

LABORATORIO 14 GENNAIO 2025 S6-L2

Exploit DVWA – XSS e SQL injection

Argomento: Sfruttamento delle Vulnerabilità XSS e SQL Injection sulla DVWA.

Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application (DVWA).

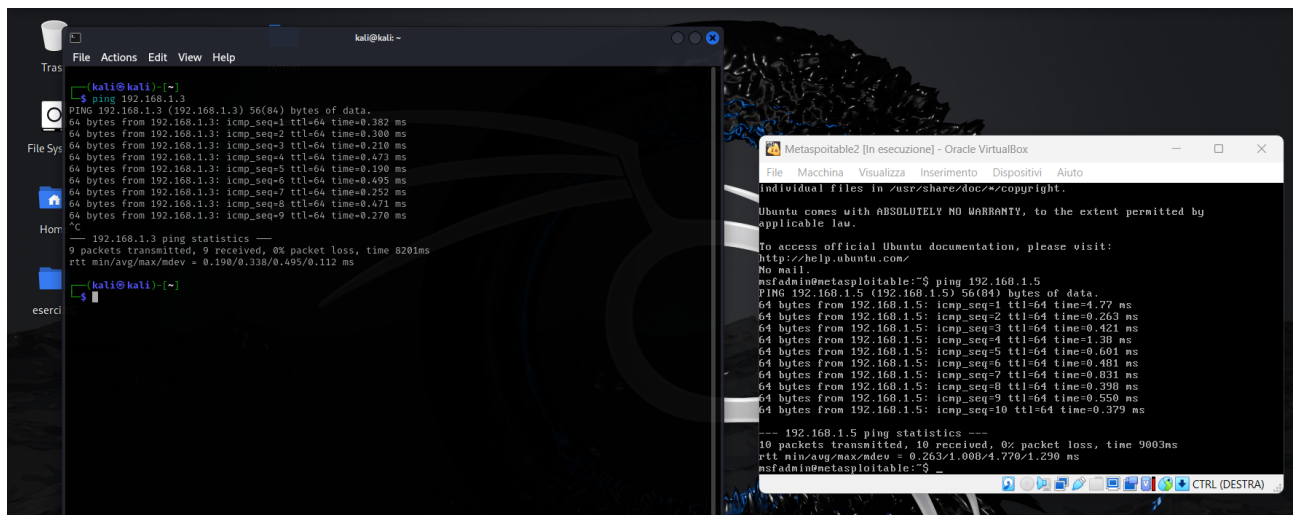
1. Configurazione del Laboratorio:

Configurare l'ambiente virtuale in modo che la macchina DVWA sia raggiungibile dalla macchina Kali Linux.

Verificare la comunicazione tra le due macchine utilizzando il comando ping.

Procediamo quindi con la configurazione dell'ambiente virtuale facendo partire le macchine virtuali Kali e Metaspitable2.

Successivamente controlliamo che le macchine comunichino tra loro tramite i ping reciproci.



2. Impostazione della DVWA:

Accedere alla DVWA dalla macchina Kali Linux tramite il browser.

Navigare fino alla pagina di configurazione e settare il livello di sicurezza a LOW.



Username

Password

Login

DVWA Security

Script Security

Security Level is currently **low**.

You can set the security level to low, medium or high.

The security level changes the vulnerability level of DVWA.

PHPIDS

PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications.

You can enable PHPIDS across this site for the duration of your session.

PHPIDS is currently **disabled**. [\[enable PHPIDS\]](#)

[\[Simulate attack\]](#) - [\[View IDS log\]](#)

Security level set to low

3. Sfruttamento delle Vulnerabilità:

Scegliere una vulnerabilità XSS Reflected e una vulnerabilità SQL Injection (non blind).

Utilizzare le tecniche viste nella lezione teorica per sfruttare con successo entrambe le vulnerabilità.

Iniziamo quindi a testare le vulnerabilità XSS Reflected dall' apposita sezione della DVWA.

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Submit

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

Inseriamo un nome e vediamo che ci verrà restituito l' output "Hello Nicolò":

192.168.1.3/dvwa/vulnerabilities/xss_r/?name=Nicolò#

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Submit

Hello Nicolò

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

ESEMPIO DI ATTACCO:

Supponiamo di avere un sito web con un modulo di ricerca che prende l'input dell'utente e lo visualizza sulla pagina dei risultati di ricerca senza sanitizzare l'input.

1:Input dell'utente: Inseriamo il seguente codice nel campo di ricerca→ `<script>alert('XSS Attack!');</script>`

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

2: Invio dell'Input: L'input viene inviato al server tramite una richiesta GET o POST.

192.168.1.3/dvwa/vulnerabilities/xss_r/?name=<script>alert('XSS+Attack!')%3B<%2Fscript>#

3: Generazione del Contenuto: Il server prende l'input dell'utente e lo inserisce direttamente nel contenuto HTML della pagina dei risultati di ricerca.

```
<div class="body_padded">
  <h1>Vulnerability: Reflected Cross Site Scripting (XSS)</h1>

  <div class="vulnerable_code_area">

    <form name="XSS" action="#" method="GET">
      <p>What's your name?</p>
      <input type="text" name="name">
      <input type="submit" value="Submit">
    </form>

    <pre>Hello <script>alert('XSS Attack!');</script></pre>

  </div>
```

4: Esecuzione del Codice Malevolo: Quando la pagina dei risultati di ricerca viene visualizzata, il browser esegue il codice JavaScript inserito dall'attaccante.



Avremo quindi come output questo popup. Possiamo quindi dedurre che qualsiasi input diamo, questo verrà inserito direttamente all'interno del codice HTML modificandolo.

Testiamo ora le vulnerabilità SQL Injection.

SQL è un linguaggio di programmazione specificamente progettato per la gestione e la manipolazione di dati all'interno di un sistema.

Procediamo alla verifica su DVWA:



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

Vulnerability: SQL Injection


User ID:

Submit

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Possiamo vedere che semplicemente dando in output il numero 1, ci verranno restituite in output le credenziali dell'utente numero 1 presenti all'interno del database.



Home

Instructions

Setup

Brute Force

Command Execution

CSRF

File Inclusion

SQL Injection

SQL Injection (Blind)

Upload

XSS reflected

XSS stored

Vulnerability: SQL Injection

User ID:

Submit

ID: 1
First name: admin
Surname: admin

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Modificando l' URL inserendo nella query una condizione “sempre vera” ovvero 1'or'1 avremo in output tutti i dati contenuti nel database.

192.168.1.3/dvwa/vulnerabilities/sqli/?id=1'or'1&Submit=Submit#

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: SQL Injection

User ID:

ID: 1'or'1
First name: admin
Surname: admin

ID: 1'or'1
First name: Gordon
Surname: Brown

ID: 1'or'1
First name: Hack
Surname: Me

ID: 1'or'1
First name: Pablo
Surname: Picasso

ID: 1'or'1
First name: Bob
Surname: Smith

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
DVWA Security
PHP Info
About
Logout