище роблять активний вплив один на одного. Потрапляючи на роботу, людина активно включається в систему

професійних і соціально-психологічних відносин конкретної трудової організації, засвоює нові для неї соціальні ролі, цінності, норми, узгоджує свою індивідуальну позицію з цілями і задачами організації (трудового колективу), тим самим, підкоряючи свою поведінку службовим розпорядженням даного підприємства або установи [20].

Адаптованість людини до конкретного трудового середовища виявляється в її реальній поведінці, у конкретних показниках трудової діяльності:

- ефективності праці;
- засвоєнні соціальної інформації і її практичної реалізації;
- росту усіх видів активності;
- задоволеності різними сторонами трудової діяльності.

Трудова адаптація може бути первинною – при первісному входженні працівника у виробниче середовище і вторинною – при зміні робочого місця без зміни і зі зміною професії або при істотних змінах середовища [21].

У процесі адаптації працівник проходить наступні стадії:

- стадія ознайомлення, на якій працівник одержує інформацію про нову ситуацію в цілому, про критерії оцінки різних дій, про еталони, норми поведінки;
- стадія пристосування, на цьому етапі працівник переорієнтується, визнаючи головні елементи нової системи цінностей, але поки продовжує зберігати багато своїх установок;
- стадія асиміляції, коли здійснюється повне пристосування до середовища, ідентифікація з новою групою;
- ідентифікація, коли особисті цілі працівника ототожнюються з цілями трудової організації, підприємства або фірми.

Швидкість адаптації залежить від багатьох факторів. Нормальний термін адаптації для різних категорій працівників складає від 1 року до 3 років. Невміння ввійти в трудову організацію (колектив), адаптуватися в ній викликає явище виробничої і соціальної дезорганізації.

Успішне (або утруднене) проходження виробничої адаптації молодого працівника залежить від того, наскільки сприятливі (або несприятливі) умови

склалися для задоволення його адаптивної потреби. З огляду на структуру останньої до таких умов належать певний рівень взаємної інформації між індивідом і виробництвом, взаємних контактів; порівняння життєвих цілей індивіда з завданнями даного підприємства, а також наявність на виробництві умов для успішної трудової діяльності індивіда. Отже, наявність і повнота прояву сукупності умов, необхідних для проходження виробничої адаптації, будуть визначати як її ефективність, динамізм, так і межі самого процесу.

4.1.1 Фактори трудової адаптації

Фактори трудової адаптації – це умови, які впливають на протікання, терміни, темні і результати цього процесу. Серед них можна виділити як об'єктивні, так і суб'єктивні.

Об'єктивні (у трудовій організації – це фактори, пов'язані з виробничим процесом) – фактори, що у меншій ступені залежать від працівника (рівень організації праці, механізації й автоматизації виробничих процесів, санітарно-гігієнічні умови праці, розмір колективу, розташування підприємства та галузева спеціалізація) [22].

До суб'єктивних (особистісних) факторів відносяться:

- соціально-демографічні характеристики працівника (стать, вік, освіта, кваліфікація, стаж роботи та соціальний стан);
- соціологічні (ступінь професійного інтересу, матеріальної і моральної зацікавленості в ефективності і якості праці).

Розрізняють наступні етапи і форми адаптації персоналу:

- випробувальний термін тривалістю 3-6 місяців;
- адаптація молодих фахівців тривалістю до 3-х років;
- програма введення в посаду керівного працівника тривалістю до 1-го року;
- наставництво і консультування; розвиток людських ресурсів.

Успішність адаптації залежить від цілого ряду умов, головними із яких є:

- якісний рівень роботи з питань профорієнтації потенційних працівників;
- об’єктивність ділової оцінки персоналу;
- особливості організації праці, які б реалізували мотиваційні настанови працівника;
- гнучкість системи навчання персоналу на підприємстві;
- особливості соціально-психологічного клімату, що склався в колективі;
- особисті якості працівника, який проходить адаптацію, пов’язані з його віком, сімейним становищем, характером.

4.2 Підбирання оптимальних параметрів мікроклімату на робочих місцях

Санітарні норми поширюються на умови мікроклімату в межах робочої зони виробничих приміщень підприємств, закладів, установ тощо, незалежно від їх форми власності та підпорядкування.

Мікрокліматичні умови виробничих приміщень характеризуються такими показниками:

- температура повітря;
- відносна вологість повітря;
- швидкість руху повітря;
- інтенсивність теплового (інфрачервоного) опромінення;
- температура поверхні.

За ступенем впливу на тепловий стан людини мікрокліматичної умови поділяють на оптимальні та допустимі.

Для робочої зони виробничих приміщень встановлюються оптимальні та допустимі мікрокліматичні умови з урахуванням важкості виконуваної роботи та періоду року. При одночасному виконанні в робочій зоні робіт різної категорії важкості рівні показників мікроклімату повинні встановлюватись з урахуванням найбільш чисельної групи працівників.

Показники температури повітря в робочій зоні по висоті та по горизонталі, а також протягом робочої зміни не повинні виходити за межі нормованих величин оптимальної температури для даної категорії робіт, вказаної на рисунку 4.1

Температура внутрішніх поверхонь робочої зони (стіни, підлога, стеля), технологічного обладнання, зовнішніх поверхонь технологічного устаткування, огороджуючих конструкцій не повинна виходити більш ніж на 2 градуси за Цельсієм за межі оптимальних величин температури повітря для даної категорії робіт, вказаних на рисунку 4.1

Період року	Категорія робіт	Температура повітря	Відносна вологість	Швидкість руху, м/сек.
Холодний період року	Легка Іа	22 - 24	60 - 40	0,1
	Легка Іб	21 - 23	60 - 40	0,1
	Середньої важкості IIа	19 - 21	60 - 40	0,2
	Середньої важкості IIб	17 - 19	60 - 40	0,2
	Важка III	16 - 18	60 - 40	0,3
Теплий період року	Легка Іа	23 - 25	60 - 40	0,1
	Легка Іб	22 - 24	60 - 40	0,2
	Середньої важкості IIа	21 - 23	60 - 40	0,3
	Середньої важкості IIб	20 - 22	60 - 40	0,3
	Важка III	18 - 20	60 - 40	0,4

Рисунок 4.1 – Оптимальні величини температури, відносної вологості та швидкості руху повітря в робочій зоні різного типу приміщень [19]

При виконанні робіт операторського типу, пов’язаних з нервово-емоційним напруженням в кабінетах, пультах і постах керування технологічними процесами, в залах обчислювальної техніки та інших приміщеннях повинні дотримуватися оптимальні умови мікроклімату (температура повітря 22-24 градуси за Цельсієм, відносна вологість 60-40%, швидкість руху повітря не більш 0,1 м/сек.).

4.3 Показники ефективності та заходи щодо покращенню умов та охорони праці

Економічне значення охорони праці визначається ефективністю заходів з покращення умов і підвищення безпеки праці та є економічним виразом соціальної значущості охорони праці. Тобто, економічне значення охорони праці оцінюється за результатами, отриманими при зміні соціальних показників шляхом впровадження заходів з покращення умов праці:

- підвищення продуктивності праці;
- зниження непродуктивних витрат часу і праці;
- збільшення фонду робочого часу;
- зниження витрат, пов’язаних з плинністю кадрів через умови праці, тощо.

На підприємствах спостерігається висока плинність кадрів серед працівників, робота яких пов’язана з важкою фізичною працею, несприятливими санітарно-гігієнічними умовами, монотонністю виробничого процесу. Із загальної кількості працівників, які звільняються за власним бажанням, від 10 до 25% складають особи, яких не влаштовують несприятливі умови праці [23].

Стимулювання заходів щодо охорони праці здійснюється згідно з розділом IV «Стимулювання охорони праці» Закону України «Про охорону праці». Так, ст. 25 «Економічне стимулювання охорони праці» визначає, що до працівників підприємств можуть застосовуватися будь-які заохочення за активну участь та ініціативу у здійсненні заходів щодо підвищення безпеки й поліпшення умов праці. Види заохочень визначаються колективним договором (угодою, трудовим договором). Порядок пільгового оподаткування коштів, спрямованих на заходи щодо охорони праці, накреслений чинним законодавством про оподаткування. Економічне стимулювання націлено насамперед на посилення діяльності та заінтересованості підприємств у поліпшенні умов праці на робочих місцях, а також підвищення економічної відповідальності власників (адміністрації) підприємств за шкоду, заподіяну несприятливими умовами праці.

Оскільки поліпшення виробничого середовища понад установлені законами норми є справою дорогою, то для досягнення високих критеріїв існує явна потреба в

економічному стимулюванні. Тому економічне стимулювання пропонується не як заміна, а як доповнення до норм законодавства про охорону праці. Однак воно може застосовуватися і на підприємствах, де стан охорони праці не відповідає вимогам законодавства.

Атестація є одним із прийомів створення і підтримки належного рівня охорони праці і є комплексною оцінкою кожного робочого місця на його відповідність передовому науково-технічному рівню виробництва, гігієнічним нормам праці, психофізіологічним параметрам працюючого.

Основною метою атестації є регулювання відношень між роботодавцем (або уповноваженого ним органом) і працюючими в сфері реалізації їхніх прав на здорові і безпечні умови праці, пільгове пенсійне забезпечення, пільги і компенсації за роботу у небезпечних і шкідливих умовах.

Належний рівень охорони праці забезпечується шляхом:

– доведення параметрів виробничого середовища до норматив них значень (технічні і технологічні рішення);

– захисту працівників від впливу небезпечних і шкідливих виробничих факторів.

Поточна оцінка стану охорони праці у виробничих цехах та дільницях може бути визначена узагальненим коефіцієнтом рівня охорони праці $К_{сп}^{ц}$, що є середньоарифметичним суми трьох коефіцієнтів:

$$К_{сп}^{ц} = \frac{К_д + К_б + К_{впр}}{3} \leq 1, \ [24]$$

де $К_д = С_д \ / \ С$ – коефіцієнт рівня дотримання правил охорони праці ($С_д$ – кількість працюючих, що дотримуються правил охорони праці; $С$ – загальна кількість працюючих);

$К_б = n_{вб}/n$ – коефіцієнт технічної безпеки обладнання ($n_{вб}$ – кількість одиниць обладнання, що відповідає вимогам техніки безпеки і санітарним вимогам; n – загальна кількість обладнання);

$К_{впр} = m_{ср}/m$ – коефіцієнт виконання планових робіт з охорони праці ($m_{ср}$ – кількість фактично виконаних запланованих робіт з охорони праці; m – загальна кількість запланованих робіт за певний відрізок часу).

Згідно з Єдиною державною системою показників обліку умов і безпеки праці визначення цього коефіцієнту виконується комісією в кінці аналізованого періоду часу. При оцінці ефективності заходів щодо охорони праці користуються сукупністю чотирьох груп показників:

- показники зміни стану умов праці;
- соціальні показники;
- економічні показники;
- соціально-економічні показники.

При цьому враховуються наступні фактори виробничого середовища:

- поліпшення санітарно-гігієнічних показників;
- зниження вмісту шкідливих речовин у повітрі;
- поліпшення умов мікроклімату;
- зниження, рівня шуму і вібрації;
- поліпшення освітлення;
- поліпшення психофізіологічних показників;
- зниження підвищених фізичних навантажень;
- зниження нервово-психічних навантажень;
- поліпшення естетичних показників;
- раціональна організація робочих місць;
- благоустрій приміщень і територій;
- естетичність оформлення інтер'єрів.

Зміна стану умов праці за факторами оцінюється різницею їхньої абсолютної величин до і після впровадження заходів чи різницею досягнутих або прогнозованих результатів, а також зіставленням відносних показників, що характеризують ступінь, відповідності тих чи інших факторів граничне припустимим концентраціям (ГПК), граничне припустимим рівням (ГПР) чи заданим значенням.

ВИСНОВОК

Автоматизована система пошуку корпоративної та особистої інформації забезпечує можливість швидкого доступу до інформації конфіденційного характеру внаслідок витоку даних. З інсталяцією подібної системи кожен користувач отримує можливість:

- визначити факт компрометації електронної пошти;
- отримати облікові дані користувача, який потраждав від витоку даних;
- проаналізувати інформацію про використовувані доменом інструменти;
- отримати оцінку безпеки домена.

Під час використання різноманітних ресурсів з пошуку конфіденційної інформації для інтеграції було обрано IntelligenceX та BuildWith, так як ці сервіси надають можливість безоплатно скористатись їх послугами, щоправда в обмеженій кількості спроб. Більшість інструментів з цінною інформацією потребує фінансового вкладення, що становить проблему під час дослідження предметної області.

Ринок чат-ботів розширюється, а месенджер Telegram дає хорошу можливість для розвитку подібних проектів. Кількість активних користувачів Telegram перевищила 500 мільйонів [25], що говорить про популярність додатку серед користувачів та можливість залучати велику аудиторію для своїх продуктів.

В ході роботи було досягнуто основні завдання, такі як:

- отримано досвід використання OSINT методик і способів пошуку корпоративної та особистої інформації;
- проаналізовано стан ринку автоматизованих систем OSINT;
- описано явище витоку корпоративної та особистої інформації;
- розроблено інструмент для автоматизації пошуку інформації використовуючи інтеграції готових засобів;
- проведено тестування готового продукту та оцінку функціональних можливостей.

Для досягнення поставлених цілей було використано методику інтеграцій сервісів з допомогою API ключів в Telegram бот, а уся програмна реалізація виконана на мові програмування Python.

Рішення автоматизації збору інформації за допомогою Telegram боту принесла велику кількість переваг:

- зручність, висока швидкість одержуваної інформації;
- простота інтерфейсу, все в одному додатку Telegram;
- відсутність вимоги проходження капчі;
- відсутність необхідності використання Tor і DarkWeb;
- відсутність реєстрації на спеціалізованих сайтах.

У майбутньому технології OSINT стануть вдосконаленими і зможуть збирати більше даних про приватних осіб та корпорації. Можливості для бізнесу збільшуватимуться, а ті, хто їх втрачає, втрачають цінну інформацію. Очікуючи майбутнього, потрібно задатись питанням, які відкриті джерела можна використовувати для збору даних. Відповідь – майже будь-що. З новими технологіями, що надходять в Інтернет щодня, не можна сказати, де ми побачимо OSINT у 2022 році та пізніше. Очікується, що світовий ринок засобів розвідки з відкритим кодом зростатиме із річними темпами у 17%, досягнувши майже 20 мільярдів доларів США до 2026 року. Зростання доступної інформації у поєднанні зі збільшенням загрози безпеці буде стимулювати попит на OSINT. Тому дуже важливо працювати над вдосконаленням технік та впровадженням нових методів пошуку інформації з метою надати ефективну оцінку клієнтам, які хочуть знати реальний стан захищеності їх персональних даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The National Academies Press. Advances from the Behavioral and Social Sciences Intelligence Analysis for Tomorrow. 2011. 5 с.
2. J. Francined, P. Alonso, S. Sánchez. Open-Source Intelligence Resources: A Visual Perspective Analysis. Department of Computer Science, Spain. 2020. 12 с.
3. Statista. Number of daily active Facebook users worldwide as of 1st quarter 2021. URL: https://www.statista.com/statistics/346167/facebook-global-dau/ (дата звернення: 01.05.2021).
4. Statista. Number of monthly active Instagram users from January 2013 to June. URL: https://www.statista.com/statistics/253577/number-of-monthly-active-instagram-users/ (дата звернення: 04.05.2021).
5. Statista. Forecast of the number of LinkedIn users in the World from 2017 to 2025. URL: https://www.statista.com/forecasts/1147197/linkedin-users-in-the-world (дата звернення: 04.05.2021).
6. B. Meyer. COMB: largest breach of all time leaked online with 3.2 billion records. Cybernews. URL: https://www.statista.com/statistics/273550/data-breaches-recorded-in-the-united-states-by-number-of-breaches-and-records-exposed/ (дата звернення: 05.05.2021).
7. E. Bekker 533 Million Facebook Users' Personal Data Leaked Online. Identityforce. URL: https://www.identityforce.com/blog/533-million-facebook-users-data-leaked-online (дата звернення: 05.05.2021).
8. E. Bekker 500 MILLION LINKEDIN ACCOUNTS LEAKED ON THE DARK WEB. URL: https://www.sontiq.com/resources/500m-linkedin-data-leak/ (дата звернення: 05.05.2021).
9. Statista. Annual number of data breaches and exposed records in the United States from 2005 to 2020. URL: https://www.statista.com/statistics/273550/data-breaches-

recorded-in-the-united-states-by-number-of-breaches-and-records-exposed/ (дата звернення: 11.05.2021).

10. Tdwi. U.S. Data Breaches Dropped by 30 Percent in 2020. URL: https://tdwi.org/articles/2021/04/02/u.s.-data-breaches-dropped-by-30-percent-in-2020.aspx (дата звернення: 10.05.2021).
11. J. Pastor-Calindo, P. Nespoli. The not yet exploited goldmine of OSINT: Opportunities, open challenges and future trends. Murcia, Spain, 2016. 2 с.
12. Mediasonar. OSINT Techniques for Organization POIs. URL: https://mediasonar.com/2020/08/26/osint-techniques-for-security-poi-organizations/ (дата звернення: 15.05.2021).
13. M. Bazzell. Open Source Intelligence Techniques: Resources for Searching and Analyzing Online Information. CreateSpace Independent Publishing Platform. 2018. 54-58 с.
14. Maltego. Tutorial – Part 1: Information gathering URL: https://cdn.ttgtmedia.com/rms/pdf/SearchSecurity.in_Maltego_tutorial_1.pdf (дата звернення: 10.05.2021).
15. C. Kubecka Hack the World with OSINT 2019. URL: https://www.researchgate.net/publication/332875854_Hack_the_World_with_OSINT (дата звернення: 13.05.2021)
16. B. Jansen. The Seventeen Theoretical Constructs of Information Searching and Information Retrieval. Journal of the American Society for Information Sciences and Technology. 2010. 10 с.
17. D. Kuropka. Models for the representation of natural language documents. Ontology-based information filtering and retrieval with relational databases. Advances in Information Systems and Management Science. 2004. 12 с.
18. M. Manning. Introduction to Information Retrieval. Cambridge University Press. 2008.
19. Міністерство охорони здоров’я України. Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. 1999. URL: https://zakon.rada.gov.ua/

rada/show/va042282-99#Text (дата звернення: 15.05.2021).

20. Нікітченко О. Ю., Нестеренко С. В. Соціальна й економічна ефективність рекомендацій з поліпшення умов праці. Харківський національний університет міського господарства імені О. М. Бекетова. Харків. 2017. 5-6 с.
21. Москальова В. М. Охорона праці. Інтерактивний комплекс навчальнометодичного забезпечення. Рівне: НУВГП, 2009. 26 с.
22. Лукашевич Н. П. Виробнича адаптація як елемент трудовий кар'єри працівника. Київ: Знання, 2004. 31 с.
23. Шпак Л. Л. Становлення робітника: адаптація і виховання робітників кадрів. Профіздат, 1997. 12-14 с.
24. Комісарів Т.А. Управління людськими ресурсами. Київ: Справа, 2002. 42 с.
25. P. Durov. Durov`s Channel. Telegram. 2021. URL: https://t.me/durov/147 (дата звернення: 25.05.2021).
26. Rzheuskyi, A., Matsuik, H., Veretennikova, N., Vaskiv, R. Selective Dissemination of Information – Technology of Information Support of Scientific Research. Lviv Polytechnic National University. Lviv. 2019. 237 с.

ДОДАТОК А

Лістинг файлу intelx_scanner.py

```
import json
import os
from typing import Tuple

from config import INTELX_API_KEY, INTELX_SEARCH_BUCKETS
from intelxapi import intelx

class IntelxScanner():
    def __init__(self):
        self.intelx_instance = intelx(INTELX_API_KEY)

    def search_email(self, email: str):
        """Method used to search email in intelx database and return results

        Args:
            mail (str): String formated email [example@domain.com]
        """
        result_str = f"No information found about {email}!"

        record_count, search = self.__search_email(email)

        if record_count == 0:
            return result_str

        result_str = f"Information found about {email}:\n\n"

        stats_str = self.__parse_email_stats(search=search)

        result_str = result_str + stats_str

        file_name = self.__download_first_file(search)

        email_data = self.__parse_downloaded_file(file_id=file_name, email=email)

        result_str = result_str + "\n\n" + email_data

        os.remove(f"downloads/intelx/{file_name}")

        return result_str

    def __search_email(self, email: str) -> Tuple[int, dict]:
        """Methods performs search for email in Intelx database
        Args:
            email (str): Search email query

        Returns:
            Tuple[int, dict]: Amount of objects found and original search responce
        """
        search       =       self.intelx_instance.search(term=email,       maxresults=1000,
buckets=INTELX_SEARCH_BUCKETS)
        record_count = len(search['records'])
        return record_count, search

    def __download_first_file(self, search: dict) -> str:
        """Method that uses Intelx API to download first file in search responce

        Args:
```

```
        search (dict): Original search responce

    Returns:
        str: Filename
    """
    first_doc = search['records'][0]
    self.intelx_instance.FILE_READ(
                        id=first_doc["systemid"],
                        type=0,
                        bucket=first_doc["bucket"],
                        filename=f"downloads/intelx/{first_doc['systemid']}"
        )

    return first_doc["systemid"]

def __parse_email_stats(self, search: dict) -> str:
    """Parses to string stats data about search
    Args:
        search (dict): Original search responce
    Returns:
        str: String formated answer
    """
    result = "Email was seen in:"
    stats = json.loads(self.intelx_instance.stats(search))
    for type, amount in stats.items():
        result = result + "\n"
        if "pastes" in type:
            result = result + f"Pastes: {amount}"
            continue
        if "government" in type:
            result = result + f"Government web pages: {amount}"
            continue
        if "web" in type:
            result = result + f"Web pages: {amount}"
            continue
        if "dumpster" in type:
            result = result + f"Dumpster: {amount}"
            continue
        if "darknet" in type:
            result = result + f"Darknet: {amount}"
            continue
        if "leaks" in type:
            result = result + f"Data Leaks: {amount}"
            continue
        if "whois" in type:
            result = result + f"Inside Whois data: {amount}"
            continue
    return result

def __parse_downloaded_file(self, file_id: str, email: str) -> str:
    """Method parses file in search of email and returns formated string
    Args:
        file_id (str): File name received from intelx
        email (str): Email that need to be searched for
    Returns:
        str: Formated string with found or missing data
    """
    with open(f"downloads/intelx/{file_id}", "r", encoding='utf-8') as f:
        lines = f.readlines()
        for line in lines:
            if email in line:
                return "Found data:\n" + line
    return "No data found"
```

ДОДАТОК Б

Лістинг файлу telegram_bot_handlers.py

```
import re
import time
from typing import Union
from telegram.constants import MAX_MESSAGE_LENGTH

from scanners.builtwith_scanner import BuiltWithScanner
from scanners.intelx_scanner import IntelxScanner
from utils import get_logger

logger = get_logger(__name__)

class TelegramBotHandlers():
    intelx_scanner = IntelxScanner()
    builtwith_scanner = BuiltWithScanner()

    @staticmethod
    def wellcome_message(update, context):
        wellcome_message = "He there, I'm OSINT bot.\n\n" +\
                           "To perform Email scan please type comand /scan_email " +\
                           "with Email next to it. Example: [/scan_email
example@domain.com]\n\n" +\
                           "Other features will be added soon."
        context.bot.send_message(chat_id=update.effective_chat.id,
text=wellcome_message)
        return None
    @staticmethod
    def unknown_message(update, context):
        unknown_message = "Sorry, I didn't understood you. Please check /help command
to see what I can do. ❤"
        context.bot.send_message(chat_id=update.effective_chat.id,
text=unknown_message)
        return None
    @staticmethod
    def scan_email(update, context):
        chat_id = update.effective_chat.id

        mail =
TelegramBotHandlers.parse_email_from_message(message=update.message.text)
        if not mail:
            error_message = "Unable to parse valid Email in message.\n" +\
                            "Please type email after comand. Example: /scan_email
example@domain.com"
            context.bot.send_message(chat_id=chat_id, text=error_message)
            return None

        info = TelegramBotHandlers.intelx_scanner.search_email(email=mail)
        context.bot.send_message(chat_id=chat_id, text=info)
        return None

    @staticmethod
    def parse_email_from_message(message: str) -> Union[str, None]:
        """Method used to parse email from message

        Args:
            message (str): Message received from user
```

```
            Returns:
                str: If mail parsed successfully
                None: If no message found
            """
            message = message.replace("/scan_email ", "")
            mail_re = r'[^@]+@[^@]+\.[^@]+'
            result = re.findall(mail_re, message)
            if result:
                return result.pop()
            return None

        @staticmethod
        def scan_domain(update, context):
            chat_id = update.effective_chat.id

            domain =
TelegramBotHandlers.parse_domain_from_message(message=update.message.text)
            if not domain:
                error_message = "Unable to parse valid domain in message.\n" +\
                                "Please type domain after comand. Example: /scan_domain
domain.com"
                context.bot.send_message(chat_id=chat_id, text=error_message)
                return None

            info = TelegramBotHandlers.builtwith_scanner.search_domain(domain=domain)
            if len(info) <= MAX_MESSAGE_LENGTH:
                context.bot.send_message(chat_id=chat_id, text=info)

                return None
            else:
                parts = []
                while len(info) > 0:
                    if len(info) > MAX_MESSAGE_LENGTH:
                        part = info[:MAX_MESSAGE_LENGTH]
                        first_lnbr = part.rfind('\n')
                        if first_lnbr != -1:
                            parts.append(part[:first_lnbr])
                            info = info[(first_lnbr+1):]
                        else:
                            parts.append(part)
                            info = info[MAX_MESSAGE_LENGTH:]
                    else:
                        parts.append(info)
                        break
                for part in parts:
                    context.bot.send_message(chat_id=chat_id, text=part)
                    time.sleep(1)
            return None
        @staticmethod
        def parse_domain_from_message(message: str) -> Union[str, None]:
            """Method used to parse domain from message
            Args:
                message (str): Message received from user
            Returns:
                str: If domain parsed successfully
                None: If no message found
            """
            message = message.replace("/scan_domain ", "")
            domain_re = r'^(?:https?:\/\/)?(?:[^@\/\n]+@)?(?:www\.)?([^:\/?\n]+)'
            result = re.findall(domain_re, message)
            if result:
                return result.pop()
```

```
    return None
```

ДОДАТОК В

Лістинг файлу main.py

```
from telegram.ext import CommandHandler, Filters, MessageHandler, Updater

from config import BOT_API_KEY
from telegram_bot_handlers import TelegramBotHandlers

updater = Updater(token=BOT_API_KEY, use_context=True)
dispatcher = updater.dispatcher

start_handler = CommandHandler('start', TelegramBotHandlers.wellcome_message)
help_handler = CommandHandler('help', TelegramBotHandlers.wellcome_message)
unknown_message_handler = MessageHandler(Filters.text & (~Filters.command),
TelegramBotHandlers.unknown_message)
scan_email_handler = CommandHandler('scan_email', TelegramBotHandlers.scan_email)
scan_domain_handler = CommandHandler('scan_domain', TelegramBotHandlers.scan_domain)

dispatcher.add_handler(start_handler)
dispatcher.add_handler(help_handler)
dispatcher.add_handler(unknown_message_handler)

dispatcher.add_handler(scan_email_handler)
dispatcher.add_handler(scan_domain_handler)

updater.start_polling()
```

ДОДАТОК Г

Лістинг файлу buildwith_scanner.py

```
import json
import requests

from config import BUILTWITH_API_KEY


class BuiltWithScanner():
    def __init__(self) -> None:
        self.__api_endpoint = "https://api.builtwith.com/v19/api.json?"

    def search_domain(self, domain: str):
        """Method used to search domain info in builtwith database and return results

        Args:
            domain (str): String formated domain [domain.com]
        """
        result_str = f"No information found about {domain}!"

        search_responce = self.__search_domain(domain)

        if not search_responce:
            return result_str

        result_str = f"Information found about {domain}:\n\n"

        parsed_data = str()
        parsed_data = self.__parse_data(search_responce=search_responce)

        result_str = result_str + parsed_data

        return result_str

    def __search_domain(self, domain: str) -> dict:
        """Methods performs search for domain in Intelx database

        Args:
            email (str): Search email query

        Returns:
            Tuple[int, dict]: Amount of objects found and original search responce
        """
        query = f"{self.__api_endpoint}KEY={BUILTWITH_API_KEY}&LOOKUP={domain}"
        responce = requests.get(url=query)

        return responce.json()

    def __parse_data(self, search_responce: dict) -> str:
        """Method parses search_responce and returns formated string

        Args:
            search_responce (dict): Responce from BuiltWith

        Returns:
            str: Formated string with found data
        """
        result = str()
```

```
            result                                                                                  +=
self.__parse_meta_info(search_responce_meta=search_responce["Results"][0]["Meta"])

            result                                                                                  +=
self.__parse_paths_info(search_responce_paths=search_responce["Results"][0]["Result"]["Paths
"])

            if search_responce["Errors"]:
                result += "Error information:\n\n"
                for id, error in enumerate(search_responce["Errors"]):
                    result   +=   f"\n\n   Error   #{id}"   +   json.dumps(error,   indent=2,
sort_keys=True)

            return result

        def __parse_meta_info(self, search_responce_meta: dict) -> str:
            """Method parses metadata info from search responce

            Args:
                search_responce_meta (dict): Meta object from search responce

            Returns:
                str: String formated meta data
            """
            ignore_list = ["Vertical"]

            result = str()
            result = "Meta information:\n\n"
            for key, value in search_responce_meta.items():
                if key in ignore_list:
                    continue
                if not value:
                    continue

                result = result + f"{key}: {value}\n"
            return result

        def __parse_paths_info(self, search_responce_paths: dict) -> str:
            """Method parses paths info from search responce
            Args:
                search_responce_paths (dict): Paths object from search responce
            Returns:
                str: String formated paths data
            """
            result = str()
            result = "Paths information:\n"

            for path in search_responce_paths:
                result += "-------------------------\n"
                result += f"Domain: {path.get('Domain', '')}\n"
                result += f"SubDomain: {path.get('SubDomain', '')}\n"
                result += f"Url: {path.get('Url', '')}\n"
                result += "Technologies information:\n"
                for tech in path.get('Technologies', ''):
                    result += "\n"
                    result += f"Name: {tech.get('Name', '')}\n"
                    result += f"Description: {tech.get('Description', '')}\n"
                    result += f"Is Premium: {tech.get('IsPremium', '')}\n"
                    result += f"Link: {tech.get('Link', '')}\n"
                    result += f"Tag: {tech.get('Tag', '')}\n"
                result += "-------------------------\n"
                break
        return result
```