10 years of Windows Privilege Escalation with Potatoes

Antonio Cocomazzi
Staff Offensive Security Researcher, SentinelOne

Andrea Pierini
Sr. Security Consultant, Semperis

Whoami

- ➔ Offensive Security Researcher @ SentinelOne
- ➔ Coding offensive tools + deepin into Windows internals
- ➔ Independent vulnerability researcher
- ➔ Gamer, League Of Legends fan, peak rank Diamond 1

@splinter_code

@antonioCoco

Why this talk

- ➔ Privilege escalation in Windows has always been our favorite pastime... well not exactly ;)
- ➔ We spent a lot of time trying to violate Windows safety and security boundaries by inventing new *potato techniques
- ➔ This is the story of our crazy ideas and sleepless nights :)

Agenda

- ➔ Privilege Escalation in Windows
- ➔ Where it all began - The RPC/DCOM trigger
- ➔ From Service -> SYSTEM
  - ◆ Rotten/JuicyPotato
  - ◆ RoguePotato
  - ◆ JuicyPotatoNG
- ➔ From User -> Admin
  - ◆ RemotePotato0
  - ◆ LocalPotato SMB edition
  - ◆ LocalPotato HTTP/WebDAV edition
- ➔ Conclusion

Privilege Escalation / Elevation of Privilege / EoP

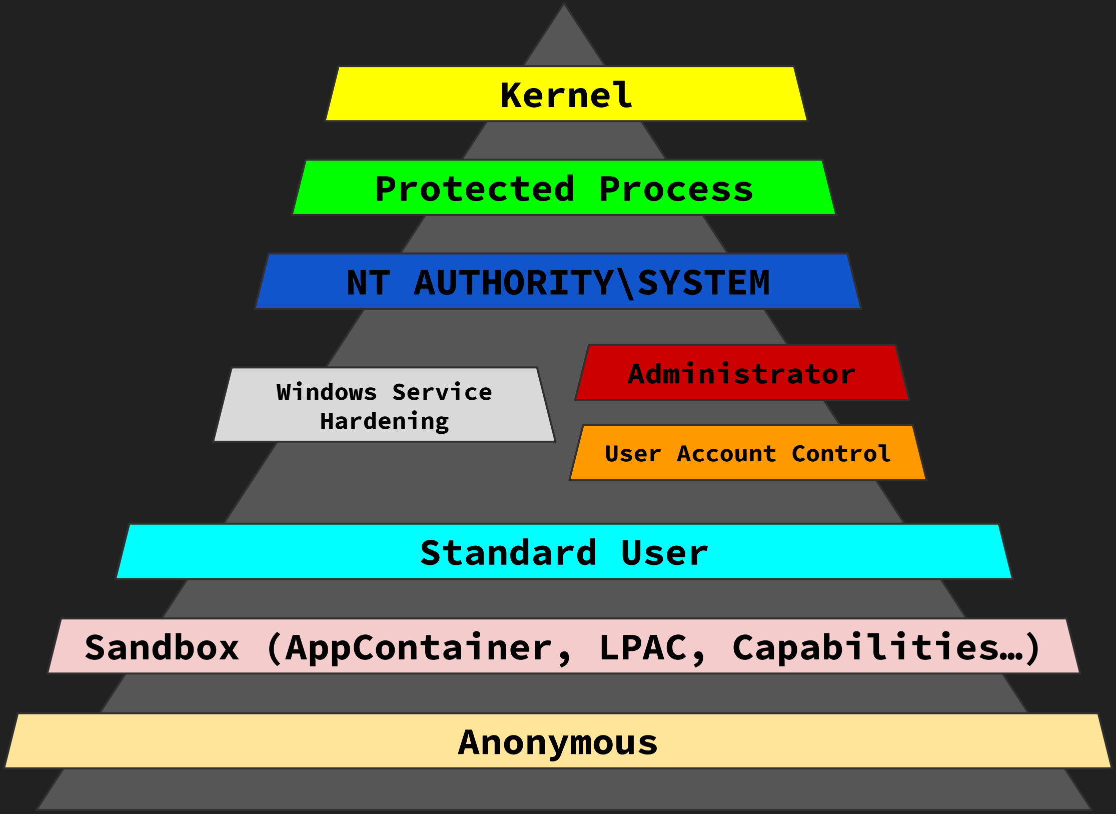
- “An elevation-of-privilege occurs when an application gains rights or privileges that should not be available to them” MSDN [1]

- Violation of a security boundary

- Security boundaries and features Microsoft intends to service [2]
  - Security boundaries (Process boundary, User boundary, AppContainer sandbox boundary, …)
  - Non-boundaries (Windows Server Containers, Administrator to Kernel, …)
  - Security features (Bitlocker, Secure Boot, WDAC, …)
  - Defense-in-depth security features (UAC, AppLocker, PPL, …)

[1] https://learn.microsoft.com/en-us/windows-hardware/drivers/ifs/elevation-of-privilege
[2] https://www.microsoft.com/en-us/msrc/windows-security-servicing-criteria

Layered Security model in Windows



Kernel

Protected Process

NT AUTHORITY\SYSTEM

Windows Service Hardening

Administrator

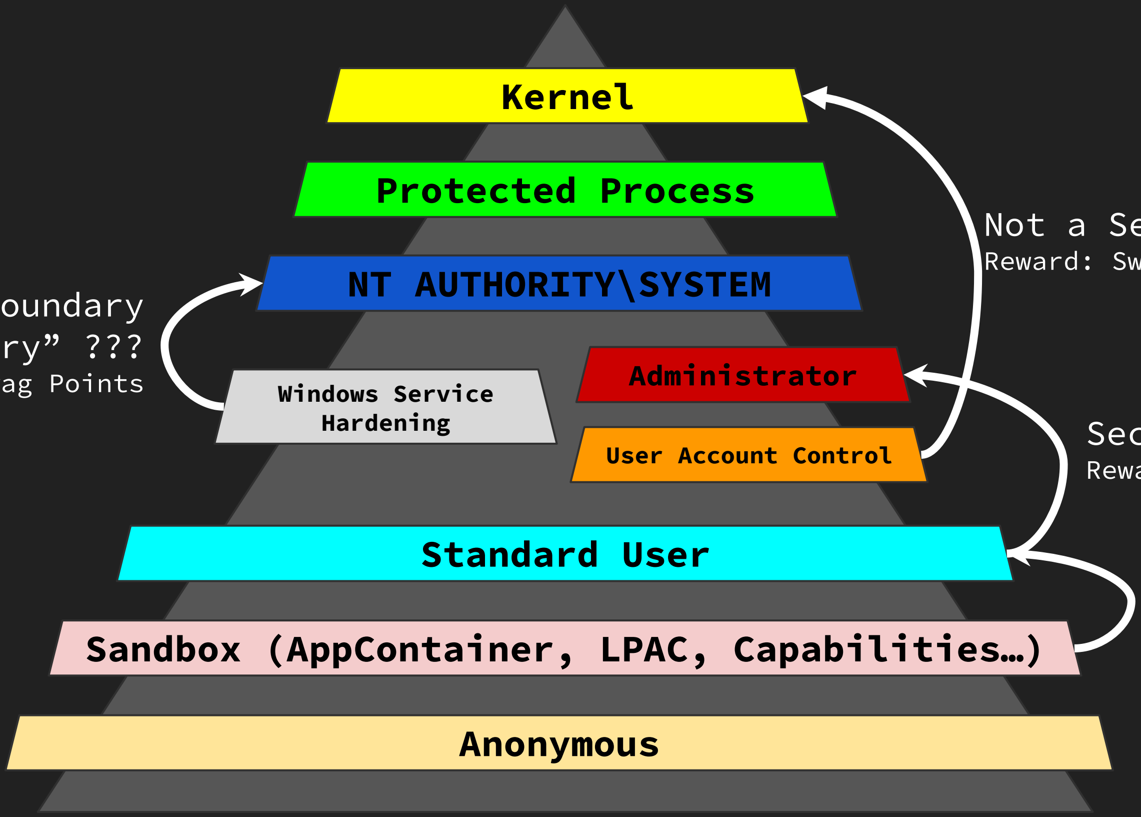
User Account Control

Standard User

Sandbox (AppContainer, LPAC, Capabilities…)

Anonymous

Layered Security model in Windows



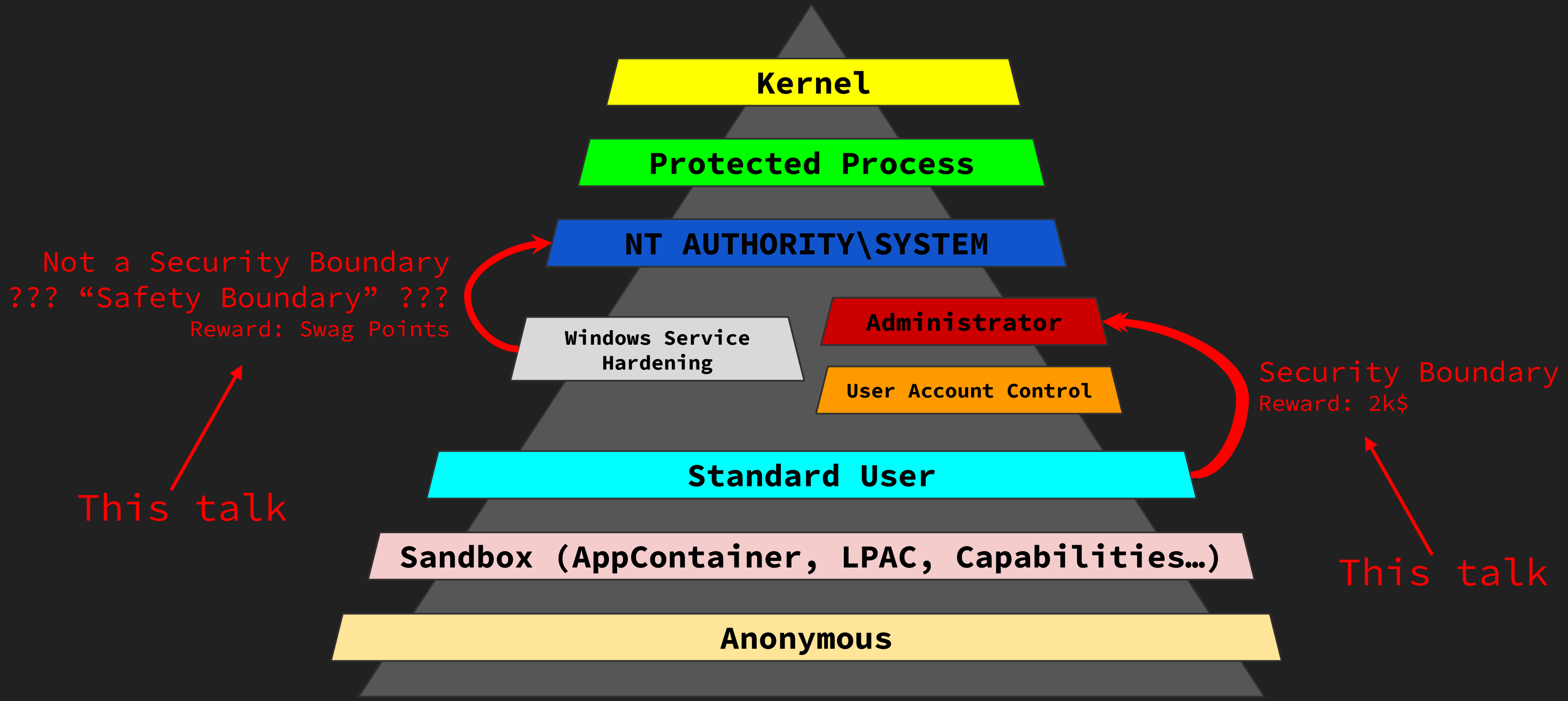
Not a Security Boundary
Reward: Swag Points

Not a Security Boundary
??? “Safety Boundary” ???
Reward: Swag Points

Security Boundary
Reward: 2k$

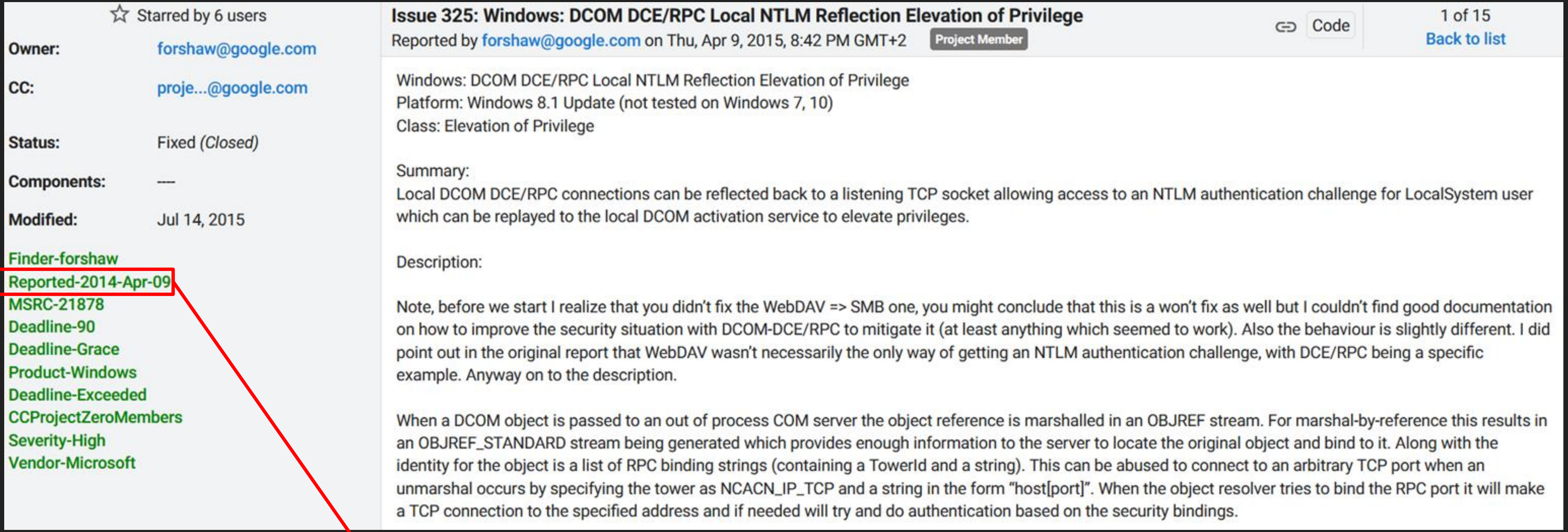
Security Boundary
Reward: 20k$ (sometimes)

Layered Security model in Windows



Where it all began

CVE-2015-2370 - DCOM DCE/RPC Local NTLM Reflection Elevation of Privilege



Owner:	forshaw@google.com
CC:	proje...@google.com
Status:	Fixed *(Closed)*
Components:	----
Modified:	Jul 14, 2015

Finder-forshaw
Reported-2014-Apr-09
MSRC-21878
Deadline-90
Deadline-Grace
Product-Windows
Deadline-Exceeded
CCProjectZeroMembers
Severity-High
Vendor-Microsoft

Issue 325: Windows: DCOM DCE/RPC Local NTLM Reflection Elevation of Privilege
Reported by forshaw@google.com on Thu, Apr 9, 2015, 8:42 PM GMT+2

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

Summary:
Local DCOM DCE/RPC connections can be reflected back to a listening TCP socket allowing access to an NTLM authentication challenge for LocalSystem user which can be replayed to the local DCOM activation service to elevate privileges.

Description:

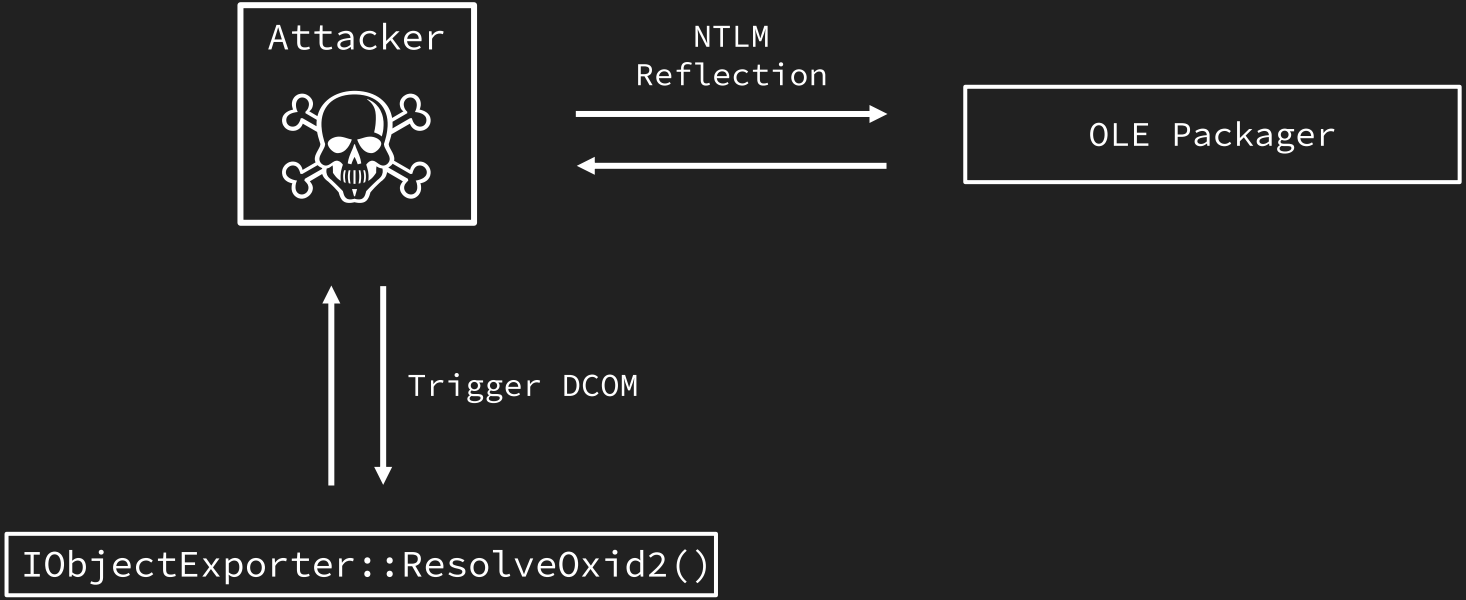
Note, before we start I realize that you didn't fix the WebDAV => SMB one, you might conclude that this is a won't fix as well but I couldn't find good documentation on how to improve the security situation with DCOM-DCE/RPC to mitigate it (at least anything which seemed to work). Also the behaviour is slightly different. I did point out in the original report that WebDAV wasn't necessarily the only way of getting an NTLM authentication challenge, with DCE/RPC being a specific example. Anyway on to the description.

When a DCOM object is passed to an out of process COM server the object reference is marshalled in an OBJREF stream. For marshal-by-reference this results in an OBJREF_STANDARD stream being generated which provides enough information to the server to locate the original object and bind to it. Along with the identity for the object is a list of RPC binding strings (containing a TowerId and a string). This can be abused to connect to an arbitrary TCP port when an unmarshal occurs by specifying the tower as NCACN_IP_TCP and a string in the form "host[port]". When the object resolver tries to bind the RPC port it will make a TCP connection to the specified address and if needed will try and do authentication based on the security bindings.

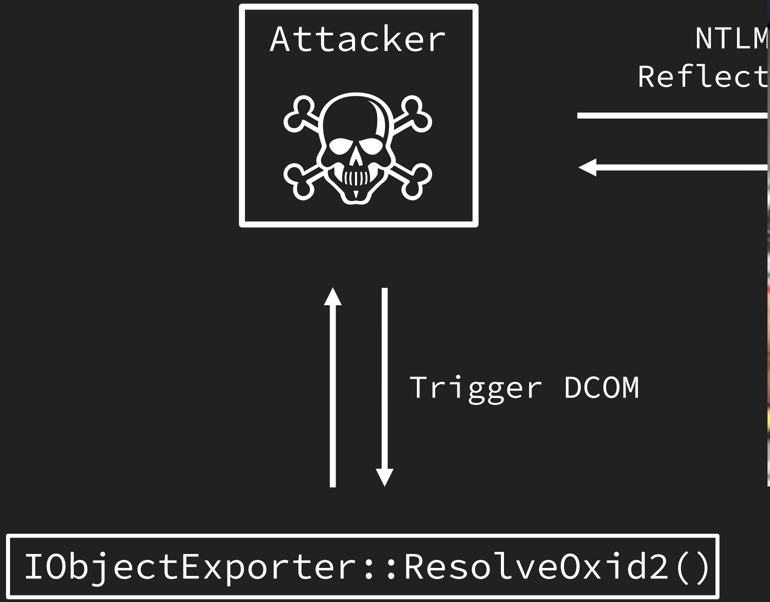
Reported-2014-Apr-09

https://bugs.chromium.org/p/project-zero/issues/detail?id=325

CVE-2015-2370 - Attack flow



CVE-2015-2370 - Microsoft Fix



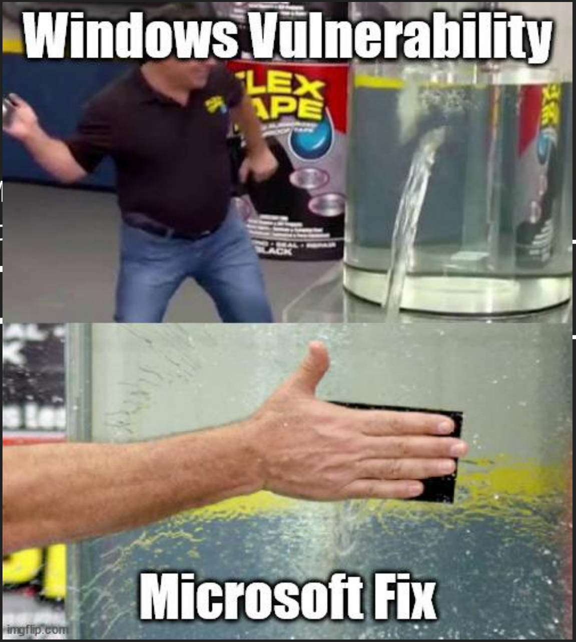
Attacker

NTLM

Reflect

Trigger DCOM

IObjectExporter::ResolveOxid2()



The RPC/DCOM trigger

- It abuses the standard COM marshalling
- Craft a malicious OBJREF_STANDARD marshalled interface
- The malicious marshalled object contains the address+port of an attacker controller RPC server as the Oxid Resolver address
- Oxid Resolution is needed for locating the binding information of the COM object. This needs to be authenticated.
- Use CoGetInstanceFromIStorage to perform the resolution in the security context of a privileged service. (DCOM activation)
- Privileged Oxid Resolution occurs from IObjectExporter::ResolveOxid2() -> privileged authentication comes to the attacker -> Profit!

https://www.youtube.com/watch?v=dfMuzAZRGm4 (James Forshaw - COM in Sixty Seconds! @ Infiltrate 2017)

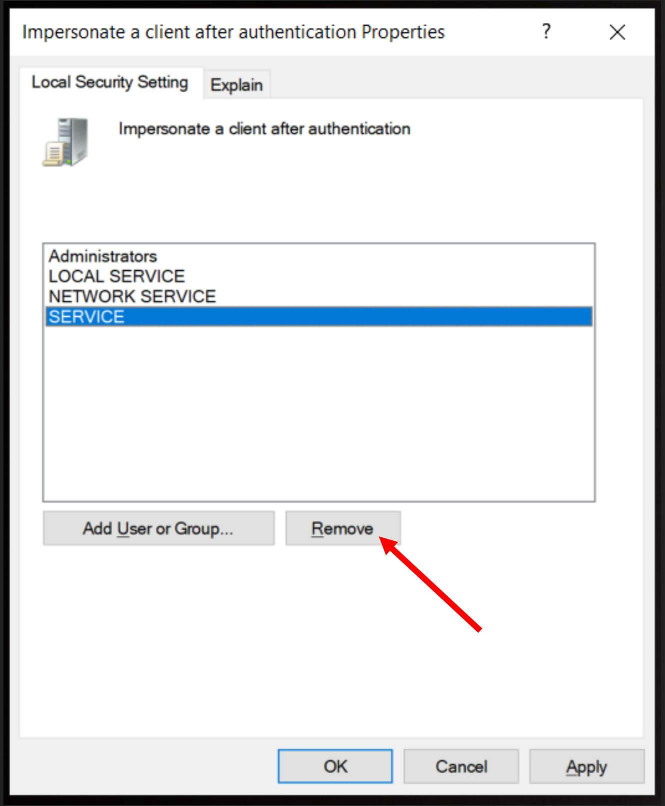
CVE-2015-2370 - after the fix

- ~~Reflect the NTLM back to a local RPC TCP endpoint~~

- ~~Use the NTLM for reflection back to the local SMB service~~

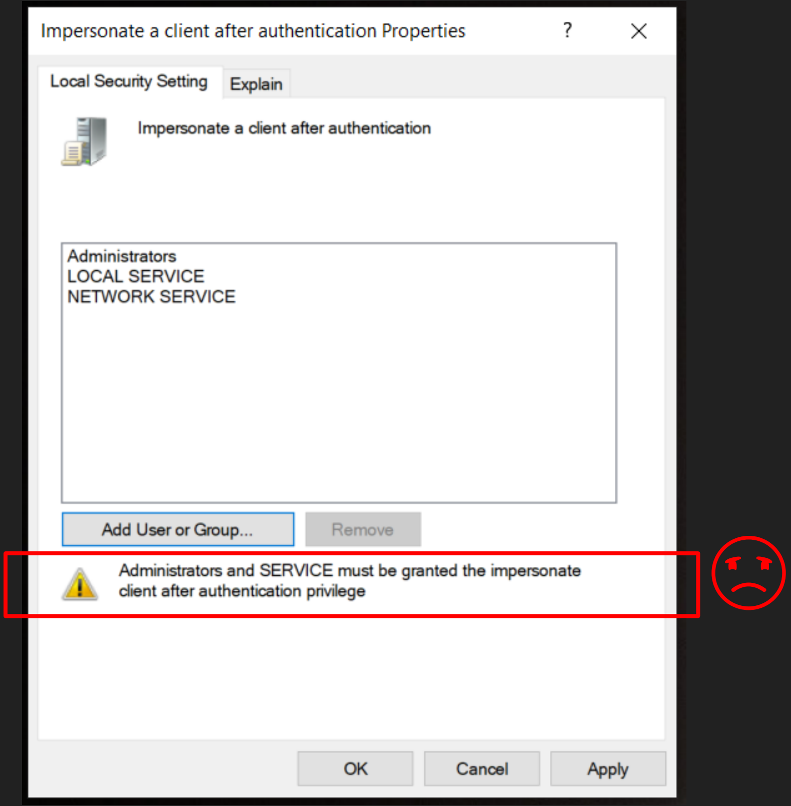
- Locally negotiate the NTLM which will give you back a full impersonation level token of SYSTEM and can break **WSH** through **Impersonation** privileges

The link between Services and Impersonation privileges



Impersonate a client after authentication

Administrators
LOCAL SERVICE
NETWORK SERVICE
SERVICE



Impersonate a client after authentication

Administrators
LOCAL SERVICE
NETWORK SERVICE

Administrators and SERVICE must be granted the impersonate client after authentication privilege

Windows Service Hardening (WSH)

- ➔ Limited Service Accounts
  - ◆ Introduction of the **LOCAL SERVICE** and **NETWORK SERVICE** accounts, less privileges than **SYSTEM** account.
- ➔ Reduced Privileges
  - ◆ Services run only with specified privileges (least privilege)
- ➔ Write-Restricted Token
- ➔ Per-Service SID
  - ◆ Service access token has dedicated and unique owner SID. No SID sharing across different services
- ➔ Session 0 Isolation
- ➔ System Integrity Level
- ➔ UIPI (User interface privilege isolation)

https://www.tiraniddo.dev/2020/01/empirically-assessing-windows-service.html
https://downloads.immunityinc.com/infiltrate-archives/WindowsServicesHacking.pdf

From Service -> SYSTEM

RottenPotato

- ➔ Released by @breenmachine and @vvalien1 in Sep 2016
- ➔ First potato exploit which leverages the DCOM trigger with the Impersonation privileges.
- ➔ Use fixed BITS CLSID to trigger a SYSTEM auth
- ➔ Use fixed 6666 port for the relay server
- ➔ Relay to local Oxid Resolver (port 135) and perform a MITM:
  - ◆ Intercept NTLM SSP exchange and negotiate a SYSTEM token
- ➔ Initially designed to be run through incognito+meterpreter shell

https://foxglovesecurity.com/2016/09/26/rotten-potato-privilege-escalation-from-service-accounts-to-system/
https://www.youtube.com/watch?v=8Wjs_mWOKI

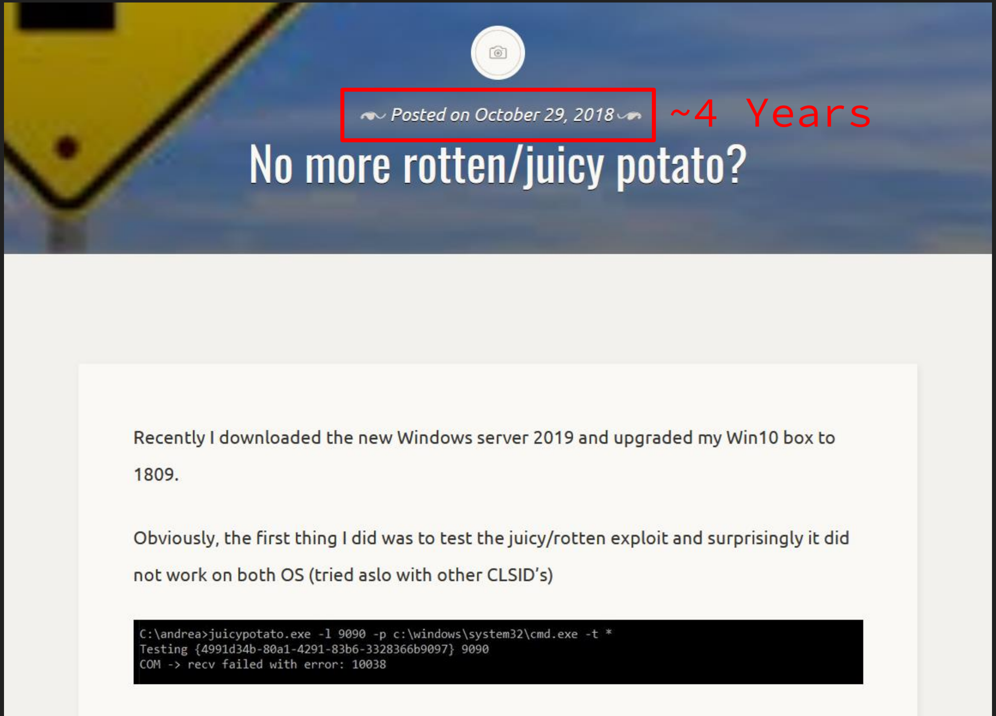
JuicyPotato (abusing the golden privileges)

- Released by @decoder_it and @Giutro in Aug 2018
- A sugared version of RottenPotatoNG, with a bit of juice:
  - Removed limitation of fixed 6666 port for the relay server
  - A lot of COM servers to abuse, not only BITS
  - Use CreateProcessAsUser() or CreateProcessWithTokenW() for arbitrary process creation as SYSTEM
- A lot of fun when doing post-exploitation on IIS or MSSQL services

https://decoder.cloud/2018/08/10/juicy-potato/
https://github.com/ohpe/juicy-potato

Demo 1 - JuicyPotato

JuicyPotato - the silent fix



Posted on October 29, 2018

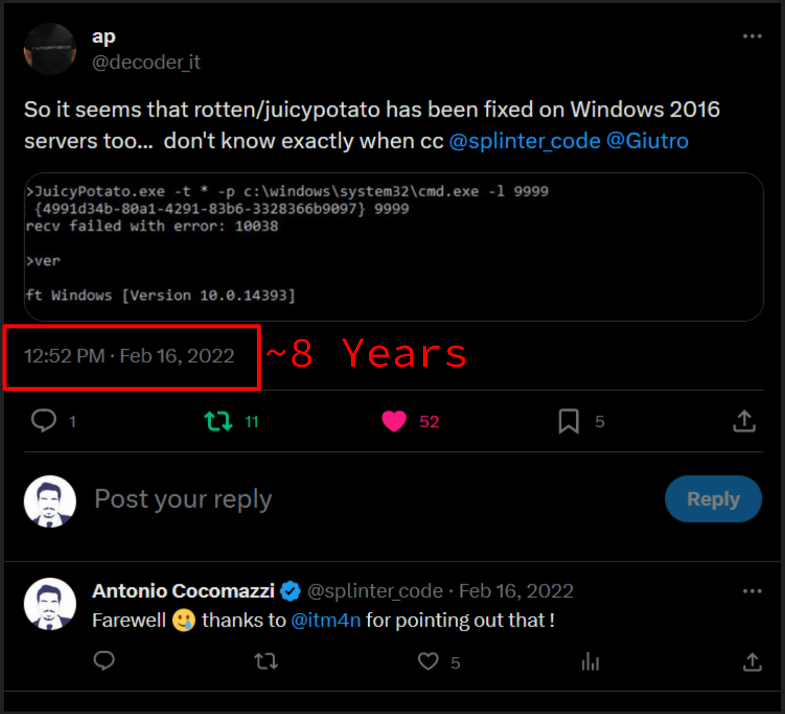
~4 Years

No more rotten/juicy potato?

Recently I downloaded the new Windows server 2019 and upgraded my Win10 box to 1809.

Obviously, the first thing I did was to test the juicy/rotten exploit and surprisingly it did not work on both OS (tried aslo with other CLSID's)

```
C:\andrea>juicypotato.exe -l 9090 -p c:\windows\system32\cmd.exe -t *
Testing {4991d34b-80a1-4291-83b6-3328366b9097} 9090
COM -> recv failed with error: 10038
```



ap
@decoder_it

So it seems that rotten/juicypotato has been fixed on Windows 2016 servers too... don't know exactly when cc @splinter_code @Giutro

```
>JuicyPotato.exe -t * -p c:\windows\system32\cmd.exe -l 9999
 {4991d34b-80a1-4291-83b6-3328366b9097} 9999
recv failed with error: 10038

>ver

ft Windows [Version 10.0.14393]
```

12:52 PM · Feb 16, 2022

~8 Years

Antonio Cocomazzi @splinter_code · Feb 16, 2022
Farewell 😢 thanks to @itm4n for pointing out that !

https://decoder.cloud/2018/10/29/no-more-rotten-juicy-potato/
https://twitter.com/decoder_it/status/1493916092493877248

JuicyPotato - the silent fix

- The ninja patch is inside rpcss.dll

- In unpatched versions the Oxid binding was created through the function MakeBinding():
  - Manually crafts the string binding with {address} + ‘[‘ + {port} + ‘]’
  - The string binding become ncacn_ip_tcp:127.0.0.1[6666]**[135]**
  - RpcBindingFromStringBinding() will use ncacn_ip_tcp:127.0.0.1[6666]

- In patched versions a new dedicated function is used CreateRemoteBindingToOr():
  - It crafts the string binding through RpcStringBindingCompose()
  - The string binding become ncacn_ip_tcp:127.0.0.1**\[6666\]**[135]
  - RpcBindingFromStringBinding() fails due to the ‘\’ chars -> Exploit breaks

JuicyPotato - the silent fix

- The ninja
- In unpatch ted through the functi
  - Manually {port} + ‘]’
  - The stri]
  - RpcBindi 0.1[6666]
- In patched used CreateRemo
  - It crafts se()
  - The stri 35]
  - RpcBindi > Exploit breaks

RoguePotato

- Instead of using a custom local port, it uses a remote IP as a custom Oxid Resolver
- Implements a fake Oxid Resolver which returns a poisoned answer:
  - ncacn_np:localhost/pipe/roguepotato[\pipe\epmapper]
  - Pipe used become **\\localhost\pipe\roguepotato\pipe\epmapper** due to a bug in converting the ‘/’ char [1]
- Intercept authentication to custom named pipe
- Authentication is performed by rpcss service as NETWORK SERVICE, but with the RpcSs LUID
- Token Kidnapping a SYSTEM token from the rpcss service
- Create a new process with the stolen token

[1] https://itm4n.github.io/printspoofer-abusing-impersonate-privileges/
https://decoder.cloud/2020/05/11/no-more-juicypotato-old-story-welcome-roguepotato/
https://github.com/antonioCoco/RoguePotato

Demo 2 - RoguePotato

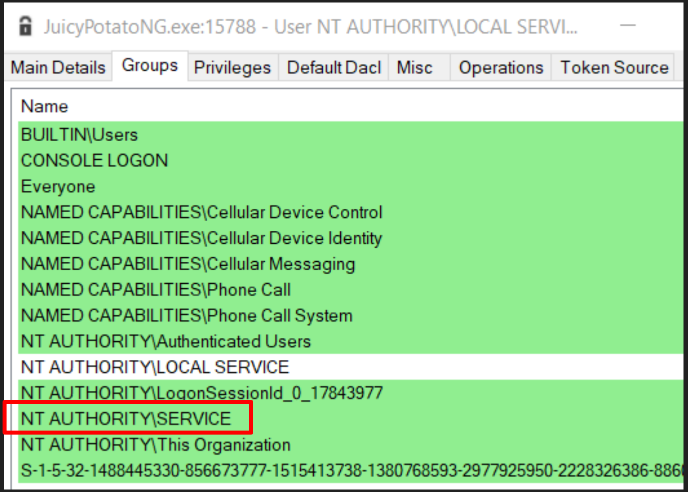
JuicyPotatoNG

- ➔ Uses RPC over TCP (ncacn_ip_tcp)
- ➔ Removed requirement for an external Oxid Resolver, fully local exploit, trick by James Forshaw [1]
- ➔ Uses a trick to recover INTERACTIVE sid and unlock interesting CLSIDs, e.g. PrintNotify service
- ➔ Basically we revived JuicyPotato [2]

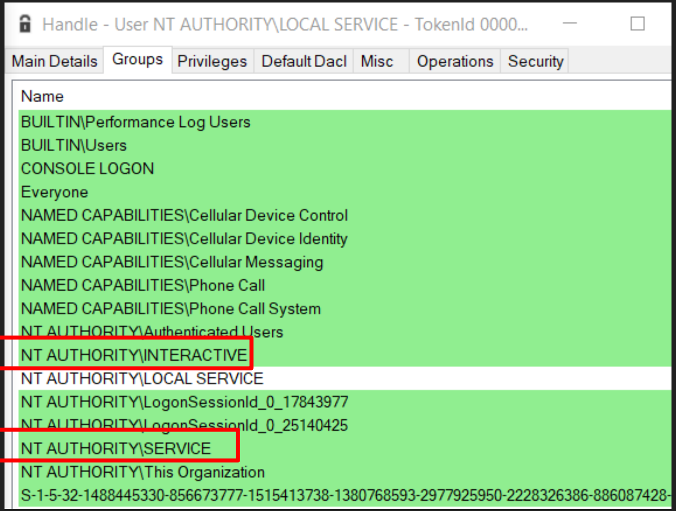
[1] https://googleprojectzero.blogspot.com/2021/10/windows-exploitation-tricks-relaying.html
[2] https://decoder.cloud/2022/09/21/giving-juicypotato-a-second-chance-juicypotatong/

JuicyPotatoNG - trick to recover INTERACTIVE sid

- ➔ LogonUserW documentation about NewCredentials logon type:
  - ◆ “This logon type allows the caller to clone its current token and specify new credentials for outbound connections…” MSDN



Name
BUILTIN\Users
CONSOLE LOGON
Everyone
NAMED CAPABILITIES\Cellular Device Control
NAMED CAPABILITIES\Cellular Device Identity
NAMED CAPABILITIES\Cellular Messaging
NAMED CAPABILITIES\Phone Call
NAMED CAPABILITIES\Phone Call System
NT AUTHORITY\Authenticated Users
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\LogonSessionId_0_17843977
NT AUTHORITY\SERVICE
NT AUTHORITY\This Organization
S-1-5-32-1488445330-856673777-1515413738-1380768593-2977925950-2228326386-886



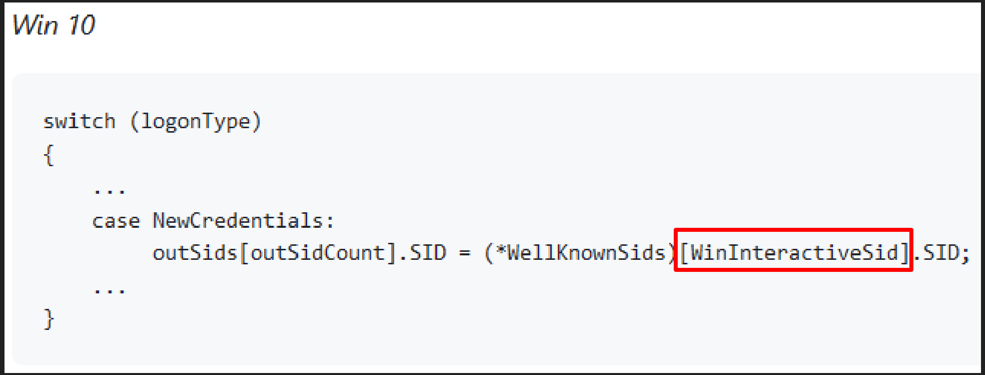
Name
BUILTIN\Performance Log Users
BUILTIN\Users
CONSOLE LOGON
Everyone
NAMED CAPABILITIES\Cellular Device Control
NAMED CAPABILITIES\Cellular Device Identity
NAMED CAPABILITIES\Cellular Messaging
NAMED CAPABILITIES\Phone Call
NAMED CAPABILITIES\Phone Call System
NT AUTHORITY\Authenticated Users
NT AUTHORITY\INTERACTIVE
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\LogonSessionId_0_17843977
NT AUTHORITY\LogonSessionId_0_25140425
NT AUTHORITY\SERVICE
NT AUTHORITY\This Organization
S-1-5-32-1488445330-856673777-1515413738-1380768593-2977925950-2228326386-886087428-

https://learn.microsoft.com/en-us/windows/win32/api/winbase/nf-winbase-logonuserw

Demo 3 - JuicyPotatoNG

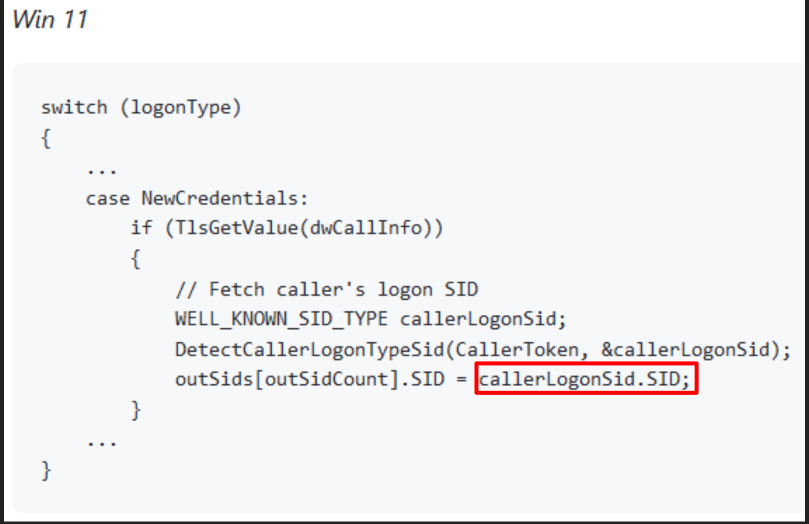
JuicyPotatoNG - the silent fix

- ➔ Starting from Win 11 22H2 a new change in lsasrv.dll!LsapAuAddStandardIds():



Win 10

```
switch (logonType)
{
    ...
    case NewCredentials:
         outSids[outSidCount].SID = (*WellKnownSids)[WinInteractiveSid].SID;
    ...
}
```



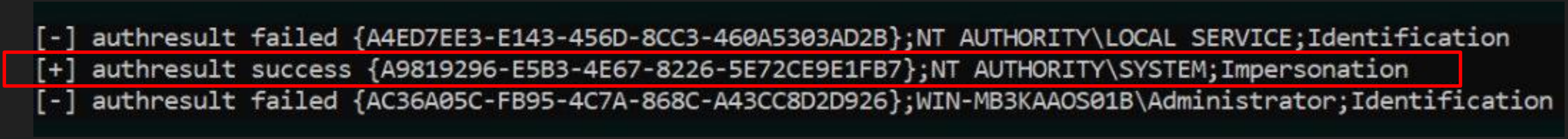
Win 11

```
switch (logonType)
{
    ...
    case NewCredentials:
        if (TlsGetValue(dwCallInfo))
        {
            // Fetch caller's logon SID
            WELL_KNOWN_SID_TYPE callerLogonSid;
            DetectCallerLogonTypeSid(CallerToken, &callerLogonSid);
            outSids[outSidCount].SID = callerLogonSid.SID;
        }
    ...
}
```

https://github.com/antonioCoco/JuicyPotatoNG/issues/4

JuicyPotatoNG - the silent fix???

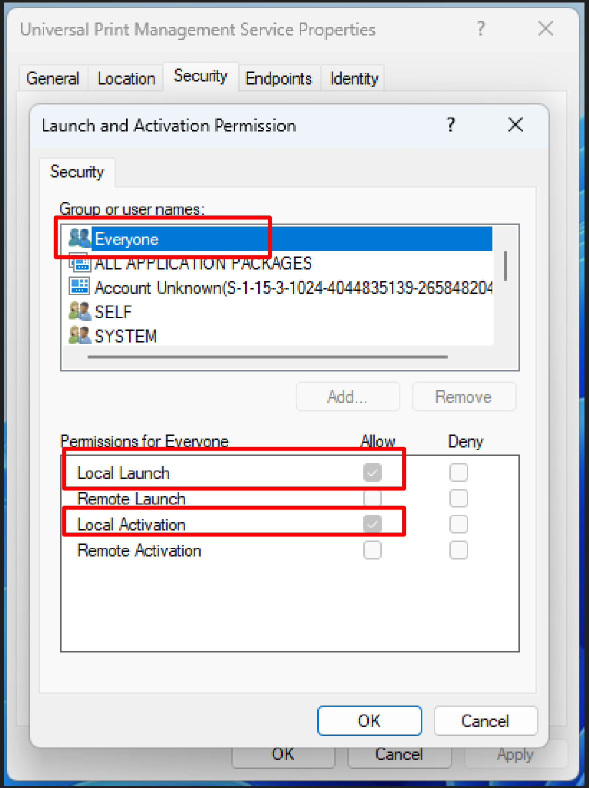
- Starting from Win 11 / Server 2022 a new available CLSID:
  - Universal Print Management Service (McpManagementService) - CLSID: {A9819296-E5B3-4E67-8226-5E72CE9E1FB7}



```
[-] authresult failed {A4ED7EE3-E143-456D-8CC3-460A5303AD2B};NT AUTHORITY\LOCAL SERVICE;Identification
[+] authresult success {A9819296-E5B3-4E67-8226-5E72CE9E1FB7};NT AUTHORITY\SYSTEM;Impersonation
[-] authresult failed {AC36A05C-FB95-4C7A-868C-A43CC8D2D926};WIN-MB3KAAOS01B\Administrator;Identification
```

https://twitter.com/decoder_it/status/1602673748234190848

JuicyPotatoNG - the silent fix???



Universal Print Management Service Properties

Launch and Activation Permission

Security

Group or user names:

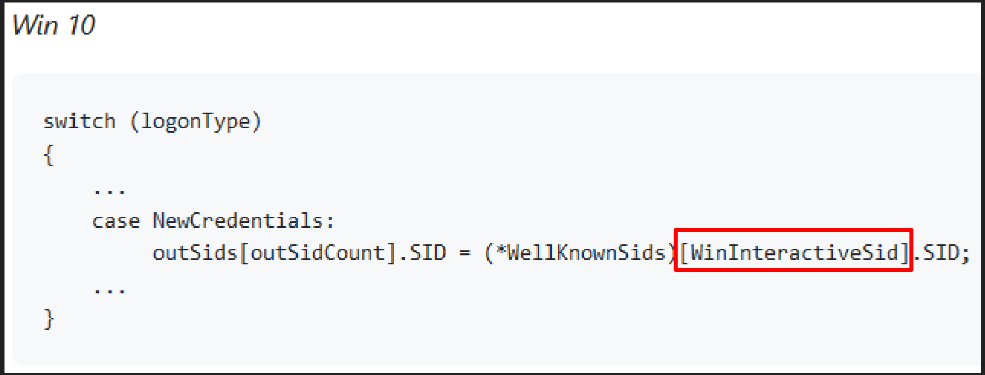
- Everyone
- ALL APPLICATION PACKAGES
- Account Unknown(S-1-15-3-1024-4044835139-265848204
- SELF
- SYSTEM

Permissions for Everyone	Allow	Deny
Local Launch	☑	□
Remote Launch	□	□
Local Activation	☑	□
Remote Activation	□	□

➔ Use the CLSID **{A9819296-E5B3-4E67-8226-5E72CE9E1FB7}** in JuicyPotatoNG and it will work also on patched Win 11 22H2 systems!

JuicyPotatoNG - the silent fix

- ➔ Starting from Win 11 22H2 a new
  lsasrv.dll!LsapAuAddStandardIds(



Win 10

```
switch (logonType)
{
    ...
    case NewCredentials:
         outSids[outSidCount].SID = (*WellKnownSids)[WinInteractiveSid].SID;
    ...
}
```

https://github.com/antonioCoco/JuicyPotatoNG/issues/4

And the Potato dynasty is not over...

- SweetPotato
  - https://github.com/CCob/SweetPotato
- GodPotato
  - https://github.com/BeichenDream/GodPotato
- PrintNotifyPotato
  - https://github.com/BeichenDream/PrintNotifyPotato
- PetitPotato
  - https://github.com/wh0amitz/PetitPotato
- EfsPotato
  - https://github.com/zcgonvh/EfsPotato
- DCOMPotato
  - https://github.com/zcgonvh/DCOMPotato

- Thanks to the community and keep them coming!

From Safety Boundary -> Security Boundary Violation

RemotePotato0

- Abuses COM servers configured with RunAs “Interactive User” and performs cross session activation [1]

- Downgrade attack in NTLM to bypass MIC and SIGNING through ResolveOxid2() response

- Relay NTLM to LDAP to elevate your privileges (main scenario)

- Particularly effective when exploiting terminal servers and multiple users are logged on

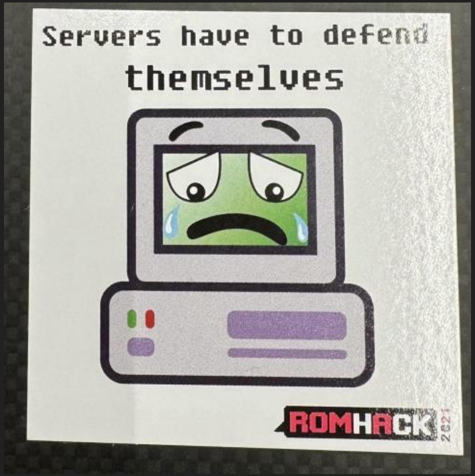
[1] https://www.tiraniddo.dev/2021/04/standard-activating-yourself-to.html
https://www.sentinelone.com/labs/relaying-potatoes-another-unexpected-privilege-escalation-vulnerability-in-windows-rpc-protocol/
https://github.com/antonioCoco/RemotePotato0
https://www.youtube.com/watch?v=vfb-bH_HaW4 - BlueHat IL 2022 - Antonio Cocomazzi & Andrea Pierini - Relaying to Greatness

Demo 4 - RemotePotato0 relay to LDAP

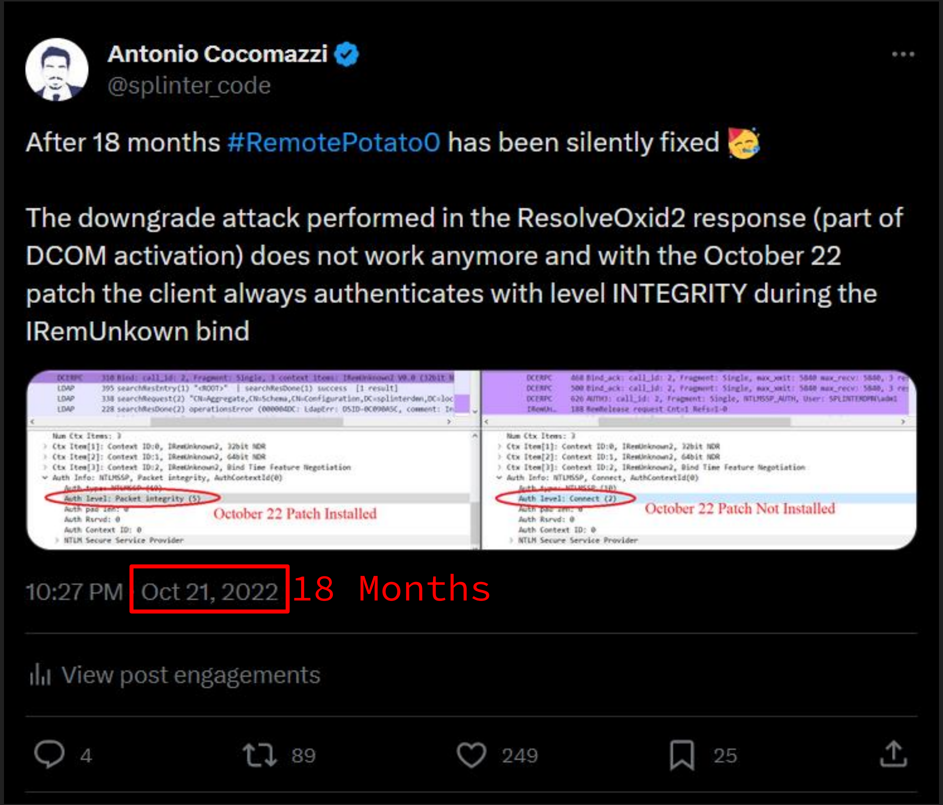
RemotePotato0 - Disclosure

- Bounty awarded: 2.000 $

- “After an extensive review, we determined that servers must defend themselves against NTLM relay attacks” MSRC



RemotePotato0 - the silent fix



Antonio Cocomazzi
@splinter_code

After 18 months #RemotePotato0 has been silently fixed 🥳

The downgrade attack performed in the ResolveOxid2 response (part of DCOM activation) does not work anymore and with the October 22 patch the client always authenticates with level INTEGRITY during the IRemUnkown bind

```
DCERPC    310 Bind: call_id: 2, Fragment: Single, 3 context items: IRemUnknown2 V0.0 (32bit N
LDAP      395 searchResEntry(1) "<ROOT>"  | searchResDone(1) success  [1 result]
LDAP      338 searchRequest(2) "CN=Aggregate,CN=Schema,CN=Configuration,DC=splinterdm,DC=loc
LDAP      228 searchResDone(2) operationsError (000004DC: LdapErr: DSID-0C090A5C, comment: In

  Num Ctx Items: 3
> Ctx Item[1]: Context ID:0, IRemUnknown2, 32bit NDR
> Ctx Item[2]: Context ID:1, IRemUnknown2, 64bit NDR
> Ctx Item[3]: Context ID:2, IRemUnknown2, Bind Time Feature Negotiation
v Auth Info: NTLMSSP, Packet integrity, AuthContextId(0)
    ~~Auth type: NTLMSSP (10)~~
    Auth level: Packet integrity (5)
    Auth pad len: 0
    Auth Rsrvd: 0
    Auth Context ID: 0
  > NTLM Secure Service Provider
```

October 22 Patch Installed

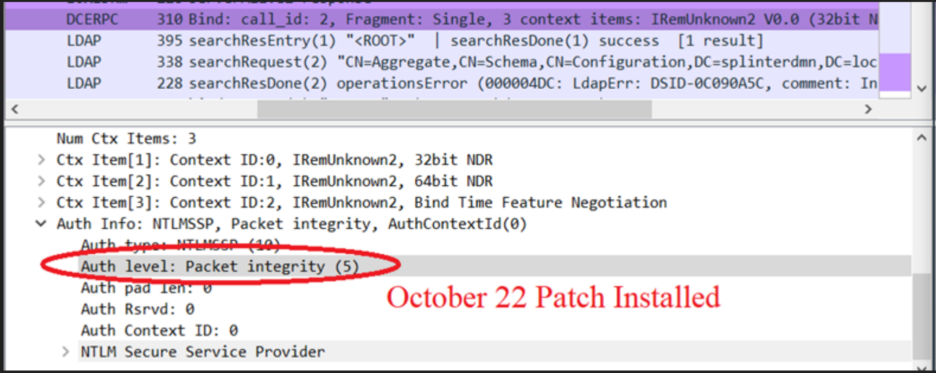
```
DCERPC    468 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 re
DCERPC    500 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 re
DCERPC    626 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: SPLINTERDMN\adm1
IRemUn..  188 RemRelease request Cnt=1 Refs=1-0

  Num Ctx Items: 3
> Ctx Item[1]: Context ID:0, IRemUnknown2, 32bit NDR
> Ctx Item[2]: Context ID:1, IRemUnknown2, 64bit NDR
> Ctx Item[3]: Context ID:2, IRemUnknown2, Bind Time Feature Negotiation
v Auth Info: NTLMSSP, Connect, AuthContextId(0)
    ~~Auth type: NTLMSSP (10)~~
    Auth level: Connect (2)
    Auth pad len: 0
    Auth Rsrvd: 0
    Auth Context ID: 0
  > NTLM Secure Service Provider
```

October 22 Patch Not Installed

10:27 PM · Oct 21, 2022 18 Months

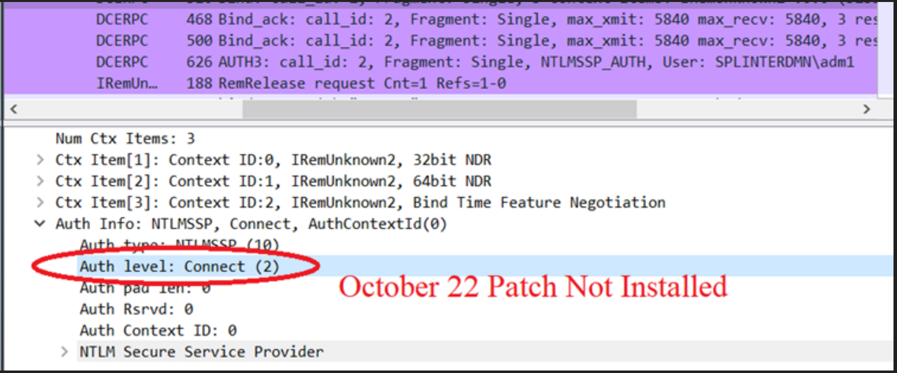
https://twitter.com/splinter_code/status/1583555613950255104

RemotePotato0 - the silent fix



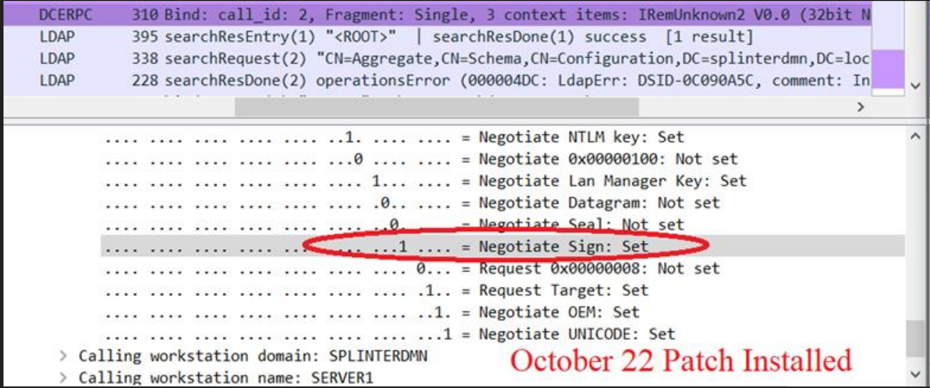
```
DCERPC     310 Bind: call_id: 2, Fragment: Single, 3 context items: IRemUnknown2 V0.0 (32bit N
LDAP       395 searchResEntry(1) "<ROOT>"  | searchResDone(1) success  [1 result]
LDAP       338 searchRequest(2) "CN=Aggregate,CN=Schema,CN=Configuration,DC=splinterdmn,DC=loc
LDAP       228 searchResDone(2) operationsError (000004DC: LdapErr: DSID-0C090A5C, comment: In

  Num Ctx Items: 3
> Ctx Item[1]: Context ID:0, IRemUnknown2, 32bit NDR
> Ctx Item[2]: Context ID:1, IRemUnknown2, 64bit NDR
> Ctx Item[3]: Context ID:2, IRemUnknown2, Bind Time Feature Negotiation
v Auth Info: NTLMSSP, Packet integrity, AuthContextId(0)
    Auth type: NTLMSSP (10)
    Auth level: Packet integrity (5)
    Auth pad len: 0
    Auth Rsrvd: 0
    Auth Context ID: 0
  > NTLM Secure Service Provider
```

October 22 Patch Installed



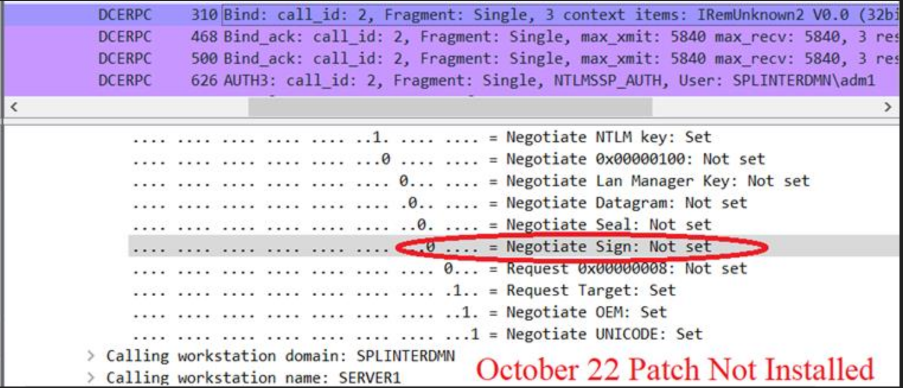
```
DCERPC     468 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 res
DCERPC     500 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 res
DCERPC     626 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: SPLINTERDMN\adm1
IRemUn…    188 RemRelease request Cnt=1 Refs=1-0

  Num Ctx Items: 3
> Ctx Item[1]: Context ID:0, IRemUnknown2, 32bit NDR
> Ctx Item[2]: Context ID:1, IRemUnknown2, 64bit NDR
> Ctx Item[3]: Context ID:2, IRemUnknown2, Bind Time Feature Negotiation
v Auth Info: NTLMSSP, Connect, AuthContextId(0)
    Auth type: NTLMSSP (10)
    Auth level: Connect (2)
    Auth pad len: 0
    Auth Rsrvd: 0
    Auth Context ID: 0
  > NTLM Secure Service Provider
```

October 22 Patch Not Installed



```
DCERPC     310 Bind: call_id: 2, Fragment: Single, 3 context items: IRemUnknown2 V0.0 (32bit N
LDAP       395 searchResEntry(1) "<ROOT>"  | searchResDone(1) success  [1 result]
LDAP       338 searchRequest(2) "CN=Aggregate,CN=Schema,CN=Configuration,DC=splinterdmn,DC=loc
LDAP       228 searchResDone(2) operationsError (000004DC: LdapErr: DSID-0C090A5C, comment: In

    .... .... .... .... .... ..1. .... .... = Negotiate NTLM key: Set
    .... .... .... .... .... ...0 .... .... = Negotiate 0x00000100: Not set
    .... .... .... .... .... .... 1... .... = Negotiate Lan Manager Key: Set
    .... .... .... .... .... .... .0.. .... = Negotiate Datagram: Not set
    .... .... .... .... .... .... ..0. .... = Negotiate Seal: Not set
    .... .... .... .... .... .... ...1 .... = Negotiate Sign: Set
    .... .... .... .... .... .... .... 0... = Request 0x00000008: Not set
    .... .... .... .... .... .... .... .1.. = Request Target: Set
    .... .... .... .... .... .... .... ..1. = Negotiate OEM: Set
    .... .... .... .... .... .... .... ...1 = Negotiate UNICODE: Set
  > Calling workstation domain: SPLINTERDMN
  > Calling workstation name: SERVER1
```

October 22 Patch Installed

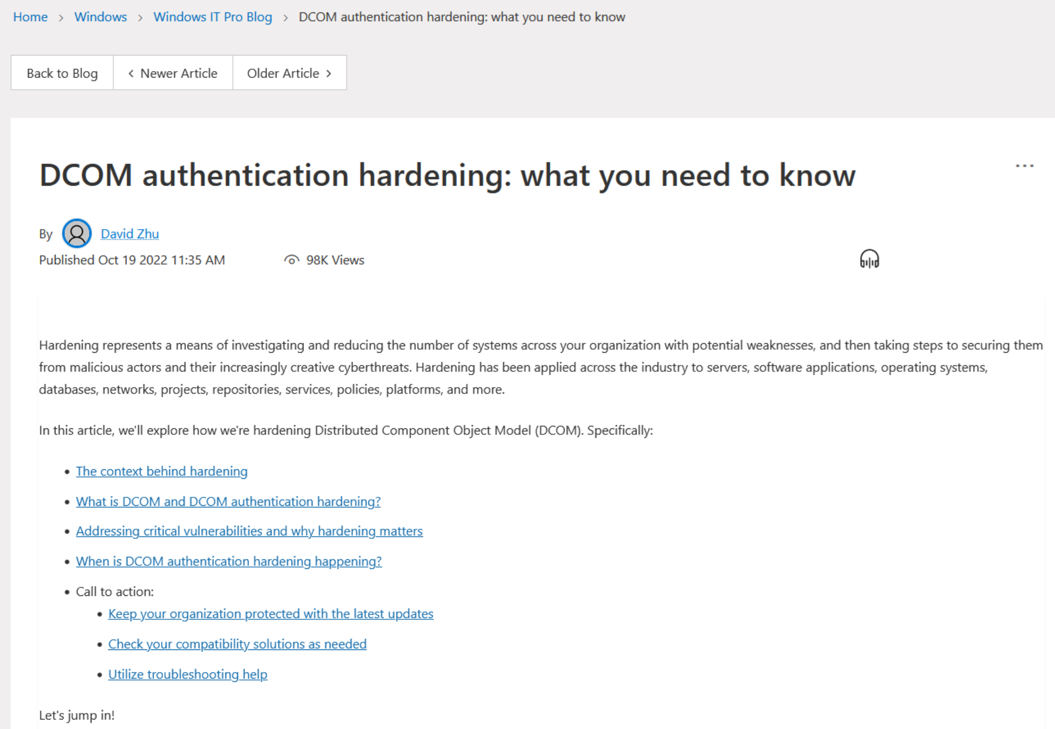


```
DCERPC     310 Bind: call_id: 2, Fragment: Single, 3 context items: IRemUnknown2 V0.0 (32bi
DCERPC     468 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 res
DCERPC     500 Bind_ack: call_id: 2, Fragment: Single, max_xmit: 5840 max_recv: 5840, 3 res
DCERPC     626 AUTH3: call_id: 2, Fragment: Single, NTLMSSP_AUTH, User: SPLINTERDMN\adm1

    .... .... .... .... .... ..1. .... .... = Negotiate NTLM key: Set
    .... .... .... .... .... ...0 .... .... = Negotiate 0x00000100: Not set
    .... .... .... .... .... .... 0... .... = Negotiate Lan Manager Key: Not set
    .... .... .... .... .... .... .0.. .... = Negotiate Datagram: Not set
    .... .... .... .... .... .... ..0. .... = Negotiate Seal: Not set
    .... .... .... .... .... .... ...0 .... = Negotiate Sign: Not set
    .... .... .... .... .... .... .... 0... = Request 0x00000008: Not set
    .... .... .... .... .... .... .... .1.. = Request Target: Set
    .... .... .... .... .... .... .... ..1. = Negotiate OEM: Set
    .... .... .... .... .... .... .... ...1 = Negotiate UNICODE: Set
  > Calling workstation domain: SPLINTERDMN
  > Calling workstation name: SERVER1
```

October 22 Patch Not Installed

https://twitter.com/splinter_code/status/1583555613950255104

RemotePotato0 - the ?accidental? fix



Home › Windows › Windows IT Pro Blog › DCOM authentication hardening: what you need to know
Back to Blog
‹ Newer Article
Older Article ›

DCOM authentication hardening: what you need to know

By David Zhu

Published Oct 19 2022 11:35 AM 98K Views

Hardening represents a means of investigating and reducing the number of systems across your organization with potential weaknesses, and then taking steps to securing them from malicious actors and their increasingly creative cyberthreats. Hardening has been applied across the industry to servers, software applications, operating systems, databases, networks, projects, repositories, services, policies, platforms, and more.

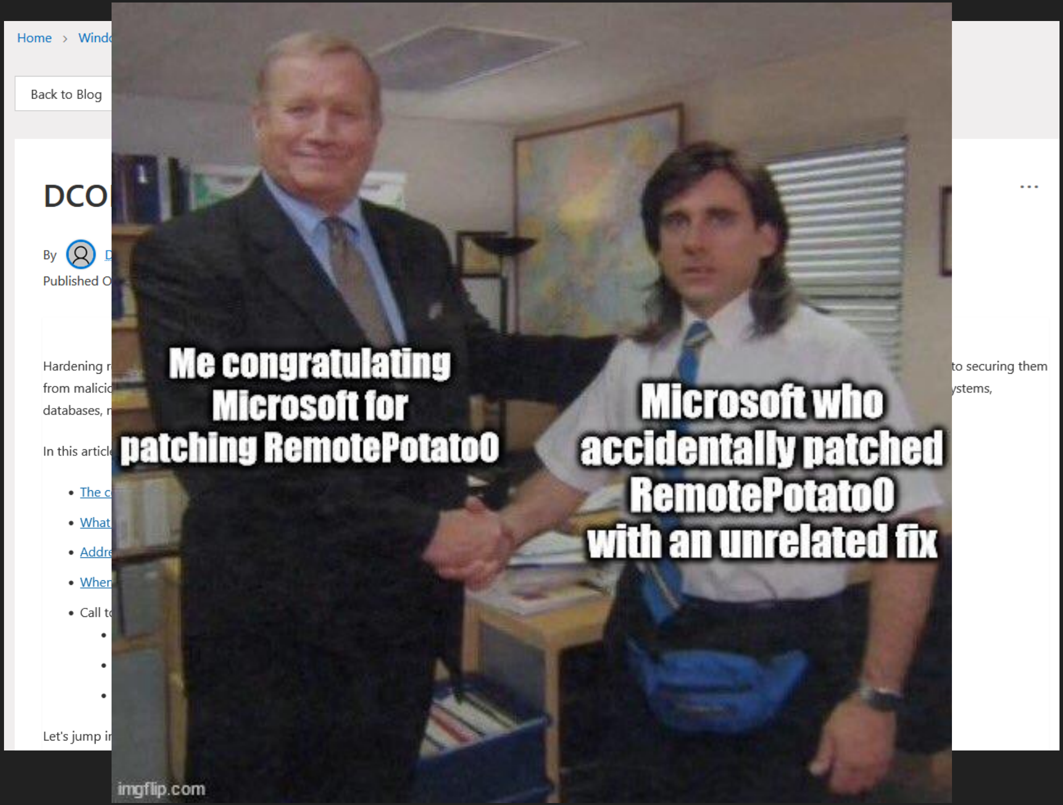
In this article, we'll explore how we're hardening Distributed Component Object Model (DCOM). Specifically:

- The context behind hardening
- What is DCOM and DCOM authentication hardening?
- Addressing critical vulnerabilities and why hardening matters
- When is DCOM authentication hardening happening?
- Call to action:
  - Keep your organization protected with the latest updates
  - Check your compatibility solutions as needed
  - Utilize troubleshooting help

Let's jump in!

https://techcommunity.microsoft.com/t5/windows-it-pro-blog/dcom-authentication-hardening-what-you-need-to-know/ba-p/3657154

RemotePotato0 - the ?accidental? fix


Home > Windo

Back to Blog

DCO

By D

Published O

Hardening r … to securing them

from malicio … ystems,

databases, r

In this article

- The c
- What
- Addre
- When
- Call to

Let's jump in

Me congratulating Microsoft for patching RemotePotato0
Microsoft who accidentally patched RemotePotato0 with an unrelated fix
imgflip.com

https://techcommunity.microsoft.com/t5/windows-it-pro-blog/dcom-authentication-hardening-what-you-need-to-know/ba-p/3657154

RemotePotato0 - exploitation scenarios

- ➔ Relay to an LDAP remote server with cross session activation

- ➔ Steal NTLMv2 response “hash” from a logged on user in another session for offline password cracking

- ➔ Relay to a remote SMB server with cross session activation

RemotePotato0 - after the ?accidental? fix

- ~~Relay to an LDAP remote server with cross session activation~~

- Steal NTLMv2 response “hash” from a logged on user in another session for offline password cracking

- Relay to a remote SMB server with cross session activation

RemotePotato0 - after the ?accidental? fix

➔ ~~Relay to an LDAP remote server with cross session activation~~

➔ Steal NTLMv2 response “hash” from a logged on user in another session for offline password cracking

➔ Relay to a remote SMB server with cross session activation

Demo 5 - RemotePotato0 relay to SMB

LocalPotato

- Logic bug we discover in NTLM local authentications:
  - Get a privileged user to authenticate on our server.
  - Start our client's NTLM authentication against a server service.
  - Intercept "B" context from the NTLM Type 2 message of our unprivileged client.
  - Get "A" context from the NTLM Type 2 message when the privileged client authenticates on our server.
  - Exchange context A and B, making privileged client authenticate as unprivileged, and vice versa.
  - Capture both NTLM Type 3 responses, and forward correctly to finish both authentications.
  - Due to the context swap bug in LSASS, our malicious client appears as the privileged user.

https://www.localpotato.com/localpotato_html/LocalPotato.html
https://github.com/decoder-it/LocalPotato

LocalPotato - attack flow

- ➔ Again, using the DCOM trigger locally to coerce a SYSTEM authentication, trick by James Forshaw [1]
- ➔ Targets the local SMB server to perform an arbitrary file write
- ➔ Specify the SPN “cifs/127.0.0.1” in the COM server authentication information [1] -> bypass NTLM Anti-Reflection SMB protection
- ➔ Exploit the context swap bug to authenticate as SYSTEM
- ➔ Hijack a dll from a privileged service and start the service, e.g. PrintConfig.dll

[1] https://googleprojectzero.blogspot.com/2021/10/windows-exploitation-tricks-relaying.html
https://www.localpotato.com/localpotato_html/LocalPotato.html
https://github.com/decoder-it/LocalPotato

Demo 6 - LocalPotato SMB edition

LocalPotato - CVE-2023-21746 fix

- The fix is in lsasrv.dll and function SsprHandleChallengeMessage()

- Ensures if ISC_REQ_UNVERIFIED_TARGET_NAME is set by the client with an SPN, it zeroed out to NULL

- Previously checked for "cifs/127.0.0.1" SPN to grant/deny access. Now, NULL SPN denies access

- Before patch, ISC_REQ_UNVERIFIED_TARGET_NAME was overlooked in NTLM authentication but was used by DCOM privileged client

https://msrc.microsoft.com/update-guide/en-US/vulnerability/CVE-2023-21746

LocalPotato - exploitation scenarios

- ➔ Context swap vs local SMB Server
- ➔ Context swap vs local HTTP Server
- ➔ Context swap vs custom authentication server which uses SSPI

LocalPotato - after the CVE-2023-21746 fix

- ~~Context swap vs local SMB Server~~

- Context swap vs local HTTP Server

- Context swap vs custom authentication server which uses SSPI

LocalPotato - CVE-2023-21746 fix

➔ The fix is in ls
SsprHandleChalle

➔ Ensures if ISC_R NAME is set by the
client with an S NULL

➔ Previously check L" SPN to grant/deny
access. Now, NUL

➔ Before patch, IS ET_NAME was
overlooked in NT was used by DCOM
privileged clien

https://msrc.microsoft.com/update-guide/en-US/vulnerability/CVE-2023-21746

RemotePotato0 - after the CVE-2023-21746 fix

- ~~Context swap vs local SMB Server~~
- Context swap vs local HTTP Server
- Context swap vs custom authentication server which uses SSPI

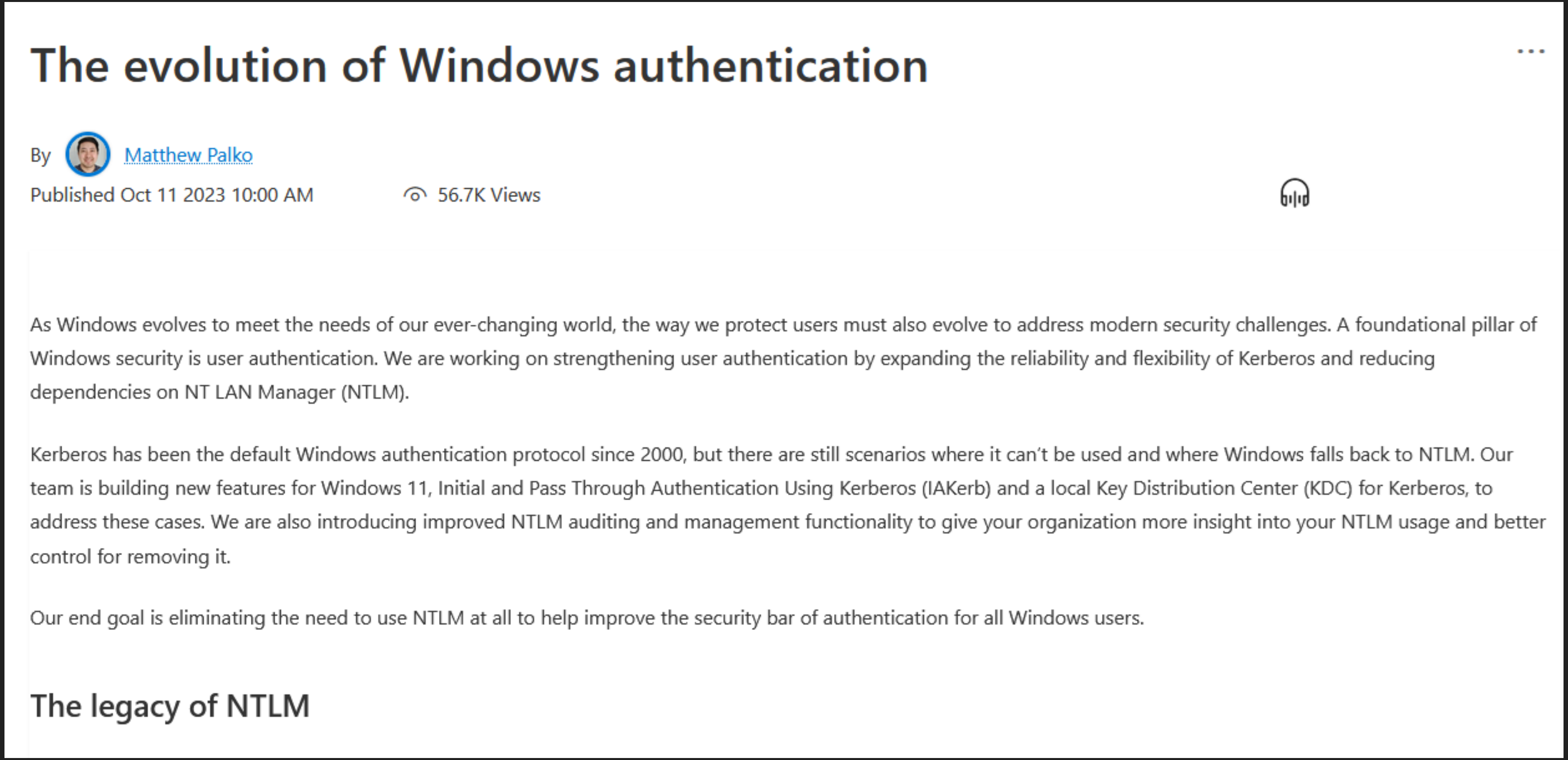
Demo 7 - LocalPotato HTTP/WebDAV edition

LocalPotato - Disclosure

- Context swap vs SMB (CVE-2023-21746)
  - Bounty awarded: 2.000 $
  - Time of fix ~3 months, well done 👍

- Context swap vs HTTP/WebDAV (CVE-404-NotFound)
  - Bounty awarded: 2.000 $
  - After 1 month: “We were having extensive internal conversations regarding your report … we are downgrading this report to a Moderate severity.” MSRC

LocalPotato - Microsoft will kill NTLM?



The evolution of Windows authentication

By Matthew Palko

Published Oct 11 2023 10:00 AM 56.7K Views

As Windows evolves to meet the needs of our ever-changing world, the way we protect users must also evolve to address modern security challenges. A foundational pillar of Windows security is user authentication. We are working on strengthening user authentication by expanding the reliability and flexibility of Kerberos and reducing dependencies on NT LAN Manager (NTLM).

Kerberos has been the default Windows authentication protocol since 2000, but there are still scenarios where it can’t be used and where Windows falls back to NTLM. Our team is building new features for Windows 11, Initial and Pass Through Authentication Using Kerberos (IAKerb) and a local Key Distribution Center (KDC) for Kerberos, to address these cases. We are also introducing improved NTLM auditing and management functionality to give your organization more insight into your NTLM usage and better control for removing it.

Our end goal is eliminating the need to use NTLM at all to help improve the security bar of authentication for all Windows users.

The legacy of NTLM

https://techcommunity.microsoft.com/t5/windows-it-pro-blog/the-evolution-of-windows-authentication/ba-p/3926848

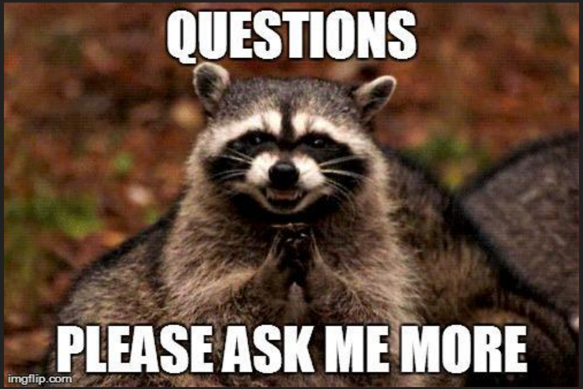
Conclusion

- Potatoes broke the boundaries!
  - Safety
  - Security

- Most MS fixes were always “partial”

- Future NTLM disablement will stop specific relay based attacks
  - What about Loopback authentication?

- Will potatoes be still alive and kicking?



Thank you for your attention!

@splinter_code

splintercod3@gmail.com