

How to Use PowerShell Empire Getting Started with Post-Exploitation of Windows Hosts

[PowerShell Empire](#) is a post-exploitation framework for computers and servers running Microsoft Windows, Windows Server operating systems, or both. In these tutorials, we will be exploring everything from how to install Powershell Empire to how to snoop around a victim's computer without the antivirus software knowing about it. If we are lucky, we might even be able to obtain domain administrator credentials and own the whole network.

A Tool for Targeting Windows

Exploit frameworks are popular, and most hackers have heard of [Metasploit](#), a framework that automates the deployment of powerful exploits. You may be asking yourself, how does PowerShell Empire differ from Metasploit? Isn't Metasploit already serving the same purpose? Well, yes and no. PowerShell Empire deals strictly with Windows machines, and it is extremely useful in a penetration test because most targets these days are running some version of Windows.

Don't Miss: [The Ultimate Command Sheet for Metasploit's Meterpreter](#)

A simple example of this point would be the widespread usage of excel on Microsoft Windows. Since Microsoft Excel has more advanced features than the Mac version (as well as Office 365), we can assume that the finance department of most target companies will be using Microsoft Windows. Finance departments also usually have access to bank account numbers and other juicy data!

PowerShell Empire also gives the attacker the ability to run commands in memory. This means that the malicious actions being taken by PowerShell Empire are not run on the hard drive, they are instead run in the computer's memory. This reduces the likelihood of being caught by antivirus software as well as the likelihood of leaving digital fingerprints for forensics investigators.

When to Use PowerShell Empire

Some of the activities and goals that can be accomplished include privilege escalation (elevating privileges from a standard user account to an administrator), network and host reconnaissance (finding out what hosts and services are present), lateral movement between hosts, and the gathering of credentials. All of these are key components of a modern day penetration test.

PowerShell Empire accomplishes this via three main components: listeners, stagers, and agents.

- A listener is a process which listens for a connection from the machine we are attacking. This helps Empire send the loot back to the attacker's computer.
- A stager is a snippet of code that allows our malicious code to be run via the agent on the compromised host.
- An agent is a program that maintains a connection between your computer and the compromised host.

Lastly, modules are where the fun is. These are what execute our malicious commands, which can harvest credentials and escalate our privileges as mentioned above.

Now that we have discussed what PowerShell Empire does and why it is useful, let's take a look at how to get it up and running.

Step 1 Installing PowerShell Empire

To run Powershell, you will need a Kali Linux machine. If you need a good starter Kali computer for hacking, you can check out our guide on setting one up on the [low-cost Raspberry Pi](#) below.

Learn More: [Build a Kali Linux Hacking Computer on the Raspberry Pi](#)

To install Empire on your Kali Linux machine, we need to clone it from GitHub. Open a terminal and type the following command as shown below.

git clone <https://github.com/EmpireProject/Empire.git>

```
root@kali:~# git clone https://github.com/EmpireProject/Empire
Cloning into 'Empire'...
remote: Counting objects: 5562, done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 5562 (delta 0), reused 2 (delta 0), pack-reused 5557
Receiving objects: 100% (5562/5562), 16.61 MiB | 1.08 MiB/s, done.
Resolving deltas: 100% (3429/3429), done.
root@kali:~#
```

This will create a new directory with the name "Empire." Move into that directory by typing **cd Empire**, then use the **ls** command to view the contents of the directory.

You can read about Empire in the *README.md* file. You will see a "setup" folder inside the Empire directory. Navigate to that folder by typing **cd setup**, then use the **ls** command to view the contents of the "setup" folder. You can see an install shell script as shown below.

```
root@kali:~# cd Empire
root@kali:~/Empire# ls
changelog  data  empire  lib  LICENSE  README.md  setup
root@kali:~/Empire# leafpad README.md
root@kali:~/Empire# cd setup
root@kali:~/Empire/setup# ls
cert.sh  install.sh  reset.sh  setup_database.py
root@kali:~/Empire/setup# ./install.sh
```

Type **./install.sh** to install Empire by running the script. The installation will start as shown below.

```
root@kali:~/Empire/setup# ./install.sh
Reading package lists... Done
Building dependency tree
Reading state information... Done
default-jdk is already the newest version (2:1.8-58).
default-jdk set to manually installed.
g++ is already the newest version (4:6.3.0-4).
g++ set to manually installed.
make is already the newest version (4.1-9.1).
make set to manually installed.
python-dev is already the newest version (2.7.13-2).
python-dev set to manually installed.
python-m2crypto is already the newest version (0.24.0-1.1).
python-m2crypto set to manually installed.
python-pip is already the newest version (9.0.1-2).
Suggested packages:
  icu-doc pkg-config swig-doc swig-examples swig3.0-examples swig3.0-doc
The following NEW packages will be installed:
  icu-devtools libicu-dev libssl-dev libssl-doc libxml2-dev swig swig3.0
The following packages will be upgraded:
  libicu57 libssl1.1
```

During the installation process, you will be asked to set up a server negotiation password. I set it as "toor," but you can choose your own password. If everything went well, the installation will finish as shown below.

```
crc32.o
gzip -c man/mkbom.1 > build/man/mkbom.1.gz
gzip -c man/dumpbom.1 > build/man/dumpbom.1.gz
gzip -c man/lsbom.1 > build/man/lsbom.1.gz
gzip -c man/ls4mkbom.1 > build/man/ls4mkbom.1.gz
install -d /usr/bin
install -d /usr/share/man/man1
install -m 0755 build/bin/mkbom build/bin/dumpbom build/bin/lsbom build/bin/ls4mkbom /usr/bin
install -m 0644 build/man/mkbom.1.gz build/man/dumpbom.1.gz build/man/lsbom.1.gz build/man/ls4mkbom.1.gz /usr/share/man/man1

[>] Enter server negotiation password, enter for random generation: toor

[*] Database setup completed!

[*] Certificate written to ../data/empire.pem

[*] Setup complete!
root@kali:~/Empire/setup#
```

We are done with the installation. Now, it's time to start Empire.

Step 2 Running Powershell Empire

Move back to the Empire directory by typing **cd ..** and run the **./empire** executable as shown. It will start as seen below.

```

root@kali:~/Empire# ls
changelog data empire lib LICENSE README.md setup
root@kali:~/Empire# ./empire
[*] Loading stagers from: /root/Empire//lib/stagers/
[*] Loading modules from: /root/Empire//lib/modules/
[*] Loading listeners from: /root/Empire//lib/listeners/

```

If Empire displays any error while starting, navigate to the "setup" folder with **cd setup** and run the **./reset.sh** script. Then restart Empire again like we did before. It will display a welcome message as shown below.

```

      .odNNmdmmNNNo` `` `` .: +yNNNNNNNNNNNN
      -sNNNmhdh/dNNhhdNNNNNNNNNNNNNNNNNNNN
      -hNNNmNo: :mNNNNNNNNNNNNNNNNNNNNNNNN
      -hNNmdNo- -/dNNNNNNNNNNNNNNNNNNNNNN
      :dNmmdmd- : +NNNNNNNNNNNNNNNNNNNNNm
      /hNNmmdmd+mNNNNNNNNNNNNNNNNds++o
      /dNNNNNNmmmmmmNNNNNNNNNNNNmdoosydd
      sNNNNdydNNNNmmmmmmNNNNNNmyoymNNNNNN
      :NNmmmdso++dNNNNmmNNNNNdhyNNNNNNNNNN
      -NmmdmNNdsyohNNNNmmNNNNNNNNNNNNNNNN
      `sdhmmNNNNdyhdNNNNNNNNNNNNNNNNNNNN
      /yhmmNNmmNNNNNNNNNNNNNNNNNNNNmmhh
      +yhmmNNNNNNNNNNNNNNNNNNNNNNmmh+:
      . /dmmmmNNNNNNNNNNNNNNNNmmmd .
      ommmmNNNNNNNNmmNNNNmmmd:
      :dmmmmNNNNmmh. . /oyhhhhy:
      `sdmmmmNNNmh/++- .+oh.
      ` /dmmmmmmmdo- : /ossd:
      ` /ohhdmmmmmdddddmh/
      ` - /osyhdddddhyo:
      .----.

Welcome to the Empire

```


Upon completion, Empire will show the following screen.

```
=====
[Empire] Post-Exploitation Framework
=====
[Version] 2.0 | [Web] https://theempire.io
=====

  EMPiRE

  267 modules currently loaded
  0 listeners currently active
  0 agents currently active

(Empire) > █
```

As of this writing, Empire has 267 modules. Don't worry if these sound like complicated ninjitsu techniques; with diligence and practice you will learn what modules, listeners, and agents are. By the end of this series, you will get a clear idea what these are and how to use them.

First, let's start by typing the **help** command. The **help** command will display the help menu as seen below.

```

(Empire) > help

Commands
=====
agents          Jump to the Agents menu.
creds           Add/display credentials to/from the database.
exit            Exit Empire
help            Displays the help menu.
interact        Interact with a particular agent.
list            Lists active agents or listeners.
listeners       Interact with active listeners.
load            Loads Empire modules from a non-standard folder.
reload          Reload one (or all) Empire modules.
reset           Reset a global option (e.g. IP whitelists).
searchmodule    Search Empire module names/descriptions.
set             Set a global option (e.g. IP whitelists).
show            Show a global option (e.g. IP whitelists).
usemodule       Use an Empire module.
usestager       Use an Empire stager.

(Empire) > █

```

Step 3 Using Listeners

[Listeners](#) in Empire are the channels which receive connections from our target machine. Before we do anything in Empire, we need to start the listeners. We can move to the listener management menu by typing command **listeners** as shown below.

```

(Empire) > listeners
[!] No listeners currently active
(Empire: listeners) > help

Listener Commands
=====
agents          Jump to the Agents menu.
back            Go back to the main menu.
exit            Exit Empire.
help            Displays the help menu.
info            Display information for the given active listener.
kill            Kill one or all active listeners.
launcher        Generate an initial launcher for a listener.
list            List all active listeners (or agents).
main            Go back to the main menu.
uselistener     Use an Empire listener module.
usestager       Use an Empire stager.

(Empire: listeners) > █

```


Once we move to the listeners management menu, as shown above, we can see its sub-menu by typing the **help** command. Let's take a look at what each command will do.

- **agents** - Will allow you to jump to agents menu.
- **back & main** – Will take you back to the main menu.
- **exit** – Will exit from Empire.
- **help** – Will display help menu as shown in the above image.
- **info** – Will display information about the active listener.
- **kill** – Will kill a particular listener.
- **launcher** – Used to generate an initial launcher for a listener.
- **list** – Will list all the active listeners.
- **usestager** – Used to use a stager (we will see below what exactly is a stager).
- **uselistener** – Used to start a listener module.

Let us now look at how to start a listener module in Empire. Type the **uselistener** command, and use tab completion to see the listeners available in Empire.

```
(Empire: listeners) > uselistener
dbx          http_com      http_hop
http         http_foreign meterpreter
(Empire: listeners) > uselistener meterpreter
(Empire: listeners/meterpreter) > help

Listener Commands
=====
agents back execute exit help info launcher listeners main set unset
(Empire: listeners/meterpreter) >
```

The types of listeners available are shown above. We will learn about different types of listeners in the upcoming sections. For now, let's see how to start a listener.

Let's use the "meterpreter" listener as an example. Type **uselistener meterpreter** as shown above. Once the particular listener is loaded, you can type **help** command to display the available options.

The **agents**, **back**, **exit**, **help**, **launcher**, **listeners**, and **main** commands have been explained above. Let us learn about the other commands.

The **info** command shows the information about the particular type of listener we want to start, as seen below.

```

(Empire: listeners/meterpreter) > info
    Name: Meterpreter
Category: client_server

Authors:
    @harmj0y

Description:
    Starts a 'foreign' http[s] Meterpreter listener.

Meterpreter Options:

  Name      Required  Value                                Description
  ----      -
  Host      True       http://192.168.91.138:80            Hostname/IP for
staging.
  Name      True       meterpreter                        Name for the li
stener.
  Port      True       80                                Port for the li
stener.

(Empire: listeners/meterpreter) >

```

Every listener requires certain options to be set. For example, the "meterpreter" listener needs the *Host* and *Port* values to be configured. The **set** command is used to assign these values. Similarly, the **unset** command is used to clear these values.

Don't Miss: [How to Use Listeners](#)

One important thing to remember is that Empire is case sensitive. For example, in the screenshot below, I am setting the "Name" value of our listener. "Name" and "name" are different in Empire, and it will give you an error if they are used incorrectly, as they cannot be used interchangeably.


```

Meterpreter Options:

  Name          Required  Value          Description
  ----          -
  Host staging. True      http://192.168.91.138:80 Hostname/IP for
  Name          True      meterpreter    Name for the li
  stener.
  Port          True      80             Port for the li
  stener.

(Empire: listeners/meterpreter) > set
[!] Error in setting listener option: list index out of range
(Empire: listeners/meterpreter) > set name meterp
[!] Invalid option specified.
(Empire: listeners/meterpreter) > set Name meterp
(Empire: listeners/meterpreter) >

```



When all options are set, we can start a listener using the **execute** command.

```

(Empire) > listeners
[!] No listeners currently active
(Empire: listeners) > uselistener meterpreter
(Empire: listeners/meterpreter) > help

Listener Commands
=====
agents back execute exit help info launcher listeners main set unset

(Empire: listeners/meterpreter) > execute
[*] Starting listener 'meterp'
[+] Listener successfully started!
(Empire: listeners/meterpreter) >

```

Once we go back to the main menu, we can see that our listener is currently active.

```
=====
[Empire] Post-Exploitation Framework
=====
[Version] 2.0 | [Web] https://theempire.io
=====

  EMPiRE

  267 modules currently loaded
  1 listeners currently active
  0 agents currently active

(Empire) > █
```

Step 4 Using Stagers

Stagers in Empire are used to set the stage for the post-exploitation activities. They are similar to payloads, which are used to create a connection back to Empire. The stagers can be accessed using the **usestager** command as shown below.

Don't Miss: [How to Use Payloads with Metasploit](#)

Type the **usestager** and then use the tab completion to see all the available stagers.

```
(Empire) > usestager
multi/bash          osx/jar             windows/dll
multi/launcher      osx/launcher        windows/ducky
multi/pyinstaller   osx/macho           windows/hta
multi/war           osx/macro           windows/launcher_bat
osx/applescript     osx/pkg             windows/launcher_sct
osx/application     osx/safari_launcher windows/launcher_vbs
osx/ducky           osx/teensy          windows/macro
osx/dylib           windows/bunny       windows/teensy
```

We will learn about different stagers in an upcoming section. First, let's take a look at how to set up a stager.

Let's start the "launcher_bat" stager as an example.

Type the **usestager windows/launcher_bat** command to load the stager.

```
(Empire) > usestager windows/launcher_bat
(Empire: stager/windows/launcher_bat) > help

Stager Menu
=====
agents          Jump to the Agents menu.
back            Go back a menu.
execute         Generate/execute the given Empire stager.
exit           Exit Empire.
generate        Generate/execute the given Empire stager.
help           Displays the help menu.
info           Display stager options.
interact        Interact with a particular agent.
list           Lists all active agents (or listeners).
listeners       Jump to the listeners menu.
main           Go back to the main menu.
options        Display stager options.
set            Set a stager option.
unset          Unset a stager option.

(Empire: stager/windows/launcher_bat) > █
```

Type the **help** command to have a look at the stager menu.

- **agents** - Will allow you to jump directly to agents menu.
- **back & main** – Will take you back to the main menu.
- **exit** – Will exit from Empire.
- **help**- Will display help menu as shown in the above image.
- **info**- Will display information about the active listener.
- **kill**- Is used to kill a particular listener.
- **execute or generate** – Will execute or generate the stager.
- **interact** – Is used to interact with a particular agent (normally used when there are multiple listeners).
- **list** - Will list all the active listeners or agents.
- **options**- Used to see all the options we need to set for the particular agent.
- **set and unset** – Used to set and unset values to particular options, respectively.
- **listeners** - Used to jump to listeners menu.

We can get more information about this particular stager by using the **info** command. As you can see in the info, it creates a self-deleting batch file.

```
(Empire: stager/windows/launcher_bat) > info
```

Name: BAT Launcher

Description:
Generates a self-deleting .bat launcher for Empire.

Options:

Name	Required	Value	Description
Listener	True		Listener to generate stager for
OutFile	False	/tmp/launcher.bat	File to output .bat launcher to, otherwise displayed on the screen.
Proxy	False	default	Proxy to use for request (default, none, or other).
Language	True	powershell	Language of the stager to generate.
ProxyCreds	False	default	Proxy credentials.

We need to set a listener in order for the stager to be able to communicate with Empire. In the last step, we have already created a listener. Let us set this listener for our "launcher_bat" stager.

```

    Language      True      powershell      or other).
ate.             ProxyCreds  False      default      Proxy credentials
                use for      ([domain\]username:password) to
r).             UserAgent  False      default      User-agent string to use for th
e staging        request (default, none, or othe
r).             Delete      False      True      Switch. Delete .bat after runni
ng.             StagerRetries  False      0      Times for the stager to retry
                connecting.

(Empire: stager/windows/launcher_bat) > set Listener meterp
(Empire: stager/windows/launcher_bat) > execute

[*] Stager output written out to: /tmp/launcher.bat
(Empire: stager/windows/launcher_bat) >

```

We can do this using **set Listener meterp** command. Type the **execute** command to generate the stager. The stager is created in the "tmp" folder as indicated by the output shown above in blue.

Step 5 Using Agents

When we send the stager to our target system and the machine engages with it, we get a reverse connection back. This is known as an agent.

The Agents menu can be accessed using **agents** command as shown below. But, as is stated in the red output, we do not currently have any agents registered. That is just around the corner.

```
(Empire) > agents
[!] No agents currently registered
(Empire: agents) > help
```

Commands

=====

back	Go back to the main menu.
clear	Clear one or more agent's taskings.
creds	Display/return credentials from the database.
exit	Exit Empire.
help	Displays the help menu.
interact	Interact with a particular agent.
kill	Task one or more agents to exit.
killdate	Set the killdate for one or more agents (killdate [agent/all]
01/01/2016).	
list	Lists all active agents (or listeners).
listeners	Jump to the listeners menu.
lostlimit	Task one or more agents to 'lostlimit [agent/all] [number of m
issued callbacks]	'
main	Go back to the main menu.
remove	Remove one or more agents from the database.
rename	Rename a particular agent.
searchmodule	Search Empire module names/descriptions.
sleep	Task one or more agents to 'sleep [agent/all] interval [jitter

clear	Clear one or more agent's taskings.
creds	Display/return credentials from the database.
exit	Exit Empire.
help	Displays the help menu.
interact	Interact with a particular agent.
kill	Task one or more agents to exit.
killdate	Set the killdate for one or more agents (killdate [agent/all]
01/01/2016).	
list	Lists all active agents (or listeners).
listeners	Jump to the listeners menu.
lostlimit	Task one or more agents to 'lostlimit [agent/all] [number of m
issued callbacks]	'
main	Go back to the main menu.
remove	Remove one or more agents from the database.
rename	Rename a particular agent.
searchmodule	Search Empire module names/descriptions.
sleep	Task one or more agents to 'sleep [agent/all] interval [jitter
]	
usemodule	Use an Empire PowerShell module.
usestager	Use an Empire stager.
workinghours	Set the workinghours for one or more agents (workinghours [age
nt/all] 9:00-17:00).	

```
(Empire: agents) > █
```

The output of the **help** command is shown above. It will display all the commands we can use when an agent establishes a connection with Empire. For example, typing the **list** command will show all the active agents we have, as shown below.

```
(Empire: agents) > list

[*] Active agents:

  Name          Lang  Internal IP      Machine Name  Username      Proc
  -----
  7A9WSDPN      ps    [REDACTED]      [REDACTED]    [REDACTED]    powe
  rshell/4032   5/0.0 2017-07-12 09:00:44
```

Step 6 Using Modules

Modules in Empire are used to perform specific functions. We can access modules using the **usemodule** command. Type **usemodule** <Space> and then use tab completion to see all the modules.

```
(Empire: agents) > usemodule
Display all 267 possibilities? (y or n)
exfiltration/Invoke_ExfilDataToGitHub
external/generate_agent
powershell/code_execution/invoke_dllinjection
powershell/code_execution/invoke_metasploitpayload
powershell/code_execution/invoke_reflectivepeinjection
powershell/code_execution/invoke_shellcode
powershell/code_execution/invoke_shellcodemsil
powershell/collection/ChromeDump
powershell/collection/FoxDump
powershell/collection/USBKeylogger
powershell/collection/WebcamRecorder
powershell/collection/browser_data
powershell/collection/clipboard_monitor
powershell/collection/file_finder
powershell/collection/find_interesting_file
powershell/collection/get_indexed_item
powershell/collection/get_sql_column_sample_data
powershell/collection/get_sql_query
```

We will learn more about different modules in a later tutorial. First, let's take a look at how to use modules in Empire. Let's use the "external/generate_agent" as an example. Type

usemodule external/generate_agent to load the module. Once the required module is loaded, type **help** to see all the commands we can use with the module.

```
(Empire: agents) > usemodule external/generate_agent
(Empire: external/generate_agent) > help

Module Commands
=====
agents      Jump to the Agents menu.
back        Go back a menu.
creds       Display/return credentials from the database.
execute     Execute the given Empire module.
exit        Exit Empire.
help        Displays the help menu.
info        Display module options.
interact    Interact with a particular agent.
list        Lists all active agents (or listeners).
listeners   Jump to the listeners menu.
main        Go back to the main menu.
options     Display module options.
reload      Reload the current module.
run         Execute the given Empire module.
set         Set a module option.
unset       Unset a module option.
usemodule   Use an Empire PowerShell module.

(Empire: external/generate_agent) > █
```

- **agents** - Will allow you to jump directly to agents menu.
- **back & main** – Will take you back to the main menu
- **exit** – Will exit from Empire.
- **help** – Will display help menu as shown in the above image.
- **info** – Will display information about the active listener.
- **kill** – Is used to kill a particular listener.
- **execute or run** – Will execute the selected module.
- **interact** – Is used to interact with a particular agent (normally used when there are multiple listeners).
- **list** – Will list all the active listeners or agents.
- **options** – Is used to see all the options we need to set for the particular agent.
- **set and unset** – Used to set and unset values for particular options.
- **listeners** – Used to jump to listeners menu.
- **reload** – Will reload the current module.

Type the **options** command to see the options required for the module.


```

(Empire: external/generate_agent) > options

      Name: Generate Agent
      Module: external/generate_agent

Authors:
  @harmj0y

Description:
  Generates an agent code instance for a specified listener,
  pre-staged, and register the agent in the database. This
  allows the agent to begin beconing behavior immediately.

Options:

  Name      Required  Value      Description
  ----      -
  Listener  True           Listener to generate the agent
for.
  OutFile   True           /tmp/agent  Output file to write the agent
code to.
  Language  True           Language to generate for the ag
ent.

```

Set the required options using the **set** command, and when complete, use the **execute** command to generate the module.

```

(Empire: external/generate_agent) > set Listener http
(Empire: external/generate_agent) > set Language powershell
(Empire: external/generate_agent) > execute
[+] Pre-generated agent 'QKHQXGMU' now registered.
[*] powershell agent code for listener http with sessionID 'QKHQXGMU' written ou
t to /tmp/agent
[*] Run sysinfo command after agent starts checking in!
(Empire: external/generate_agent) > █

```

We will get into more detail about Empire in the upcoming sections. These are the first steps in getting Empire up and running, so stay tuned for more! You can leave any questions in the comments below.