JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 34 (2025) 36–43
Received: 23 September 2024
Accepted: 2 November 2024

Comparison of the effectiveness of tools for testing the security of web applications

Porównanie skuteczności narzędzi do testowania bezpieczeństwa aplikacji internetowych

Izabela Kinga Kaźmierak*

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract
This article presents a comparative analysis of the effectiveness of three web application security scanners: ZAP, Wapiti, and Skipfish. Automated scanning was conducted on deliberately unsecured applications, followed by an analysis of the detected vulnerabilities. The results were presented in the form of comparative tables and graphs illustrating the number and types of detected threats. The analysis showed that ZAP detected the most vulnerabilities, particularly in low-risk categories, Skipfish excelled in identifying specific threats, while Wapiti was effective in finding simple vulnerabilities. The study demonstrated the need to combine different scanners and supplement them with manual tests for a comprehensive assessment of web application security.

Keywords: web application security; testing tools; cybersecurity

Streszczenie
W niniejszym artykule przedstawiono analizę porównawczą skuteczności trzech skanerów bezpieczeństwa aplikacji internetowych: ZAP, Wapiti oraz Skipfish. Przeprowadzono automatyczne skanowanie aplikacji, które zostały celowo niezabezpieczone, a następnie dokonano analizy wykrytych podatności. Wyniki zostały przedstawione w formie tabel porównawczych oraz wykresów ilustrujących liczbę i rodzaje wykrytych zagrożeń. Analiza wykazała, że ZAP wykrył najwięcej podatności, szczególnie w kategoriach niskiego ryzyka, Skipfish wyróżnił się w identyfikacji specyficznych zagrożeń, a Wapiti okazał się skuteczny w odnajdywaniu prostych luk. Badanie wykazało potrzebę łączenia różnych skanerów i uzupełniania ich manualnymi testami dla kompleksowej oceny bezpieczeństwa aplikacji internetowych.

Słowa kluczowe: bezpieczeństwo aplikacji internetowych; narzędzia do testowania; cyberbezpieczeństwo

*Corresponding author
Email address: **izabela.kazmierak@pollub.edu.pl** (I. K. Kaźmierak)

1. Wstęp

Testowanie bezpieczeństwa aplikacji internetowych stało się kluczowym elementem w procesie tworzenia oprogramowania. W obliczu ewoluujących zagrożeń cyberbezpieczeństwa, niezbędne jest, aby aplikacje internetowe były nie tylko funkcjonalne i wydajne, ale przede wszystkim bezpieczne i odporne na ataki. Właściwie przeprowadzone testy bezpieczeństwa mogą znacząco zminimalizować ryzyko naruszenia poufności danych, integralności systemów oraz dostępności usług.

Wybór odpowiednich narzędzi do testowania bezpieczeństwa jest kluczowy i powinien być uwzględniony już na etapie planowania procesu zabezpieczania aplikacji. Kilka lat temu głównym wyzwaniem dla zespołów ds. bezpieczeństwa był wybór między testowaniem manualnym a automatycznym. Obecnie, dzięki rozwojowi zaawansowanych narzędzi automatyzujących proces testowania, kluczowe stało się porównanie ich skuteczności i wydajności.

Jednym z problemów, który często pojawia się w kontekście wyboru narzędzi do testowania bezpieczeństwa, jest trudność w ocenie ich rzeczywistej skuteczności w wykrywaniu różnorodnych luk bezpieczeństwa. Wynika to z faktu, że każde narzędzie może mieć swoje mocne i słabe strony w zależności od rodzaju testowanej aplikacji czy specyfiki potencjalnych zagrożeń.

W odpowiedzi na przedstawione wyzwania, coraz większą popularnością cieszą się kompleksowe porównania narzędzi do testowania bezpieczeństwa aplikacji internetowych. Dzięki nim możliwe jest dokonanie świadomego wyboru narzędzia najlepiej dopasowanego do potrzeb konkretnego projektu, co przekłada się na zwiększenie efektywności procesu testowania i, w konsekwencji, poprawę ogólnego poziomu bezpieczeństwa aplikacji.

Celem niniejszego artykułu jest porównanie skuteczności trzech popularnych narzędzi do testowania bezpieczeństwa aplikacji internetowych: Zed Attack Proxy (ZAP), Wapiti oraz Skipfish. Zostanie to zbadane na podstawie przygotowanego środowiska testowego, składającego się z aplikacji internetowych zawierających celowo wprowadzone luki bezpieczeństwa.

Na tej podstawie postawiona została następująca hipoteza: Zed Attack Proxy osiągnie najlepsze wyniki w wykrywaniu zagrożeń o wysokim poziomie zagrożenia.

2. Przegląd literatury

W dziedzinie badania skuteczności narzędzi do testowania bezpieczeństwa aplikacji internetowych istnieje wiele

publikacji, które starają się ocenić i porównać różne skanery oraz metody badawcze. Badania te skupiają się na wykrywaniu luk bezpieczeństwa oraz ocenie efektywności narzędzi w identyfikowaniu i likwidowaniu zagrożeń w aplikacjach internetowych.

Autorzy artykułów [1, 2] zajęli się porównaniem skanerów Skipfish i Zed Attack Proxy. Według autorów ZAP okazał się bardziej efektywny od Skipfish pod względem wskaźnika precyzji, prawie dwukrotnie przewyższając go. W ramach przyszłych prac, autorzy zamierzają przeprowadzić kolejne eksperymenty, ale tym razem bez używania specjalnie podatnej aplikacji internetowej. Zamiast tego, planują skorzystać ze standardowej strony internetowej jako celu testów, co może zapewnić bardziej realistyczne warunki i lepiej odzwierciedlić typowy scenariusz ataku.

Po przeprowadzeniu testów oraz szczegółowej analizie narzędzi skanujących, takich jak w3af, ZAP i Skipfish, pod kątem różnorodnych parametrów, w tym czasu skanowania oraz liczby oraz typu wykrytych luk bezpieczeństwa, autorzy pracy [3] doszli do wniosku, że skaner OWASP ZAP osiągnął lepsze wyniki niż Skipfish i w3af. W ramach tych eksperymentów badawczych wykorzystano podatną na ataki aplikację internetową DVWA. Autorzy odkryli, że nie istnieje jeszcze skaner podatności, który byłby w stanie wykryć wszystkie luki z listy OWASP Top 10.

W innej pracy [4], porównano skuteczność narzędzi open-source z komercyjnymi rozwiązaniami. W artykule przedstawiono analizę porównawczą skuteczności ośmiu różnych skanerów w wykrywaniu luk w zabezpieczeniach dwóch podatnych na ataki aplikacji internetowych (DVWA i WebGoat). Spośród badanych narzędzi, trzy były płatnymi rozwiązaniami komercyjnymi (Acunetix, HP Webinspect i IBM Appscan), a pięć to narzędzia typu open-source (ZAP, Skipfish, Arachni, Vega i IronWASP). Ich skuteczność została zbadana przy użyciu następujących wskaźników: precyzji, czułości, czasu skanowania, indeksu Youdena, OWASP WBE i Web Application Security Scanner Evaluation Criteria (WASSEC). Wyniki eksperymentów wskazują, że choć komercyjne skanery były efektywne w wykrywaniu luk w zabezpieczeniach, istniały również skanery typu open-source (ZAP i Skipfish), które odznaczały się równie wysoką skutecznością w identyfikacji pewnych rodzajów podatności, takich jak cross-site scripting i ataki typu SQL injection.

Głównym celem autorów artykułu [5] była ocena skuteczności różnych skanerów podatności aplikacji internetowych w wykrywaniu luk w nowoczesnych aplikacjach internetowych, które wykorzystują Node.js. W badaniu brały udział cztery narzędzia do skanowania aplikacji internetowych, w tym trzy o otwartym kodzie źródłowym (OWASP ZAP, Wapiti i Arachni), oraz jedno komercyjne narzędzie, Burp Suite Professional. Żadne z narzędzi nie było w stanie wykryć połowy zaprojektowanych luk w testowanych aplikacjach. Autorzy nie doszli do jednoznacznych wniosków co do najbardziej skutecznego narzędzia i zalecili dalsze badania w tej dziedzinie.

Autorki artykułu [6] porównały narzędzia OWASP ZAP, Burp Suite, Nikto i Skipfish w testowaniu bezpieczeństwa aplikacji internetowych. Badania opierały się na wykryciu i ocenie luk bezpieczeństwa pod względem rodzaju, liczby wystąpień i poziomu zagrożenia. Według badań, OWASP ZAP okazało się najskuteczniejszym narzędziem, wykrywając dużą liczbę zagrożeń wysokiego i średniego poziomu w badanych aplikacjach. Burp Suite również osiągnęło znaczące wyniki, identyfikując dużą liczbę zagrożeń wysokiego poziomu. Natomiast narzędzia Nikto i Skipfish wykazały się mniejszą skutecznością, wykrywając znacznie mniej zagrożeń niż OWASP ZAP i Burp Suite. Dodatkowo, OWASP ZAP i Burp Suite wykryły istotne podatności, których Nikto i Skipfish nie zidentyfikowały. Luki wykryte przez Nikto i Skipfish nie stanowiły poważnych zagrożeń dla bezpieczeństwa aplikacji.

Przegląd literatury pokazał, że istnieje wiele badań porównawczych dotyczących narzędzi do testowania bezpieczeństwa aplikacji internetowych, a wyniki tych analiz wskazują na zróżnicowaną skuteczność poszczególnych narzędzi w identyfikacji różnorodnych zagrożeń. Wciąż istnieje potrzeba dalszych badań mających na celu opracowanie bardziej wszechstronnych i efektywnych rozwiązań testowania bezpieczeństwa aplikacji internetowych.

3. Badane narzędzia do testowania bezpieczeństwa aplikacji internetowych

Narzędzia Zed Attack Proxy (ZAP), Wapiti i Skipfish zostały wybrane do porównania skuteczności narzędzi do testowania bezpieczeństwa aplikacji internetowych ze względu na ich popularność, różnorodność podejść do testowania oraz dostępność jako narzędzia open-source.

3.1. Zed Attack Proxy

Zed Attack Proxy (ZAP) jest najczęściej stosowanym na świecie skanerem aplikacji internetowych [7]. Jest to narzędzie przeznaczone do testów penetracyjnych, które ma na celu identyfikację potencjalnych luk bezpieczeństwa w aplikacjach internetowych. ZAP jest narzędziem o otwartym kodzie źródłowym, które jest bezpłatne i aktywnie rozwijane przez międzynarodowy zespół wolontariuszy. Narzędzie Zed Attack Proxy jest dostępne na platformach Windows, Linux i macOS oraz obsługuje wiele języków, dzięki czemu jest dostępne dla szerokiego grona użytkowników.

3.2. Wapiti

Wapiti jest narzędziem do analizy bezpieczeństwa aplikacji internetowych, działającym na zasadzie „czarnej skrzynki” [8]. Skaner ten nie analizuje kodu źródłowego aplikacji, lecz przeprowadza testy na zasadzie eksploracji i interakcji z działającą aplikacją. Wapiti odwiedza strony aplikacji, analizuje dane wejściowe, a następnie próbuje znaleźć luki bezpieczeństwa, symulując różne ataki. Obecnie Wapiti nie posiada graficznego interfejsu użytkownika, działa tylko jako narzędzie

wiersza poleceń. Narzędzie jest dostępne na platformach Linux i macOS.

3.3. Skipfish

Skipfish to w pełni zautomatyzowany skaner bezpieczeństwa aplikacji internetowych, opracowany przez zespół inżynierów z Google, w tym Michała Zalewskiego, Nielsa Heinena oraz Sebastiana Roschke [9]. Jest to narzędzie w pełni zoptymalizowane pod kątem prędkości, co jest możliwe dzięki efektywnemu zarządzaniu zasobami sieciowymi i używaniu zaawansowanych technik analizy HTTP. Jego działanie jest oparte na metodzie brute force, które generuje szczegółowe mapy witryn internetowych poprzez aktywne skanowanie i testowanie punktów wejściowych aplikacji. Narzędzie to jest dostępne w dystrybucji Kali Linux, czyli popularnym systemie operacyjnym do testów penetracyjnych [10].

4. Metoda badawcza

Do weryfikacji skuteczności wybranych narzędzi zostały wybrane dwie aplikacje internetowe Gin & Juice Shop i Damn Small Vulnerable Web (DSVW). Obie aplikacje oferują szeroki wachlarz potencjalnych podatności i scenariuszy ataków, co pozwala na kompleksowe przetestowanie możliwości narzędzi do wykrywania zagrożeń. Gin & Juice Shop jest znany z tego, że symuluje podatności występujące w rzeczywistych aplikacjach e-commerce [11], podczas gdy DSVW oferuje uproszczony zestaw podatności [12].

Skanowanie aplikacji przeprowadzono na maszynie wirtualnej Kali Linux. Ta dystrybucja systemu Linux jest specjalnie zaprojektowana do testów penetracyjnych i audytów bezpieczeństwa. Zawiera ona preinstalowane narzędzia do testowania bezpieczeństwa, w tym wykorzystane w badaniu skanery.

Za pomocą każdego z narzędzi wykonano automatyczny skan bezpieczeństwa aplikacji używając domyślnych ustawień. Skanowanie przeprowadzono kolejno dla każdego narzędzia, aby uniknąć wzajemnych zakłóceń.

Otrzymane rezultaty przedstawiono w tabelach z wynikami skanowania dla każdego narzędzia. Przygotowano tabele zbiorcze pokazujące skuteczność skanerów w wykrywaniu podatności obecnych w aplikacjach testowych. Na podstawie wyników przedstawionych w tych tabelach zostały obliczone następujące metryki:

$$\text{czułość} = \frac{TP}{TP + FN} \tag{1}$$

$$\text{precyzja} = \frac{TP}{TP + FP} \tag{2}$$

$$\text{F1} = \frac{2\ (precyzja * czułość)}{precyzja + czułość} \tag{3}$$

gdzie jako liczba prawdziwych pozytywnych (TP) została przyjęta liczba podatności wykrytych poprawnie przez skaner. Liczba fałszywych pozytywnych (FP) to liczba zagrożeń, które zostały wykryte, ale nie istniały w aplikacji. Fałszywe negatywne (FN) oznaczają liczbę podatności, które istnieją w aplikacji, ale nie zostały wykryte przez narzędzie. Czułość mierzy zdolność skanera do wykrywania rzeczywistych podatności. Precyzja ocenia dokładność wykryć, co minimalizuje fałszywe alarmy. Metryka F1 łączy te dwie wartości, dostarczając ogólną ocenę skuteczności skanera.

Analiza zakładała sprawdzenie liczby wykrytych podatności. W przypadku narzędzi ZAP i Skipfish, poziomy zagrożeń były bezpośrednio raportowane przez skanery, co ułatwiło analizę wyników. Natomiast w przypadku Wapiti konieczne było ręczne dopasowanie poziomów zagrożeń do klasyfikacji stosowanej przez pozostałe narzędzia, aby zapewnić spójność porównania wyników. Ostatecznie, przyjęto następującą ujednoliconą klasyfikację poziomów zagrożeń:

- wysoki – np. SQL Injection, Cross-Site Scripting, Path Traversal,
- średni – np. Missing Anti-clickjacking Header, Absence of Anti-CSRF Tokens, Buffer Overflow,
- niski – np. Cookie No HttpOnly Flag, X-Content-Type-Options Header Missing,
- informacyjny – np. Sensitive Information in URL, User Controllable Charset.

5. Analiza wyników

5.1. Wyniki skanowania aplikacji Gin & Juice Shop

W Tabelach 1, 2 i 3 zostały przedstawione wyniki skanowania aplikacji internetowej Gin & Juice Shop.

Tabela 1: Wyniki skanowania aplikacji Gin & Juice Shop narzędziem Zed Attack Proxy

Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
Cross Site Scripting (Reflected)	Wysoki	1
Vulnerable JS Library	Średni	1
Content Security Policy (CSP) Header Not Set	Średni	36
Absence of Anti-CSRF Tokens	Średni	40
Cookie No HttpOnly Flag	Niski	216
Cookie with SameSite Attribute None	Niski	110
Cookie without SameSite Attribute	Niski	109
Cookie Without Secure Flag	Niski	108
Strict-Transport-Security Header Not Set	Niski	106
X-Content-Type-Options Header Missing	Niski	101
Session Management Response Identified	Informacyjny	976
User Agent Fuzzer	Informacyjny	312
Modern Web Application	Informacyjny	36
Re-examine Cache-control Directives	Informacyjny	33
Information Disclosure - Suspicious Comments	Informacyjny	27
User Controllable HTML Element Attribute (Potential XSS)	Informacyjny	16

Narzędzie Zed Attack Proxy wykazało się największą skutecznością w wykrywaniu różnorodnych zagrożeń w aplikacji Gin & Juice Shop. Zidentyfikowało ono jedno zagrożenie wysokiego poziomu – podatność na atak Cross-Site Scripting, które pozwala atakującemu na wstrzyknięcie złośliwego kodu JavaScript. Oprócz tego ZAP wykrył trzy zagrożenia średniego poziomu, w tym

obecność podatnej biblioteki JavaScript oraz brak odpowiedniej konfiguracji polityki bezpieczeństwa treści (CSP). ZAP zidentyfikował również liczne problemy niskiego poziomu, głównie związane z niewłaściwą konfiguracją plików cookie, co może prowadzić do wycieków informacji lub ataków na sesje użytkowników.

Tabela 2: Wyniki skanowania aplikacji Gin & Juice Shop narzędziem Wapiti

Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
Content Security Policy Configuration	Średni	1
HttpOnly Flag cookie	Niski	2
HTTP Strict Transport Security (HSTS)	Niski	1
MIME Type Confusion	Niski	1
Secure Flag cookie	Niski	1

Wapiti, w porównaniu do pozostałych skanerów, dostarczył najmniej informacji o potencjalnych zagrożeniach. Wykrył on pięć problemów, z których cztery zostały sklasyfikowane jako zagrożenia średniego poziomu. Skupiły się one na kwestiach konfiguracyjnych, takich jak brak flagi HttpOnly dla plików cookie, co może zwiększyć ryzyko ataków Cross-Site Scripting (XSS), oraz problemy z konfiguracją polityki bezpieczeństwa treści i protokołu HTTP Strict Transport Security (HSTS). Mimo że Wapiti nie wykrył żadnych zagrożeń wysokiego poziomu, zidentyfikowane przez niego problemy wskazują na obszary, które wymagają uwagi w celu poprawy ogólnego poziomu bezpieczeństwa aplikacji.

Tabela 3: Wyniki skanowania aplikacji Gin & Juice Shop narzędziem Skipfish

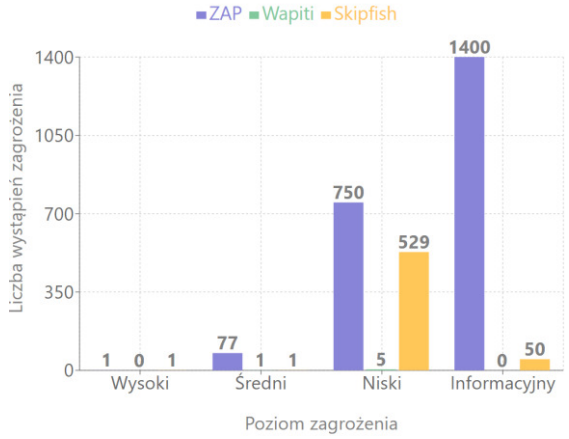
Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
Query injection vector	Wysoki	1
HTTP response header splitting	Średni	1
HTML form with no apparent XSRF protection	Niski	528
HTTP header injection vector	Niski	1
Incorrect or missing charset	Informacyjny	17
Hidden files / directories	Informacyjny	13
New HTTP cookie added	Informacyjny	5
HTML form (not classified otherwise)	Informacyjny	3
Unknown form field (can't autocomplete)	Informacyjny	3
Numerical filename - consider enumerating	Informacyjny	2
Incorrect or missing MIME type	Informacyjny	2
New 'X-*' header value seen	Informacyjny	2
Server error triggered	Informacyjny	1
New 404 signature seen	Informacyjny	1
SSL certificate issuer information	Informacyjny	1

Skipfish zajął pozycję pośrednią pod względem liczby i różnorodności wykrytych zagrożeń. Najpoważniejszym wykrytym problemem było zagrożenie wysokiego poziomu związane z potencjalną podatnością na wstrzykiwanie zapytań, co może prowadzić do nieautoryzowanego dostępu do danych lub manipulacji bazą danych. Skaner wykrył również jedno zagrożenie średniego poziomu – możliwość dzielenia nagłówków odpowiedzi HTTP. To zagrożenie może być wykorzystane przez atakującego do manipulowania nagłówkami HTTP, a w konsekwencji umożliwić ataki Cross-Site Scripting lub przekierowywać użytkowników na złośliwe strony bez ich wiedzy. Skipfish zwrócił szczególną uwagę na brak ochrony przed atakami CSRF, identyfikując 528 formularzy HTML bez widocznej ochrony atakami typu Cross-Site Request Forgery (XSRF), które pozwalają atakującym na wykonywanie działań w imieniu zalogowanego użytkownika bez jego wiedzy.

Analizując wykres umieszczony na Rysunku 1, można zauważyć, że ZAP, reprezentowany na wykresie kolorem fioletowym, wykazał się największą czułością w wykrywaniu potencjalnych zagrożeń. Narzędzie to zidentyfikowało jedno zagrożenie wysokiego poziomu, 77 zagrożeń średniego poziomu, 750 zagrożeń niskiego poziomu oraz aż 1400 zagrożeń informacyjnych. Ta wszechstronność Zed Attack Proxy w wykrywaniu problemów na różnych poziomach sugeruje, że może on być najbardziej kompleksowym narzędziem spośród testowanych.

W przeciwieństwie do narzędzia ZAP, Wapiti, oznaczony na wykresie kolorem zielonym, wykazał się najmniejszą liczbą wykrytych zagrożeń. Skaner ten zidentyfikował tylko jedno zagrożenie średniego poziomu oraz pięć zagrożeń niskiego poziomu, nie wykrywając żadnych zagrożeń o wysokim poziomie ryzyka.

Skipfish, przedstawiony na wykresie kolorem żółtym, zajął pozycję pośrednią między ZAP a Wapiti pod względem liczby wykrytych zagrożeń. Narzędzie to zidentyfikowało jedno zagrożenie wysokiego poziomu, jedno średniego poziomu, znaczącą liczbę 529 zagrożeń niskiego poziomu oraz 50 zagrożeń informacyjnych. Wyniki skanera Skipfish, szczególnie w kategorii niskiego poziomu zagrożeń, sugerują, że narzędzie to może być szczególnie skuteczne w wykrywaniu mniej krytycznych, ale potencjalnie licznych problemów bezpieczeństwa.



Rysunek 1: Wykres przedstawiający liczbę wykrytych podatności w podziale na poziomy zagrożenia w aplikacji Gin & Juice Shop.

Analiza wyników skanowania aplikacji Gin & Juice Shop przedstawiona w Tabeli 4 ujawnia ograniczoną

skuteczność narzędzi w wykrywaniu celowo wprowadzonych podatności.

Skaner ZAP okazał się jednym z dwóch najbardziej efektywnych narzędzi, wykrywając dwie kluczowe podatności: Cross-Site Scripting (reflected) i podatną bibliotekę JavaScript. Skipfish również wykrył dwie podatności: SQL Injection oraz problemy związane z wstrzykiwaniem nagłówków HTTP. Oba narzędzia osiągnęły identyczne wyniki w kluczowych metrykach: czułość na poziomie 11,76%, precyzję 100% i F1-score 21,05%, co zostało przedstawione w Tabeli 5.

Czułość na poziomie 11,76% dla ZAP i Skipfish potwierdza ich ograniczoną zdolność do wykrywania rzeczywistych podatności w tej złożonej aplikacji. Precyzja wynosząca 100% wskazuje na brak fałszywych alarmów. Wszystkie wykryte podatności okazały się rzeczywistymi problemami bezpieczeństwa.

Wapiti nie wykrył żadnej podatności, co potwierdza, że jego skuteczność w kontekście tej nowoczesnej aplikacji jest ograniczona. Konsekwentnie, wszystkie metryki dla Wapiti wyniosły 0%.

Niektóre podatności, jak Cross-site scripting (reflected) i DOM data manipulation, występowały w aplikacji wielokrotnie, ale zostały wykryte tylko częściowo lub wcale. To dodatkowo tłumaczy niską czułość narzędzi i podkreśla ograniczenia automatycznych skanerów w wykrywaniu wszystkich instancji danej podatności.

Tabela 4: Wykrywalność celowych podatności w aplikacji Gin & Juice Shop

Nazwa podatności	ZAP	Wapiti	Skipfish	Rzeczywista liczba wystąpień zagrożenia
Base64-encoded data in parameter	0	0	0	1
Request URL override	0	0	0	1
Client-side prototype pollution	0	0	0	1
Client-side template injection	0	0	0	2
Cross-site scripting (DOM-based)	0	0	0	1
Open redirection (DOM-based)	0	0	0	1
Cross-site scripting (reflected)	1	0	0	3
DOM data manipulation (reflected DOM-based)	0	0	0	2
HTTP response header injection	0	0	1	1
Link manipulation (reflected DOM-based)	0	0	0	1
SQL injection	0	0	1	1
XML external entity injection	0	0	0	1
Vulnerable JavaScript dependency	1	0	0	1

Tabela 5: Metryki oceny skuteczności narzędzi dla aplikacji Gin & Juice Shop

Metryka	ZAP	Wapiti	Skipfish
Czułość	11,76%	0%	11,76%
Precyzja	100%	0%	100%
F1-score	21,05%	0%	21,05%

5.2. Wyniki skanowania aplikacji DSVW

W Tabelach 6, 7 i 8 zostały przedstawione wyniki skanowania aplikacji internetowej DSVW.

Tabela 6: Wyniki skanowania aplikacji DSVW narzędziem Zed Attack Proxy

Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
SQL Injection - SQLite	Wysoki	4
Cross Site Scripting (Reflected)	Wysoki	3
External Redirect	Wysoki	1
Path Traversal	Wysoki	1
Remote File Inclusion	Wysoki	1
Content Security Policy (CSP) Header Not Set	Średni	15
Missing Anti-clickjacking Header	Średni	12
Buffer Overflow	Średni	5
Server Leaks Version Information via “Server” HTTP Response Header Field	Niski	19
X-Content-Type-Options Header Missing	Niski	15
Application Error Disclosure	Niski	1
User Agent Fuzzer	Informacyjny	22
Information Disclosure - Suspicious Comments	Informacyjny	15
Information Disclosure - Sensitive Information in URL	Informacyjny	2
Authentication Request Identified	Informacyjny	1
User Controllable Charset	Informacyjny	1
User Controllable HTML Element Attribute (Potential XSS)	Informacyjny	1

Skaner ZAP wykazał się skutecznością w wykrywaniu szerokiego spektrum zagrożeń. Zidentyfikował on cztery różne typy zagrożeń wysokiego poziomu, w tym cztery przypadki podatności na wstrzyknięcie kodu SQL specyficzne dla bazy danych SQLite. Wystąpiły również trzy przypadki podatności na ataki Cross-Site Scripting (XSS), co może zostać wykorzystane do wykradania danych użytkowników lub przejmowania sesji.

Wynik skanowania wykazał również pojedynczy przypadek External Redirect, który może pozwolić atakującemu na przekierowanie użytkownika do niebezpiecznej witryny zewnętrznej, co zwiększa ryzyko phishingu i innych oszustw. W aplikacji wykryto także jedno zagrożenie typu Remote File Inclusion, które pozwala na zdalne włączenie pliku do aplikacji, otwierając drzwi do potencjalnego wykonania złośliwego kodu.

W obszarze zagrożeń średniego poziomu narzędzie wykryło liczne braki w konfiguracji nagłówków bezpieczeństwa, takie jak brak polityki CSP oraz brak nagłówków Anti-clickjacking. Co więcej, pojawiły się zagrożenia związane z przepełnieniem bufora oraz wyciekiem informacji serwera poprzez nagłówek HTTP.

Zagrożenia niskiego poziomu dotyczyły przede wszystkim brakujących nagłówków X-Content-Type-

Options oraz ujawnienia wersji serwera, co może dostarczyć cennych informacji atakującemu. Narzędzie zidentyfikowało także kilka zagrożeń informacyjnych, takich jak podatności związane z kontrolą użytkownika nad atrybutami HTML, co potencjalnie może prowadzić do ataków XSS, oraz różne przypadki wycieku informacji.

Tabela 7: Wyniki skanowania aplikacji DSVW narzędziem Wapiti

Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
SQL Injection	Wysoki	3
Path Traversal	Wysoki	2
Server Side Request Forgery	Wysoki	2
Command execution	Wysoki	1
Stored Cross Site Scripting	Wysoki	1
Reflected Cross Site Scripting	Wysoki	2
Open Redirect	Wysoki	2
Content Security Policy Configuration	Średni	1
Clickjacking Protection	Średni	1
MIME Type Confusion	Niski	1

Wapiti, choć wykrył mniej zagrożeń niż ZAP, to zidentyfikował więcej zagrożeń wysokiego poziomu. Narzędzie to wykryło trzy przypadki podatności na wstrzyknięcie SQL, które mogą prowadzić do nieautoryzowanego dostępu do bazy danych. Dodatkowo pojawiły się przypadki Path Traversal i Server Side Request Forgery (SSRF), które pozwalają na nieautoryzowany dostęp do wewnętrznych zasobów serwera. Odkryto także jedno zagrożenie związane z wstrzykiwaniem poleceń systemu operacyjnego, co stanowi duże ryzyko przejęcia kontroli nad serwerem. Wapiti wykrył również podatność na ataki Stored Cross-Site Scripting i Reflected Cross-Site Scripting, które umożliwiają atakującym wstrzyknięcie złośliwego kodu JavaScript. Zostały zidentyfikowane dwie podatności na Open Redirect, które mogą umożliwić przekierowanie użytkowników na złośliwe strony, zwiększając ryzyko phishingu i innych ataków socjotechnicznych.

Oprócz zagrożeń o wysokim priorytecie, narzędzie to wykryło również zagrożenia o średnim i niskim poziomie, w tym brak odpowiedniej konfiguracji polityki bezpieczeństwa treści (CSP) oraz brak ochrony przed clickjackingiem, które choć stanowią mniejsze ryzyko, nadal mogą prowadzić do potencjalnych naruszeń bezpieczeństwa.

Tabela 8: Wyniki skanowania aplikacji DSVW narzędziem Skipfish

Zagrożenie	Poziom zagrożenia	Liczba wystąpień zagrożenia
File inclusion	Wysoki	4
Remote file inclusion	Wysoki	2
Query injection vector	Wysoki	2
Interesting server message	Średni	11
Incorrect or missing charset (higher risk)	Średni	9
XSS vector in document body	Średni	6
Incorrect or missing MIME type (higher risk)	Średni	7
Signature match detected (higher risk)	Średni	2
Interesting file	Średni	2
Generic MIME type (higher risk)	Średni	1
HTTP response header splitting	Średni	1
XSS vector via arbitrary URLs	Średni	1
Redirection to attacker-supplied URLs	Niski	2
HTTP header injection vector	Niski	1
User-controlled response prefix (BOM / plugin attacks)	Niski	1
Incorrect or missing charset (low risk)	Informacyjny	412
Incorrect or missing MIME type (low risk)	Informacyjny	14
Server error triggered	Informacyjny	11
Generic MIME used (low risk)	Informacyjny	5
Hidden files / directories	Informacyjny	4
Resource not directly accessible	Informacyjny	1
New 404 signature seen	Informacyjny	1
New ‘X-*’ header value seen	Informacyjny	1
New ‘Server’ header value seen	Informacyjny	1

Skipfish dostarczył najbardziej szczegółowego raportu, wykrywając szeroki zakres zagrożeń o różnych poziomach ryzyka. Najpoważniejsze zagrożenia obejmują cztery przypadki File Inclusion oraz dwa przypadki Remote File Inclusion, które mogą umożliwić atakującemu dostęp do nieautoryzowanych plików serwera. Dodatkowo zidentyfikowano dwa przypadki Query Injection Vector, gdzie atakujący może wstrzykiwać złośliwe dane w zapytaniach do serwera, co może prowadzić do nieautoryzowanego dostępu lub manipulacji danymi.

Na poziomie średnim Skipfish wykrył liczne problemy, w tym nieprawidłowe lub brakujące zestawy znaków i typów MIME oraz wektory XSS w treści dokumentu. Narzędzie to zwróciło również uwagę na niższe poziomy ryzyka, takie jak przekierowania do adresów URL dostarczonych przez atakującego i możliwość wstrzykiwania nagłówków HTTP.

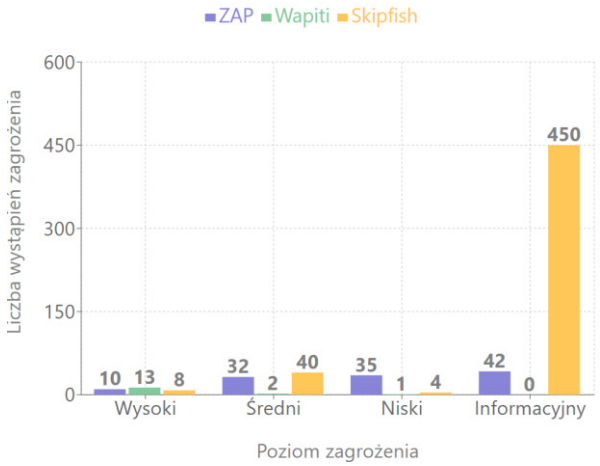
Z wykresu przedstawionego na Rysunku 2 można odczytać, że w kategorii zagrożeń wysokiego poziomu, wszystkie trzy skanery wykazały się podobną skutecznością. ZAP wykrył 10 takich zagrożeń, podczas gdy Wapiti 13, a Skipfish 8. Ta zbliżona liczba sugeruje, że wszystkie narzędzia są w stanie efektywnie identyfikować najpoważniejsze luki w zabezpieczeniach, co jest ich kluczowym zadaniem. Niewielka przewaga Wapiti może świadczyć o jego nieco większej czułości lub szerszym zakresie analizy w tej krytycznej kategorii.

Znaczące różnice pojawiają się w wykrywaniu zagrożeń średniego poziomu. ZAP i Skipfish okazały się tutaj zdecydowanie skuteczniejsze, wykrywając odpowiednio 32 i 40 zagrożeń, podczas gdy Wapiti zidentyfikował tylko dwa. Skuteczność w tej kategorii jest istotna, gdyż zagrożenia średniego poziomu, choć mniej krytyczne, nadal mogą stanowić poważne ryzyko dla bezpieczeństwa aplikacji.

W zakresie zagrożeń niskiego poziomu ZAP wykazał się największą skutecznością, wykrywając 35 przypadków, podczas gdy Skipfish zidentyfikował cztery zagrożenia, a Wapiti tylko jedno. Choć zagrożenia niskiego poziomu nie stanowią bezpośredniego ryzyka, ich

identyfikacja może być cenna dla kompleksowego zabezpieczenia aplikacji.

Najbardziej zauważalna różnica w skuteczności skanerów widoczna jest w kategorii zagrożeń informacyjnych. Skipfish zdecydowanie dominuje, wykrywając aż 450 takich przypadków, co znacząco przewyższa wyniki ZAP (42) i Wapiti (0). Ta dysproporcja może świadczyć o bardzo szczegółowym podejściu Skipfish do analizy i raportowania, obejmującym szeroki zakres potencjalnych problemów, które mogą nie stanowić bezpośredniego zagrożenia, ale mogą być cenne z perspektywy ogólnej oceny bezpieczeństwa i zgodności z najlepszymi praktykami.



Rysunek 2: Wykres przedstawiający liczbę wykrytych podatności w podziale na poziomy zagrożenia w aplikacji DSVW.

Analiza wyników skanowania przeprowadzonego przez trzy narzędzia: ZAP, Wapiti i Skipfish zawarta w Tabeli 9 ujawniła zróżnicowaną skuteczność w wykrywaniu różnych typów podatności w aplikacji internetowej DSVW.

Wyniki zawarte w Tabeli 10 pokazują, że ZAP i Wapiti okazały się najbardziej efektywnymi narzędziami, osiągając identyczną czułość na poziomie 37,5%. Oznacza to, że oba narzędzia były w stanie wykryć ponad jedną trzecią wszystkich rzeczywistych podatności obecnych w aplikacji. Skipfish wykazał się nieco niższą czułością, wynoszącą 33,33%, co wskazuje na jego ograniczoną zdolność do wykrywania niektórych typów zagrożeń.

Wapiti wyróżnił się najwyższą precyzją, osiągając wynik 69,23%, co oznacza, że blisko dwie trzecie wykrytych podatności stanowiło rzeczywiste zagrożenia. To sugeruje, że Wapiti jest najbardziej wiarygodnym narzędziem pod względem minimalizacji fałszywych alarmów.

Wykrywanie Cross-Site Scripting było zróżnicowane. Wszystkie skanery wykryły reflected XSS: ZAP 3 przypadki, Wapiti 2, a Skipfish 1, mimo że aplikacja zawierała tylko jeden przypadek tego zagrożenia. Tylko Wapiti wykrył 1 przypadek stored XSS, podczas gdy Skipfish błędnie zidentyfikował 5 przypadków. Żaden ze skanerów nie wykrył DOM-based XSS ani XSS w kontekście JSONP.

Path Traversal został zidentyfikowany przez wszystkie skanery, z ZAP wykrywającym 1 przypadek, Wapiti 2, a Skipfish 4, pomimo że w aplikacji znajdowało się tylko jedno wystąpienie tej podatności. ZAP i Skipfish wykryły również podatność na zdalne dołączanie plików.

Żaden ze skanerów nie wykrył kilku zaawansowanych zagrożeń, w tym Login Bypass, HTTP Parameter Pollution, XML External Entity, Blind XPath Injection, Cross Site Request Forgery. Ta obserwacja wyjaśnia, dlaczego nawet najwyższa czułość (37,5% dla ZAP i Wapiti) jest stosunkowo niska i podkreśla ograniczenia automatycznych skanerów w wykrywaniu złożonych podatności.

Tabela 9: Wykrywalność celowych podatności w aplikacji DSVW

Nazwa podatności	ZAP	Wapiti	Skipfish	Rzeczywista liczba wystąpień zagrożenia
SQL Injection	4	3	2	3
Login Bypass	0	0	0	1
HTTP Parameter Pollution	0	0	0	1
Cross Site Scripting (reflected)	3	2	1	1
Cross Site Scripting (stored)	0	1	6	1
Cross Site Scripting (DOM)	0	0	0	1
Cross Site Scripting (JSONP)	0	0	0	1
XML External Entity (local)	0	0	0	1
XML External Entity (remote)	0	0	0	1
Server Side Request Forgery	0	2	0	1
Blind XPath Injection (boolean)	0	0	0	1
Cross Site Request Forgery	0	0	0	1
Clickjacking	12	1	0	1
Unvalidated Redirect	1	2	2	1
Arbitrary Code Execution	0	0	0	1
Full Path Disclosure	0	0	0	1
Source Code Disclosure	0	0	0	1
Path Traversal	1	2	4	1
File Inclusion (remote)	1	0	2	1
HTTP Header Injection (phishing)	0	0	1	1
Component with Known Vulnerability (pickle)	0	0	0	1
Denial of Service (memory)	5	0	0	1

Tabela 10: Metryki oceny skuteczności narzędzi dla aplikacji DSVW

Metryka	ZAP	Wapiti	Skipfish
Czułość	37,5%	37,5%	33,33%
Precyzja	33,33%	69,23%	44,44%
F1-score	35,29%	48,65%	38,1%

6. Wnioski

Przeprowadzone badania nad skutecznością narzędzi do testowania bezpieczeństwa aplikacji internetowych

dostarczyły cennych informacji na temat możliwości i ograniczeń trzech popularnych skanerów: Zed Attack Proxy (ZAP), Wapiti i Skipfish.

Hipoteza badawcza zakładała, że Zed Attack Proxy osiągnie najlepsze wyniki w wykrywaniu zagrożeń o wysokim poziomie zagrożenia. Analiza wyników częściowo potwierdza tę hipotezę. W aplikacji DSVW ZAP wykrył 10 zagrożeń wysokiego poziomu, ale to Wapiti zidentyfikował ich najwięcej, bo aż 13. Z kolei w aplikacji Gin & Juice Shop, ZAP i Skipfish zidentyfikowały po jednym zagrożeniu, podczas gdy Wapiti nie wykrył żadnego. Ponadto, ZAP zidentyfikował najszersze spektrum zagrożeń, w tym podatności na SQL Injection i Cross-Site Scripting (XSS).

W przypadku celowo wprowadzonych podatności w aplikacji Gin & Juice Shop, narzędzia ZAP i Skipfish wykazały podobną skuteczność, podczas gdy Wapiti nie wykrył żadnych podatności. Dla aplikacji DSVW wszystkie trzy narzędzia osiągnęły zbliżone wyniki w wykrywaniu podatności, przy czym Wapiti wykazał najwyższą precyzję, a ZAP i Wapiti miały najwyższą czułość.

Badanie ujawniło zróżnicowaną skuteczność testowanych narzędzi. Każde z nich wykazało się unikalnymi mocnymi stronami. ZAP okazał się wszechstronny, Skipfish dostarczył najbardziej szczegółowych raportów, a Wapiti, choć wykrył mniej zagrożeń, zidentyfikował niektóre zaawansowane podatności pominięte przez inne narzędzia. Ta obserwacja podkreśla komplementarność narzędzi i sugeruje, że użycie kilku skanerów może zapewnić bardziej kompleksową ocenę bezpieczeństwa aplikacji. Każde narzędzie wykryło unikalne zagrożenia, co wskazuje, że łączne wykorzystanie różnych skanerów może zwiększyć ogólną skuteczność testów bezpieczeństwa.

Jednocześnie badanie ujawniło pewne ograniczenia testowanych narzędzi. Wszystkie miały trudności z wykryciem bardziej złożonych lub specyficznych dla aplikacji podatności, szczególnie tych związanych z manipulacją danymi po stronie klienta i zaawansowanymi atakami DOM-based. Ta obserwacja podkreśla potrzebę dalszego rozwoju narzędzi automatycznych w kierunku wykrywania bardziej subtelnych i kontekstowych zagrożeń bezpieczeństwa.

Analizując wyniki, należy wziąć pod uwagę możliwe źródła błędów. Gin & Juice Shop oraz DSVW, choć zawierają celowo wprowadzone podatności, mogą nie reprezentować pełnego spektrum współczesnych aplikacji internetowych. Wyniki mogą być zależne od konkretnych ustawień i konfiguracji użytych w badaniu, takich jak głębokość skanowania, czas trwania skanów lub zastosowanie autoryzacji w trakcie skanowania.

Rozszerzenie zestawu testowanych narzędzi o inne popularne skanery bezpieczeństwa mogłoby dostarczyć bardziej kompleksowego obrazu skuteczności dostępnych rozwiązań. Badanie wpływu różnych konfiguracji na skuteczność skanowania mogłoby pomóc w opracowaniu najlepszych praktyk dla ich wykorzystania.

Podsumowując, badanie to podkreśla zarówno potencjał, jak i ograniczenia narzędzi do testowania bezpieczeństwa aplikacji internetowych. Wyniki sugerują, że najlepsze praktyki powinny obejmować wykorzystanie kilku narzędzi oraz połączenie automatycznego skanowania z analizą manualną. Dalsze badania w tej dziedzinie są kluczowe dla poprawy bezpieczeństwa aplikacji internetowych w obliczu stale ewoluujących zagrożeń cybernetycznych.

Literatura

[1] Y. Makino, V. Klyuev, Evaluation of web vulnerability scanners, In 2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (2015) 399-402, https://doi.org/10.1109/IDAACS.2015.7340766.

[2] B. Zukran, M. M. Siraj, Performance Comparison on SQL Injection and XSS Detection using Open Source Vulnerability Scanners, In 2021 International Conference on Data Science and Its Applications (2021) 61-65, https://doi.org/10.1109/ICoDSA53588.2021.9617484.

[3] D. Sagar, S. Kukreja, J. Brahma, S. Tyagi, P. Jain, Studying open source vulnerability scanners for vulnerabilities in web applications, IIOAB Journal 9(2) (2018) 43-49.

[4] R. Amankwah, J. Chen, P. K. Kudjo, D. Towey, An empirical comparison of commercial and open-source web vulnerability scanners, Software: Practice and Experience 50(9) (2020) 1842-1857, https://doi.org/10.1002/spe.2870.

[5] A. Al Anhar, Y. Suryanto, Evaluation of Web Application Vulnerability Scanner for Modern Web Application, In 2021 International Conference on Artificial Intelligence and Computer Science Technology (2021) 200-204, https://doi.org/10.1109/ICAICST53116.2021.9497831.

[6] A. Kondraciuk, A. Bartos, B. Pańczyk, Comparative analysis of the effectiveness of OWASP ZAP, Burp Suite, Nikto and Skipfish in testing the security of web applications, Journal of Computer Sciences Institute 24 (2022) 176–180, https://doi.org/10.35784/jcsi.2929.

[7] Dokumentacja narzędzia Zed Attack Proxy, https://www.zaproxy.org, [12.06.2024].

[8] Dokumentacja narzędzia Wapiti, https://wapiti-scanner.github.io, [12.06.2024].

[9] Kod źródłowy i dokumentacja narzędzia Skipfish, https://gitlab.com/kalilinux/packages/skipfish, [29.08.2024].

[10] Dokumentacja narzędzia Skipfish, https://www.kali.org/tools/skipfish/, [29.08.2024].

[11] Aplikacja Gin&Juice Shop, https://ginandjuice.shop/about, [29.08.2024].

[12] Kod źródłowy i dokumentacja aplikacji DSVW, https://github.com/stamparm/DSVW, [12.06.2024].