Sfruttamento Vulnerabilità Web su DVWA

1. Introduzione

Lo scopo di questa esercitazione è comprendere le meccaniche di due delle vulnerabilità web più critiche: Cross-Site Scripting (XSS) e SQL Injection (SQLi).

Per farlo in un ambiente controllato e sicuro utilizzerò la DVWA (Damn Vulnerable Web Application) un'applicazione web intenzionalmente vulnerabile progettata per scopi didattici. L'obiettivo è dimostrare come la mancanza di input sanitization (pulizia degli input utente) permetta a un attaccante di eseguire codice arbitrario (XSS) o manipolare il database (SQLi).

2. Strumenti Utilizzati

- Host (Attaccante): Kali Linux
- Target (Vittima): Macchina Virtuale DVWA
- Virtualizzazione: VirtualBox
- Browser: Firefox (in Kali Linux).

3. Svolgimento dell'Esercizio

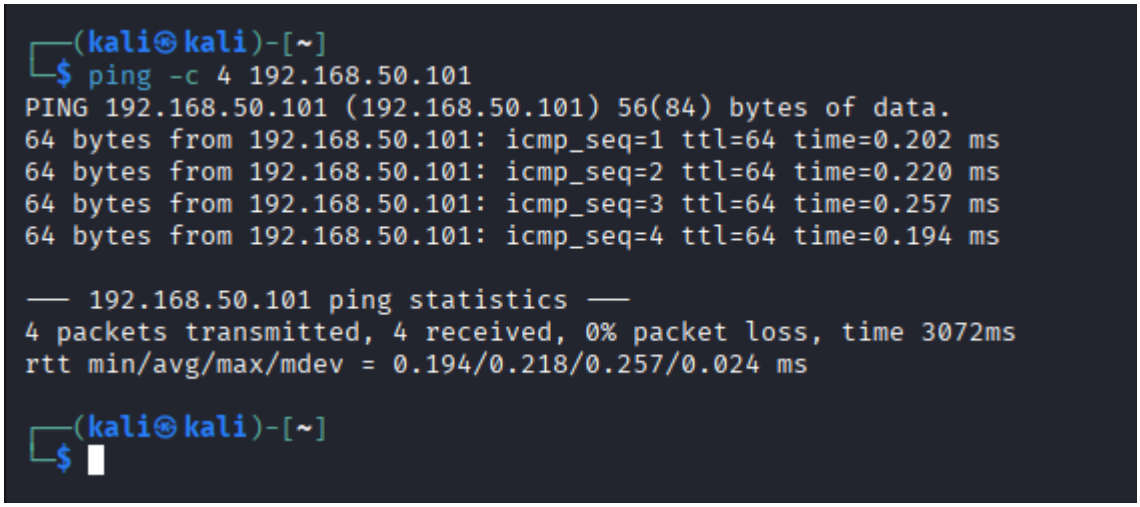
Configurazione del Laboratorio (Networking)

Prima di attaccare è fondamentale che le due macchine possano comunicare tra loro.

Procedura:

1. Assicurarsi che entrambe le macchine virtuali siano impostate sulla stessa rete.
2. Sulla macchina DVWA avviare il sistema e annotare l'indirizzo IP mostrato al login.
3. Sulla macchina Kali aprire il terminale.
4. Verificare la connettività con il comando ping

Il comando ping utilizza il protocollo ICMP per verificare se l'host di destinazione è raggiungibile. Se non ricevo risposta allora i pacchetti sono andati persi.



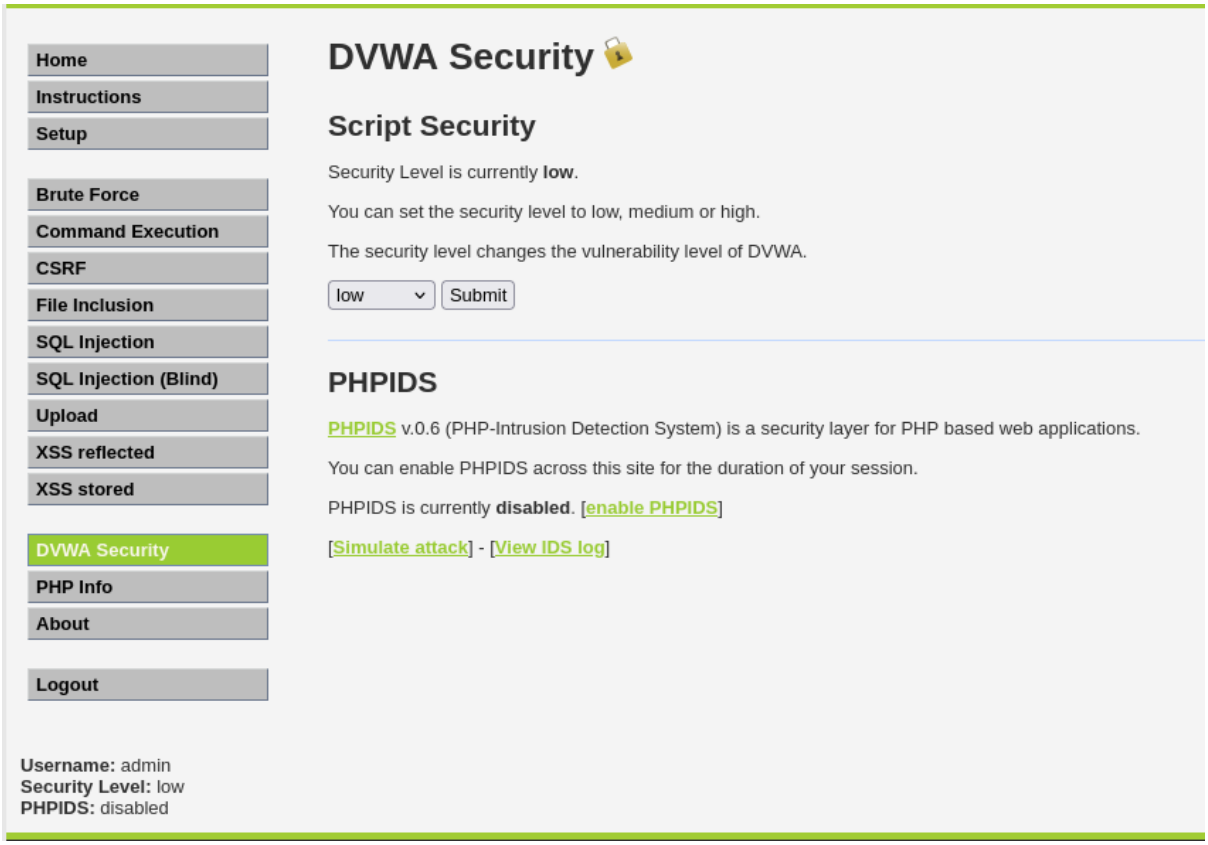
Impostazione della DVWA

Ora devo preparare l'ambiente bersaglio affinché sia vulnerabile alle tecniche di base.

Procedura:

1. Su Kali Linux, aprire il browser Firefox.
2. Nella barra degli indirizzi digitare l'IP della DVWA
3. Effettuare il login con le credenziali di default
4. Nel menu cliccare su "DVWA Security"
5. Impostare il "Security Level" su Low

Impostando il livello su "Low" disabilito completamente i filtri di sicurezza lato server. In questo modo, l'applicazione accetterà qualsiasi input senza controllarlo permettendomi di vedere chiaramente come funziona l'attacco "puro".



Sfruttamento XSS Reflected

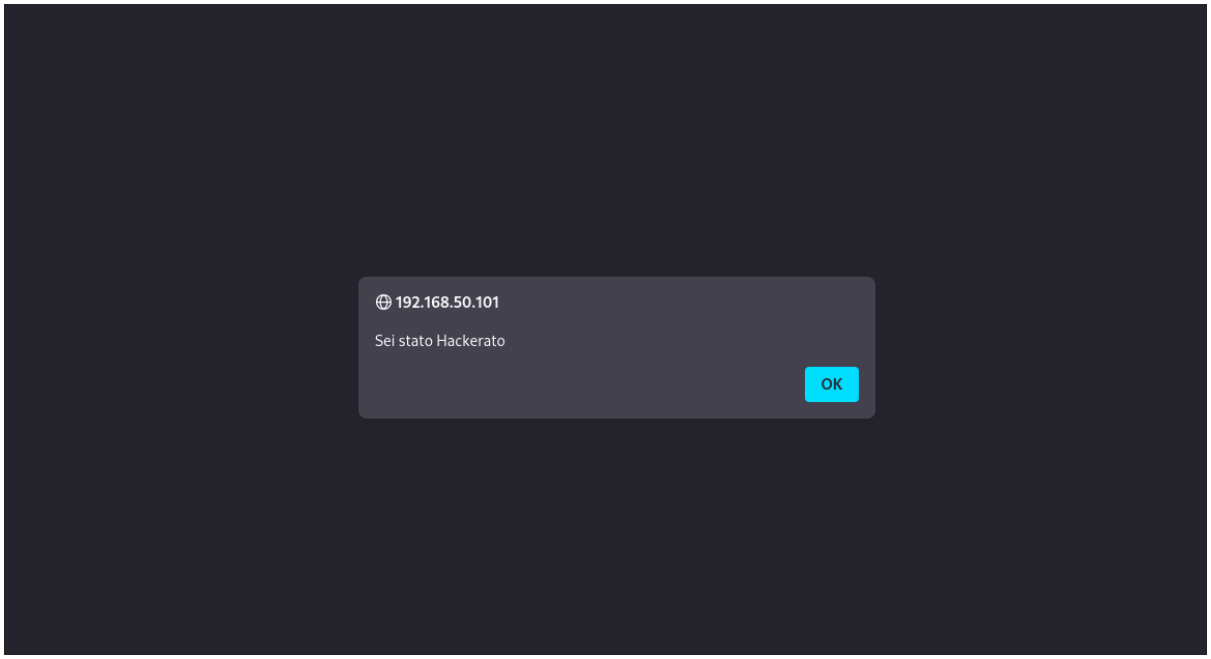
Obiettivo: Far eseguire al browser della vittima uno script JavaScript non autorizzato.

Procedura:

1. Nel menu a sinistra vado su "XSS (Reflected)".
2. Inserisco il codice malevolo

3. Cliccare su Submit.

Apparirà una finestra popup (alert box) con la scritta "Sei stato Kackerato".



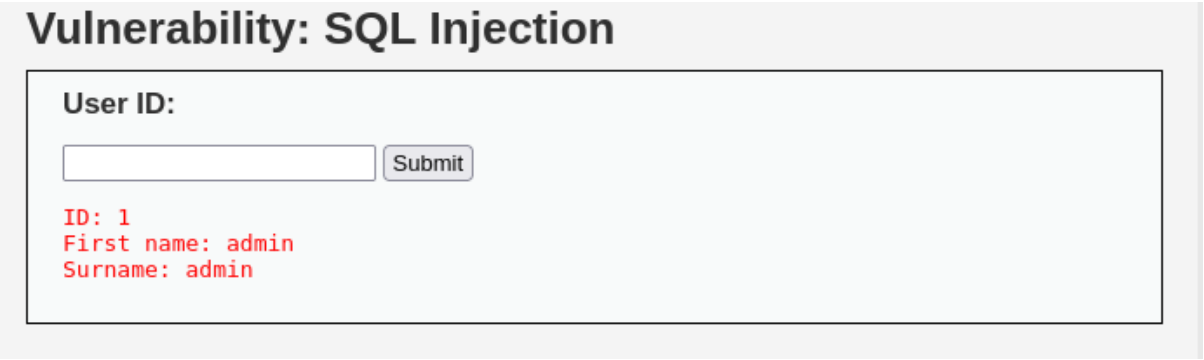
Questa vulnerabilità si verifica perché l'applicazione PHP prende l'input dell'utente la variabile name e lo "riflette" (lo stampa) direttamente nella pagina HTML di risposta senza codificarlo. Il browser interpreta i tag <script> non come testo semplice ma come istruzioni eseguibili. In uno scenario reale al posto di un semplice alert(), un attaccante potrebbe inserire script per rubare i cookie di sessione (document.cookie) e account.

Sfruttamento SQL Injection (Non-Blind)

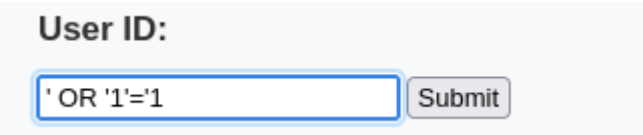
Manipolare la query al database per estrarre informazioni non autorizzate

Procedura:

1. Nel menu clicco su "SQL Injection".
2. L'applicazione chiede "User ID" provo prima un uso legittimo inserendo 1 e premendo Submit (restituisce "admin").



3. Ora sfrutto la vulnerabilità inserendo il seguente payload



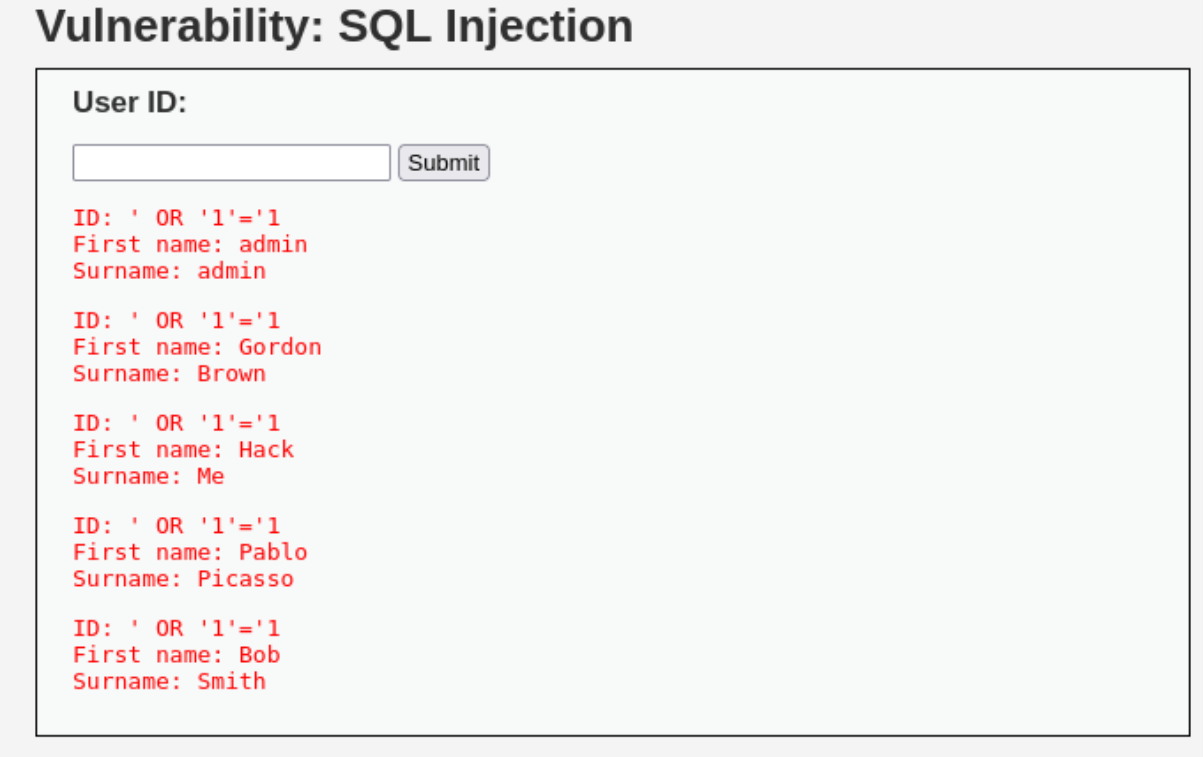
4. Clicco su Submit.

L'applicazione restituirà l'elenco di tutti gli utenti presenti nel database non solo quello richiesto.

Il codice vulnerabile lato server è questo SELECT prima_nome, cognome FROM utenti WHERE user_id = '$id';

Inserendo il payload la query diventa SELECT prima_nome, cognome FROM utenti WHERE user_id = '' OR '1'='1';

In SQL l'operatore OR rende vera l'intera condizione se almeno una delle parti è vera. Poiché '1'='1' è sempre vero il database ignora il filtro sull'ID e restituisce tutte le righe della tabella.



4. Conclusione

L'esercitazione ha dimostrato con successo come configurare un ambiente di test e sfruttare vulnerabilità critiche.

Dall'analisi emerge che:

1. La XSS è possibile quando l'applicazione si fida ciecamente dell'input utente e lo inserisce nel DOM della pagina.
2. La SQL Injection avviene quando l'input utente viene concatenato direttamente nelle stringhe di comando SQL.

Per mitigare questi rischi in un ambiente di produzione (livello "Impossible" su DVWA), è necessario:

- Per XSS: Utilizzare funzioni di HTML Entity Encoding (es. htmlspecialchars() in PHP) per convertire i caratteri speciali in testo innocuo.
- Per SQLi: Utilizzare esclusivamente Prepared Statementsche separano la struttura del codice dai dati.