



Demystifying COM

James Forshaw (@tiraniddo)
Troopers 2017

Agenda

- Component Object Model (COM) Internals
- Attacking COM
 - Enumerating attack surface for EoP
 - Reverse engineering COM components
- Bugs and “Features”

We'll be using my OleViewDotNet tool throughout.

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

<https://github.com/tyranid/oleviewdotnet.binaries>

<https://goo.gl/Bkhlos>



COM Internals

COM is Scary!

**“Any sufficiently
complex middleware is
indistinguishable from
magic.”**

Arthur C. Clarke's Third Law of Software Development

Mostly Documented'ish

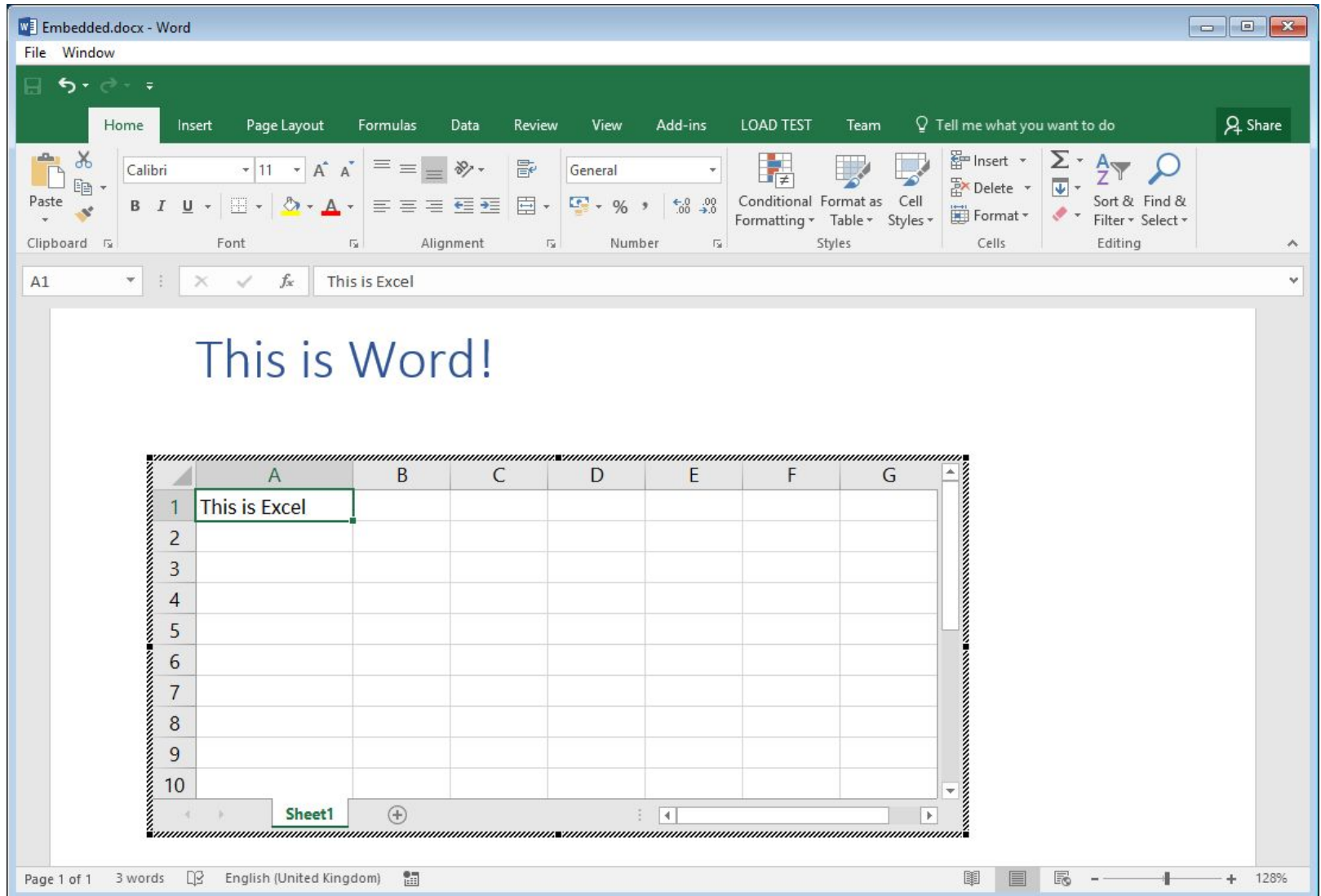
[MS-DCOM]:

Distributed Component Object Model (DCOM) Remote Protocol

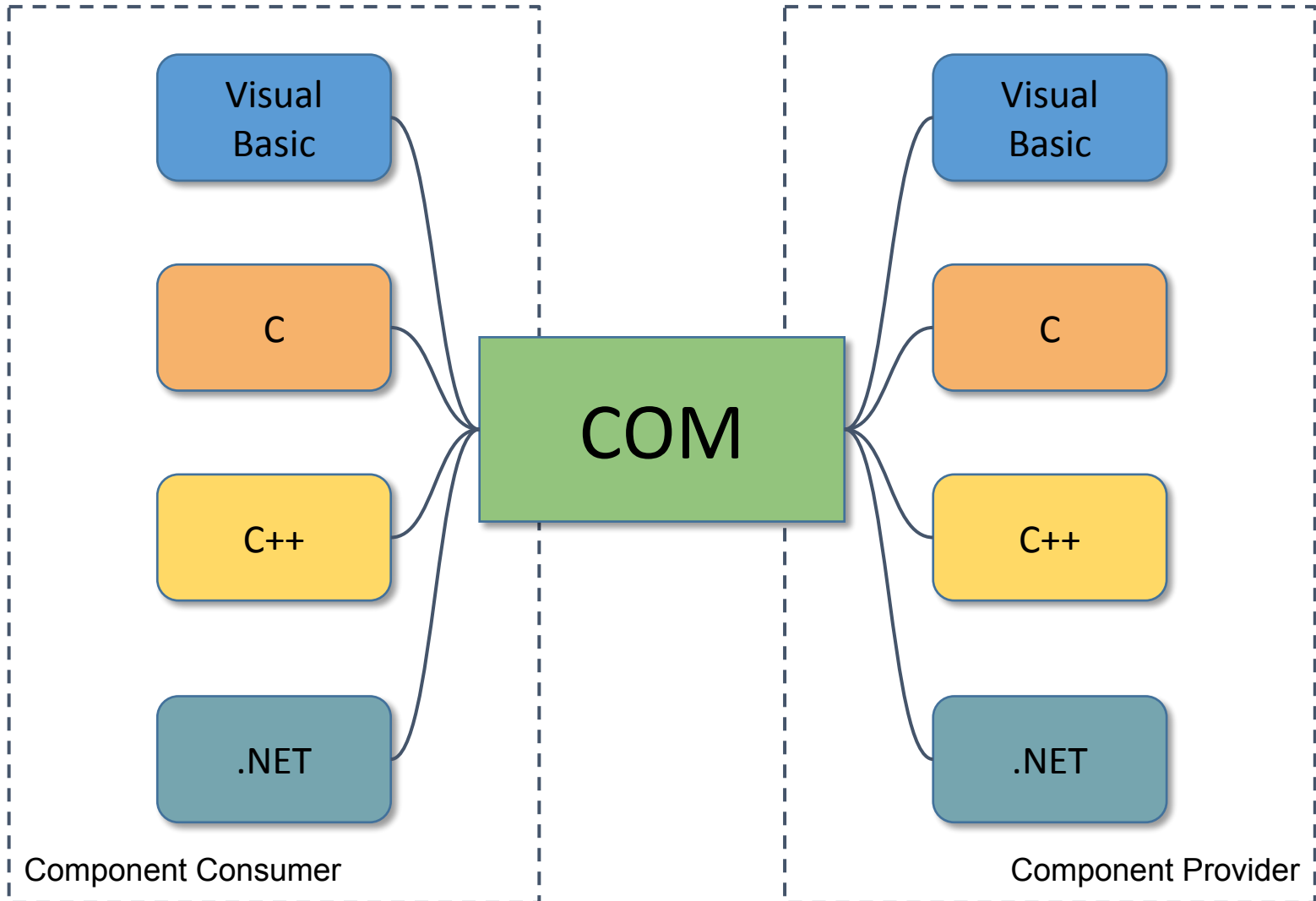
Intellectual Property Rights Notice for Open Specifications Documentation

- **Technical Documentation.** Microsoft publishes Open Specifications documentation (“this documentation”) for protocols, file formats, data portability, computer languages, and standards support. Additionally, overview documents cover inter-protocol relationships and interactions.
- **Copyrights.** This documentation is covered by Microsoft copyrights. Regardless of any other terms that are contained in the terms of use for the Microsoft website that hosts this documentation, you can make copies of it in order to develop implementations of the technologies that are described in this documentation and can distribute portions of it in your implementations that use these technologies or in your documentation as necessary to properly document the implementation. You can also distribute in your implementation, with or without modification, any schemas, IDLs, or code samples that are included in the documentation. This permission also applies to any documents that are referenced in the Open Specifications documentation.

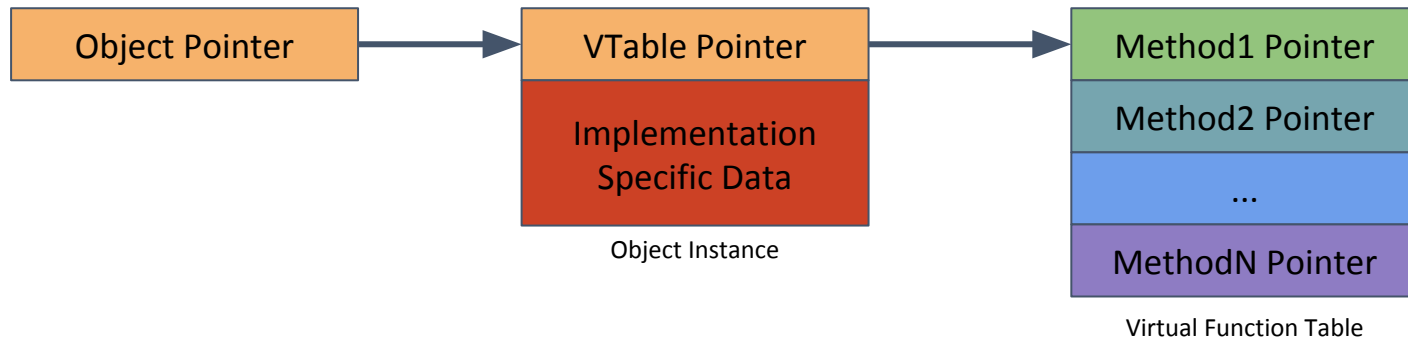
In the Beginning was OLE



Interoperability Heaven



Common ABI



```
struct ObjectVTable {  
    void (*SetInt)(struct Object* This, int i);  
    int (*GetInt)(struct Object* This);  
};  
  
struct Object {  
    struct ObjectVTable* Vtbl;  
    // Implementation specific data follows.  
};  
  
struct Object* obj;  
obj->Vtbl->SetInt(obj, 1234);
```

Object pointer is first.
Arguments passed left to right

VTable Pointer at start

C++ Implementation

```
struct Interface {  
    virtual void SetInt(int i) = 0;  
    virtual int GetInt() = 0;  
};  
  
class Object : public Interface {  
public:  
    void SetInt(int i) {}  
    int GetInt() {}  
};  
  
Interface* obj = new Object;  
obj->SetInt(1234);
```

Define a pure virtual "Interface"

Derive from Interface and implement

The Casting Problem

```
struct Interface1 {  
    virtual void A() = 0;  
};  
  
struct Interface2 {  
    virtual void B() = 0;  
};  
  
class Object : public Interface1,  
               public Interface2 {  
    void A() {}  
    void B() {}  
};
```

```
// Okay (mostly).  
Interface1* intf1 = new Object;  
// Incredibly bad idea.  
Interface2* intf2 = (Interface2*)intf1;
```


IUnknown, the Root of all COM Evil

```
DEFINE_GUID(IID_IUnknown,  
            "00000000-0000-0000-C000-000000000046");  
struct IUnknown {  
    HRESULT QueryInterface(GUID& iid, void** ppv);  
    LONG AddRef();  
    LONG Release();  
};
```

Standard
COM error
code.
>= 0 is
Success

Used to
reference count
the object.

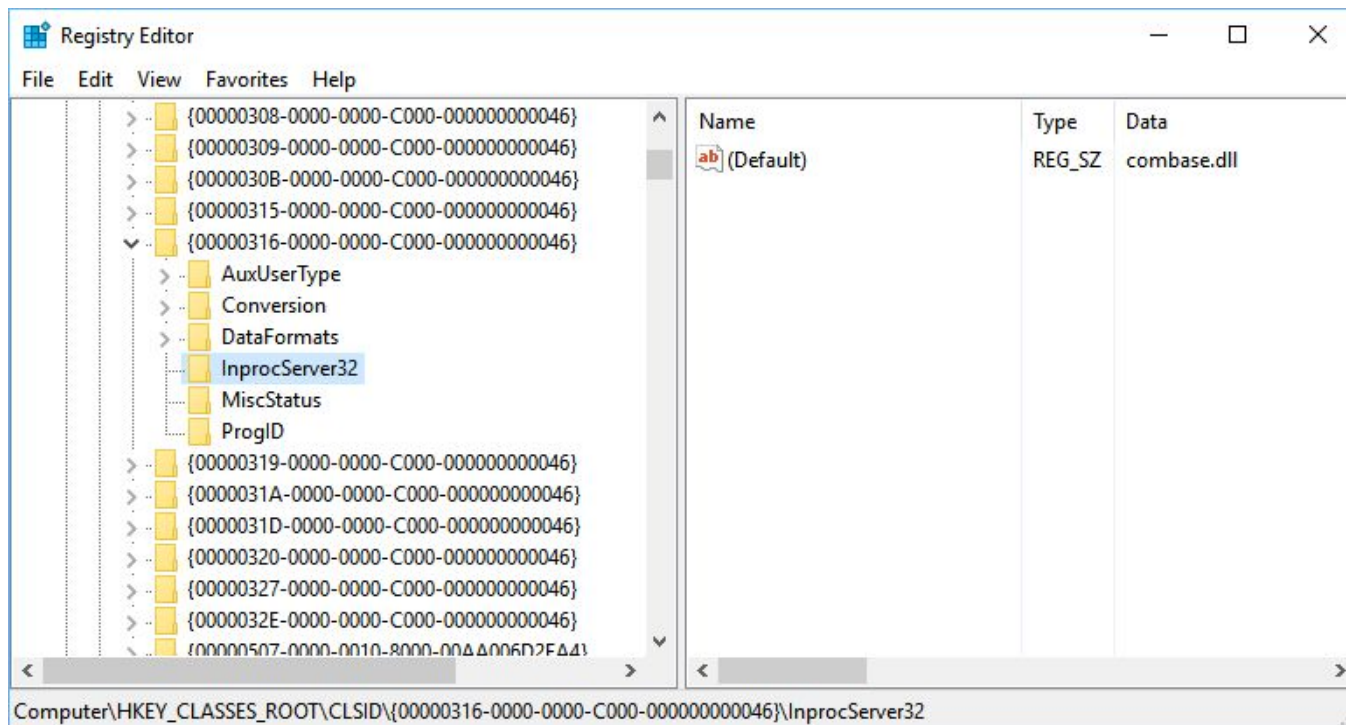
Interfaces defined using a
128 bit Globally Unique ID.

```
struct Interface1 : public IUnknown {};  
struct Interface2 : public IUnknown {};  
  
Interface2* intf2;  
if (intf1->QueryInterface(IID_Interface2,  
                          (void**)&intf2) >= 0) {  
    // Success, we can call methods.  
    intf2->Release();  
}
```

Cast is a GIANT
code smell!
~\(_ツ)_/~

Class Registration

- Each Class Factory has a unique GUID, the Class ID (CLSID)
- Classes are registered in the Windows Registry under HKEY_CLASSES_ROOT\CLSID\{GUID}
- Registration information depends on the type of server



Class Factories

- Component classes can't be directly 'newed' so COM defines a factory interface, *IClassFactory*

```
DEFINE_GUID(IID_ClassFactory,  
            "00000001-0000-0000-C000-000000000046");  
  
struct IClassFactory : public IUnknown {  
    HRESULT CreateInstance(  
        IUnknown *pUnkOuter, ← Used for COM  
        REFIID riid,           Aggregation. Best  
        void **ppvObject);     ignored :-)  
  
    HRESULT LockServer(BOOL fLock);  
};
```


Creating Class Factories and Instances

```
HRESULT CoGetClassObject(  
    REFCLSID      rclsid,  
    DWORD         dwClsContext,  
    COSERVERINFO *pServerInfo,  
    REFIID        riid,  
    LPVOID        *ppv  
);
```

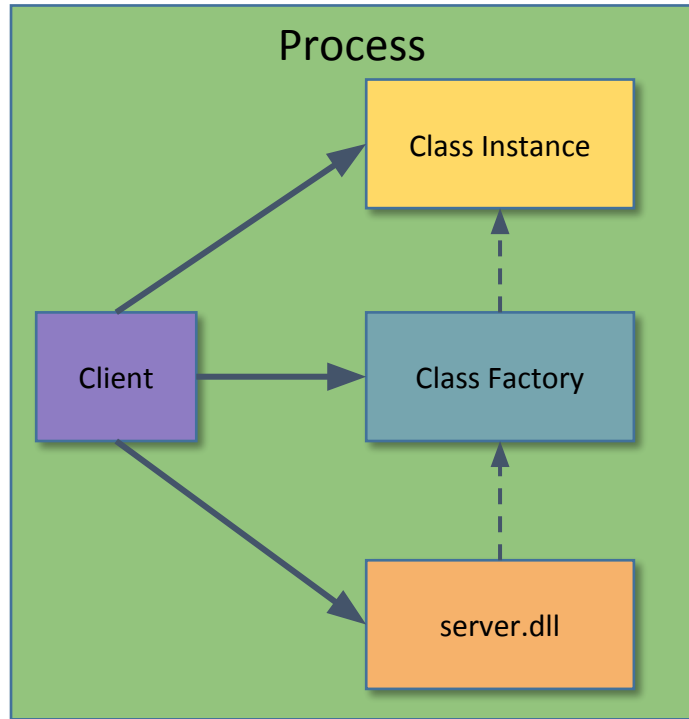
Specifies what type of server to lookup:

- CLSCTX_INPROC_SERVER
- CLSCTX_INPROC_HANDLER
- CLSCTX_LOCAL_SERVER
- CLSCTX_REMOTE_SERVER

Specify remote server
information is required
(more on this later).

```
HRESULT CoCreateInstanceEx(  
    REFCLSID      rclsid,  
    IUnknown      *punkOuter,  
    DWORD         dwClsCtx,  
    COSERVERINFO *pServerInfo,  
    DWORD         dwCount,  
    MULTI_QI      *pResults  
);
```

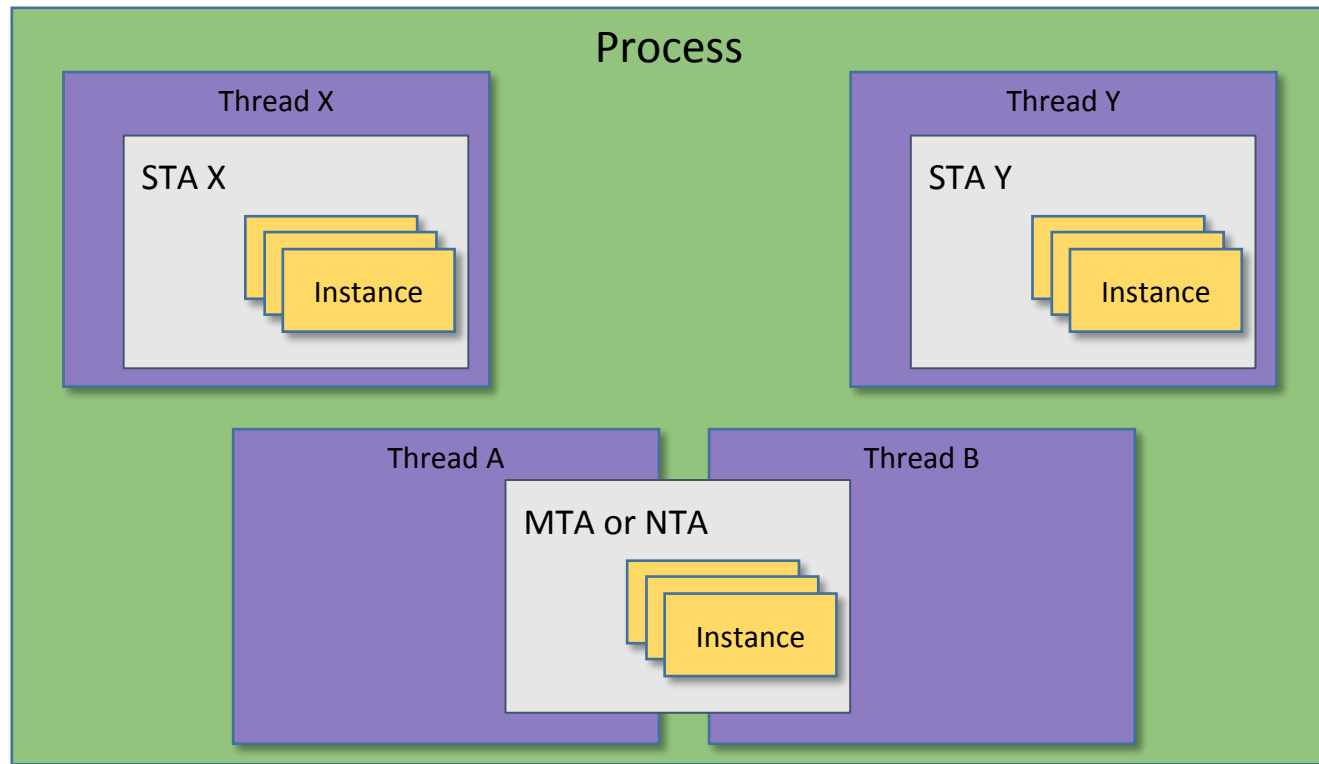
In-Process Server



- DLL server filename specified in *InProcServer32* key.
- DLL loaded into process and class factory created by calling exported method:

```
HRESULT DllGetClassObject(REFCLSID rclsid,  
                           REFIID riid,  
                           LPVOID *ppv);
```

COM Apartments

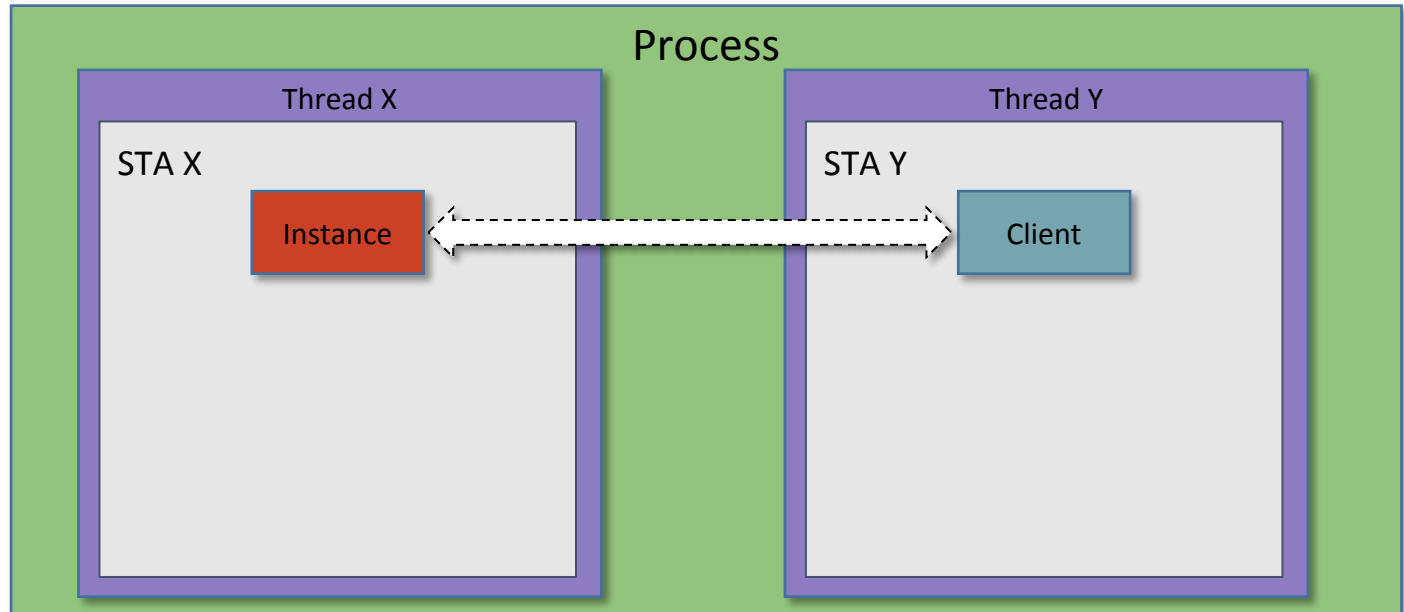


```
HRESULT CoInitializeEx(  
    LPVOID pvReserved,  
    DWORD dwCoInit  
);
```

COINIT_APARTMENTTHREADED
or
COINIT_MULTITHREADED

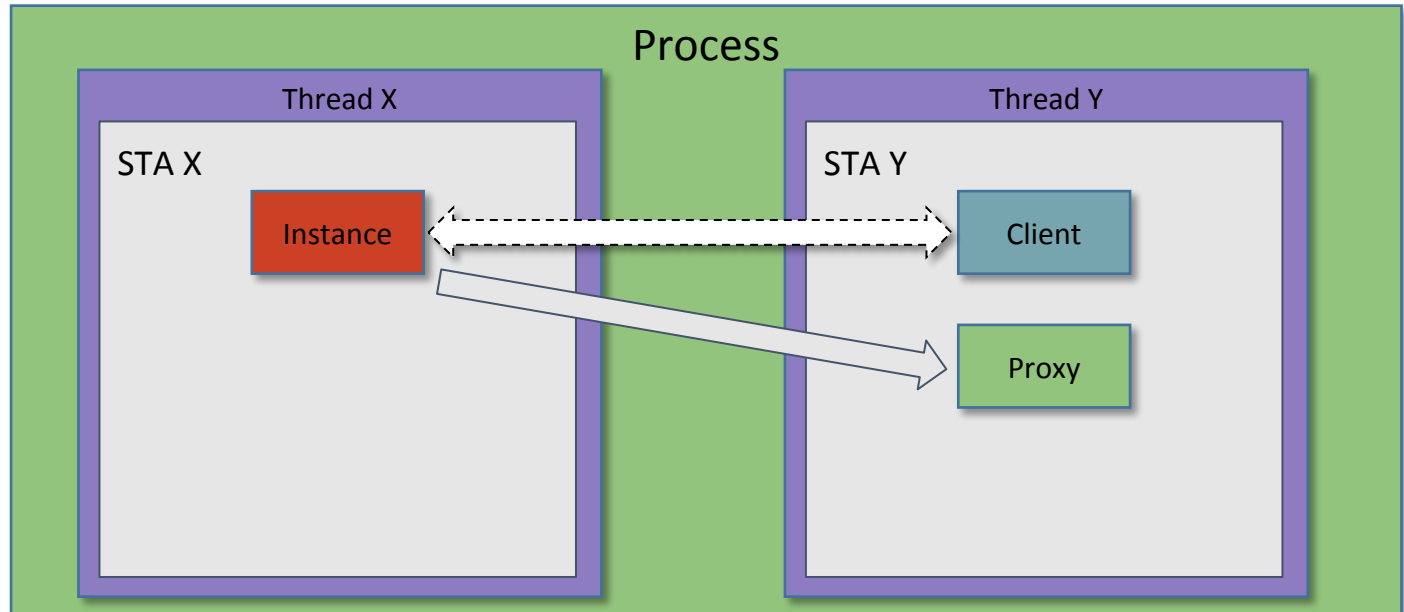
Single Threaded Apartments (STA)

- One STA per-thread. Initialized with `COINIT_APARTMENTTHREADED`.
- Created instance can only be called directly from the thread it was created on.
- Any other callers need to “Unmarshal” a Proxy to the object. Calls are sent to the original thread via a hidden Window and Window messaging



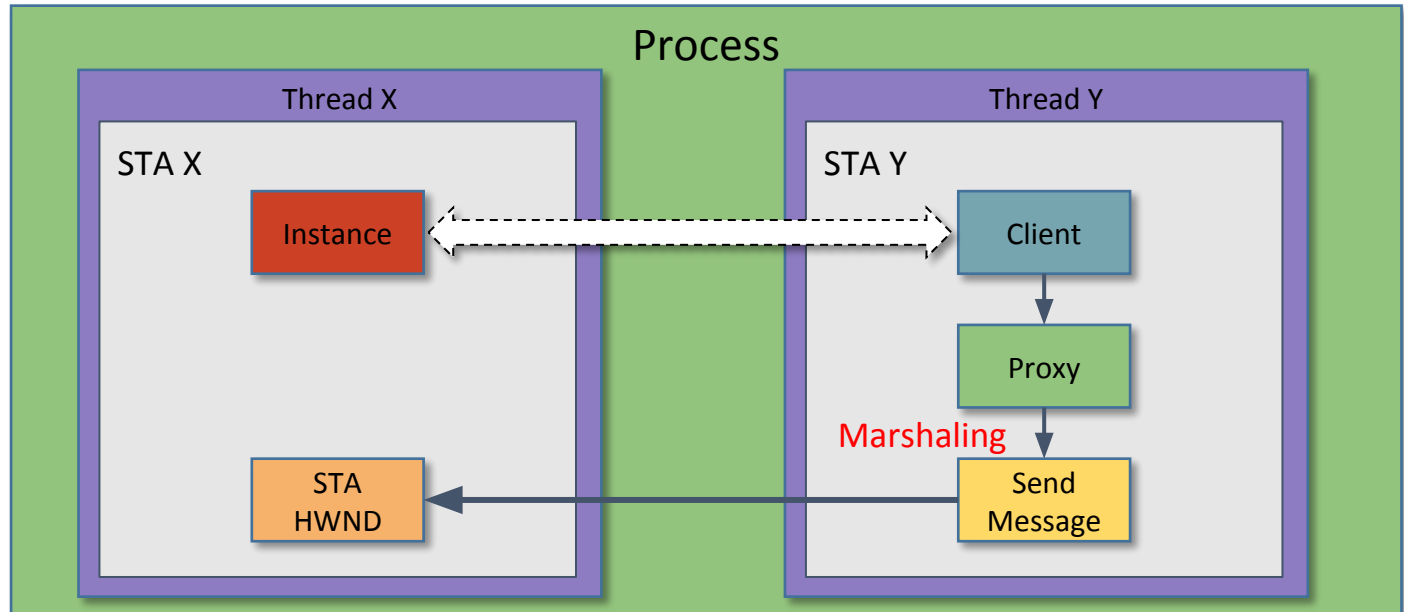
Single Threaded Apartments (STA)

- One STA per-thread. Initialized with `COINIT_APARTMENTTHREADED`.
- Created instance can only be called directly from the thread it was created on.
- Any other callers need to “Unmarshal” a Proxy to the object. Calls are sent to the original thread via a hidden Window and Window messaging



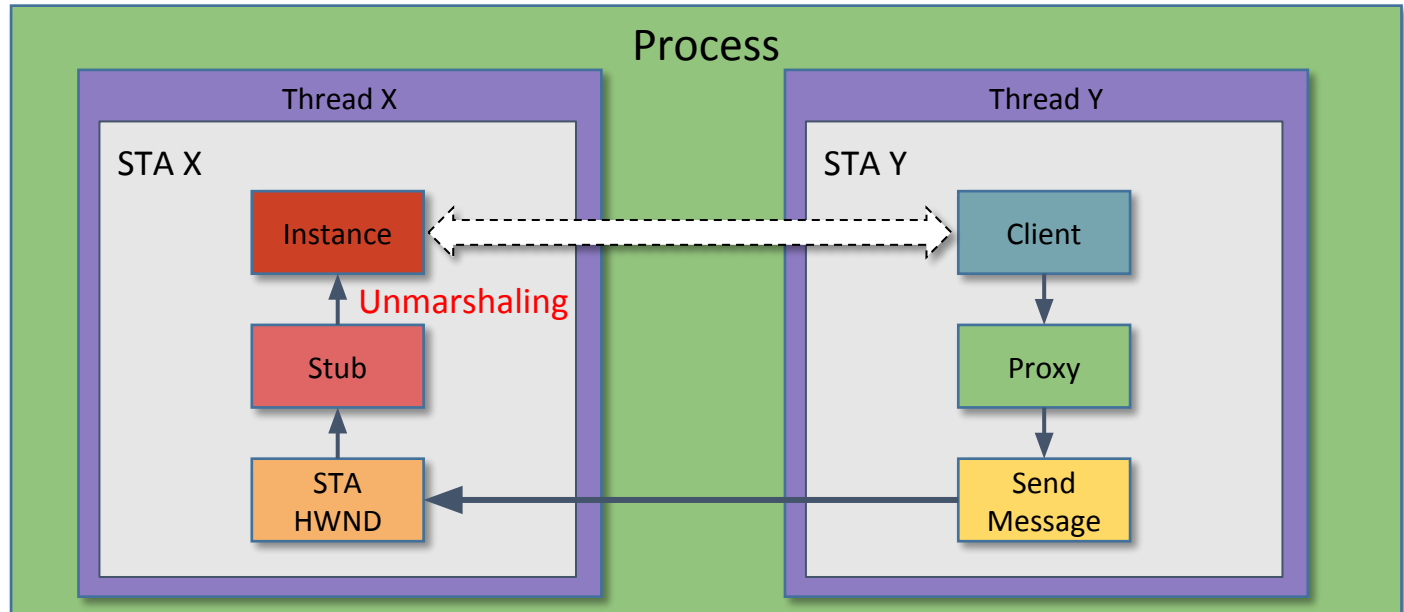
Single Threaded Apartments (STA)

- One STA per-thread. Initialized with COINIT_APARTMENTTHREADED.
- Created instance can only be called directly from the thread it was created on.
- Any other callers need to “Unmarshal” a Proxy to the object. Calls are sent to the original thread via a hidden Window and Window messaging



Single Threaded Apartments (STA)

- One STA per-thread. Initialized with COINIT_APARTMENTTHREADED.
- Created instance can only be called directly from the thread it was created on.
- Any other callers need to “Unmarshal” a Proxy to the object. Calls are sent to the original thread via a hidden Window and Window messaging



The Mystery Window

- Initializing a STA creates a hidden “Message Only” Window

```
HWND hwnd = FindWindowEx (HWND_MESSAGE, NULL, NULL, NULL);  
while (hwnd) {  
    WCHAR name[256] = {};  
    if (GetWindowText (hwnd, name, _countof (name))  
        && _wcsnicmp (name, L"ole", 3) == 0) {  
        DWORD pid = 0;  
        DWORD tid = GetWindowThreadProcessId (hwnd, &pid);  
        printf ("%p %5d %5d %ls\n", hwnd, pid, tid, name);  
    }  
    hwnd = GetNextWindow (hwnd, GW_HWNDNEXT);  
}
```

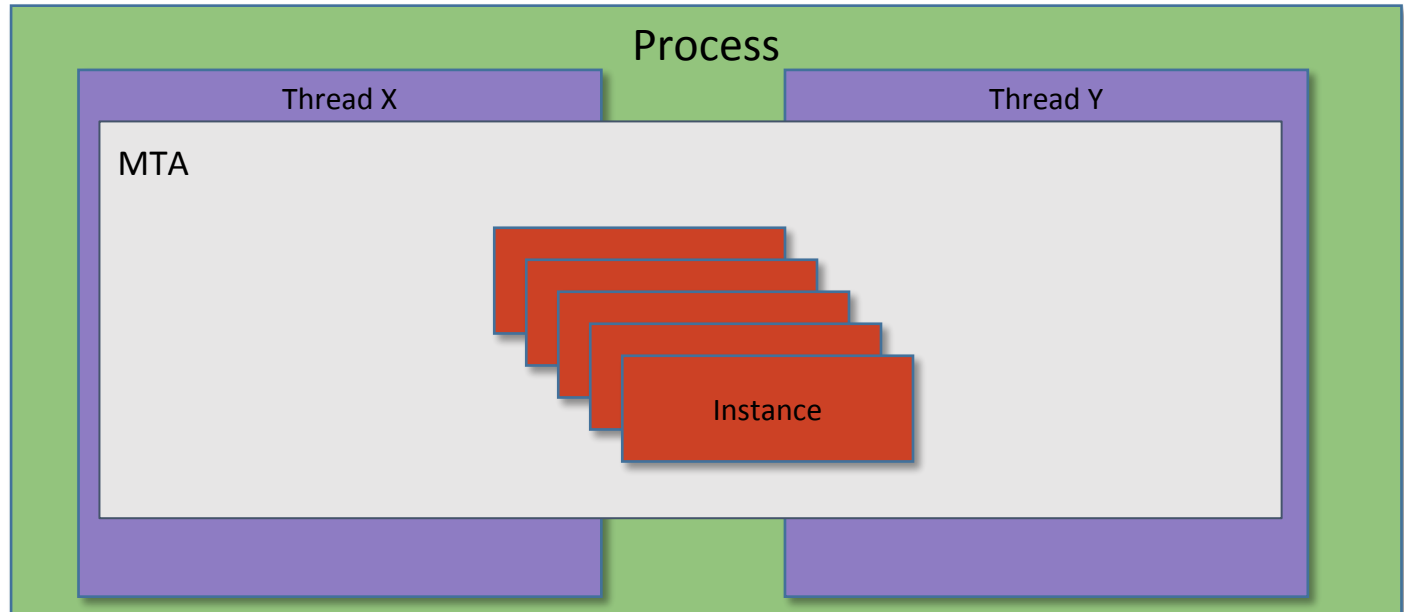
HWND	PID	TID	NAME
002906E4	21300	13416	OleMainThreadWndName
00510942	4316	8464	OleChannelWnd
005F0A4A	4316	21232	OleChannelWnd
00210D3E	4316	6028	OleChannelWnd
001A048E	5880	19768	OleChannelWnd
005C077E	5880	16680	OleChannelWnd
004E08A2	5880	12464	OleChannelWnd
000702E4	19600	20756	OleMainThreadWndName
00530832	5880	12636	OleMainThreadWndName
004E07EE	10952	18772	OleChannelWnd
00110D58	10952	19568	OleMainThreadWndName
003D0302	5404	19032	OleChannelWnd
004B0B7A	2244	2416	OleMainThreadWndName
001E09D8	19828	19832	OleMainThreadWndName

“Main” STA window.

Per-Thread STA window.

Multi-Threaded Apartments (MTA)

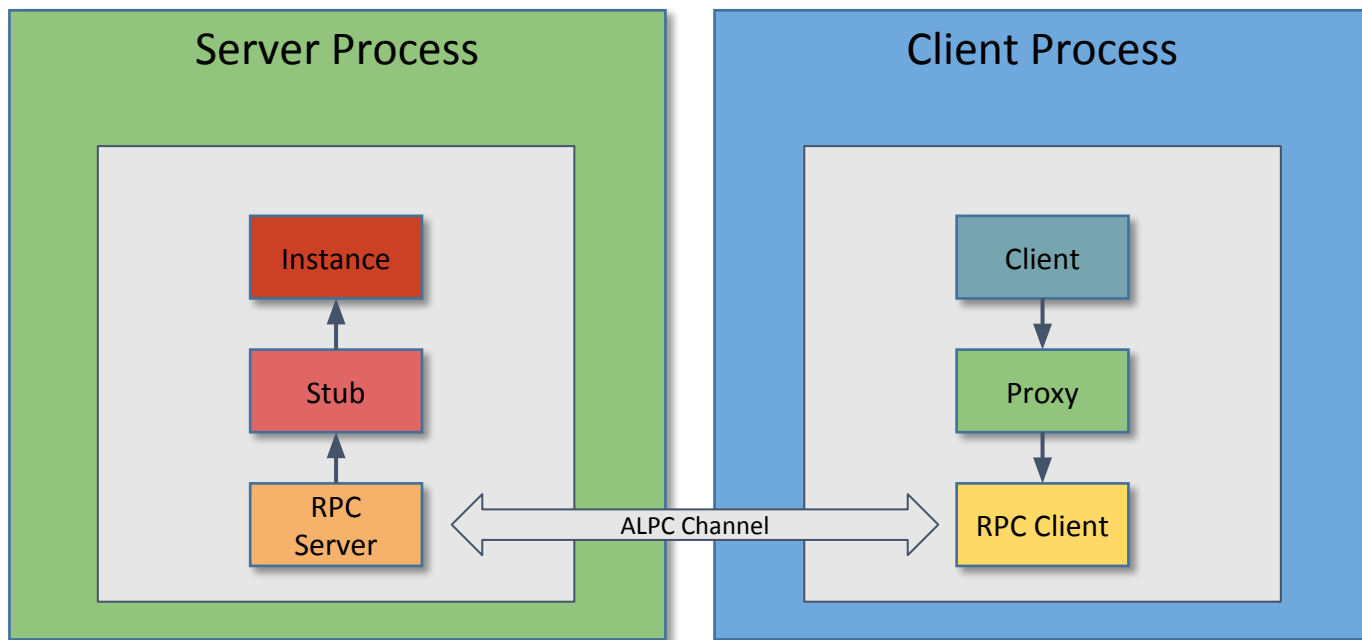
- One MTA per-process. Initialized with *COINIT_MULTITHREADED*.
- Created instance can be accessed directly in any thread in the MTA.
- STA instances need to be marshaled and vice-versa
- Also similar Neutral Apartment (NTA) which optimizes away some marshaling.



Threading Models

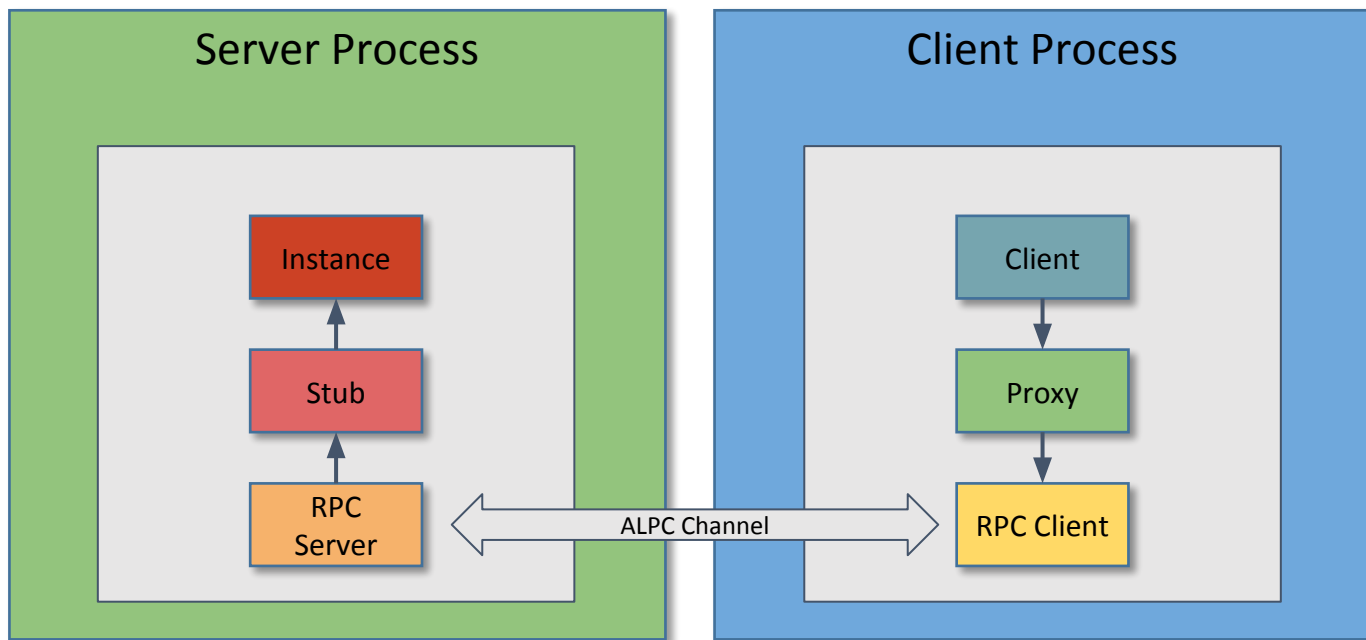
- In Process servers should be registered with a *ThreadingModel* registry value.
- Takes a string which must be one of the following values:
 - “Apartment” - Object must be created inside a STA
 - “Free” - Free-threaded, object must be created in the MTA
 - “Both” - Object can be created and used directly in either STA or MTA
 - “Neutral” - Object must be created in a NTA (COM+ Wizardry)
- If no *ThreadingModel* object can ONLY be used and created in the first STA in the process.
- If *Free* object created in a STA then MTA is created if it doesn’t exist and marshaled back.
- Likewise if *Apartment* object created in an MTA a new STA is created and object marshaled back

Local Server Activation

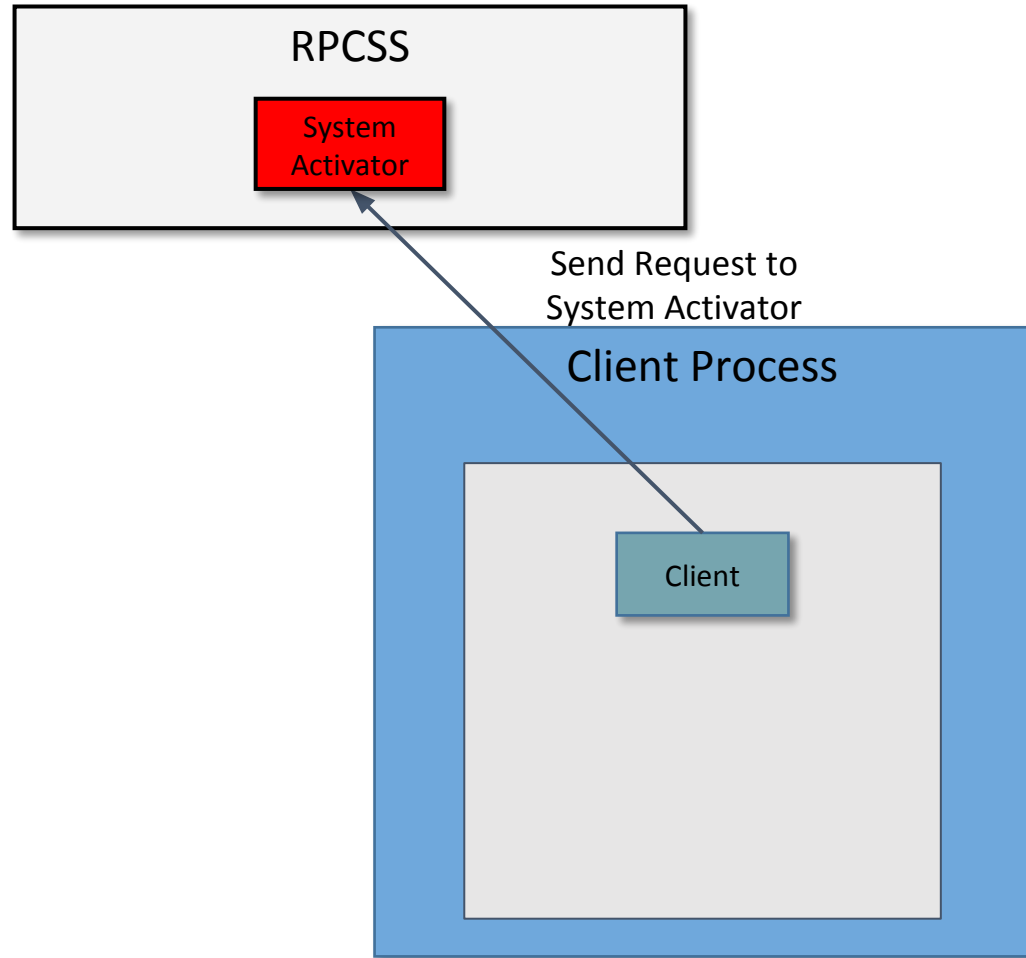


Local Server Activation

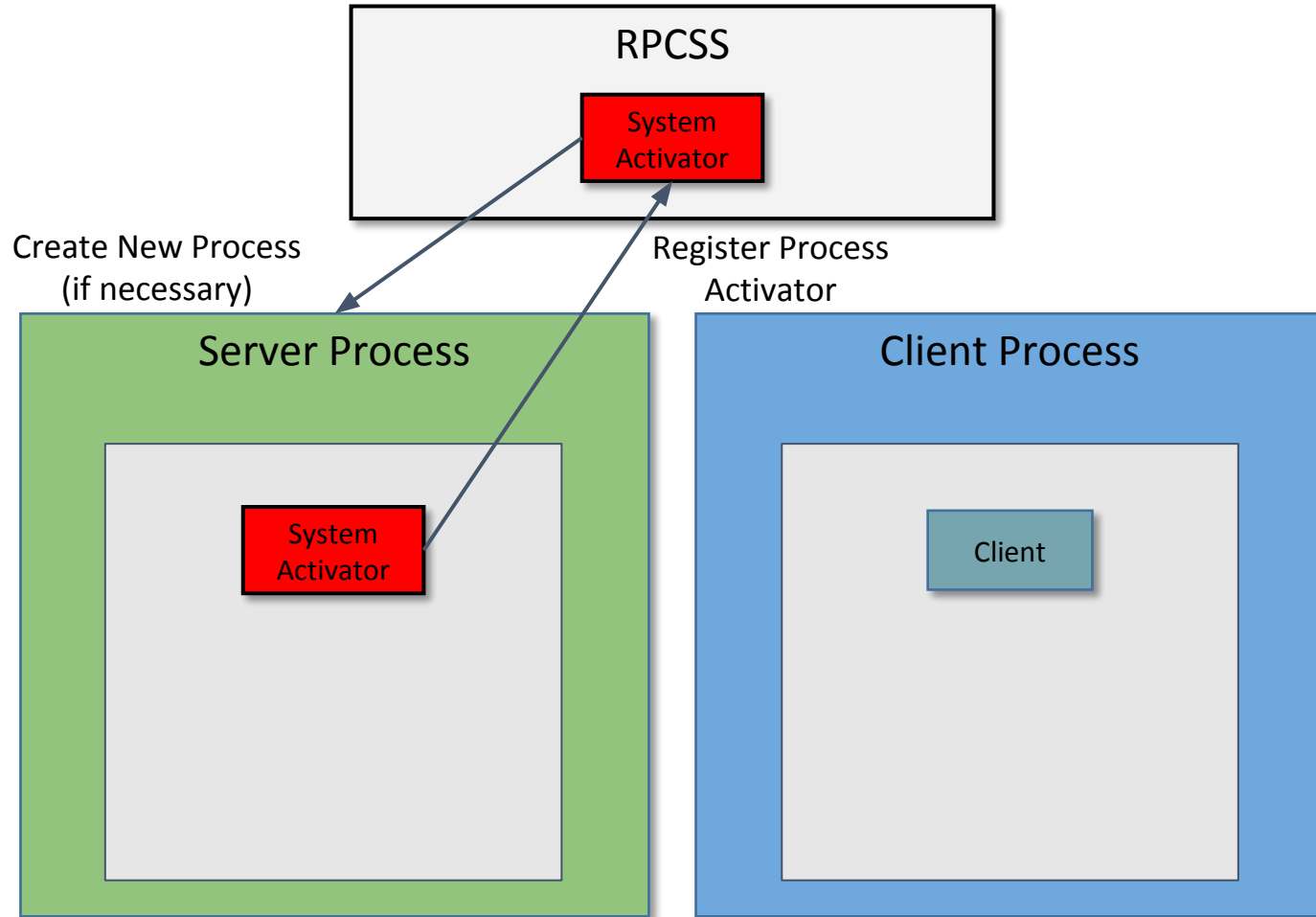
DCOM Activator
(part of RPCSS)



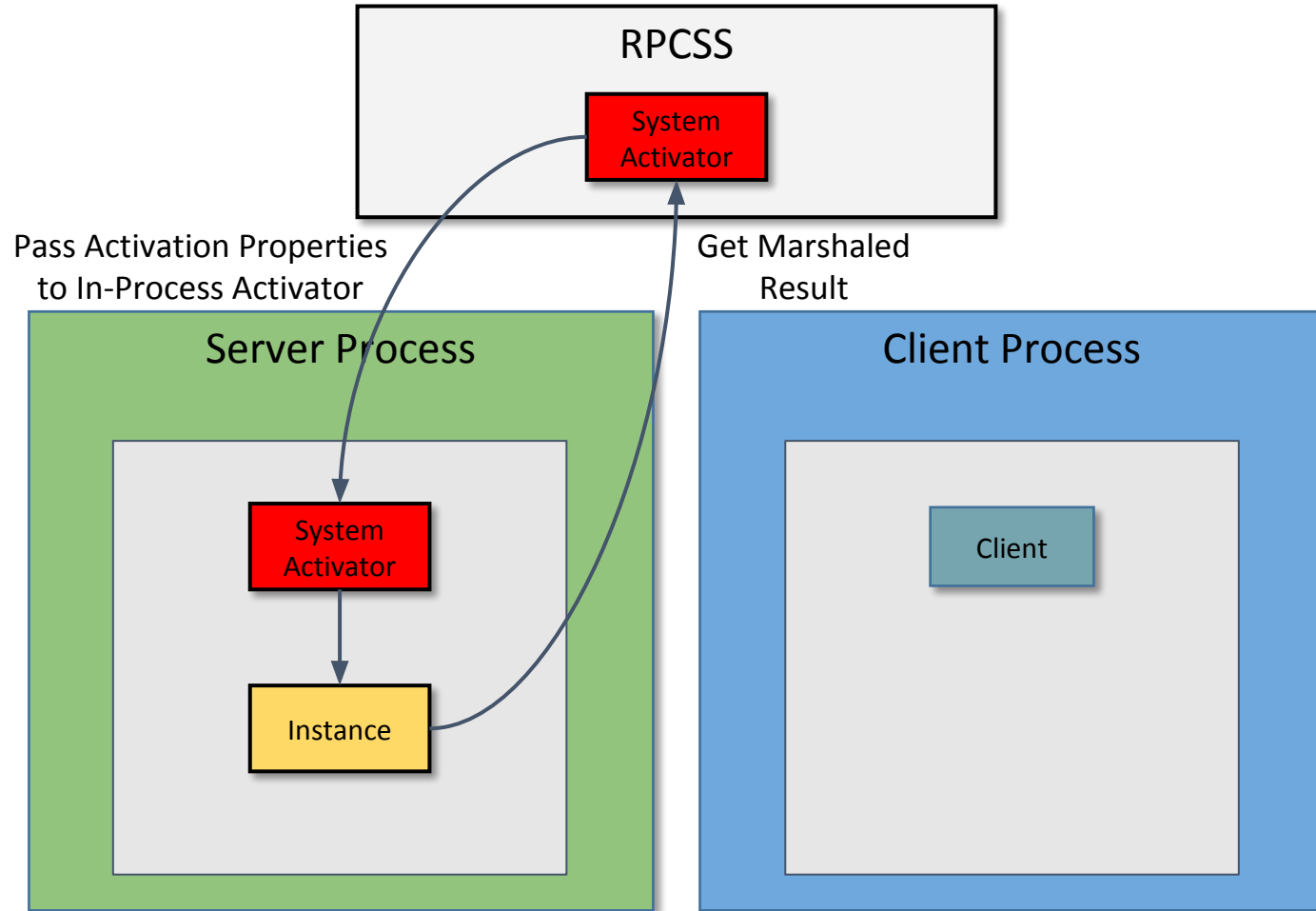
Local Server Activation



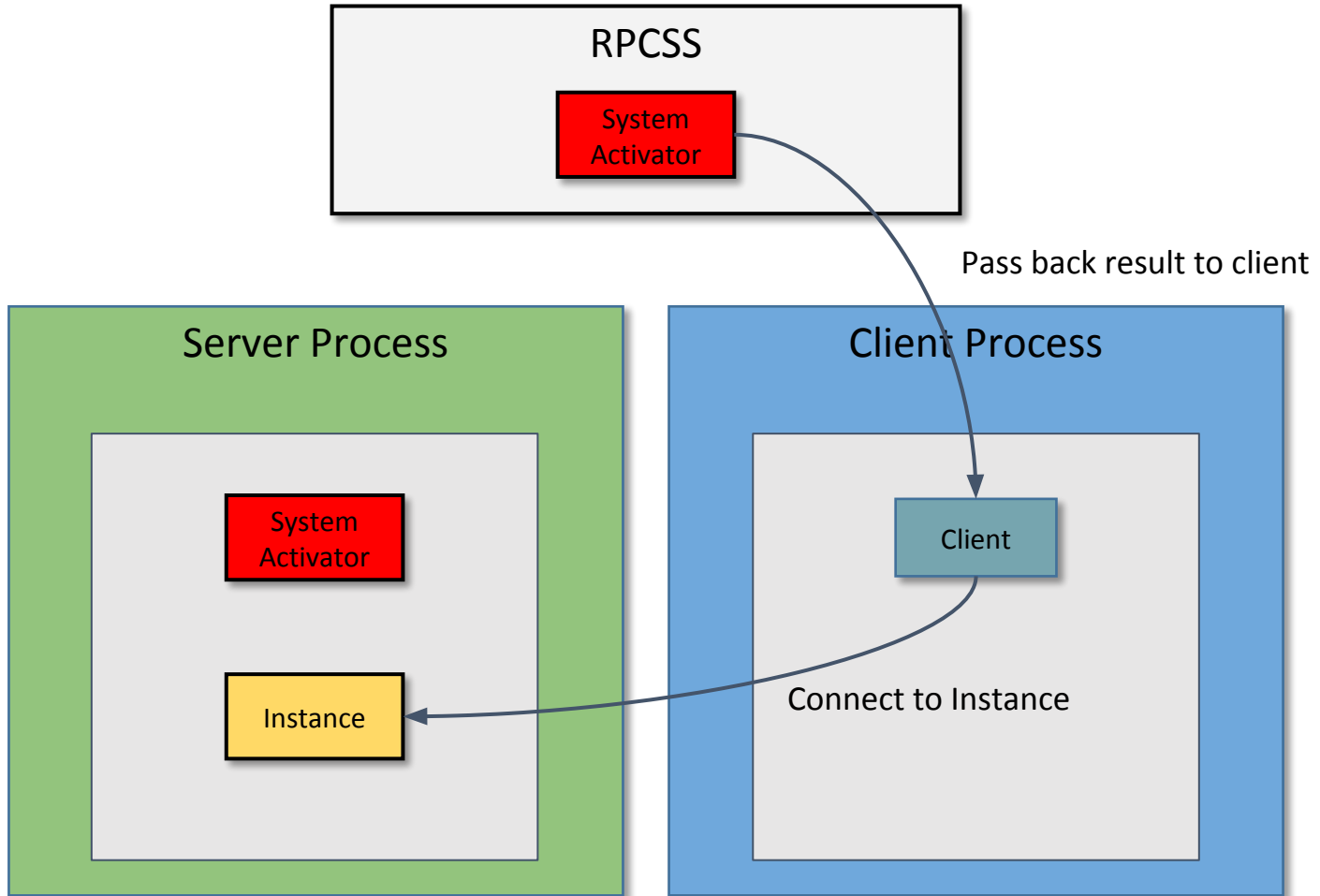
Local Server Activation



Local Server Activation



Local Server Activation



System Activator

```
DEFINE_GUID(IID_ISystemActivator,  
            "000001a0-0000-0000-c000-000000000046")  
struct ISystemActivator : public IUnknown {  
    HRESULT GetClassObject(  
        IActivationPropertiesIn *pActProperties,  
        IActivationPropertiesOut **ppActProperties);  
  
    HRESULT CreateInstance(  
        IUnknown *pUnkOuter,  
        IActivationPropertiesIn *pActProperties,  
        IActivationPropertiesOut **ppActProperties);  
};
```

Activation Properties In



```
struct CustomHeader {  
    DWORD totalSize;  
    DWORD headerSize;  
    DWORD dwReserved;  
    DWORD destCtx;  
    DWORD cIfs;  
    CLSID classInfoClsid;  
    CLSID *pclsid;  
    DWORD *pSizes;  
    CustomOpaqueData *opaqueData;  
};
```

List of GUIDs and Sizes of following Property Blobs

```
struct InstantiationInfoData {  
    CLSID classId;  
    DWORD classCtx;  
    DWORD actvflags;  
    long fIsSurrogate;  
    DWORD ciID;  
    DWORD instFlag;  
    IID *piID;  
    DWORD thisSize;  
    COMVERSION clientCOMVersion;  
};
```

CLSID of class to create.

```
enum ACTIVATION_FLAGS {  
    ACTVFLAGS_DISABLE_AAA,  
    ACTVFLAGS_ACTIVATE_32_BIT_SERVER,  
    ACTVFLAGS_ACTIVATE_64_BIT_SERVER,  
    ACTVFLAGS_NO_FAILURE_LOG,  
    ACTVFLAGS_WINRT_LOCAL_SERVER,  
    ACTVFLAGS_WINRT_PER_USER_OK,  
    ACTVFLAGS_APPCONTAINER,  
};
```

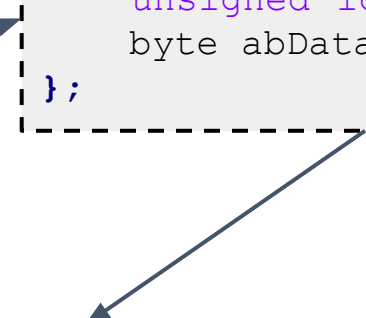
List of IIDs to query for.

Activation Properties Out

- Same structure as ActivationPropertiesIn just with different property sets.

```
struct PropsOutInfo {  
    DWORD cIifs;  
    IID *piid;  
    HRESULT *phresults;  
    MInterfacePointer **ppIntfData;  
};
```

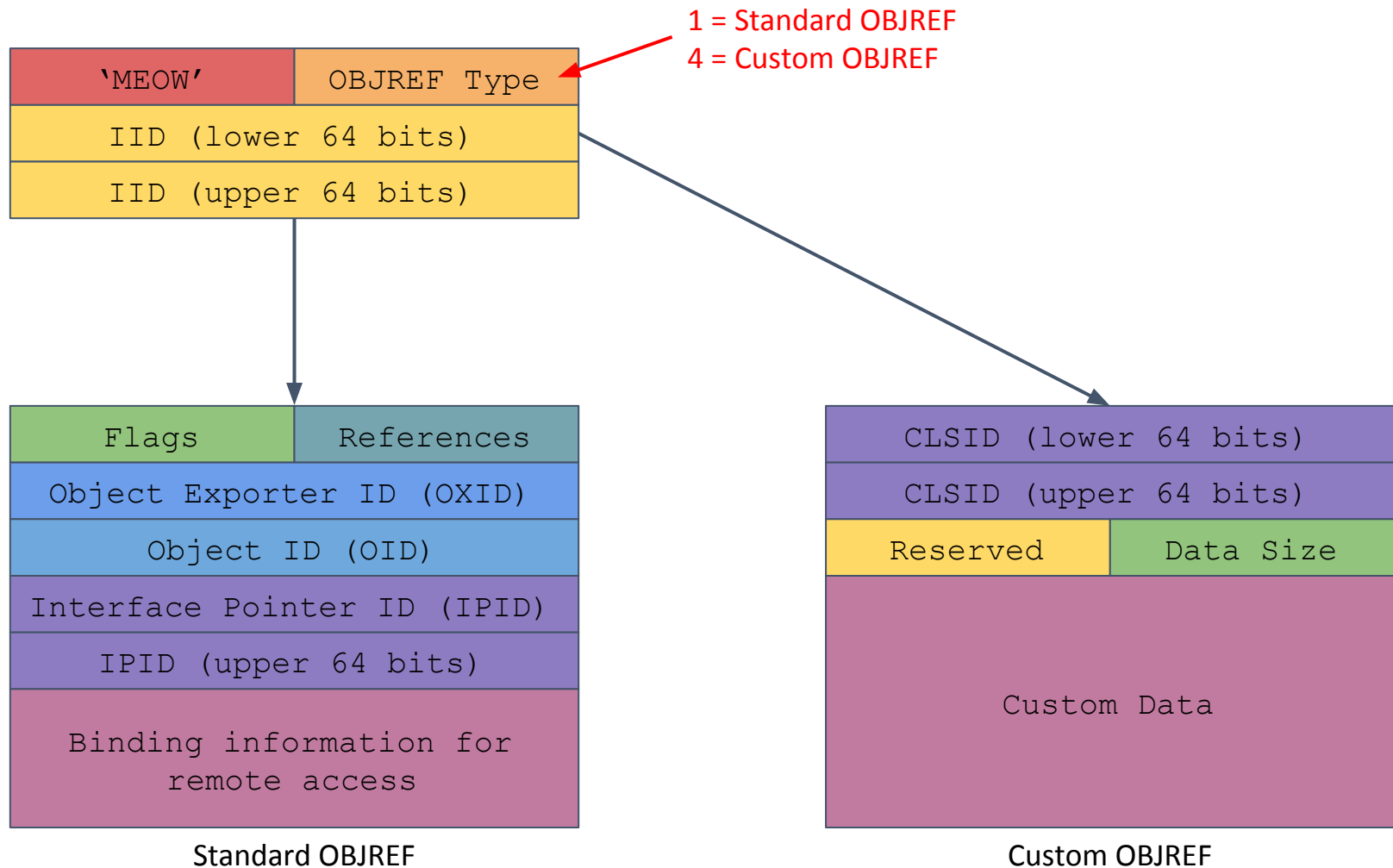
```
struct MInterfacePointer {  
    unsigned long ulCntData;  
    byte abData [];  
};
```



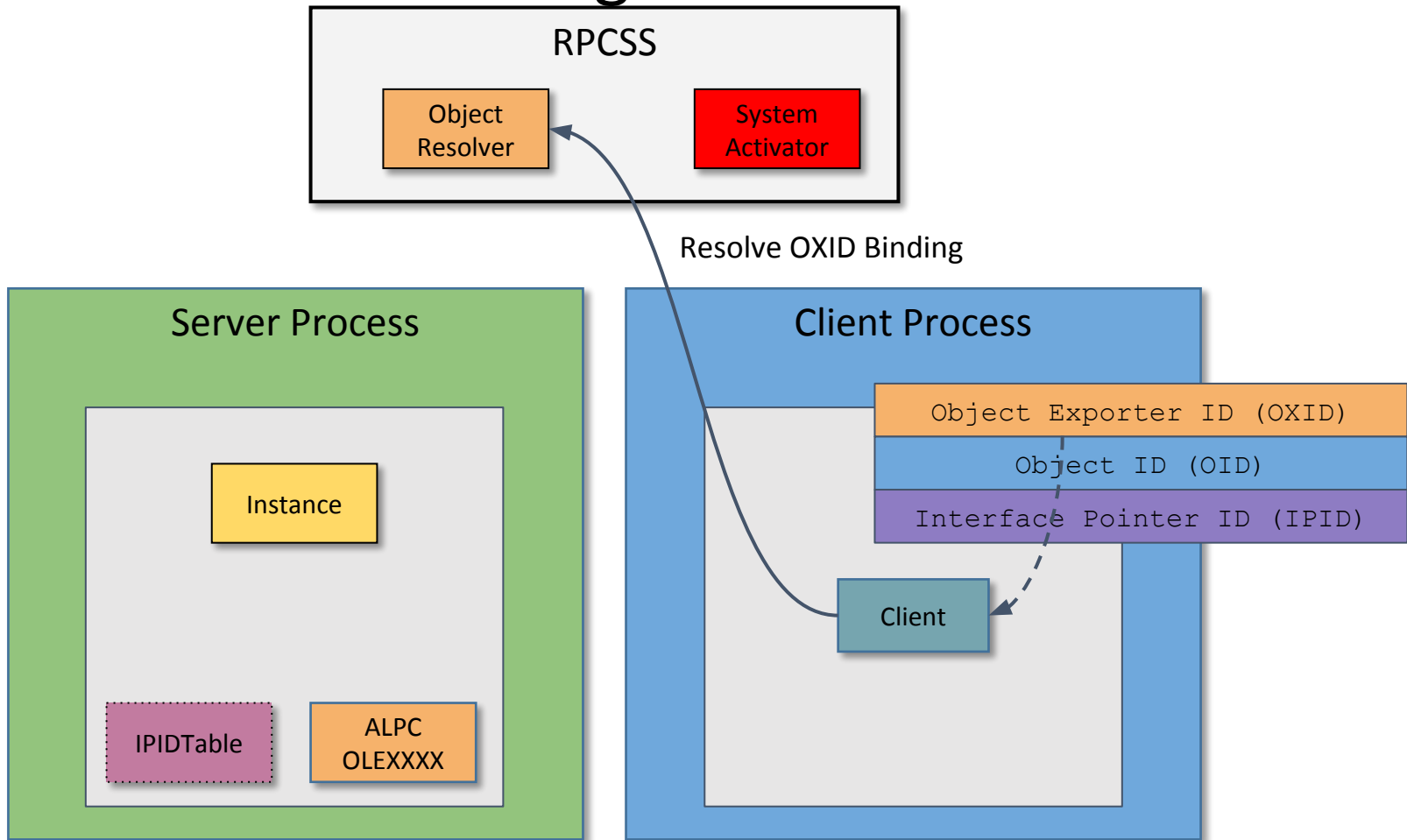
Memory dump showing hexadecimal and ASCII values:

00000000	4D 45 4F 57 01 00 00 00 A0 01 00 00 00 00 00 00	MEOW.... ..
00000010	C0 00 00 00 00 00 00 00 46 00 00 00 00 01 00 00	À.....F.....
00000020	19 14 A0 F8 BF 3D 72 6A D5 41 68 BD 84 C3 08 E7	..ø¿=rjÕAh%.Ã.ç
00000030	03 5C 00 00 38 16 E4 46 37 53 80 9C 87 A9 08 45	.\..8.äF7S...@.E
00000040	00 00 00 00	

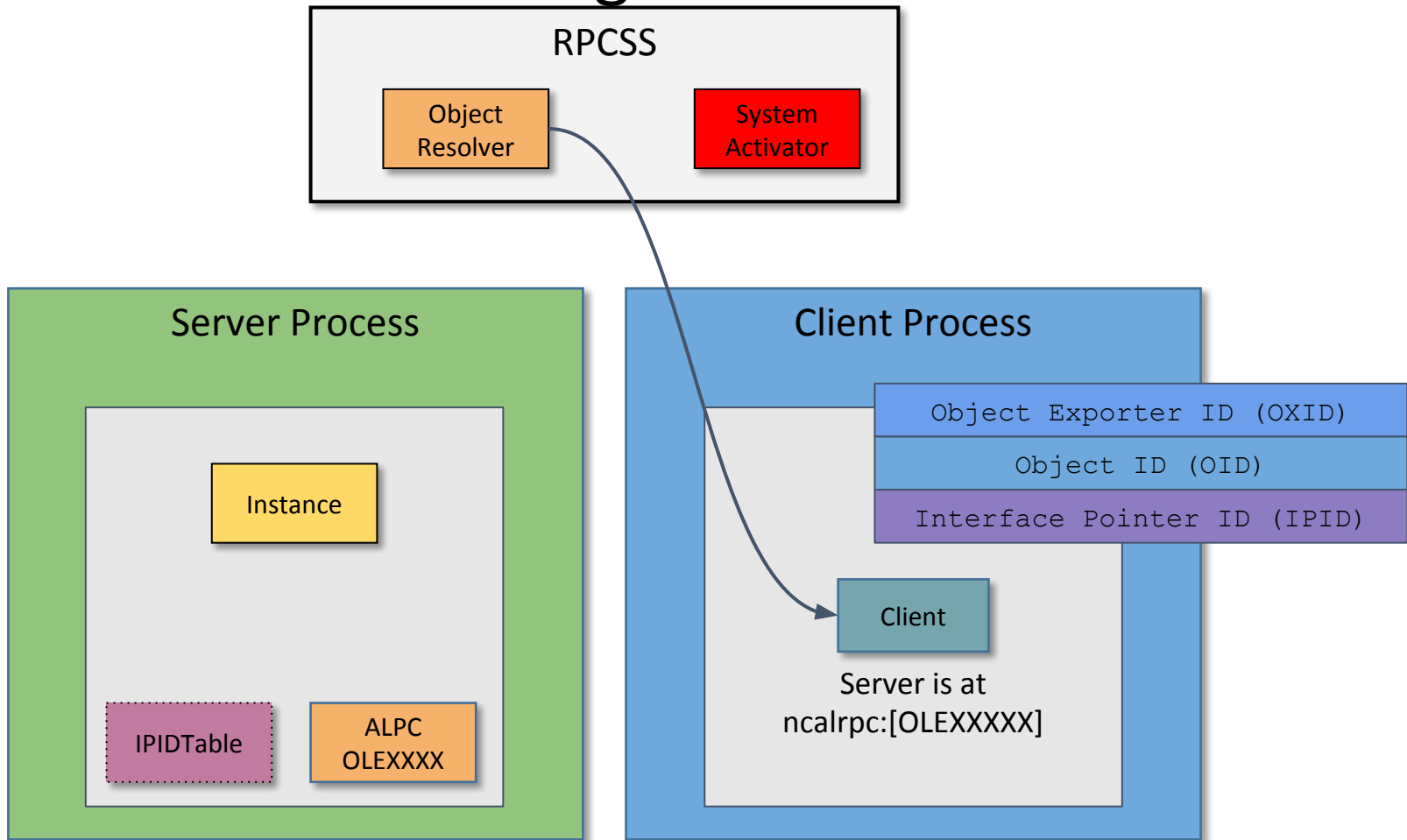
It's like Marshaling Cats



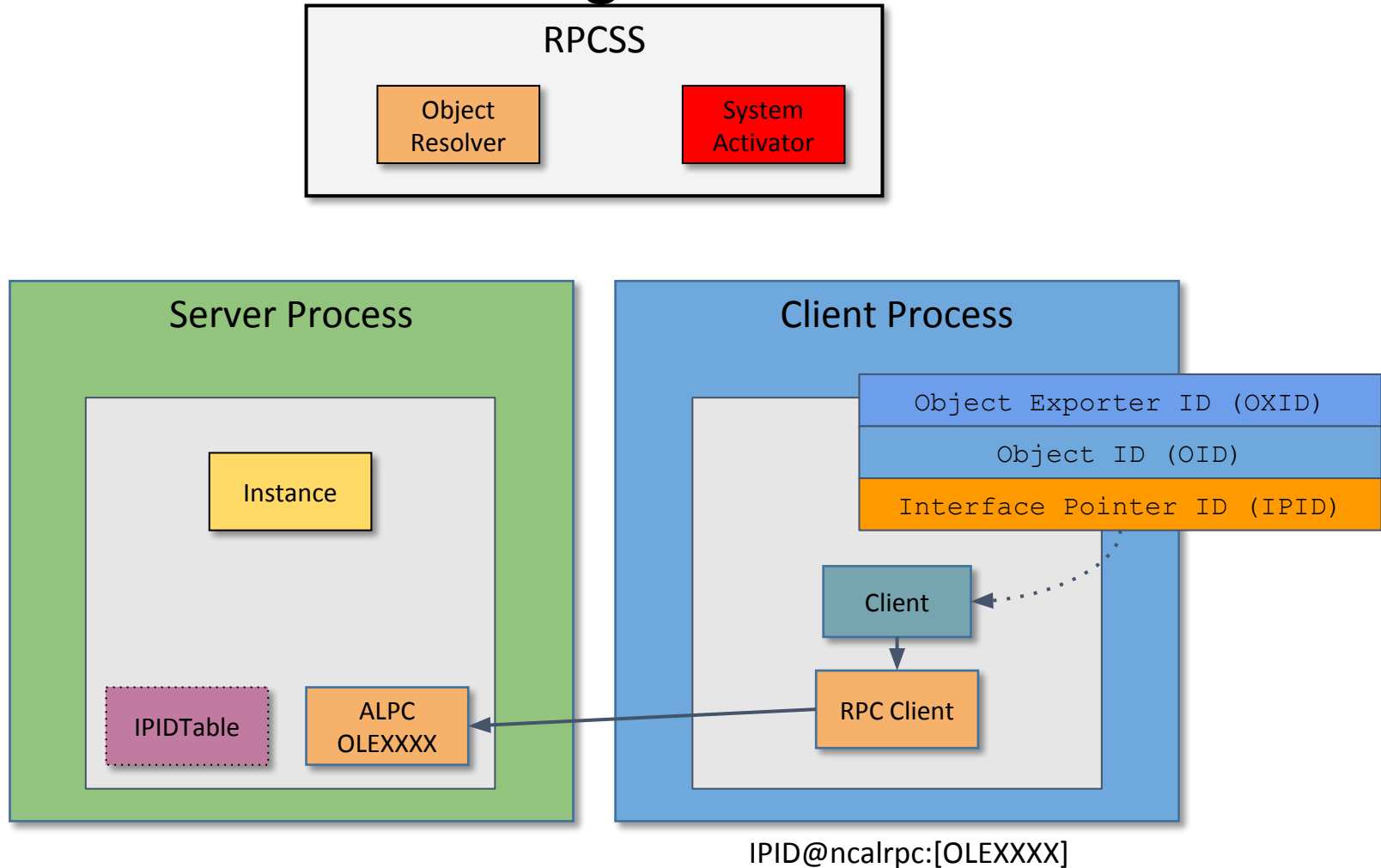
Standard Unmarshaling



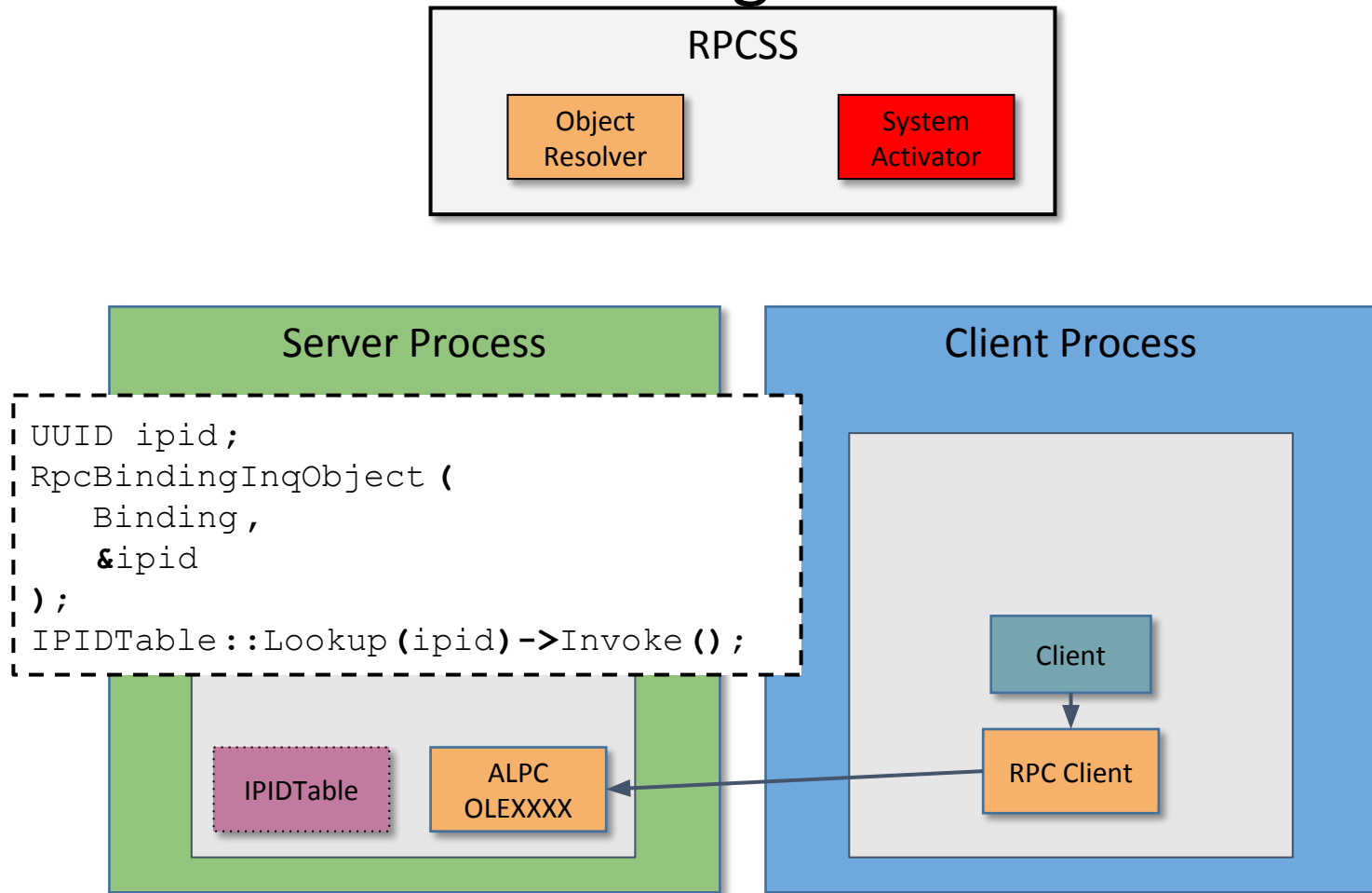
Standard Unmarshaling



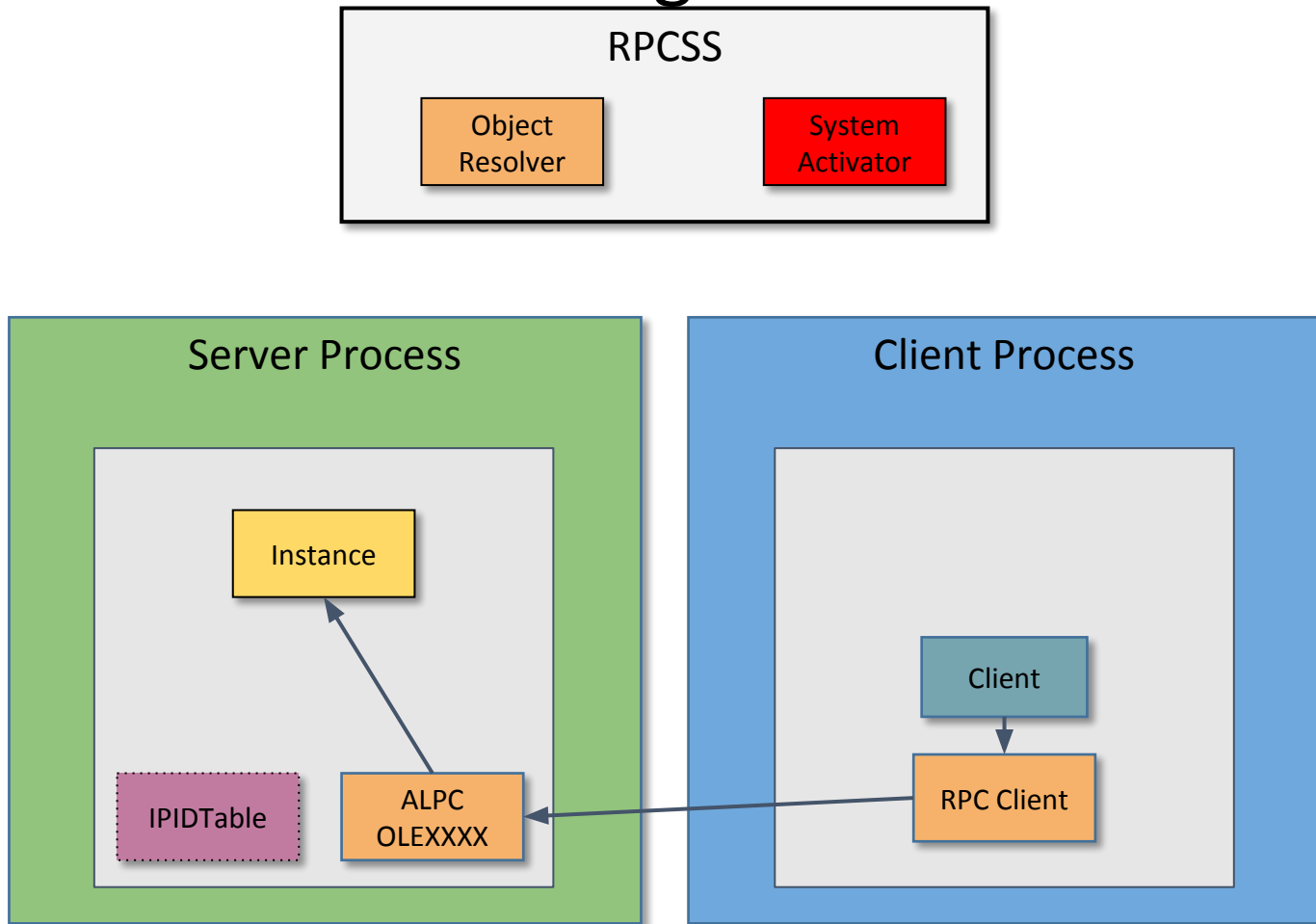
Standard Unmarshaling



Standard Unmarshaling



Standard Unmarshaling



Interface Proxies and Stubs

The image shows two windows from a Windows operating system. The top window is the Registry Editor, displaying the path `Computer\HKEY_CLASSES_ROOT\Interface\{475CA8F3-9417-48B8-B9D7-4163A7844C02}`. The bottom window is OleViewDotNet, showing the details for the CLSID `e1ba88ba-7a83-421a-a05d-71f96c3fffd0`. The OleViewDotNet window has tabs for 'CLSID', 'Supported Interfaces', and 'Proxies'. The 'Proxies' tab is active, displaying a list of interfaces and their IIDs. A red arrow points from the 'ProxyStubClsid32' value in the Registry Editor to the 'Proxies' tab in OleViewDotNet.

Registry Editor Data:

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

OleViewDotNet Data (Proxies Tab):

Name	IID
IMagneticStripeReaderBankCardDataReceivedEventArgs	2E958823-A31A-4763-882C-23725E39B08E
IPosPrinterStatusUpdatedEventArgs	2EDB87DF-13A6-428D-BA81-B0E7C3E5A3CD
ICashDrawerStatusUpdatedEventArgs	30AAE98A-0D70-459C-9553-87E124C52488
ITypedEventHandler_2_Windows_CDevices_CPointOfService_CCclaimedPosP...	31424F6F-CFEB-5031-8A95-BEA59B09E584
IAsyncOperationCompletedHandler_1_Windows_CDevices_CPointOfService_...	32C55F7B-8EE3-555D-998B-78C98AA9627B
IBarcodeScannerStatusUpdatedEventArgs	355D8586-9C43-462B-A91A-816DC97F452C
IJournalPrinterCapabilities	3B5CCC43-E047-4463-BB58-17B5BA1D8056
IAsyncOperation_1_Windows_CDevices_CPointOfService_CCclaimedMagnetic...	41630BD4-F45A-590D-8A4E-F70C9E49AD01
IBarcodeScannerDataReceivedEventArgs	4234A7E2-ED97-467D-AD2B-01E44313A929
IAsyncOperation_1_Windows_CDevices_CPointOfService_CCashDrawer	45007467-92F2-5BFF-B191-AA5000FEDD9E
IClaimedMagneticStripeReader	475CA8F3-9417-48B8-B9D7-4163A7844C02
IClaimedBarcodeScanner	4A63B49C-8FA4-4332-BB26-945D11D81E0F
ITypedEventHandler_2_Windows_CDevices_CPointOfService_CCclaimedBarc...	4F64E49A-BD8C-549D-970C-A5A250BD27CA
IMagneticStripeReaderCardTypesStatics	528F2C5D-2986-474F-8454-7CCD05928D5F
IAsyncOperation_1_FIVectorView_1_UINT32	52C56F3C-713A-5162-9E62-362CE7ED53BE
IReceiptOrSlipJob	532199BE-C8C3-4DC2-89E9-5C4A37B34DDC
IMagneticStripeReaderEncryptionAlgorithmsStatics	53B57350-C3DB-4754-9C00-41392374A109
IAsyncOperationCompletedHandler_1_FIVectorView_1_UINT32	55772F29-DA64-5C87-871C-074337A84573
IAsyncOperationCompletedHandler_1_Windows_CDevices_CPointOfService_...	57836710-F186-5636-891D-F8C5398EA6DF
IPosPrinterCharacterSetIdsStatics	5C709EFF-709A-4FE7-B215-06A748A38B39
IBarcodeScannerReport	5CE4D8B0-A489-4B96-86C4-F0BF8A37753D
IBarcodeScannerStatics	5D11EECE-D4A8-A1E9-8C9C-8B8E73A9AFC

Proxy and Stub contain Network Data Representation (NDR) bytecode to marshal raw parameters.

Proxy-Stub Factory

- Proxy-Stub CLSID doesn't use IClassFactory. Instead uses IPSFactoryBuffer.

```
DEFINE_GUID(IID_IPSFactoryBuffer ,  
            "D5F569D0-593B-101A-B569-08002B2DBF7A" )  
struct IPSFactoryBuffer : public IUnknown  
{  
    HRESULT CreateProxy (  
        IUnknown *pUnkOuter ,  
        REFIID riid ,  
        IRpcProxyBuffer **ppProxy ,  
        void **ppv);  
  
    HRESULT CreateStub (  
        REFIID riid ,  
        IUnknown *pUnkServer ,  
        IRpcStubBuffer **ppStub);  
};
```

IRemUnknown

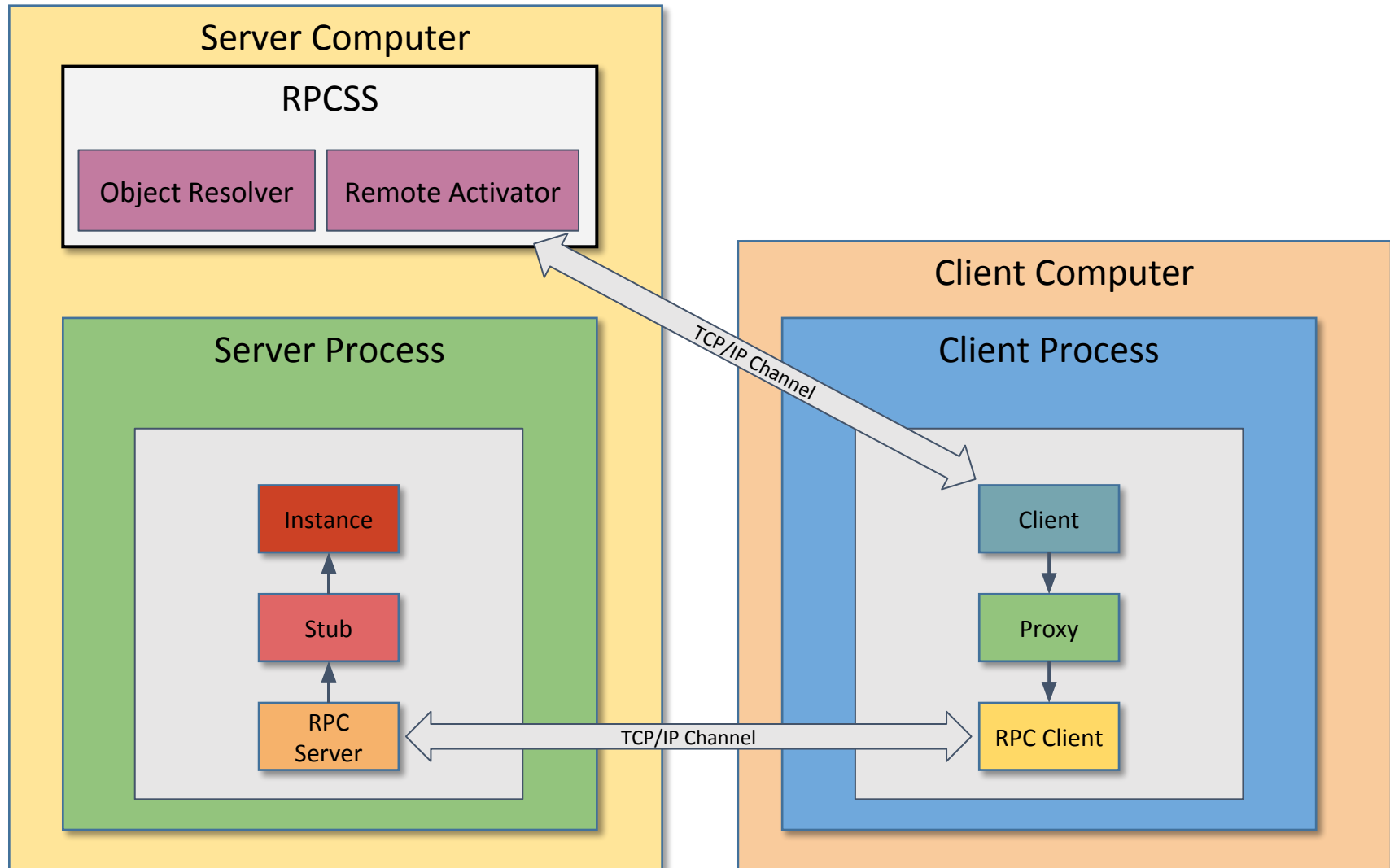
- Proxies don't directly forward QueryInterface/AddRef/Release to the remote object, these calls are used on the proxy object.
- Actual IUnknown calls done on special interface, IRemUnknown

```
struct customREMOTE_REPLY_SCM_INFO {  
    OXID                Oxid;  
    DUALSTRINGARRAY    *pdsaOxidBindings;  
    IPID                ipidRemUnknown;  
    DWORD               authnHint;  
    COMVERSION          serverVersion;  
};
```

Returned in ActivationPropertiesOut

```
DEFINE_GUID(IID_IRemUnknown,  
            "00000131-0000-0000-C000-000000000046")  
struct IRemUnknown : public IUnknown {  
    HRESULT RemQueryInterface(  
        REFIID riid,  
        unsigned long cRefs,  
        unsigned short cIids,  
        IID *iids,  
        PREMQIRERESULT *ppQIREsults);  
  
    HRESULT RemAddRef(  
        unsigned short cInterfaceRefs,  
        REMINTERFACEREF InterfaceRefs[],  
        HRESULT *pResults);  
  
    HRESULT RemRelease(  
        unsigned short cInterfaceRefs,  
        REMINTERFACEREF InterfaceRefs[]);  
};
```

Remote Server



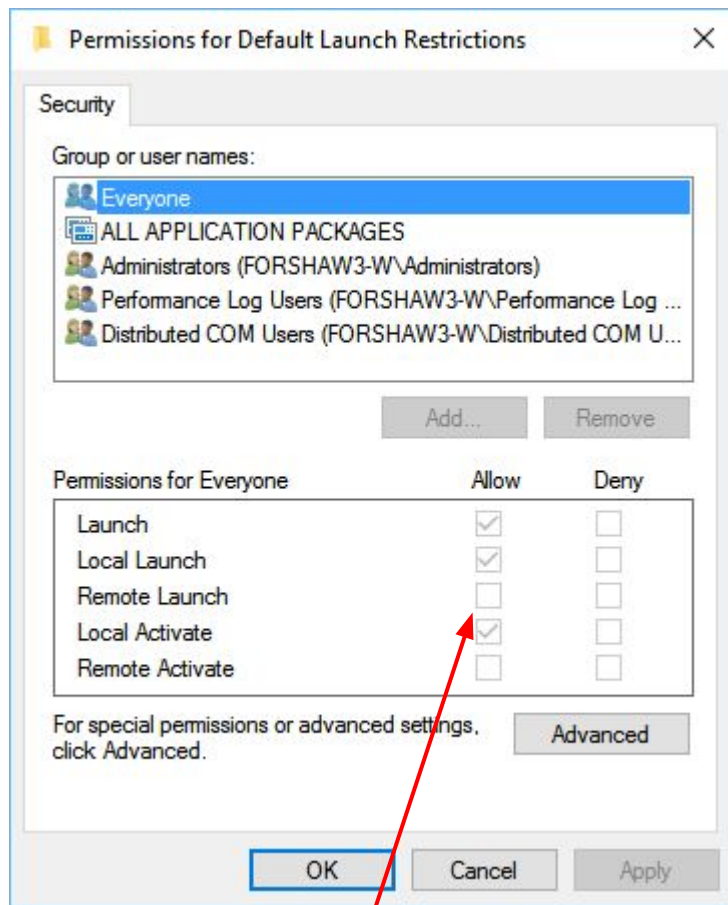
COSERVERINFO

```
struct COSERVERINFO {  
    DWORD dwReserved1;  
    LPWSTR pwszName;  
    COAUTHINFO *pAuthInfo;  
    DWORD dwReserved2;  
};
```

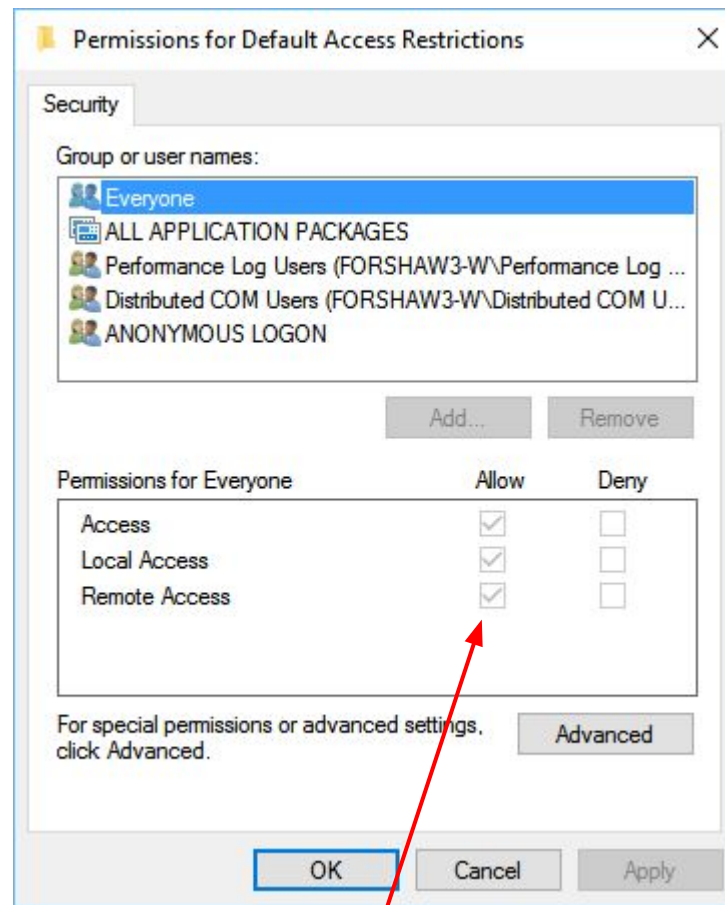
Can specify better
authentication level when
connecting such as CALL or
even better PKT_PRIVACY

```
struct COAUTHINFO {  
    DWORD dwAuthnSvc;  
    DWORD dwAuthzSvc;  
    LPWSTR pwszServerPrincName;  
    DWORD dwAuthnLevel;  
    DWORD dwImpersonationLevel;  
    void* pAuthIdentityData;  
    DWORD dwCapabilities;  
};
```


COM Security Restrictions

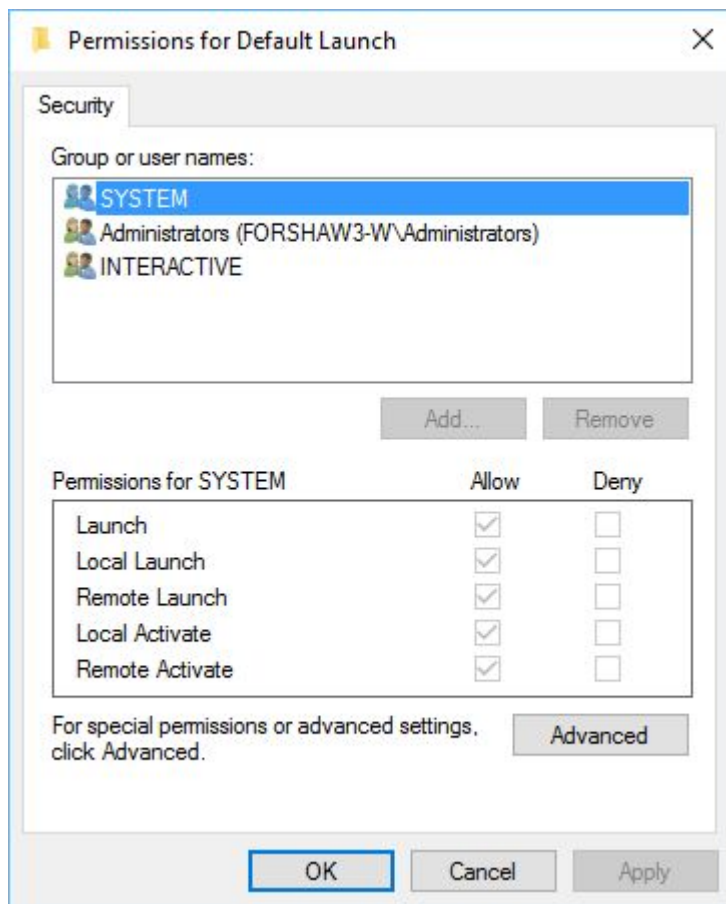


'Everyone' not allowed to launch or activate a new object remotely

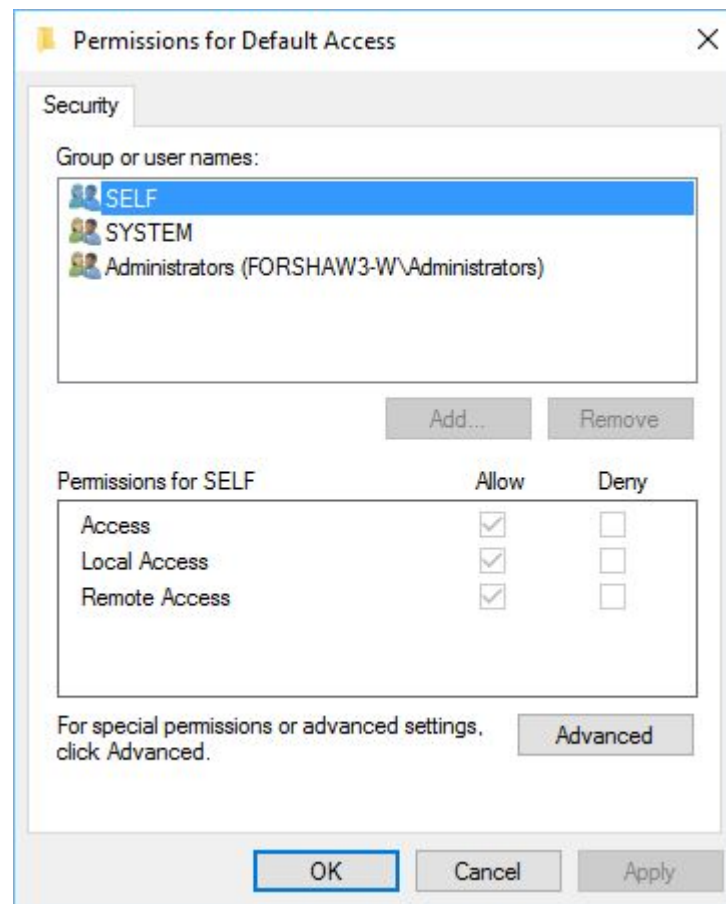


'Everyone' can access an existing object remotely

COM Security



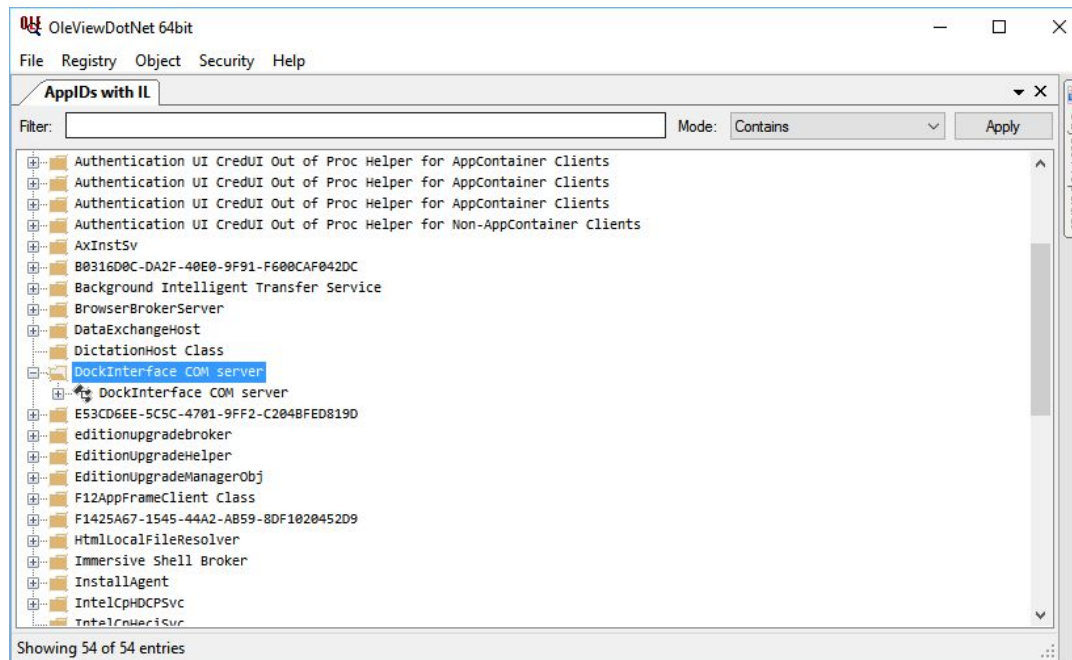
Launch = Create a new instance of the server.
Activate = Create new object on existing server.
Enforced in RPCSS



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

Integrity Levels

- Launching/Activating an OOP COM object from a Low IL needs an explicit entry in the SD for that IL.
- Accessing an object however DOES NOT need an IL label in the Access SD.



Security Through CoInitializeSecurity

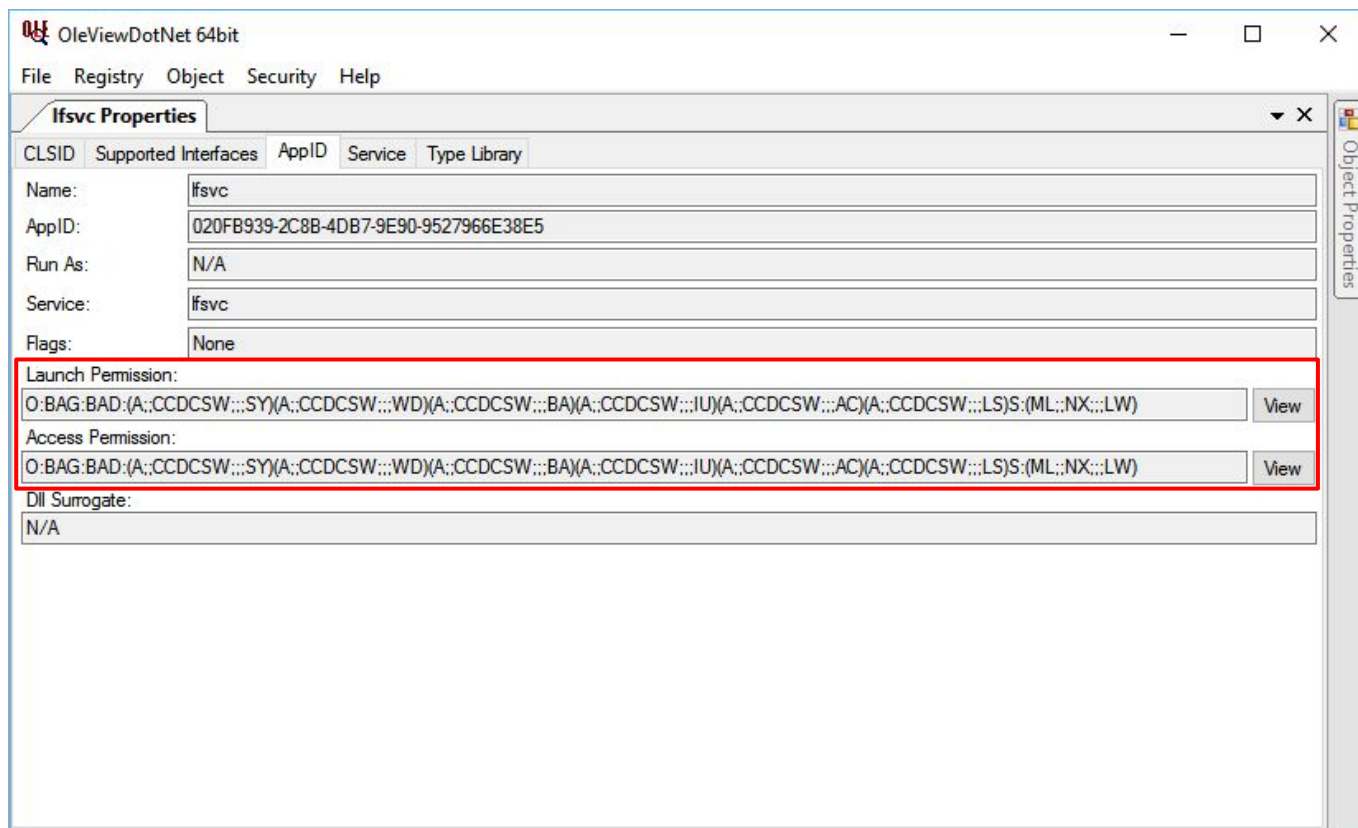
```
HRESULT CoInitializeSecurity(  
    PSECURITY_DESCRIPTOR pSecDesc,  
    LONG cAuthSvc,  
    SOLE_AUTHENTICATION_SERVICE * asAuthSvc,  
    void * pReserved1,  
    DWORD dwAuthnLevel,  
    DWORD dwImpLevel,  
    void * pAuthList,  
    DWORD dwCapabilities,  
    void * pReserved3  
);
```

Optional SD:
NULL = No Access Security!

EOAC_APPID - pSecDesc is a AppID GUID
EOAC_ACCESS_CONTROL - pSecDesc is a
pointer to an IAccessControl
implementation
EOAC_NO_CUSTOM_MARSHAL - Disables
custom marshaling in the process.

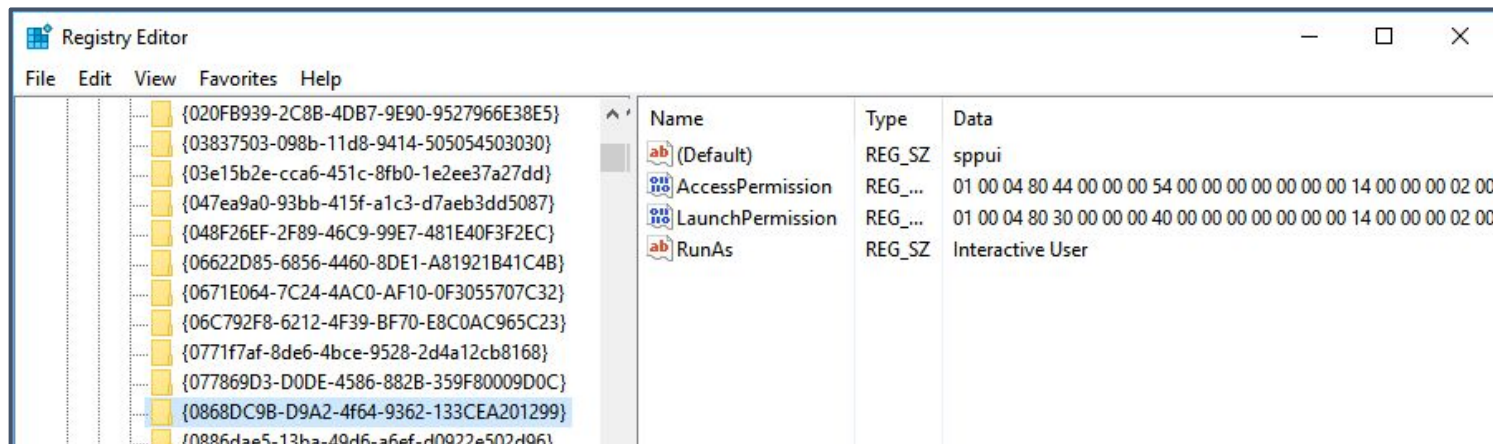
Security Through AppID

- Launch/Activation and Access Security can be specified through the AppID registration for the Class

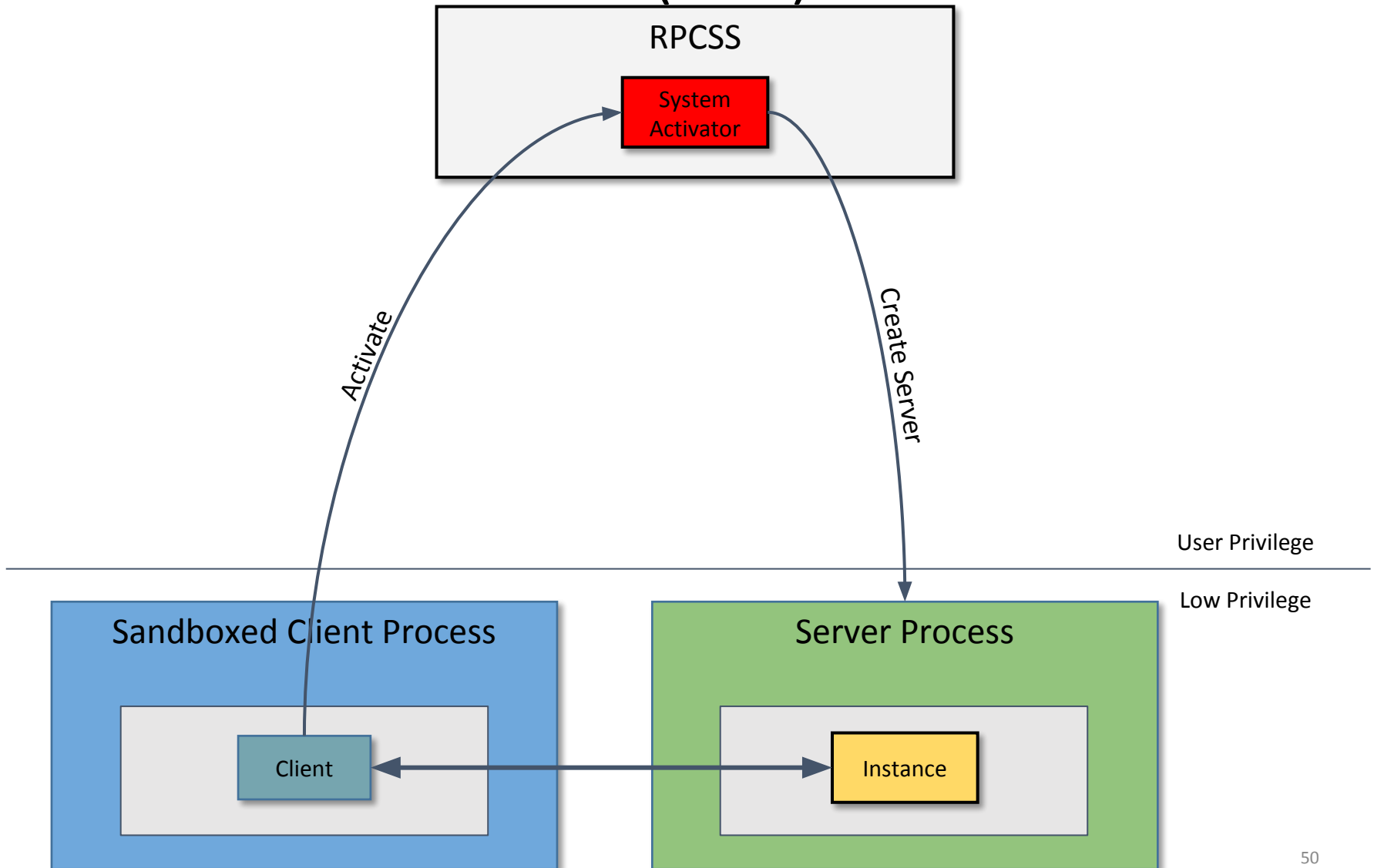


Application IDs

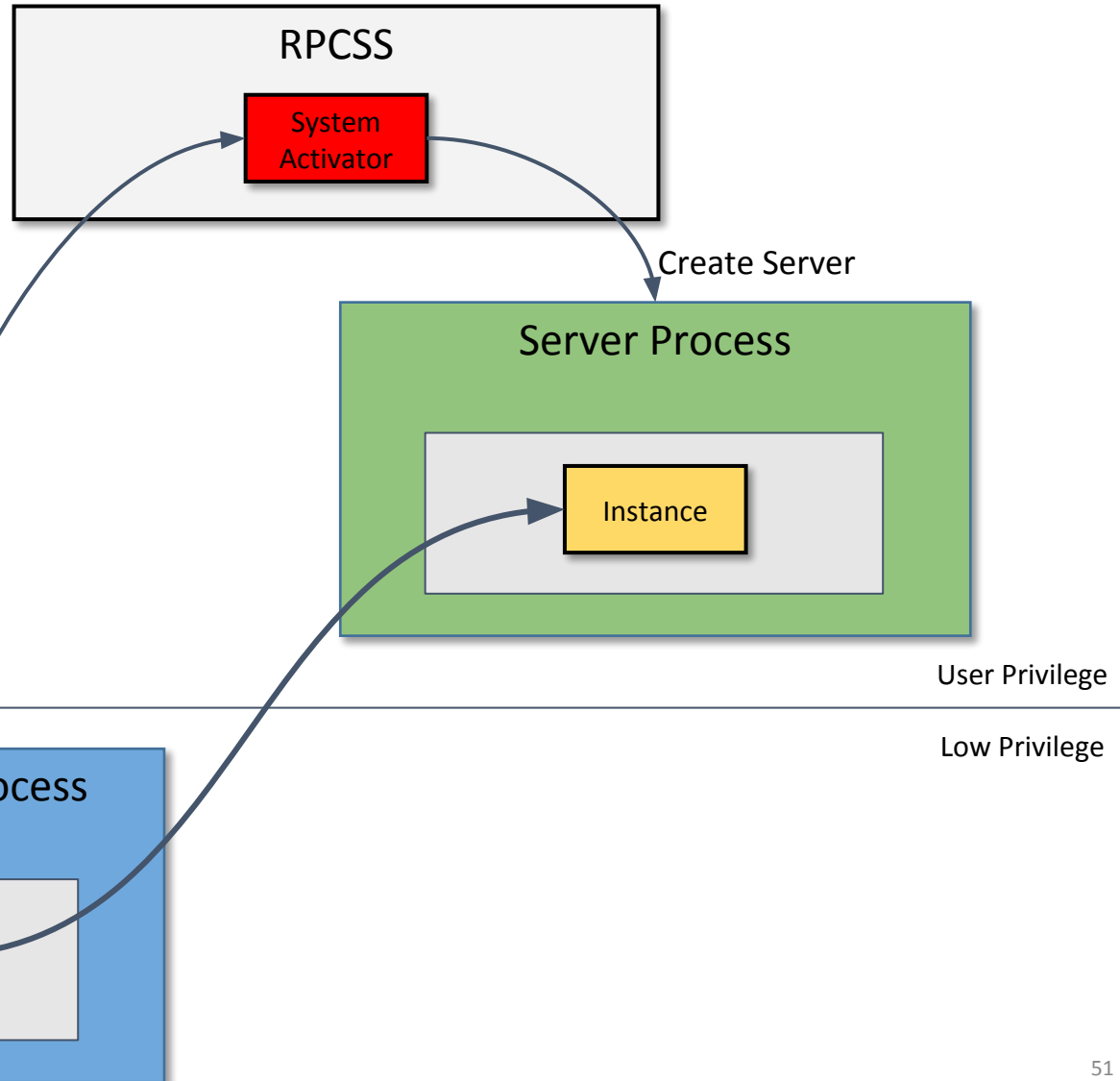
- AppIDs can configure more than just Launch/Access security
- Used to specify a number of features:
 - Allows you to specify a DLL Surrogate. This allows you to create in-process DLL servers as OOP local servers
 - Specify the object is hosted in a Windows service rather than a separate executable
 - Specify running as interactive user.



Activate as Activator (AAA)



RunAs Interactive User

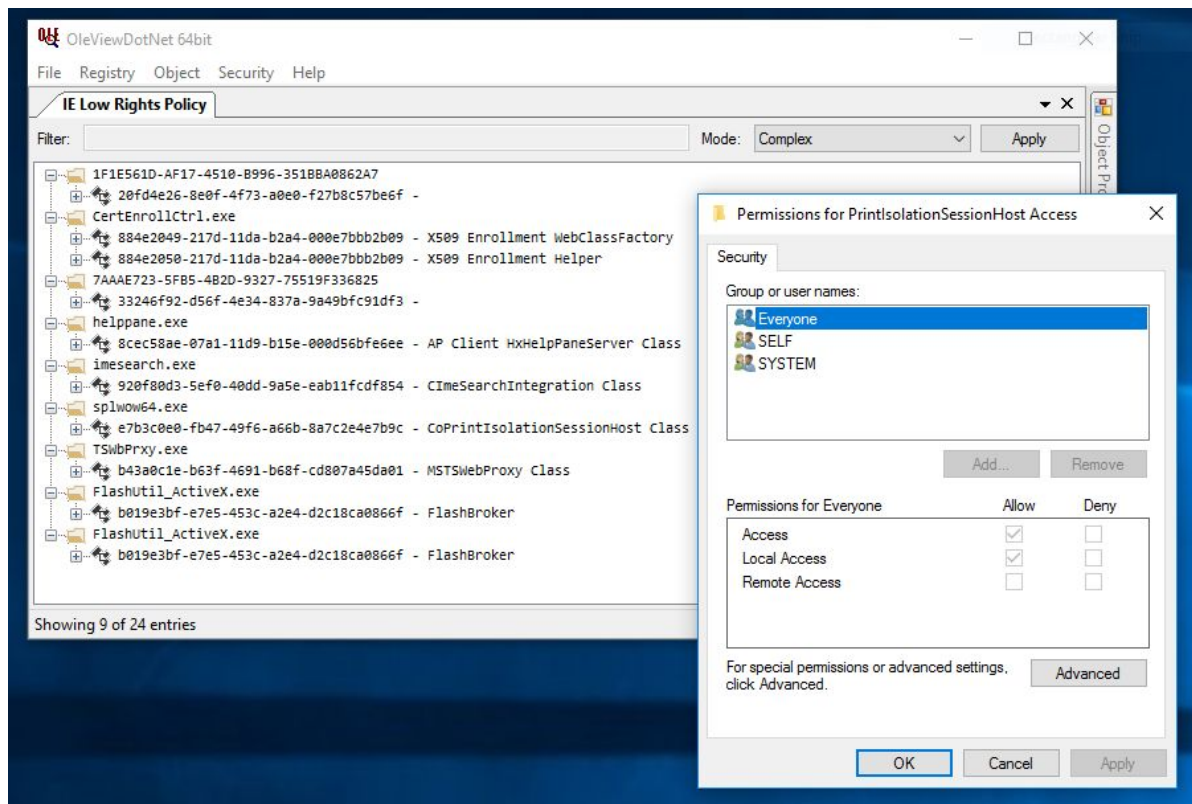




Enumerating Attack Surface and Reverse Engineering

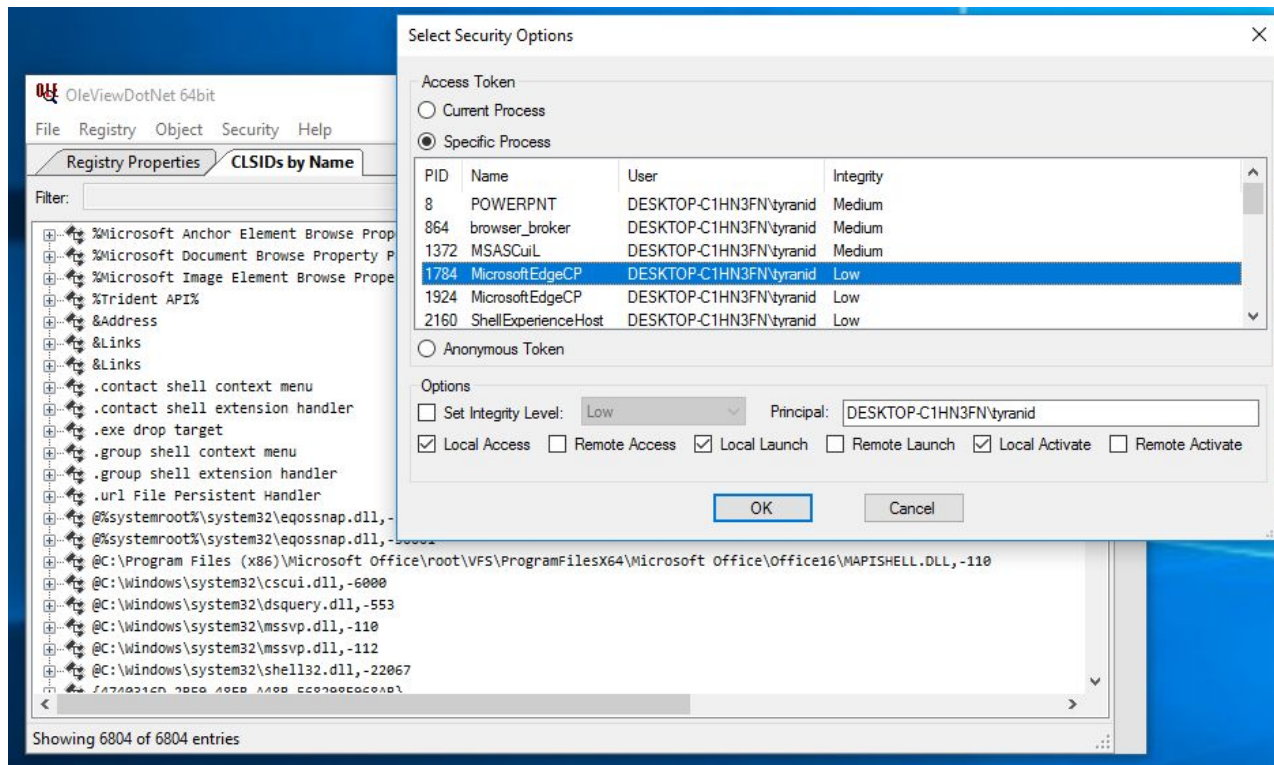
Accessible Objects from a Sandbox

- There's two ways to get a reference to a COM object
 - Activate via a broker (such as IE elevation policy)



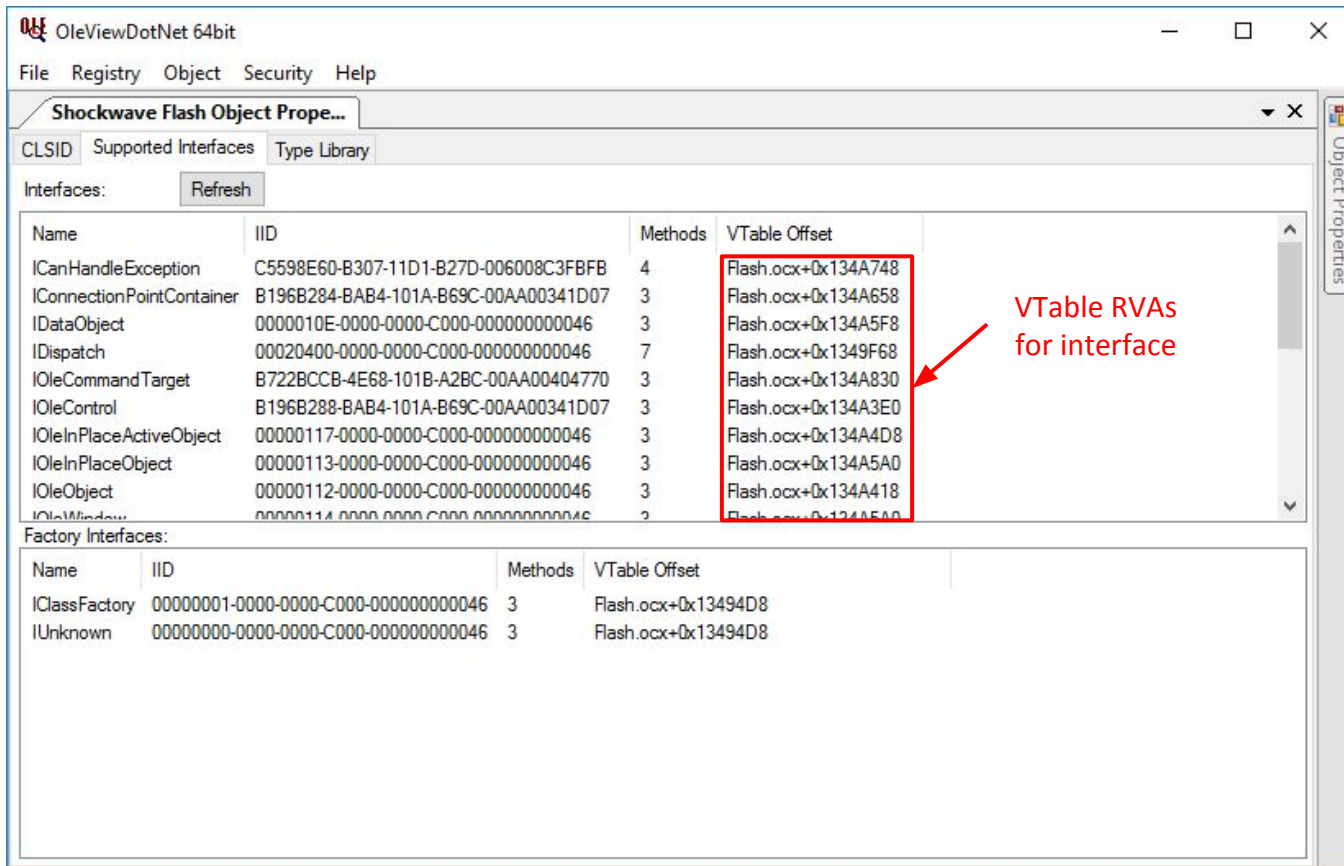
Accessible Objects from a Sandbox

- There's two ways to get a reference to a COM object
 - Activate via a broker (such as IE elevation policy)
 - Launch/Activate one directly



Reverse Engineering COM Components

- In-process you can just inspect the vtable address.



OleViewDotNet 64bit

File Registry Object Security Help

Shockwave Flash Object Properties

CLSID Supported Interfaces Type Library

Interfaces: Refresh

Name	IID	Methods	VTable Offset
ICanHandleException	C5598E60-B307-11D1-B27D-006008C3F8FB	4	Flash.ocx+0x134A748
IConnectionPointContainer	B196B284-BAB4-101A-B69C-00AA00341D07	3	Flash.ocx+0x134A658
IDataObject	0000010E-0000-0000-C000-000000000046	3	Flash.ocx+0x134A5F8
IDispatch	00020400-0000-0000-C000-000000000046	7	Flash.ocx+0x1349F68
IOleCommandTarget	B722BCCB-4E68-101B-A2BC-00AA00404770	3	Flash.ocx+0x134A830
IOleControl	B196B288-BAB4-101A-B69C-00AA00341D07	3	Flash.ocx+0x134A3E0
IOleInPlaceActiveObject	00000117-0000-0000-C000-000000000046	3	Flash.ocx+0x134A4D8
IOleInPlaceObject	00000113-0000-0000-C000-000000000046	3	Flash.ocx+0x134A5A0
IOleObject	00000112-0000-0000-C000-000000000046	3	Flash.ocx+0x134A418
IOleWindow	00000114-0000-0000-C000-000000000046	2	Flash.ocx+0x134A5A0

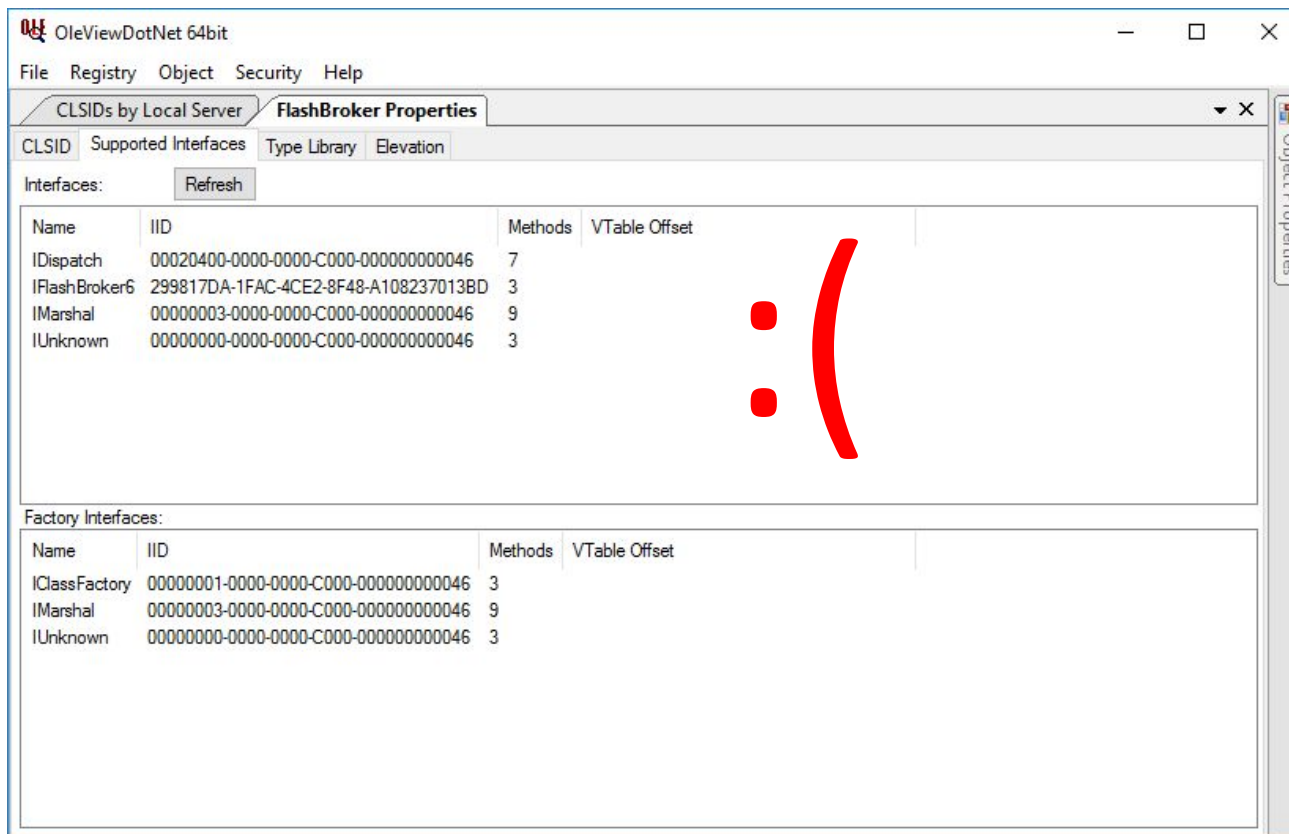
Factory Interfaces:

Name	IID	Methods	VTable Offset
IClassFactory	00000001-0000-0000-C000-000000000046	3	Flash.ocx+0x13494D8
IUnknown	00000000-0000-0000-C000-000000000046	3	Flash.ocx+0x13494D8

VTable RVAs for interface

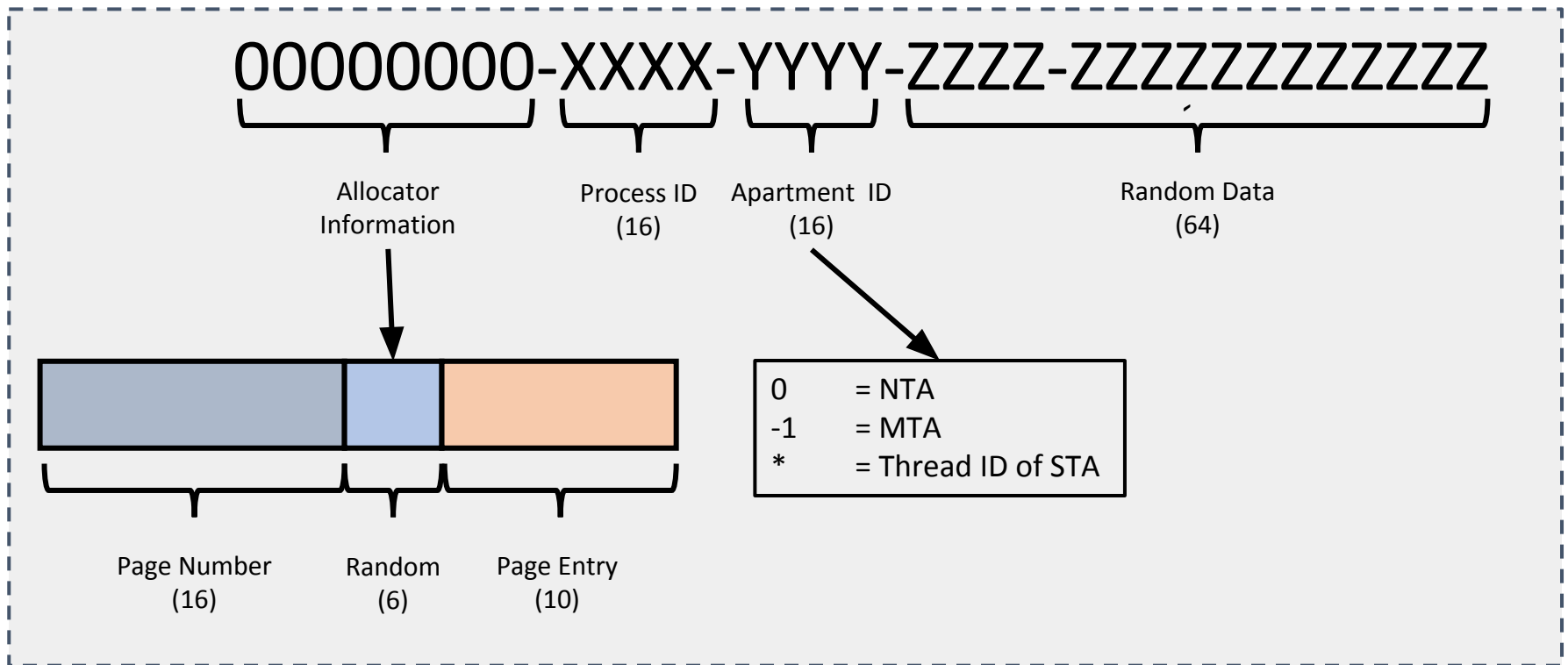
Reverse Engineering COM Components

- No such luck for OOP objects.

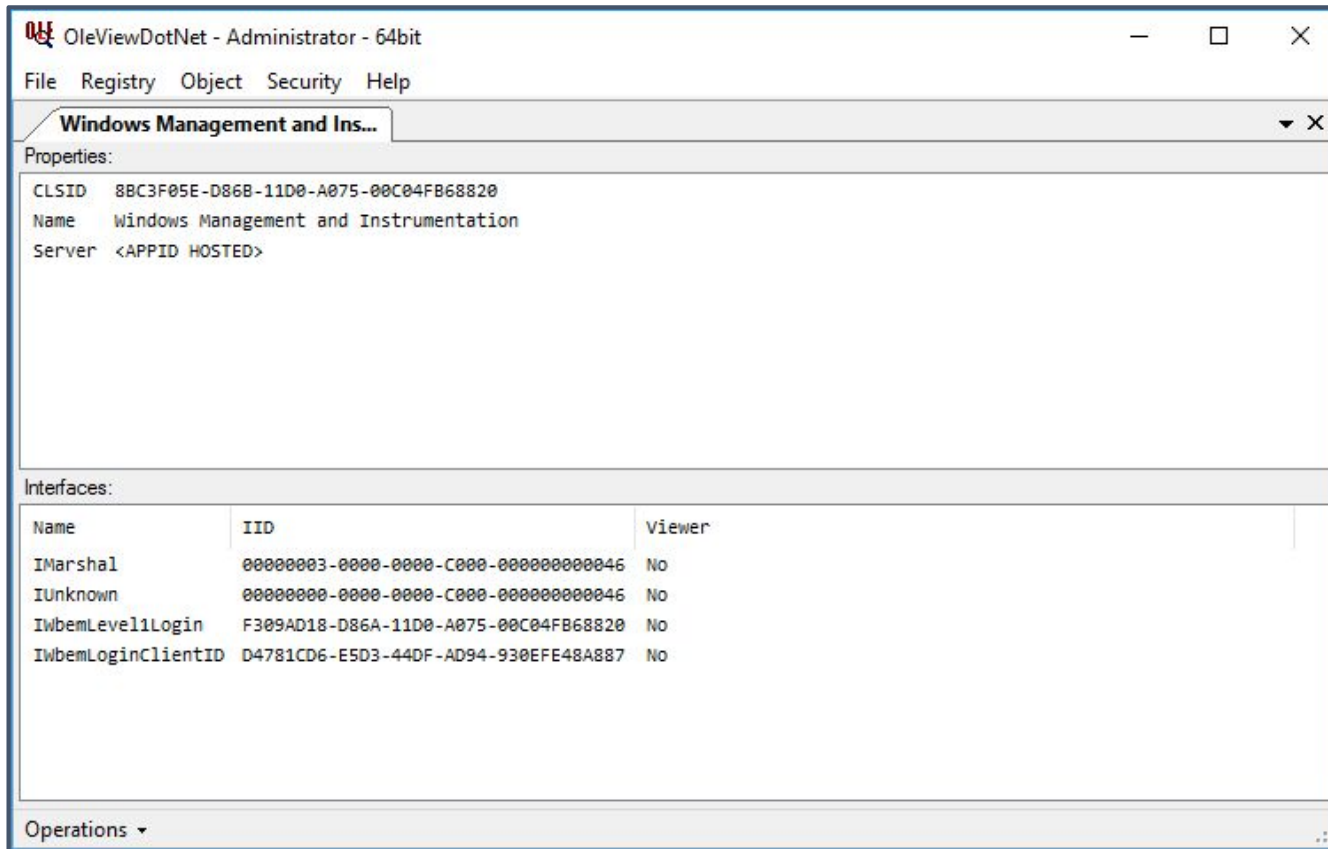


IPID Structure

- You'd assume that the IPID was a random GUID, but nope.



Tracking Down OOP VTable



OleViewDotNet - Administrator - 64bit

File Registry Object Security Help

Windows Management and Ins...

Properties:

CLSID 8BC3F05E-D86B-11D0-A075-00C04FB68820
Name Windows Management and Instrumentation
Server <APPID HOSTED>

Interfaces:

Name	IID	Viewer
IMarshal	00000003-0000-0000-C000-000000000046	No
IUnknown	00000000-0000-0000-C000-000000000046	No
IwbemLevel1Login	F309AD18-D86A-11D0-A075-00C04FB68820	No
IwbemLoginClientID	D4781CD6-E5D3-44DF-AD94-930EFE48A887	No

Operations ▾

Tracking Down OOP VTable

OleViewDotNet - Administrator - 64bit

File Registry Object Security Help

Windows Management and Ins...

Properties:

CLSID 8BC3F05E-D86B-11D0-A075-00C04FB68820

Name Windows Management and Instrumentation

Server <APPID HOSTED>

Interfaces:

Name	IID	Viewer
IMarshal	00000003-0000-0000-C000-000000000046	No
IUnknown	00000000-0000-0000-C000-000000000046	No
IWbemLevel1Login	F309AD18-D86A-11D0-A075-00C04FB68820	No
IWbemLoginClientID	D4781CD6-E5D3-44DF-AD94-930EFE48A887	No

Operations ▾

OleViewDotNet - Administrator - 64bit

File Registry Object Security Help

Windows Management and Instr... Marshal Viewer - Standard

OBJREF Type: Standard IID: F309AD18-D86A-11D0-A075-00C04FB68820 IID Name: IWbemLevel1Login

Standard Flags: 0x0 Public Refs: 1

OXID: 0x4DDFA1BDC588A3DB OID: 0x2DA8A0652473E24F

IPID: 0001482F-01A8-0000-F9F2-3464FAED95DA Apartment: NTA

Process ID: 424 Process Name: svchost

String Bindings:

Tower ID	Network Address
Tcp	DESKTOP-C1HN3FN
Tcp	10.0.2.15
Tcp	192.168.56.103

Security Bindings:

Authentication Service	Principal Name
GSS_Negotiate	
NegoExtender	
GSS_Kerberos	
WinNT	
22	
PKU2U	
GSS_SChannel	

Marshal

PID from IPID

Tracking Down OOP VTable

OleViewDotNet - Administrator - 64bit

File Registry Object Security Help

Windows Management and Instr... Marshal Viewer - Standard COM Processes

Filter: Mode: Contains Apply

IPID: 0001E822-01A8-1860-201F-388915ABA257	- IID: INotifyNetworkListManagerEvents
IPID: 0001DC25-01A8-0C10-B9FD-96998B9D361D	- IID: INotifyNetworkInterfaceEvents
IPID: 0001E826-01A8-0C10-5A1D-914532F13B4C	- IID: IConnectionProvider
IPID: 0001C428-01A8-0000-A69E-AC93C53B7739	- IID: ILocalSystemActivator
IPID: 00017429-01A8-0000-8C3C-18452FEFBC21	- IID: IApplicationStateChangeHandler
IPID: 0001042A-01A8-FFFF-D514-DE4E8E21260C	- IID: IBackgroundAccessStateChangeHandler
IPID: 0001A42D-01A8-0000-7C28-E4B4A7FC544F	- IID: INotifyNetworkInterfaceEvents
IPID: 00016C2E-01A8-FFFF-8FEC-9ADA33A7C4C5	- IID: IWpnAppEndpoint
IPID: 0001482F-01A8-0000-F9F2-3464FAED95DA	- IID: IwbemLevel1Login
IPID: 0001F030-01A8-0C10-F6AD-C870F3AACB09	- IID: INotifyNetworkListManagerEvents
IPID: 00019431-01A8-FFFF-F8F1-A726FFE8C121	- IID: IConnectionManagerCallbacks2
IPID: 00029400-01A8-1860-4EBB-2D48E0D39C90	- IID: INotifyNetworkEvents
IPID: 00028401-01A8-1860-7544-6CFF90C6CC0B	- IID: INotifyNetworkInterfaceEvents
IPID: 0002B402-01A8-0000-EEFA-F3265F191235	- IID: INotifyNetworkEvents
IPID: 0002F003-01A8-0000-368F-A5B8C0221761	- IID: IUnknown
IPID: 0002DC04-01A8-0000-2DC3-02A1FAC540A8	- IID: INotifyNetworkListManagerEvents
IPID: 00020008-01A8-0000-82DC-CA1895D39166	- IID: INetworkEvents
IPID: 0002BC0A-01A8-0000-E591-02EAAE6261D1	- IID: INotifyNetworkInterfaceEvents
IPID: 0002400C-01A8-0C10-2BDB-44DEC0901080	- IID: IUPnPDeviceFinder
IPID: 0002BC0E-01A8-0000-DD8C-B15F7334B223	- IID: INotifyNetworkGlobalCostEvents
IPID: 00029010-01A8-0000-5699-106337F32C27	- IID: INotifyNetworkGlobalCostEvents

Showing 29 of 29 entries

Find IPID in Process List

Tracking Down OOP VTable

The image shows two windows of OleViewDotNet. The top window displays a list of 29 COM processes. The bottom window shows the 'Properties' of the selected process, IPID: 0001482F-01A8-0000-F9F2-3464FAED95DA.

COM Processes List:

IPID	IID
0001E822-01A8-1860-201F-388915ABA257	INotifyNetworkListManagerEvents
0001DC25-01A8-0C10-B9FD-96998B9D361D	INotifyNetworkInterfaceEvents
0001E826-01A8-0C10-5A1D-914532F13B4C	IConnectionProvider
0001C428-01A8-0000-A69E-AC93C53B7739	ILocalSystemActivator
00017429-01A8-0000-8C3C-18452FEFBC21	
0001042A-01A8-FFFF-D514-DE4E8E21260C	
0001A42D-01A8-0000-7C28-E4B4A7FC544F	
00016C2E-01A8-FFFF-8FEC-9ADA33A7C4C5	
0001482F-01A8-0000-F9F2-3464FAED95DA	
0001F030-01A8-0C10-F6AD	
00019431-01A8-FFFF-F8F1	
00029400-01A8-1860-4EBB	
00028401-01A8-1860-7544-6CFF90C6CC0B	
0002B402-01A8-0000-EEFA-F3265F191235	
0002F003-01A8-0000-368F-A5B8C0221761	
0002DC04-01A8-0000-2DC3-02A1FAC540A8	
00020008-01A8-0000-82DC-CA1895D39166	
0002BC0A-01A8-0000-E591-02EAAE6261D1	
0002400C-01A8-0C10-2BDB-44DEC0901080	
0002BC0E-01A8-0000-DD8C-B15F7334B223	
00029010-01A8-0000-5699-106337F32C27	

Properties of IPID: 0001482F-01A8-0000-F9F2-3464FAED95DA:

Property	Value
IPID	0001482F-01A8-0000-F9F2-3464FAED95DA
IID	F309AD18-D86A-11D0-A075-00C04FB68820
IID Name	IWbemLevel1Login
Flags	IPIDF_SERVERENTRY
Interface	0x13AFD765910
VTable	wbemcore+0xDA448
Stub	0x13AF52B64F0
VTable	wbemsvc+0x7650
OXID	C588A3DB-A1BD-4DDF-6117-1CF7D8ED199D
References	Strong: 5, Weak: 0, Private: 0
PID	424
Apartment	NTA
STA Hwnd	0x0

Showing 29 of 29 entries

Red arrows point from the 'Properties' label to the 'Flags' field and from the 'VTable RVA' label to the 'VTable' field.

VTable Reverse Engineering

```
.rdata:000000001800D79E0 ; const CJobManagerExternal::`vftable'
.rdata:000000001800D79E0 ??_7CJobManagerExternal@@6B@ dq offset ?QueryInterface@?$RuntimeClass@U?$InterfaceList@U
.rdata:000000001800D79E0 ; DATA XREF: CJobManagerExternal::CJobManagerExt
.rdata:000000001800D79E0 ; CJobManagerExternal::~CJobManagerExternal(void
.rdata:000000001800D79E0 ; Microsoft::WRL::Details::RuntimeClass<Microsof
.rdata:000000001800D79E8 dq offset ?AddRef@?$RuntimeClass@U?$InterfaceList@UIBackgroundCopyFile@
.rdata:000000001800D79F0 dq offset ?Release@?$RuntimeClass@U?$InterfaceList@UIBackgroundCopyManag
.rdata:000000001800D79F8 dq offset ?CreateJob@CJobManagerExternal@@UEAAJPEBGW4__MIDL_IBackgroundC
.rdata:000000001800D7A00 dq offset ?GetJob@CJobManagerExternal@@UEAAJAEBU_GUID@@PEAPEAUIBackground
.rdata:000000001800D7A08 dq offset ?EnumJobs@CJobManagerExternal@@UEAAJKPEAPEAUIEnumBackgroundCop
.rdata:000000001800D7A10 dq offset ?GetErrorDescription@CJobManagerExternal@@UEAAJJKPEAPEAGZ ; C
.rdata:000000001800D7A18 dq offset ??_ECJobManagerExternal@@UEAAPEAXI@Z ; CJobManagerExternal::`u
```

IUnknown

Interface
Specific

QueryInterface Implementations

```
HRESULT QueryInterface (REFIID riid, void** ppv) {  
    if (riid == IID_IUnknown || riid == IID_Interface1)  
    {  
        *ppv = this;  
    } else if (riid == IID_Interface2) {  
        *ppv = static_cast<Interface2*>(this);  
    } else {  
        return E_NOINTERFACE;  
    }  
    ((IUnknown*)*ppv)->AddRef ();  
    return S_OK;  
}
```

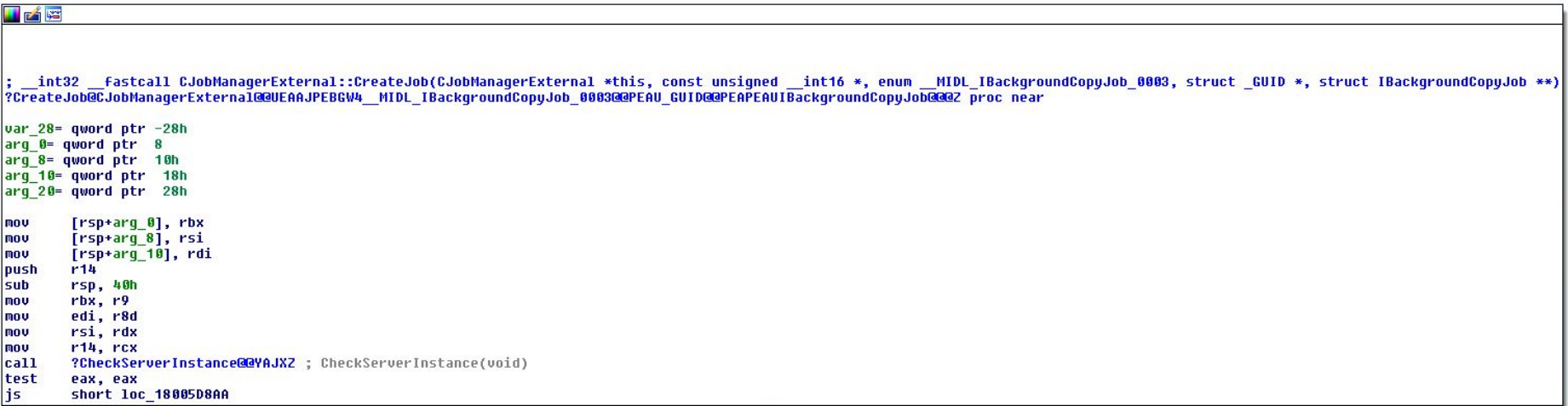
Explicit Implementation

```
HRESULT QueryInterface (REFIID riid, void** ppv) {  
    QITAB qit[] = {{ &IID_Interface1, 0 },  
        { &IID_Interface2, 8 } { NULL, 0 } };  
    return QISearch (this, qit, riid, ppv);  
}
```

Using QISearch Helper

Interface Information - Public Symbols

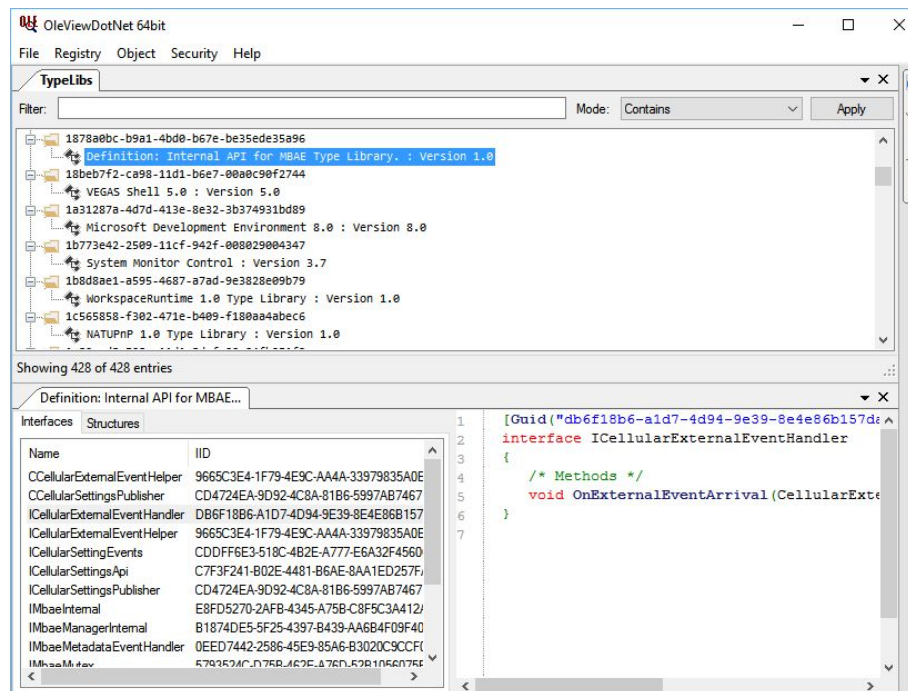
- Most Windows components come with public symbols available
- Most COM code written in C++
- So, use C++ managed names to recover some parameter information



```
;__int32 __fastcall CJobManagerExternal::CreateJob(CJobManagerExternal *this, const unsigned __int16 *, enum __MIDL_IBackgroundCopyJob_0003, struct _GUID *, struct IBackgroundCopyJob **)  
?CreateJob@CJobManagerExternal@@UEAAJPEBGW4__MIDL_IBackgroundCopyJob_0003@@PEAU_GUID@@PEAUIBackgroundCopyJob@@@Z proc near  
  
var_28= qword ptr -28h  
arg_0= qword ptr 8  
arg_8= qword ptr 10h  
arg_10= qword ptr 18h  
arg_20= qword ptr 28h  
  
mov     [rsp+arg_0], rbx  
mov     [rsp+arg_8], rsi  
mov     [rsp+arg_10], rdi  
push    r14  
sub     rsp, 40h  
mov     rbx, r9  
mov     edi, r8d  
mov     rsi, rdx  
mov     r14, rcx  
call    ?CheckServerInstance@@YAJXZ ; CheckServerInstance(void)  
test    eax, eax  
js      short loc_18005D8AA
```

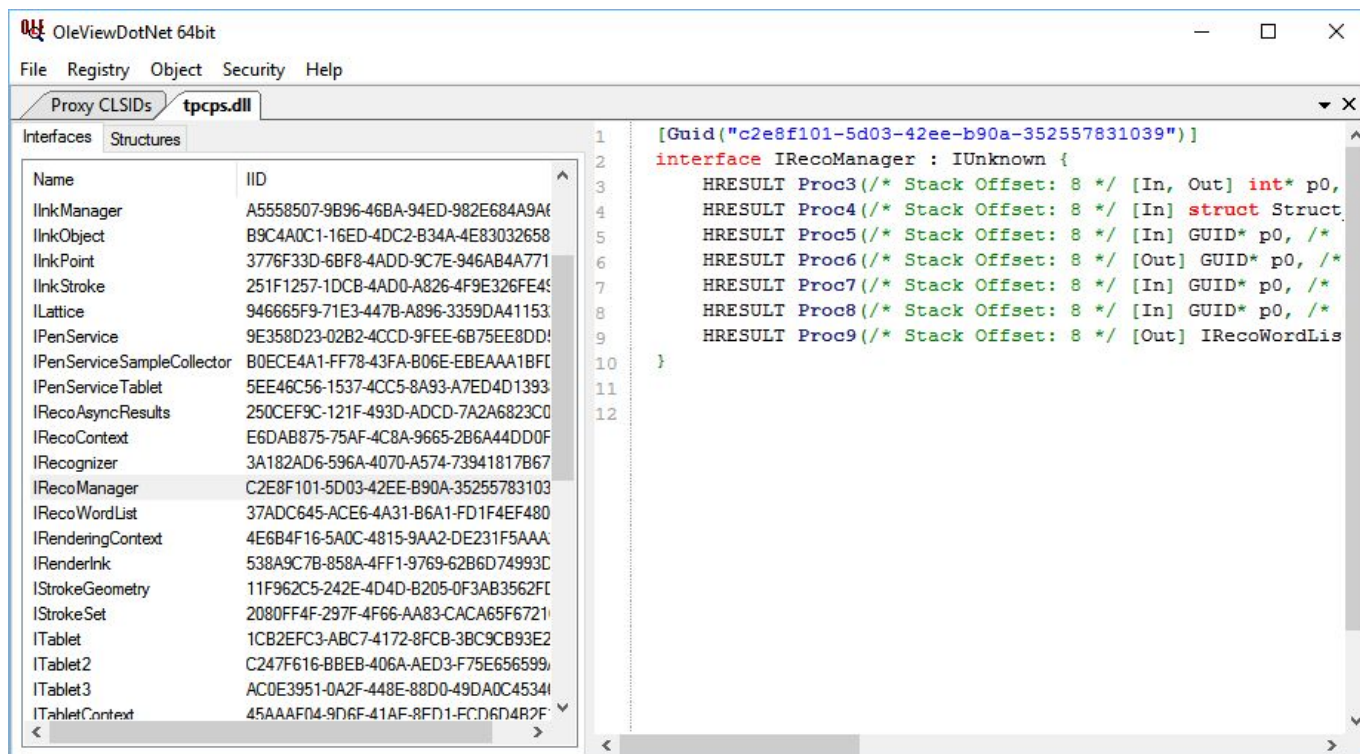

Interface Information - Type Libraries

- Some objects have type library information which is a structured format for describing COM interfaces.
- “import” the library into a C++ project with `#import <name.tlb>` preprocessor directory



Interface Information - Proxy/Stub NDR

- Proxies and Stubs contains NDR bytecode which describe how to marshal parameters back and forth.
- Contains information about interfaces and structures



OleViewDotNet 64bit

File Registry Object Security Help

Proxy CLSIDs tpcps.dll

Interfaces Structures

Name	IID
IInkManager	A5558507-9B96-46BA-94ED-982E684A9A6
IInkObject	B9C4A0C1-16ED-4DC2-B34A-4E83032658
IInkPoint	3776F33D-6BF8-4ADD-9C7E-946AB4A771
IInkStroke	251F1257-1DCB-4AD0-A826-4F9E326FE49
ILattice	946665F9-71E3-447B-A896-3359DA41153
IPenService	9E358D23-02B2-4CCD-9FEE-6B75EE8DD
IPenServiceSampleCollector	B0ECE4A1-FF78-43FA-B06E-EBEAAA1BFC
IPenServiceTablet	5EE46C56-1537-4CC5-8A93-A7ED4D1393
IRecoAsyncResult	250CEF9C-121F-493D-ADCD-7A2A6823C0
IRecoContext	E6DAB875-75AF-4C8A-9665-2B6A44DD0F
IRecognizer	3A182AD6-596A-4070-A574-73941817B67
IRecoManager	C2E8F101-5D03-42EE-B90A-352557831039
IRecoWordList	37ADC645-ACE6-4A31-B6A1-FD1F4EF480
IRenderingContext	4E6B4F16-5A0C-4815-9AA2-DE231F5AAA
IRenderInk	538A9C7B-858A-4FF1-9769-62B6D74993C
IStrokeGeometry	11F962C5-242E-4D4D-B205-0F3AB3562F
IStrokeSet	2080FF4F-297F-4F66-AA83-CACA65F6721
ITablet	1CB2EFC3-ABC7-4172-8FCB-3BC9CB93E2
ITablet2	C247F616-BBEB-406A-AED3-F75E656599
ITablet3	AC0E3951-0A2F-448E-88D0-49DA0C4534
ITabletContext	45AAAF04-9D6F-41AF-8FD1-FCDD6D4R2F

```
1 [Guid("c2e8f101-5d03-42ee-b90a-352557831039")]
2 interface IRecoManager : IUnknown {
3     HRESULT Proc3(/* Stack Offset: 8 */ [In, Out] int* p0,
4     HRESULT Proc4(/* Stack Offset: 8 */ [In] struct Struct
5     HRESULT Proc5(/* Stack Offset: 8 */ [In] GUID* p0, /*
6     HRESULT Proc6(/* Stack Offset: 8 */ [Out] GUID* p0, /*
7     HRESULT Proc7(/* Stack Offset: 8 */ [In] GUID* p0, /*
8     HRESULT Proc8(/* Stack Offset: 8 */ [In] GUID* p0, /*
9     HRESULT Proc9(/* Stack Offset: 8 */ [Out] IRecoWordList
10 }
11
12
```



Bugs and "Features"

Activation Properties In- SPD

```
struct SpecialPropertiesData {  
    unsigned long dwSessionId;  
    long fRemoteThisSessionId;  
    long fClientImpersonating;  
    long fPartitionIDPresent;  
    DWORD dwDefaultAuthnLvl;  
    GUID guidPartition;  
    DWORD dwPRTFlags;  
    DWORD dwOrigClsctx;  
    DWORD dwFlags;  
    DWORD dwPid;  
    unsigned __int64 hwnd;  
    DWORD ulServiceId;  
    DWORD Reserved[ 4 ];  
};
```

Choosing a Session ID?

UAC Related Stuff

```
enum SPD_FLAGS {  
    SPD_FLAG_USE_CONSOLE_SESSION,  
    SPD_FLAG_USE_DEFAULT_AUTHN_LVL,  
    SPD_FLAG_USE_SERVER_PID,  
    SPD_FLAG_USE_LUA_LEVEL_ADMIN,  
    SPD_FLAG_COAUTH_USER_IS_NULL,  
    SPD_FLAG_COAUTH_DOMAIN_IS_NULL,  
    SPD_FLAG_COAUTH_PWD_IS_NULL,  
    SPD_FLAG_USE_LUA_LEVEL_HIGHEST  
};
```

Session and Elevation Monikers

Session-to-Session Activation with a Session Moniker

Session-to-session activation (also called cross-session activation) allows a client process to start (activate) a local server process on a specified session. This feature is available for applications that are configured to run in the security context of the interactive user, also known as the "RunAs Interactive User" object activation mode. For more information about security contexts, see [The Client's Security Context](#).

Distributed COM (DCOM) enables object activation on a per-session basis by using a system-supplied [Session Moniker](#). Other system-supplied monikers include [file monikers](#), [item monikers](#), generic [composite monikers](#), [anti-monikers](#), [pointer monikers](#), and [URL monikers](#).

The COM Elevation Moniker

The COM elevation moniker allows applications that are running under user account control (UAC) to activate COM classes with elevated privileges. For more information, see [Focus on Least Privilege](#).

When to Use the Elevation Moniker

The elevation moniker is used to activate a COM class to accomplish a specific and limited function that requires elevated privileges, such as changing the system date and time.

Elevation requires participation from both a COM class and its client. The COM class must be configured to support elevation by annotating its registry entry, as described in the Requirements section. The COM client must request elevation by using the elevation moniker.

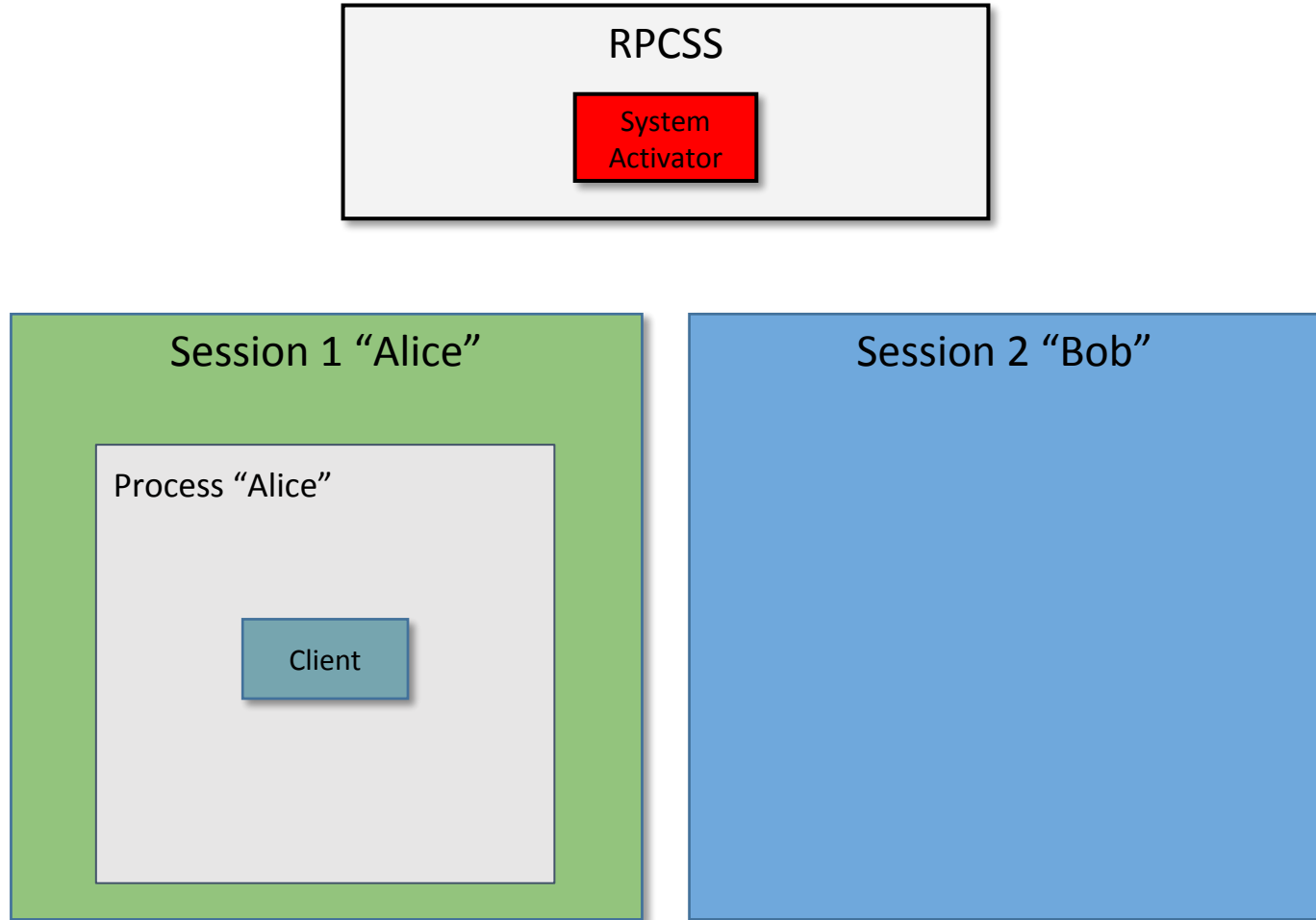
Monikers and Binding

- Monikers are string representation of COM objects.
- Pass string to CoGetObject to “Bind” to the underlying object.

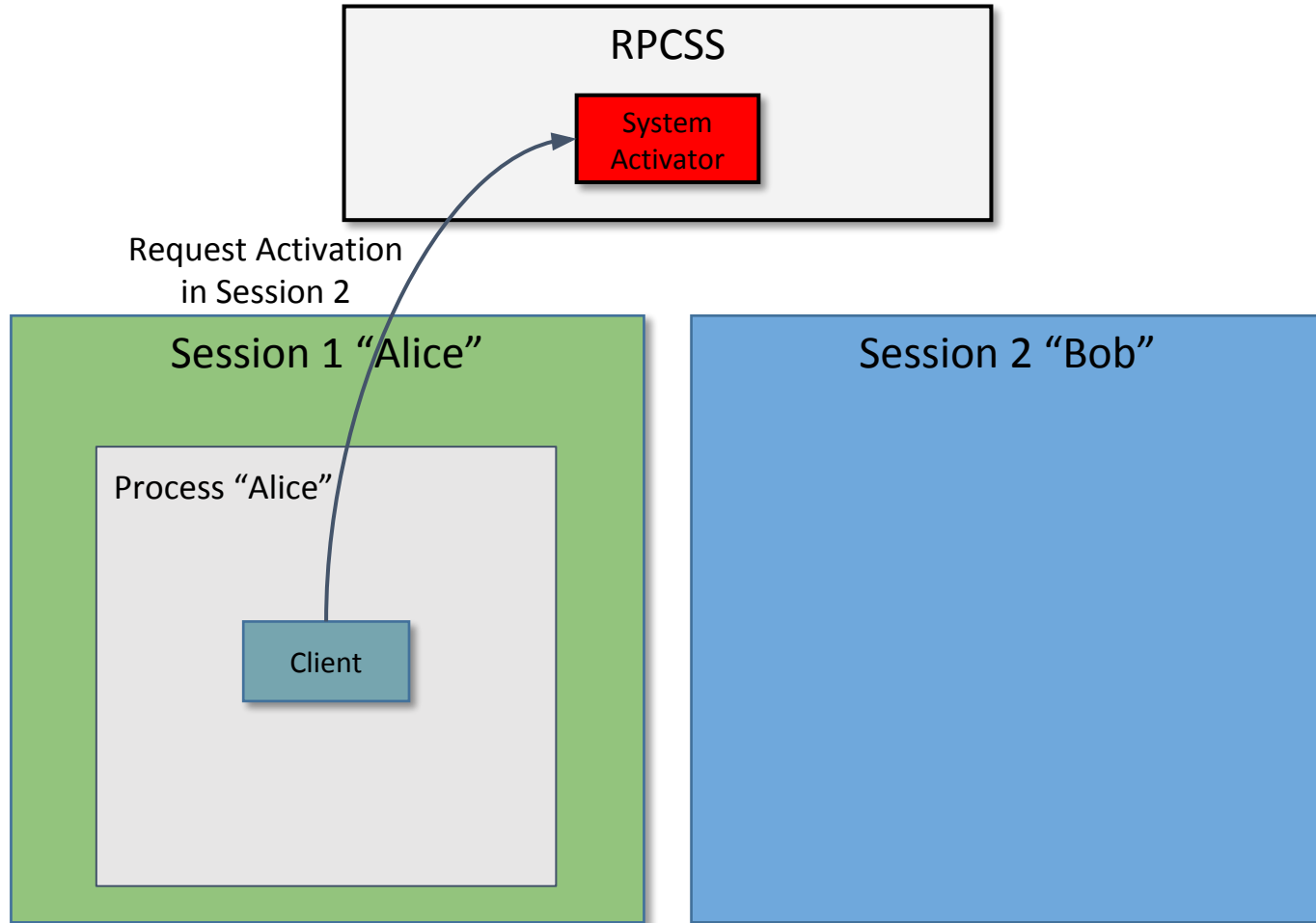
```
HRESULT CoCreateInstanceInSession (DWORD session,
    REFCLSID rclsid, REFIID riid, void ** ppv) {
    BIND_OPTS3 bo = {};
    WCHAR  wszCLSID [50];
    WCHAR  wszMonikerName [300];

    StringFromGUID2 (rclsid, wszCLSID, _countof (wszCLSID));
    StringCchPrintf (wszMonikerName, _countof (wszMonikerName),
        L"session:%d!new:%s", session, wszCLSID);
    bo.cbStruct = sizeof (bo);
    bo.dwClassContext = CLSCTX_LOCAL_SERVER;
    return CoGetObject (wszMonikerName, &bo, riid, ppv);
}
```

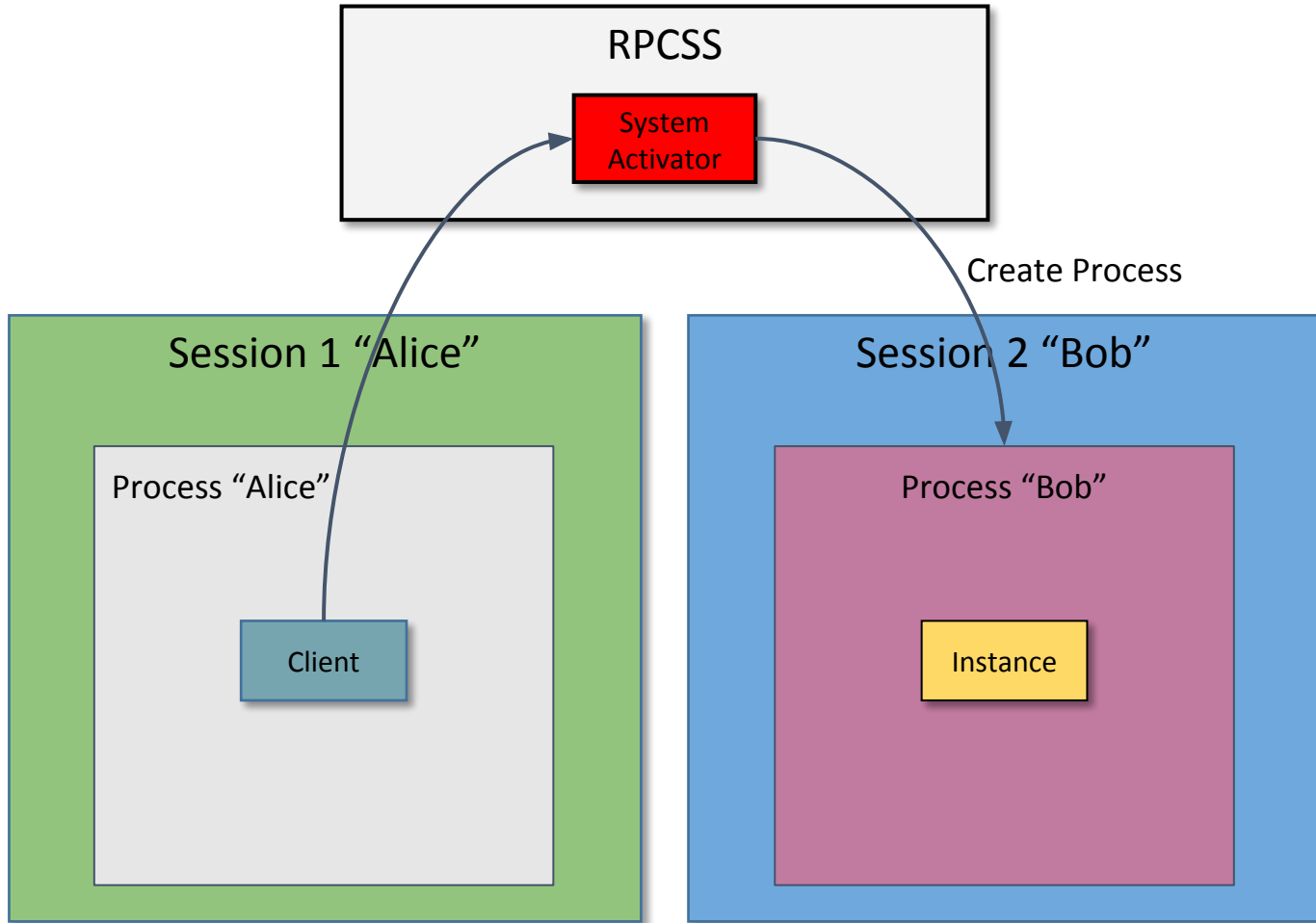
Session Moniker in Action



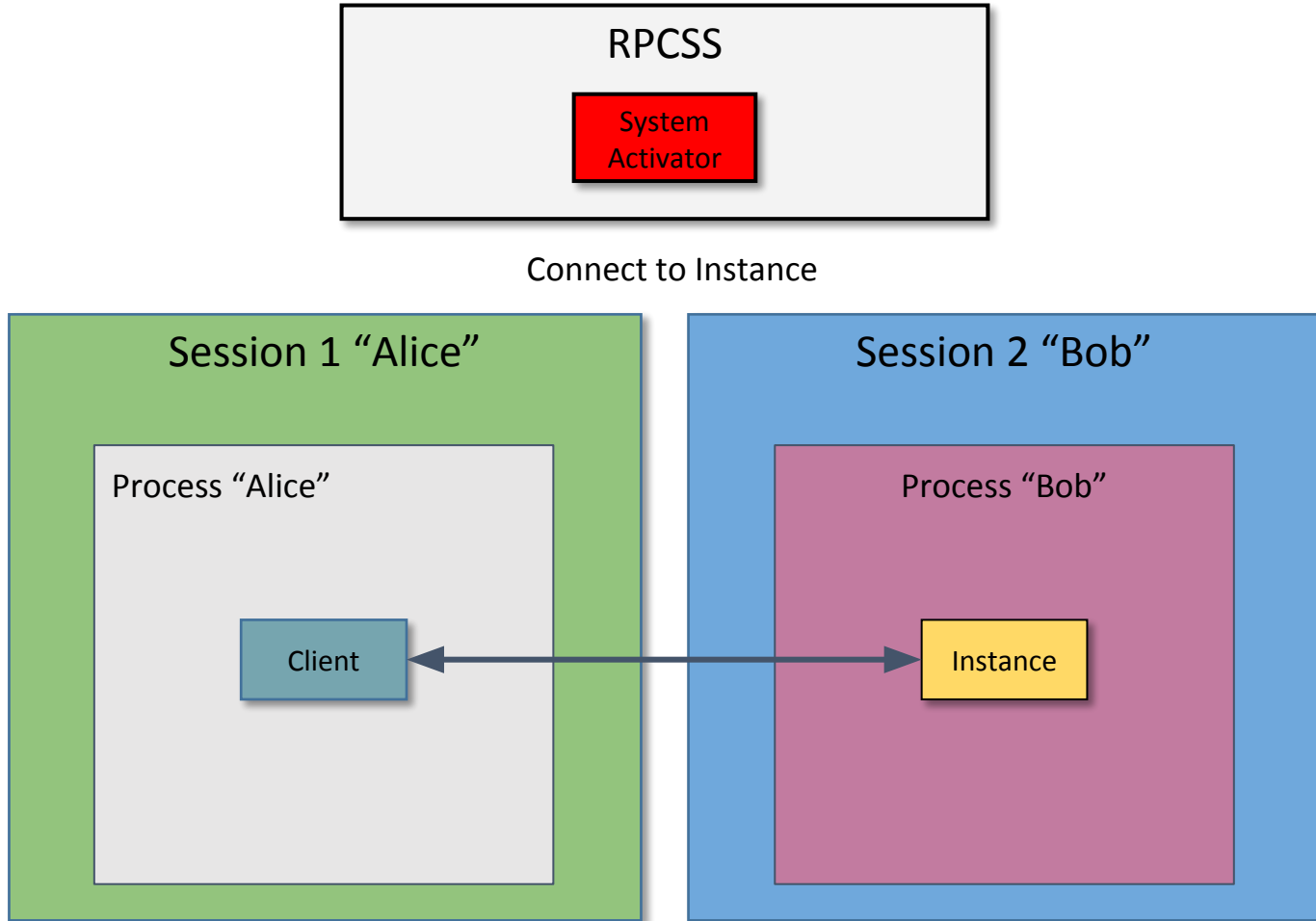
Session Moniker in Action



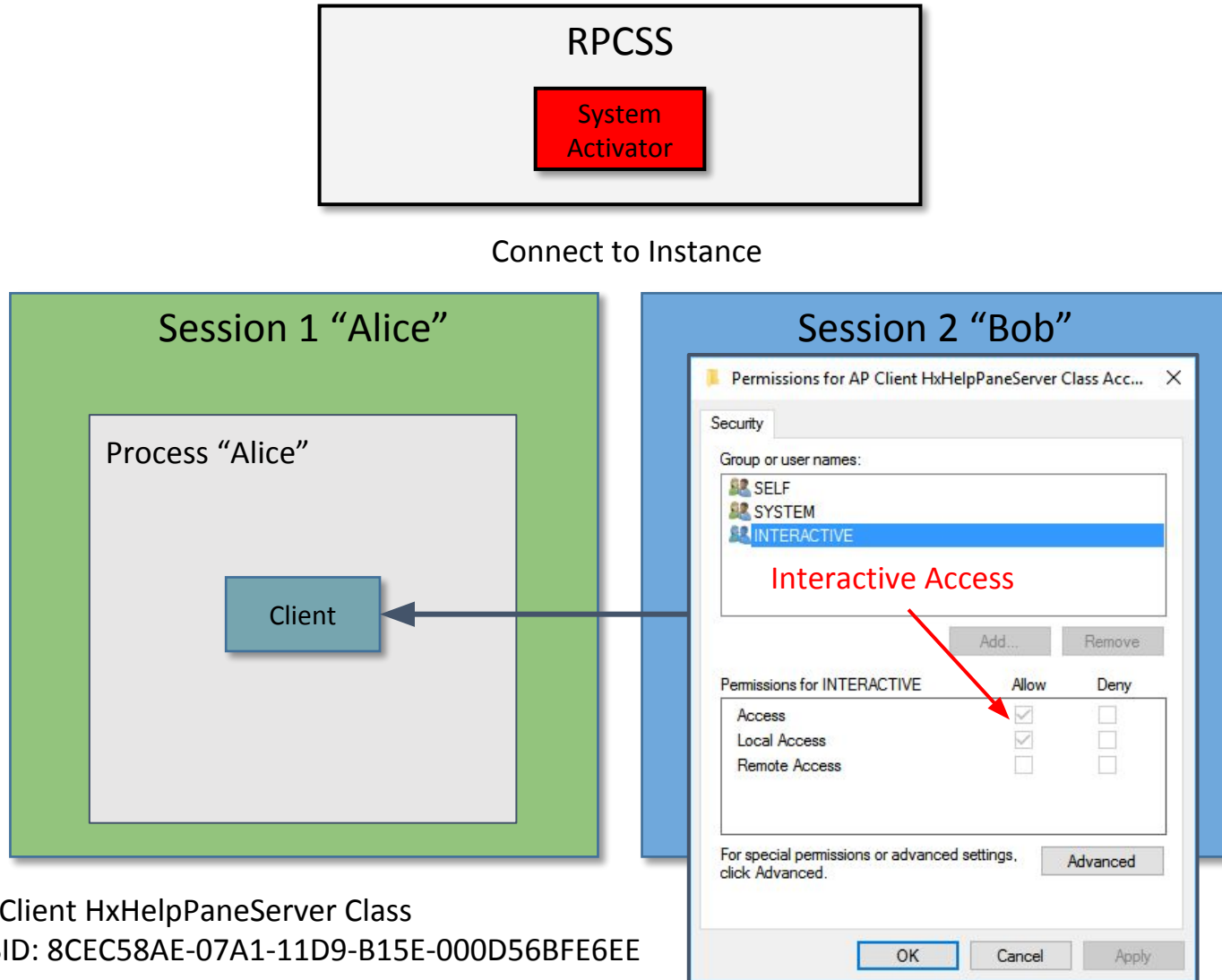
Session Moniker in Action



Session Moniker in Action



Session Moniker in Action



Abusing Information Disclosure

OleViewDotNet 64bit

File Registry Object Security Help

COM Processes

Filter: Mode: Contains Apply

5324 - ApplicationFrameHost - DESKTOP-C1HN3FN\tyranid
5872 - backgroundTaskHost - DESKTOP-C1HN3FN\tyranid
864 - browser_broker - DESKTOP-C1HN3FN\tyranid
6312 - dllhost - DESKTOP-C1HN3FN\tyranid
3684 - explorer - DESKTOP-C1HN3FN\tyranid

IPID: 0000B400-0E64-0954-2160-F92A42C0B66C - IID: IUnknown
IPID: 00005C01-0E64-0E68-5FE9-1B2558D62EA8 - IID: IRundown
IPID: 00009002-0E64-0000-6D46-E0FBA5074E6C - IID: IRundown
IPID: 00005403-0E64-FFFF-054D-BCD481B4ADCF - IID: IRundown
IPID: 0000B004-0E64-0F24-10DD-5EBB46C0BBF7 - IID: IRundown
IPID: 00001805-0E64-0F24-ABAC-45874BAB019B - IID: IAccPropServer
IPID: 0000D406-0E64-0F24-EE5A-8F89FB17DF34 - IID: IAccPropServer
IPID: 00005407-0E64-0F24-EC1D-9F88CE690B46 - IID: IAccPropServer
IPID: 0000C008-0E64-0F24-0C3B-FF105E6DA4CE - IID: IPrivDragDrop
IPID: 0000B409-0E64-FFFF-AD03-31A9EE08AB95 - IID: IUnknown
IPID: 0000840A-0E64-0F24-32EB-7B5EDD56A25B - IID: IUnknown
IPID: 0000140b-0e64-0e68-e75c-d4cd455824f1 - IID: ILocalSystemActivator
IPID: 0000040C-0E64-0FB4-C5C5-86FF7BCFEE64 - IID: IServiceProvider
IPID: 0000400D-0E64-0FE8-FDF3-B5835318D6C1 - IID: IUnknown
IPID: 0000F00E-0E64-0F24-5841-660C62BCCD1C - IID: IPrivDragDrop
IPID: 0000F80F-0E64-FFFF-91A8-C2DD19432CE6 - IID: IUnknown
IPID: 00005410-0E64-0F24-8880-58CB7D43AE6C - IID: IUnknown
IPID: 00001011-0E64-0FB4-45EC-8FD281180D55 - IID: IServiceProvider
IPID: 00005012-0E64-0FB4-211C-62E5DE6C4570 - IID: IRundown
IPID: 0000D013-0E64-0F24-7B48-18DF9D2C3DBF - IID: IPrivDragDrop
IPID: 00008414-0E64-FFFF-202F-1A9B16B75297 - IID: IUnknown
IPID: 00003015-0E64-0F24-A707-B81F475714E2 - IID: IUnknown
IPID: 00009417-0E64-0F98-6E04-AE548A3ACD43 - IID: IRundown
IPID: 00008818-0E64-0F98-FBD4-4C3302396301 - IID: ILocalSystemActivator

IPID 0000140b-0e64-0e68-e75c-d4cd455824f1

Properties:

CLSID 00000000-0000-0000-0000-000000000000

Interfaces:

Name	IID	Viewer
ILocalSystemActivator	00000132-0000-0000-C000-000000000046	No
IMarshal	00000003-0000-0000-C000-000000000046	No
ISystemActivator	000001A0-0000-0000-C000-000000000046	No
IUnknown	00000000-0000-0000-C000-000000000046	No

Showing 16 of 16 entries

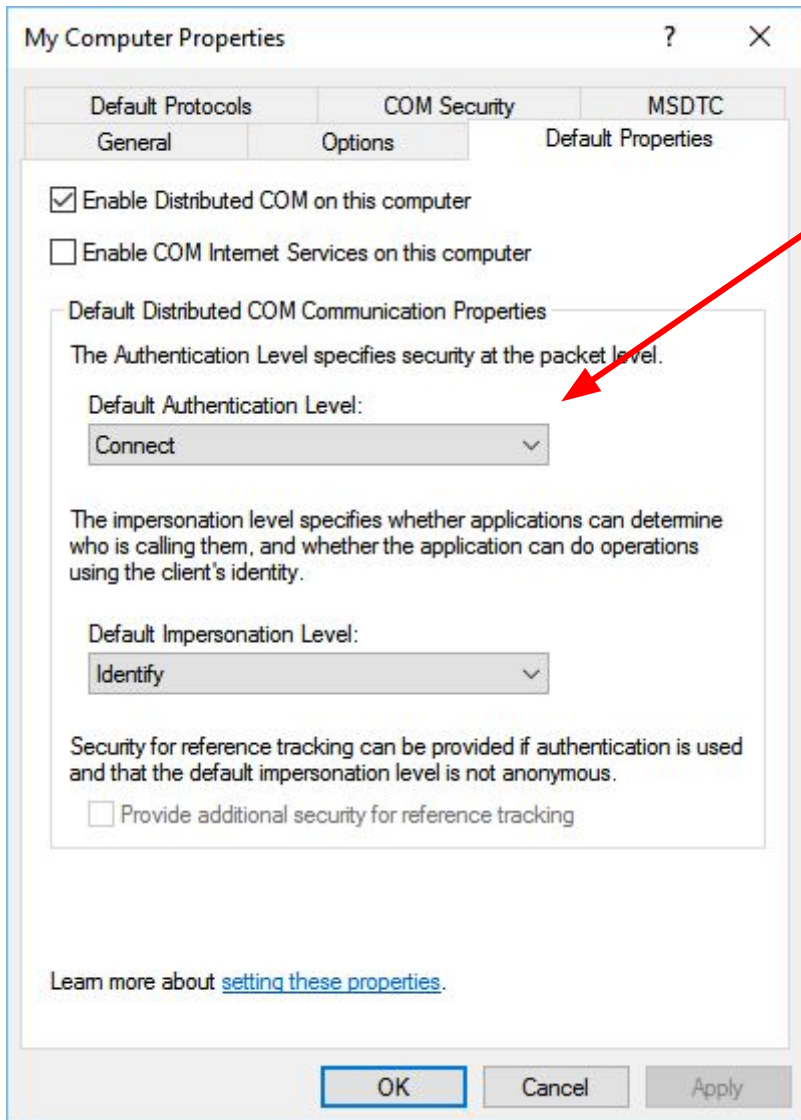
Operations

Unmarshal

You Can't Help Loving DCOM



Plain Text Communications?



Defaults to **CONNECT**
Authentication Level?

Authentication Level	
CONNECT	Authenticates the credentials of the client and server.
CALL/PKT	Same as CONNECT but also prevents replay attacks.
PKT_INTEGRITY	Same as CALL/PKT but also verifies that none of the data transferred between the client and server has been modified.
PKT_PRIVACY	Same as PKT_INTEGRITY but also ensures that the data transferred can only be seen unencrypted by the client and the server.

Good old Wireshark

The image shows the Wireshark network traffic analysis interface. The top menu bar includes File, Edit, View, Go, Capture, Analyze, Statistics, Telephony, Wireless, Tools, and Help. Below the menu is a toolbar with various icons for packet capture and analysis. A filter bar at the top displays the filter `dcerpc && ip.addr == 192.168.56.102`. The main packet list shows several packets, with packet 150 selected. The packet details pane on the right shows the structure of the selected packet, including IActProperties, CntData, and OBJREF. The packet bytes pane at the bottom shows the raw data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
126	58.048448	192.168.56.50	192.168.56.102	DCERPC	138	Bind_ack: call_id: 2, Fragment: Single, max_xm...
127	58.049465	192.168.56.102	192.168.56.50	IOXIDResolver	78	ServerAlive2 request IOXIDResolver V0
128	58.050659	192.168.56.50	192.168.56.102	IOXIDResolver	210	ServerAlive2 response[Long frame (2 bytes)]
145	58.062317	192.168.56.102	192.168.56.50	DCERPC	325	Bind: call_id: 3, Fragment: Single, 1 context ...
147	58.063403	192.168.56.50	192.168.56.102	DCERPC	290	Bind_ack: call_id: 3, Fragment: Single, max_xm...
148	58.063842	192.168.56.102	192.168.56.50	DCERPC	274	Alter_context: call_id: 3, Fragment: Single, 1...
149	58.064777	192.168.56.50	192.168.56.102	DCERPC	159	Alter_context_resp: call_id: 3, Fragment: Sing...
150	58.069377	192.168.56.102	192.168.56.50	ISystemActivator	854	RemoteCreateInstance request
151	58.101186	192.168.56.50	192.168.56.102	ISystemActivator	1022	RemoteCreateInstance response
169	58.118628	192.168.56.102	192.168.56.50	DCERPC	412	Bind: call_id: 2, Fragment: Single, 3 context ...

Packet 150 details:

- [No Specification Available: 00000000]
- IActProperties
 - CntData: 728
 - OBJREF
 - Signature: MEOW (0x574f454d)
 - Flags: OBJREF_CUSTOM (0x00000004)
 - IID: 000001a2-0000-0000-c000-000000000046

Packet 150 bytes:

```
0070 00 00 00 00 02 00 d8 02 00 00 d8 02 00 00 4d 45 .....ME
0080 4f 57 04 00 00 00 a2 01 00 00 00 00 00 00 c0 00 OW.....
0090 00 00 00 00 00 46 38 03 00 00 00 00 00 00 c0 00 ....F8.....
00a0 00 00 00 00 00 46 00 00 00 00 b0 02 00 00 a0 02 ....F.....
00b0 00 00 00 00 00 00 01 10 08 00 cc cc cc cc b0 00 .....
00c0 00 00 00 00 00 00 a0 02 00 00 c0 00 00 00 00 00 .....
00d0 00 00 02 00 00 00 06 00 00 00 00 00 00 00 00 .....
00e0 00 00 00 00 00 00 00 00 00 00 00 02 00 04 00 .....
00f0 02 00 00 00 00 00 06 00 00 b9 01 00 00 00 00 .....
0100 00 00 c0 00 00 00 00 00 00 46 ab 01 00 00 00 00 .....F.....
0110 00 00 c0 00 00 00 00 00 00 46 a5 01 00 00 00 00 .....F.....
```

OBJREF (dcom.objref), 728 bytes | Packets: 478 · Displayed: 216 (45.2%) · Dropped: 0 (0.0%) | Profile: Default

Access Permissions for WMI

The screenshot shows the OleViewDotNet application window titled "OleViewDotNet - Administrator - 64bit". The main window displays details for process 424, which is "svchost - NT AUTHORITY\SYSTEM". The details include the executable path, process ID, AppID, access permissions, LRPC permissions, user, security flags, STA HWND, and IPIDs.

Overlaid on this is a "Permissions for svchost Access" dialog box. The "Security" tab is active, showing a list of group or user names: "ALL APPLICATION PACKAGES", "Authenticated Users", and "INTERACTIVE". The "Authenticated Users" group is selected. Below the list are "Add..." and "Remove" buttons.

The "Permissions for Authenticated Users" section shows a table with columns for "Allow" and "Deny". The "Access" row has the "Allow" checkbox checked, while "Local Access" and "Remote Access" have both checkboxes unchecked.

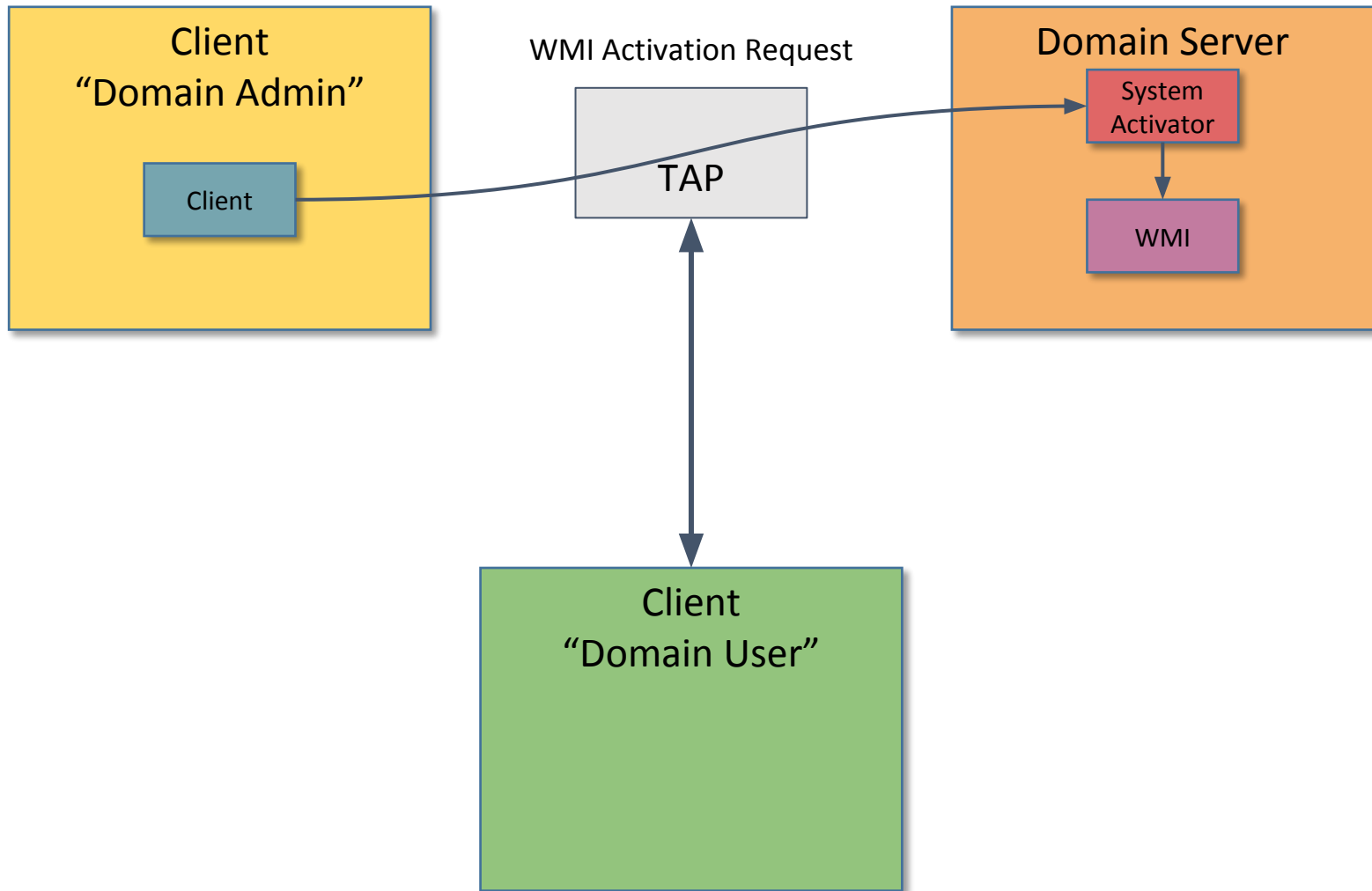
Below the table, a red text overlay states: "All authenticated users can access WMI remotely". At the bottom of the dialog, there is an "Advanced" button and a note: "For special permissions or advanced settings, click Advanced."

At the very bottom of the dialog are "OK", "Cancel", and "Apply" buttons.

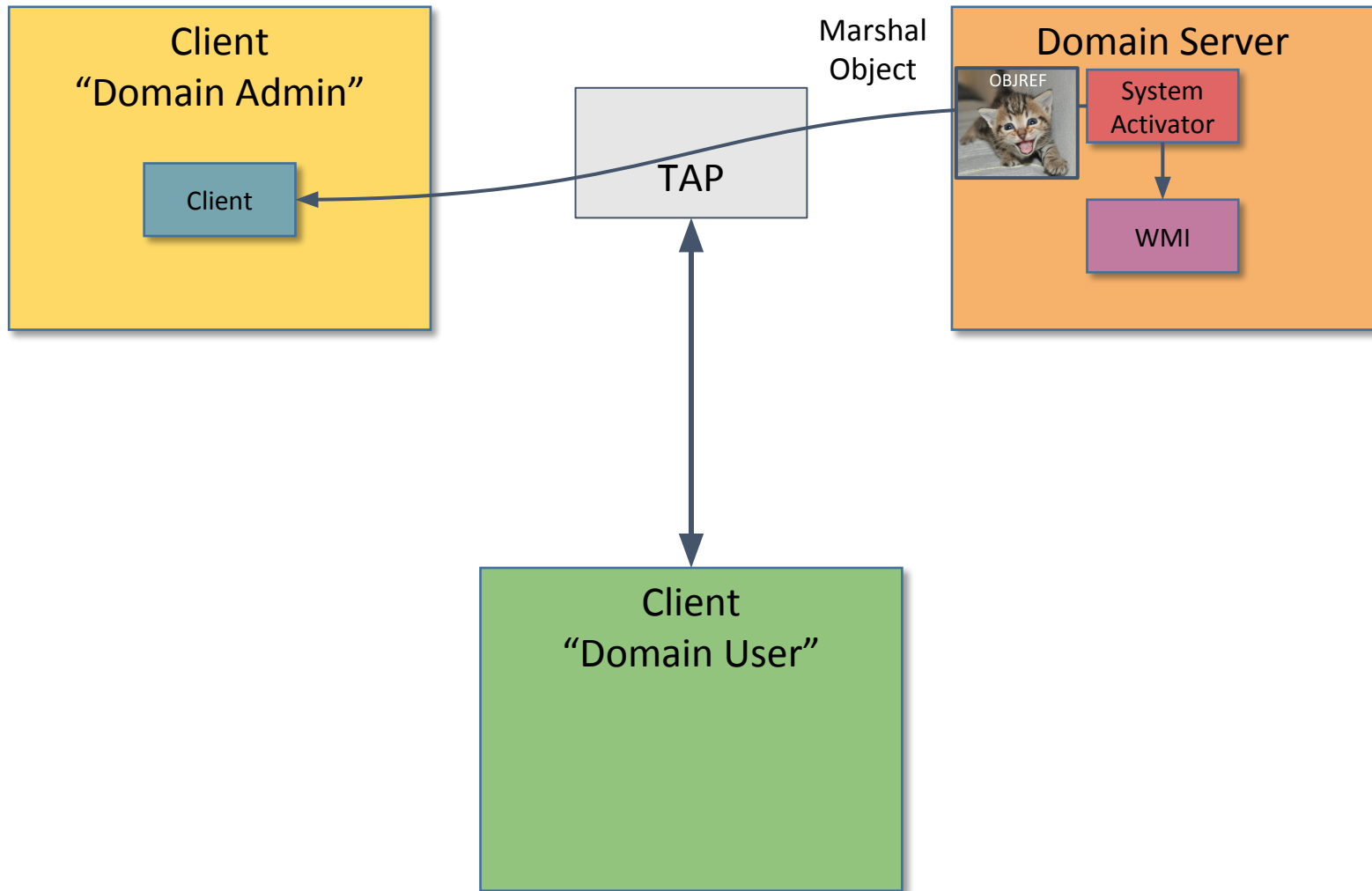
IPID	IID
00010011-01a8-ffff-30b4-9909b1525531	Windows.System.IUserStatics
00016012-01a8-ffff-4a39-00ff6dafaab0	Windows.System.IUserWatcher
0001a813-01a8-0000-6734-f05bd51503a5	IDLLHost
00015814-01a8-0000-d2f3-a3750b2f55e8	IWinRTInprocActivator
00016415-01a8-1860-2115-1ab567c6c430	INotifyNetworkEvents
00011816-01a8-1860-cae4-0979e3c555d1	INotifyNetworkListManagerEver
00015417-01a8-0000-f00-d6925df16405	IWbemServices

	Allow	Deny
Access	☒	☐
Local Access	☐	☐
Remote Access	☐	☐

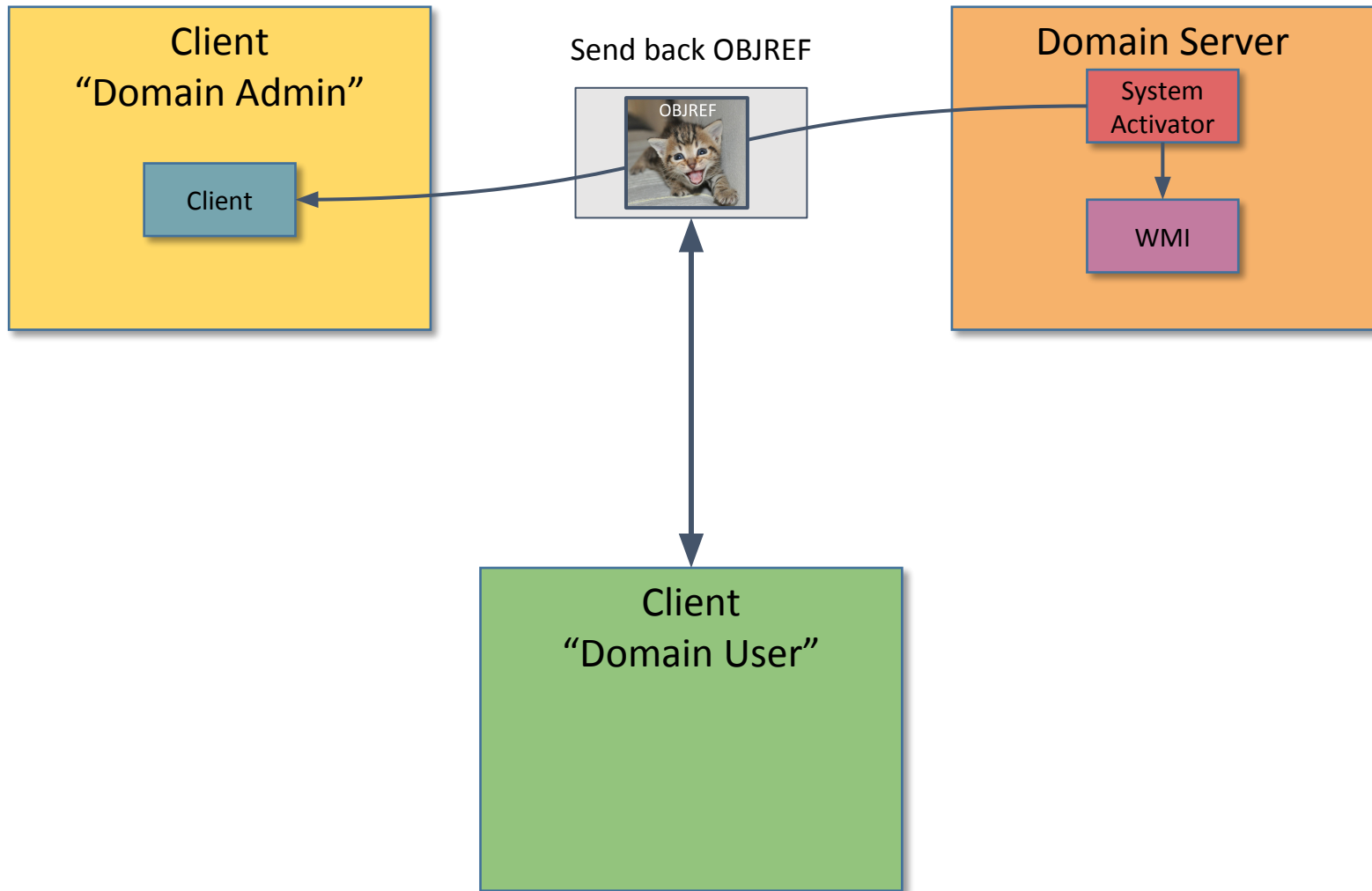
MEOW-Jacking!



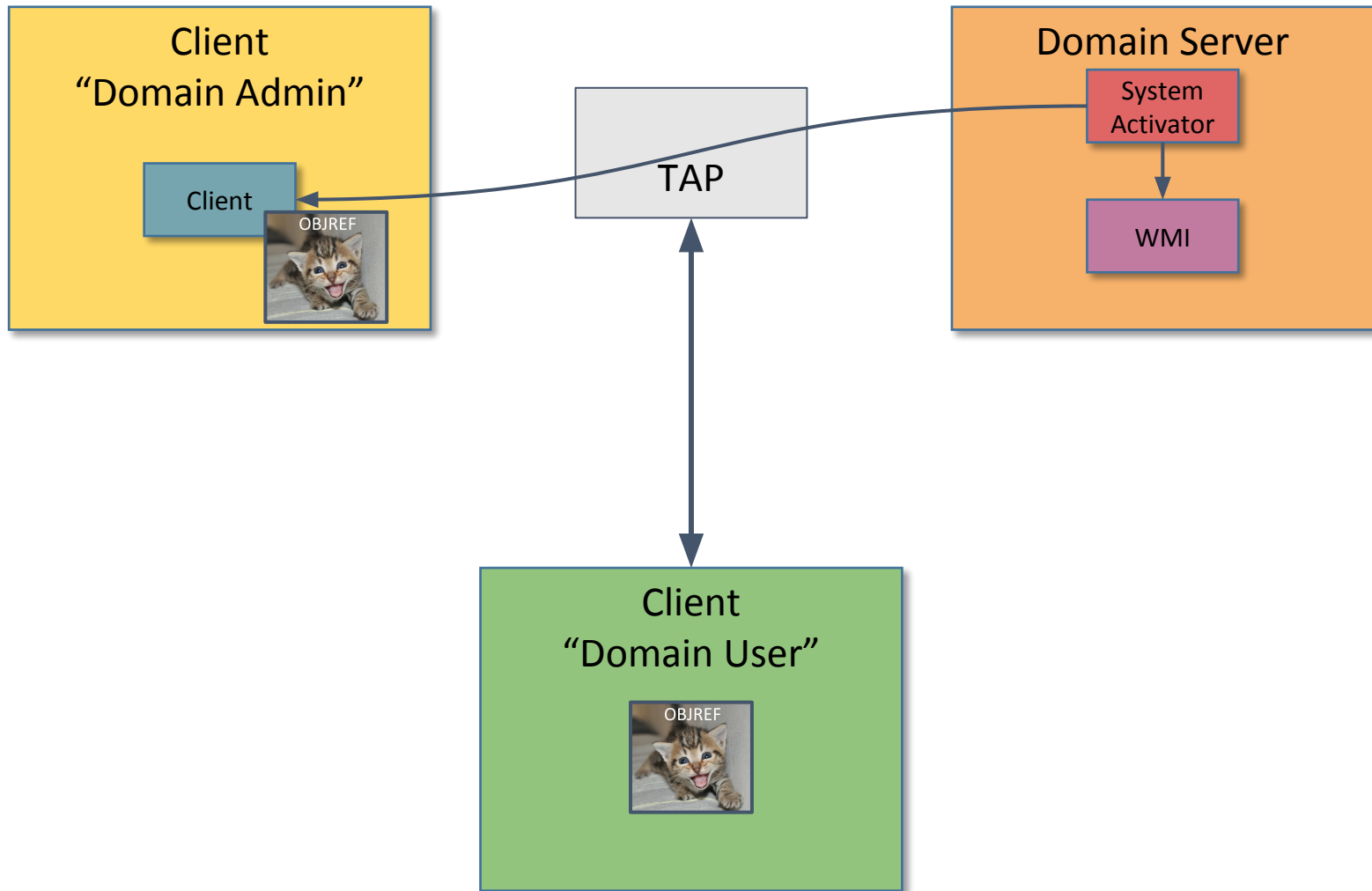
MEOW-Jacking!



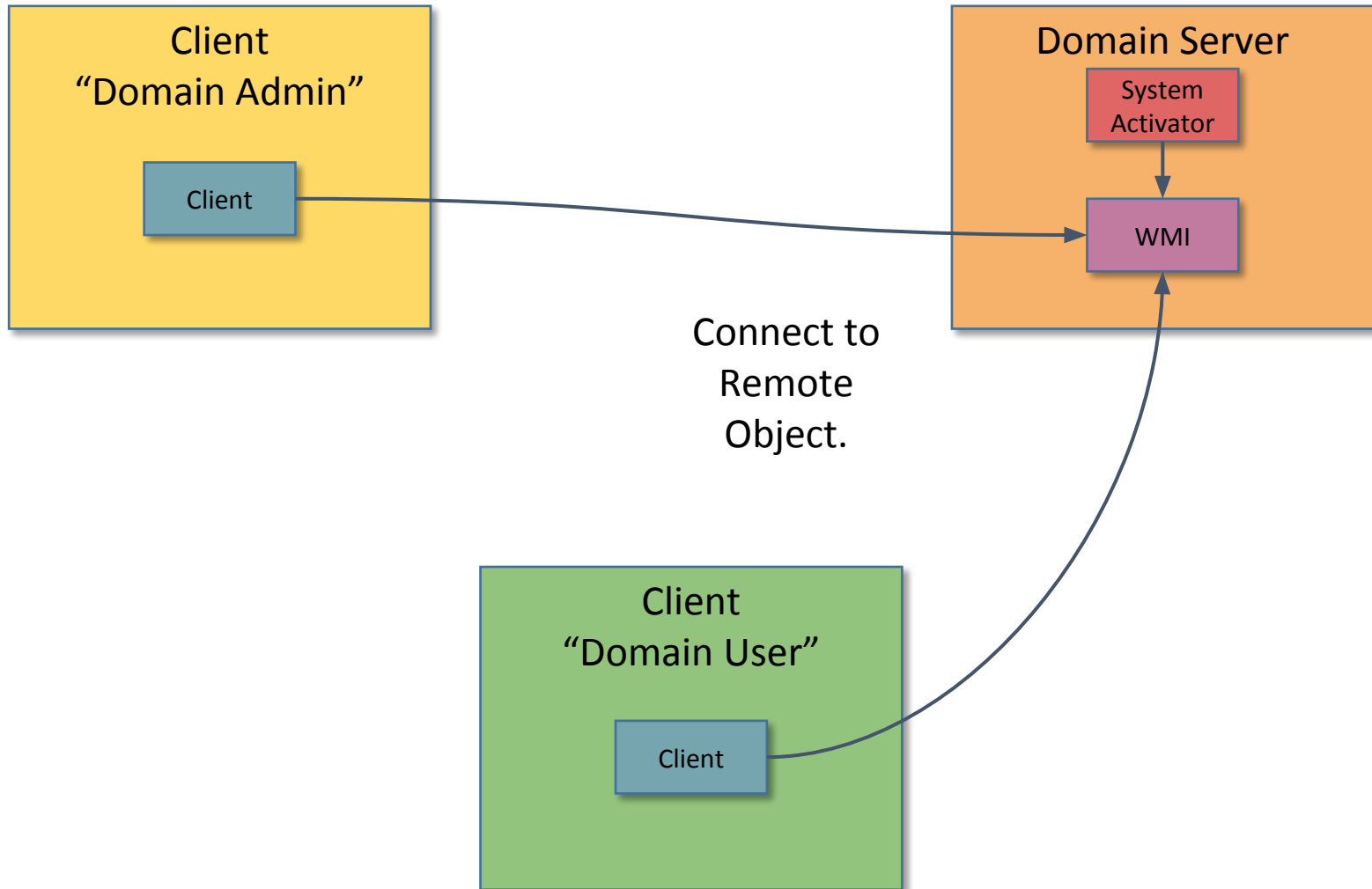
MEOW-Jacking!



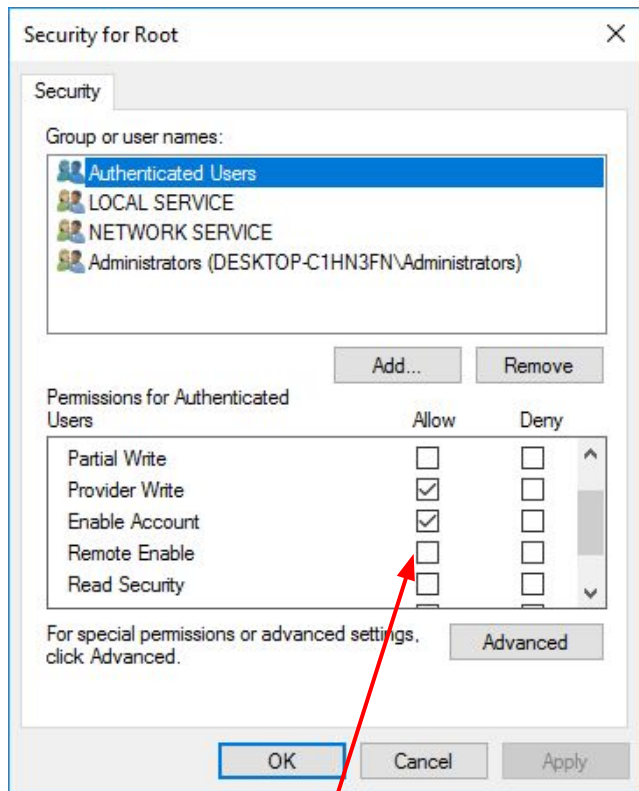
MEOW-Jacking!



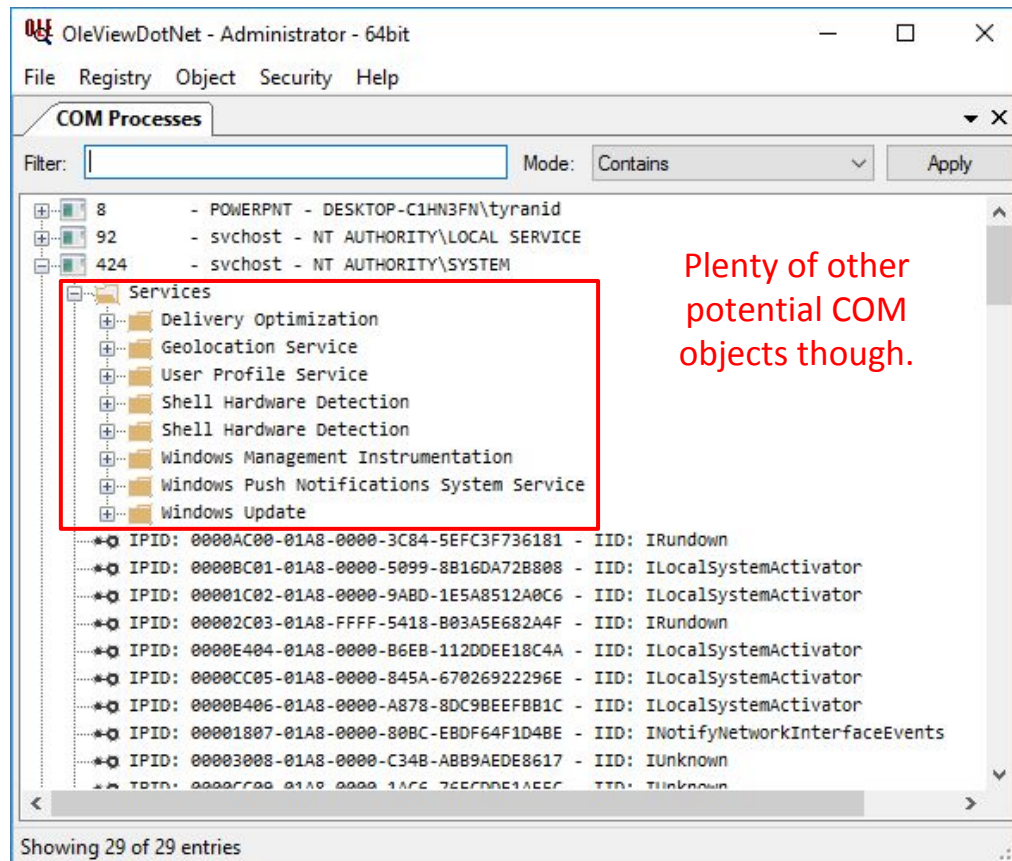
MEOW-Jacking!



Not as Bad as it Could Be



User's not allowed to access WMI remotely.





DEMO TIME



Thanks for Listening
Any Questions?