

# Testy mechanizmu logowania i ochrona przed brute-force

## 1 Informacje ogólne






**Narzędzie:** OWASP ZAP v2.16.1 (Fuzzer)


**Data skanu:** 2025-05-03

**Zakres:** <https://inwood.pl/wp-login.php>

**Metoda:** Fuzzing pól logowania (lista ~100 najpopularniejszych haseł)

## 2 Wyniki testów

 Żądanie	 Próbek	 Kod HTTP	 Rozmiar odpowiedzi	 Alerty
Oryginalne	1	200 OK	9 778 B	0
Fuzzowane	~100	200 OK	9 052 B (stałe)	0

 **Brak nowych alertów** – żadne payloady nie wygenerowały błędów 500 ani wykrytych luk (SQLi, XSS, RCE).

## 3 Wnioski

- Odporność na proste brute-force**
  - Wszystkie próby zwracały 200 OK i identyczną stronę.
- Brak wycieku informacji o użytkowniku**
  - Nie ma różnic w treści ani kodzie odpowiedzi dla poprawnych/niepoprawnych loginów.
- Sugerowana ochrona wbudowana**
  - Najpewniej działa limit prób lub jednolita odpowiedź na nieudaną autoryzację.

## 4 Rekomendacje

### ● Wzmocnienie ochrony przed brute-force

- Dodaj **CAPTCHA** po 3–5 nieudanych próbach.
- Wdróż **blokadę konta** (np. 15 min) po określonej liczbie błędnych logowań.
- Zastosuj **opóźnienia** (rate-limit / delay) rosnące z każdą próbą.

### ● Ujednolicenie odpowiedzi

- Zwracaj zawsze **ten sam** kod HTTP (np. 200) i **identyczny** widok formularza przy błędnych danych, aby nie ujawniać istnienia konta.

### ● Monitorowanie i alarmowanie

- Zaloguj wszystkie próby logowania (udane i nieudane).
- Generuj alerty przy nietypowej liczbie nieudanych prób z jednego IP.

## 5 Narzędzia użyte

- OWASP ZAP Fuzzer v2.16.1