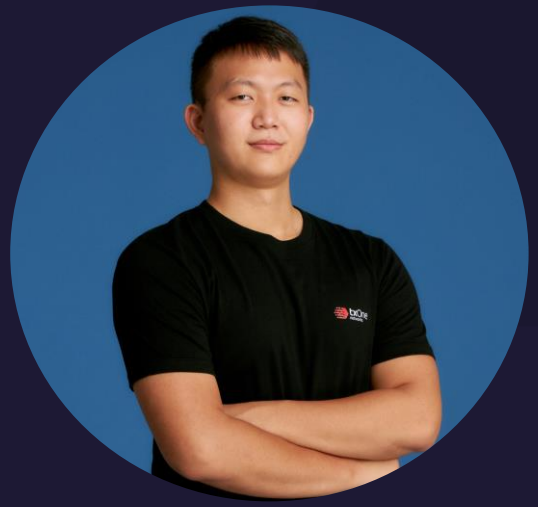
注意看，這些Windows的Potatoes太狠了!
解析5種基於 MS-RPCE 的攻擊手法

Hank Chen and Sheng-Hao Ma

PSIRT and Threat Research at TXOne Networks

May 11, 2023 | CYBERSEC2023

Who Are We?



Hank Chen

Threat Researcher
PSIRT and Threat Research

- Spoke at BlackHat USA, FIRST, HITCON, VXCON, and ThreatCon
- Instructor of Ministry of National Defense
- Teaching assistant of Cryptography and Information Security Course in Taiwan NTHU and CCoE Taiwan
- Member of CTF team 10sec and TSJ

Sheng-Hao Ma

Threat Researcher
PSIRT and Threat Research

- Spoke at Black Hat, DEFCON, HITB, VXCON, HITCON, ROOTCON, and CYBERSEC
- Instructor of CCoE Taiwan, Ministry of National Defense, Ministry of Education, and etc.
- The author of the popular security book "Windows APT Warfare: The Definitive Guide for Malware Researchers"

Outline

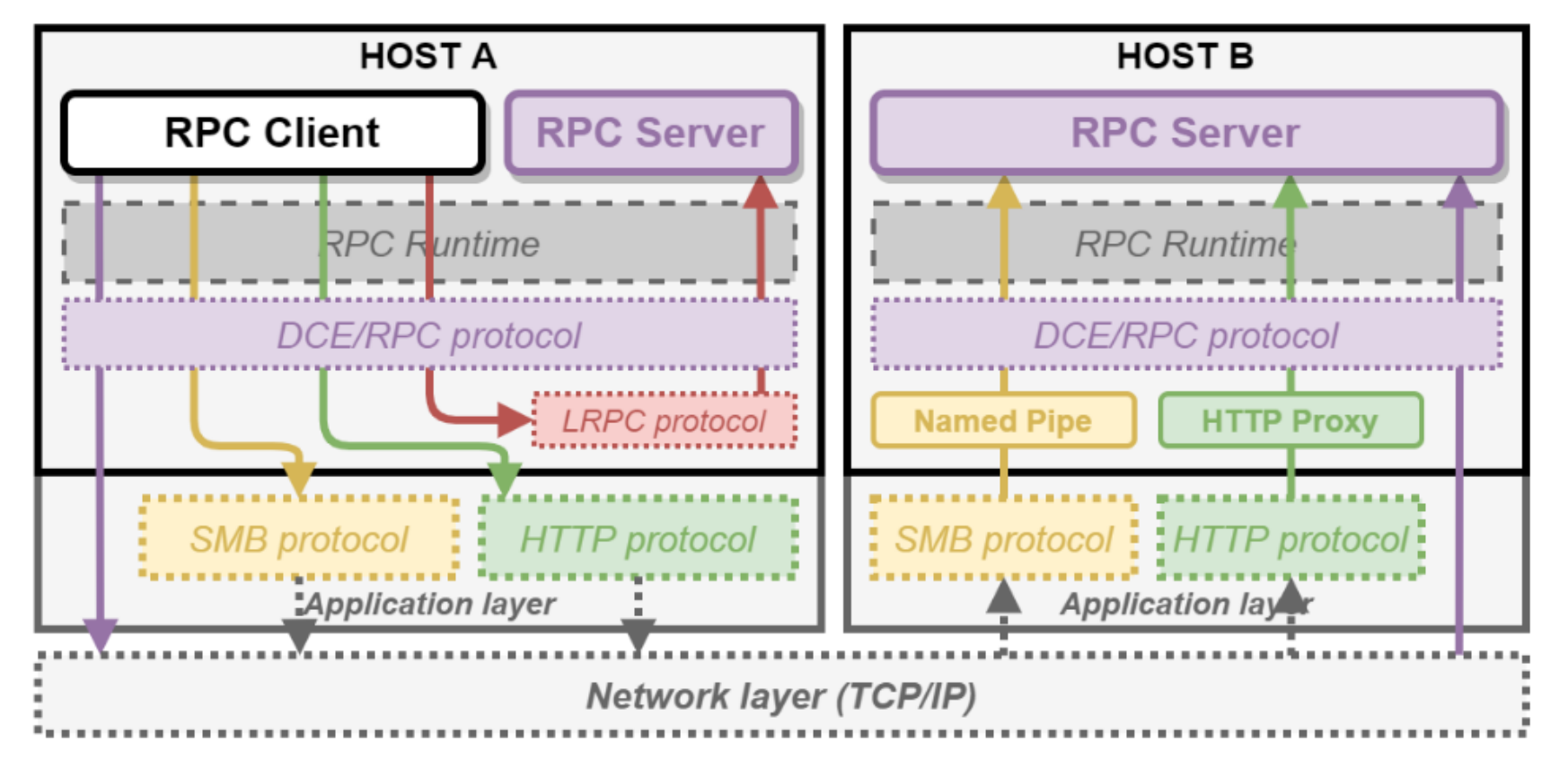
- The Knowledge about Potatoes
- How These Potatoes Work?
- How to Mitigate These Potatoes Attack?

The Knowledge about Potatoes

The Knowledge about Potatoes

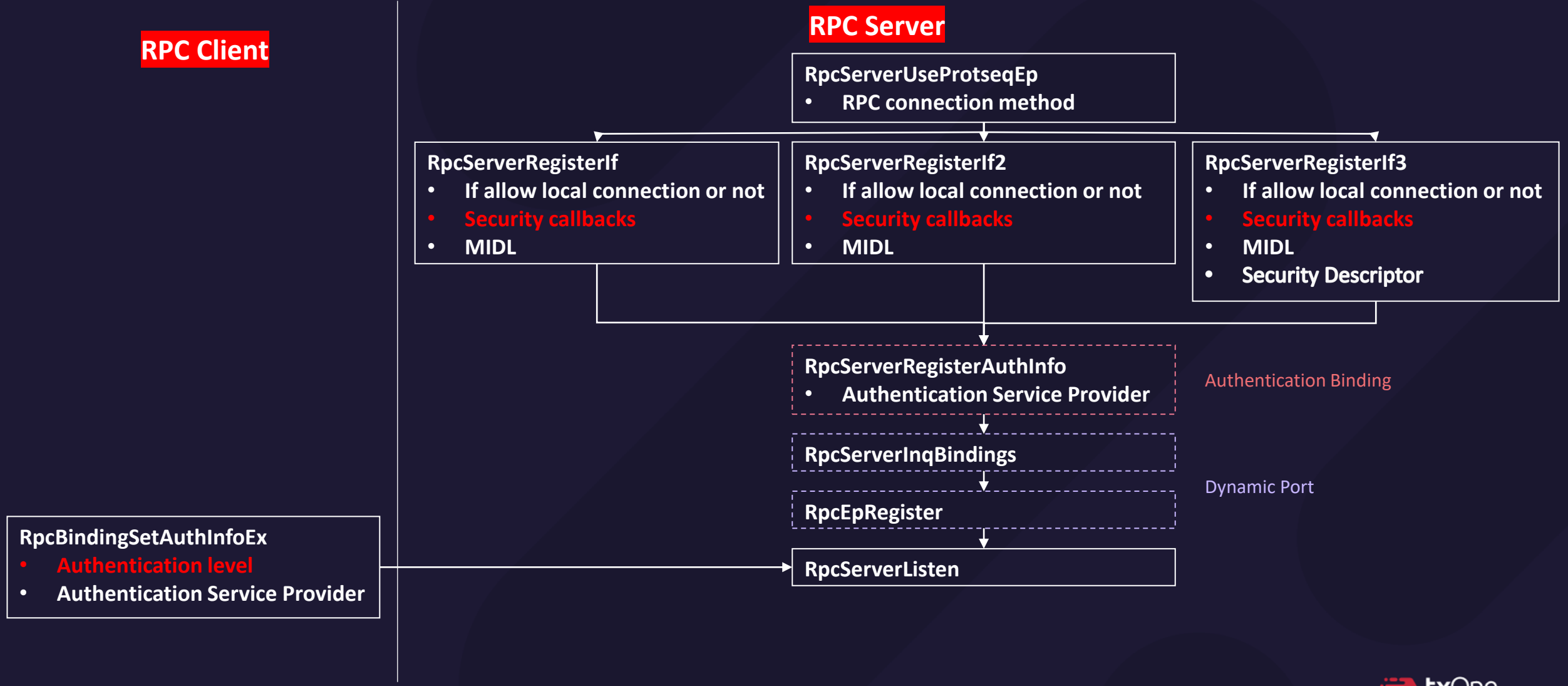
- RPC
- DCOM
- Access Token
- Service Account

How RPC works?



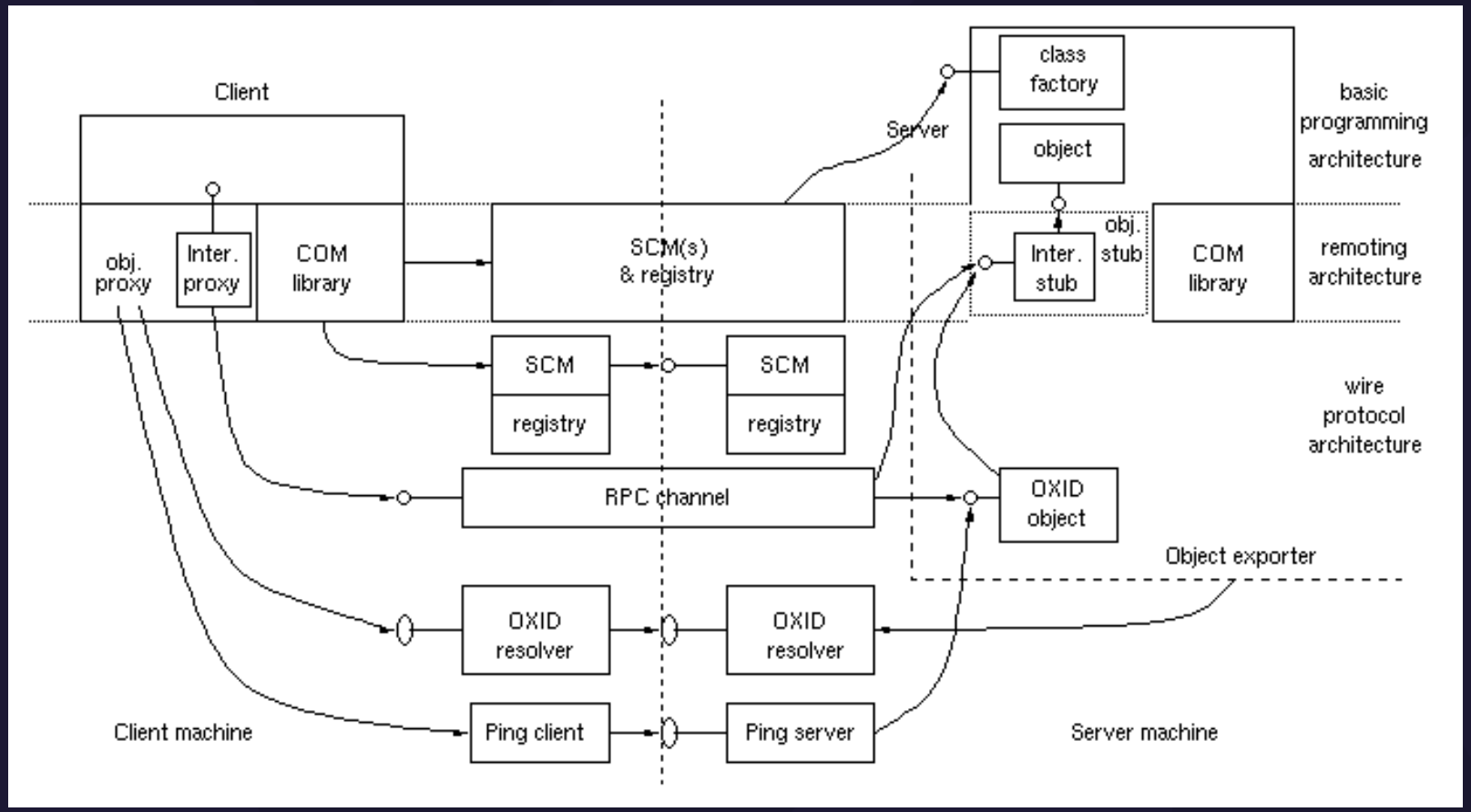
https://itm4n.github.io/from-rpcview-to-petitpotam/

Register RPC Server by Windows APIs



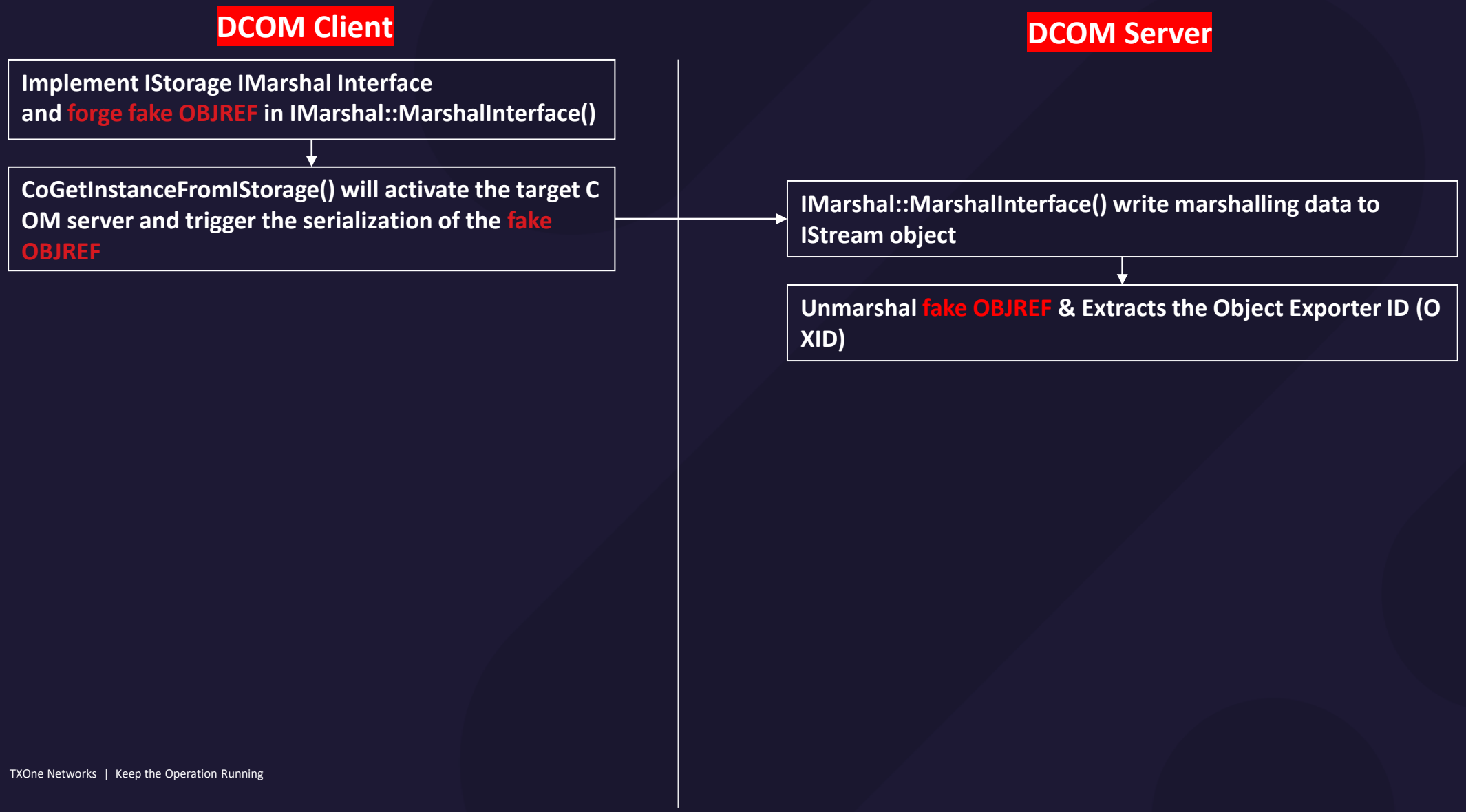
[MS-DCOM]: Distributed Component Object Model (DCOM) Remote Protocol

- Built on top of MS-RPCE, DCOM extends the Component Object Model (COM) over a network by providing facilities for creating and activating objects, and for managing object references, object lifetimes, and object interface queries.

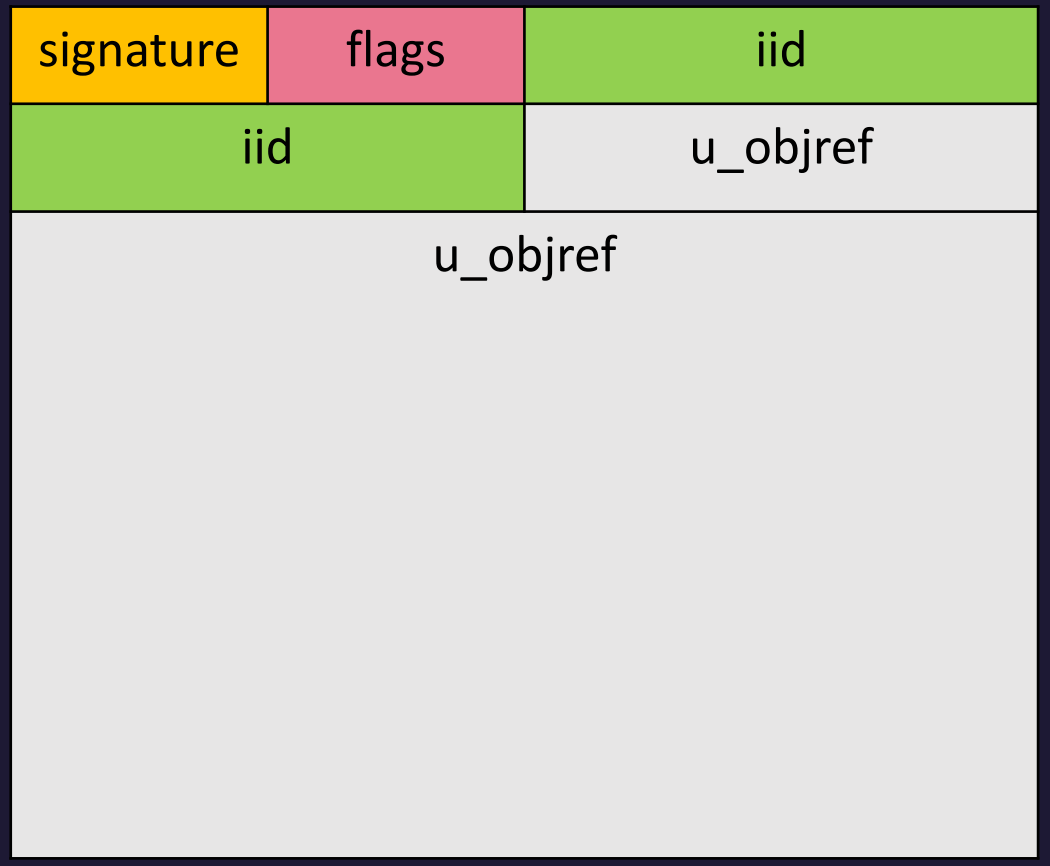


https://www.ime.usp.br/~reverbel/SOD-97/Textos/dcom_corba/Paper.html

Abuse DCOM NTLM Authentication with OBJREF



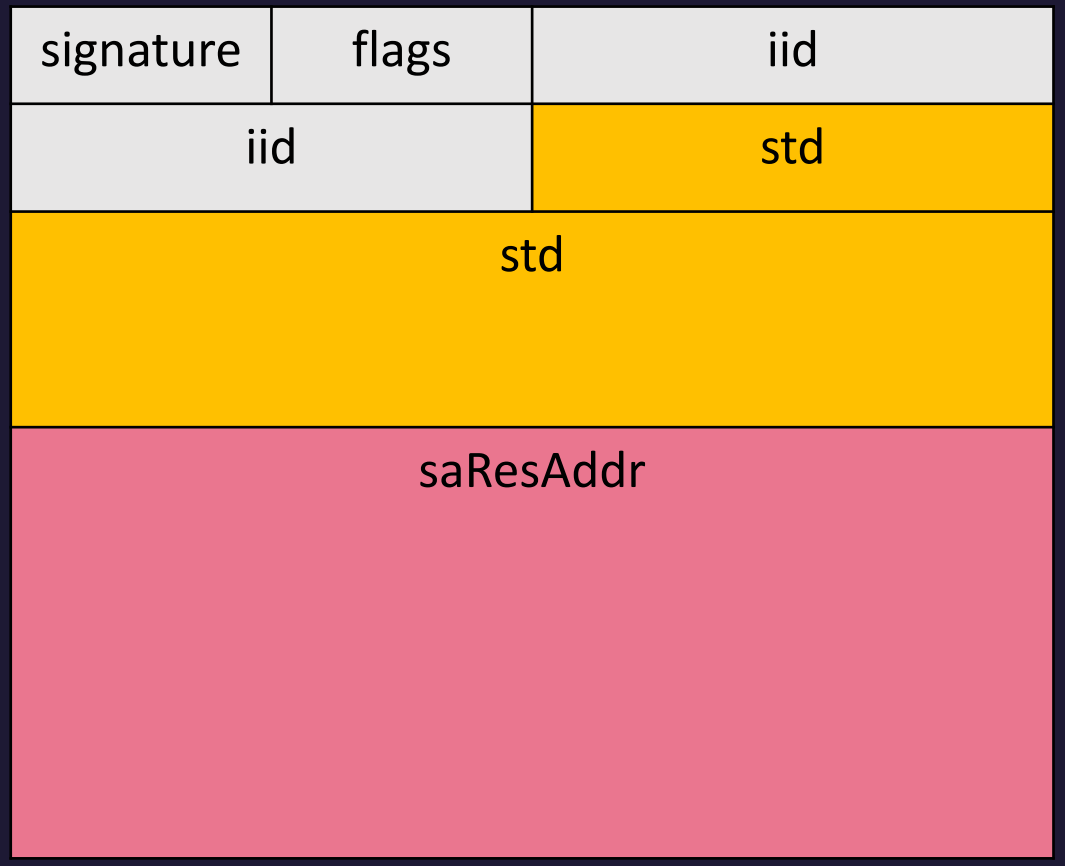
OBJREF



- signature (4 bytes): This MUST be set to the value 0x574f454d (MEOW).
- flags (4 bytes): This MUST be set to ONE of the following values.

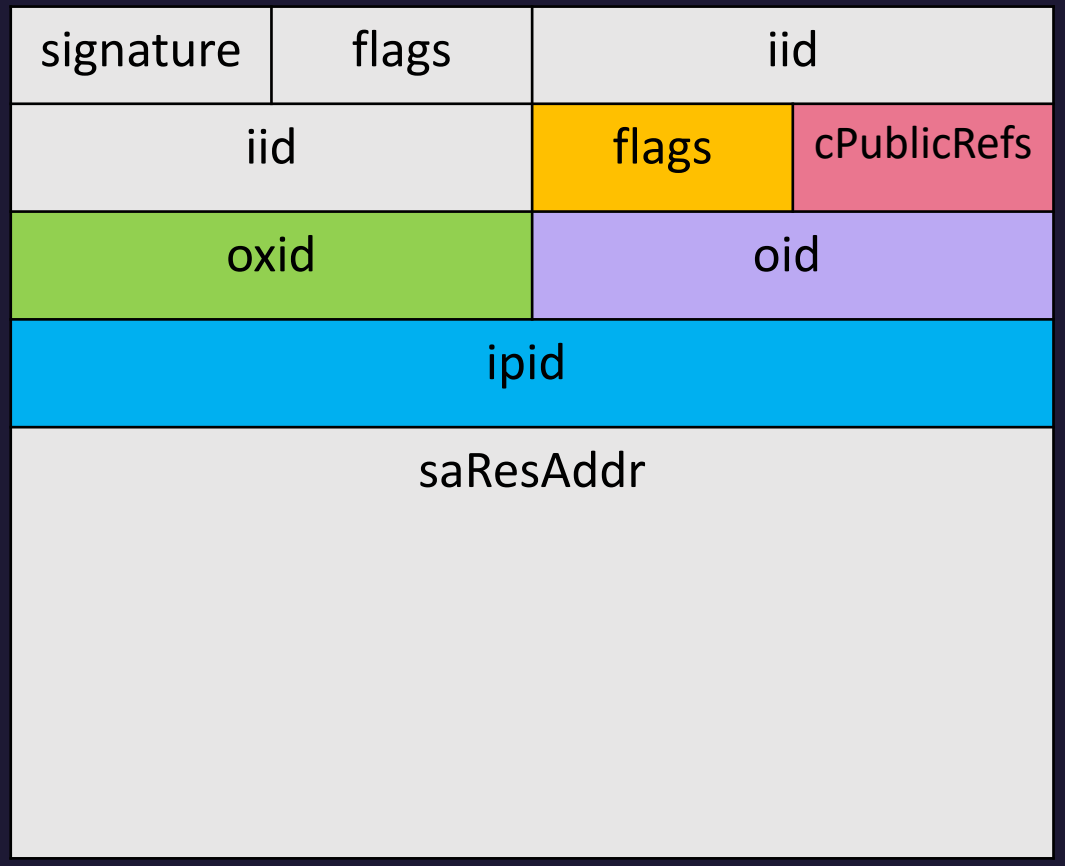
Value	Meaning
OBJREF_STANDARD 0x00000001	**u_objref** MUST contain an OBJREF_STANDARD.
OBJREF_HANDLER 0x00000002	**u_objref** MUST contain an OBJREF_HANDLER.
OBJREF_CUSTOM 0x00000004	**u_objref** MUST contain an OBJREF_CUSTOM.
OBJREF_EXTENDED 0x00000008	**u_objref** MUST contain an OBJREF_EXTENDED.

OBJREF_STANDARD



- std (40 bytes): This MUST be an STDOBJREF.
- saResAddr (variable): A DUALSTRINGARRAY that MUST contain the network and security bindings for the object resolver service on the server.

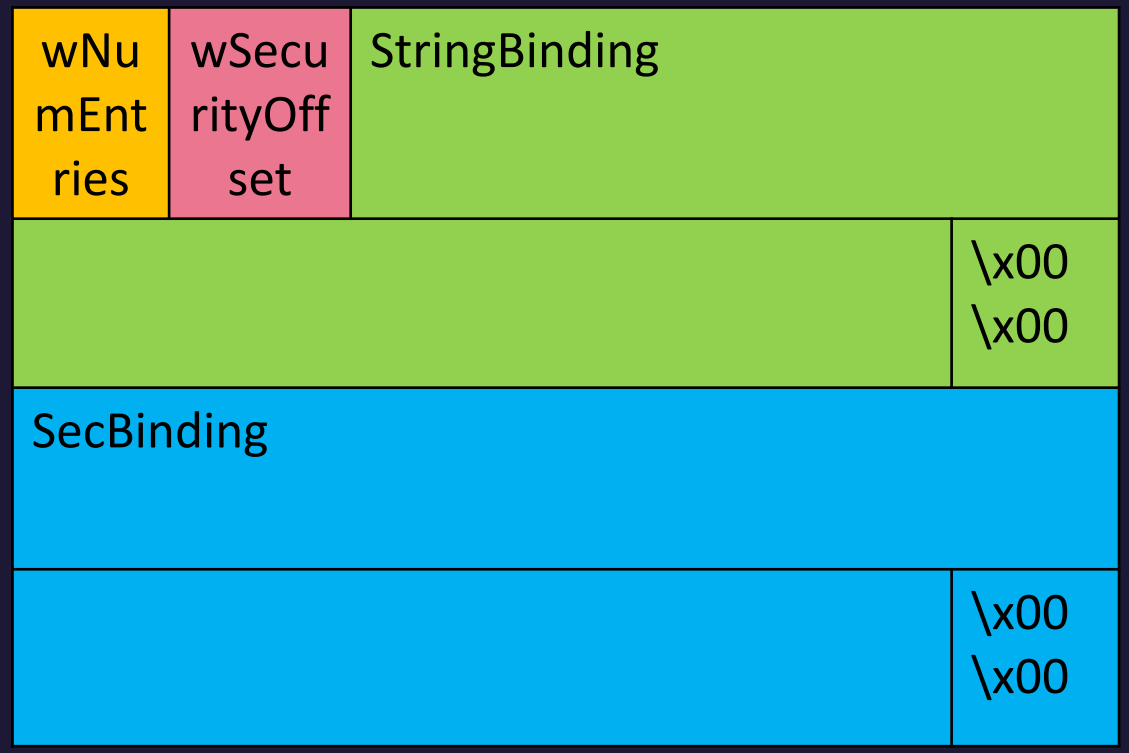
STDOBJREF



- flags (4 bytes): This can be one of the following values. Any other value MUST be ignored by the client.
- cPublicRefs (4 bytes): The number of public references on the server object, which MUST be released later.
- oxid (8 bytes): This MUST be an OXID identifying the object exporter that contains the object.
- oid (8 bytes): This MUST be an OID identifying the object.
- ipid (16 bytes): This MUST be an IPID identifying a specific interface on the object.

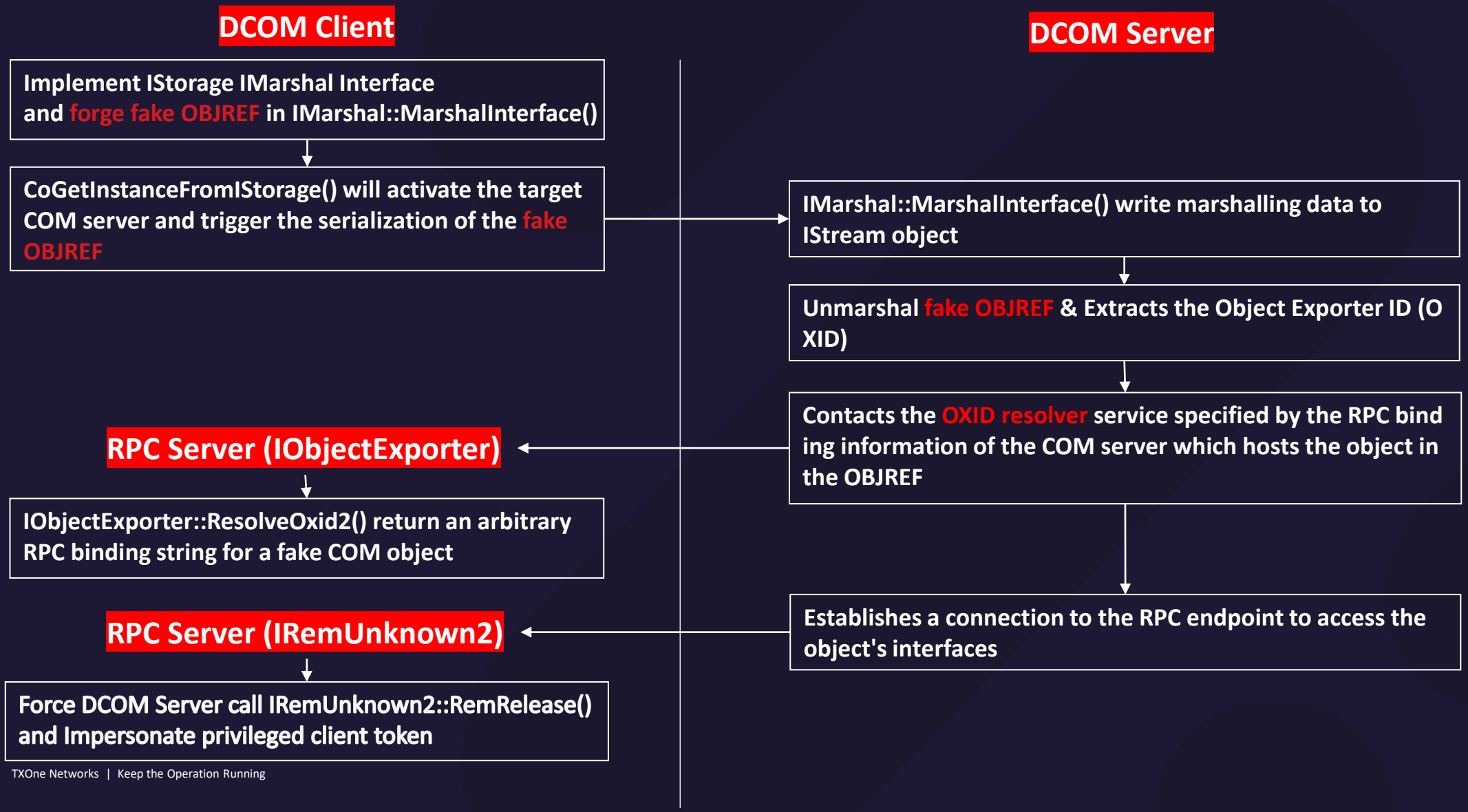
OBJREF_STANDARD

DUALSTRINGARRAY

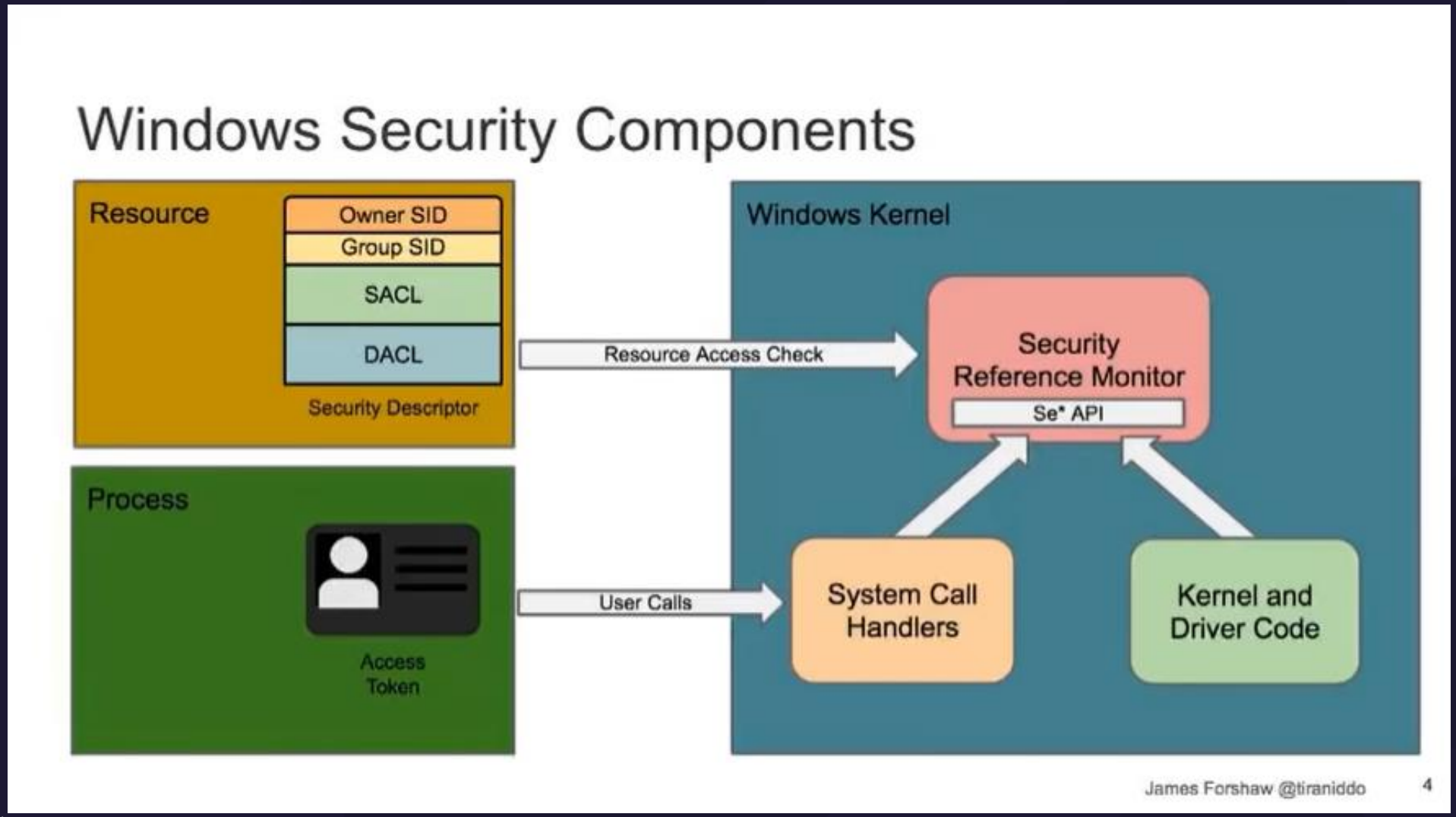


- wNumEntries (2 bytes): The number of unsigned shorts from the first entry in the StringBinding array to the end of the buffer.
- wSecurityOffset (2 bytes): The number of unsigned shorts from the first entry in the StringBinding array to the first entry in the SecBinding array.
- StringBinding (variable): An array of one or more STRINGBINDING structures that SHOULD be ordered in decreasing order of preference by the object server.
- SecBinding (variable): An array of one or more SECURITYBINDING structures that SHOULD be ordered in decreasing order of preference by the object server.

Abuse DCOM NTLM Authentication with OBJREF



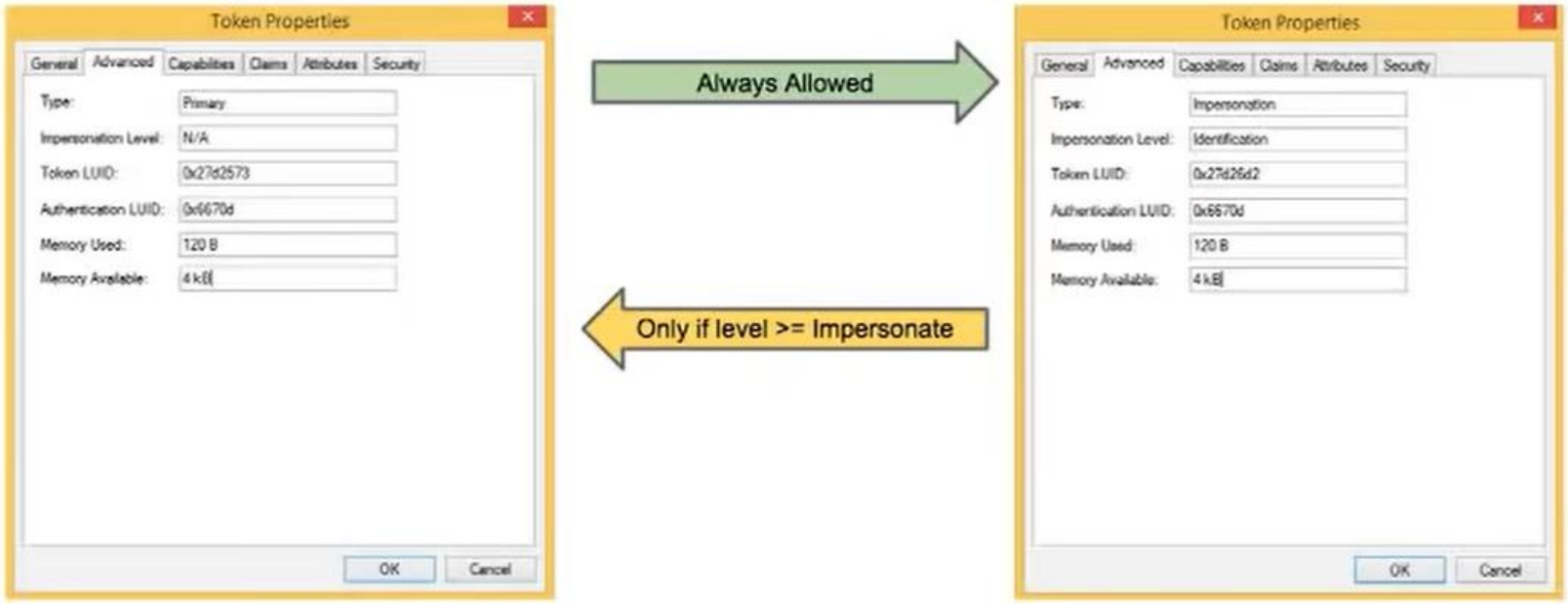
James Forshaw - Social Engineering The Windows Kernel: Finding And Exploiting Token Handling Vulnerabilities



James Forshaw - Social Engineering The Windows Kernel: Finding And Exploiting Token Handling Vulnerabilities

Token Duplication

Can duplicate token's between types if have TOKEN_DUPLICATE access on handle
Duplicated Token can be referenced with any set of access rights



Primary

Impersonation

James Forshaw @tiraniddo

15

SSPI Function from Lsass

Function	Description
AcceptSecurityContext (General)	Used by a server to create a *security context* based on an opaque message received from a client.
ApplyControlToken	Applies a supplemental security message to an existing security context.
CompleteAuthToken	Completes an authentication token. This function is used by protocols, such as DCE, that need to revise the security information after the transport application has updated some message parameters.
DeleteSecurityContext	Frees a security context and associated resources.
FreeContextBuffer	Frees a memory buffer allocated by a security package.
ImpersonateSecurityContext	Impersonates the security context to appear as the client to the system.
InitializeSecurityContext (General)	Used by a client to initiate a security context by generating an opaque message to be passed to a server.
QueryContextAttributes (General)	Enables a transport application to query a *security package* for certain *attributes* of a security *context*.
QuerySecurityContextToken	Obtains the *access token* for a client *security context* and uses it directly.
SetContextAttributes	Enables a transport application to set *attributes* of a security *context* for a *security package*. This function is supported only by the Schannel security package.
RevertSecurityContext	Allows a *security package* to discontinue the impersonation of the caller and restore its own *security context*.

https://learn.microsoft.com/en-us/windows/win32/secauthn/authentication-functions#context-management

James Forshaw - Social Engineering The Windows Kernel: Finding And Exploiting Token Handling Vulnerabilities

NTLM Negotiation

- LSASS exposes APIs to do network authentication
- Can get localhost NTLM authentication using redirected WebDAV
- Creates an impersonation level token even for a normal user

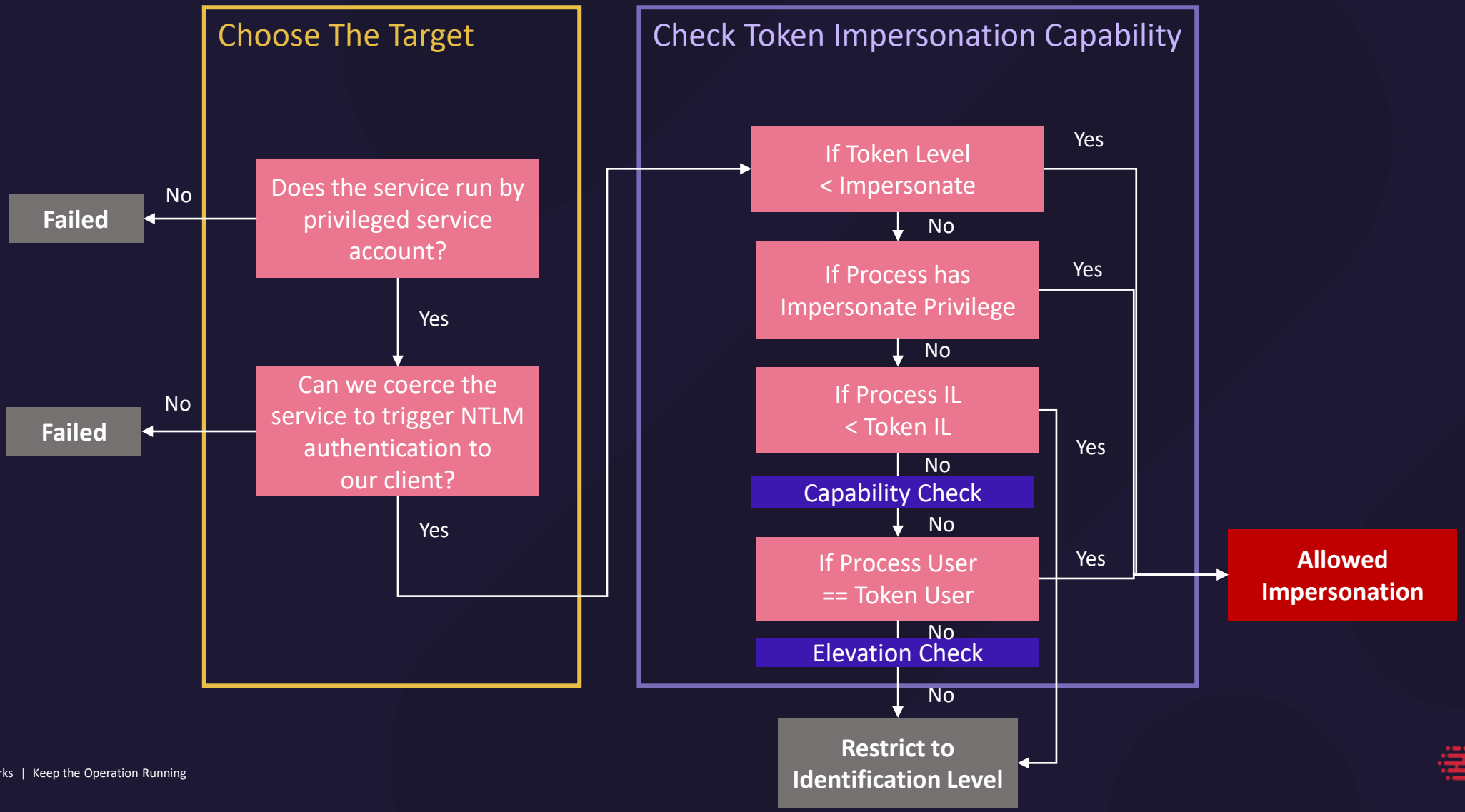
```
// Init a AV scan to \\localhost\fake\fake
BYTE* type1 = GetType1();
BYTE* challenge;

AcceptSecurityContext(hContext type1, &challenge);
BYTE* result = SetChallengeAndGetResult(challenge);
AcceptSecurityContext(hContext, result, ...);

QuerySecurityContextToken(hContext, &hToken);
```

James Forshaw @tiraniddo 26

Windows Service

- Windows Service Accounts
  - Windows Service must be run with a Service Account User
  - Service Account types
    - Local System
    - Local Service / Network Service Accounts
    - Managed Service & Virtual Accounts
- Windows Service Hardening (WSH)
  - Until Windows Server 2003/XP every service was run as SYSTEM
    - Compromised service == Compromised machine
  - Limited Service Accounts, Reduced Privileges, Write-Restricted Token, …

How Attackers Think …



Potatoes

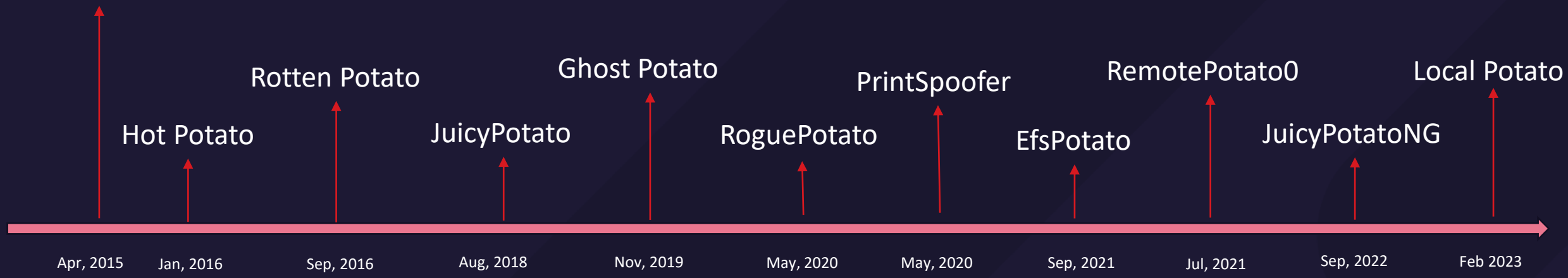
From a Windows Service Accounts to NT AUTHORITY\SYSTEM

A Brief History of Potatoes

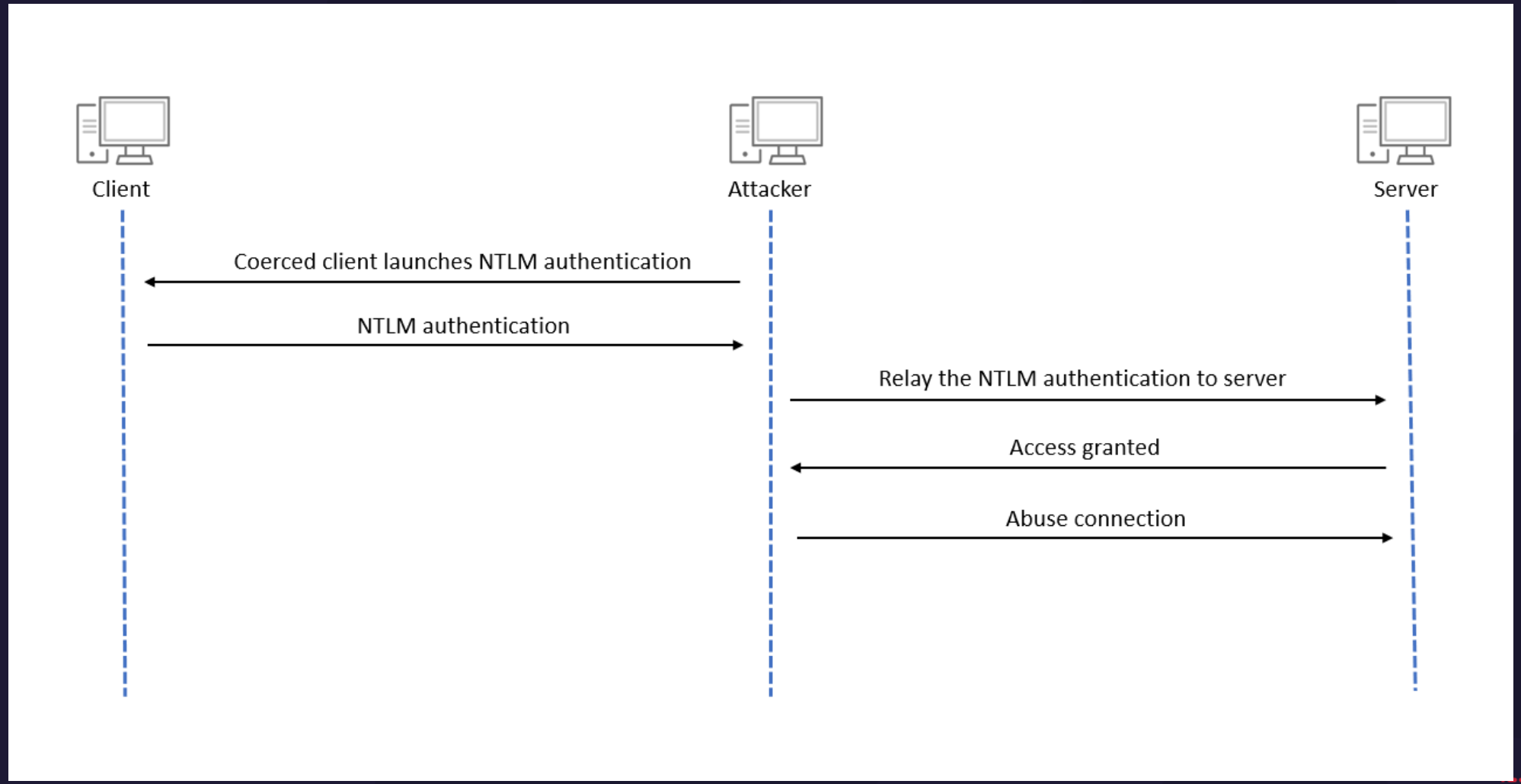
Issue 325: Windows: DCOM DCE/RPC Local NTLM Reflection Elevation of Privilege

Reported by forshaw@google.com on Fri, Apr 10, 2015, 2:42 AM GMT+8 **Project Member**

Windows: DCOM DCE/RPC Local NTLM Reflection Elevation of Privilege
Platform: Windows 8.1 Update (not tested on Windows 7, 10)
Class: Elevation of Privilege



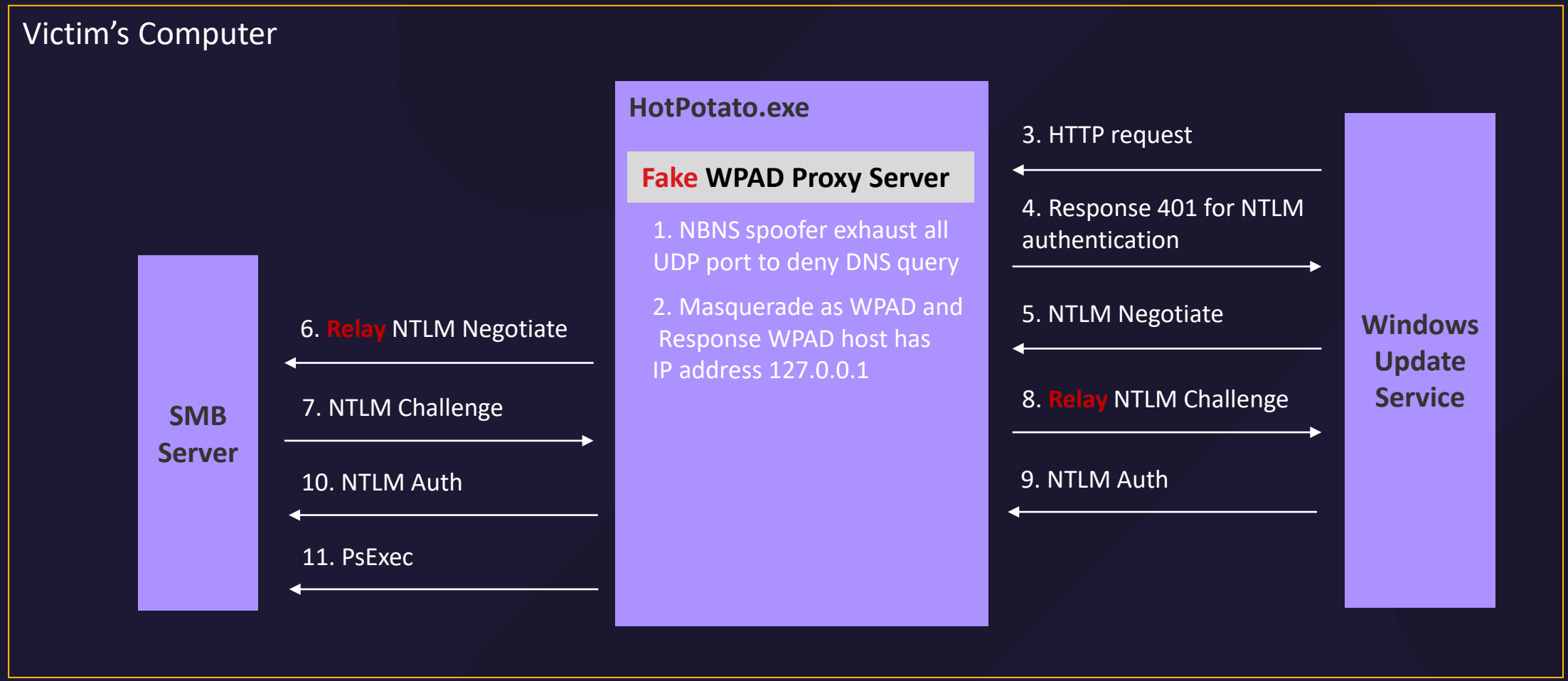
NTLM Relay 101



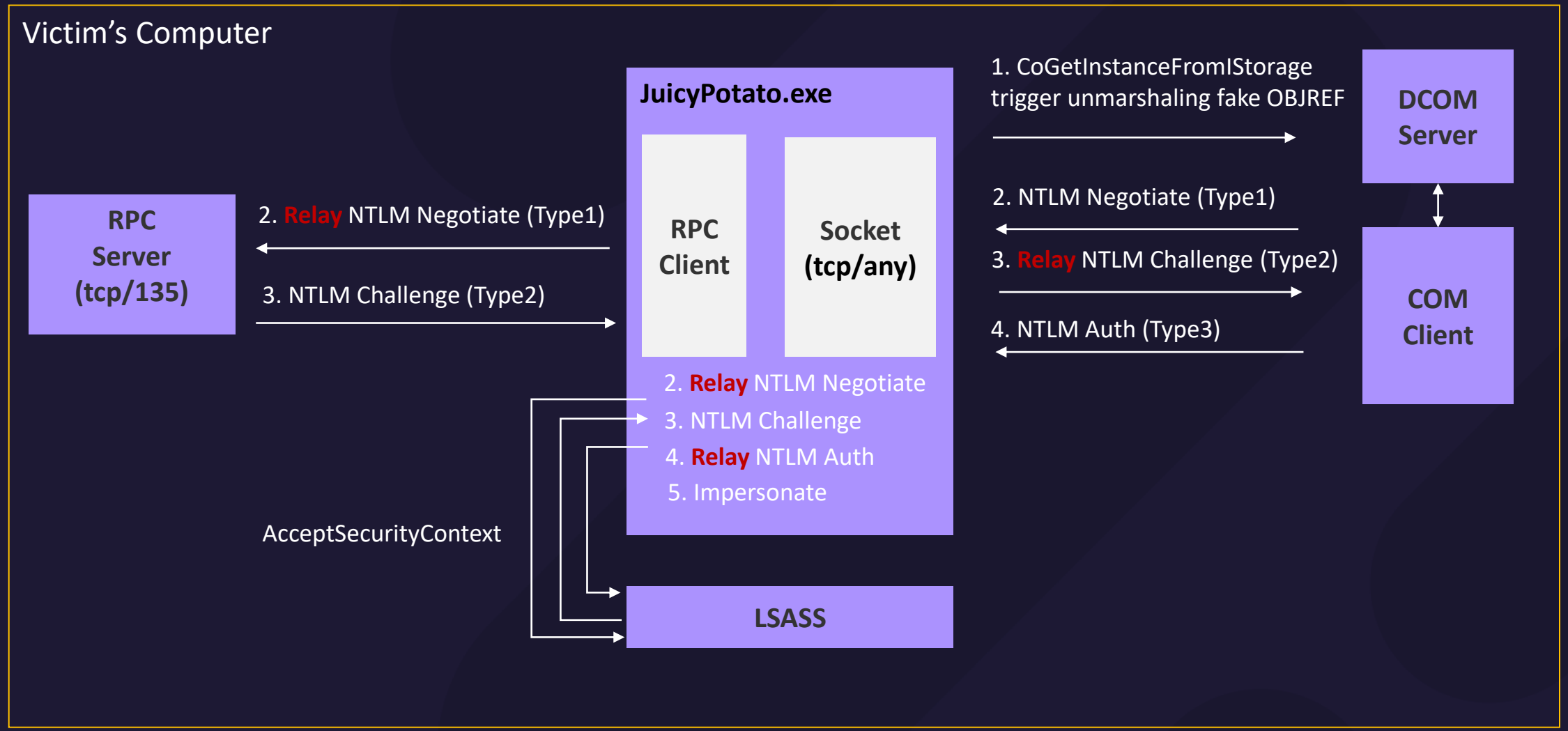
The Techniques Adopted by Potatoes

- Ghost Potato
  - NTLM Challenge Cache Timeout (CVE-2019-1384), ntlmrelayx
- Hot Potato
  - Windows Update Service <-> SMB
  - NBNS Spoofing, WPAD Poisoning, NTLM Relay
- Juicy Potato
  - DCOM <-> RPC
  - NTLM Relay
- Efs Potato — Impersonate
  - EFSRPC -> Named Pipe
  - NamedPipe Path Resolution, PetitPotam
- Rogue Potato — Impersonate
  - DCOM -> Named Pipe
  - External OXID Resolving, Fake Epmapper, NamedPipe Path Resolution
- RemotePotato0
  - DCOM -> HTTP <-> LDAP
  - External OXID Resolving, IRemUnknown2::RemRelease, ntlmrelayx
- JuicyPotatoNG — Impersonate
  - DCOM -> RPC
  - OBJREF Moniker Binding, SSPI Hook
- Local Potato
  - DCOM -> RPC <-> SMB
  - OBJREF Moniker Binding, SSPI Hook, Swap NTLM Context Handle (CVE-2023-21746)

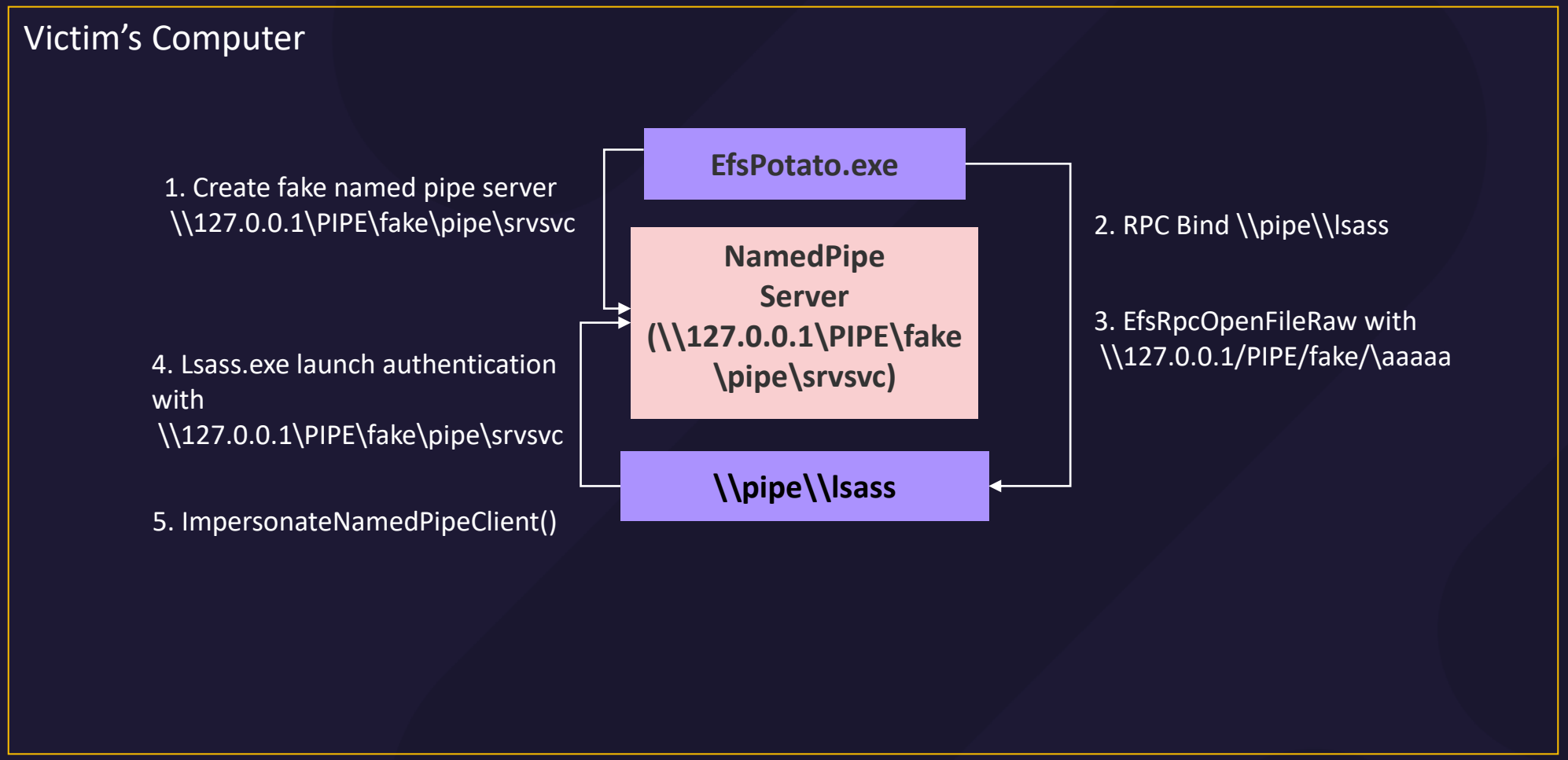
Hot Potato


Victim's Computer
HotPotato.exe
Fake WPAD Proxy Server
1. NBNS spoofer exhaust all UDP port to deny DNS query
2. Masquerade as WPAD and Response WPAD host has IP address 127.0.0.1
3. HTTP request
4. Response 401 for NTLM authentication
5. NTLM Negotiate
8. Relay NTLM Challenge
9. NTLM Auth
Windows Update Service
SMB Server
6. Relay NTLM Negotiate
7. NTLM Challenge
10. NTLM Auth
11. PsExec

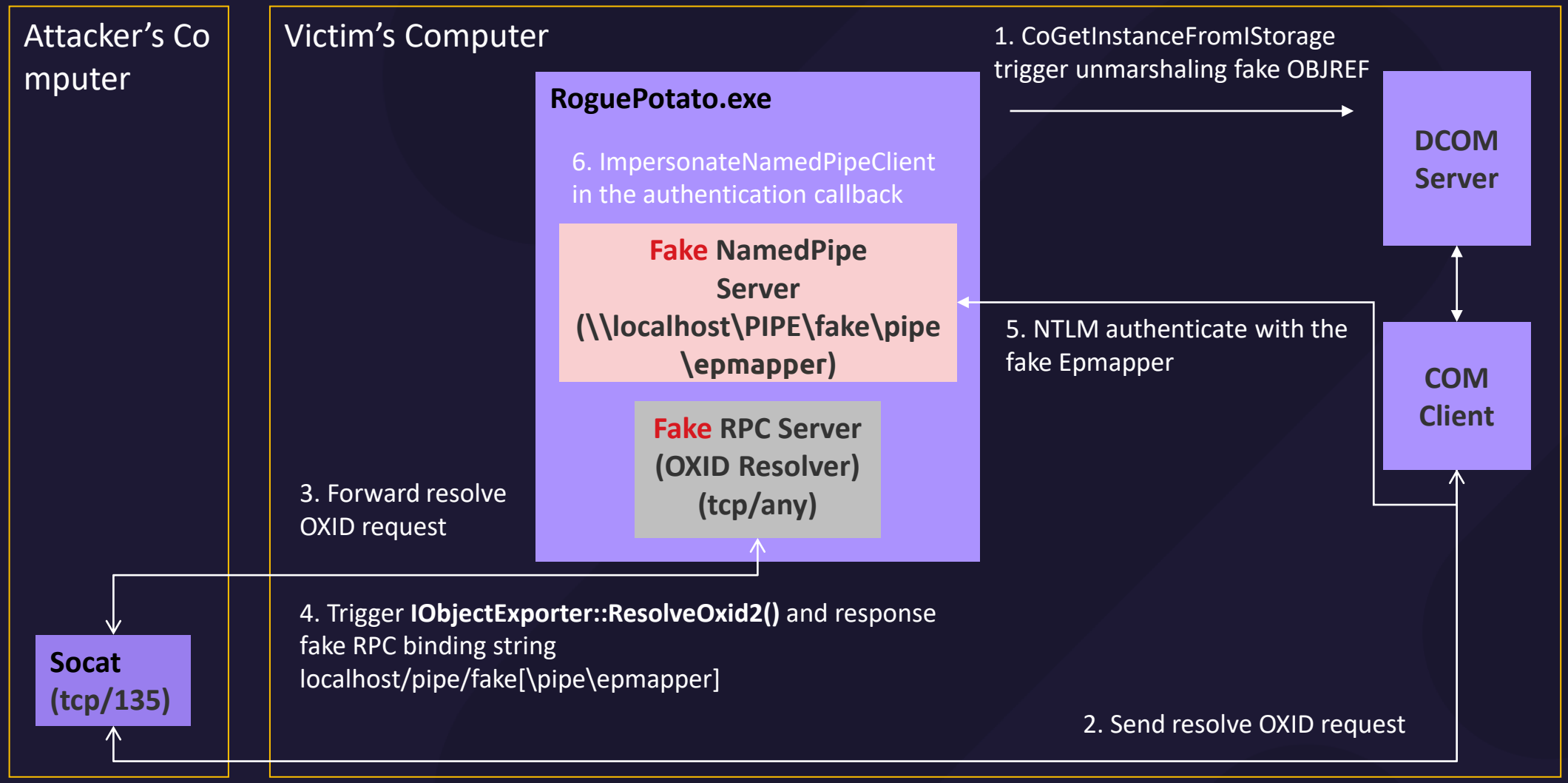
Juicy Potato



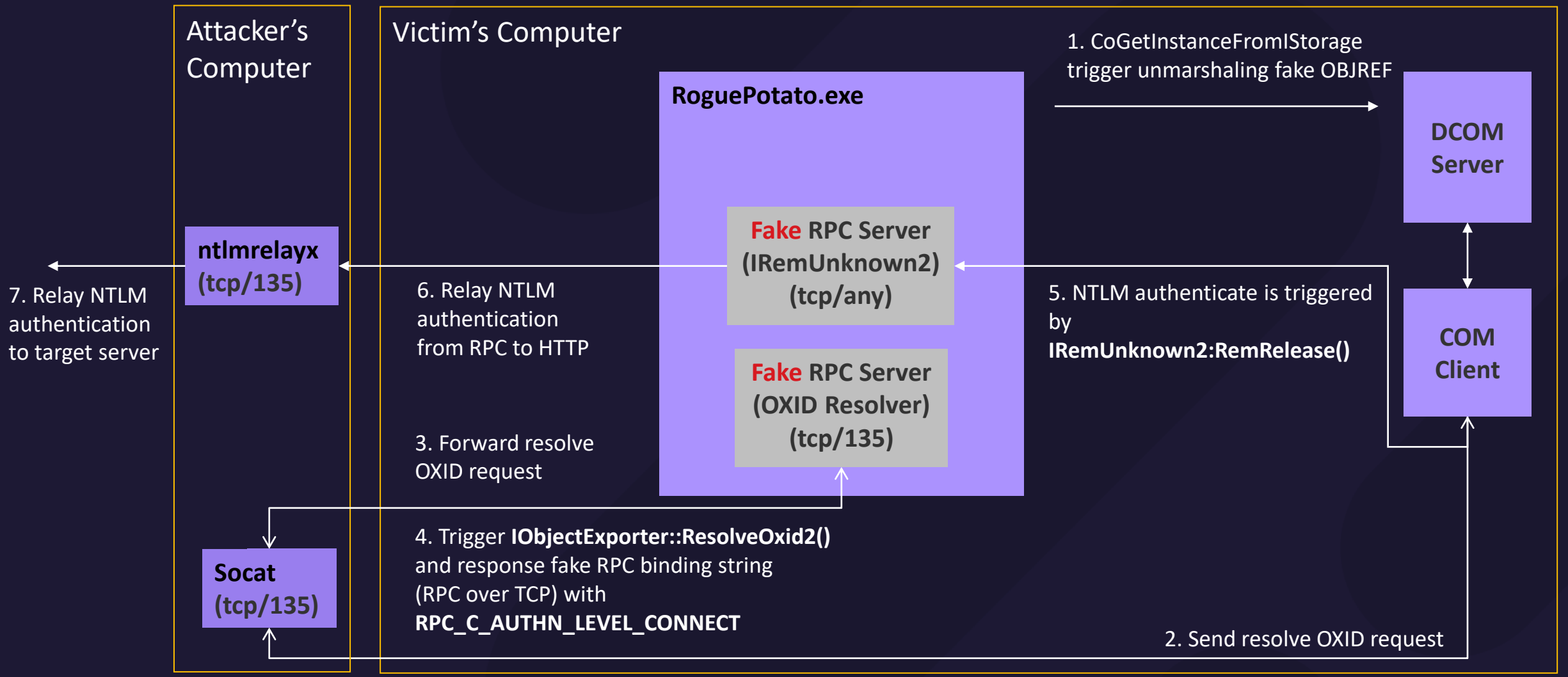
Efs Potato


Victim's Computer
EfsPotato.exe
1. Create fake named pipe server
\\127.0.0.1\PIPE\fake\pipe\srvsvc
2. RPC Bind \\pipe\\lsass
NamedPipe
Server
(\\127.0.0.1\PIPE\fake
\pipe\srvsvc)
3. EfsRpcOpenFileRaw with
\\127.0.0.1/PIPE/fake/\aaaaa
4. Lsass.exe launch authentication
with
\\127.0.0.1\PIPE\fake\pipe\srvsvc
\\pipe\\lsass
5. ImpersonateNamedPipeClient()

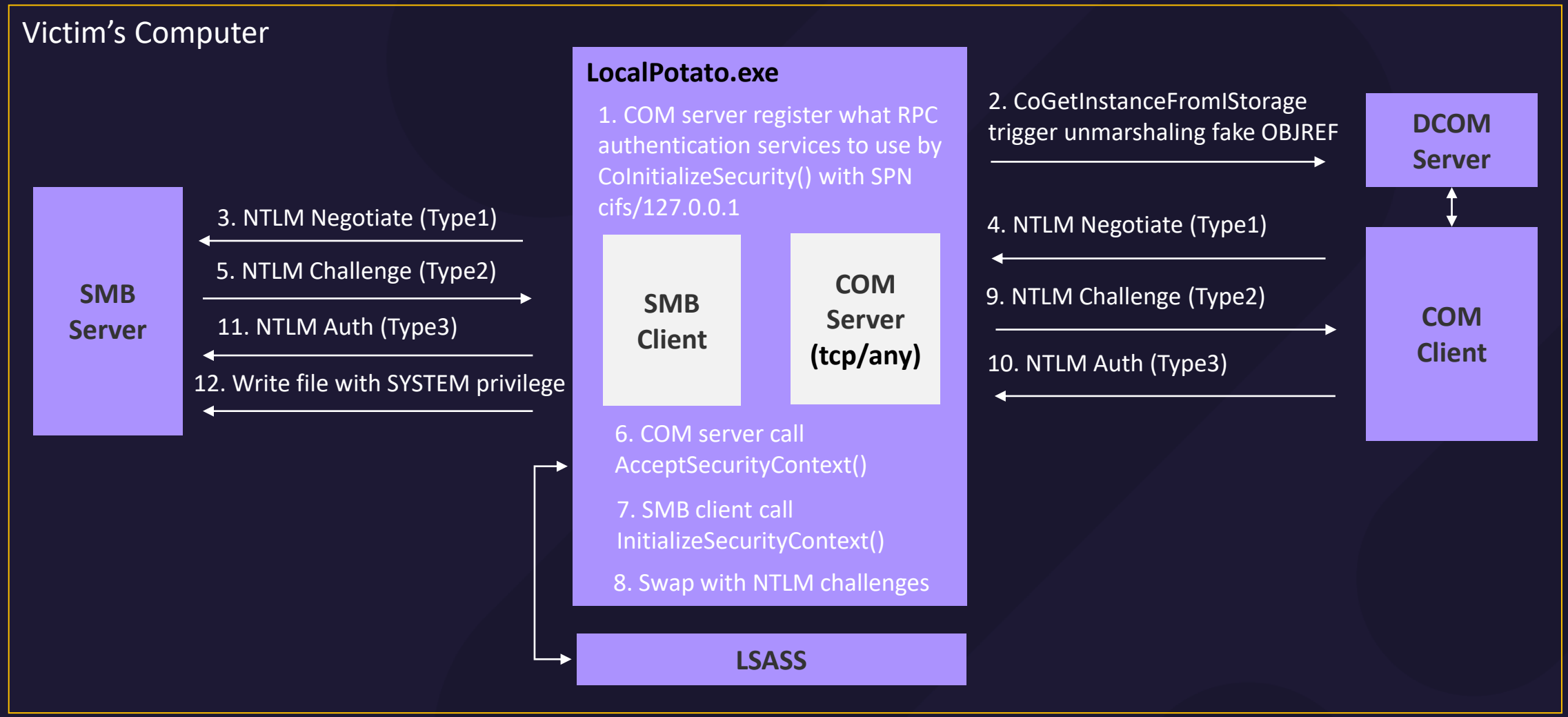
Rogue Potato



RemotePotato0



Local Potato



How to Mitigate These Potatoes Attack?

For Example: Yara Rule for RemotePotato0

```
rule SentinelOne_RemotePotato0_privesc {
   meta:
      author = "SentinelOne"
      description = "Detects RemotePotato0 binary"
      reference = "https://www.sentinelone.com/wp-content/uploads/relaying-potatoes-dce-rpc-ntlm-relay-eop"
   strings:
            $import1 = "CoGetInstanceFromIStorage"
      $istorage_clsid = "{00000306-0000-0000-c000-000000000046}" nocase wide ascii
      $meow_header = { 4d 45 4f 57 }
      $clsid1 = "{11111111-2222-3333-4444-555555555555}" nocase wide ascii
      $clsid2 = "{5167B42F-C111-47A1-ACC4-8EABE61B0B54}" nocase wide ascii
  condition:
       (uint16(0) == 0x5A4D) and $import1 and $istorage_clsid and $meow_header and 1 of ($clsid*)
}
```

https://www.sentinelone.com/labs/relaying-potatoes-another-unexpected-privilege-escalation-vulnerability-in-windows-rpc-protocol/

What does James Forshaw say?

Experiment Results

We now need to do our basic analysis of the results. Let's start with calculating the percentage of writable resources for each token type relative to the total number of resources. From my single experiment run I got the following table:

Token	Writable	Writable (WR)	Total
Control	99.83%	N/A	13171
Network Service	65.00%	0.00%	300
Local Service	62.89%	0.70%	574

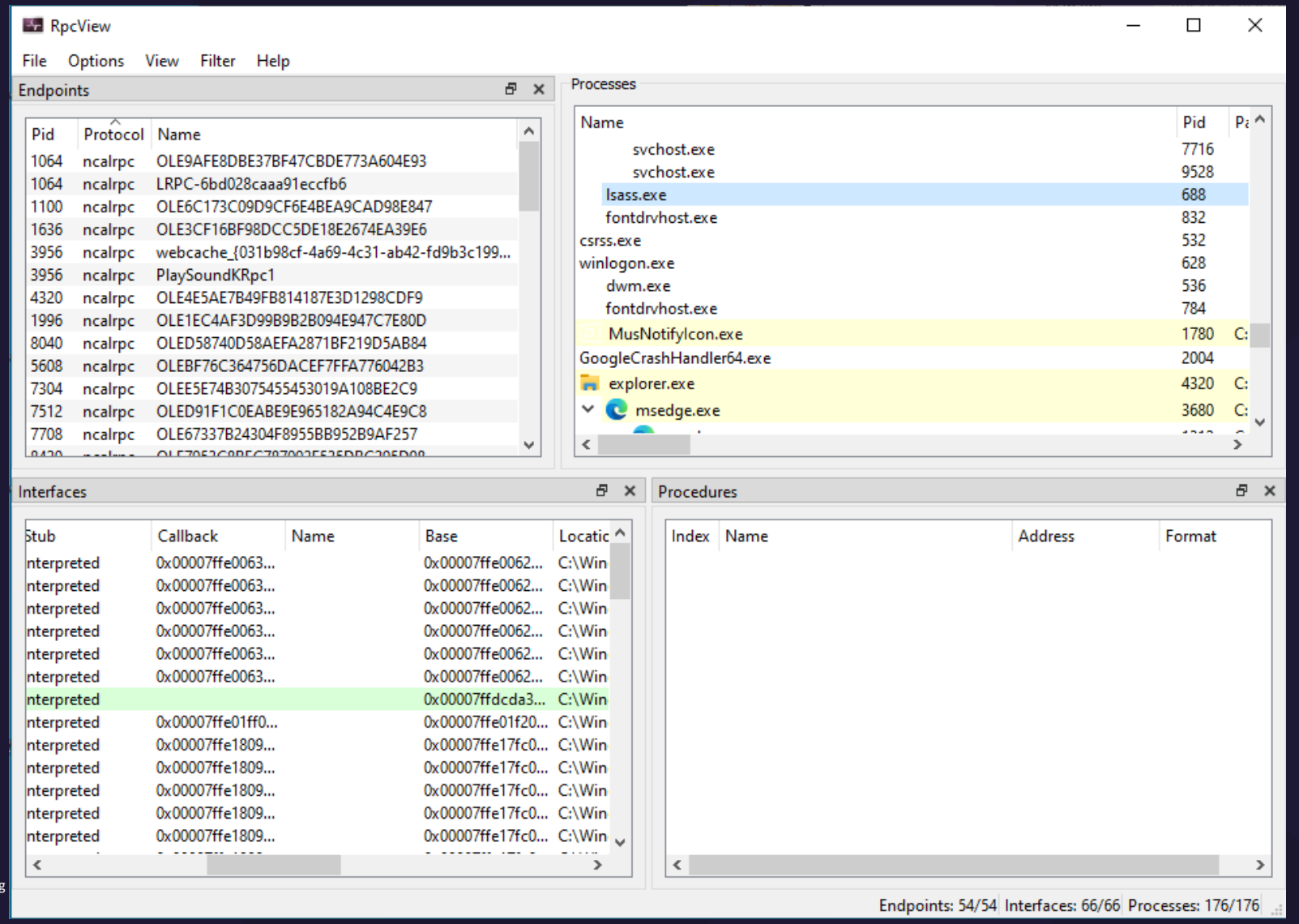
As we expected the control token had almost 100% of the owned resources writable by the user. However for the two service accounts both had over 60% of their owned resources writable when using an unrestricted token. That level is almost completely eliminated when using a WR token, there were no writable resources for NS and only 4 resources writable from LS, which was less than 1%. Those 4 resources were just Events, from a service perspective not very exciting though there were ACL'ed to everyone which is unusual.

https://www.tiraniddo.dev/2020/01/empirically-assessing-windows-service.html

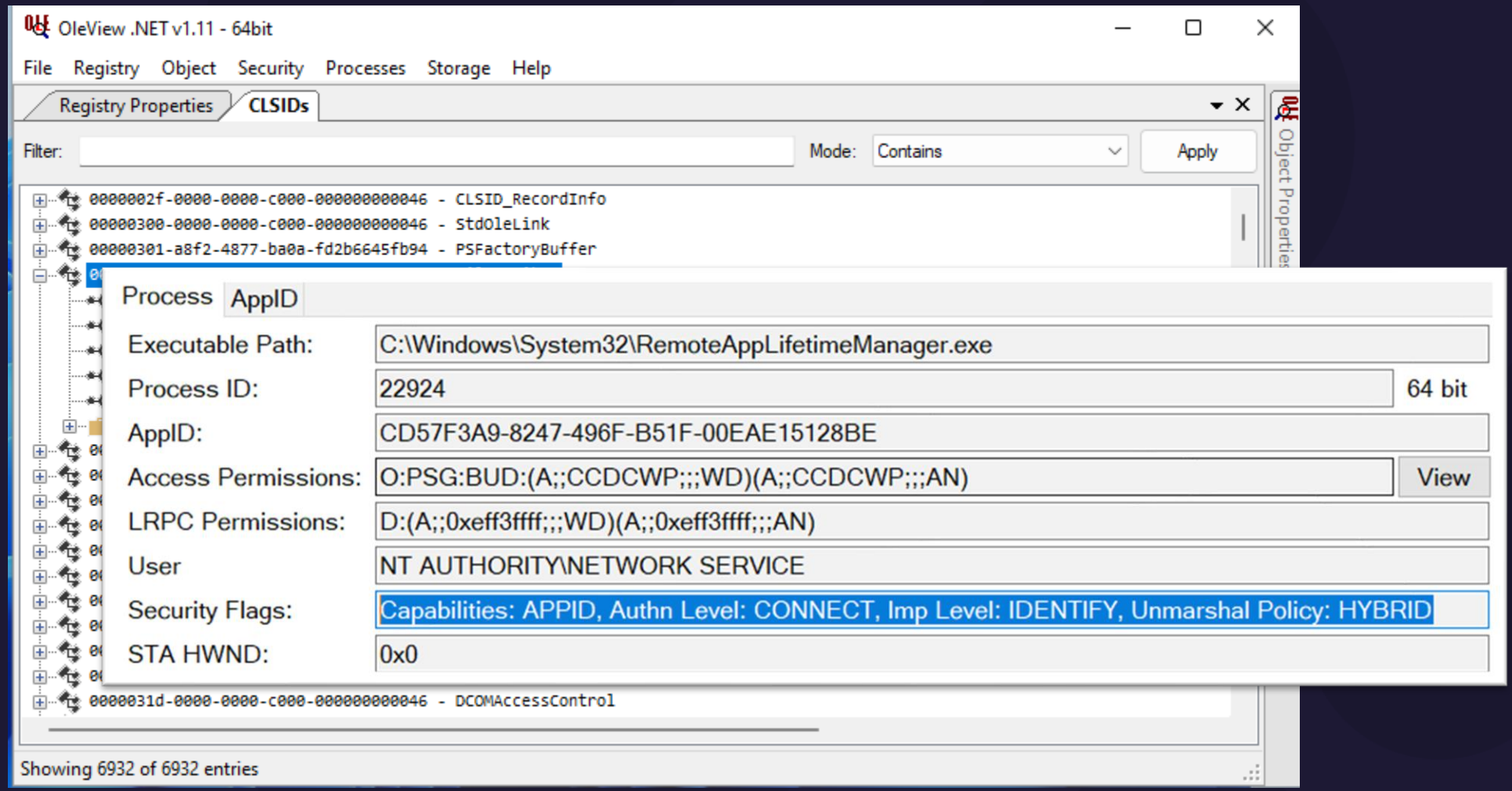
Mitigations from Antonio Cocomazzi

1. Change the sid type of the service to “WRITE RESTRICTED”
   - sc.exe sidtype SampleService restricted
2. Use virtual service accounts or create your own
   - sc.exe config SampleService obj= "NT SERVICE\SampleService“
3. Remove the impersonation privileges by specifying the only required privileges for the service (Least-Privilege)
   - sc.exe privs SampleService SeChangeNotifyPrivilege/SeCreateGlobalPrivilege

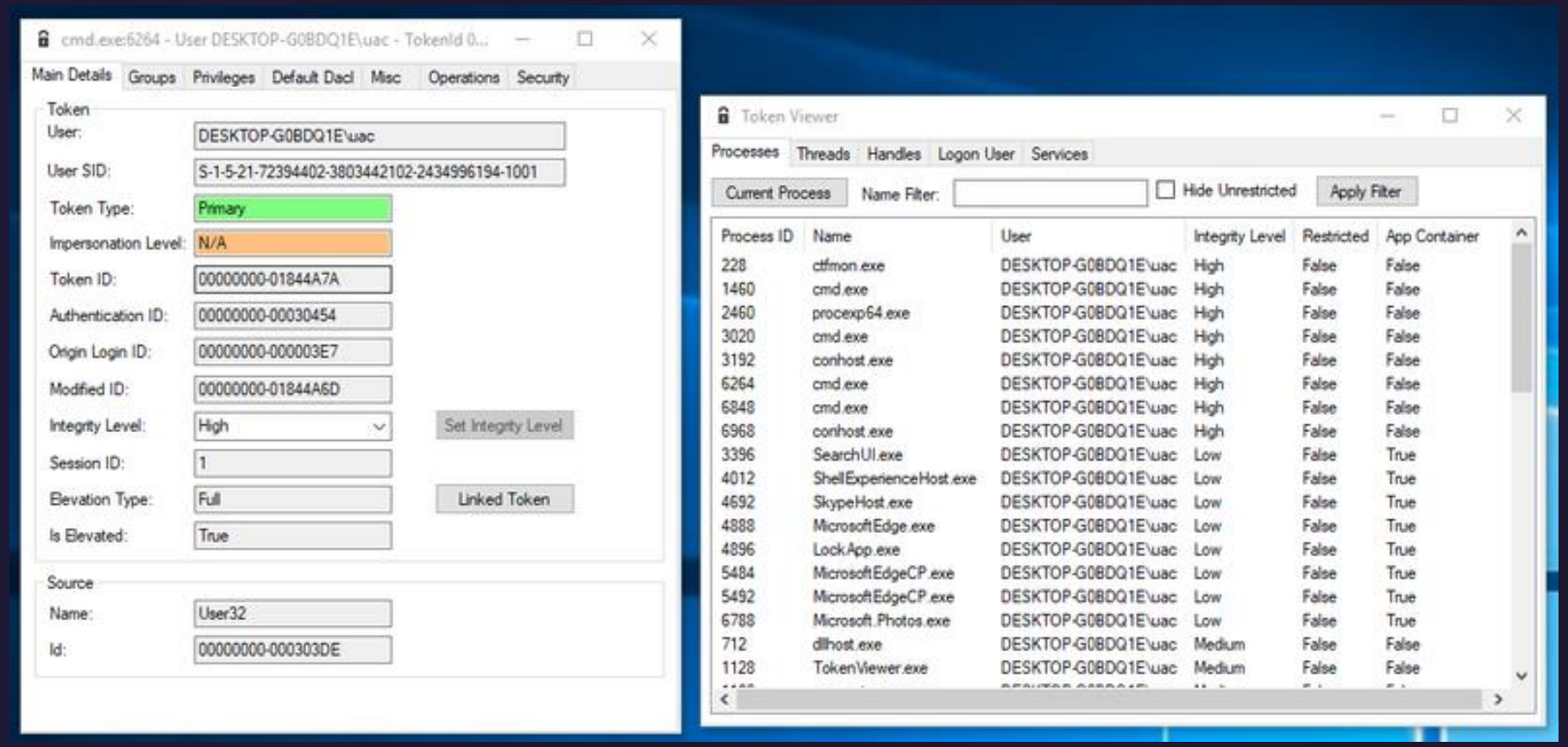
Review the Interfaces Security - RpcView



Review The Security of Interfaces - OleViewDoNet



Review the Interfaces Security – Token Viewer



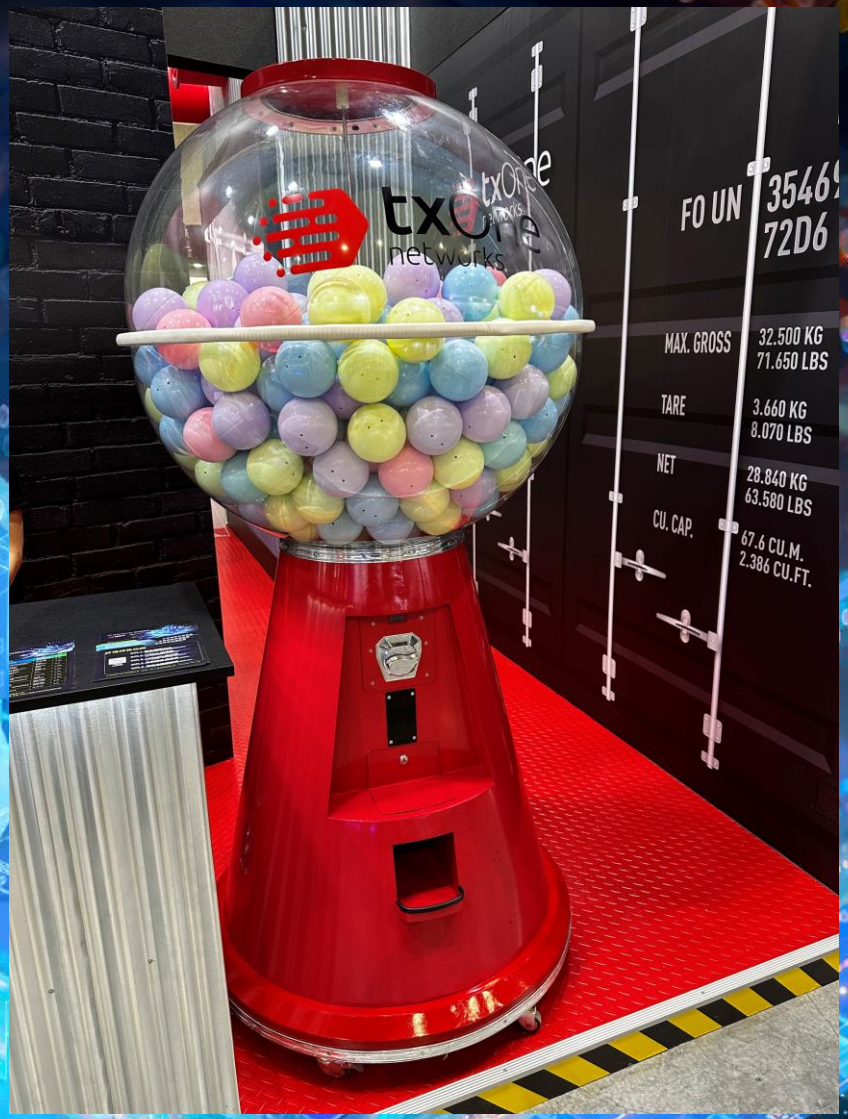
References

- Itm4n
  - Website: https://itm4n.github.io/
  - Keywords: RpcView, PrintSpoofer, PetimPotam, Efs Potato
- Foxglovesecurity
  - Website: https://foxglovesecurity.com/
  - Keywords: Hot Potato, Rotten Potato
- **Andrea Pierini & Antonio Cocomazzi**
  - Website: https://decoder.cloud/, https://www.localpotato.com/
  - Keywords: Rogue Potato, RemotePotato0, JuicyPotatoNG, Local Potato
- ophe
  - Website: https://ohpe.it/
  - Keywords: JuicyPotato
- shenaniganslabs
  - Website: https://shenaniganslabs.io/
  - Keywords: Ghost Potato
- **James Forshaw**
  - Website: https://googleprojectzero.blogspot.com/
  - Token Impersonation, SSPI Hook, OXID Resolver, DCOM Relay

Thanks for your listening!

OT Cybersecurity. **Simplified.**

txOne networks | Keep the Operation Running



掃描QR Code到TXOne攤位#C240玩扭蛋換好禮