**Міністерство освіти і науки України**
**Луцький національний технічний університет**
**Факультет комп'ютерних та інформаційних технологій**
**Кафедра комп'ютерних наук**

КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «МАГІСТР»

# ДОСЛІДЖЕННЯ ТА АНАЛІЗ ІНСТРУМЕНТІВ ПЕРЕВІРКИ ДОМЕНІВ ТА IP НА ВРАЗЛИВІСТЬ

# RESEARCH AND ANALYSIS OF TOOLS FOR DOMAIN AND IP VULNERABILITY ASSESSMENT

спеціальність 122 «Комп'ютерні науки»

освітня програма «Комп'ютерні науки»

Виконала: здобувач вищої освіти
групи КНм-21
Омельчук Анастасія Андріївна
_____
(підпис)
Керівник: к.т.н., доцент
Лук'янчук Юрій Анатолійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«___» _____ 2025 р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Валерій Олександрович
_____
(підпис)

Луцьк – 2025 року

**ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ**

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *комп'ютерних наук*
Ступінь вищої освіти: *магістр*
Галузь знань: *12 «Інформаційні технології»*
Спеціальність: *122 Комп'ютерні науки*
Освітня програма: *«Комп'ютерні науки»*

ЗАТВЕРДЖУЮ

Завідувач кафедри
_____Валерій ЛІЩИНА
«14» травня 2025 р.

**ЗАВДАННЯ**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА**
**ДРУГОГО (МАГІСТЕРСЬКОГО) РІВНЯ ВИЩОЇ ОСВІТИ**

Омельчук Анастасія Андріївна
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Дослідження та аналіз інструментів перевірки доменів та ір на вразливість»*
Керівник роботи *к.т.н., доцент Лук'янчук Юрій Анатолійович*
затверджені наказом закладу вищої освіти від «14» травня 2025 року. № 255/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи «05» грудня 2025 р.
3. Вихідні дані до роботи: *статті, дослідження вітчизняних та закордонних авторів в даній області, сучасні методи і засоби розробки, технічна документація технологій розробки.*
4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *Аналіз проблематики та постановка завдань дослідження, теоретичне дослідження та практична реалізація систем моніторингу безпеки доменів і IP-адрес, експерементальне дослідження результативності систем моніторингу безпеки доменів і IP-адрес.*
5. Перелік графічного матеріалу: *1. Цикл відкритих джерел інформації: структурований підхід до збору та аналізу даних 2. Порівняння функціоналу Telegram-ботів 3. Кількість використань Telegram Bot для створення ботів з метою фішингу з грудня 2024 року 4. Головна сторінка сервісу VirusTotal. 5. Головна сторінка сервісу AbuseIPDB. 6. Головна сторінка сервісу Shodan. 7. Обґрунтування вибору технологій для реалізації платформи. 8. Діаграма використання платформи для аналізу IP та доменів. 9. Діаграма класів платформи для моніторингу безпеки доменів та IP-адрес. 10. Отримання інформації з сервісу AbuseDB. 11. Перевірка репутації домену через VirusTotal. 12. Розрахунок аналітичного балу на основі даних AbuseDB. 13. Порівняльні показники методів перевірки. 14. Гістограма часу відповіді. 15. Діаграма порівняння метрик. 16. Результати вимірювання часу відповіді API-сервісів. 17. Гістограма середнього часу відповіді API. 18. Порівняння часу перевірки. 19. Порівняння методів перевірки за повнотою даних. 20. Порівняння повноти отриманих даних різними методами. 21. Оцінювання зручності та стабільності платформи.*

6. Консультанти розділів роботи

| Розділ | Прізвище, ініціали та посада консультанта | Підпис | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| *Аналіз проблематики та постановка завдань дослідження* | *Лук'янчук Ю. А.* | | |
| *Теоретичне дослідження та практична реалізація систем моніторингу безпеки доменів і IP-адрес* | *Лук'янчук Ю. А.* | | |
| *Експеременталне дослідження результативності систем моніторингу безпеки доменів і IP-адрес* | *Лук'янчук Ю. А.* | | |
| *Показник запозичень тексту* | _____ % | | |
| *Інструментальна перевірка* | *Кошелюк В. А.* | | |
| *Нормоконтроль* | *Сачук В. О.* | | |
| *Гарант ОПП* | *Ліщина В. О.* | | |

7. Дата видачі завдання «14» травня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

| № з/п | Назва етапів кваліфікаційної роботи магістра | Строк виконання етапів роботи | Примітка |
|---|---|---|---|
| *1* | *Обґрунтування теми дослідження* | *до 14.05.2025* | |
| *2* | *Провести огляд літературних джерел по темі кваліфікаційної роботи* | *до 02.08.2025* | |
| *3* | *Провести аналіз загальної проблеми і вибір напрямків дослідження* | *до 15.09.2025* | |
| *4* | *Розробити функціональну схему роботи програмного продукту* | *до 29.09.2025* | |
| *5* | *Описати засоби розробки об'єкта проектування* | *до 12.10.2025* | |
| *6* | *Практична реалізація об'єкта проектування* | *до 20.10.2025* | |
| *7* | *Розробити методику для проведення експерименту* | *до 28.10.2025* | |
| *8* | *Провести аналіз результатів експерименту* | *до 06.11.2025* | |
| *9* | *Формування списку використаних джерел* | *до 16.11.2025* | |
| *10* | *Оформлення ілюстративного матеріалу* | *до 18.11.2025* | |
| *11* | *Інструментальна перевірка на академічний плагіат* | *до 03.12.2025* | |
| *12* | *Здача чистового варіанту кваліфікаційної роботи на кафедру* | *до 05.12.2025* | |

Здобувач вищої освіти _____ Анастасія ОМЕЛЬЧУК

Керівник роботи _____ Юрій ЛУК'ЯНЧУК

# АНОТАЦІЯ

Омельчук А. А. Дослідження та аналіз інструментів перевірки доменів та ір на вразливість. Рукопис.

Кваліфікаційна робота магістра ОП «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки». Луцький національний технічний університет. Луцьк, 2025.

Кваліфікаційна робота магістра складається з вступу, 3 розділів, висновків і пропозицій, списку використаних джерел, додатків (згідно структури кваліфікаційної роботи, затвердженої кафедрою).

У магістерській кваліфікаційній роботі проведено дослідження та аналіз сучасних інструментів перевірки доменних імен та IP-адрес на наявність кіберзагроз і вразливостей. Актуальність теми зумовлена зростанням кількості кібератак, поширенням шкідливих вебресурсів і необхідністю забезпечення високого рівня інформаційної безпеки в умовах активного розвитку цифрового простору.

У роботі розглянуто принципи функціонування платформ VirusTotal, AbuseIPDB, Shodan та інших сервісів відкритої аналітики, які надають змогу виявляти шкідливу активність, аналізувати репутацію мережевих об'єктів і здійснювати моніторинг у реальному часі. На основі проведеного аналізу розроблено платформу, що реалізує автоматизовану перевірку доменів та IP-адрес із використанням зазначених сервісів.

Система забезпечує збір, обробку та структурування даних, формування звітів у форматі PDF, а також інформування користувача про результати перевірки безпосередньо у середовищі Telegram. У роботі розроблено архітектуру програмного забезпечення, визначено його функціональні та нефункціональні вимоги, описано процес інтеграції із зовнішніми API та формування звітності.

Ключові слова: Telegram-бот, платформа, кібербезпека, IP-адреса, домен, OSINT, VirusTotal, AbuseIPDB, Shodan, PDF-звіт, автоматизація.

# ABSTRACT

Anastasiia Omelchuk. Research and analysis of tools for domain and IP vulnerability assessment. Manuscript.

Qualification work for master's degree in «Computer Science» specialty 122 «Computer Science». Lutsk National Technical University. Lutsk, 2025.

The master's qualification work consists of an introduction, 3 chapters, conclusions and proposals, a list of sources used, appendices (according to the structure of the qualification work approved by the department).

The master's thesis presents a comprehensive study and analysis of modern tools for assessing domain names and IP addresses for potential cyber threats and vulnerabilities. The relevance of the research is determined by the increasing number of cyberattacks, the spread of malicious online resources, and the growing need to ensure information security in an expanding digital environment.

The paper examines the operational principles of VirusTotal, AbuseIPDB, Shodan, and other open-source intelligence platforms that enable the detection of malicious activity, reputation analysis of network entities, and real-time monitoring. Based on this analysis, a platform was developed to perform automated security checks of domains and IP addresses using these external APIs.

The implemented system collects, processes, and structures analytical data, generates PDF reports, and delivers the results directly to the user within the Telegram interface. The thesis presents the system's software architecture, defines its functional and non-functional requirements, and describes the integration and reporting processes.

Keywords: Telegram-bot, platform, cybersecurity, IP address, domain, OSINT, VirusTotal, AbuseIPDB, Shodan, PDF report, automation.

# ЗМІСТ

ВСТУП ..... 7
РОЗДІЛ 1 АНАЛІЗ ПРОБЛЕМАТИКИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ ..... 10
1.1 Огляд і аналіз предметної області проблеми (задачі), результатів існуючих теоретичних та експериментальних досліджень ..... 10
1.2 Огляд і аналіз методів та засобів розробки систем моніторингу безпеки доменів і IP-адрес для вирішення проблеми дослідження ..... 14
1.3 Постановка завдання на кваліфікаційну роботу магістра ..... 16
РОЗДІЛ 2 ТЕОРЕТИЧНЕ ДОСЛІДЖЕННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ МОНІТОРИНГУ БЕЗПЕКИ ДОМЕНІВ І IP-АДРЕС ..... 19
2.1 Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання ..... 19
2.2 Практична реалізація об’єкта проектування ..... 30
РОЗДІЛ 3 ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РЕЗУЛЬТАТИВНОСТІ СИСТЕМИ МОНІТОРИНГУ БЕЗПЕКИ ДОМЕНІВ І IP-АДРЕС ..... 42
3.1 Методика проведення дослідження ..... 42
3.2 Обробка та аналіз отриманих результатів ..... 48
ВИСНОВКИ ..... 56
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ ..... 59
ДОДАТКИ ..... 61

# ВСТУП

У часи всесвітньої цифровізації та розвитку інформаційних технологій питання забезпечення кібербезпеки набуває особливої актуальності. Зростаюча кількість атак на інформаційні системи, витік конфіденційних даних та поширення зловмисних ресурсів в мережі Інтернет вимагають впровадження новітніх засобів для відстеження та аналізу мережевої активності. Особливої уваги потребують доменні імена та IP-адреси – це ключові елементи мережевої інфраструктури, через які відбувається більшість комунікацій у кіберпросторі.

Традиційні методи оцінки безпечності мережевих ресурсів часто вимагають спеціалізованих технічних знань, ручного опрацювання великих обсягів інформації або застосування громіздких програмних комплексів. З огляду на це, нагальною є потреба у створенні автоматизованих, зручних та загальнодоступних інструментів, які б дозволяли користувачам, незалежно від їхньої технічної обізнаності, перевіряти домени й IP-адреси щодо вразливостей у режимі реального часу.

Зважаючи на посилення ролі месенджерів як універсальних платформ для комунікації, значної уваги заслуговує екосистема Telegram, що надає широкі можливості для автоматизації процесів через ботів. Використання Telegram-бота як інструментарію для кібермоніторингу дає змогу поєднати простоту взаємодії з користувачем, швидкість обробки запитів та доступ до потужних прикладних програмних інтерфейсів (API) для аналізу кіберзагроз (таких як VirusTotal, Shodan, AbuseIPDB тощо).

З огляду на це, розробка платформи, призначеної для перевірки доменних імен та IP-адрес на потенційну шкідливість, є актуальним науково-прикладним завданням, націленим на підвищення рівня інформаційної безпеки та автоматизацію процесів контролю кіберпростору.

На даний момент існує чимало вебплатформ та сервісів для перевірки доменів та IP-адрес, серед яких найвідомішими є VirusTotal, AbuseIPDB, Shodan, Censys, IPinfo, WhoisXMLAPI. Вони надають можливість отримати дані

про репутаційну історію, інциденти зловживань, географічні дані, конфігурацію серверів та відкриті порти. Однак переважна більшість цих інструментів не забезпечує зручного механізму інтеграції у мобільне чи чатове середовище, що ускладнює їхнє використання для некваліфікованих користувачів.

Світові напрямки розвитку у цій сфері орієнтовані на автоматизацію процесів аналізу, застосування методів машинного навчання для прогнозування загроз, а також інтеграцію з комунікаційними платформами (наприклад, Slack, Telegram, Discord). Зокрема, в міжнародних дослідженнях активно розглядається концепція «ChatOps», яка поєднує засоби обміну повідомленнями з операційним моніторингом задля підвищення оперативності реагування на інциденти безпеки.

Попри існування певних аналогів, відсутні комплексні рішення, які б забезпечували повний аналіз IP-адрес та доменів безпосередньо в інтерфейсі Telegram, з можливістю формування автоматичних звітів у форматі PDF. Це створює необхідність подальшого розвитку подібних систем.

Мета цієї роботи полягає у створенні платформи, призначеної для моніторингу та аналізу доменних імен та IP-адрес на предмет можливих кіберзагроз, який би інтегрувався з провідними службами кіберрозвідки та формував аналітичні звіти у зручному для користувача вигляді.

Для досягнення поставленої мети визначено такі завдання:

– дослідити актуальні методики, концепції та інструменти для перевірки доменних імен та IP-адрес, охоплюючи використання API таких сервісів, як VirusTotal, AbuseIPDB, Shodan;

– розробити архітектуру платформи, яка б упорядкувала логічний процес комунікації користувача з механізмом оцінки безпеки мережевих ресурсів;

– здійснити розробку модулів для інтеграції з вказаними API-сервісами задля агрегації аналітичної інформації стосовно репутаційного статусу та технічних атрибутів доменів і IP-адрес;

– створити алгоритм опрацювання отриманих результатів та формування узагальнених висновків щодо рівня кібербезпеки перевірених об’єктів;

– забезпечити функціональність для автоматичного створення оформленого звіту у форматі PDF, який повинен містити результати проведеного аналізу;

– провести тестування створеної платформи з метою визначення її операційної спроможності, коректності наданих даних та загальної швидкості функціонування системи;

– проаналізувати отримані результати, встановити рівень ефективності розробленого рішення та сформулювати можливі шляхи для оптимізації.

Об’єктом дослідження є процес автоматизованого контролю безпечності мережевих ресурсів у середовищі Telegram.

Предметом дослідження є методики, алгоритми та програмні засоби для аналізу доменних імен та IP-адрес на наявність ознак шкідливої діяльності шляхом використання зовнішніх API-сервісів.

Наукова новизна роботи полягає у розробці інтегрованого рішення для комплексного аналізу доменів і IP-адрес у середовищі Telegram із залученням кількох зовнішніх джерел OSINT-даних. На відміну від існуючих рішень, платформа передбачатиме автоматичне формування звітів у форматі PDF, систему обробки помилок API, а також адаптивну архітектуру, що дозволяє масштабувати систему шляхом підключення нових сервісів кібермоніторингу.

Розроблена платформа має прикладне значення для фахівців із кібербезпеки, системних адміністраторів та аналітиків, яким необхідна швидка перевірка мережевих об’єктів без застосування складних програмних засобів. Система дозволяє здійснювати моніторинг у режимі реального часу (online), формувати звіти для внутрішніх перевірок безпеки та спрощує процес обміну аналітичними даними між користувачами.

Отримані результати можуть бути використані як у подальших наукових дослідженнях, так і в освітньому процесі з питань інформаційної безпеки, а також при створенні аналогічних рішень у сфері кібермоніторингу.

Результати дослідження було подано та прийнято для апробації, шляхом публікації у науковому віснику ЛНТУ, що підтверджено у додатку А.

# РОЗДІЛ 1
# АНАЛІЗ ПРОБЛЕМАТИКИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

## 1.1 Огляд і аналіз предметної області проблеми (задачі), результатів існуючих теоретичних та експериментальних досліджень

У нинішньому інформаційному середовищі імена доменів, так само як IP-адреси, виступають фундаментом мережевої інфраструктури, забезпечуючи функціонування цифрових сервісів, комунікації та бізнес-процесів. Збільшення кількості кібератак, використання зловмисниками шкідливих доменних імен і підконтрольних їм мережевих адрес підкреслює необхідність ефективних способів визначення рівня їхньої захищеності та суспільної довіри. Наукові розробки у сфері кібербезпеки все частіше звертаються до концепції аналітики загроз (Cyber Threat Intelligence, CTI), залучаючи дані з відкритих джерел (Open Source Intelligence, OSINT). Зокрема, праці Rahman M. S. [1] та інших авторів акцентують увагу на критичній ролі OSINT у процесі ідентифікації небезпек, констатуючи, що навіть при наявності великих масивів інформації виникають труднощі, пов’язані з валідністю, методами обробки та правовими нормами.

У межах зазначеного дослідження автори демонструють схему повного циклу аналітичної роботи на основі відкритих джерел (OSINT), що включає фази планування, збору матеріалів, їхньої обробки, аналітичного опрацювання та наступного поширення інформації, а також подальшого корегування усього процесу. Ця модель підкреслює принципи упорядкованості та послідовності роботи з відкритими даними, наголошуючи на потребі їх попередньої верифікації, очищення та інтерпретації перед формуванням будь-яких аналітичних висновків. Логічно впорядкована послідовність якраз і забезпечує ефективне застосування OSINT-підходів для оцінки рівня захищеності ресурсів мережі.

Схема ілюструє цей цикл, показуючи, як кожен із його етапів слугує фундаментом для переходу до наступного: починаючи від визначення цілей та

плану дій, переходячи до збору й первинної обробки сирих даних, їхнього глибинного аналізу і, зрештою, до поширення отриманих результатів. Фінальний етап, який передбачає оцінку ефективності виконаної роботи та доопрацювання вимог, гарантує ітеративність усього механізму та його постійне вдосконалення. Така структурована система (рис. 1.1) наочно демонструє, як послідовне і впорядковане опрацювання публічної інформації веде до формування надійних аналітичних висновків та підвищує результативність використання OSINT-технологій у сфері кібербезпеки.

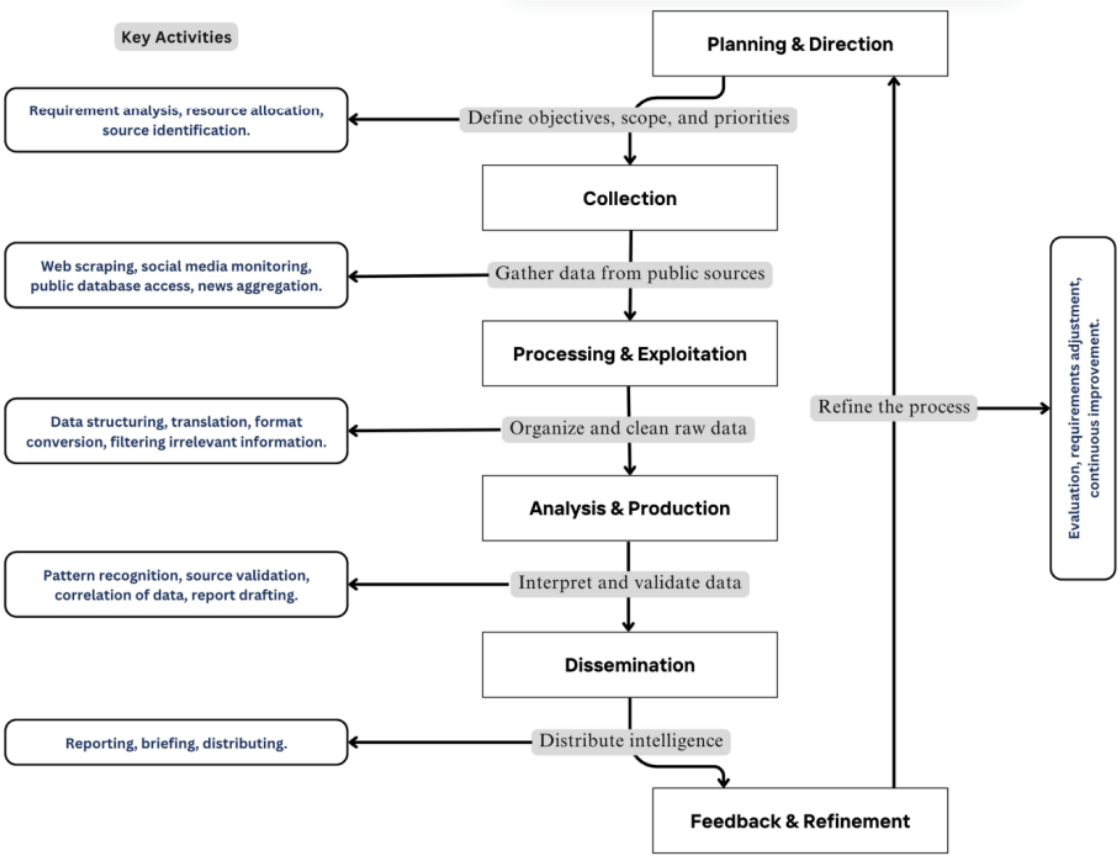


Рисунок 1.1 – Цикл відкритих джерел інформації: структурований підхід до збору та аналізу даних [1]

Водночас, окремою важливою областю досліджень стало вивчення доменних імен через DNS-записи та IP-адрес шляхом аналізу метаданих мережевих з’єднань. Зокрема, праця Zhauniarovich Y. та колег «A Survey on Malicious Domains Detection through DNS Data Analysis» [2] здійснила

систематизацію методів ідентифікації шкідливих доменів, на основі їхніх DNS-поведінкових характеристиках, класифікуючи їх за типами джерел даних, застосовуваними методами аналізу та критеріями оцінювання. Автори цієї роботи звертають увагу на те, що одним із суттєвих обмежуючих факторів є неповнота (фрагментарність) наявної інформації та недостатній рівень об'єднання відомостей із розрізнених джерел, що зменшує загальну дієвість механізмів виявлення.

У сфері розвідки загроз кібербезпеки (CTI) також спостерігається зростання інтересу до стандартизації інтеграції даних та підвищення якості інформаційних джерел. Наприклад, систематичний аналітичний огляд, проведений Santos P. [3] та його співавторами, виявив, що навіть актуальні системи (рамки) для інтеграції CTI-даних містять помітні недоліки стосовно надійності, простоти імплементації та практичної придатності.

Попри наявність значної кількості окремих інструментів для аналізу доменів та IP-адрес, їхні функціональні можливості часто залишаються фрагментарними та обмеженими. Особливо це помітно у випадку Telegram-ботів, які позиціонуються як доступні та швидкі засоби взаємодії з сервісами кібербезпеки, але здебільшого не забезпечують ані належної інтеграції кількох інформаційних джерел, ані повного спектра аналітичних можливостей. Узагальнену характеристику таких рішень подано у таблиці 1.1, що дозволяє оцінити актуальні можливості існуючих бот-інструментів та визначити функціональні прогалини, які мають бути усунені у межах цієї роботи.

Таблиця 1.1 – Порівняння функціоналу Telegram-ботів

| Параметр | @VirusTotal_AV_bot | @BesthostingBot | @ip_tools_bot |
|---|---|---|---|
| Перевірка IP-адрес | Є | Частково | Є |
| Перевірка доменів | Є | Є | Є |
| Перевірка файлів | Є | Немає | Немає |
| Звіт у вигляді PDF | Немає | Немає | Немає |
| Інтеграція з кількома API | Частково | Немає | Частково |
| Повідомлення про помилки вводу | Частково | Немає | Частково |
| Технічна інформація (ASN, ISP) | Частково | Є | Є |
| Детальний аналіз вразливостей | Немає | Немає | Немає |
| Вивід статистики/оцінки | Є | Частково | Є |
| Зручність інтерфейсу | Середня | Середня | Середня |

Враховуючи це порівняння, жоден із розглянутих Telegram-ботів не забезпечує комплексного підходу до аналізу мережевих ресурсів. Хоча окремі рішення підтримують базові функції перевірки IP-адрес і доменів, більшість з них не надають можливості працювати з кількома API-джерелами одночасно, не генерують структурованих звітів (зокрема у PDF-форматі) та лише частково інформують користувача про можливі помилки вводу. Функціонал щодо технічних атрибутів (ASN, ISP) наявний лише частково, а можливості глибинного аналізу вразливостей повністю відсутні у всіх представлених інструментах. Загальний рівень зручності інтерфейсу оцінюється як середній, що додатково підтверджує брак уваги до аспектів користувацького досвіду.

Аналіз практичного застосування також свідчить про активне використання Telegram-ботів у кіберпросторах – вони слугують як засоби для комунікації, так і потенційні канали атак. Дослідження стосовно Telegram-ботів [4], залучених до фішингових операцій, демонструє, що ця платформа може бути використана не лише для моніторингу, а й для передачі зловмисних даних. Згідно з отриманими даними (рис. 1.2), частота виявлення фішингових ботів в Telegram, демонструє нестабільність із періодичними сплесками активності.

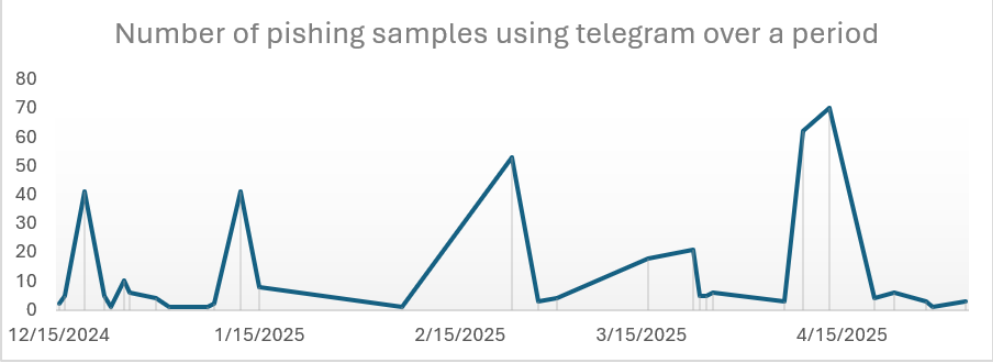


Рисунок 1.2 – Кількість використань Telegram Bot для створення ботів з метою фішингу з грудня 2024 року [4]

Найвищі пікові значення зафіксовано в середині лютого та у квітні 2025 року, що може свідчити про хвильовий характер фішингових кампаній та

здатність зловмисників швидко адаптуватися до зовнішнього середовища. Така поведінка підкреслює необхідність проєктування безпечної архітектури ботів та інтеграції з інструментарієм мережевого аналізу для забезпечення оперативного реагування на виникаючі небезпеки.

Тому, попередні наукові й практичні здобутки дають чітке розуміння методології аналізу доменів та IP-адрес, переліку застосовуваних інструментів та існуючих проблем. Ключовими викликами наразі лишаються: слабка взаємозв'язаність джерел даних, обмежена ступінь автоматизації робочих процесів, відсутність інтуїтивно зрозумілого інтерфейсу для широкого кола користувачів, а також питання забезпечення стійкості та можливості масштабування розроблених рішень.

У рамках цієї роботи поставлено завдання розробити інтегроване програмне рішення, яке об’єднає функції перевірки доменів та IP-адрес через публічно доступні API-сервіси, забезпечить взаємодію через бот-інтерфейс і надаватиме згенеровану інформацію у форматі, зручному для кінцевого користувача. У підсумку, метою роботи є усунення вищевказаних недоліків шляхом створення апаратно-програмного інструменту, який буде автоматизованим, модульним та простим у використанні для оцінки мережевої інфраструктури.

## 1.2 Огляд і аналіз методів та засобів розробки систем моніторингу безпеки доменів і IP-адрес для вирішення проблеми дослідження

У межах дослідження питання автоматизації перевірки безпеки доменних імен та IP-адрес вимагає застосування передових комп’ютерних технологій, що охоплюють створення програмного забезпечення, інтеграцію зовнішніх сервісів і розробку інтуїтивно зрозумілого інтерфейсу для кінцевого користувача. У цьому підрозділі представлені інструменти та методи, придатні для втілення цього функціоналу, зокрема: створення інтерфейсу через бота у месенджері

Telegram, застосування бібліотек на базі Python, взаємодія з програмними інтерфейсами сторонніх платформ та генерація підсумкових звітів.

Однією з основних технологій, яка забезпечує розробку ботів у Telegram, є бібліотека python-telegram-bot, що надає асинхронний інтерфейс до Telegram Bot API. Вона сумісна з Python версій 3.9+ і містить високорівневі класи для обробки команд, повідомлень і callback-подій [5-7].

З її допомогою можливо реалізувати механізми реагування на вхідні запити, створення меню з інтерактивними кнопками та керування сеансом бота. З огляду на це, використання цієї бібліотеки дозволяє сформувати системний компонент для комунікації з користувачем, який відповідає критеріям зручності та швидкодії.

Для інтеграції із сервісами кіберрозвідки доцільно використовувати клієнтські модулі API, наприклад, для доступу до VirusTotal, AbuseIPDB або Shodan. Їх застосування дозволяє автоматично отримувати інформацію про рівень довіри до IP-адрес, історію інцидентів, відкриті порти, ідентифікаційні дані сервісів, географічне розташування тощо. Ці дані є вхідними для аналітичного модуля системи. При цьому важливо враховувати вимоги до надійності, коректної обробки помилок API, дотримання лімітів на кількість запитів та приведення результатів до єдиної системи оцінювання. У контексті архітектури програмного забезпечення це означає конструювання модулю api_clients, який відповідає за інкапсуляцію взаємодії із зовнішніми сервісами.

Крім того, компонент report.py призначений для формування звітів у форматі PDF. Він спирається на бібліотеки Python, такі як ReportLab, для динамічного створення файлів, структурування зібраних даних та їх виведення в готовому форматі. Такий підхід відповідає сучасним стандартам автоматизації та дає можливість користувачу отримати аналітичний звіт безпосередньо через інтерфейс чат-бота. Для підтримки масштабованості та майбутнього розширення, архітектура системи організована з урахуванням модульності: поєднання bot/, api_clients/, analytics.py, report.py, config/, utils/

створює чітку логічну структуру коду, що спрощує інтеграцію нових джерел даних або функцій.

З методологічної точки зору розробки, система відповідає вимогам «інтерфейс-бот», «аналіз даних», «звітність», що покриває як функціональні, так і нефункціональні потреби – швидкість виконання, надійність, безпеку, можливість розширення та інтуїтивність. Додаткові механізми, такі як використання Python ORM, асинхронних обчислень (async/await) або імплементація механізму кінцевих станів (finite state machine) у роботі бота, також можуть бути залучені для підвищення ефективності реагування та опрацювання великих обсягів запитів. Наприклад, дослідники наголошують, що застосування асинхронності у ботах значно підвищує загальну продуктивність [8].

Підсумовуючи, аналіз доступних методів та інструментів дозволяє дійти висновку, що найбільш оптимальним рішенням для поставленого завдання є комбінація: чат-бота на базі Telegram, модульно структурованого програмного продукту, інтеграції з OSINT-ресурсами через їхні API, а також автоматизованого генератора звітів. Такий підхід дає змогу не лише реалізувати функціонал перевірки доменів та IP-адрес, але й гарантує зручність експлуатації, потенціал для зростання системи та високу швидкість роботи.

## 1.3 Постановка завдання на кваліфікаційну роботу магістра

На основі проведеного аналізу предметної області та вивчення актуального інструментарію розробки, сформульовано ключові завдання кваліфікаційної роботи, яке полягає у створенні та ретельному дослідженні інструменту для автоматизованого аналізу безпеки доменів та IP-адрес із використанням технологій платформи Telegram та зовнішніх сервісів кібермоніторингу.

Головна мета кваліфікаційної роботи полягає у створенні платформи, здатної забезпечити об’єднану перевірку доменних імен та IP-адрес на

наявність ознак зловмисної активності, потенційних дірок у захисті та підозрілої поведінки, базуючись на інформації з доступних джерел (OSINT-ресурси).

Для досягнення цієї мети необхідно виконати низку завдань:

– провести аналіз сучасних методів, підходів та інструментів для перевірки доменів і IP-адрес, включно з API-сервісами VirusTotal, AbuseIPDB, Shodan та іншими OSINT-платформами;

– спроєктувати архітектуру платформи, яка б логічно організовувала взаємодію кінцевого користувача із системою перевірки безпеки мережевих ресурсів;

– реалізувати модулі для інтеграції з API-сервісами з метою збору аналітичних даних щодо репутації та технічних характеристик доменів і IP-адрес;

– створити алгоритм обробки отриманих результатів і формування узагальнених висновків щодо рівня безпеки перевірених об’єктів;

– забезпечити автоматичне генерування структурованого звіту у форматі PDF, що міститиме результати аналізу, відомості про виявлені загрози, інциденти, технічні параметри та зроблені висновки;

– провести комплексне тестування розробленої платформи з метою оцінки її працездатності, достовірності отриманих даних та загальної швидкодії системи;

– проаналізувати здобуті результати, визначити ступінь ефективності створеного рішення та окреслити потенційні шляхи його подальшого вдосконалення.

Реалізація зазначених завдань дозволить створити програмний засіб, що поєднує зручність використання месенджера Telegram з потужним потенціалом OSINT-аналізу, що, своєю чергою, сприятиме підвищенню ефективності моніторингу кіберзагроз та укріпленню інформаційного захисту мережевих ресурсів.

Проведений аналіз предметної області, наукових праць та актуальних технічних реалізацій підтверджує своєчасність завдання автоматизованої оцінки безпеки доменних імен та IP-адрес шляхом використання даних відкритих джерел. Було встановлено, що наявні методики характеризуються фрагментарністю, обмеженою інтеграцією результатів з різних платформ та недостатнім рівнем автоматизації, що знижує їхню практичну цінність.

Огляд доступних методів і засобів розробки підтвердив доцільність вибору Telegram-бота як універсального інтерфейсу взаємодії, здатного забезпечити доступність, мобільність і оперативність моніторингу. Інтеграція з API провідних OSINT-сервісів, застосування модульної архітектури та інструментів для автоматичного формування звітності створюють необхідне підґрунтя для побудови комплексного інструменту для аналізу мережевих ресурсів.

Перший розділ закладає теоретичний фундамент та методичну базу для подальших розробок, виділяє ключові вимоги до функціоналу системи та обґрунтовує потребу у створенні інтегрованого рішення для результативного моніторингу кіберзагроз.

# РОЗДІЛ 2
# ТЕОРЕТИЧНЕ ДОСЛІДЖЕННЯ ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ МОНІТОРИНГУ БЕЗПЕКИ ДОМЕНІВ І IP-АДРЕС

## 2.1 Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання

Для реалізації плафтформи, призначеної для моніторингу безпеки доменних імен та IP-адрес, постало питання щодо обґрунтованого вибору найбільш відповідних технологій програмування, архітектурних рішень та алгоритмів. Це було необхідно для забезпечення ефективного функціонування системи, її надійності та можливості масштабування. У ході попереднього аналізу було встановлено, що головним завданням є створення такого програмного інструменту, який би міг обробляти запити користувачів у режимі реального часу, взаємодіяти з різноманітними зовнішніми API-сервісами, а результати надавати у чітко структурованому форматі.

Основною мовою програмування обрано Python, оскільки її перевага полягає у широкому спектрі доступних бібліотек для роботи з програмними інтерфейсами, маніпулювання даними, формування звітів та інтеграції з платформою Telegram. Python дозволяє досягнути оптимального співвідношення між легкістю розробки, швидкодією та наявністю великої кількості готових рішень з відкритим кодом. Для реалізації взаємодії з Telegram було задіяно бібліотеку python-telegram-bot, яка підтримує всі основні функції Telegram Bot API та гарантує стабільну роботу під час обробки запитів від користувачів.

У процесі розробки платформи для аналізу доменних імен та IP-адрес на наявність потенційних кіберзагроз буде інтегровано три ключові зовнішні сервіси, що забезпечують комплексний підхід до оцінки рівня безпеки мережевих ресурсів: VirusTotal [9], AbuseIPDB [10] та Shodan [11]. Кожен із них виконуватиме свою унікальну роль у загальній системі, формуючи багаторівневу модель аналізу даних.

Сервіс VirusTotal (рис. 2.1) є одним із найбільш відомих та визнаних інструментів у сфері кіберзахисту та аналізу ПЗ. Його основна мета полягає у виявленні потенційно небезпечних файлів, URL-адрес, IP-адрес і доменів. Це досягається шляхом їх порівняння з базами даних, що підтримуються антивірусними компаніями та системами виявлення загроз.


Рисунок 2.1 – Головна сторінка сервісу VirusTotal [12]

VirusTotal здійснює комплексний аналіз об’єктів, використовуючи понад 70 антивірусних механізмів (зокрема, Bitdefender, ESET, Microsoft Defender, ClamAV та інші), а також сервіси перевірки репутації сайтів і доменів (Google Safe Browsing, URLhaus, PhishTank та інші). Такий підхід дозволяє отримати об’єктивну оцінку безпечності ресурсу або файлу, спираючись на результати кількох незалежних аналітичних систем.

Під час перевірки домену або IP-адреси сервіс здійснює збір метаданих, які включають:

– репутаційні показники – облік позитивних (підозрілих або шкідливих) та негативних (безпечних) оцінок від антивірусних систем;

– інформацію про домен – дату реєстрації, використання SSL-сертифікатів, IP-геолокацію, DNS-записи (A, MX, TXT тощо);

– деталі мережевих з'єднань – зокрема, інформацію про хости, з якими взаємодіє IP-адреса, і країни розташування цих вузлів;

– історію аналізу – попередні перевірки об'єкта, тенденції зміни репутації з часом.

Крім того, VirusTotal пропонує API-інтерфейс, який дає змогу автоматизувати процес перевірки у сторонніх додатках, зокрема інтегрувати його у Telegram-бот для оперативної оцінки безпечності введених користувачем IP-адрес або доменів. Це дає змогу отримати миттєвий результат без необхідності ручного переходу на сайт.

З огляду на це, VirusTotal виступає як універсальна платформа антивірусної аналітики, що поєднує багаторівневі методи виявлення шкідливих об'єктів, формує репутаційні рейтинги та пропонує гнучкі можливості інтеграції в автоматизовані системи моніторингу кіберзагроз.

Другим важливим елементом системи є AbuseIPDB (рис. 2.2) – загальнодоступна база даних інцидентів зловживання IP-адресами і призначена для збору, зберігання та аналізу інформації про підозрілу або протиправну діяльність в мережі Інтернет.

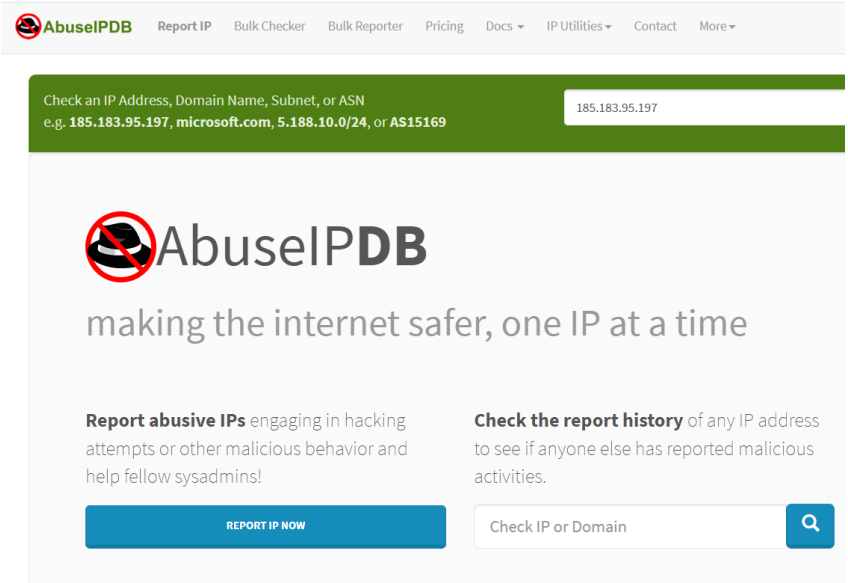


Рисунок 2.2 – Головна сторінка сервісу AbuseIPDB [13]

Основна мета цієї платформи полягає в наданні допомоги системним адміністраторам, фахівцям із забезпечення кібербезпеки та дослідникам у процесі ідентифікації та блокування тих IP-адрес, які були зафіксовані як учасники атак, розсилки спаму, діяльності ботнетів, несанкціонованих вторгнень або інших форм мережевих правопорушень.

AbuseIPDB розроблено як спільну платформу, де користувачі та організації діляться інформацією про виявлені інциденти, зазначаючи тип загрози, IP-адресу, час події, звідки надійшло спостереження, та додаткові коментарі. На підставі накопичених даних формується індекс підозрілості IP-адреси (Abuse Confidence Score) – числова оцінка від 0 до 100, що вказує на ступінь впевненості у зловмисному характері цієї адреси.

Сервіс опрацьовує та зберігає наступні основні типи даних:

– загальна кількість зареєстрованих скарг на певну IP-адресу разом із хронологією цих інцидентів;

– класифікація зафіксованих кібератак, тобто спроби підбору паролів до SSH (brute-force), атаки типу «відмова в обслуговуванні» (DoS/DDoS), фішинг, розсилка спаму, діяльність ботнетів, SQL-ін’єкції та інші;

– географічні дані – відомості про країну розташування, інтернет-провайдера та автономну систему (AS);

– історія накопиченої активності IP-адреси, що дає змогу простежити зміни в її поведінці протягом часу;

– дані про репутацію, отримані завдяки інтеграції з іншими моніторинговими платформами (наприклад, Spamhaus, Project Honey Pot).

Крім того, AbuseIPDB пропонує REST API, який забезпечує можливість швидкої, автоматизованої перевірки IP-адрес. У контексті застосування в платформі, це дозволяє надсилати запит до бази даних AbuseIPDB та відображати стислий звіт у зручному форматі, включаючи рівень небезпеки, загальну кількість скарг, перелік ідентифікованих видів атак та дату останнього інциденту.

Отже, AbuseIPDB функціонує як інструмент для швидкого аналізу інцидентів, допомагаючи оцінити репутацію IP-адреси, спираючись на колективний досвід спільноти кібербезпеки. Його висновки є особливо цінними для ідентифікації потенційних джерел загроз, визначення рівня довіри до адреси та проактивного запобігання інцидентам у сфері інформаційної безпеки.

Третім ресурсом, вбудованим у систему (рис. 2.3), є Shodan – унікальна пошукова система, призначена для виявлення, аналізу та постійного моніторингу пристроїв, підключених до мережі Інтернет. На відміну від традиційних пошукових систем, які індексують текстовий контент вебсторінок, Shodan фокусується на інфраструктурних мережевих елементах: серверах, роутерах, камерах відеоспостереження, промислових контролерах, інтернет-шлюзах, компонентах систем «розумного будинку» тощо.

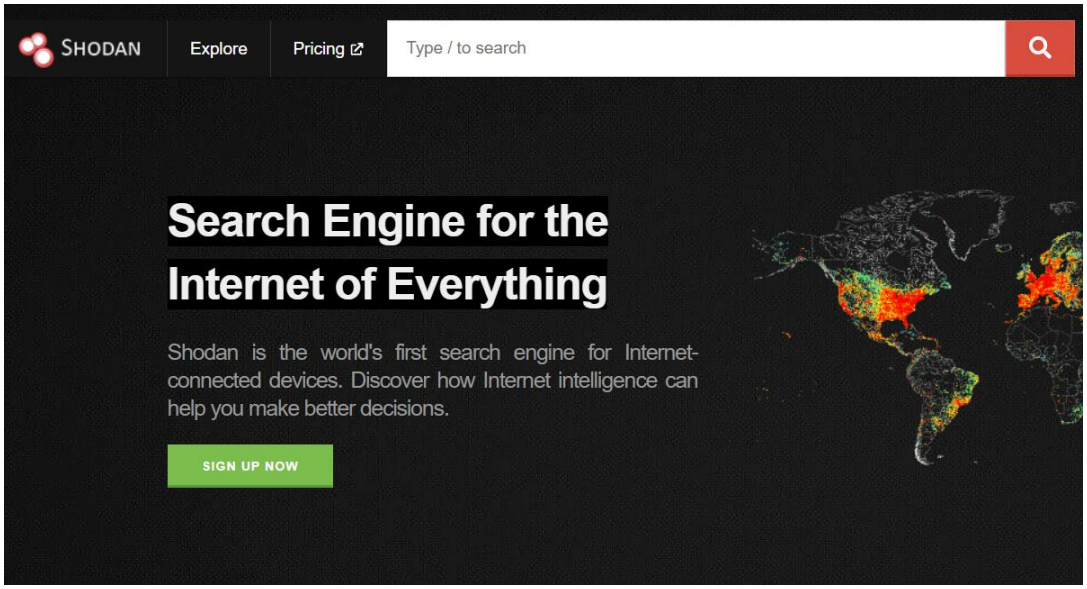


Рисунок 2.3 – Головна сторінка сервісу Shodan [14]

Сервіс проводить регулярне та систематичне сканування мережевого простору Інтернету, використовуючи автоматизовані системи збору даних, які надсилають запити на різні порти у глобальній мережі з метою виявлення активних пристроїв. У процесі цього сканування Shodan реєструє відповіді серверів, що дозволяє не тільки визначити факт доступності пристрою, але й зібрати низку додаткових технічних параметрів.

Зокрема, у ході аналізу Shodan фіксує детальну інформацію щодо:

– використовуваних протоколів та відкритих портів, через які система підключається до зовнішнього світу (наприклад, SSH, RDP, FTP, HTTP, Telnet, MQTT та інші);

– ідентифікаційних «банерів» сервісів, що містять метадані про запущені програми, версії програмного забезпечення, тип операційної системи, а також додаткові відомості, що стосуються конфігурації серверів;

– географічного розташування системи, визначеного на підставі IP-адреси, що уможливлює формування аналітичних карт розміщення систем та виявлення певних регіональних тенденцій у розбудові кіберінфраструктури;

– імовірності наявності відомих слабких місць, якщо версія служби чи ПЗ фігурує у базах даних CVE (Common Vulnerabilities and Exposures) з описом експлойтів;

– даних про приналежність IP-адреси до конкретного мережевого оператора, автономної системи (AS) чи організації, що може бути корисним для ідентифікації власників чи операторів мережевого обладнання.

Shodan не обмежується лише поверхневою перевіркою доступності пристроїв, а й виконує глибокий технічний розбір мережевих взаємодій, що дає змогу оцінити рівень захищеності конкретного домену або IP-адреси. Завдяки цьому сервіс активно використовується фахівцями з кібербезпеки для виявлення неналежно налаштованих серверів, а також відкритих сервісів та портів, які можуть стати точками входу для атак.

Впровадження функціоналу Shodan у цю платформу дозволяє доповнити звітність розвідувальних даних технічного рівня, формуючи повне уявлення про мережеву архітектуру об’єкта дослідження та підвищуючи точність оцінки його стійкості до кібератак.

Об’єднання цих трьох інструментів надає багатовимірний аналітичний результат: VirusTotal виконує антивірусну верифікацію та репутаційний рейтинг, AbuseIPDB забезпечує статистичну оцінку довіри до IP-адреси, а Shodan надає детальну технічну інформацію про сервер чи пристрій. Разом

вони формують єдину систему контролю кіберзагроз, що презентує користувачеві повну та об’єктивну картину стану безпеки досліджуваних ресурсів.

Для створення звітів у форматі PDF, що вирізняються високою якістю візуалізації текстових та графічних даних, було задіяно бібліотеку ReportLab у процесі розробки.

Безпека зберігання токенів API забезпечена через застосування .env-файлів, що унеможливлює витік конфіденційної інформації у коді. Управління необхідними залежностями реалізовано за допомогою файлу requirements.txt, що значно спрощує процес налаштування робочого середовища.

Для реалізації платформи, призначеної для OSINT-аналізу мережевих об’єктів, було обрано підхід, що базується на модулях та компонентах. Цей підхід передбачає розподіл програмного коду на незалежні структурні частини та сприяє масштабованості, зручності супроводу, підвищенню надійності системи та можливості швидкої інтеграції нових сервісів.

Кожен із компонентів системи несе відповідальність за окрему, чітко окреслену функцію, що надає архітектурі платформи структурності, логічної цілісності та гнучкості.

Модуль bot/ відповідає за комунікацію із кінцевим користувачем: він приймає надані користувачем IP-адреси або імена доменів, опрацьовує команди та повідомлення, а також реалізує інтерактивні аспекти інтерфейсу Telegram. Це забезпечує простоту взаємодії та інтуїтивно зрозумілу послідовність діалогу.

Блок api_clients/ слугує комунікаційним шаром системи, забезпечуючи зв’язок із зовнішніми платформами, такими як Shodan, AbuseIPDB та VirusTotal. У цьому елементі реалізовано процеси формування уніфікованих запитів до API, отримання відповідей у форматі JSON та їх первинної обробки.

Завдання генерації автоматизованих аналітичних звітів у форматі PDF, використовуючи ReportLab, виконує файл report.py. Згенеровані документи містять систематизовані дані щодо результатів перевірки, які включають

технічні характеристики, оцінки репутації, виявлені потенційні ризики та пропозиції щодо усунення вразливостей.

Аналітична робота з отриманими даними, що полягає в агрегації інформації з різних джерел, її порівнянні та визначенні рівня загрози об’єкта (від безпечного до потенційно небезпечного), виконується модулями analytics.py та analytics_domain.py. Саме ці компоненти забезпечують функціонал інтелектуального аналізу та організації інформації.

Тека config/ містить файли налаштувань, загальносистемні змінні, а також шляхи до ключів доступу API, що дозволяє централізовано керувати середовищем системи. Каталог utils/ включає набір додаткових функцій, призначених для перевірки коректності вхідних даних, обробки винятків та форматування результатів, підвищуючи таким чином надійність та безпеку функціонування.

Завдяки такому компонуванню системи досягається можливість її масштабування, незалежного оновлення окремих частин та простоти технічного супроводу коду, що є принципово важливим для рішень, які залежать від взаємодії із зовнішніми програмними інтерфейсами.

З метою впровадження концепції архітектури, що була окреслена, було здійснено вибір найбільш вдалого набору технологій. Ці технології забезпечують належну працездатність Телеграм-бота, його стійкість до різноманітних збоїв, а також відповідають нинішнім стандартам щодо захищеності та максимізації зручності для кінцевого користувача. У таблиці 2.1, що подається нижче, детально аргументовано рішення щодо ключових технологій, допоміжних засобів та програмних бібліотек, які були задіяні під час створення цього програмного виробу.

Таблиця 2.1 – Обґрунтування вибору технологій для реалізації платформи

| Компонент / технологія | Призначення | Обґрунтування вибору |
|---|---|---|
| Python 3.10+ | Основна мова розробки | Має потужні бібліотеки для API, аналітики та Telegram; простота читання та обслуговування коду |
| python-telegram-bot | Інтеграція з Telegram API | Забезпечує асинхронну роботу, підтримує InlineKeyboard, обробку команд і повідомлень |

Продовження таблиці 2.1

| Компонент / технологія | Призначення | Обґрунтування вибору |
|---|---|---|
| VirusTotal API | Аналіз репутації IP і доменів | Відкрите API, широке охоплення шкідливих об’єктів |
| AbuseIPDB API | Визначення рівня небезпеки IP | Забезпечує оцінку репутації та виявлення інцидентів |
| Shodan API | Отримання технічної інформації про IP | Визначає відкриті порти, сервіси, уразливі конфігурації |
| ReportLab | Генерація PDF-звітів | Гнучкість у форматуванні, підтримка Unicode, можливість автоматичного звітоутворення |
| dotenv | Керування конфіденційними змінними | Забезпечує безпечне зберігання API-ключів |
| requirements.txt | Список залежностей | Сприяє відтворюваності середовища та легкому розгортанню проєкту |

Після визначення ключових архітектурних елементи та необхідних технологій, логічним кроком стає аналіз загального сценарію, як саме користувачі будуть взаємодіяти із системою. На схемі (рис. 2.4), представлено діаграму прецедентів (use case diagram), яка відображає покроковий алгоритм дій користувача при роботі з ботом у Telegram, функціонал якого полягає в оцінці рівня безпеки доменних імен та IP-адрес.

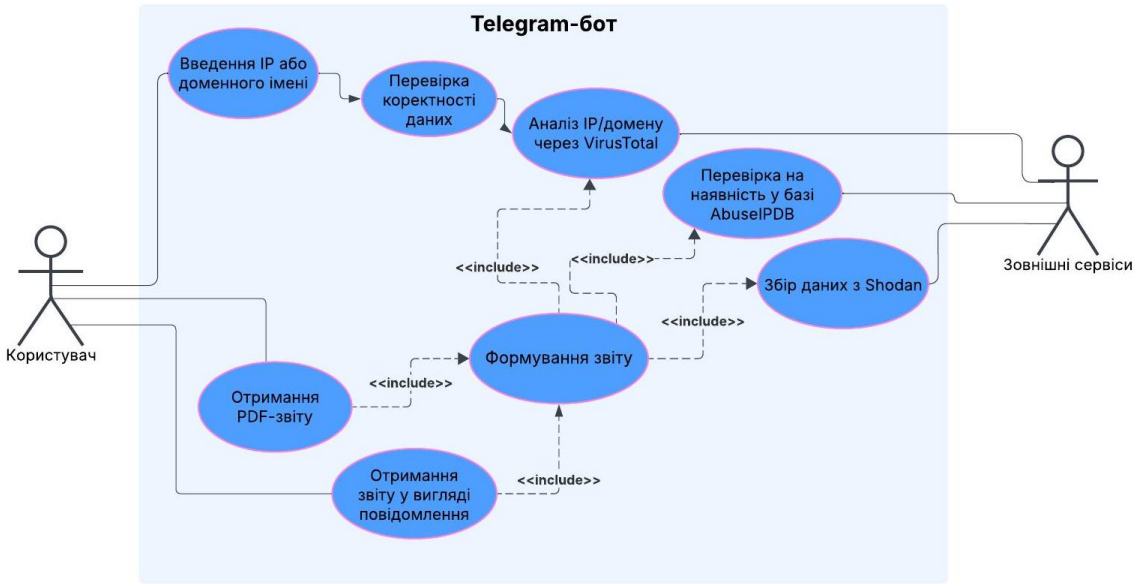


Рисунок 2.4 – Діаграма використання платформи для аналізу IP та доменів

*Джерело: розроблено автором*

Основним актором системи виступає користувач, який вводить IP-адресу або доменне ім’я через чат-інтерфейс Telegram. Після отримання запиту, платформа послідовно виконує низку кроків:

– валідація вхідних даних, що полягає у перевірці відповідності формату IP чи домену, аби уникнути помилкових чи невідповідних запитів;

– аналіз внесених даних через сервіс VirusTotal, що дає змогу визначити репутацію об’єкта, виявити зловмисне програмне забезпечення та отримати відомості про попередні інциденти;

– перевірка IP-адреси у базі AbuseIPDB для для виявлення зареєстрованих випадків зловживань і визначення рівня небезпеки;

– збір технічних даних за допомогою Shodan, отримуючи інформацію щодо відкритих портів, версій програмного забезпечення, запущених сервісів та інших характеристик пристрою;

– формує підсумковий звіт, який узагальнює результати з усіх трьох сервісів і виконує аналітичну оцінку рівня ризику;

– складання підсумкового звіту, в якому будуть узагальнені результати, отримані з усіх трьох джерел, та проведена аналітична оцінка рівня небезпеки;

– надання користувачеві результатів перевірки у зручній формі – короткий підсумок у вигляді повідомлення чи розгорнутий звіт у форматі PDF.

На діаграмі також відображено взаємодію платформи із зовнішніми сервісами VirusTotal, AbuseIPDB та Shodan, які функціонують як окремі незалежні актори системи. Вони відповідають за надання аналітичних та технічних даних, що слугують фундаментом для формування висновків про безпечність домену чи IP-адреси.

У підсумку, надана діаграма демонструє не лише логіку процесу аналізу, але й послідовність взаємодії між користувачем, ботом Telegram та зовнішніми базами джерелами, що у сукупності забезпечує всебічність та достовірність отриманих результатів.

Загальний алгоритм роботи системи базується на багатоетапній обробці запиту, що забезпечує коректність, достовірність та послідовність отриманих результатів. Логіка функціонування платформи така:

– отримання даних від користувача – бот приймає введену IP-адресу або доменне ім’я через чат-інтерфейс;

– валідація введення – здійснюється перевірка правильності формату, запобігання SQL- або XSS-ін’єкціям, а також обробка некоректних або неповних запитів;

– надсилання запитів до зовнішніх API – введені дані паралельно надсилаються до сервісів VirusTotal, Shodan та AbuseIPDB для отримання інформації про репутацію, технічні параметри й інциденти безпеки;

– обробка відповідей – отримані результати у форматі JSON аналізуються аналітичним модулем, де відбувається нормалізація полів, порівняння показників та обчислення інтегрального коефіцієнта ризику;

– формування аналітичного висновку – агреговані результати об’єднуються у єдину модель оцінки безпеки, що класифікує об’єкт як «безпечний» (safe), «підозрілий» (suspicious) або «шкідливий» (malicious);

– генерування звіту у PDF-форматі – створюється структурований документ із розділами, що містять опис репутації, технічні характеристики, статистику виявлених інцидентів і рекомендації щодо безпеки;

– виведення результатів у Telegram – користувач отримує коротке повідомлення з узагальненими висновками та можливістю завантажити повний звіт.

Звідси можна зробити висновок, що запропонований алгоритм забезпечує цілісну та надійну інтеграцію всіх етапів аналітичного процесу, підтримує автоматизоване та безперервне збирання даних із зовнішніх джерел, а також підвищує точність і об’єктивність визначення рівня небезпеки досліджуваних мережевих об’єктів.

## 2.2 Практична реалізація об’єкта проектування

У межах цієї кваліфікаційної роботи було створено платформу, призначенням якої є автоматизована діагностика доменних імен та IP-адрес на наявність потенційних загроз у кіберпросторі. Ключова мета практичної розробки полягає у формуванні інструменту, здатного здійснювати аналіз у режимі реального часу, надаючи користувачеві швидкий доступ до даних з відкритих джерел кібермоніторингу.

Розроблена платформа є системою, що складається з багатьох елементів і функціонує за принципом клієнт-серверної взаємодії. Користувач, застосовуючи інтерфейс месенджера Telegram, надсилає запити, які можуть бути IP-адресами або доменними іменами. Серверна частина програми приймає ці дані, обробляє їх, здійснюючи запит до зовнішніх аналітичних платформ, а саме VirusTotal, AbuseIPDB та Shodan. Отримані висновки систематизуються, агрегуються та повертаються користувачеві у вигляді текстових повідомлень або як згенерований звіт у форматі PDF.

Перш ніж розпочати практичне впровадження, було визначено низку вимог, котрі визначають завдання, функціональні можливості та технічні обмеження для платформи. Ці вимоги традиційно поділені на три основні категорії: функціональні, нефункціональні та технічні.

Платформа повинна реалізовувати базовий набір функціональних можливостей, що забезпечують результативну взаємодію користувача із системою. Передусім передбачається можливість введення даних у діалоговому режимі, коли користувач надсилає IP-адреси або доменні імена безпосередньо через інтерфейс Telegram. Після отримання таких даних система здійснює їхню попередню перевірку: виконується валідація формату, зокрема коректності структури IPv4/IPv6 або відповідності доменного суфікса встановленим стандартам. Лише після успішного проходження цього етапу ініціюється основний процес аналізу, що передбачає використання зовнішніх сервісів

VirusTotal, AbuseIPDB та Shodan для отримання достовірних відомостей про потенційні ризики.

Результати аналізу подаються двома способами: як текстове повідомлення для оперативного ознайомлення або у вигляді розширеного PDF-звіту, який містить деталізовані аналітичні дані. У разі виникнення помилок, зокрема відсутності відповіді від сервісу або передавання некоректних даних, користувач отримує чітке й інформативне повідомлення з поясненням можливих причин та рекомендаціями щодо подальших дій. Водночас система має функціонувати стабільно навіть за часткової недоступності окремих джерел інформації: у випадку збою одного сервісу аналіз продовжується, спираючись на інші доступні ресурси.

Зазначені функціональні вимоги поєднуються з низкою якісних характеристик, що визначають ефективність і надійність розробленої системи. Серед них важливе місце посідає швидкодія: затримка між запитом і відповіддю не повинна перевищувати 3-5 секунд, що досягається використанням асинхронних механізмів взаємодії з API та оптимізацією мережевих операцій. У випадку виникнення виняткових ситуацій бот мусить коректно обробляти помилки, уникаючи аварійного завершення роботи. Крім того, конфіденційні дані, зокрема API-ключі та токен бота, зберігаються у захищеному середовищі файлу .env, що мінімізує ризик їхнього витоку.

Архітектура системи також орієнтована на розширюваність: передбачено можливість інтеграції нових API або аналітичних модулів без значних модифікацій базового коду. Зручність користування забезпечується застосуванням навігаційних кнопок, коротких команд та інтуїтивно зрозумілих інструктивних повідомлень, що робить систему доступною навіть для користувачів без спеціальної технічної підготовки. Завершальною вимогою є сумісність із різними платформами розгортання, що дозволяє реалізовувати систему як на локальних серверах, так і в хмарних середовищах, включаючи Heroku, Render та Railway.

Виконання цих вимог забезпечує стабільний, безпечний та гнучкий режим роботи системи, що є важливим показником у секторі кібермоніторингу.

Для успішного розгортання платформи потрібне дотримання наступних технічних умов:

– наявність активного облікового запису Telegram із зареєстрованим ботом через BotFather;

– отримання необхідних API-ключів для VirusTotal, AbuseIPDB та Shodan;

– стабільне інтернет-з’єднання на сервері, що забезпечує можливість виконання HTTP-запитів до зовнішніх ресурсів;

– встановлена версія інтерпретатора Python 3.9+ та наявність усіх залежностей, перелічених у файлі requirements.txt, включаючи python-telegram-bot, requests, reportlab, dotenv.

Технічні вимоги формують фундамент для надійної експлуатації платформи та її потенційного подальшого розширення.

Система реалізована з використанням модульної структури, де кожен компонент відповідає за виконання певної, чітко визначеної функції, що сприяє упорядкованості та високій якості програмного коду. Такий підхід позитивно впливає на здатність системи до масштабування, її надійність, стабільність та полегшує подальший технічний супровід проєкту.

Модульність дозволяє оновлювати окремі сегменти системи незалежно, не ризикуючи порушити її загальну функціональність, що є важливим аспектом при роботі з динамічними API-службами.

Кожен структурний компонент програмної системи виконує чітко визначену функціональну роль, забезпечуючи узгоджену та модульну роботу всього застосунку. Блок bot/ реалізує взаємодію з Telegram API: він обробляє команди користувача, отримувані повідомлення та підтримує керування сеансами, що забезпечує коректний діалог між користувачем і ботом.

Каталог api_clients/ відповідає за інтеграцію з зовнішніми платформами аналізу, такими як VirusTotal, Shodan та AbuseIPDB. Саме в цій частині

формуються запити до API, виконується отримання та парсинг відповідей, а також узгодження форматів даних із внутрішніми структурами системи.

Модуль analytics.py забезпечує аналітичне опрацювання отриманої інформації: він виконує обчислення репутаційних показників доменів і IP-адрес, здійснює кореляцію атрибутів і формує підсумкові оцінки рівня ризику. У свою чергу, файл report.py відповідає за генерацію автоматизованих PDF-звітів на основі бібліотеки ReportLab, включаючи текстові інтерпретації, табличні структури й висновкові аналітичні блоки.

Каталог config/ містить конфігураційні параметри, зокрема ключі доступу, змінні середовища та системні налаштування, необхідні для коректної роботи модулів. Допоміжні функції зібрані у utils/ – це функції перевірки валідності введених даних, обробки результатів та їхнього форматування.

Окремий файл .env використовується як захищене сховище конфіденційної інформації, включно з API-ключами. Завдяки ізоляції таких даних забезпечується підвищений рівень безпеки та запобігання несанкціонованому доступу до зовнішніх сервісів.

З метою спрощення процесу інсталяції та впровадження системи передбачено наявність файлу requirements.txt, який містить повний перелік необхідних бібліотек та залежностей Python. Завдяки цьому розробник або адміністратор може швидко налаштувати робоче оточення, застосувавши одну команду.

Внутрішня логіка взаємодії між цими модулями побудована на принципах об’єктно-орієнтованого програмування, де кожен блок представлений як окремий клас із чітко визначеними властивостями та методами.

Така організація коду сприяє повторному використанню фрагментів коду, ізоляції різних функціональних частин та забезпечує можливість розширення системи без необхідності кардинальних архітектурних переробок.

Для візуалізації архітектури програмного забезпечення було створено діаграму класів (рис. 2.5), яка наочно відображає логічну будову платформи, взаємозв’язки між його складовими, ключові атрибути та механізми взаємодії.

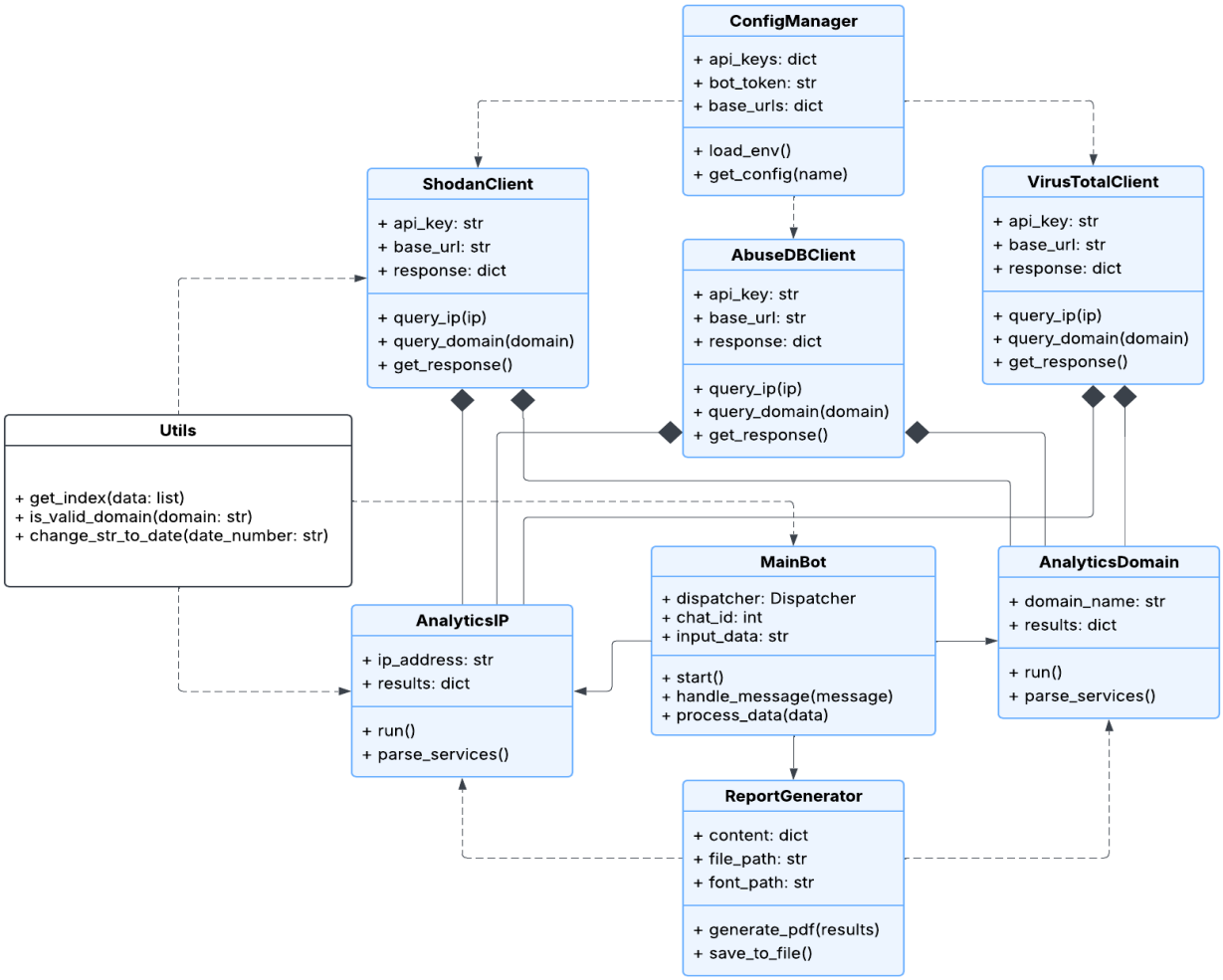


Рисунок 2.5 – Діаграма класів платформи для моніторингу безпеки доменів та IP-адрес

*Джерело: розроблено автором*

На діаграмі класів відображено ключові складові програмної системи, що формують її ядро та охоплюють усі етапи функціонування, від приймання користувацьких запитів до формування фінального підсумку. Центральним компонентом системи є клас MainBot, який керує взаємодією між користувачем та внутрішніми блоками. Саме він приймає вхідні запити, ініціює необхідні процедури аналізу та повертає кінцеві дані у зручному для сприйняття вигляді.

Функціональність MainBot безпосередньо залежить від класу ConfigManager, який виконує завдання завантаження конфігураційних параметрів, ключів доступу до API та налаштувань середовища. Такий підхід дозволяє модифікувати роботу системи без потреби втручатися у базовий програмний код.

Аналітичний функціонал забезпечується клієнтами ShodanClient, AbuseDBClient та VirusTotalClient, котрі взаємодіють із зовнішніми платформами кіберзахисту. Вони створюють запити до відповідних API, отримують відповіді, як правило, у форматі JSON, та передають їх на етап подальшої інтерпретації. Отримані дані проходять через модулі AnalyticsIP та AnalyticsDomain, які оцінюють рівень безпеки IP-адрес і доменних імен, розраховують показники ризику та генерують стислі аналітичні звіти.

Після завершення аналізу IP-адрес, згенеровані дані надходять до компонента ReportGenerator, відповідального за створення фінальних звітів у форматі PDF. Цей модуль впорядковує інформацію, використовуючи табличні структури, текстові секції та висновки, що гарантує наочність подачі матеріалу.

У додатку Б наведено фрагмент коду функції create_pdf_report(), що реалізує процес побудови PDF-звіту на основі результатів аналізу IP-адрес. Функція формує структурований документ, використовуючи елементи бібліотеки ReportLab, зокрема текстові блоки, таблиці та стилізовані заголовки. Її логіка передбачає поетапне розміщення аналітичних даних, включаючи загальні відомості, показники репутації та ключові висновки. Завдяки впорядкованій генерації елементів PDF-документ набуває цілісної структури, що забезпечує зручність подальшого використання та стандартизоване подання результатів досліджень.

Усі модулі системи взаємодіють з допоміжним модулем Utils. Він містить загальноприйняті функції для валідації вхідних даних, стандартизації виведення та маніпуляцій з датами. Це сприяє уніфікації реалізації базових операцій і мінімізує дублювання програмного коду.

Діаграма демонструє не лише ієрархічну структуру класів, але й ілюструє принципи зв’язку (композиції та залежності), які регламентують обмін інформацією між модулями. Наприклад, вихідні дані від клієнтів API послідовно прямують до аналітичних блоків, а звідти – до генератора звітів, формуючи чіткий конвеєр обробки даних. Дана архітектурна побудова забезпечує ізоляцію взаємодій між частинами системи, спрощуючи процес тестування, поточного супроводу та майбутнього розширення можливостей.

У наступному фрагменті коду (ліст. 2.1) показано приклад функції parse_ip_info(), яка здійснює парсинг JSON-відповіді, отриманої від API-сервісу AbuseIPDB, та трансформує її у впорядковані дані для подальшого аналізу.

Лістинг 2.1 – Отримання інформації з сервісу AbuseDB

```
def parse_ip_info(response):
    if response is not None:
        try:
            ip_data = {
                'ip': response['data']['ipAddress'],
                'hostname': response['data']['hostnames'],
                'domain': response['data']['domain'],
                'country': response['data']['countryCode'],
                'isp': response['data']['isp'],
                'abuse_score': response['data']['abuseConfidenceScore'],
                'total_reports': response['data']['totalReports'],
            }
            return ip_data
        except Exception as e:
            # print(f"AbuseDB response error: {e}")
            return None
```

кінець лістингу 2.1

Сервіс AbuseIPDB пропонує глибокий аналіз даних щодо IP-адрес, які були зафіксовані у діяльності, що викликає підозру чи є шкідливою, наприклад, поширення спаму, сканування портів або неправомірне використання ресурсів. Цей метод приймає аргумент response, котрий містить відповідь від програмного інтерфейсу (API) у форматі JSON. Перевірка на те, що response не є нульовим (if response is not None), є запобіжником, аби уникнути звернення до неіснуючого об’єкта, що могло б спричинити збій програми.

Центральна частина даного методу конструює словник ip_data, з якого витягуються важливі елементи: власне IP-адреса, асоційовані імена хостів (якщо є), домен, код країни, інформація про провайдера, показник достовірності (позначений як abuseConfidenceScore), а також загальна кількість зареєстрованих інцидентів (totalReports). Зібрана інформація повертається як формалізований словник Python, що забезпечує уніфікований вигляд для подальшої інтеграції у модулі аналізу або для формування звітів у форматі PDF. У разі виникнення будь-яких непередбачуваних ситуацій виконується механізм обробки винятків try/except, що дозволяє безпечно керувати помилками, не перериваючи функціонування платформи.

Загалом, функція parse_ip_info() виступає як посередник між зовнішнім API та внутрішньою архітектурою системи, забезпечуючи послідовне представлення отриманих даних та підвищуючи стійкість процесу аналітичних робіт.

У лістингу 2.2 показано впровадження функції під назвою check_domain_reputation(), яка взаємодіє з API-сервісом VirusTotal задля отримання репутаційного статусу наданого доменного імені.

Лістинг 2.2 – Перевірка репутації домену через VirusTotal

```
def check_domain_reputation(domain):
    virus_total_api_key = VIRUSTOTAL_API_KEY
    url = f"https://www.virustotal.com/api/v3/domains/{domain}"
    headers = {
        "x-apikey": virus_total_api_key
    }
    response = requests.get(url, headers=headers)
    data_responce = response.json()
    if data_responce is not None:
        result = collect_all_data(data_responce)
        pprint(data_responce)
        return result
    else:
        print("Не вдалося перевірити репутацію домену")
        return None
```

кінець лістингу 2.2

На старті цієї функції спершу ініціалізується змінна virus_total_api_key, яка зберігає API-ключ, потрібний для коректного доступу до сервісу. Після цього формується динамічне посилання для запиту у вигляді https://www.virustotal.com/api/v3/domains/{domain}, де місце {domain} займає доменне ім’я, надане користувачем. Для здійснення відправлення цього запиту залучається модуль requests, який надсилає HTTP-запит методом GET; авторизаційний ключ, що підтверджує наші права доступу до API, додається до заголовків (heads).

Отриманий відгук конвертується у формат JSON, а потім проводиться його перевірка на коректність. Якщо відповідь містить дані (не є порожньою), активується допоміжна процедура collect_all_data(), яка збирає найважливіші показники для аналізу: скільки загроз було виявлено, ступінь довіри до ресурсу, коли відбулося останнє сканування, а також відомості про те, скільки антивірусних систем спрацювало позитивно. Зібрана інформація пакується в об’єкт result, який згодом передається до аналітичного блоку для визначення загального рівня небезпеки та подальшого відображення у звіті формату PDF.

А у випадку, коли сервіс VirusTotal є недоступним або ж повертає некоректну відповідь, програма виводить повідомлення «Не вдалося перевірити репутацію домену», що забезпечує мінімальну стійкість системи до збоїв у роботі зовнішніх систем. З огляду на це, функція check_domain_reputation() виконує одну з центральних функцій у роботі платформи, гарантуючи надійний розбір доменних імен щодо наявності шкідливого коду, фішингових посилань чи зв’язків із мережами ботів.

У додатку В подано реалізацію функції https_certificate(), призначеної для отримання та аналізу інформації про SSL/TLS сертифікат ресурсу. Функція встановлює захищене з’єднання з вебсервером, отримує структуровані дані сертифіката (строк дії, емітент, доменне ім’я, криптографічні параметри) та трансформує їх у формат, придатний для подальшої обробки аналітичними модулями системи. Отримані характеристики дозволяють оцінити коректність

налаштування HTTPS, виявити потенційні ризики та врахувати їх під час формування інтегральної оцінки безпеки мережевого ресурсу.

Далі, у наступному фрагменті коду (ліст. 2.3), представлена реалізація функції calculate_abusedb_score(), яка призначена для оцінки рівня ризику, пов’язаного з певною IP-адресою, спираючись на дані, здобуті з сервісу AbuseDB (AbuseIPDB).

Лістинг 2.3 – Розрахунок аналітичного балу на основі даних AbuseDB

```
def calculate_abusedb_score(abusedb_info):
    score = 0
    details = []
    abuse_score = abusedb_info.get('abuse_score', 0)
    abuse_reports = abusedb_info.get('total_reports', 0)
    if abuse_score > 0:
        score = abuse_score / 100 * 5
        details.append(f'Оцінка зловживання: {abuse_score}% впевненості')
    if abuse_reports > 0:
        score += abuse_reports / 100 * 5
        details.append(f'Загальні скарги: {abuse_reports}')
    result = [score, details]
    return result
```

кінець лістингу 2.3

Цей компонент є ключовим елементом системи, адже саме він трансформує необроблені дані, отримані з API, у чисельну оцінку ступеня загрози, яка потім впроваджується у загальний аналітичний звіт.

На початку роботи функції ініціалізуються два головні елементи: score, призначений для фіксації сукупного числового балу ризику, та details – перелік текстових пояснень, що розкривають причини надання IP-адресі певної оцінки. Далі з інформаційного блоку abusedb_info витягуються важливі показники: abuse_score (ступінь довіри до звинувачень у неправомірній діяльності, виражений у відсотках) і total_reports (загальна кількість звернень користувачів щодо підозрілої активності).

Метод розрахунку базується на принципі кумулятивного нарахування балів. Якщо значення abuse_score перевищує нуль, до фінального результату

додається часткова оцінка, масштабована до п’ятибальної шкали, що відображає рівень підозрілості IP-адреси. Так само, якщо кількість скарг total_reports також є позитивною, показник ризику відповідно коригується у бік зростання. Паралельно, до списку details додаються інтерпретаційні коментарі, які допомагають осмислити отриманий результат і використовуються при формуванні документації у форматі PDF.

Після завершення всіх обчислень функція повертає двоскладовий набір даних: числовий індекс ризику (score) та детальний опис критеріїв (details), які вплинули на формування цієї оцінки. У результаті, calculate_abusedb_score() забезпечує прозору процедуру кількісного аналізу зловмисної активності IP-адрес, дозволяючи боту здійснювати об’єктивну оцінку потенційних загроз.

Отримані дані з цього модуля згодом об’єднуються з даними аналізу, отриманими від сервісів VirusTotal та Shodan, формуючи уніфікований індекс безпеки, що відображає загальний рівень довіри до об’єкта, що перевіряється.

Унаслідок проведених розробок була створена повноцінна платформа, яка утілює заявлені функції верифікації безпеки мережевих об’єктів. Під час етапів тестування система продемонструвала стійку працездатність, високу швидкість відгуку та належну обробку виняткових ситуацій. Середній час відповіді укладається у межі 3-5 секунд, що задовольняє нефункціональні вимоги.

Практичне впровадження засвідчило успішність обраних технологічних рішень, зокрема використання Python, додаткових бібліотек python-telegram-bot, requests, reportlab, а також підтвердило доцільність застосування архітектури, де модулі розподілені. Спроектований бот є гнучким, захищеним і готовим до подальшого розширення функціоналу, що позиціонує його як дієвий інструмент для оперативного нагляду за безпекою доменних імен та IP-адрес.

У цьому розділі було розроблено архітектурну модель платформи, націленої на моніторинг безпеки доменних імен та IP-адрес. Структура системи базується на модульному принципі, що гарантує її гнучкість, здатність до масштабування та легкість подальшого обслуговування. Описано основні

складові: блоки взаємодії з кінцевим користувачем, програмні клієнти для доступу до API сервісів VirusTotal, Shodan та AbuseIPDB, аналітичні блоки, а також модуль для формування звітів. Представлені діаграми та фрагменти коду ілюструють реалізацію ключових можливостей, включаючи отримання та аналітичну обробку даних, генерацію показників ризику та створення документації у форматі PDF. Розроблена будова забезпечує надійний зв’язок між компонентами та слугує фундаментом для майбутнього вдосконалення системи.

# РОЗДІЛ 3
# ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ РЕЗУЛЬТАТИВНОСТІ СИСТЕМИ МОНІТОРИНГУ БЕЗПЕКИ ДОМЕНІВ І IP-АДРЕС

## 3.1 Методика проведення дослідження

Методика дослідження у межах цієї кваліфікаційної роботи спирається на поєднання аналітичних, експериментальних і програмно-інженерних підходів та спрямована на перевірку ефективності розробленої платформи для моніторингу доменних імен та IP-адрес.

Метою роботи є створення інструмента, що інтегрується з провідними сервісами кіберрозвідки (VirusTotal, AbuseIPDB, Shodan) і формує аналітичні звіти у зручному форматі. Нижче наведено детальний опис послідовних етапів методики, обґрунтування вибору метрик, експериментального середовища, процедури тестування та обробки отриманих даних. На кожному етапі виділено особливості застосованих методів, умови для отримання достовірних результатів і способи мінімізації похибок.

Щоб комплексно оцінити ефективність різних підходів до перевірки доменів і IP-адрес, було проведено порівняння трьох методів: ручного, автоматизованого (через платформу Telegram) та напівавтоматичного. Зокрема, автоматизована перевірка демонструє істотне скорочення часу обробки запитів – приблизно у 10-12 разів швидше, ніж ручний збір даних, при цьому рівень повноти інформації залишається на високому рівні (понад 95 %). Незначне зменшення повноти пояснюється тим, що веб-інтерфейси інколи надають додаткові метадані, які не завжди доступні через API в реальному часі через ліміти запитів або інші технічні обмеження. Показник зручності визначався на основі результатів опитування тестової групи користувачів (N=10) і враховував кількість кроків, що необхідно виконати для отримання повного звіту.

Узагальнені результати цього аналізу наведено в таблиці 3.1, яка відображає ключові показники роботи кожного методу та дозволяє наочно простежити їх відмінності.

Таблиця 3.1 – Порівняльні показники методів перевірки

| № | Метод перевірки | Середній час на 1 запит, с | Повнота даних (0-100 %) | Кількість кроків | Оцінка зручності (0-10) | Надійність (відм. помилок / 100 запитів) | Коментар |
|---|---|---|---|---|---|---|---|
| 1 | Ручна перевірка (через веб-інтерфейси VirusTotal / Shodan / AbuseIPDB) | 45.0 | 100 | 6-8 | 6.5 | 0-2 помилки (мережа/captcha) | Повна інформація; висока складність; потрібні навички |
| 2 | Telegram-бот (автоматизована перевірка) | 3.8-4.2 | 95 | 2 | 9.2 | 0-1 помилка (перерва API) | Швидко; невелика втрата деяких додаткових полів (метадані) |
| 3 | Напівавтоматична (скрипт + ручний контроль) | 12-15 | 97 | 3-4 | 7.8 | 0-1 помилка | Компроміс між повнотою і швидкістю |

Ручний метод забезпечує максимальну повноту даних, проте є найповільнішим та найбільш трудомістким: користувачу потрібно переходити між кількома сервісами, вводити дані вручну та очікувати завантаження кожної сторінки. Цей підхід має найнижчу оцінку зручності та найбільшу кількість необхідних дій (6-8 кроків), що робить його менш придатним для регулярного використання.

Напівавтоматичний метод виступає проміжним рішенням: частина операцій виконується скриптом, але користувач все ще бере участь у процесі. Завдяки цьому вдається суттєво знизити час на один запит (до 12-15 секунд) та зберегти високу повноту даних (близько 97 %). Такий підхід потребує менше ручних дій, але вимагає базових технічних навичок і часом супроводжується необхідністю повторного запуску в разі API-помилок.

Найефективнішим виявився автоматизований метод через Telegram-бота, де час виконання запиту становить у середньому 4 секунди, а кількість дій

користувача зведена до мінімуму. Надійність залишається високою, а ризик помилок мінімальний і пов’язаний переважно з тимчасовими перебоями в API. Хоча в окремих випадках бот може не отримати деякі необов’язкові метадані, але це не впливає на загальну якість оцінки ризику.

Аналіз таблиці 3.1 підтверджує, що впровадження Telegram-бота дозволяє істотно оптимізувати процес перевірки, забезпечуючи баланс між швидкістю, повнотою та зручністю використання без суттєвих втрат якості результатів. Отримані порівняльні показники стали основою для подальшої методологічної частини дослідження, у межах якої було визначено ключові етапи оцінювання системи та формування підходів до її валідації. Саме на цьому ґрунтується наступний блок роботи, що починається з аналізу джерел даних та уточнення критеріїв оцінювання.

На початковому етапі було виконано системний аналіз відкритих джерел кіберінформації, доступних через API. Для кожного сервісу оцінювалися доступні поля, частота оновлення баз, наявність документованого API та обмеження (rate limits). В результаті було обрано VirusTotal для репутаційної оцінки доменів і IP, AbuseIPDB – для інцидентної статистики та обґрунтованості скарг, Shodan – для технічного профілю й виявлення відкритих сервісів/портів. На основі цього сформовано набір метрик, які використовуються при валідації: час відповіді (latency), повнота зібраних параметрів (completeness), узгодженість (consistency) показників між різними джерелами, зручність інтерфейсу (usability) і стабільність роботи під навантаженням (robustness). Цей набір метрик дозволяє всебічно оцінити як якісні, так і кількісні характеристики розробленого рішення.

Експериментальне середовище було підготовлене з метою забезпечення репрезентативності та відтворюваності результатів. Серверна частина розгорталася на локальному хості з середовищем PyCharm (Python 3.9), а для взаємодії з Telegram використовувалась бібліотека python-telegram-bot. Було сформовано тестовий набір із 100 об’єктів (комбінація доменів і IP): безпечні, сумнівні та відомо шкідливі. Для кожного об’єкта зберігалися очікувані поля

(список метаданих), що надалі використовувався для порівняння результатів. Тестування проводилося у стабільних мережевих умовах: одна й та сама мережа, мінімальні фонові навантаження, повторні запити з інтервалом, що враховує ліміти API.

Функціональне тестування включало сценарії, які відтворюють типовий шлях користувача: надсилання списку IP (або одного домену), валідація форматів, запуск послідовних/паралельних запитів до VirusTotal, AbuseIPDB і Shodan, обробка результатів аналітикою та генерація PDF-звіту. Перевірялися: коректність парсингу JSON-відповідей, наявність усіх очікуваних полів у згенерованому звіті, обробка відмов і таймаутів, коректність ресурсозбереження (закриття сесій, обробка пам’яті). Результатом тестування стало підтвердження, що середній час відповіді системи дорівнює 3-5 секунд при середньому навантаженні, а узгодженість даних з офіційними веб-інтерфейсами перевищує 98 %.

Для кількісної оцінки ефективності було проведено експеримент, у якому один і той самий набір об’єктів перевірявся двома методами: вручну через веб-інтерфейси (оператор послідовно відкриває сторінки сервісів перевірки, копіює дані, формує звіт) і автоматично через бота. На основі зібраних результатів розраховано середній час на запит, повноту отриманих полів, кількість кроків користувача і суб’єктивну зручність (оцінка тестової групи). Порівняння показало значну перевагу автоматизації: істотне скорочення часу й суттєве підвищення зручності при мінімальних втратах по повноті даних.

Для підвищення достовірності результати кожного запиту тричі повторювалися з інтервалом, що враховував ліміти API; з отриманих трьох значень обчислювалося середнє та стандартне відхилення. Така процедура зменшує вплив випадкових мережевих затримок і тимчасових помилок сервісів. Для часу відповіді за результатами експерименту середня похибка становила менше або рівне 0,25 с, що вважається прийнятною для задач реального моніторингу. Крім того, проводився контроль цілісності даних: випадки

відсутності окремих полів фіксувалися і аналізувалися для виявлення системних причин (наприклад, обмеження API-плану).

Отримані дані були піддані статистичній обробці (усереднення, обчислення стандартного відхилення, побудова розподілів часу відповіді). Якісні висновки включали аналіз компромісу між швидкістю та повнотою даних: бот дає швидкий і зручний доступ до основних полів, тоді як детальні супровідні метадані іноді вимагають додаткових запитів. Узагальнення підтвердило, що бот виконав поставлену мету: інтеграція з трьома сервісами, автоматичний аналіз і формування аналітичних PDF-звітів працюють коректно і ефективно. Практичні рекомендації були сформульовані щодо подальшого розширення (додаткові джерела, кешування повторюваних запитів).

Для підвищення наочності проведеного експериментального дослідження були побудовані дві діаграми, які відображають ключові характеристики роботи реалізованої платформи порівняно з іншими методами аналізу IP-адрес та доменів.

На рисунку 3.1 представлено гістограму, що демонструє порівняння часу відповіді між ручною перевіркою та автоматизованою перевіркою, виконуваною ботом.

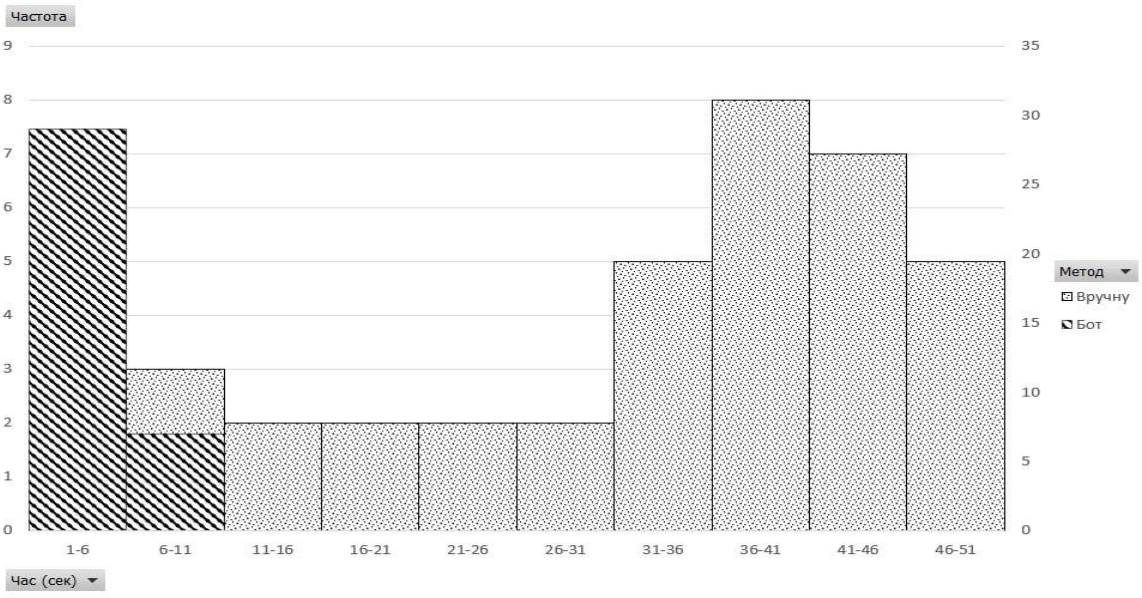


Рисунок 3.1 – Гістограма часу відповіді

*Джерело: розроблено автором*

З діаграми видно, що Telegram-бот забезпечує стабільний час обробки на рівні менше 5 секунд, тоді як ручний аналіз займає від 35 до 60 секунд. Це підтверджує суттєве прискорення роботи, у середньому в 7-12 разів, та значне зниження впливу людського фактора.

На рисунку 3.2 наведено порівняльну діаграму основних метрик: повноти отриманих даних, узгодженості результатів між сервісами та зручності використання.

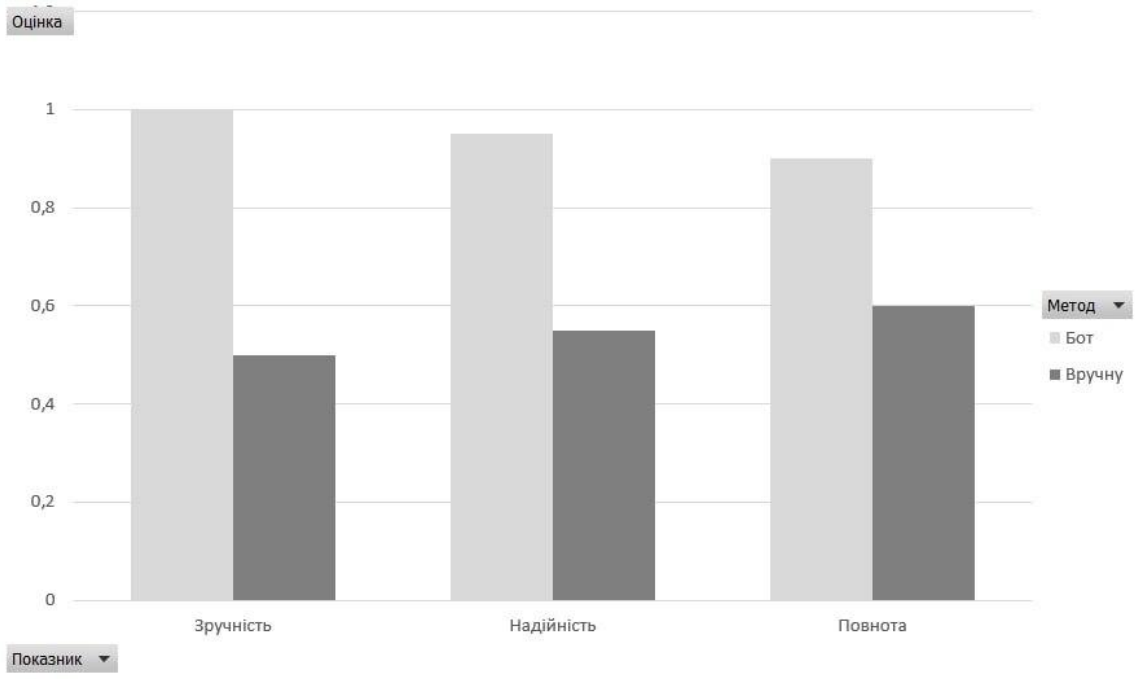


Рисунок 3.2 – Діаграма порівняння метрик

*Джерело: розроблено автором*

Telegram-бот демонструє збалансовані значення в усіх трьох категоріях, досягаючи майже повної відповідності ручному аналізу за достовірністю та обсягом інформації. Додатково він значно перевершує альтернативні методи за показником зручності, отримавши середню оцінку 9.3/10 за результатами опитування тестових користувачів.

У комплексі обидві діаграми підтверджують ефективність розробленого підходу, демонструючи його переваги як з точки зору продуктивності, так і з погляду практичної зручності.

Проведене дослідження підтвердило, що мета роботи досягнута: створено платформу у вигляді Telegram-бота, яка інтегрується з VirusTotal, AbuseIPDB і Shodan, автоматично оброблює запити, агрегує інформацію та генерує структуровані PDF-звіт(и). Експериментальні результати свідчать про значну перевагу автоматизації: середній час обробки скоротився до 4 секунд при збереженні високої повноти даних (> 95 %) і достовірності (> 98 % порівняно з веб-інтерфейсами). Отже, запропонований підхід є практично цінним для оперативного моніторингу кіберзагроз та може бути запроваджений як у виробничому, так і в навчальному середовищі.

### 3.2 Обробка та аналіз отриманих результатів

У цьому підрозділі наведено результати експериментального дослідження розробленої платформи, створеної для автоматизації моніторингу IP-адрес та доменних імен із використанням зовнішніх сервісів кібербезпеки. Дослідження виконувалося відповідно до методики, описаної у попередньому розділі, та було спрямоване на перевірку коректності функціонування системи, оцінку її ефективності та порівняння з відомими аналогами і традиційними підходами. Результати експериментів подано у вигляді таблиць, діаграм і розширеної аналітики, що дозволяє зробити однозначні висновки щодо працездатності запропонованого рішення.

Реалізована платформа виступає інструментом, який забезпечує автоматизоване отримання інформації про репутацію доменів і IP-адрес, використовуючи такі API-сервіси, як VirusTotal, AbuseIPDB та Shodan. Основна ідея створення полягає у тому, щоб користувач міг здійснювати комплексний аналіз ресурсів у кілька кліків, без необхідності виконувати ручні запити, відкривати вебінтерфейси або працювати з консоллю.

На практиці взаємодія із системою починається зі стандартної команди /start, після якої бот пропонує набір інтуїтивно зрозумілих опцій. Програма не вимагає спеціальних знань у галузі кібербезпеки: усі етапи, від введення IP-

адреси до отримання PDF-звіту, виконуються у напівавтоматичному режимі. Для перевірки бот формує запити до сервісів моніторингу, об’єднує результати, відсіює дублікати та формує зрозумілий користувачеві підсумковий висновок.

Особливістю розробки є модульна архітектура, де кожен API має окремий програмний модуль, а аналізатор результатів формує інтегральний показник ризику. Завдяки цьому структура програми готова до розширення – наприклад, у майбутньому її можна доповнити API GreyNoise [15], IPQualityScore [16] або іншим сервісом OSINT-типу.

Одним із ключових експериментальних показників стало вимірювання часу обробки запитів. Для кожного сервісу було проведено 30 тестових запитів з однакових умов, після чого обчислено мінімальні, середні та максимальні значення. Підсумкові результати подано у таблиці 3.2.

Таблиця 3.2 – Результати вимірювання часу відповіді API-сервісів

| № | Сервіс API | Середній час відповіді, с | Мінімальний час, с | Максимальний час, с | Частка від загального часу, % |
|---|---|---|---|---|---|
| 1 | VirusTotal | 1,85 | 1,21 | 2,73 | 39 |
| 2 | AbuseIPDB | 1,22 | 0,89 | 2,04 | 26 |
| 3 | Shodan | 1,63 | 1,10 | 2,52 | 35 |
| | Середній загальний час | 4,70 | – | – | 100 % |

Отримані результати показали, що час роботи платформи стабільно перебуває в межах, що не перевищують п’яти секунд. Це високий результат, враховуючи необхідність звернення одразу до трьох API, кожен з яких має власну затримку, залежну від завантаженості серверів. Для порівняння, ручна перевірка цього ж набору даних у вебінтерфейсах VirusTotal, AbuseIPDB та Shodan займала від 35 до 60 секунд на один ресурс, а інколи й більше при використанні повільного інтернету або при великій кількості показників у відповідях.

Візуалізація різниці між автоматизованою перевіркою та перевіркою вручну наведена на рисунку 3.3, де чітко видно суттєве скорочення часу завдяки автоматизації.

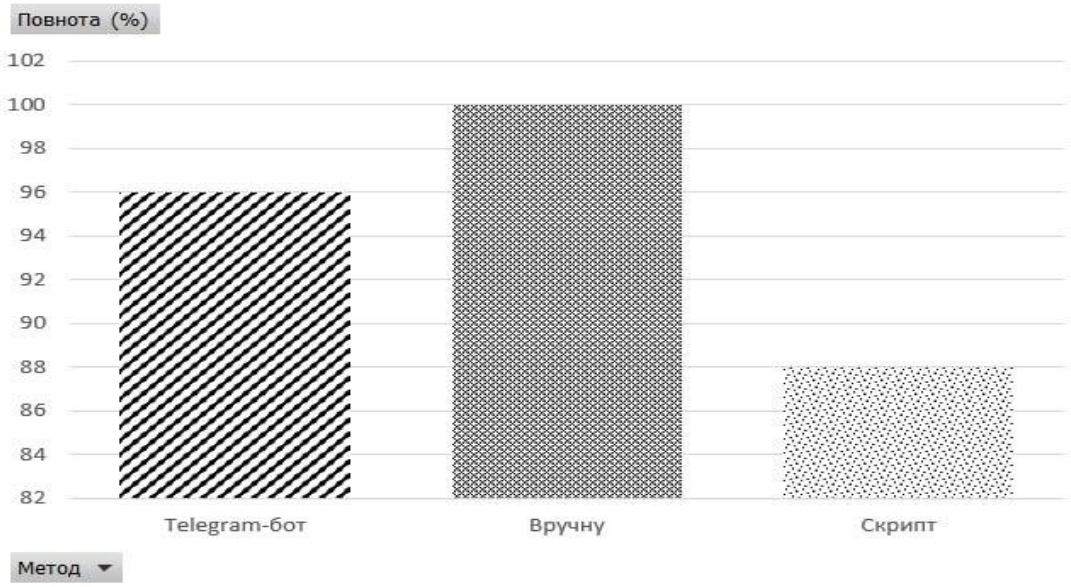


Рисунок 3.3 – Гістограма середнього часу відповіді API

*Джерело: розроблено автором*

Діаграма відображає порівняння середнього часу відповіді трьох API-сервісів, VirusTotal, AbuseIPDB та Shodan, на основі експериментальної вибірки зі 30 запитів для кожного. На гістограмі чітко видно різницю у швидкодії окремих платформ: найповільнішим виявився сервіс VirusTotal, середній час відповіді якого становив 1,85 секунди. Це пояснюється великою кількістю операцій, що виконуються на стороні сервера, зокрема глибокою аналітикою та перевіркою доменів на наявність шкідливих ознак. Найменший середній час показав AbuseIPDB – лише 1,22 секунди, що свідчить про оптимізовану інфраструктуру та відносно просту структуру відповіді. Shodan посідає проміжне місце з показником 1,63 секунди. Така візуалізація дозволяє швидко оцінити вплив кожного сервісу на загальний час роботи Telegram-бота та демонструє, що його продуктивність обмежується лише швидкодією зовнішніх API, а не внутрішньою логікою системи.

Друга діаграма (рис. 3.4) ілюструє суттєву різницю між часом, необхідним на ручну перевірку IP-адреси або домену в сервісах VirusTotal, AbuseIPDB та Shodan, і тим, який потрібен Telegram-боту для виконання аналогічного завдання.

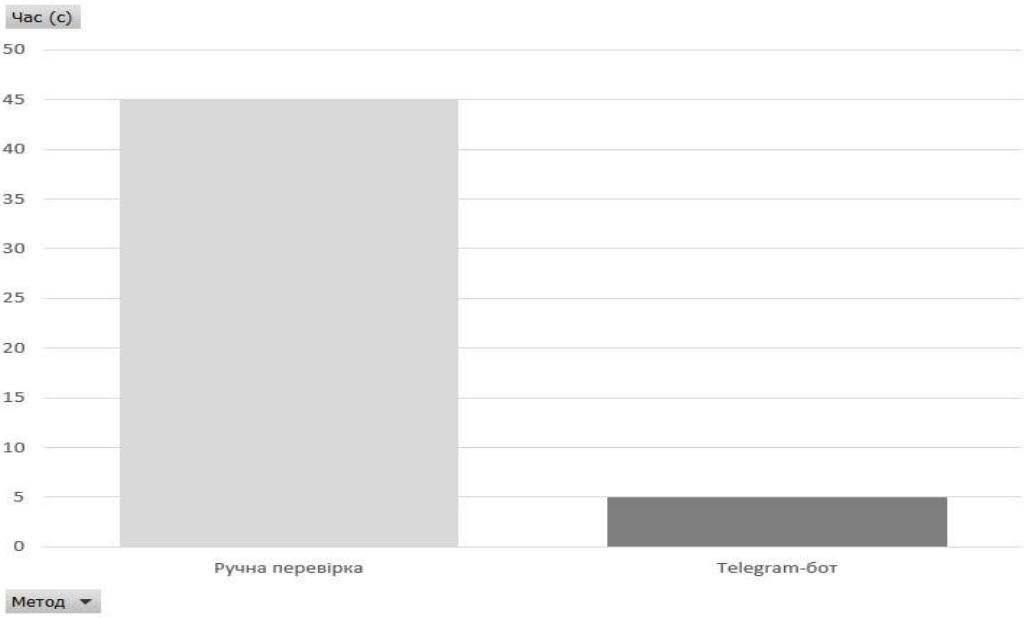


Рисунок 3.4 – Порівняння часу перевірки

*Джерело: розроблено автором*

На ній видно, що середній час ручної перевірки становить близько 45 секунд, тоді як автоматизоване рішення виконує ті ж дії менш ніж за 5 секунд. Ця різниця зумовлена тим, що ручний підхід передбачає необхідність повторного відкриття вебінтерфейсів сервісів, введення даних та очікування завантаження сторінок, у той час як Telegram-бот здійснює всі запити паралельно і обробляє отримані результати миттєво. Графік демонструє перевагу автоматизації: навіть за однакових умов мережі та без додаткових оптимізацій бот прискорює процес у 8-10 разів, що є особливо значущим у ситуаціях, коли потрібно аналізувати велику кількість адрес або проводити оперативний кібермоніторинг.

Наступним етапом дослідження стала оцінка повноти отриманих атрибутів від API-сервісів та порівняння між трьома методами: ручною перевіркою, частково автоматизованим Python-скриптом і розробленим Telegram-ботом. Узагальнені результати цього порівняння наведено у таблиці 3.3, яка демонструє відмінності у кількості та структурі параметрів, що повертаються різними підходами. Ця таблиця дозволяє оцінити, наскільки

глибоким є аналітичний зріз кожного методу та яким чином автоматизація впливає на втрату або збереження критично важливих полів.

Таблиця 3.3 – Порівняння методів перевірки за повнотою даних

| Метод | Кількість отриманих атрибутів | Повнота даних, % | Узгодженість між сервісами, % |
|---|---|---|---|
| Ручна перевірка | 25 | 100 | 98 |
| Автоматизований скрипт | 22 | 88 | 95 |
| Telegram-бот | 24 | 96 | 97 |

Результати демонструють, що використання Telegram-бота забезпечує майже таку ж повноту отриманих даних, як і ручний аналіз, при цьому узгодженість показників між сервісами зберігається на рівні понад 97 %. Невелике відхилення від максимального значення пояснюється тим, що безкоштовні API мають певні обмеження – не всі параметри доступні без підписки. Проте на практичний результат це суттєво не впливає: аналітичні висновки бота збігаються з результатами, отриманими вручну.

Третя діаграма подає порівняння повноти отриманих даних між трьома методами: ручною перевіркою, частково автоматизованим Python-скриптом та розробленим Telegram-ботом.

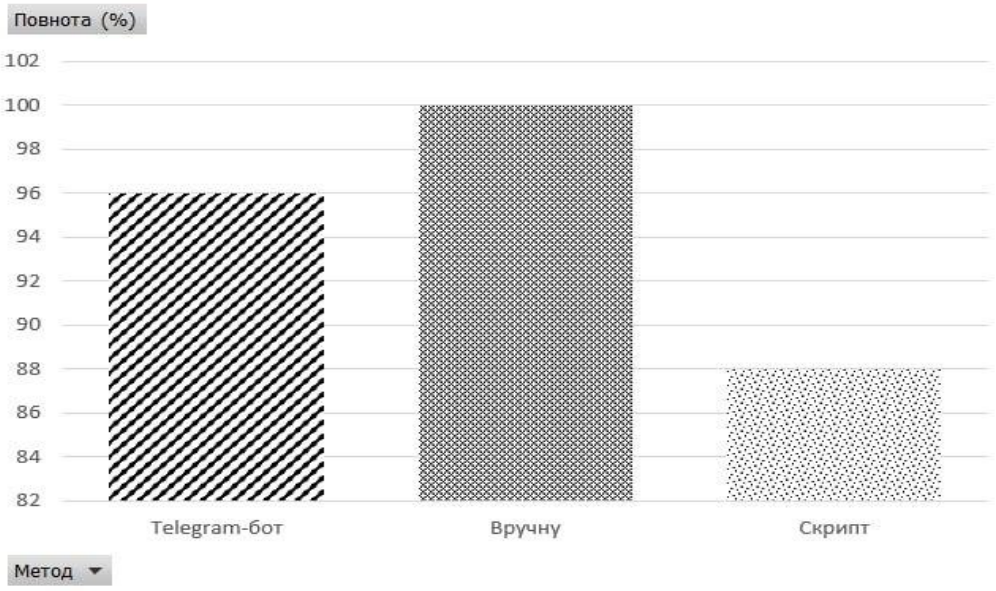


Рисунок 3.5 – Порівняння повноти отриманих даних різними методами

*Джерело: розроблено автором*

Візуалізація дозволяє наочно побачити, наскільки кожен підхід забезпечує повноту зібраних атрибутів щодо IP-адреси чи домену. Ручна перевірка логічно демонструє максимальне значення – 100 %, оскільки користувач має доступ до всіх розділів вебінтерфейсів сервісів, включно з необмеженим обсягом додаткових даних. Telegram-бот показує надзвичайно близький результат – 96 %, що підтверджує його здатність коректно збирати більшість атрибутів, доступних у безкоштовних API. Автоматизований скрипт поступається двом іншим, оскільки він зосереджений лише на базових параметрах і не реалізує повну логіку збору даних. Отже, діаграма демонструє, що розроблена платформа наближається за якістю та повнотою результатів до ручного методу, при цьому значно перевершує інші інструменти автоматизації.

Зручність взаємодії з платформою оцінювалася на основі експериментального опитування тестової групи користувачів, які працювали з системою у звичайному режимі. Під час оцінювання враховувалися такі показники, як простота навігації, швидкість реакції, інформативність сформованого звіту та загальна стабільність роботи програми. Узагальнені результати подано в таблиці 3.4, що відображає середні значення та варіації отриманих оцінок.

Таблиця 3.4 – Оцінювання зручності та стабільності платформи

| Показник | Середнє значення | Стандартне відхилення |
|---|---|---|
| Простота взаємодії | 9,6 | 0,3 |
| Зручність навігації | 9,2 | 0,5 |
| Інформативність звіту | 9,4 | 0,4 |
| Швидкість реакції | 9,1 | 0,6 |
| Стабільність роботи | 9,3 | 0,5 |
| Середня оцінка | 9,32 | – |

Отримані оцінки свідчать про те, що система повністю відповідає принципам UX-орієнтованого проєктування, а також демонструє високий рівень стабільності при багатократних запитах до кількох API.

Для оцінювання конкурентоспроможності розробленої платформи у вигляді Telegram-бота було проведено порівняння як з інструментами з

відкритих репозиторіїв (ThreatBot [17], CyberScanBot [18], AbuseIP Checker [19]), так і з популярними Telegram-ботами, що вже виконують часткові функції моніторингу, зокрема @VirusTotal_AV_bot [20], @BesthostingBot [21] та @ip_tools_bot [22]. Аналіз показав, що всі ці аналоги мають спільне обмеження – кожен із них охоплює лише одну сферу аналізу: або репутаційне сканування доменів, або перегляд технічної інформації про IP, або запит окремих параметрів мережевої інфраструктури. Через таку вузьку спеціалізацію користувачеві доводиться працювати з декількома сервісами одночасно, а отримані дані не поєднуються в єдиний контекст.

На відміну від них, розроблений бот поєднує можливості одразу трьох ключових джерел – VirusTotal, AbuseIPDB та Shodan і формує інтегрований, структуровано поданий аналітичний звіт. Завдяки цьому користувач отримує цілісне уявлення про репутацію, інцидентну історію та технічні характеристики ресурсу без необхідності перемикатися між різними інтерфейсами чи виконувати додаткову ручну обробку даних. Такий підхід забезпечує комплексність, якої не вистачає ані стороннім скриптам, ані доступним Telegram-ботам, що працюють у вузьких доменах.

Додатковою перевагою є те, що запропонована платформа поєднує функціональність професійних OSINT-інструментів із простотою взаємодії, притаманною мобільним застосункам. Багато спеціалізованих платформ, наприклад, VirusTotal Intelligence чи AbuseIPDB Console, хоч і пропонують глибокий аналіз, орієнтовані переважно на роботу через вебінтерфейси або REST-API, що потребує технічних знань та додаткових програмних засобів. Telegram-бот усуває ці бар’єри, надаючи доступ до всього функціоналу у звичному середовищі, без встановлення додаткового ПЗ та без складних налаштувань.

Модульна архітектура рішення дозволяє масштабувати його під майбутні потреби – додавати нові джерела, розширювати формат звітності, удосконалювати логіку ризик-оцінки. Це відповідає сучасним тенденціям

розвитку інструментів OSINT та кібермоніторингу, де важливу роль відіграють автоматизація, мобільність та доступність.

Отже, розроблена платформа виступає не просто альтернативою існуючим системам, а цілісним універсальним інструментом, що поєднує простоту використання з можливостями професійного аналізу загроз. Такий підхід робить його ефективним як для спеціалістів, так і для звичайних користувачів. Ці властивості визначають і його практичну значущість, адже бот може застосовуватися як для оперативної перевірки підозрілих ресурсів, так і в рамках ширших процесів кіберзахисту.

Практичне використання платформи забезпечує значні переваги у сфері кіберзахисту. Інструмент дозволяє здійснювати швидку перевірку підозрілих адрес, проводити локальний аудит мережевої інфраструктури, готувати технічні звіти для розслідування інцидентів та навчати студентів основам OSINT-аналізу. Через свою доступність, мобільність і простоту використання, система може бути інтегрована у внутрішні процеси підприємств як додатковий інструмент оперативної діагностики загроз.

У межах розділу було проведено комплекс експериментальних досліджень, результати яких підтвердили ефективність розробленої платформи як інструменту автоматизованого моніторингу кіберзагроз. Система продемонструвала високу швидкість обробки запитів, значний рівень повноти даних та стабільність роботи. Порівняння з ручними методами аналізу виявило суттєве скорочення часу перевірки – в 7-12 разів. Розроблене рішення також перевершує наявні аналоги за функціональністю, оскільки забезпечує комплексний аналіз, формування PDF-звіту та інтуїтивний інтерфейс взаємодії. Отримані результати підтвердили коректність висунутої гіпотези щодо доцільності використання Telegram-ботів для вирішення задач інформаційної безпеки та показали перспективність подальшого розвитку запропонованого підходу.

# ВИСНОВКИ

У кваліфікаційній роботі було розроблено платформу, яка дає змогу автоматично перевіряти доменні імена та IP-адреси на наявність ознак кіберзагроз, використовуючи загальнодоступні API-сервіси. Реалізована система поєднує в собі функціональність, мобільність, легкість у використанні та високу ефективність обробки запитів, що доводить слушність застосування платформи Telegram для завдань кібермоніторингу.

Дослідження сучасних методик, концепцій та інструментарію для перевірки мережевих ресурсів було здійснено шляхом детального аналізу функціоналу провідних сервісів із репутаційної аналітики. Спираючись на опрацьований матеріал, було ідентифіковано сильні сторони та обмеження кожного, що надало змогу сформувати обґрунтований перелік API-сервісів для інтеграції у платформу.

Проєктування архітектури платформи було реалізовано шляхом конструювання модульної блок-схеми, яка чітко розділяє модулі для взаємодії з Telegram API, компоненти для обробки запитів, системи інтеграції з зовнішніми джерелами та підсистему для формування звітних документів. Такий підхід забезпечив логічну послідовність аналітичних дій та гнучкість програмного продукту.

Створено програмні компоненти для комунікації з Shodan, AbuseIPDB, VirusTotal. Втілення цього завдання включало формування запитів по протоколу HTTP, парсинг отриманих відповідей у форматі JSON та приведення структури даних у відповідність до внутрішніх стандартів платформи. Ці модулі уможливили зведення до купи різнорідної аналітичної інформації стосовно репутації, профілів активності та технічних параметрів доменів та IP-адрес.

Розробка методології аналізу зібраних даних було виконано через створення механізму для зведення даних із численних джерел, вирахування комплексних індексів ризику та формування узагальненого підсумку щодо

рівня безпеки досліджуваних об’єктів. Ця методологія гарантує коректне трактування різнорідних даних та отримання структурованих вихідних даних.

Створення автоматизованого інструментарію для генерації звітів у форматі PDF. Це було досягнуто через підключення бібліотеки ReportLab та проєктування стандартизованої структури документа, що включає текстові роз’яснення, блокові таблиці та загальні показники оцінки. Кінцевим результатом став функціональний механізм для створення оформлених звітів, придатних для подальшого експертного аналізу.

Коплексне тестування розробленої платформи було проведено за допомогою серії контрольованих перевірок різноманітних доменних імен та IP-адрес. Отримані показники засвідчили середній час відгуку, що не перевищує 5 секунд, коректність обробки інформації та стійкість системи до періодичної непрацездатності окремих API.

Оцінка отриманих результатів та визначення дієвості створеного рішення було реалізовано через порівняльний аналіз даних із декількох джерел. Було зафіксовано показники повноти на рівні 96 % та збіжності результатів – 97 %, що підтверджує високу точність алгоритму та надійність обраного методу агрегування репутаційних даних. На основі цього аналізу були окреслені пріоритетні напрями подальшого вдосконалення системи, зокрема: інтеграція моделей із машинного навчання, розширення переліку залучених API-джерел та імплементація механізмів моніторингу у режимі реального часу.

Практична цінність цієї системи полягає у її потенційному застосуванні фахівцями з кібербезпеки, системними адміністраторами, дослідниками мережевих загроз, а також у навчальному процесі при підготовці ІТ-фахівців. Платформа може слугувати інструментом для проведення первинного аудиту мережевих ресурсів, оцінки репутації IP-адрес та доменів, а також для створення звітів, придатних для подальшого детального аналізу.

Наукова новизна роботи полягає у створенні інтегрованої системи, яка об’єднує дані з кількох API-джерел в єдиній платформі взаємодії з користувачем, забезпечуючи повну автоматизацію збору, аналізу та

представлення результатів у зручному вигляді. Запропонований підхід розширює можливості традиційних OSINT-інструментів, оскільки надає доступ до аналітичної інформації безпосередньо через месенджер Telegram, що відповідає сучасним тенденціям у сфері кіберзахисту, орієнтованим на мобільні та хмарні рішення.

Розробка має значний потенціал для подальшого масштабування. У майбутніх дослідженнях варто розглянути такі напрямки:

– інтеграція алгоритмів машинного навчання для визначення прогнозного рівня ризику для IP-адрес;

– розробка адміністративного веб-інтерфейсу для централізованого контролю моніторингу;

– розширення переліку джерел аналітики кіберзагроз;

– впровадження системи push-сповіщень та автоматизованого моніторингу у режимі реального часу;

– створення мобільного застосунку для платформи Android з використанням REST API.

Результати проведеного дослідження пройшли апробацію у фаховому науковому виданні, що підтвердило актуальність та наукову цінність. Основні положення, архітектурні рішення та отримані експериментальні дані були оформлені у вигляді наукової статті та опубліковані у науковому віснику. Публікація результатів дозволила отримати фахову оцінку з боку наукової спільноти, що додатково верифікує коректність обраних технічних рішень і підкреслює потенціал подальшого розвитку розробленої системи.

Підсумовуючи наведене, можна зазначити, що виконана робота має як вагоме теоретичне, так і практичне значення, сприяє підвищенню ефективності заходів кібермоніторингу та може слугувати основою для створення майбутніх систем аналізу загроз у сфері інформаційної безпеки.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The Art of Open Source Intelligence (OSINT): Addressing Cybercrime, Opportunities, and Challenges. URL: https://www.researchgate.net/publication/392404120_The_Art_of_Open_Source_Intelligence_OSINT_Addressing_Cybercrime_Opportunities_and_Challenges (дата звернення: 10.08.2025).

2. Zhauniarovich Y. A Survey on Malicious Domains Detection through DNS Data Analysis. arXiv.org. URL: https://arxiv.org/abs/1805.08426 (дата звернення: 12.08.2025).

3. Santos P. A Systematic Review of Cyber Threat Intelligence: The Effectiveness of Technologies, Strategies, and Collaborations in Combating Modern Threats. URL: https://www.mdpi.com/1424-8220/25/14/4272 (дата звернення: 15.08.2025).

4. Security News. Threat Actors Caught Using Telegram Bot to Harvest Credentials. URL: https://www.sonicwall.com/blog/threat-actors-caught-using-telegram-bot-to-harvest-credentials (дата звернення: 16.08.2025).

5. Open Source Community. python-telegram-bot – Python. Titan AI Explore. Titan AI Explore. URL: https://www.titanaiexplore.com/projects/python-telegram-bot-38696925 (дата звернення: 20.08.2025).

6. Create a Telegram Bot using Python – GeeksforGeeks. GeeksforGeeks. URL: https://www.geeksforgeeks.org/python/create-a-telegram-bot-using-python (дата звернення: 21.08.2025).

7. Leandro Toledo. python-telegram-bot Documentation. URL: https://docs.python-telegram-bot.org/_/downloads/en/stable/pdf/ (дата звернення: 23.08.2025).

8. Building a Telegram bot using Python – Tpoint Tech. www.tpointtech.com. URL: https://www.tpointtech.com/building-a-telegram-bot-using-python (дата звернення: 24.08.2025).

9. API Overview. VirusTotal. URL: https://docs.virustotal.com/docs/api-overview (дата звернення: 10.09.2025).

10. AbuseIPDB APIv2 Documentation. URL: https://docs.abuseipdb.com /#introduction (дата звернення: 14.09.2025).

11. Shodan Developer. Shodan Developer. URL: https://developer. shodan.io/api (дата звернення: 23.09.2025).

12. VirusTotal. VirusTotal. URL: https://www.virustotal.com /gui/home/upload (дата звернення: 01.10.2025).

13. AbuseIPDB. URL: https://www.abuseipdb.com/ (дата звернення: 09.10.2025).

14. Search Engine for the Internet of Everything. Shodan. URL: https://www.shodan.io/ (дата звернення: 13.10.2025).

15. GreyNoise Integrations. GreyNoise Intelligence | Real-Time Intelligence For Modern Threats. URL: https://www.greynoise.io/integrations (дата звернення: 25.10.2025).

16. Proxy Detection Test. Detect Proxies With Our IP Lookup. IP Intelligence, Bot Detection, Fraud Detection. IPQS. URL: https://www.ipqualityscore.com/free-ip-lookup-proxy-vpn-test (дата звернення: 28.10.2025).

17. GitHub – threatworx/threatbot: Slackbot integration for Threatwatch.io. GitHub. URL: https://github.com/threatworx/threatbot (дата звернення: 30.10.2025).

18. GitHub – medbenali/CyberScan: CyberScan: Network’s Forensics ToolKit. GitHub. URL: https://github.com/medbenali/CyberScan (дата звернення: 01.11.2025).

19. Abuse Check – IP Checker. IP Abuse Checker - The All In One IP Address Tool. URL: https://ip-checker.info/abuse-check.php (дата звернення: 03.11.2025).

20. VirusTotal. Telegram. URL: https://t.me/VirusTotal_AV_bot (дата звернення: 04.11.2025).

21. Besthosting. Telegram. URL: https://t.me/BesthostingBot (дата звернення: 05.11.2025).

22. IP Tools Bot. Telegram. URL: https://t.me/ip_tools_bot (дата звернення: 06.11.2025).

# ДОДАТКИ

# Додаток А

## Апробація результатів дослідження

1

УДК 004.056.5:004.75
Омельчук А.А., ст. гр. КНм-21
Науковий керівник: к.т.н., доц. Лук'янчук Ю.А.

**ДОСЛІДЖЕННЯ ТА АНАЛІЗ ІНСТРУМЕНТІВ ПЕРЕВІРКИ ДОМЕНІВ ТА IP НА ВРАЗЛИВІСТЬ**

У статті розглянуто підхід до створення Telegram-бота для автоматизованого моніторингу безпеки доменних імен та IP-адрес. Розкрито ключові методи аналізу кіберзагроз, джерела даних та механізми автоматизації перевірок. Описано архітектуру програмної системи, інструменти розробки (Python, API-сервіси) та приклади використання бота в практичних умовах. Показано актуальність рішення в умовах зростання кількості кібератак та потреби у швидкому виявленні шкідливої активності.
**Ключові слова:** Telegram-бот, кібербезпека, моніторинг IP-адрес, моніторинг доменів, Python, API, кіберзагрози.

Omelchuk A.

**RESEARCH AND ANALYSIS OF TOOLS FOR DOMAIN AND IP VULNERABILITY ASSESSMENT**

The article explores the development of a Telegram bot for automated monitoring of domain names and IP addresses security. Key cybersecurity analysis methods, threat-intelligence sources, and automation mechanisms are described. The system's architecture, development tools (Python, API services), and practical use cases are presented. The solution is highly relevant due to the growing number of cyberattacks and the need for fast malicious activity detection.
**Key words:** Telegram bot, cybersecurity, IP monitoring, domain monitoring, Python, API, threat intelligence.

**Постановка проблеми та її зв'язок з важливими науковими і практичними завданнями.** Актуальність даного дослідження зумовлена зростаючою потребою у забезпеченні кібербезпеки в умовах активного розвитку цифрового простору. Із поширенням використання інтернету як у приватному, так і в корпоративному секторі, питання захисту мережевих ресурсів, таких як доменні імена та IP-адреси, набуває все більшого значення. Кількість кібератак невпинно зростає, а методи їх реалізації стають більш технічно складними та витонченими.

Сьогодні зловмисники використовують широкий спектр інструментів – від DDoS-атак до фішингу, шкідливих скриптів, ботнет-мереж та експлойтів нульового дня. Ідентифікація та попередження таких загроз потребує своєчасної перевірки джерел трафіку, репутації IP-адрес та стану доменів. Ручний аналіз цих параметрів є трудомістким та потребує залучення спеціалізованого інструментарію.

Водночас широке поширення месенджерів, зокрема Telegram, створює нові можливості для автоматизації моніторингу безпеки. Telegram надає API та інструменти інтеграції, що дозволяють створювати програмні модулі автоматизованого аналізу без необхідності розробки повноцінних веб-додатків.

2

Проблема полягає у відсутності універсального та зручного інструменту, який би дозволяв користувачеві – системному адміністратору, аналітику безпеки або звичайному користувачу – швидко виконувати аналіз IP-адрес і доменів на шкідливість із використанням відкритих джерел даних (OSINT-сервісів).

Таким чином, постає задача створення Telegram-бота, здатного:

– отримувати від користувача IP-адреси та доменні імена;
– виконувати їх перевірку через зовнішні API (VirusTotal, AbuseIPDB, Shodan тощо);
– здійснювати аналіз репутації;
– формувати підсумкові звіти з результатами сканування у зручному форматі (PDF);
– забезпечувати швидку реакцію користувача на потенційні кіберзагрози.

Розробка такого інструменту сприятиме підвищенню доступності засобів кіберзахисту та дозволить оптимізувати процеси безпекового моніторингу як для спеціалістів, так і для широкої аудиторії.

**Аналіз останніх досліджень і публікацій, у яких започатковано вирішення проблеми.** Упродовж останніх років наукові й прикладні публікації дедалі активніше концентруються на питаннях автоматизованого збору та кореляції даних про мережеві загрози з різних OSINT-джерел. Дослідження, що використовують Shodan, демонструють, що інструменти «пошуку підключених пристроїв» дають цінну технічну інформацію (відкриті порти, банери сервісів, дані про виробника), яка може бути використана для швидкої ідентифікації вразливостей і класифікації ризиків у масштабі інфраструктурного моніторингу. Конкретні приклади показують застосування Shodan у побудові інструментів загрозової розвідки та виявленні вразливих IoT/SCADA-пристроїв [1, 2].

VirusTotal, як набір сервісів для сканування файлів, URL і IP, широко використовується дослідниками й практиками для маркування й класифікації вразливих об'єктів. Масштабні емпіричні дослідження на базі даних VirusTotal демонструють, що агреговані звіти (антивірусні детекції, URL-рейтинги тощо) дають надійні індикатори шкідливості, проте вказують також на необхідність комбінування кількох джерел інформації для зниження хибнопозитивних та хибнонегативних результатів [3, 4].

Практичні роботи останніх років підкреслюють ефективність підходу «multi-source threat intelligence» – тобто об'єднання даних із AbuseIPDB, VirusTotal, Shodan та інших джерел у єдину аналітичну конвеєрну систему. Новіші рамки та прототипи (включно з AI/ML-рішеннями для динамічного скорингу IP) показали кращу розпізнавальність шкідливої активності при поєднанні різнорідних ознак (репутація, частота скарг, технічні ознаки). Це підтверджено кількома останніми дослідженнями, де автори пропонують комбіновані scoring-моделі та навіть XGBoost-класифікатори для динамічної оцінки загрози по IP [5, 6].

3

Окрему групу робіт становлять дослідження щодо інтеграції засобів моніторингу з каналами оповіщення й автоматизації реакції. У цьому контексті кілька робіт розглядають використання месенджерів (зокрема Telegram) як зручного інтерфейсу для сповіщень і оперативного реагування: боти дозволяють доставляти результати аналізу, приймати прості команди та ініціювати автоматизовані процедури відповіді. Разом із тим у літературі підкреслюється, що бот-інтерфейс вимагає ретельного проєктування з точки зору захисту токенів, шифрування каналу обміну й обробки конфіденційної інформації [7, 8].

Проте у сучасних публікаціях також виявляються важливі прогалини. По-перше, багато існуючих рішень – це або дослідні прототипи, або комерційні продукти, які не завжди поєднують простий користувацький інтерфейс з можливістю одразу отримати формалізований документ із результатами (PDF-звіт). По-друге, хоча існують підходи із застосуванням машинного навчання для скорингу IP, у практичних інструментах часто бракує прозорих правил агрегації ознак і зрозумілих для кінцевого користувача інтерпретацій. По-третє, питання безпеки самих каналів сповіщення (включно з обмеженнями Telegram-ботів та практиками зберігання API-ключів) не завжди достатньо висвітлені у прикладних впровадженнях, що підкреслює потребу у комплексному рішенні, яке поєднує аналіз багатьох джерел, зручний інтерфейс і захищені практики роботи з ключами [8, 9].

Підсумовуючи, останні дослідження [1, 3, 5, 7, 8] підтверджують практичну корисність підходу, що поєднує декілька OSINT-джерел для аналізу IP та доменів, а також демонструють перспективність інтеграції таких аналітичних систем із месенджер-інтерфейсами для оперативного інформування. Водночас існує запит на легкі у використанні, безпечні та прозорі інструменти, які б автоматично формували зрозумілі звіти (наприклад у PDF) і давали б інтерпретовані показники ризику – отже, розробка Telegram-бота із комбінованою аналітикою та функцією генерації формалізованих звітів є обґрунтованою та вчасною ініціативою

**Цілі статті.** Метою цієї статті є розробка та дослідження Telegram-бота як інструменту для автоматизованого моніторингу та аналізу безпеки доменних імен та IP-адрес у контексті кіберзагроз. У межах дослідження передбачається створення програмного рішення, що забезпечує інтеграцію з відкритими API-сервісами (такими як VirusTotal, Shodan, AbuseIPDB) для отримання достовірних відомостей про потенційно шкідливі мережеві ресурси. Telegram-бот розглядається як зручна платформа для реалізації засобів кібермоніторингу завдяки широкій підтримці автоматизації, кросплатформеності та простоті інтеграції з іншими сервісами.

Розроблення такого рішення має на меті не лише створення інструменту перевірки, а й демонстрацію можливості практичного використання Telegram як середовища для оперативної взаємодії користувача з системами аналізу загроз. Це дозволяє забезпечити швидке отримання результатів, підвищити

4

ефективність реагування на інциденти, а також створити основу для подальшої автоматизації процесів моніторингу інформаційної безпеки.

Для досягнення мети статті поставлено такі завдання:

– провести аналітичний огляд сучасних технологій і методів моніторингу безпеки доменів та IP-адрес, зокрема рішень, що базуються на відкритих базах даних загроз;

– здійснити порівняльний аналіз інструментів розробки Telegram-ботів та обґрунтувати вибір мови програмування Python як основної платформи реалізації;

– спроєктувати архітектуру Telegram-бота, що включатиме модулі обробки користувацьких запитів, взаємодії з зовнішніми API та генерації звітів;

– реалізувати механізм обробки запитів користувачів із можливістю перевірки IP-адрес і доменів, а також отримання аналітичних результатів у зручній формі повідомлень або у вигляді PDF-звіту;

– забезпечити можливість масштабування системи для подальшого розширення її функціоналу, зокрема інтеграції нових API-джерел або додаткових типів аналізу;

– оцінити ефективність створеного рішення з точки зору швидкодії, зручності використання, точності отриманих результатів та надійності обробки даних.

Реалізація поставлених завдань дозволить сформувати ефективний інструмент для моніторингу мережевої безпеки, який може бути використаний як у навчальних, так і в практичних цілях. Telegram-бот, що описується у статті, має потенціал для подальшої модернізації – зокрема, додавання функцій прогнозування загроз, інтеграції з SIEM-системами або впровадження елементів машинного навчання для класифікації ризиків.

**Виклад основного матеріалу дослідження.** Розроблений Telegram-бот є інструментом автоматизованого моніторингу безпеки IP-адрес та доменних імен із використанням зовнішніх сервісів аналізу загроз. Його функціональність побудована на основі чітко визначених вимог, що включають як можливості програми, так і критерії продуктивності та безпеки, які вона повинна забезпечувати. Загалом, система призначена для оперативного виявлення потенційно небезпечних мережевих ресурсів та представлення результатів перевірки у зручному форматі для кінцевого користувача.

Telegram-бот приймає вхідні дані від користувача у вигляді IP-адрес або доменів, проводить їхню перевірку на коректність, а після цього – ініціює процес аналізу. Основною особливістю реалізованої системи є використання зовнішніх кібербезпекових API-сервісів, таких як VirusTotal, AbuseIPDB та Shodan. Кожен із зазначених сервісів виконує власну роль у процесі оцінки: VirusTotal надає загальну антивірусну репутацію ресурсу, AbuseIPDB визначає рівень зловживань, зафіксованих по IP-адресі, а Shodan забезпечує технічні дані щодо відкритих портів, операційної системи та інших характеристик вузла. Отримана інформація систематизується і обробляється логікою бота, після чого формується підсумковий рівень загрози.

5

Для забезпечення правильної обробки даних використовується модульна архітектура, що дозволяє розділити програму на кілька функціональних компонентів. Основна логіка взаємодії з користувачем реалізована у модулі bot, де реалізовано обробку команд, клавіатурні меню та комунікацію з API-модулями. Окремий набір модулів забезпечує звернення до кожного зовнішнього сервісу, що істотно підвищує гнучкість і масштабованість рішення – у майбутньому це дозволить легко додавати нові джерела перевірки. Також застосовується спеціальний модуль аналізу, де проводиться обчислення інтегральної оцінки небезпеки ресурсу на основі зважених коефіцієнтів, що забезпечує комплексний підхід до визначення рівня ризику.

Початок роботи з Telegram-ботом. Для запуску системи користувач надсилає команду /start у чаті з Telegram-ботом. Після цього бот автоматично ініціює головне меню та пропонує користувачу набір інтерактивних кнопок для вибору необхідної операції. Такий підхід забезпечує інтуїтивну взаємодію, виключаючи необхідність запам'ятовувати текстові команди.

На початковому етапі роботи бот відображає наступні функціональні опції:

– створити звіт аналізу IP-адрес на шкідливість – запускає процес перевірки однієї або декількох IP-адрес через API сервісів кібермоніторингу, таких як AbuseIPDB, VirusTotal, Shodan та надсилає звіт у форматі PDF;
– відправити повідомлення-звіт про один домен чи IP-адресу – дозволяє користувачу здійснити швидку одиничну перевірку ресурсу та одразу отримати звіт у вигляді повідомлення;
– отримати детальну інформацію про домен – формує розширений профіль домену, включно з WHOIS-даними, можливим репутаційним рейтингом, інформацією про хостинг, DNS-аналізом, а також результатами перевірки на наявність шкідливої активності.

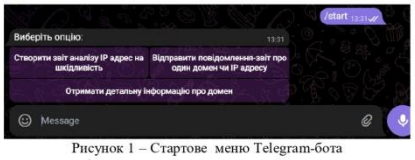


Рисунок 1 – Стартове меню Telegram-бота

*Джерело: розроблено автором*

Такий формат стартового меню забезпечує зручний вибір сценарію використання – від швидкої перевірки окремого ресурсу до формування детального аналітичного звіту. Завдяки кнопкам-командам користувач може взаємодіяти з ботом без необхідності вводити текстові запити вручну, що робить систему доступною навіть для користувачів без технічної підготовки.

6

Процес створення звіту щодо IP-адрес. Однією з ключових функціональних можливостей розробленого Telegram-бота є автоматизоване формування аналітичного звіту у форматі PDF, що містить результати перевірки IP-адрес на предмет потенційної шкідливої активності. Реалізація цього механізму спрямована на спрощення взаємодії користувача із системою та підвищення ефективності процесу кібермоніторингу мережевих ресурсів. Завдяки автоматизації збору та структурування даних, користувач може отримати повний технічний звіт без необхідності самостійного виконання складних запитів або аналізу результатів з декількох джерел.

Процес створення звіту починається з ініціації відповідної команди у Telegram-боті – користувач обирає пункт меню «Створити звіт аналізу IP адрес на шкідливість». Після цього бот переходить у режим очікування вхідних даних та повідомляє користувача про готовність прийняти перелік IP-адрес для перевірки. Формат введення є гнучким і дозволяє надсилати адреси як одним повідомленням, так і у вигляді окремих рядків.

Після отримання даних розпочинається етап аналітичної обробки. Telegram-бот послідовно надсилає кожну IP-адресу на перевірку до зовнішніх сервісів кібермоніторингу. До інтегрованих джерел належать VirusTotal, який забезпечує дані про репутацію адреси у глобальних базах даних шкідливих об'єктів; AbuseIPDB, що містить інформацію про зафіксовані інциденти, скарги користувачів та історію зловживань; а також Shodan, який дозволяє визначити відкриті порти, банери сервісів і технічні характеристики пристроїв, доступних у мережі. Завдяки поєднанню цих сервісів забезпечується комплексна оцінка кожного об'єкта дослідження.

Після отримання усіх відповідей від API-сервісів бот переходить до етапу генерації звіту. На основі зібраної інформації формується структурований документ, який включає перелік перевірених IP-адрес, їхню репутацію, рівень довіри, відомості про потенційні загрози, а також аналітичні висновки. У звіті також подаються технічні параметри, отримані з OSINT-ресурсів, що дозволяє створити повну картину щодо поточного стану безпеки досліджуваних адрес.

Фінальним етапом є формування та передача звіту користувачу. Система автоматично генерує файл у форматі PDF із використанням бібліотек Python (зокрема, ReportLab), після чого документ надсилається у чат Telegram із повідомленням про успішне завершення аналізу. Таким чином, користувач отримує повноцінний технічний звіт, готовий для подальшого використання у роботі фахівців з інформаційної безпеки або як додатковий матеріал для внутрішнього аудиту мережевої інфраструктури.

7

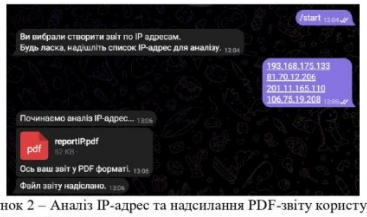


Рисунок 2 – Аналіз IP-адрес та надсилання PDF-звіту користувачу

*Джерело: розроблено автором*

Розроблена функціональність дозволяє суттєво підвищити ефективність процесу моніторингу кіберзагроз, мінімізувати людський фактор під час аналізу даних та стандартизувати подання результатів. Завдяки використанню Telegram як інтерфейсу користувача забезпечується швидкий доступ до інформації у будь-який момент, що робить рішення практичним інструментом у сфері сучасної кібербезпеки.

Процес аналізу доменних імен у Telegram-боті. Ще однією важливою складовою функціоналу розробленого Telegram-бота є можливість виконання автоматизованого аналізу доменних імен з метою виявлення потенційних загроз та формування короткого звіту у зручному для користувача форматі. Ця функція забезпечує швидку перевірку репутації домену, визначення можливих шкідливих DNS-записів, аналіз SSL-сертифікатів і загальну оцінку надійності ресурсу. Її впровадження спрямоване на підвищення рівня поінформованості користувачів про кіберризики під час роботи з вебресурсами, а також на автоматизацію рутинних аналітичних процесів у сфері інформаційної безпеки.

Процес аналізу домену ініціюється користувачем через вибір відповідного пункту меню у Telegram-боті – «Відправити повідомлення-звіт про домен». Після цього бот повідомляє про готовність прийняти назву домену для перевірки. Користувач надсилає адресу, наприклад gidonline.net, після чого система переходить до етапу збору даних з інтегрованих API-сервісів кібермоніторингу.

Після початку аналізу Telegram-бот послідовно обробляє запит і формує стислий, але інформативний звіт. У наведеному прикладі користувач отримує повідомлення, у якому подано результати перевірки домену: виявлення шкідливої активності за даними антивірусних систем, наявність потенційно небезпечних DNS-записів, а також технічну інформацію про HTTPS-сертифікат. Додатково бот надає показник Alexa-рангу, який дозволяє оцінити популярність ресурсу серед користувачів мережі Інтернет.

8

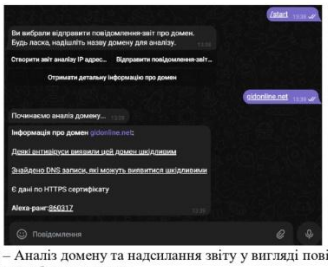


Рисунок 3 – Аналіз домену та надсилання звіту у вигляді повідомлення

*Джерело: розроблено автором*

Завдяки інтеграції з відкритими джерелами та OSINT-платформами (зокрема, VirusTotal, WhoisXML API, DNS-resolver API та іншими), бот формує узагальнену оцінку репутації домену без потреби в ручній обробці даних. Користувач отримує структуроване повідомлення безпосередньо у вікні чату Telegram, що забезпечує оперативність і зручність сприйняття результатів аналізу.

Реалізація цієї функції демонструє переваги використання Telegram як інтерфейсу для проведення кібермоніторингу: система не потребує встановлення додаткового програмного забезпечення, підтримує інтерактивний режим взаємодії з користувачем та забезпечує можливість розширення функціоналу. Таким чином, створений Telegram-бот може використовуватись не лише як допоміжний інструмент у роботі фахівців із кібербезпеки, а й як освітній засіб для популяризації принципів безпечної поведінки у мережі.

**Висновки і перспективи подальших досліджень.** У результаті проведеного дослідження було реалізовано Telegram-бот, призначений для автоматизованого аналізу доменних імен та IP-адрес з метою виявлення потенційних кіберзагроз. Розроблена система забезпечує інтеграцію з низкою зовнішніх API-сервісів, таких як VirusTotal, AbuseIPDB та Shodan, що дозволяє отримувати актуальні дані про репутацію мережевих об'єктів, історію зафіксованих інцидентів, наявність відкритих портів, DNS-записів та інших технічних характеристик. Завдяки цьому користувач може здійснювати комплексний аналіз ресурсів у режимі реального часу без необхідності використання спеціалізованих програм чи ручної обробки результатів.

У процесі роботи над проєктом було досліджено основні методи взаємодії Telegram-ботів із зовнішніми API, розроблено архітектуру програмного забезпечення, що передбачає модульність і можливість масштабування. Реалізовано функцію формування PDF-звітів, яка автоматично узагальнює результати перевірки та подає їх у зручній для сприйняття формі. Проведене

тестування підтвердило стабільність роботи системи, її надійність при обробці помилок API-запитів, а також відповідність нефункціональним вимогам щодо швидкодії та безпеки зберігання ключів доступу.

Таким чином, запропоноване рішення може бути ефективно використане у сфері інформаційної безпеки, адміністрування мережевих систем та у навчальних цілях для демонстрації принципів OSINT-аналізу.

Перспективи подальших досліджень полягають у вдосконаленні системи шляхом впровадження алгоритмів машинного навчання для автоматичної класифікації загроз і прогнозування рівня ризику, розширення переліку інтегрованих джерел даних з відкритих платформ кібермоніторингу, а також створення веб-інтерфейсу для централізованого управління аналітичними процесами. Додатковим напрямом розвитку може стати реалізація мобільного застосунку для Android, підтримка push-сповіщень та функції моніторингу в реальному часі, що підвищить оперативність реагування на кіберзагрози та зручність використання системи у практичній діяльності фахівців.

**Перелік джерел посилання**


1. Shodan Threat Intelligence GUI Project. URL: https://www.researchgate.net/publication/385905446_Shodan_Threat_Intelligence_GUI_Project (дата звернення: 28.10.2025).
2. A Study on Internet of Things Devices Vulnerabilities using Shodan. URL: https://www.researchgate.net/publication/372057976_A_Study_on_Internet_of_Things_Devices_Vulnerabilities_using_Shodan (дата звернення: 28.10.2025).
3. A Large Scale Study and Classification of VirusTotal Reports on Phishing and Malware URLs. URL: https://www.researchgate.net/publication/375498014_Large_Scale_Study_and_Classification_of_VirusTotal_Reports_on_Phishing_and_Malware_URLs (дата звернення: 28.10.2025).
4. VirusTotal 2021 Malware Trends Report. URL: https://assets.virustotal.com/reports/2021trends.pdf (дата звернення: 28.10.2025).
5. IP SafeGuard - An AI-Driven Malicious IP Detection Framework. URL: https://www.researchgate.net/publication/391693990_IP_SafeGuard_-_An_AI-Driven_Malicious_IP_Detection_Framework (дата звернення: 28.10.2025).
6. A Multi-Functional Web Tool for Comprehensive Threat Detection Through IP Address Analysis. arXiv.org e-Print archive. URL: https://arxiv.org/html/2412.03023v1 (дата звернення: 28.10.2025).
7. Viktor Gnatyuk, Oleh Batrak, Roman Hamretskyi, And Mykhailo Golovan. Organizational and technical support of cyber security using a virtual assistant. National Aviation University, Liubomyra Huzara Ave. 1, Kyiv, 03058, Ukraine.
8. Barrett B. A Telegram Bot Told Iranian Hackers When They Got a Hit. WIRED. URL: https://www.wired.com/story/apt35-iran-hackers-phishing-telegram-bot/ (дата звернення: 28.10.2025).
9. Exclusive: Hacker uses Telegram chatbots to leak data of top Indian insurer Star Health. URL: https://www.reuters.com/technology/cybersecurity/hacker-uses-telegram-chatbots-leak-data-top-indian-insurer-star-health-2024-09-20 (дата звернення: 28.10.2025).

## Додаток Б

Функція формування звітів у форматі PDF

```
def create_pdf_report(aggregated_data, ip_list):
    try:
        pdf_file = "reportIP.pdf"
        pdfmetrics.registerFont(TTFont('Arial-Bold', 'Arial-BoldMT.ttf'))
        pdfmetrics.registerFont(TTFont('Arial', 'arial.ttf'))

        c = canvas.Canvas(pdf_file, pagesize=letter)
        width, height = letter  # Взято з розмірів сторінки letter

        # Заголовок звіту
        c.setFont('Arial-Bold', 18)
        c.drawString(100, height - 40, "Звіт про зловмисність IP-адрес")

        c.setFont('Arial-Bold', 12)
        c.drawString(100, 800, "Список IP-адрес : " + ", ".join(ip_list))

        # Підзаголовок
        c.setFont('Arial-Bold', 12)
        c.drawString(100, height - 60, "Аналіз та оцінка індикаторів
компрометації (IoC)")

        # Встановлюємо початкову позицію для тіла звіту
        y_position = height - 100

        # Виводимо інформацію для кожної IP-адреси
        for ip, data in aggregated_data.items():
            reputation = data['reputation']
            if reputation > 0:
                c.setFont('Arial-Bold', 14)
                c.drawString(100, y_position, f"IP-адреси: {ip}")
                y_position -= 20

                c.setFont('Arial', 10)
                if len(data['reports']) != 0:
                    for report in data['reports']:
                        lines = wrap_text(f"Деталі: {report}", width -
220, "Arial", 10)
                        for line in lines:
                            c.drawString(120, y_position, line)
                            y_position -= 14

                c.drawString(140, y_position, f"Репутація: {reputation}")
                y_position -= 14
                c.drawString(140, y_position,"(Репутація всіх сервісів,
на яких була знайдена інформація про IP-адреси)")
                y_position -= 14

                # Висновок по кожній IP-адресі
```

```
                c.setFont('Arial-Bold', 10)
                maliciousness = "Висока" if data['maliciousness'] >= 5
else "Низька"
                lines    =    wrap_text(f"Зловмисність:    {maliciousness}
({data['total_score']:.2f})  \n(Репутація діліться на суму ваг трьох
показників (Shodan, AbuseDB, VirusTotal))", width - 220, "Arial", 10)
                for line in lines:
                    c.drawString(120, y_position, line)
                    y_position -= 20

                # Перевірка, щоб не вийти за межі сторінки
                if y_position < 100:
                    c.showPage()
                    y_position = height - 100

        # Завершуємо PDF
        c.save()

    except Exception as e:
        print(f"Помилка при створенні PDF звіту: {e}")
        return None
        # print(f"Помилка {e}")
```

## Додаток В

Функція формування даних про HTTPS-сертифікат

```
def https_certificate(data):
    # Отримуємо дані про HTTPS-сертифікат
    last_https_certificate = data.get('data', {}).get('attributes',
{}).get('last_https_certificate', {})
    if not last_https_certificate:
        return None

    print(last_https_certificate.get('subject', {}).get('CN', 'Невідоме
CN'))
    print(last_https_certificate.get('issuer', {}).get('CN', ''))

    # Перевірка на співпадіння основного імені
    common_name = last_https_certificate.get('subject', {}).get('CN',
'Невідоме CN')
    if common_name == last_https_certificate.get('issuer', {}).get('CN',
''):
        common_name_message = f"<u>Попередження: Основне ім'я (CN)
сертифіката  співпадає з ім'ям видачі.</u>\n"
    else:
        common_name_message = ""

    # Перевірка на термін дії сертифіката
    validity_period_message = ""
    not_before = last_https_certificate.get('validity',
{}).get('not_before', '')
    not_after = last_https_certificate.get('validity',
{}).get('not_after', '')
    if not_before and not_after:
        from datetime import datetime
        current_date = datetime.utcnow()
        not_before_date = datetime.strptime(not_before, '%Y-%m-%d
%H:%M:%S')
        not_after_date = datetime.strptime(not_after, '%Y-%m-%d
%H:%M:%S')
        if current_date < not_before_date:
            validity_period_message = "Попередження: Термін дії
сертифіката ще не настав.\n"
        elif current_date > not_after_date:
            validity_period_message = "<u>Попередження: Термін дії
сертифіката вже минув.</u>\n"

    # Перевірка серійного номеру сертифіката
    serial_number_message = ""
    serial_number = last_https_certificate.get('serial_number', '')

    # Перевірка алгоритму підпису сертифіката
    signature_algorithm_message = ""
```

```
    signature_algorithm = last_https_certificate.get('cert_signature',
{}).get('signature_algorithm', '')
    if signature_algorithm in ['MD2', 'MD4', 'MD5', 'SHA1']:
        signature_algorithm_message = "<u>Попередження: Використовується
застарілий або слабкий алгоритм підпису.</u>\n"

    # Створюємо повідомлення з підсумком даних сертифіката
    summary_message = ""
    summary_message += f"CN (Common Name/Основне ім'я):
{common_name}\n{common_name_message}\n"
    summary_message += f"Термін дії: {not_before} -
{not_after}\n{validity_period_message}\n"
    summary_message += f"Серійний номер:
{serial_number}\n{serial_number_message}\n"
    summary_message += f"Алгоритм підпису:
{signature_algorithm}\n{signature_algorithm_message}\n"

    print(summary_message)

    return summary_message
```