УДК 004.056:004.942
DOI: 10.31673/2412-9070.2025.045866
І. В. ЗАМРІЙ, доктор техн. наук, професор;
ORCID: 0000-0001-5681-1871
І. О. ШАХМАТОВ, викладач,
ORCID: 0009-0004-9628-0365
Державний університет інформаційно-комунікаційних технологій, Київ

АВТОМАТИЗАЦІЯ ОЦІНКИ БЕЗПЕКИ ВЕБЗАСТОСУНКІВ ЗАСОБАМИ PYTHON

Методи оцінки ефективності сканерів безпеки вебзастосунків часто страждають від відсутності уніфікованих кількісних критеріїв, що ускладнює об'єктивне порівняння інструментів. Для вирішення цього обмеження пропонується модель автоматизованої оцінки, заснована на зіставленні результатів сканування з еталонним набором вразливостей (Ground Truth) та подальшому розрахунку метрик Precision, Recall і Accuracy. Модель реалізується за допомогою Python-інструментів: інтеграція сканерів (Wapiti, OWASP ZAP API, SQLMap) забезпечує збір даних, бібліотеки Pandas і Scikit-learn використовуються для класифікації вразливостей (TP, FP, FN, TN) та обчислення метрик, а Matplotlib відповідає за візуалізацію результатів. Перевагою підходу є стандартизація процесу оцінки, зменшення суб'єктивності та можливість гнучкої інтеграції нових сканерів через єдиний інтерфейс. Запропонована архітектура спрощує відтворення експериментів, забезпечує масштабованість мультисканерних платформ із централізованим аналізом безпеки.

Ключові слова: кібербезпека; Python; вебзастосунок; вразливості; автоматизація; оцінка безпеки; сканери вразливостей; метрики точності; інформаційна безпека.

Вступ

Залежності суспільства від інформаційних технологій, зокрема вебзастосунків, призвело до значного зростання потенційних загроз інформаційній безпеці. Сучасні вебзастосунки, які активно використовуються у таких критично важливих сферах, як електронна комерція, фінансові послуги, охорона здоров'я та державне управління, часто стають цілями кіберзлочинців. Основними векторами атак залишаються такі типи вразливостей, як SQL-ін'єкції, міжсайтовий скриптинг (XSS) та неправильна конфігурація систем безпеки, що призводить до серйозних наслідків: витоків конфіденційної інформації, фінансових втрат та суттєвого порушення довіри з боку користувачів.

Для оперативного виявлення потенційних загроз і зменшення ризиків, пов'язаних із використанням вебзастосунків, широко застосовуються автоматизовані сканери безпеки. Однак ефективність таких інструментів нерідко оцінюється поверхнево, а порівняльні дослідження часто мають фрагментарний характер через відсутність стандартизованих кількісних критеріїв. Як наслідок, розробникам і аудиторам складно зробити обґрунтований вибір інструментів, що відповідатимуть конкретним вимогам безпеки.

Аналіз існуючих досліджень свідчить, що значна увага приділяється удосконаленню алгоритмів детекції окремих типів вразливостей або суб'єктивному порівнянню сканерів, таких як OWASP ZAP і Burp Suite. Проте недостатньо уваги приділяється систематичному застосуванню кількісних метрик (Precision, Recall, Accuracy та F1-score), які дозволяють об'єктивно оцінити компроміс між точністю детекції загроз та кількістю хибних спрацювань. Крім того, більшість доступних підходів не передбачають автоматизацію процесів оцінки, що ускладнює повторюваність експериментів і обмежує можливості масштабування.

Актуальність даного дослідження обумовлена потребою розробки стандартизованої, автоматизованої методики, здатної забезпечити об'єктивну оцінку ефективності сканерів безпеки вебзастосунків. Запропонований підхід ґрунтується на формалізації методики оцінки

через використання еталонного набору вразливостей (Ground Truth), автоматичне зіставлення результатів сканування та кількісний аналіз результатів з використанням чітко визначених метрик Precision, Recall, Accuracy і F1-міри. Реалізація цього підходу на основі сучасних Python-інструментів (зокрема, бібліотек Pandas, Scikit-learn, Matplotlib та інтеграції сканерів Wapiti, OWASP ZAP і SQLMap) дозволяє автоматизувати та стандартизувати процес оцінки, значно зменшити суб'єктивність результатів і забезпечити масштабованість систем.

Основна частина

Аналіз літературних даних. Сучасна автоматизація аналізу безпеки вебзастосунків передбачає використання кількох сканерів з подальшим узагальненням результатів. Один із перспективних підходів представлено системою OCVA, що базується на роєвому інтелекті й імітує поведінку соціального павука. Вона забезпечує автоматичне виявлення вразливостей, їх пріоритизацію та візуалізацію [1]. Ключовим аспектом є коректна реалізація контролю доступу та протидії витоку даних. У цьому контексті важливо зіставляти вразливості з рівнями привілейованого доступу, а також впроваджувати механізми автентифікації й авторизації, що ізолюють ризики залежно від ролей користувачів [2]. Дослідження підтверджують ефективність комбінування автоматизованого сканування з ручним тестуванням. Жоден інструмент не забезпечує повного покриття всіх загроз OWASP Top 10, тому інтеграція кількох рішень дозволяє покращити точність виявлення [3]. Окремо виділяється підхід із залученням великих мовних моделей (LLM), зокрема GPT, які підвищують якість статичного аналізу фронтенд-коду. Використання семантичного аналізу у форматі JSON дозволяє автоматично класифікувати вразливості без участі людини [4].

Реалізація автоматизованих підходів у Python значною мірою залежить від вибору та комбінування інструментів. Інтеграція сканерів Wapiti, ZAP, W3af, а також краулера Arachni й автентифікації OAuth 2.0 покращує охоплення вхідних точок і знижує хибнопозитивні спрацьовування. Це підтверджується результатами оцінки за метриками Precision, Recall та Accuracy [5]. Дедалі більше досліджень акцентують на синергії традиційного сканування з інтелектуальними системами на базі машинного навчання. Інструменти Bandit і Semgrep демонструють здатність до аномалійного аналізу та поступової адаптації до нових типів загроз, що формує проактивну модель безпеки [6]. Окрему увагу привертає аналіз текстових даних із відкритих джерел. Python-бібліотеки для LDA, VADER, roBERTa дозволяють виявляти не лише технічні вразливості, а й соціальні тренди щодо кіберзагроз. Зокрема, аналіз мільйонів повідомлень про безпеку ChatGPT виявив ризики фішингу й генерації шкідливого коду, підкреслюючи потребу в інтеграції нетрадиційних джерел у системи автоматизованого захисту [7].

Важливим напрямом розвитку автоматизованої оцінки безпеки є впровадження глибокого навчання для протидії цілеспрямованим атакам (adversarial attacks). Використання нейронних мереж типу LSTM і RNN, оптимізованих алгоритмами на кшталт Grey Wolf Optimization, підвищує стійкість до атак DeepFool, L-BFGS та інших, які важко виявити традиційними сканерами. Така інтеграція забезпечує комплексний захист, доповнюючи класичне сканування виявленням аномалійних даних [8]. Сучасні інструменти автоматичного сканування, як правило, фокусуються на поширених загрозах (SQL-ін'єкції, XSS), але часто ігнорують інші важливі уразливості з OWASP Top 10. Використання відкритих Python-інструментів (Wapiti, OWASP ZAP API) у поєднанні з порівнянням результатів з еталонними наборами даних дозволяє підвищити точність, визначаючи показники TP, FP, FN, Precision та Recall. Це сприяє кращій адаптації до змін у вебсередовищі [9]. Комплексний підхід до безпеки включає не лише статичне та динамічне тестування, а й аналіз сторонніх компонентів. Інтеграція з CI/CD-процесами дозволяє виявляти вразливості вже на етапі розробки. Застосування OWASP ZAP API, Wapiti та відкритих форматів виводу забезпечує просте зіставлення результатів і формування надійного пайплайну управління ризиками [10].

Повний цикл аналізу передбачає не лише сканування, а й обробку результатів засобами Python. Бібліотеки Pandas, Matplotlib, mini-batch k-means забезпечують глибокий аналіз, структуризацію даних, зниження інформаційного шуму та адаптацію до різних типів загроз [11, 12]. Перспективним є підхід, що поєднує аналітичні та технологічні компоненти в межах модульної

архітектури. Застосування OWASP ZAP API, Wapiti у зв'язці з Django дозволяє побудувати масштабовану систему, здатну обчислювати метрики Precision, Recall та Accuracy у реальному часі. Подібно до моделей, що поєднують блокчейн із економічною аналітикою, сучасні рішення враховують не лише технічні, а й поведінкові характеристики користувачів, прокладаючи шлях до самонавчальних систем моніторингу [13]. Подальший розвиток пов'язаний з інтеграцією бібліотек Scikit-learn, Pandas, Seaborn, а також контейнеризацією (Docker) і застосуванням Flask/Django. Це забезпечує адаптивність, інтеграцію з REST API та відповідність сучасним підходам до CI/CD і персоналізованого захисту [14].

Одним із ключових напрямів розвитку сучасних рішень з автоматизації оцінки безпеки є організація безперервного моніторингу з гнучким управлінням результатами аналізу. Комбінація Flask із бібліотеками Pandas, Scikit-learn, Matplotlib дозволяє реалізувати неперервне сканування та водночас візуалізувати результати у зручному для користувача вигляді. Такі системи все частіше поєднуються з концепцією цифрових двійників та браузерних операційних платформ, де керування об'єктами здійснюється через API. Це сприяє швидкому реагуванню на зміни в середовищі та адаптації до нових загроз [15]. Важливою складовою є точна класифікація дефектів коду, особливо специфічних для Python. Надійність систем залежить від застосування стандартизованих таксономій (ODC, CWE, OWASP Top 10), а також від урахування специфічних для Python вразливостей — наприклад, небезпечне використання eval або недостатня перевірка вхідних даних. Формування якісних навчальних вибірок на основі анотованих прикладів (у т.ч. згенерованих штучним інтелектом) суттєво покращує точність автоматизованого виявлення [16]. Системи безпеки часто реалізуються на базі мікросервісної архітектури, що забезпечує модульність, масштабованість та зручне управління компонентами. Використання Flask або Django з MySQL/PostgreSQL у поєднанні з RESTful або GraphQL API дозволяє створювати продуктивні сервіси. До функціональності таких систем входить хешування, цифрові підписи, алгоритми довіри, а також виявлення аномалій за допомогою машинного навчання, що особливо важливо в умовах високої інтенсивності кіберзагроз [18, 19].

Однак, попри наявність рішень, залишаються проблеми: відсутність уніфікованих критеріїв оцінки ефективності, велика кількість хибнопозитивних спрацювань і складність інтеграції різних інструментів. Більшість платформ використовує комбінацію сканерів (Wapiti, OWASP ZAP, Arachni, W3af), що підвищує якість виявлення, але не забезпечує стандартизованої оцінки. Запропонована модель автоматизованої оцінки безпеки, реалізована засобами Python, вирішує ці обмеження завдяки централізованому збору даних, класифікації вразливостей та обчисленню метрик Precision, Recall та Accuracy. Інтеграція з Wapiti, ZAP API, SQLMap, а також використання бібліотек Pandas і Scikit-learn забезпечує стандартизовану оцінку (TP, FP, FN, TN), знижуючи суб'єктивність і спрощуючи інтеграцію нових інструментів у платформу.

Модель та методологія дослідження. Запропонована модель автоматизації оцінки безпеки вебзастосунків має на меті ефективне виявлення вразливостей із подальшою кількісною оцінкою результатів сканування. Формалізація проблеми базується на розгляді задачі як задачі класифікації, що дозволяє оцінити якість роботи автоматизованих сканерів з використанням кількісних метрик: точності (Precision), повноти (Recall), загальної точності (Accuracy) та інтегральної F1-міри. Поставлену задачу формалізовано таким чином. Припустимо, для аналізу задано множину еталонних (істинно присутніх) вразливостей вебзастосунку $V = \{v_1, v_2, \dots, v_n,\}$, визначених на основі тестового середовища. Під час автоматизованого сканування за допомогою засобів безпеки (таких як Wapiti, W3af, SQLMap або OWASP ZAP API) формується множина виявлених сканером вразливостей $\widehat{\mathrm{V}} = \{\hat{v}_1, \hat{v}_2, \dots, \hat{v}_m\}$.

На основі цих множин визначаються категорії результатів, які дозволяють оцінити точність роботи сканерів. До них належать істинно позитивні випадки (TP), що відповідають множині правильно ідентифікованих вразливостей: $TP = V \cap \widehat{\mathrm{V}}$; хибно позитивні ($FP$), що характеризують помилкові спрацьовування: $FP = \widehat{\mathrm{V}} \setminus V$; хибно негативні ($FN$), що відповідають пропущеним вразливостям: $FN = \mathrm{V} \setminus \widehat{\mathrm{V}}$. Четверта категорія — істинно негативні (TN) — представляє ситуації правильної класифікації відсутності вразливості; її визначення ускладнене у реальних умовах через неможливість чіткого окреслення повного простору потенцій-

них вразливостей. Для подальшої оцінки ефективності роботи сканера використовуються стандартні метрики інформаційного пошуку та машинного навчання. Метрика Precision (1) (точність) [5] дозволяє кількісно оцінити частку коректно ідентифікованих вразливостей від загального числа виявлених сканером:

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

Метрика Recall (2) (повнота, чутливість) демонструє здатність сканера ідентифікувати всі наявні вразливості, визначаючи частку правильно виявлених загроз відносно їх реальної кількості:

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

Accuracy (3) (загальна точність класифікації) оцінює співвідношення правильно класифікованих випадків до загальної кількості випадків, що аналізуються. Ця метрика стає обмежено застосовною у випадках неможливості точного визначення істинно негативних випадків:

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (3)$$

Для отримання інтегрального показника, який враховує баланс між точністю та повнотою, використовується F1-міра (4). Вона є гармонійним середнім між Precision та Recall і дозволяє комплексно оцінити ефективність інструменту:

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (4)$$

На підставі зазначених метрик модель може бути узагальнено представлена оператором оцінювання ефективності сканера як функцією M, що переводить множини V та $\widehat{\mathrm{V}}$ у простір кількісних критеріїв:

$$M: \left(V, \widehat{\mathrm{V}}\right) \longrightarrow (Precision, Recall, F_1) \quad (5)$$

Запропонована модель передбачає можливість узагальнення та застосування до багатосканерних систем, де кожен із використаних сканерів формує множину результатів $\hat{V}_i$. Об'єднаний результат усіх сканерів визначається як:

$$\hat{V}_{combined} = \bigcup_{i=1}^{k} \hat{V}_i \quad (6)$$

Такий підхід дозволяє оцінювати не лише ефективність окремих сканерів, а й ефект від їх спільного використання (ансамблевий підхід), а також аналізувати, які вразливості виявляються лише певними інструментами, а які — багатьма одночасно.

Процес починається із надходження вхідних даних, якими виступають URL цільового вебзастосунку або спеціалізоване тестове середовище. Ці дані подаються на вхід одному або кільком обраним сканерам безпеки (Wapiti, OWASP ZAP або SQLMap) (рис. 1). Сканери здійснюють автоматичне тестування застосунку з метою виявлення вразливостей.

Отримані результати сканування порівнюються з еталонною множиною вразливостей (Ground Truth), яка заздалегідь

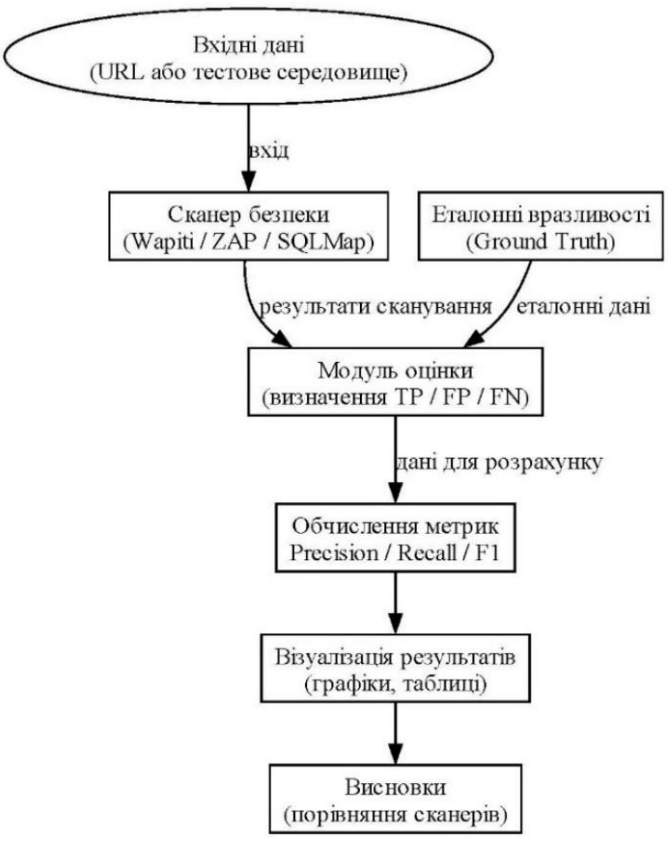

Рис. 1. Схема роботи моделі оцінки безпеки вебзастосунку з використанням Python-інструментів

відома для використаного тестового середовища. На основі цього порівняння у спеціальному модулі оцінки визначаються множини істинно пози-тивних (TP), хибно позитивних (FP) і хибно негативних (FN) результатів. Визначені множини даних є основою для розрахунку кількісних метрик ефективності: точності ($Precision$), пов-ноти ($Recall$) та інтегральної міри (F_1). Після отримання значень зазначених метрик резуль-тати візуалізуються у вигляді графіків та таблиць, що дозволяє провести порівняльний аналіз ефективності використаних сканерів. На останньому етапі формулюються висновки щодо результативності та особливостей роботи кожного з досліджуваних інструментів, що допо-магає у прийнятті подальших рішень щодо вибору та конфігурації засобів автоматичного вияв-лення вебвразливостей.

Проведено порівняльний аналіз ефективності кількох широко використовуваних сканерів вебвразливостей: BurpSuite, Wapiti, Acunetix, SkipFish, Netsparker, W3AF, AppSpider, IronWASP, Arachni, ZAP та Vega. Для аналізу використано дані тестування на платформі WAVSEP, які відображають метрики Precision, Recall та F1-міру для різних типів вразливостей, таких як SQL Injection (SQLI), Cross-Site Scripting (XSS), Remote File Inclusion (RFI) та Local File Inclusion (LFI) [17]. Графічне порівняння цих сканерів, побудоване з використанням мови програмування Python і бібліотеки Matplotlib (рис. 2).

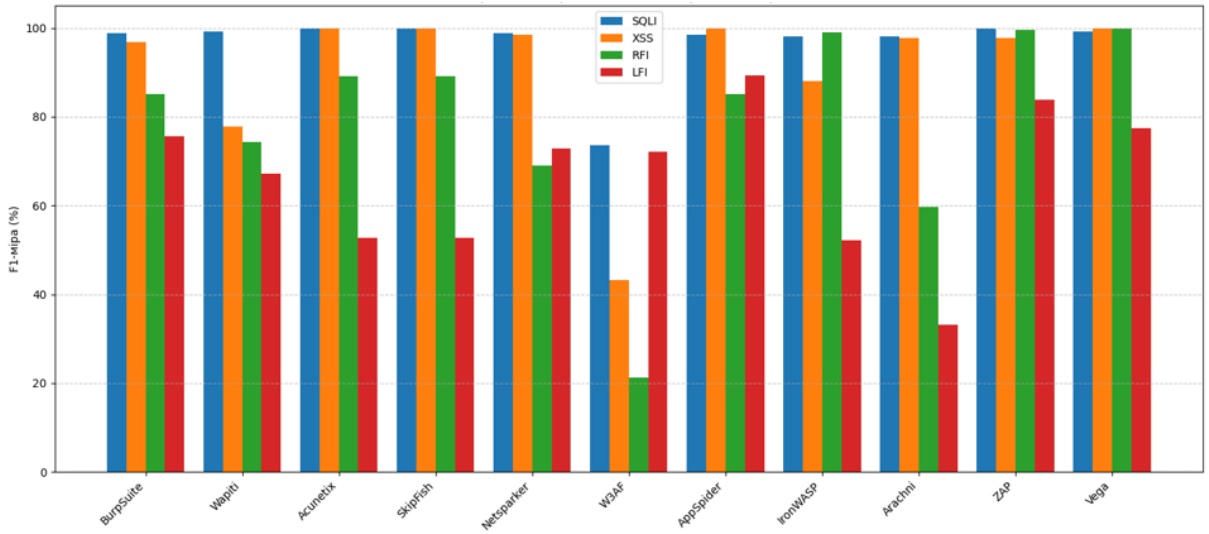

Рис. 2. Порівняння ефективності різних сканерів за F1-мірою для різних типів вразливостей (SQLI, XSS, RFI, LFI)

Різні сканери мають суттєві відмінності у виявленні різних типів вразливостей, що підкреслює важливість багатосканерного підходу для більш повної та достовірної оцінки безпеки вебзастосунків. Наприклад, сканери Acunetix, SkipFish і ZAP демонструють високі значення F1-міри для більшості типів вразливостей, тоді як W3AF має значно нижчу точність для RFI.

Реалізація моделі на Python

В основу практичної реалізації покладено наступні інструменти: Wapiti (CLI-сканер вебвразливостей із підтримкою Python API), OWASP ZAP API для автоматизації сканування, Pandas — бібліотека для обробки та структурування отриманих результатів, а також Matplotlib для візуалізації даних і побудови наочних графіків, що дозволяють аналізувати отримані результати. Процес реалізації моделі охоплює декілька послідовних етапів. На першому етапі здійснюється запуск автоматизованого сканування вебзастосунку за допомогою зазначених вище інструментів, результати якого зберігаються у структурованому форматі даних (JSON або XML), зручному для подальшої автоматизованої обробки.

Другий етап включає зіставлення результатів сканування з еталонним набором вразливостей (Ground Truth), що представляє собою множину реальних вразливостей, попередньо визначених у спеціалізованих тестових середовищах. У результаті цього етапу формуються три множини: істинно позитивні (TP) — правильно виявлені вразливості, хибно позитивні (FP) — по-

милкові спрацьовування сканера і хибно негативні (FN) — реальні вразливості, які були пропущені сканером. Істинно негативні (TN) визначаються умовно, через складність повного опису простору потенційних вразливостей.

Підхід реалізації розглядається, як множина еталонних вразливостей, що містить чотири відомі типи: SQL Injection, XSS, CSRF та File Inclusion. Після автоматизованого сканування отримується множина знайдених вразливостей: SQL Injection, XSS і Open Redirect. Використовуючи Python, множини TP, FP, FN розраховуються наступним чином:

```
# Еталонний список вразливостей
ground_truth = {"SQL Injection", "XSS", "CSRF", "File Inclusion"}

# Результати роботи сканера
scanner_output = {"SQL Injection", "XSS", "Open Redirect"}

# Обчислення множин TP, FP, FN
TP = ground_truth & scanner_output
FP = scanner_output - ground_truth
FN = ground_truth - scanner_output
```

Отримані результати: TP = {"SQL Injection", "XSS"}, FP = {"Open Redirect"}, FN = {"CSRF", "File Inclusion"}. Наступний етап включає розрахунок основних метрик Precision, Recall та Accuracy. Precision визначається як частка правильно виявлених вразливостей серед усіх знайдених сканером, Recall — як частка правильно виявлених вразливостей відносно їх загальної кількості у вебзастосунку, Accuracy — як співвідношення правильно визначених вразливостей до загальної кількості еталонних випадків (у спрощеному варіанті, без TN). Код на Python для розрахунку цих метрик має вигляд:

```
precision = len(TP) / (len(TP) + len(FP)) if (len(TP) + len(FP)) > 0 else 0
recall = len(TP) / (len(TP) + len(FN)) if (len(TP) + len(FN)) > 0 else 0
accuracy = len(TP) / len(ground_truth)

print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"Accuracy: {accuracy:.2f}")
```

Для комплексної автоматизації оцінки можна використовувати бібліотеку Scikit-learn, яка дозволяє обчислювати метрики більш лаконічно. Наведений нижче фрагмент демонструє застосування цієї бібліотеки:

```
from sklearn.metrics import precision_score, recall_score, accuracy_score

# Умовна бінарна розмітка: 1 – вразливість присутня, 0 – відсутня
true_labels = [1, 0, 1, 1, 0, 1, 0]
predicted_labels = [1, 0, 1, 0, 0, 1, 1]

precision = precision_score(true_labels, predicted_labels)
recall = recall_score(true_labels, predicted_labels)
accuracy = accuracy_score(true_labels, predicted_labels)

print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"Accuracy: {accuracy:.2f}")
```

Реалізована модель забезпечує стандартизований і відтворюваний процес оцінки ефективності засобів виявлення вразливостей. Python як основа реалізації гарантує гнучкість, масштабованість і легку інтеграцію нових інструментів, що робить підхід актуальним як для наукових досліджень, так і для практичного використання у галузі кібербезпеки.

Висновки

У рамках даного дослідження було запропоновано та реалізовано модель автоматизації оцінки ефективності сканерів безпеки вебзастосунків з використанням засобів програмування мовою Python. Основна мета роботи полягала у розв'язанні проблеми суб'єктивності та фрагментарності існуючих методів оцінювання ефективності інструментів автоматизованого пошуку вебвразливостей. Запропонована модель спирається на класичні кількісні метрики машинного навчання й інформаційного пошуку (Precision, Recall, Accuracy та інтегральна F1-міра), що дозволяють об'єктивно порівнювати різні сканери та забезпечують прозорість отриманих результатів. У ході дослідження була здійснена формалізація проблеми автоматизованого виявлення вразливостей, яка базується на розгляді задачі як задачі бінарної класифікації. Запропонований підхід передбачає чітке визначення категорій результатів роботи сканерів: істинно позитивних (TP), хибно позитивних (FP), хибно негативних (FN) та істинно негативних (TN). Це дозволяє комплексно оцінити роботу засобів тестування за допомогою кількісних критеріїв, зменшуючи ризики суб'єктивних суджень і підвищуючи точність отриманих результатів.

Розроблена алгоритмічна модель передбачає автоматизоване зіставлення результатів сканування з еталонною множиною вразливостей (Ground Truth), формуючи основу для обчислення зазначених метрик. Завдяки такій формалізації модель здатна ефективно вирішувати завдання оцінки не лише окремих сканерів, але й мультисканерних систем, де результати кількох інструментів агрегуються для комплексного аналізу ефективності. Це створює основу для формування рекомендацій щодо оптимального вибору та конфігурації інструментів автоматизованого тестування вебзастосунків, зокрема в умовах обмежених ресурсів та необхідності швидкого реагування на нові кіберзагрози. Практична реалізація моделі здійснена за допомогою сучасних Python-інструментів, зокрема сканерів Wapiti та OWASP ZAP, бібліотек Pandas і Scikit-learn для автоматичної обробки та аналізу результатів, а також Matplotlib для візуалізації. Реалізована програмна модель дозволяє здійснювати як швидкий порівняльний аналіз результативності сканерів, так і глибокий аналіз результатів, що забезпечує ефективний контроль за якістю проведення автоматизованих тестувань вебзастосунків.

Проведений приклад аналізу результатів сканування продемонстрував практичну ефективність моделі та висвітлив її переваги. На прикладі умовних даних було підтверджено, що запропонований підхід дозволяє прозоро та чітко ідентифікувати як сильні, так і слабкі сторони роботи інструментів автоматизованого пошуку вразливостей. Водночас, інтегральні метрики забезпечують комплексне уявлення про роботу сканерів та дають змогу приймати обґрунтовані рішення щодо вибору інструментів у контексті конкретних завдань інформаційної безпеки. Використання API забезпечує гнучкість архітектури, дозволяючи швидко інтегрувати нові сканери або конфігурації без суттєвих змін у кодовій базі, що є ключовим чинником у розвитку мультисканерних платформ.

Список літератури

1. Chahal N. S., Bali P., Khosla P. K. A proactive approach to assess web application security through the integration of security tools in a Security Orchestration Platform // Computers & Security. – 2022. – Vol. 122. – Article № 102886. – DOI: https://doi.org/10.1016/j.cose.2022.102886.

2. Pant P., Rajawat A. S., Goyal S. B., Bedi P., Verma C., Raboaca M. S., Enescu F. M. Authentication and authorization in modern web apps for data security using Nodejs and role of Dark Web // Procedia Computer Science. – 2022. – Vol. 215. – P. 781–790. – DOI: https://doi.org/10.1016/j.procs.2022.12.080.

3. Altulaihan E. A., Alismail A., Frikha M. A survey on web application penetration testing // Electronics. – 2023. – Vol. 12, No. 5. – Article № 1229. – DOI: https://doi.org/10.3390/electronics12051229.

4. Szabó Z., Bilicki V. A new approach to web application security: Utilizing GPT language models for source code inspection // Future Internet. – 2023. – Vol. 15, No. 10. – Article № 326. – DOI: https://doi.org/10.3390/fi15100326.

5. Shahid J., Hameed M. K., Javed I. T., Qureshi K. N., Ali M., Crespí N. A comparative study of web application security parameters: Current trends and future directions // Applied Sciences. – 2022. – Vol. 12, No. 8. – Article № 4077. – DOI: https://doi.org/10.3390/app12084077.

6. Shapovalova O.O. Artificial intelligence in cybersecurity and computer science // Матеріали XVI-ої Міжнародної науково-практичної конференції «Free and Open Source Software», 13–14 лютого 2025 р., м. Харків. – Харків: Харківський національний економічний університет імені Семена Кузнеця, 2025. – С. 16–18.

7. Okey O. D., Udo E. U., Rosa R. L., Rodríguez D. Z., Kleinschmidt J. H. Investigating ChatGPT and cybersecurity: A perspective on topic modeling and sentiment analysis // Computers & Security. – 2023. – Vol. 135. – Article № 103476. – DOI: https://doi.org/10.1016/j.cose.2023.103476.

8. Barik K., Misra S. Adversarial attack defense analysis: An empirical approach in cybersecurity perspective // Software Impacts. – 2024. – Vol. 21. – Article № 100681. – DOI: https://doi.org/10.1016/j.simpa.2024.100681.

9. Alazmi S., De Leon D. C. A systematic literature review on the characteristics and effectiveness of web application vulnerability scanners // IEEE Access. – 2022. – Vol. 10. – P. 33200–33219. – DOI: https://doi.org/10.1109/ACCESS.2022.3161522.

10. Cruz D. B., Almeida J. R., Oliveira J. L. Open source solutions for vulnerability assessment: A comparative analysis // IEEE Access. – 2023. – Vol. 11. – P. 100234–100255. – DOI: https://doi.org/10.1109/ACCESS.2023.3315595.

11. Zhurakovskyi B., Averichev I., Shakhmatov I. Using the latest methods of cluster analysis to identify similar profiles in leading social networks // CEUR Workshop Proceedings. – 2024. – Vol. 3646. – P. 110–120. – URL: http://ceur-ws.org/Vol-3646/Paper_12.pdf.

12. Замрій І. В., Шахматов І. О., Яскевич В. О. BlockchainSQLSecure: інтеграція блокчейн-технології для зміцнення захисту від SQL-ін'єкцій // Вісник Київського національного університету імені Тараса Шевченка. – 2024. – № 1(29). – С. 160–167. – DOI: https://doi.org/10.17721/1812-5409.2024/1.29.

13. Шахматов І. О. Технологія Blockchain як інструмент протидії неправомірному використанню доступу до вебсайтів // Зв'язок. – 2024. – № 1(167). – С. 42–47. – DOI: https://doi.org/10.31673/2412-9070.2024.012025.

14. Integrating machine learning into web applications for personalized content delivery using Python // International Journal of Current Engineering and Technology. – 2021. – Vol. 11, No. 4. – P. 652–660. – URL: https://www.researchgate.net/publication/386523063.

15. Bonney M. S., de Angelis M., Dal Borgo M., Andrade L., Beregi S., Jamia N., Wagg D. J. Development of a digital twin operational platform using Python Flask // Data-Centric Engineering. – 2022. – Vol. 3. – Article e1. – DOI: https://doi.org/10.1017/dce.2022.1.

16. Bogaerts F. C. G., Ivaki N., Fonseca J. A taxonomy for Python vulnerabilities // IEEE Open Journal of the Computer Society. – 2024. – Vol. 5. – P. 368–379. – DOI: https://doi.org/10.1109/OJCS.2024.3422686.

17. El Idrissi S., Berbiche N., Guerouate F., Sbihi M. Performance evaluation of web application security scanners for prevention and protection against vulnerabilities // International Journal of Applied Engineering Research. – 2017. – Vol. 12, No. 21. – P. 11068–11076. – ISSN 0973-4562. – URL: https://www.ripublication.com/ijaer17/ijaerv12n21_76.pdf.

18. Шахматов І.О., Вєрягін К. О. Розробка та аналіз веб-сервісу для оптимізації процесів в автомобільному бізнесі // Матеріали Всеукраїнської науково-технічної конференції «Застосування програмного забезпечення в інформаційно-комунікаційних технологіях», 24 квітня 2024 р., Київ. – Київ: Державний університет інформаційно-комунікаційних технологій, 2024. – С. 449–453. – URL: https://duikt.edu.ua/uploads/p_2661_45497999.pdf.

19. Замрій І.В., Шахматов І. О. Технологія Blockchain як інструмент протидії неправомірному використанню доступу до вебсайтів // Матеріали V Міжнародної науково-практичної конференції молодих вчених та студентів «Інженерія програмного забезпечення і передові інформаційні технології (Soft Tech-2023)», 19–21 грудня 2023 р., Київ. – Київ: Державний університет інформаційно-комунікаційних технологій, 2023. – С. 360–364.

I. Zamrii, I. Shakhmatov

AUTOMATING WEB APPLICATION SECURITY ASSESSMENT USING PYTHON

Methods for evaluating the effectiveness of web application security scanners are often constrained by subjectivity and the lack of unified quantitative criteria, complicating the comparison of their performance. This paper proposes an automated model for assessing web application security based on a Ground Truth dataset of vulnerabilities and the calculation of standardized metrics, such as Precision, Recall, Accuracy, and F1-score. The proposed model is implemented using advanced Python tools. Specifically, popular vulnerability scanners such as Wapiti, OWASP ZAP, and SQLMap are integrated to automate vulnerability data collection. Processing and classification of the obtained results (TP, FP, FN, TN) are performed using pandas and scikit-learn libraries, facilitating effective quantitative data analysis. Visualization of metrics and research results is provided by the matplotlib library. The proposed architecture offers significant advantages, including reduced subjectivity in evaluating scanner effectiveness, standardization of the comparison process, and flexibility in system expansion through the addition of new scanners via a unified programming interface. The implemented model enhances scalability and reproducibility of experiments, crucial for centralized information security management in large multi-scanner platforms. Experimental analysis demonstrates the practical efficacy of the proposed solution: the model transparently identifies both strengths and weaknesses of individual scanners, providing comprehensive information for informed decision-making in the field of information security. Future directions include expanding the range of metrics (notably ROC curves), integrating additional scanners, and developing comprehensive ensemble models based on machine learning.

Keywords: cybersecurity; Python; web applications; vulnerabilities; automation; security assessment; vulnerability scanners; accuracy metrics; information security.