



BLUEHAT

SECURITY ABOVE ALL ELSE

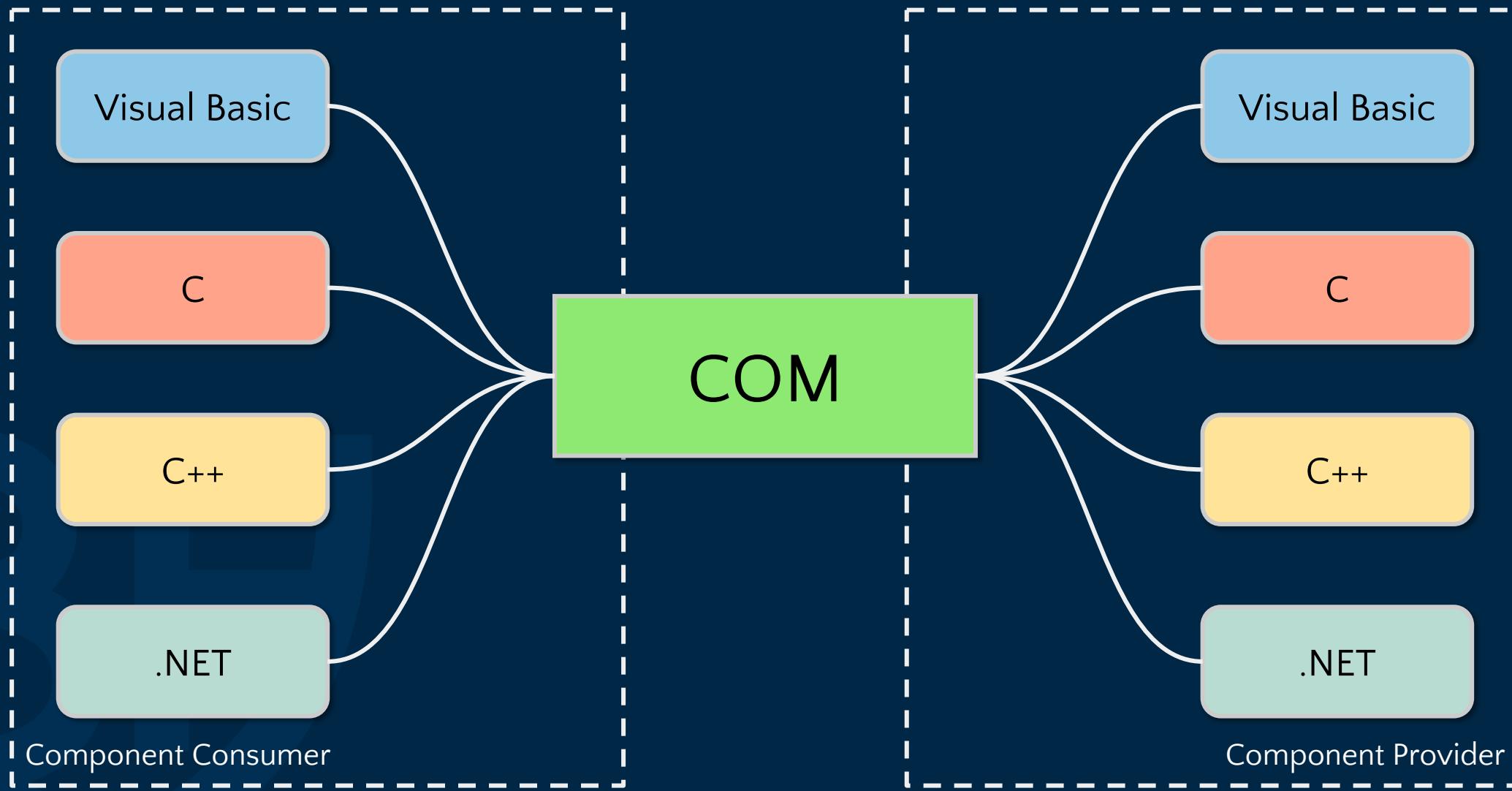
DCOM Research for Everyone!

James Forshaw - Google Project Zero

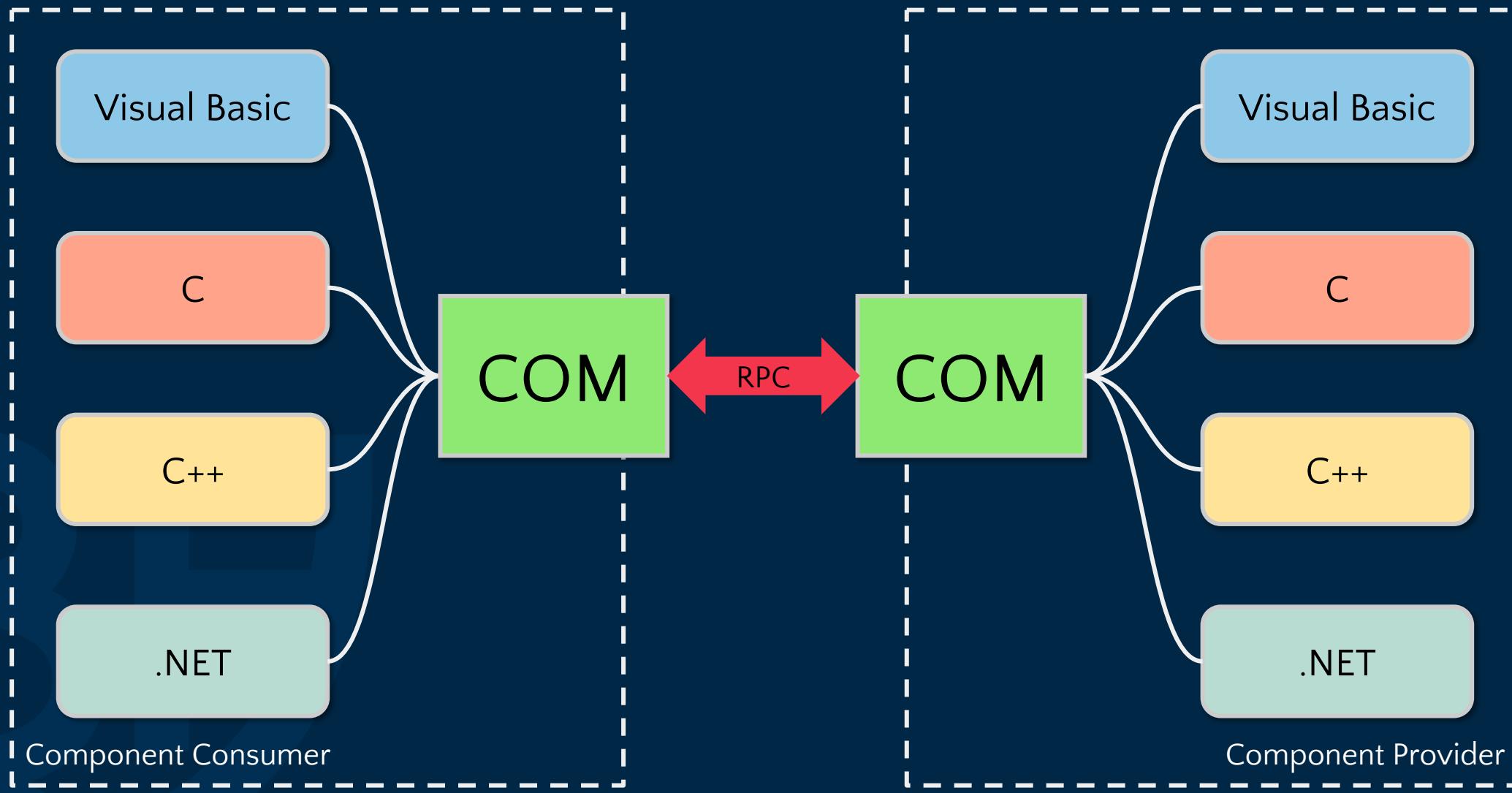
Agenda

1. Very brief introduction to COM
2. Why do we care anyway?
3. Using the Right Tool
4. DCOM Inter-process communication
5. Implementing DCOM clients
6. Conclusions

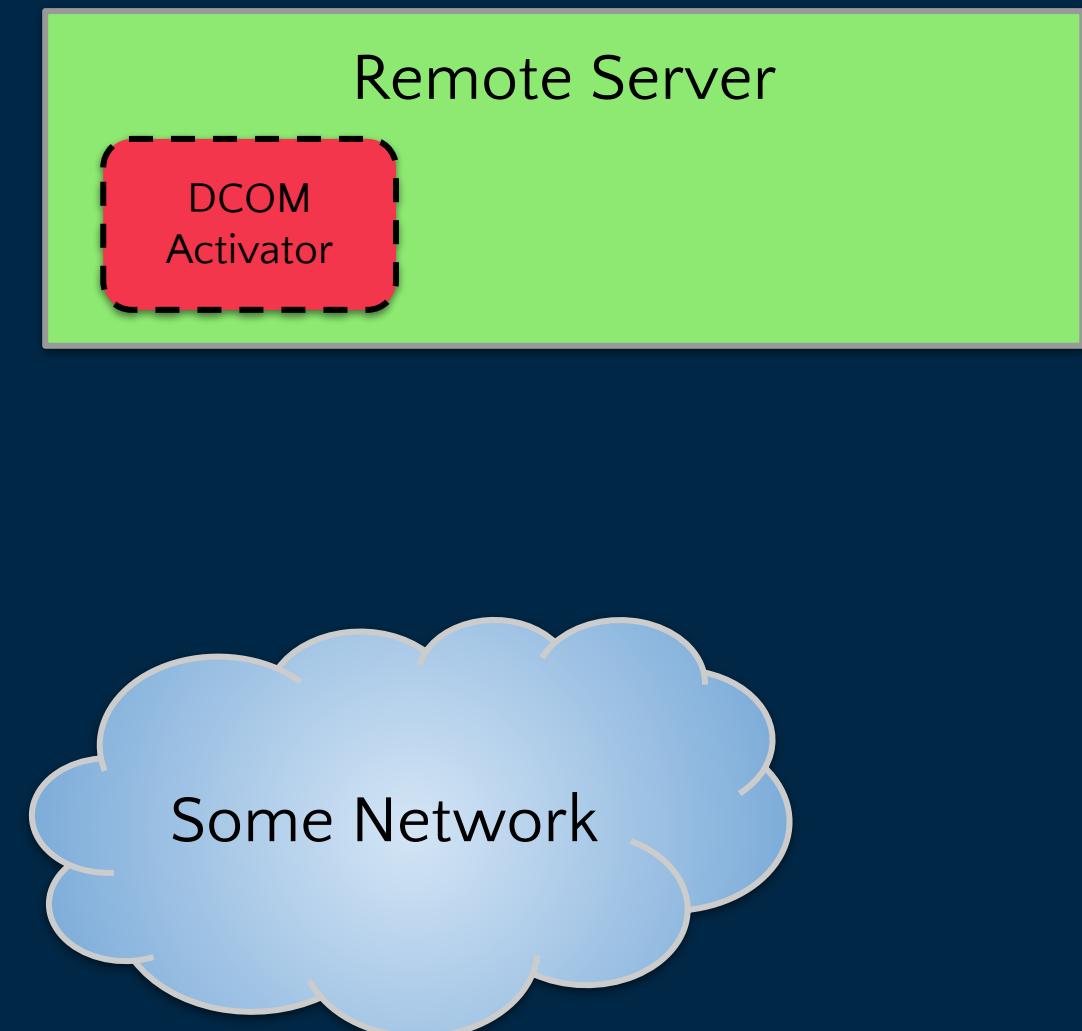
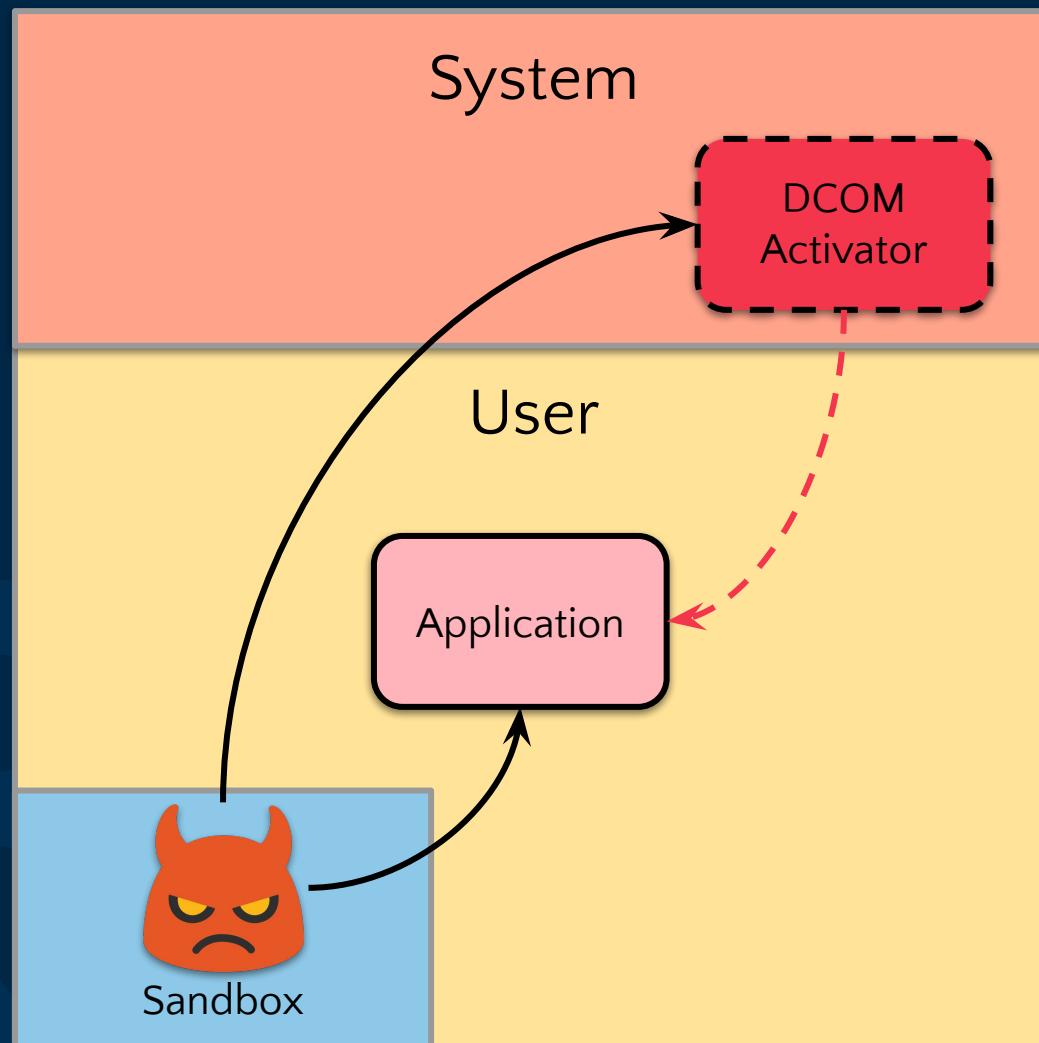
COM as Glue Logic



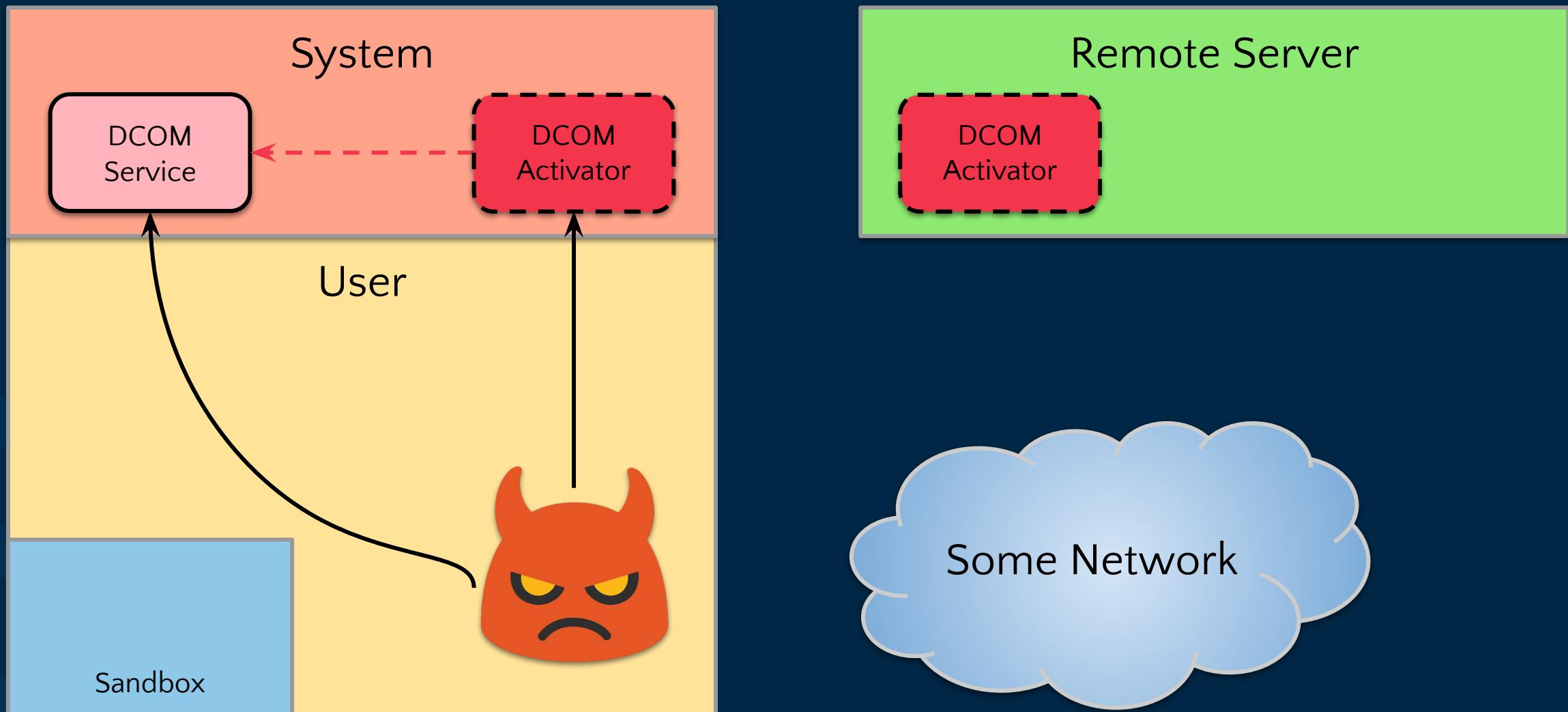
Distributed COM



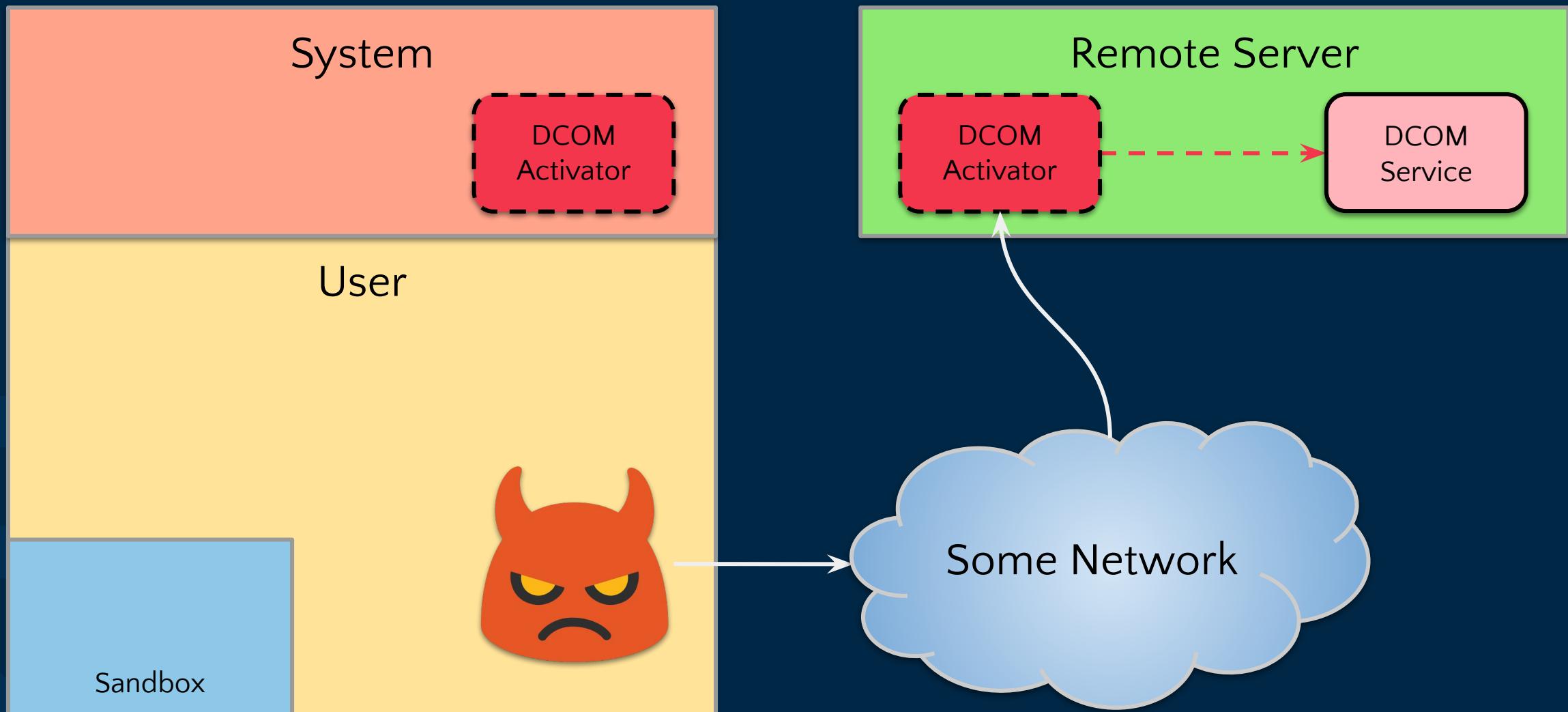
DCOM Attack Surface



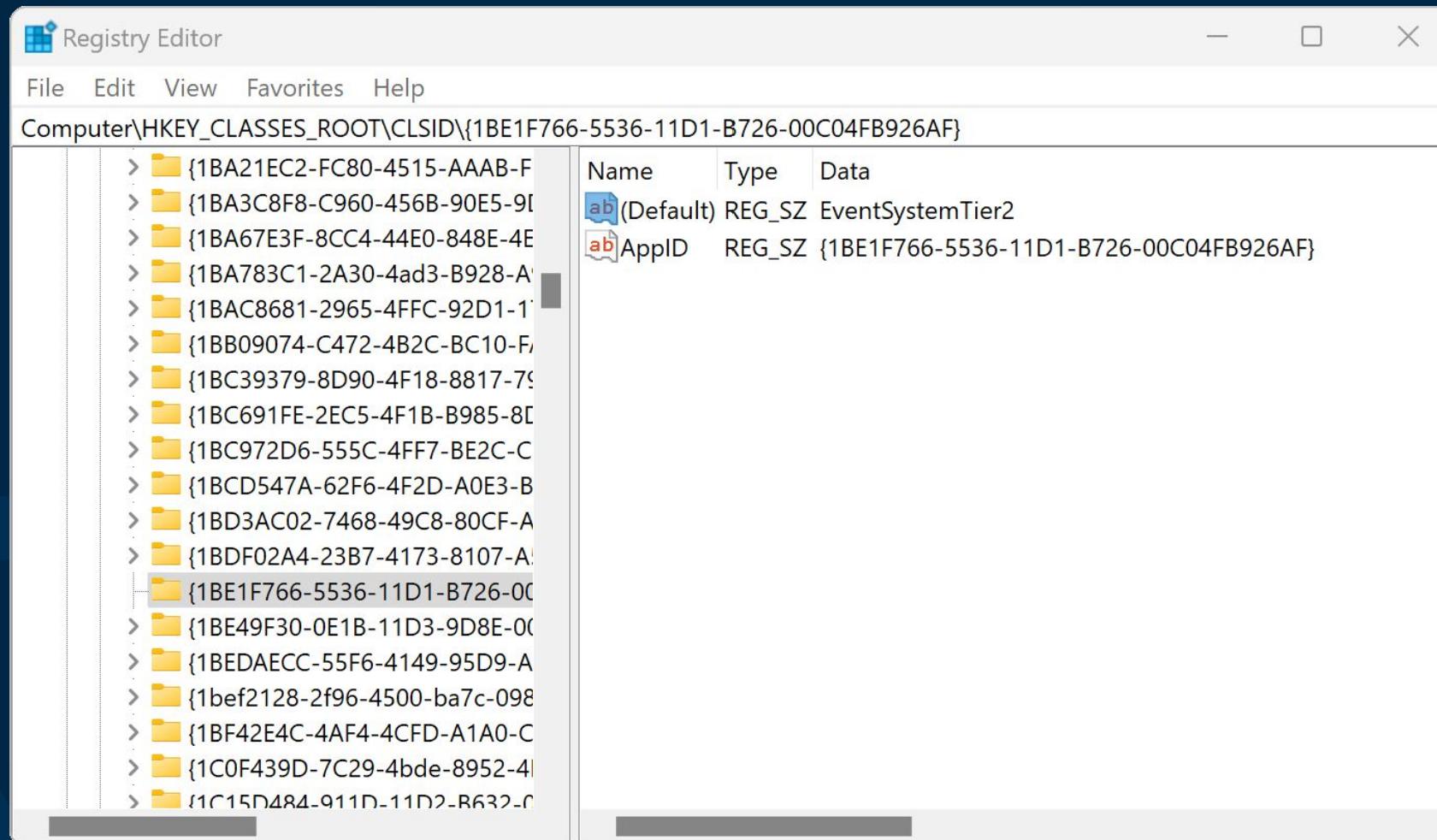
DCOM Attack Surface



DCOM Attack Surface



Old School COM Research



Old School COM Research

The image shows two windows side-by-side: the Windows Registry Editor and the OLE/COM Object Viewer.

Registry Editor (Left Window):

- Path: Computer\HKEY_CLASSES_ROOT\CLSID\{1BE1F766-5536-11D1-B726-00C04FB926AF}
- Content: A list of various COM class entries, including EvalRat Class, Event Class, Event Publisher, Event Subscription, Event System, EventRegistrar Class, EventSystem.EventObjectChange, EventSystem.EventObjectChange2, and EventSystemTier2.

OLE/COM Object Viewer (Right Window):

- Path: Computer\HKEY_CLASSES_ROOT\CLSID\{1BE1F766-5536-11D1-B726-00C04FB926AF}\EventSystemTier2
- Content:
 - No icon Available
 - EventSystemTier2 {1BE1F766-5536-11D1-B726-00C04FB926AF}
 - Activation tab (selected):
 - Use default launch permissions (radio button)
 - Use these launch permissions:
 - Launch Permissions tab:

User/Group	Can Launch
\Everyone	Yes
BUILTIN\Administrators	Yes
NT AUTHORITY\INTERACTIVE	Yes
NT AUTHORITY\SYSTEM	Yes

OleView.NET

OleView .NET v1.14 - 64bit (X64)

File Registry Object Security Processes Storage View Help

CLSIDs by Name

Filter: Mode: Contains Apply

- + * EvalRat Class
- + * Event Class
- + * Event Publisher
- + * Event Subscription
- + * Event System
- + * EventRegistrar Class
- + * EventSystem.EventObjectChange
- + * EventSystem.EventObjectChange2
- + * EventSystemTier2**
- IClientSecurity
- IEVENTSystemTier2
- IMarshal
- IMarshal2
- IMultiQI
- IUnknown
- Factory Interfaces
- + * EVR Graph Optimizer
- + * Exchange Active Sync Policies Broker
- + * ExecModelProxy
- + * Execute OpenSearch view site command
- + * Execute Unknown
- + * ExecuteAssociation

Showing 7961 of 7961 entries

EventSystemTi...

CLSID	Supported Interfaces	AppID	Service	Access	Security	Launch	Security

Owner: BUILTIN\Administrators
Group: BUILTIN\Administrators
Integrity: N/A

DACL

Flags: None

ACL Entries

Type	Account	Access	Flags
Allowed	BUILTIN\Administrators	GenericAll	None
Allowed	Everyone	Execute ActivateLocal	None
Allowed	NT AUTHORITY\INTERACTIVE	Execute ExecuteLocal ActivateLocal	None
Allowed	NT AUTHORITY\SYSTEM	Execute ExecuteLocal ActivateLocal	None

Specific Access

Na... Ac...

More Comprehensive Analysis

The screenshot shows the OleView .NET interface with two main panes displaying lists of runtime interfaces.

Left Pane (Runtime Services):

- Filter: (empty)
- Mode: Contains
- Apply
- bthserv
 - + Microsoft.Bluetooth.Core.Interface.GapAdvertisementPublisher
 - + Microsoft.Bluetooth.Core.Interface.GapAdvertisementSection
 - + Microsoft.Bluetooth.Core.Interface.GapAdvertisementSectionsBuilder
 - + Microsoft.Bluetooth.Core.Interface.GapAdvertisementWatcher
 - + Microsoft.Bluetooth.Core.Interface.GapRadio
 - ** Error querying COM interfaces - No Instance Interfaces
 - + Factory Interfaces
 - + IActivationFactory
 - + IClientSecurity
 - + IGapRadioStatics
 - + IIInspectable
 - + IMarshal
 - + IMarshal2
 - + IMultiQI
 - + IUnknown
 - + Microsoft.Bluetooth.Core.Interface.GapRssiFilterConfiguration
 - + Microsoft.Bluetooth.Core.Interface.GapRssiFilterConstraint
 - + Microsoft.Bluetooth.Core.Interface.GapRssiFilterConstraintBuilder
 - + Microsoft.Bluetooth.Core.Interface.GapRssiFilterDeviceOptions
 - + Microsoft.Bluetooth.Profiles.Csip.Interface.CsipSetCoordinator
 - + Microsoft.Bluetooth.Profiles.Gatt.Interface.GattClientDevice

Showing 53 of 53 entries

Right Pane (Runtime Interfaces):

- Filter: (empty)
- Mode: Contains
- Apply
- Windows
 - + AI
 - + ApplicationModel
 - + Data
 - + Devices
 - + Foundation
 - + Collections
 - + Diagnostics
 - + Metadata
 - + IAsyncAction
 - + IAsyncActionWithProgress`1
 - + IAsyncInfo
 - + IAsyncOperation`1
 - + IAsyncOperationWithProgress`2
 - + IClosable
 - + IDeferral
 - + IDeferralFactory
 - + IGetActivationFactory
 - + IGuidHelperStatics
 - + IMemoryBuffer
 - + IMemoryBufferFactory
 - + IMemoryBufferReference

Showing 1 of 1 entries

Filtering

View Filter X

Type	Field	Comparison	Value	Decision
Interface	Base	Contains		Include

Reset **Add** **Remove**

Type	Field	Comparison	Value
<input checked="" type="checkbox"/>	HasRunAs	Contains	True
<input checked="" type="checkbox"/>	SafeForScripting	Contains	True

OK **Cancel**

Test Container

OleView .NET v1.14 - 64bit (X64)

File Registry Object Security Processes Storage View Help

Windows Medi... Windows Me...

Invoke

Name: IWMPPlayer4

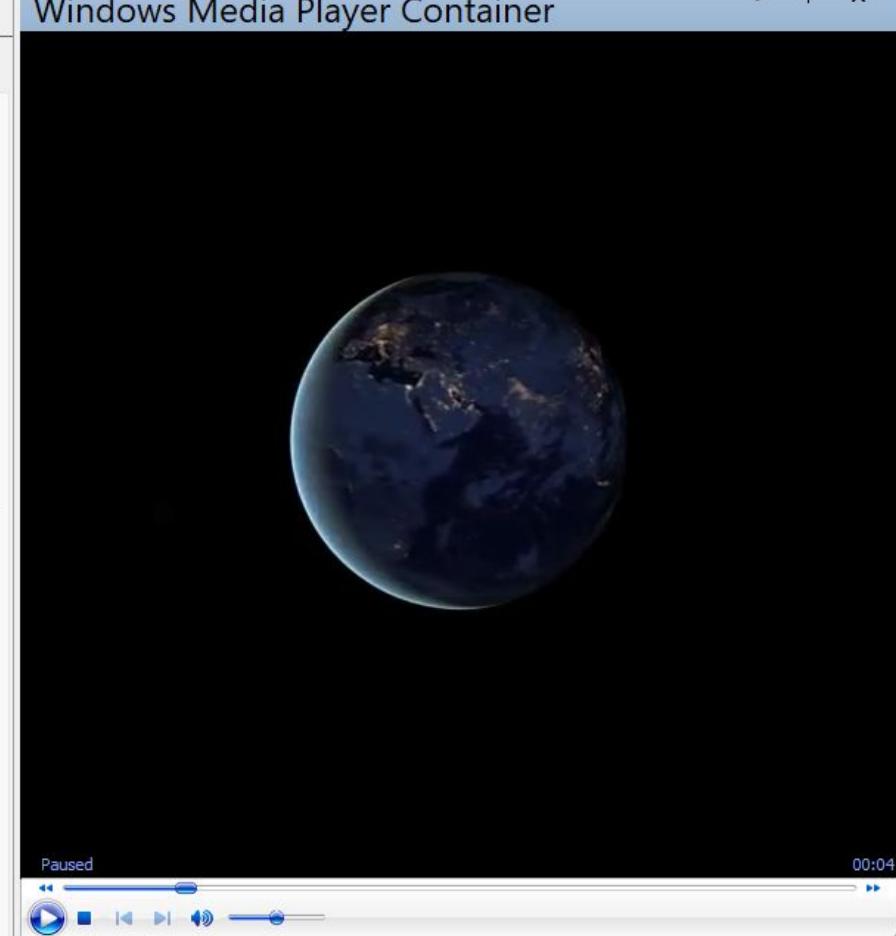
Methods:

Name	Return	Params
close	System.Void	
launchURL	System.Void	[in] String bst
newMedia	6bf52a50-394a-11d3-b153-00c04f79faa6.IWMPMedia	[in] String bst
newPlaylist	6bf52a50-394a-11d3-b153-00c04f79faa6.IWMPPPlaylist	[in] String bst
openPlayer	System.Void	[in] String bst

Properties:

Name	Type	Value
stretchToFit	System.Boolean	False
uiMode	System.String	full
URL	System.String	D:\file_example_MOV_480_700kB.mov
versionInfo	System.String	12.0.22621.4249
windowlessVideo	System.Boolean	False

Windows Media Player Container



The Windows Media Player Container window displays a video of Earth from space, showing clouds and continents against a dark background. At the bottom of the window, there is a control bar with buttons for play, pause, stop, and volume, along with a progress bar indicating the video is at 00:04.

DEMO



DCOM Security Research



Enumerate Classes

PowerShell Scripting

```
PS D:\> $cls = Get-ComClass -Name 'EventSystemTier2'
PS D:\> Get-ComInterface -Class $cls

Name                IID                               HasProxy  HasTypeLib
----              ----
IUnknown            00000000-0000-0000-c000-000000000046 False    False
IMultiQI            00000020-0000-0000-c000-000000000046 False    False
IClientSecurity   0000013d-0000-0000-c000-000000000046 False    False
IEventSystemTier2  609b954b-4fb6-11d1-9971-00c04fbbb345 True     False
IMarshal           00000003-0000-0000-c000-000000000046 False    False
IMarshal2          000001cf-0000-0000-c000-000000000046 False    False

PS D:\> $o = New-ComObject $cls
PS D:\> $o

InterfaceName Iid
----- ----
IUnknown      00000000-0000-0000-c000-000000000046
```

Parse registry information into a database.

```
PS> Get-ComDatabase
```

Get all COM classes from the current database.

```
PS> Get-ComClass
```

Get all COM classes which have a registered OOP server.

```
PS> Get-ComClass -ServerType LocalServer32
```

Get all COM classes with the name "EventSystemTier2" in their registration

```
PS> $cls = Get-ComClass -Name "EventSystemTier2"
```

DEMO



DCOM Security Research



Enumerate Classes



Verify Accessibility

DCOM Security

OLE Launch Security

Owner: S-1-5-21-528768928-1231760990-50533070-500
Group: S-1-5-21-528768928-1231760990-50533070-500
Integrity: Low

DACL **SACL**

ACL Entries

Type	Account	Access	Flags
Allowed	BUILTIN\Administrators	Execute	None
Allowed	NT AUTHORITY\INTERACTIVE	Execute	None
Allowed	NT AUTHORITY\SYSTEM	Execute	None
Allowed	NT AUTHORITY\SYSTEM	Execute	None

Specific Access

Name	Access Mask
<input checked="" type="checkbox"/> Execute	0x00000001
<input type="checkbox"/> Execute Local	0x00000002
<input type="checkbox"/> Execute Remote	0x00000004
<input type="checkbox"/> Activate Local	0x00000008
<input type="checkbox"/> Activate Remote	0x00000010

Launch = Create a new instance of the server.
Activate = Create new object on existing server.
Enforced in RPCSS

OLE Access Security

Owner: BUILTIN\Administrators
Group: BUILTIN\Administrators
Integrity: N/A

DACL

ACL Entries

Type	Account	Access	Flags
Allowed	NT AUTHORITY\SYSTEM	Execute, ExecuteLocal	None
Allowed	NT AUTHORITY\SELF	GenericAll	None
Allowed	NT AUTHORITY\INTERACTIVE	Execute, ExecuteLocal	None

Specific Access

Name	Access Mask
<input checked="" type="checkbox"/> Execute	0x00000001
<input checked="" type="checkbox"/> Execute Local	0x00000002
<input checked="" type="checkbox"/> Execute Remote	0x00000004

Access = Call methods on existing objects.
Enforced in Server Process
SELF = Process Token User SID

Print the COM launch security descriptor for a class to the console.

```
PS> Format-ComSecurityDescriptor $cls
```

Print the COM access security descriptor for a class to the console.

```
PS> Format-ComSecurityDescriptor $cls -ShowAccess
```

Get service hosted classes accessible by the current user.

```
PS> Get-ComClass -Service | Select-ComAccess
```

Get interactive user classes accessible by the token from PID 1234

```
PS> Get-ComClass -InteractiveUser |  
Select-ComAccess -ProcessId 1234
```

DEMO



DCOM Security Research



Enumerate Classes

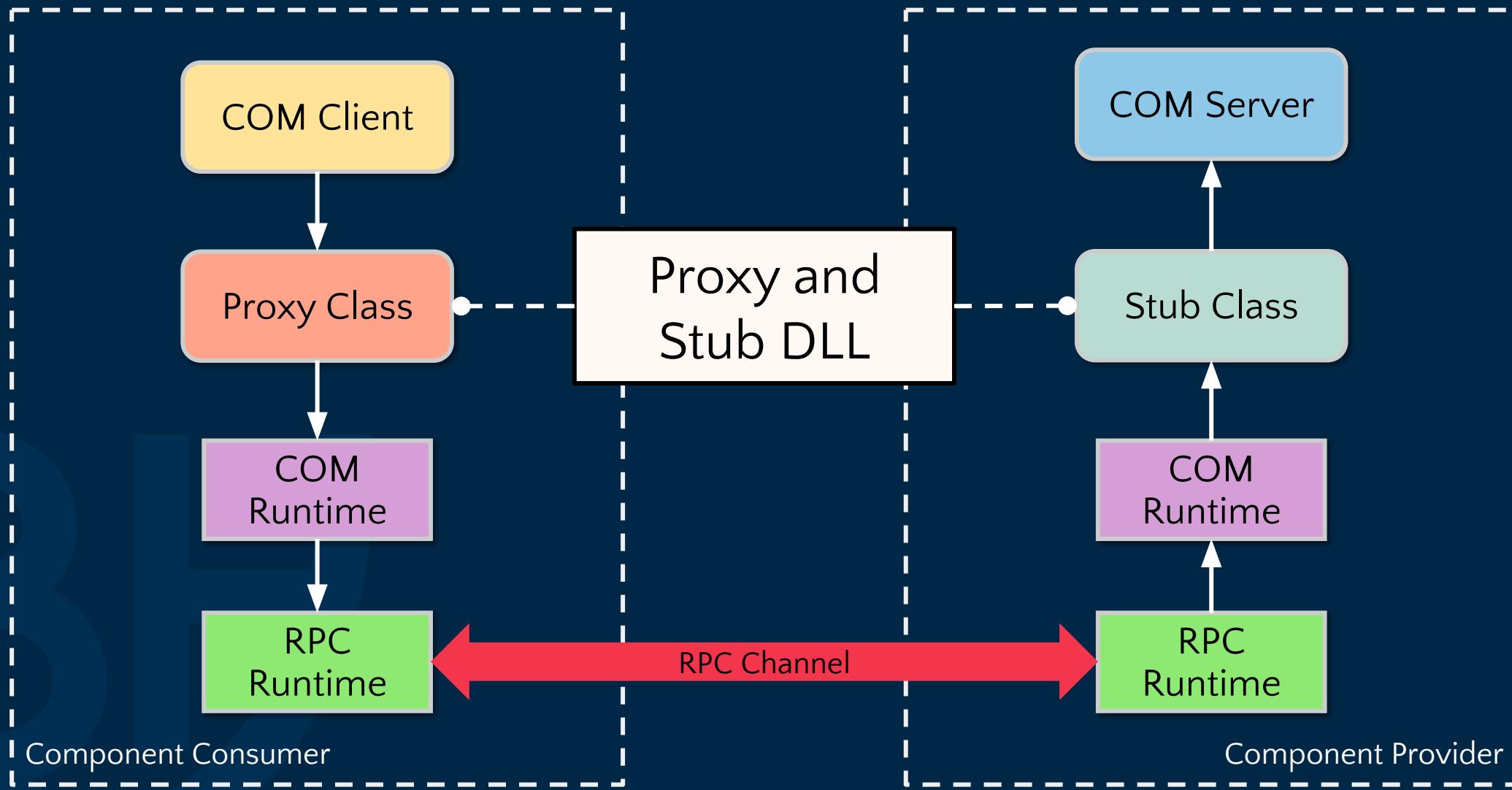


Verify Accessibility



Call Methods

DCOM Proxies



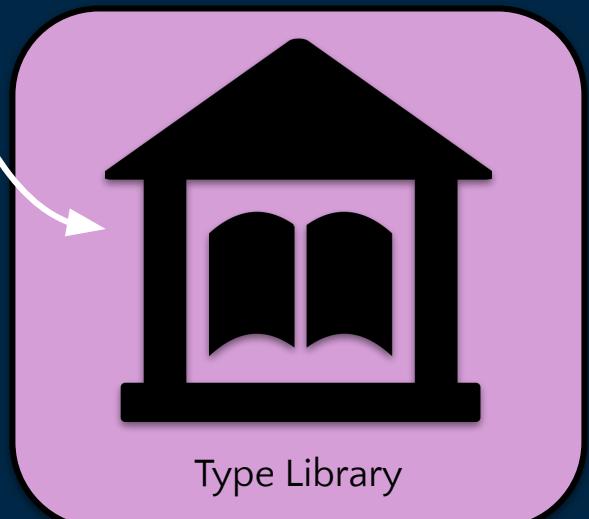
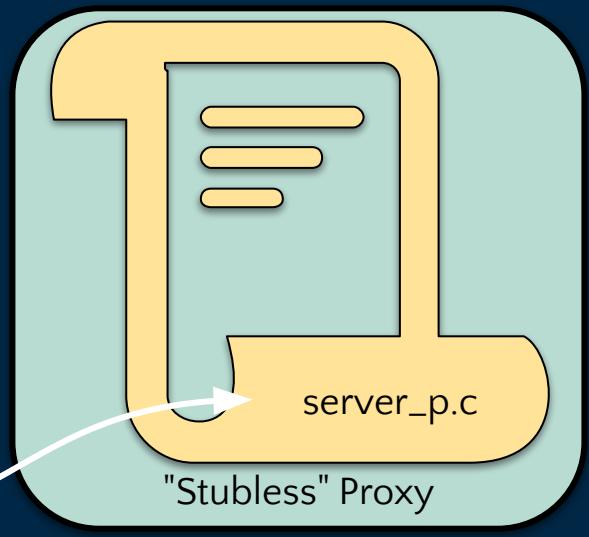
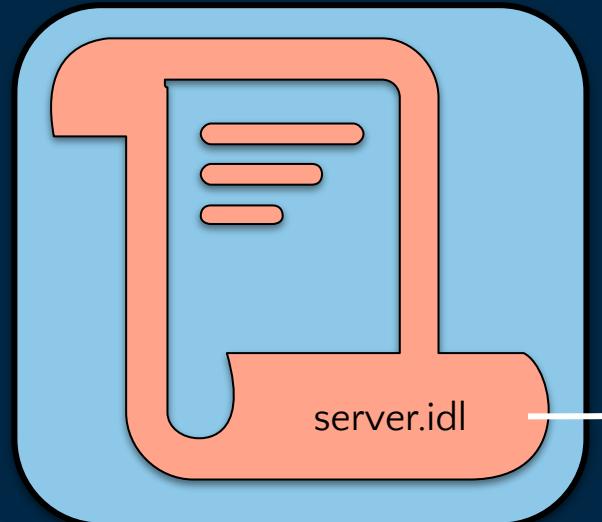
Proxy Interface Registration

The screenshot shows the Windows Registry Editor window. The left pane displays a tree view of registry keys under the path `HKEY_CLASSES_ROOT\Interface`. One key, `{475CA8F3-9417-48BC-B9D7-4163A7844C02}\ProxyStubClSID32`, is selected and highlighted in blue. The right pane shows a table with three columns: Name, Type, and Data. There is one entry in the table:

Name	Type	Data
ab (Default)	REG_SZ	{E1BA88BA-7A83-421A-A05D-71F96C3FFFD0}

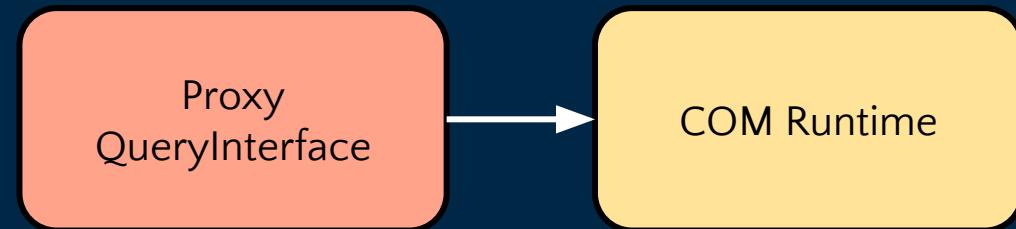
The status bar at the bottom of the window shows the full path: `Computer\HKEY_CLASSES_ROOT\Interface\{475CA8F3-9417-48BC-B9D7-4163A7844C02}\ProxyStubClSID32`.

Building a Proxy Class

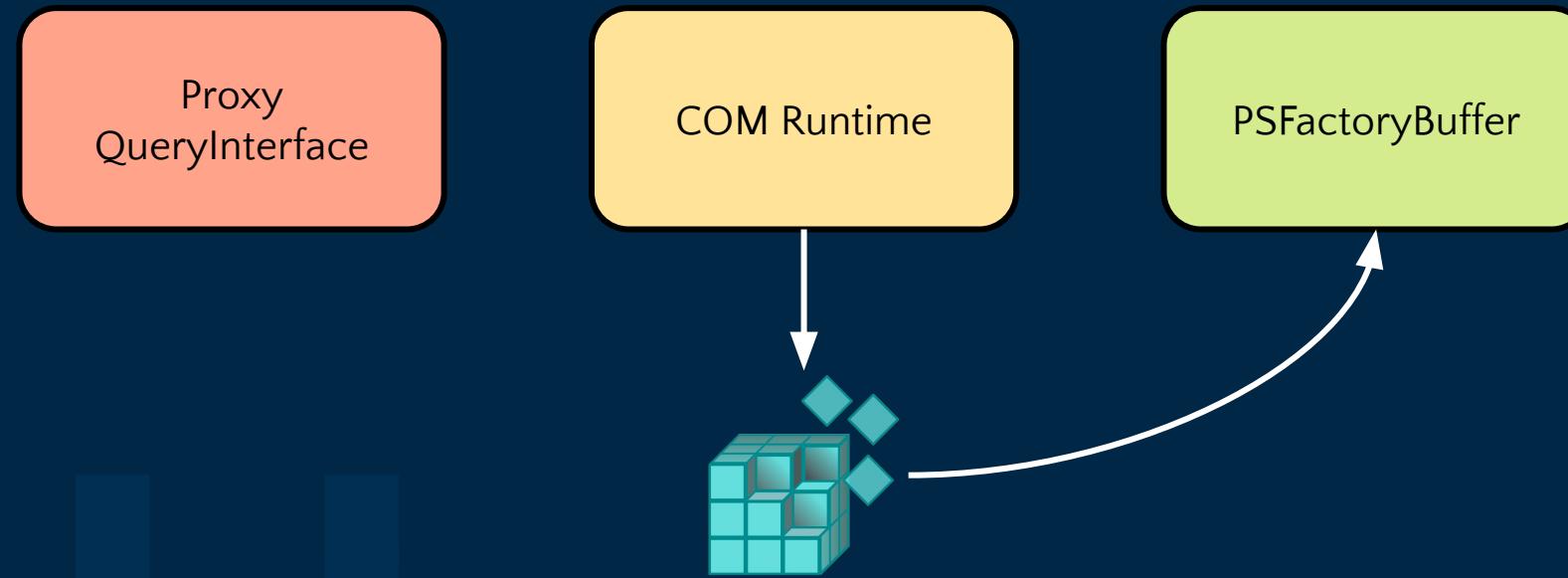


```
[  
    object,  
    uuid(CBF6EAC2-5AFF-4DD8-81B4-41B0F98728B3),  
    oleautomation  
]  
interface ISimpleObject : IUnknown  
{  
    HRESULT Run([out] DWORD* result);  
};
```

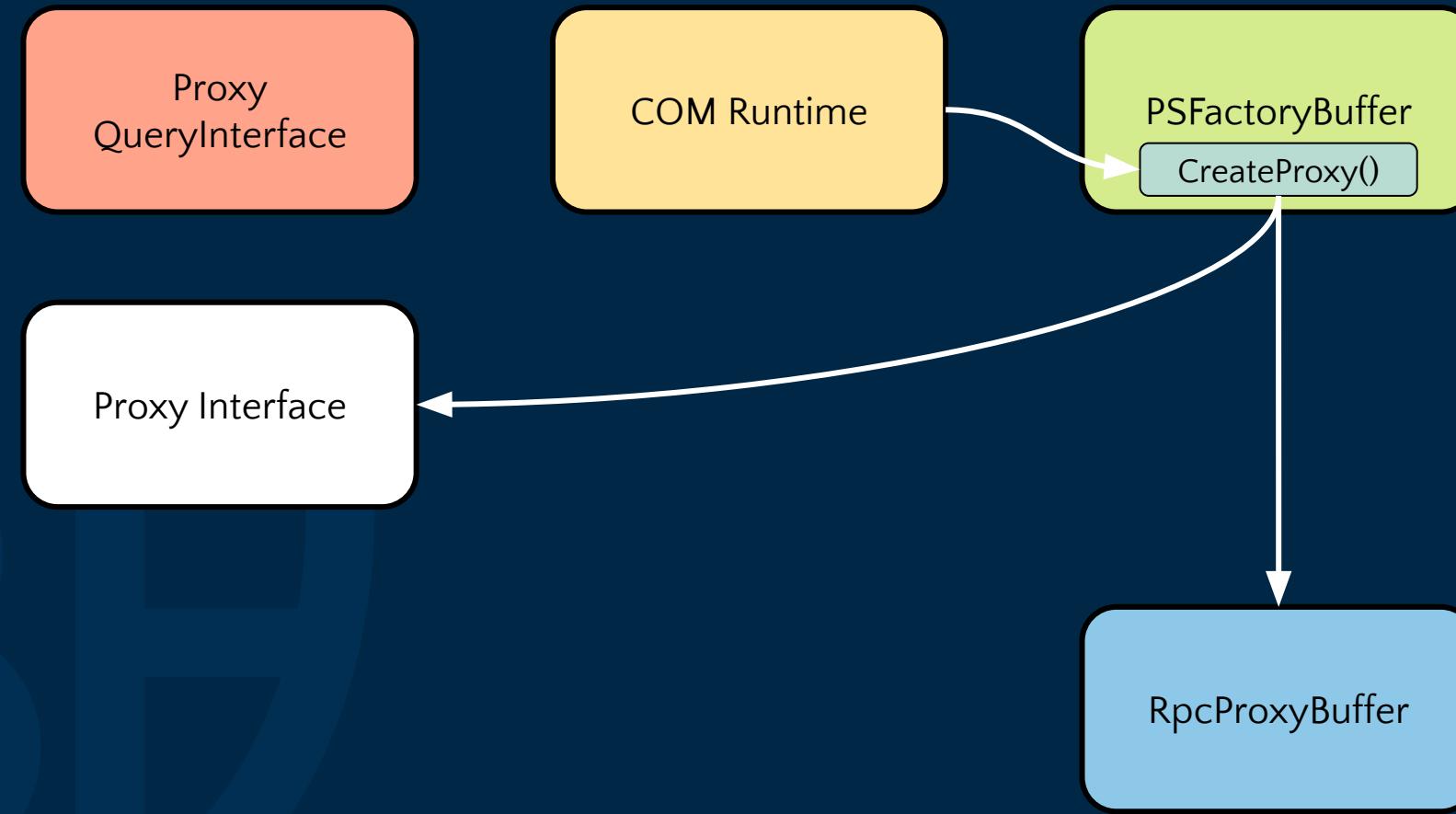
Creating a Proxy Instance



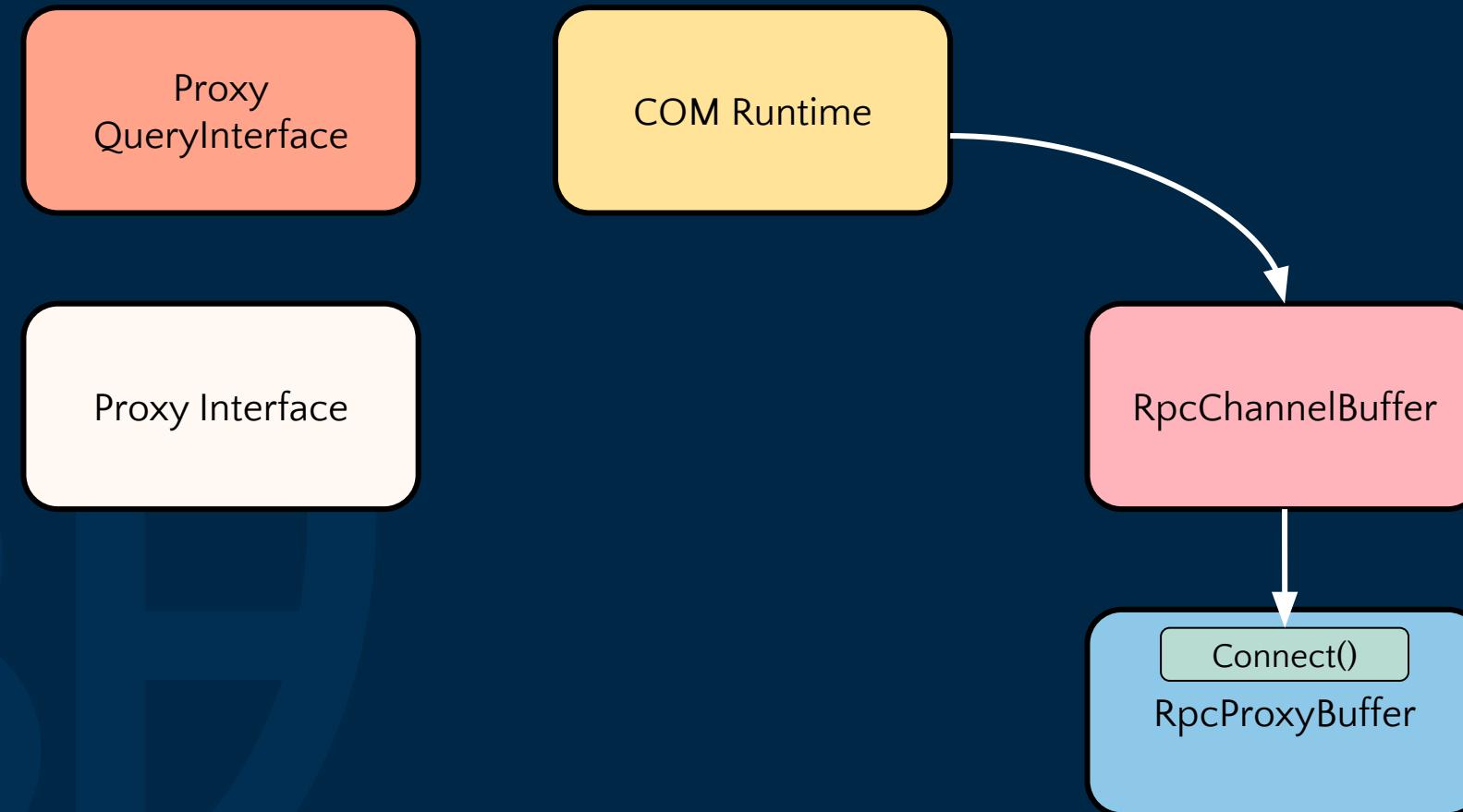
Creating a Proxy Instance



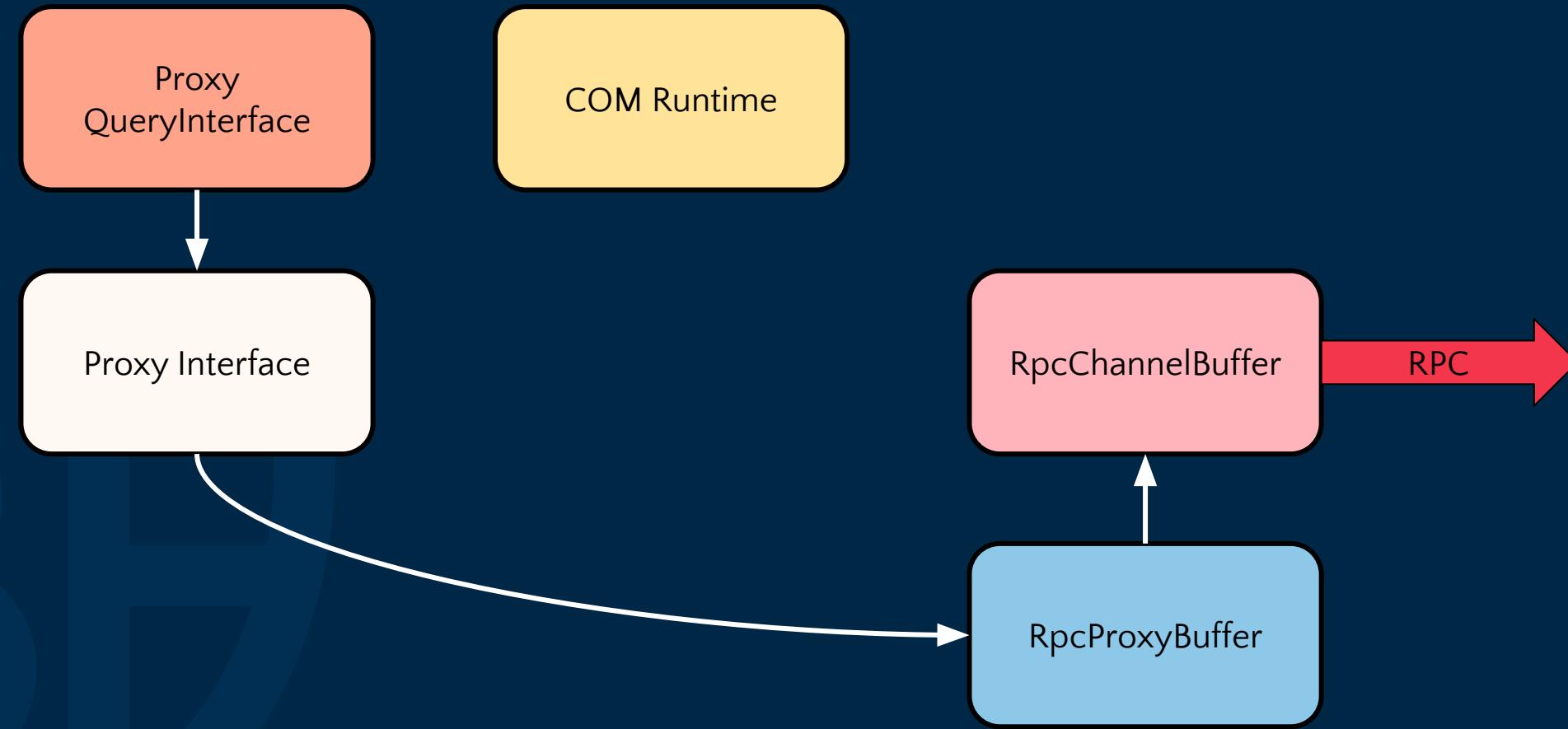
Creating a Proxy Instance



Creating a Proxy Instance



Creating a Proxy Instance



Enumerating Supported Interfaces

```
PS> Get-ComInterface -Class $cls
```

Name	IID	HasProxy	HasTypeLib
---	---	-----	-----
IUnknown	00000000-0000-... . . .	False	False
IDispatch	00020400-0000-... . . .	True	False
...			

Calling a Known Interface

```
PS> $src = @"using System;  
using System.Runtime.InteropServices;  
[ComImport, Guid("12345678-1234-1234-1234-121212121212"),  
 InterfaceType(ComInterfaceType.InterfaceIsIUnknown)]  
public interface IDemoInterface {  
    int Run();  
}"@  
PS> Add-Type -TypeDefintion $src  
PS> [IDemoInterface]$obj  
Cannot convert the value of type "System.__ComObject" to  
type "IDemoInterface".
```

A Second Try

```
PS> $src = @"
public static class DemoInterfaceCast {
    public static IDemoInterface Cast(object obj) {
        return (IDemoInterface)obj;
    }
}"@
PS> Add-Type -TypeDefinition $src
PS> $i = [DemoInterfaceCast]::Cast($obj)
PS> $i.Run()
Method invocation failed because [System.__ComObject]
does not contain a method named 'Run'.
```

Custom Forwarding Wrapper

```
public class CustomWrapper {  
    private readonly IDemoInterface _o;  
    public CustomWrapper(IDemoInterface o) {  
        _o = o;  
    }  
    public int Run() {  
        return _o.Run();  
    }  
}
```

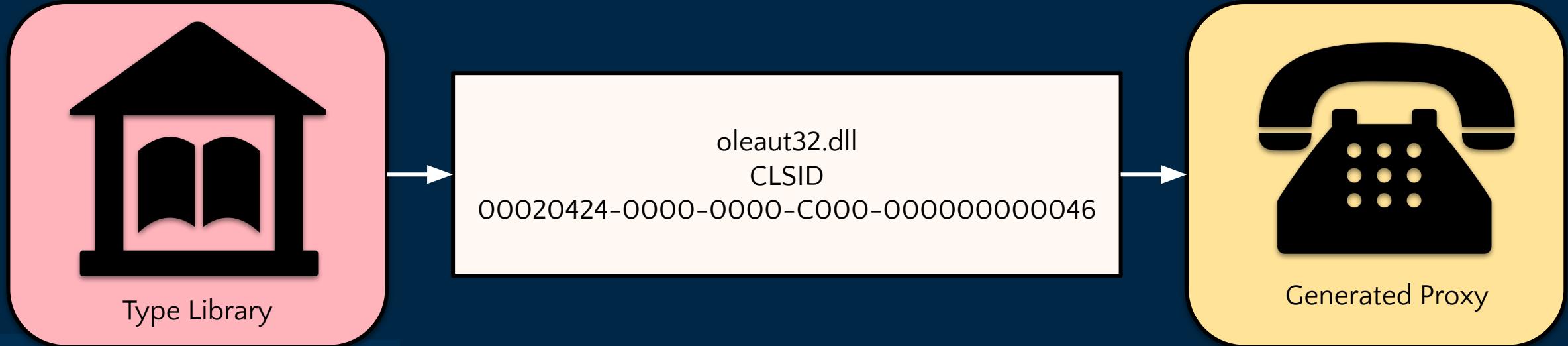
Custom Forwarding Wrapper

```
public class CustomWrapper {  
    private readonly IDemoInterface _o;
```

```
PS> $t = [IDemoInterface]  
PS> $i = Get-ComObjectInterface $obj -Type $t  
PS> $i.Run()  
12345678
```

```
}
```

Type Library Proxy



```
PS> $i = Get-ComInterface $cls | ? IsAutomationProxy
```

```
PS> $i | ConvertTo-ComSourceCode -Parse
```

Marshal.GetTypeForITypeInfo(IntPtr) Method

Namespace: [System.Runtime.InteropServices](#)

Assembly: mscorelib.dll

Converts an unmanaged [ITypeInfo](#) object into a managed [Type](#) object.

C#

 Copy

```
[System.Security.SecurityCritical]
public static Type GetTypeForITypeInfo (IntPtr piTypeInfo);
```

Marshal.GetTypeForITypeInfo(IntPtr) Method

Namespace: [System.Runtime.InteropServices](#)

```
PS> $o = Get-ComObjectInterface $obj -Interface $i
```

C#

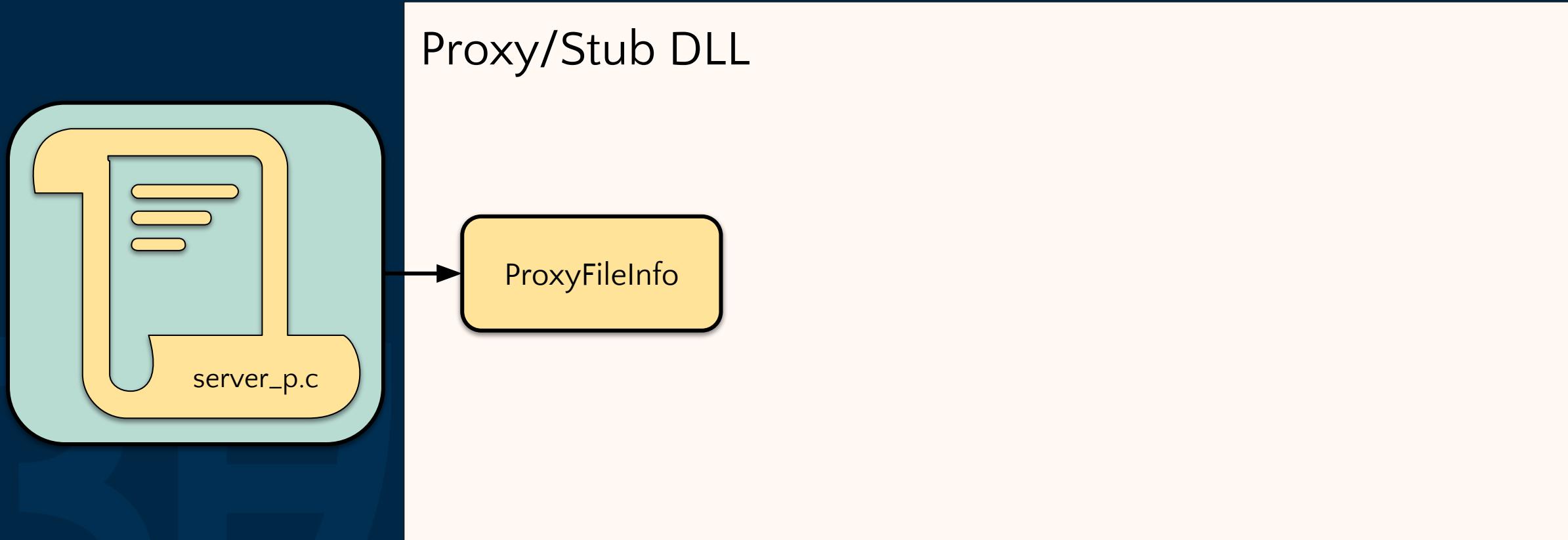
 Copy

```
[System.Security.SecurityCritical]
public static Type GetTypeForITypeInfo (IntPtr piTypeInfo);
```

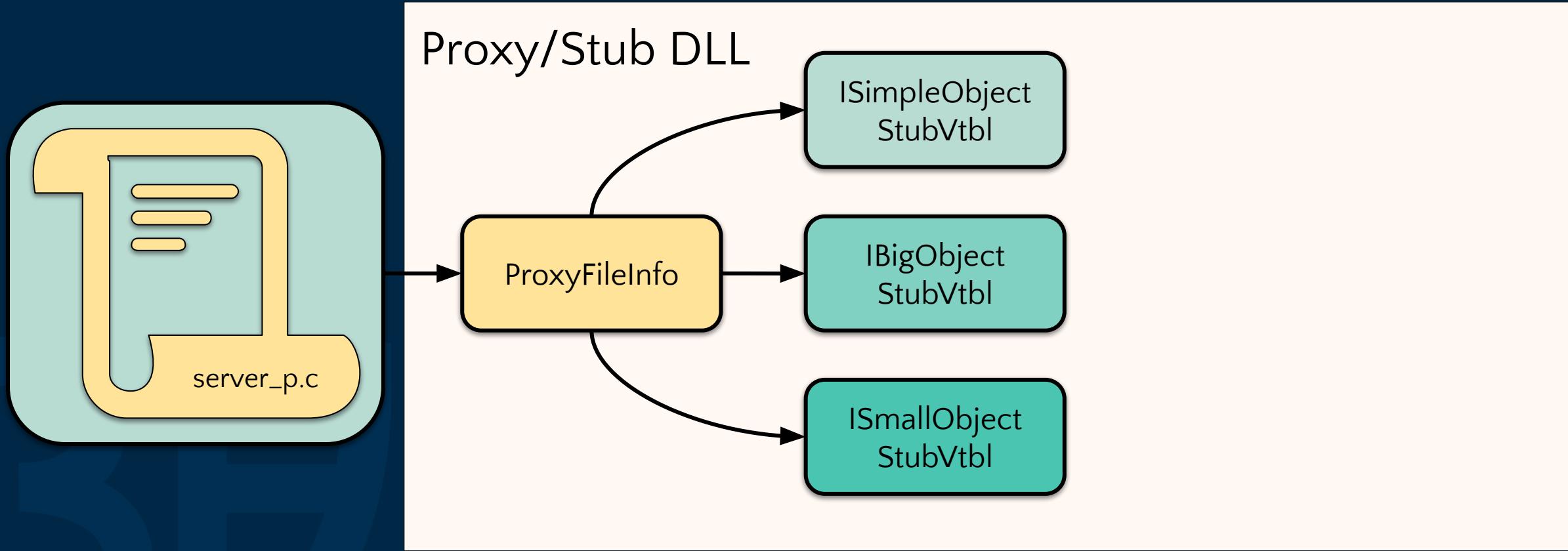
DEMO



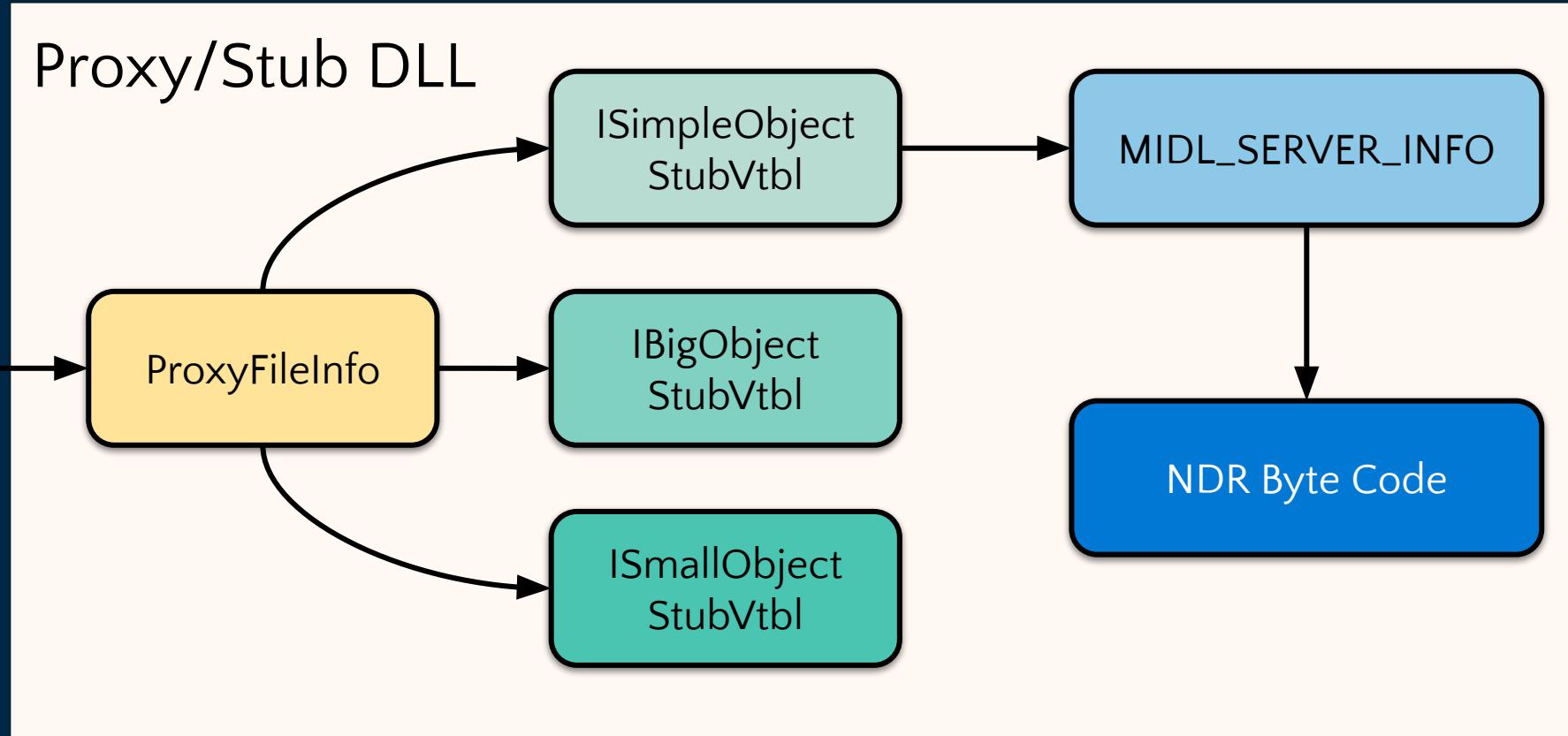
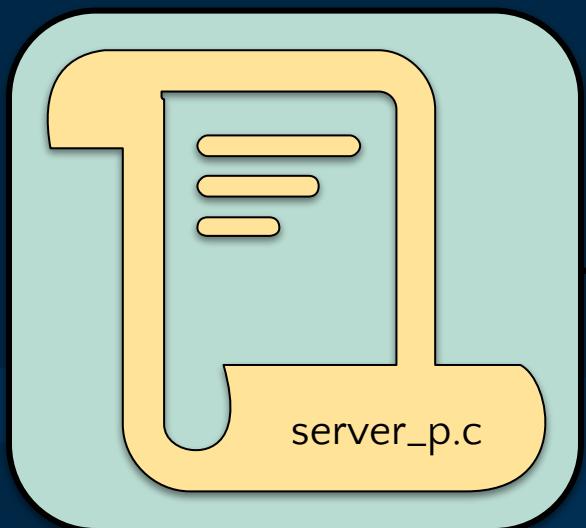
Stubless Proxy



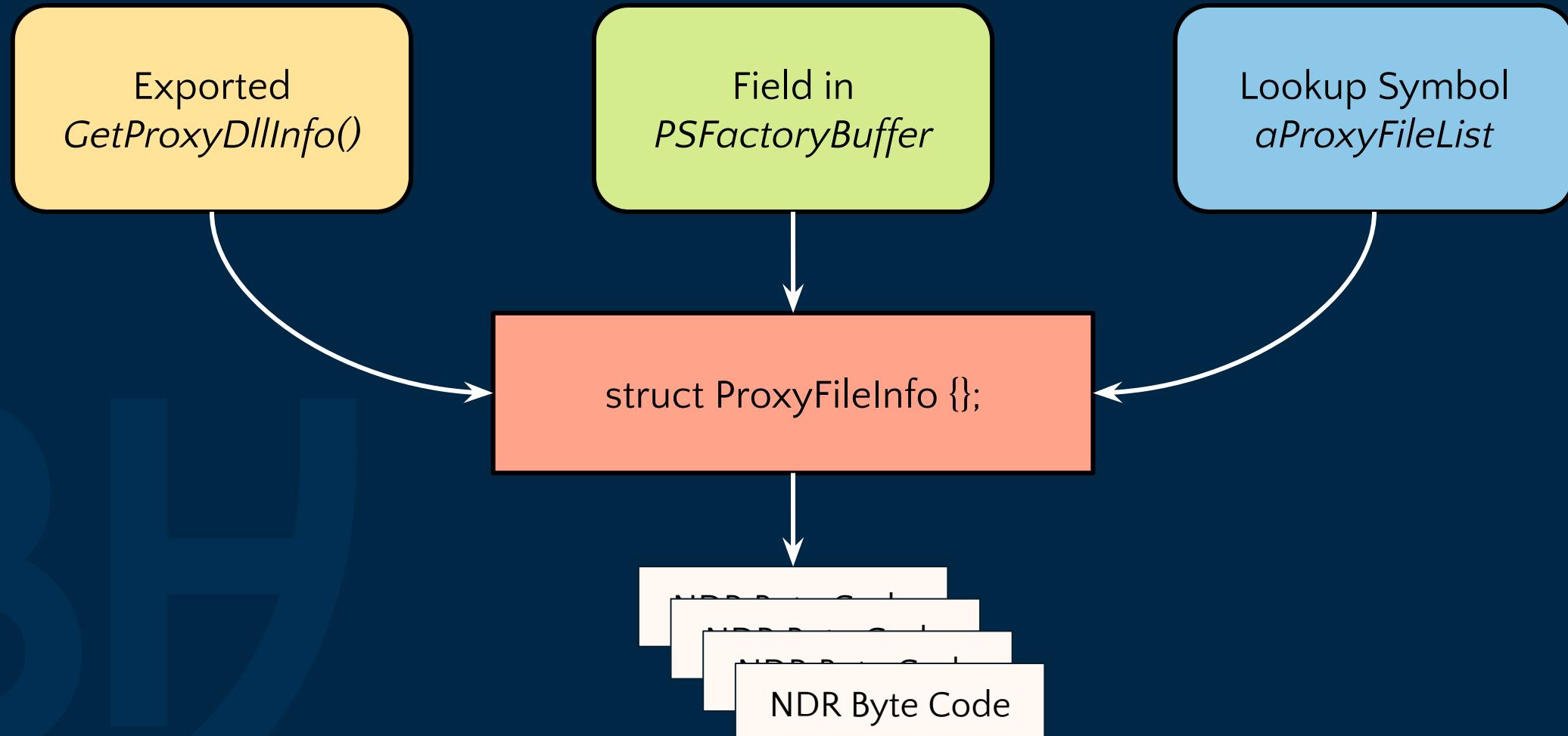
Stubless Proxy



Stubless Proxy



Extract Proxy Information



Converting NDR to a Callable Interface

Generate .NET COM Interface

Pros:

- Use built-in COM/.NET marshalling

Cons:

- Difficult to represent complex structures
- Passing invalid data could crash process

Generate .NET RPC Client

Pros:

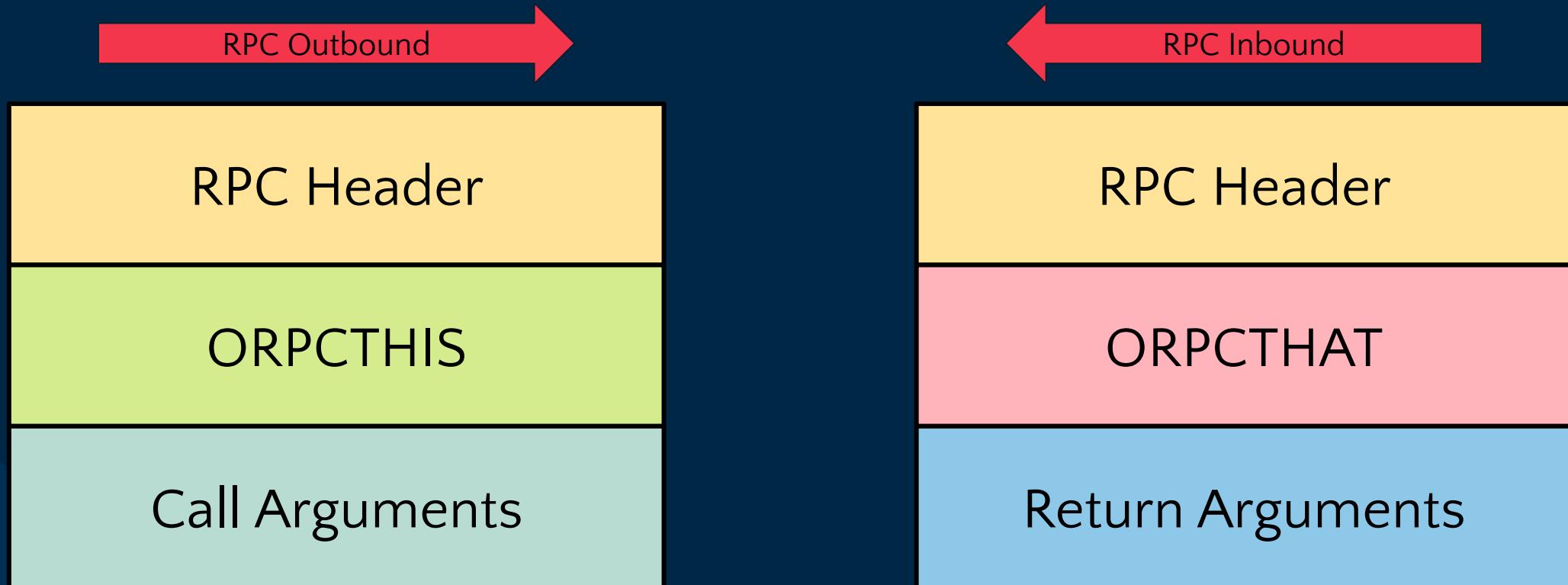
- Accurate representation of underlying proxy call
- Can't crash processes^t

Cons:

- Need to implement the RPC transport

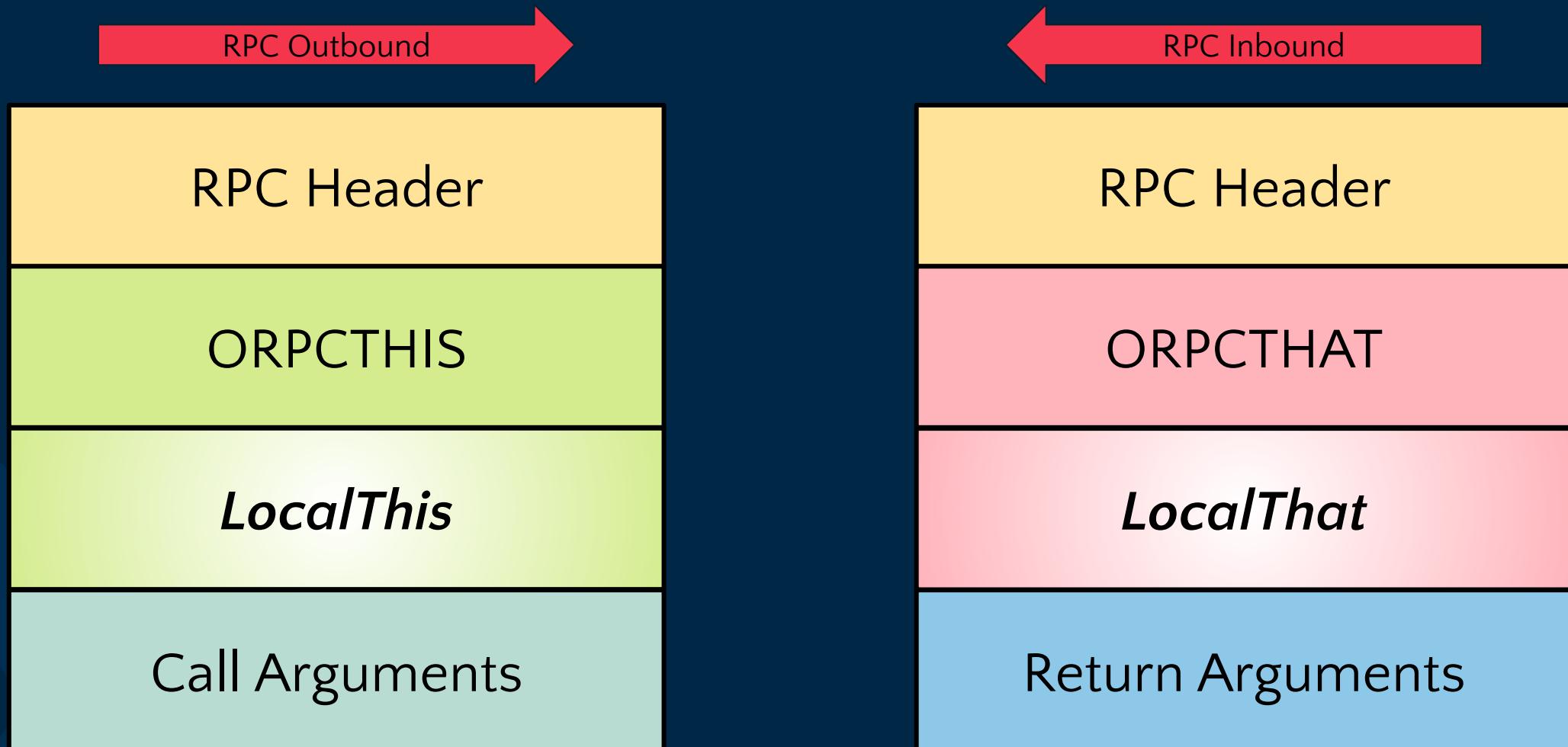
+ Never say never

DCOM RPC Protocol



https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-dcom/86b9cf84-df2e-4f0b-ac22-1b957627e1ca

"ALPC" DCOM RPC Protocol



Reusing IRpcChannelBuffer

```
void NdrProxyInitialize(  
    [in] void *This,  
    [in] PRPC_MESSAGE pRpcMsg,  
    [in, out] PMIDL_STUB_MESSAGE pStubMsg,  
    [in] PMIDL_STUB_DESC pStubDescriptor,  
    [in] unsigned int ProcNum  
) ;
```

Pass pointer to proxy

Reusing IRpcChannelBuffer

```
void NdrProxyInitialize(  
    [in]      void             *This,  
    [in]      PRPC_MESSAGE     pRpcMsg,  
    [in, out] PMIDL_STUB_MESSAGE pStubMsg,  
    [in]      PMIDL_STUB_DESC   pStubDescriptor,  
    [in]      unsigned int      ProcNum  
) ;
```

```
struct MIDL_STUB_MESSAGE {  
    // ...  
    IRpcChannelBuffer* pRpcChannelBuffer;  
    // ...  
};
```

Reusing IRpcChannelBuffer

```
void NdrProxyInitialize(  
    [in]      void          *This,  
    [in]      PRPC_MESSAGE  pRpcMsg,  
    [in, out] PMIDL_STUB_MESSAGE pStubMsg,  
    [in]      PMIDL_STUB_DESC   pStubDescr  
    [in]      unsigned int    ProcNum  
)
```

```
struct MIDL_STUB_MESSAGE {  
    // ...  
    IRpcChannelBuffer* pRpcChannelBuffer;  
    // ...  
};
```

```
public byte[] SendReceive(byte[] ndr_data, int proc_num) {  
    RPC_MESSAGE msg = new()  
    {  
        BufferLength = ndr_data.Length,  
        ProcNum = proc_num  
    };  
    m_buffer.GetBuffer(ref msg, in m_iid);  
    Marshal.Copy(ndr_data, 0, msg.Buffer, ndr_data.Length);  
    m_buffer.SendReceive(ref msg, out int status);  
    byte[] ret = new byte[msg.BufferLength];  
    Marshal.Copy(msg.Buffer, ret, 0, msg.BufferLength);  
    m_buffer.FreeBuffer(ref msg);  
    return ret;  
}
```

Putting it All Together

```
C:\WINDOWS\System32\WindowsPowerShell\v1.0\> PS D:\> $o = New-ComObject $cls -Iid dfd6ad8d-c419-4027-95e9-8f435f76bcfc
PS D:\> $o | gm
TypeName: IWSearchForContainerAgent_RpcClientWrapper
Create object with IID

Name      MemberType Definition
----      -----
Dispose   Method    void IDisposable.Dispose()
Equals    Method    bool Equals(System.Object obj)
GetHashCode Method   int GetHashCode()
GetType   Method    type GetType()
Proc3     Method    int Proc3()
Proc4     Method    int Proc4(string p0, System.Nullable[guid] p1, System.Nullable[g...]
Proc5     Method    int Proc5(string p0)
Proc6     Method    int Proc6(string p0, int p1)
QueryInterface Method  OleViewDotNet.Wrappers.BaseComWrapper.QueryInterface(guid iid)
ToString   Method    string ToString()
Unwrap    Method    System.Object Unwrap()
Iid      Property   guid Iid {get;}

Callable Proxy Methods
```

DEMO



Conclusions

Get the Tooling

GITHub Source
<https://oleview.net>

PS> Install-Module
"OleViewDotNet"



© Copyright Microsoft Corporation. All rights reserved.

BLUEHAT
SECURITY ABOVE ALL ELSE