УДК 004.056.52:004.77

СИСТЕМА МОНІТОРИНГУ КОМПРОМЕТАЦІЇ ОБЛІКОВИХ ДАНИХ

Гетьман І.І.
getman_II@gmail.com
Черкаський державний фаховий бізнес-коледж
Медолиз М.М.
м.Черкаси, Україна

Актуальність проблеми витоку облікових даних у відкритих джерелах зумовлена постійним збільшенням кількості інтернет-сервісів, накопиченням персональної інформації, поширенням кіберзлочинності та використанням компрометованих даних у шахрайських схемах. Витік облікових даних призводить до фінансових втрат користувачів, компрометації корпоративних систем, втрати конфіденційності та зниження довіри до цифрових сервісів [1]. Аналіз сучасних методів моніторингу дозволяє класифікувати джерела витоків на публічні бази даних, форуми, сайти з витоками, а також платні та безкоштовні REST API сервіси перевірки компрометацій [2].

Функціональна архітектура системи, що розробляється, базується на високоефективній мові програмування Python, що забезпечує стабільність обчислювальних процесів та сучасну підтримку асинхронності. Взаємодія з користувачем реалізована через інтерактивний інтерфейс Telegram-бота, побудований на базі фреймворку Aiogram, який дозволяє гнучко опрацьовувати події в режимі реального часу. Для забезпечення цілісності обміну даними із зовнішніми ресурсами інтегровано бібліотеку requests, яка виконує роль сполучного механізму з різноманітними REST API сервісами.

Програмна структура системи спроєктована за модульним принципом, що дозволяє досягти високого рівня автономності окремих елементів та спрощує подальше масштабування продукту. Зокрема, архітектура передбачає чіткий поділ на рівні логіки, де виокремлюються блоки обробки вхідних повідомлень, модулі формування запитів до зовнішніх серверів та компоненти валідації отриманих результатів. Такий підхід не лише підвищує відмовостійкість системи за рахунок ізоляції можливих помилок у межах окремих модулів,а й дозволяє

інтегрувати нову функціональність шляхом розширення існуючих модулів без ризику порушення стабільної роботи фундаментальних алгоритмів. Система складається з наступних модулів:

1. Модуль збору даних – забезпечує регулярне сканування відкритих джерел, включаючи форуми, сайти з витоками та API сервісів; виконує парсинг та структурування отриманої інформації; застосовує фільтри для виділення релевантних даних.
2. Модуль обробки даних – перевіряє коректність отриманих даних, визначає тип витоку, ступінь ризику та пріоритет оповіщення; реалізує алгоритми валідації та нормалізації даних, включаючи перевірку формату логінів та хешів паролів.
3. Модуль логування та аудиту – фіксує всі події системи, веде історію перевірок, зберігає статистику ефективності та точності моніторингу; підтримує інтеграцію з Telegram-ботом для оперативного інформування.
4. Telegram-бот – надає користувачу можливість запускати перевірки, отримувати результати, переглядати історію моніторингу, а також конфігурувати частоту перевірок та типи джерел.
5. Модуль тестування – реалізує сценарії з використанням Mock-об’єктів для безпечного моделювання витоків, емулює роботу API та бази даних без доступу до реальних облікових даних.

Система підтримує інтеграцію з платними REST API сервісами, використовуючи захищене підключення через HTTPS та токени доступу. Підключення здійснюється в тестовому режимі перед дипломним захистом, що дозволяє перевірити точність та ефективність сервісів, порівняти локальне тестування та реальні дані [3].

Для забезпечення високої точності оцінювання результатів моніторингу в системі впроваджено спеціалізовані алгоритми автоматизованої класифікації. Ці інструменти дозволяють диференціювати виявлені витоки за критеріями потенційного ризику та встановлювати пріоритетність подальшого оповіщення користувачів. Процес обробки інформації базується на комплексному аналізі

критичності скомпрометованих даних, що дозволяє формувати чітку ієрархію загроз.

Усі отримані відомості підлягають суворій структуризації: фіксується точна дата виявлення інциденту, ідентифікується тип джерела та обчислюється обсяг скомпрометованих облікових записів. Підсумковим етапом аналізу є генерація адаптивних рекомендацій, спрямованих на нейтралізацію безпекових ризиків для кінцевого споживача. Деталізований приклад результатів такої аналітичної обробки подано в табл. 1.

Таблиця 1 – Результати верифікації та класифікації виявлених інцидентів інформаційної безпеки

№	Джерело	Кількість записів	Тип даних	Ступінь ризику
1	ForumXYZ	25	Email + Password	Високий
2	Pastebin	12	Email	Середній
3	API Service	40	Email + Hash	Високий

Тестування системи включає модульне та інтеграційне тестування. Модульне тестування перевіряє роботу кожного компонента окремо, а інтеграційне – взаємодію модулів між собою та з Telegram-ботом. Використання Mock-об’єктів дозволяє створювати контрольовані сценарії витоків для оцінки реакції системи без ризику для реальних даних [4].

Результати показали, що застосування модульної архітектури та інтеграція з Telegram-інтерфейсом дозволяє ефективно моніторити витоки облікових даних, надавати швидке сповіщення користувачам та вести історію перевірок для подальшого аналізу. Перспективи розвитку включають розширення бази джерел, застосування алгоритмів машинного навчання для класифікації ризиків та автоматичного визначення типу витоку, а також інтеграцію з корпоративними системами безпеки [5].

Список використаних джерел

1. Aiogram Documentation. URL: https://docs.aiogram.dev (дата звернення 03.03.2026)
2. Have I Been Pwned API Documentation. URL: https://haveibeenpwned.com/API/v3 (дата звернення 05.03.2026)
3. Mock Library Documentation. URL: https://docs.python.org/3/library/unittest.mock.html (дата звернення 08.03.2026)
4. OWASP. Top 10 Security Risks for Web Applications. URL: https://owasp.org (дата звернення 10.03.2026)
5. Python 3.11 Documentation. URL: https://www.python.org/doc/ (дата звернення 12.03.2026)

УДК 004.41

ОСОБЛИВОСТІ ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЇ ВЕБ-ОРІЄНТОВАНИХ ІНФОРМАЦІЙНИХ СИСТЕМ НА ОСНОВІ ШАБЛОНУ MVT

Синьогуб А. Р.
alina.sinegu@icloud.com
Черкаський державний бізнес-коледж
Подорошко Д. І.
м. Черкаси, Україна

У роботі розглянуто особливості проєктування та реалізації веб-орієнтованої інформаційної системи «Beauty Time» на основі шаблону MVT із використанням фреймворку Django. Актуальність теми зумовлена потребою створення масштабованих, безпечних і зручних веб-застосунків у сфері послуг, де важливо забезпечити надійну обробку записів клієнтів, мінімізувати помилки під час бронювання та підвищити якість обслуговування.

Проєктується веб-орієнтована інформаційна система для салону краси, яка призначена для автоматизації процесу онлайн-запису клієнтів на послуги. Розробка такої системи обумовлена необхідністю переходу від неефективного ручного менеджменту записів до єдиного цифрового середовища, що дозволяє