blackhat Asia 2023

MAY 11-12

BRIEFINGS

PPLdump Is Dead. Long Live PPLdump!

Selastic security labs

Gabriel Landau Principal, Elastic





Gabriel Landau is a principal at Elastic Security. His public research includes Process Ghosting, AV sandboxing attacks, Kernel Mode Threats and Practical Defenses (Black Hat USA), Hide Your Valuables -Mitigating Physical Credential Dumping Attacks (Shmoocon), PPLGuard, and CI Spotter. His non-public work includes endpoint protections, exploit mitigation, product and DRM evaluation, and malware reversing. Though he mostly wears blue these days, his heart will always be red.









- Introduction
 - What is a protected process?
 - \circ Implementation
- Attacks
 - \circ Historical
 - Current
- New Research
 - Novel Attack
 - Chaining Exploits
 - Mitigation





Protected Process (PP)

- Introduced in Windows 8
- Process hardened against code injection and memory tampering
- Created to isolate DRM processing from piracy tools with admin rights
- Will only load specially-signed code (EXEs/DLLs)
 No DLL side-loading
- Handles are hardened:
 - No PROCESS_VM_WRITE, THREAD_SET_CONTEXT, etc
- Also protects System, Registry, and and System Guard Runtime processes

lemory	Environment	Handles	GPU	Disk and	Network	Comment			
General	Statistics	Performan	nce	Threads	Token	Modules			
File									
	NT Kernel 8	System							
	(Verified) M	icrosoft Wind	ows						
Version	n: 10.0.19041	.2728							
Image	file name:					_			
C:\W	indows\system	132\ntoskrnl.e	xe			6			
Proces	S								
Comma	ind line:	N/A				2			
Curren	t directory:	N/A							
Started	f: [5 days and 2	hours a	ago (11:29:2	0 AM 3/18/	2023)			
PEB ad	dress:	0x0			Image typ	e: 64-bit			
Parent	: [System Idle P	rocess	(0)		2			
Mitigati	on policies:	N/A				Details			
Protect	tion: Full			Permission	s Terr	minate			





Protected Process Light (PPL)

- Introduced in Windows 8.1 as an extension of PP
- Similar signature requirements and process/thread HANDLE hardening
- Protect OS internals and AV from tampering
 - CSRSS highly trusted by kernel 0
 - LSASS credential dumping Ο
 - SCM service control manager Ο
 - AntiMalware prevent trivial termination of AV Ο
- Later extended to prevent application tampering
 - Hyper-V Shielded VMs 0
- The rest of this talk is about PPL

Statistics Performance Threads Token Modules Memory File Local Security Authority Process (Verified) Microsoft Windows Publisher Version: 10.0.19041.2130 Image file name: C: \Windows\\$ystem32\\sass.exe Process Command line: C: \WINDOWS\\$ystem32\\sass.exe Process Current directory: C: \WINDOWS\\$ystem32\ Started: 3 minutes and 45 seconds ago (11:10:08 AM 3/23/20) PEB address: 0x6b 12800000 Image type: 64-bit Parent: wininit.exe (532) Mitigation policies: DEP (permanent); ASLR (high entropy); Details Protection: Light (Lsa)	nvironment	Handle	es	Services	GPU	Disk and	d Network	k C	Comment	
Local Security Authority Process (Verified) Microsoft Windows Publisher Version: 10.0.19041.2130 Image file name: C:\Windows\System32\sass.exe C:\Windows\System32\sass.exe Process Command line: C:\WINDOWS\system32\sass.exe Current directory: C:\WINDOWS\system32\ Started: 3 minutes and 45 seconds ago (11:10:08 AM 3/23/20) PEB address: 0x6b 12800000 Image type: 64-bit Parent: wininit.exe (532) Mitigation policies: DEP (permanent); ASLR (high entropy); Details	eneral S	tatistics	Per	formance	Threads	Token	Modu	les	Memory	
Command line: C:\WINDOWS\system32\sass.exe Current directory: C:\WINDOWS\system32\ Started: 3 minutes and 45 seconds ago (11:10:08 AM 3/23/20) PEB address: 0x6b12800000 Image type: 64-bit Parent: wininit.exe (532) Mitigation policies: DEP (permanent); ASLR (high entropy);	Version: Image file	(Verified) 10.0.190 name:	<u>Micr</u> 41.2	osoft Wind 130	ows Publish	<u>ier</u>				
Started: 3 minutes and 45 seconds ago (11:10:08 AM 3/23/20 PEB address: 0x6b12800000 Image type: 64-bit Parent: wininit.exe (532) Mitigation policies: DEP (permanent); ASLR (high entropy);		line:	C:		5\ <mark>system</mark> 32	2¥sass.e>	ĸe		9	
PEB address: 0x6b12800000 Image type: 64-bit Parent: wininit.exe (532)	Current d	rectory:	C:	WINDOWS	S\system32	2\				
Parent: wininit.exe (532) Mitigation policies: DEP (permanent); ASLR (high entropy); Details	Started:		31	minutes and	and 45 seconds ago (11:10:08 AM 3/23/20					
Mitigation policies: DEP (permanent); ASLR (high entropy); Details	PEB addre	ss:	0x	6b1280000	00		Image t	ype:	64-bit	
	Parent:		wi	ninit.exe (S	32)				0	
	Mitigation	policies:	DE	P (perman	ent); ASLR	(high en	tropy);	Det	ails	
	Protection	: Light (l	.sa)		P	ermissior	is To	ermina	ate	

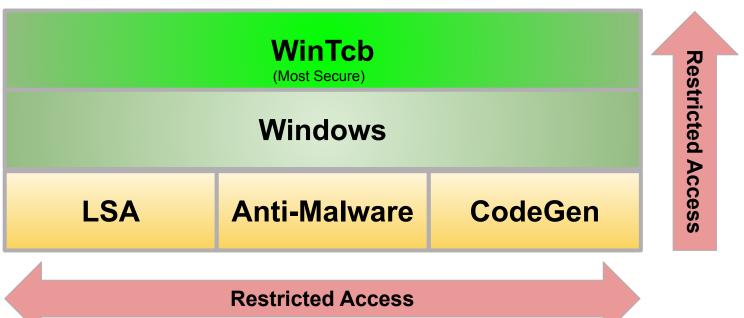




PPL Implementation - EPROCESS

- Structure within kernel EPROCESS
- Assigned at process creation
- Protection type
 - None, Protected Process, or PPL 0
- **Protection signer**
 - See diagram Ο

PPL Signers (Simplified)







Code Integrity - Signatures

SgrmBroker.exe Properties	and the second se	svchost.exe Properties					s.exe Pr	operties
Certificate	6	🔋 Certi	ificate			🔋 Cert	ificate	
General Details Certification Path		General	Details Cer	tification Path		General	Details	Certification
Show: <all></all>	~	Show:	<all></all>		~	Show:	< Al >	
Field Value	e	Field			Value	Field		
Enhanced Key UsageWindows TCB Component (1.3.Subject Key Identifier793165f0dbf15e5c04453d756.Subject Alternative NameDirectory Address:SERIALNUM.Authority Key IdentifierKeyID=a92902398e16c49778CRL Distribution Points[1]CRL Distribution Point: Distr.Authority Information Access[1]Authority Info Access: AccBasic ConstraintsSubject Type=End Entity, PatThumborint08647820d503fd505df763ab2		Image: Subject Key Identifier 01f0d3a457341838ebb3125 Image: Subject Alternative Name Directory Address:SERIALNU Image: Subject Alternative Name Directory Address:SERIALNU Image: Authority Key Identifier KeyID=a92902398e16c4977 Image: CRL Distribution Points [1]CRL Distribution Point: Distribution Point			Protected Process Light Verific. 01f0d3a457341838ebb31253 Directory Address:SERIALNUM. KeyID=a92902398e16c49778. [1]CRL Distribution Point: Distr. [1]Authority Info Access: Acc Subject Type=End Entity, Pat c60a14a6bd925780e9f0463ba	Subject Key Identifie Subject Alternative N Authority Key Identifie CRL Distribution Poin Authority Information Basic Constraints		r Identifier ernative Name ey Identifier ution Points oformation Ac raints
Windows TCB Component (1.3.6.1.4.1.3) Protected Process Verification (1.3.6.1.4. Windows System Component Verification Code Signing (1.3.6.1.5.5.7.3.3)	.1.311.10.3.24)	Window		ponent Verific	(1.3.6.1.4.1.311.10.3.22) cation (1.3.6.1.4.1.311.10.3.6)	Window Window	ws TCB Co ws System	ess Light Verif omponent (1 n Component .3.6.1.5.5.7.



5	

on Path	
	~
	Value
	Protected Process Light Verific.
	7d3af1a3055c18fdf39399016
ne	Directory Address:SERIALNUM.
r:	KeyID=a92902398e16c49778
	[1]CRL Distribution Point: Distr.
Access	[1]Authority Info Access: Acc
	Subject Type=End Entity, Pat
	e94a68h056ce2fa8ah046a84f

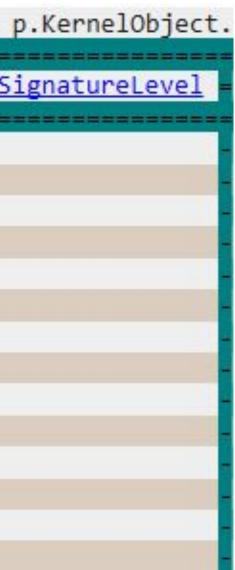
1.3.6.1.4.1.311.10.3.23) nt Verification (1.3.6.1.4.1.311.10.3.6) 7.3.3)



PPL Implementation - EPROCESS

	= <u>Name</u>	Type	Signer	<u>SectionS</u>
[0x4]	- System	0x2	- 0x7	0xc
[0x8c]	- Registry	- 0x2	- 0x7	0x0
[0x970]	- SgrmBroker.exe	- 0x2	- 0x6	0x8
[0x1d8]	- smss.exe	- 0x1	- 0x6	0x8
[0x22c]	- csrss.exe	- 0x1	- 0x6	0x8
[0x278]	- wininit.exe	- 0x1	- 0x6	0x8
[0x280]	- csrss.exe	- 0x1	- 0x6	0x8
[0x2cc]	- services.exe	- 0x1	- 0x6	0x8
[0xba8]	 svchost.exe 	- 0x1	- 0x5	0x8
[0x11bc]	- svchost.exe	- 0x1	- 0x5	0x8
[0x21f8]	 SecurityHealthService.exe 	- 0x1	- 0x5	0x8
[0x21dc]	- elastic-endpoint.exe	- 0x1	- 0x3	0x8
[0x3a0]	- svchost.exe	- 0x0	- 0x0	0x0
[0x3c8]	 fontdrvhost.exe 	- 0x0	- 0x0	0x8







Processes and Thread Protection

- Process and Thread Hardening
 - Read/write access rights blocked to less-privileged callers
 - No PROCESS_TERMINATE, PROCESS_VM_WRITE, PROCESS_VM_READ, etc.
 - Checked in kernel by RtlTestProtectedAccess
 - No exceptions for SeDebugPrivilege
 - New limited-access rights
 - PROCESS_QUERY_LIMITED_INFORMATION, PROCESS_SET_LIMITED_INFORMATION
 - THREAD_QUERY_LIMITED_INFORMATION, THREAD_SET_LIMITED_INFORMATION





Processes and Thread Protection

PS C:∖Wi	ndows\System	32> Get-Nt	oken	Select User	, Integr	rityLev	el				
User		IntegrityLe									
NT AUTHO	RITY\SYSTEM	Sys									
PS C:∖Wi	ndows\System	132> (Get-Nt	:Token).	Groups Whe	re {\$N	Wame -1	ike "*Tr	ustedIns	taller"}		
Name		Att	ributes								
NT SERVI	CE\TrustedIr	staller Ena	abledByD	efault, Ena	bled, Ow	mer					
PS C:∖Wi	ndows\System	132> (Get-Nt	:Token).	Privileges	where	{\$Na	me -eq "	5eDebugP	rivilege	"}	Select
	Ena	bled									
 SeDebugP	rivilege	True									
A REAL PROPERTY AND A REAL	ndows\Systen ndows\Systen							imitedIn	formatio	n	
Handle N	ame			ProtectFro	mClose						
3244 s	 ervices.exe	Process		False							



Name, Enabled



Resource Protection

• Token Trust Level

• New token attribute which indicates the trust level of the acting process or thread

PS C:\Windows\System32>	<pre>\$explorer = Get-NtProcess -Name explorer.exe -Access Qu</pre>	lery
PS C:\Windows\System32>	(Get-NtToken - Process \$explorer).TrustLevel	
PS C:\Windows\System32>		
PS C:\Windows\System32>	<pre>\$services = Get-NtProcess -Name services.exe -Access Qu</pre>	lery
PS C:\Windows\System32>	(Get-NtToken - Process \$services).TrustLevel	
Name	Sid	
TRUST LEVEL\ProtectedLig	ght-WinTcb S-1-19-512-8192	



yLimitedInformation yLimitedInformation



Resource Protection

Trust Labels

- New System Access Control List Entry (SACL ACE) type that allow trust level test for any securable object Ο
- Examples: Ο
 - Protecting KnownDlls against modification by malicious administrators
 - Protect PPL process tokens against sandboxing by malicious administrators*

PS C:\Windows\Syst	tem32>	(Get-NtDirectory \KnownDlls).Secu	rityDescriptor.Sacl
Туре	User		Flags	Mask
 ProcessTrustLabel	TRUST	LEVEL\ProtectedLight-WinTcb	None	00020003
				ervices.exe -Access QueryLimitedInformat SecurityDescriptor.Sacl Where {\$Type
Туре	User		Flags	Mask
 ProcessTrustLabel	TRUST	LEVEL\ProtectedLight-WinTcb	None	0002001E

* Recent addition. See my work: https://www.elastic.co/security-labs/sandboxing-antimalware-products









- Introduction
 - What is a protected process?
 - \circ Implementation
- Attacks
 - \circ Historical
 - Current
- New Research
 - Novel Attack
 - Chaining Exploits
 - Mitigation





Attack: Cached Signing Level

- NtSetCachedSigningLevel race condition
- CI caches signing information for performance reasons
- Cache entries are automatically invalidated by NTFS if file is modified
- Race condition in CI allowed file to be modified before cache entry is finalized
- Fixed as CVE-2017-11830







Attack: Counterfeit \KnownDlls via Silos

- Windows containers (aka silos) are similar to docker containers.
- Containers created ability to "chroot" a process into a new object manager namespace
- "chroot" ability creates a unique namespace for all named objects including drives, network shares, events, mutexes, named pipes, etc
- \KnownDlls section object cache is part of the Object Manager namespace Protected by trust label so this cannot normally be modified by attackers 0
- Windows treats \KnownDlls as verified no additional checks before loading into PPL
- Attacker can create a counterfeit KnownDlls directory then spawn a new "chrooted" PPL, which will use their KnownDlls, loading DLLs specified therein
- Fixed in 7/2022 by removing KnownDlls support from PPL





Attack: Script Engine COM Hijack

- Some script interpreter DLLs will automatically load scripts specified in the registry
- Use DotNetToJScript to convert .NET payload to JS
- Find COM used by PPL, and hijack its registry run a script interpreter DLL instead
- Script interpreter loads attacker JS based on registry key, which loads .NET payload
- Fixed in 1803 by blocking script interpreters from loading into PPL
 - New function nt!CipMitigatePPLBypassThroughInterpreters blocks PPL from loading interpreter DLLs Ο





Attack: Bring Your Own Vulnerable EXE

- Windows Error Reporting process memory dumper (WerFaultSecure) encrypts dumps to protect PP and PPL confidentiality
- Bug in Windows 8.1 build can lead to creation of unencrypted dumps
- Microsoft fixed the WerFaultSecure bug ~2014
- Latest Win11 will still run old vulnerable builds as WinTcb-Full
 - Easy RunAsPPL LSASS defeat 0







Attack: COM IRundown::DoCallback

- Use vulnerable Windows 8.1 WerFaultSecure to dump process and find secrets and addresses
- Use COM hijack to exploit undocumented COM feature: IRundown::DoCallback
- Use acquired secrets and addresses to call an arbitrary function within WerFault.exe
- Call existing code in process, achieving arbitrary write primitive
- Use arbitrary write primitive to overwrite LdrpKnownDllDirectoryHandle
- With counterfeit KnownDlls installed, attack proceeds like DefineDosDevice exploit





Attack: AntiMalware Blight

- ELAM Early Launch AntiMalware Driver
 - Driver containing certificate hashes
 - Special signature from Microsoft
 - Any certificate listed in an ELAM driver can sign a file to run as AntiMalware-Light
- Overly-permissive ELAM
 - Some Antimalware vendors included hashes of certificates third-party certificates
 - Microsoft didn't vet certificate lists before signing ELAM drivers
- There are many overly-permissive ELAM drivers
 - Microsoft CAs included
- Example: You can run msbuild.exe as AntiMalware-Light with arbitrary parameters

- vvab	oot.sys Properties		2
Certi	ficate		>
General	Details Certification Path		
Show:	< <mark>All</mark> >	~	
Field		Value	^
Pu	blic key	RSA (2048 Bits)	
Pu	blic key parameters	05 00	
En	hanced Key Usage	Early Launch Antimalware Driv	
	bject Key Identifier	cae78438d42319ee552a37bb	
	bject Alternative Name	Directory Address:SERIALNUM	
	thority Key Identifier	KeyID=e6fc5f7bbb220058e47	
CR CR	L Distribution Points	[1]CRL Distribution Point: Distr	5





Attack: DefineDosDevice Bug

e <u>V</u> iew <u>O</u> bject <u>F</u> ind <u>E</u> xtras Pl	ugins <u>H</u> elp				
🍫 🌐 🔍					
🛅 NLS	👻 Name	Туре	Additional Information		
> 🛅 Windows	& BTH#MS_BTHPAN#7&20f38eb4&0&2#{cac88484	SymbolicLink	\Device\00000091		
GLOBAL??	& BTH#MS_RFCOMM#7&20f38eb4&0&0#{9e16888d	SymbolicLink	\Device\000008f		
RPC Control	🔗 BthPan	SymbolicLink	\Device\BthPan		
> C BaseNamedObjects	êc:	SymbolicLink	\Device\HarddiskVolume3		
	CdRom0	SymbolicLink	\Device\CdRom0		
	CimfsControl	SymbolicLink	\Device\cimfs\control		

- The DefineDosDevice API defines, redefines, or deletes MS-DOS device names
- Implemented via RPC to WinTcb-PPL CSRSS
 - Remember this is the highest level of PPL
- TOCTOU enables attackers to trick CSRSS into creating entries in \KnownDlls
- Attacker can inject entries into KnownDlls, which PPL will load without verification
- Publicly documented in 2018 by James Forshaw
- Turnkey implementation released in April 2021 by Clément Labro as PPLdump
- Fixed in 7/2022 by removing KnownDlls support from PPL







Attack: COM Proxy Type Library Confusion

- .NET Runtime Optimization Service runs as CodeGen PPL and hosts COM service
- Modify COM proxy configuration for service to trigger type confusion
- Use type confusion to trigger arbitrary write, replacing KnownDlls handle with counterfeit directory that is pre-loaded with attacker's DLL
- With counterfeit KnownDlls installed, attack proceeds like DefineDosDevice exploit
- Leverage CodeGen PPL access to create a signing cache entry making any DLL as trusted so it can be side-loaded into WinTcb PPL (highest level)
- Variant implemented as turnkey <u>PPLmedic</u> tool in March 2023 by Clément Labro
- Microsoft: KnownDlls handle mitigation coming in June 2023







- Introduction
 - What is a protected process?
 - \circ Implementation
- Attacks
 - \circ Historical
 - Current
- New Research
 - Novel Attack
 - Chaining Exploits
 - Mitigation



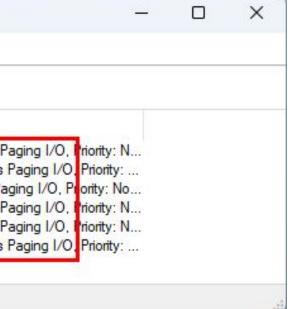


Planning the Attack

- Attacks so far focus on:
 - CachedSigningLevel
 - \circ KnownDlls
 - \circ COM
- Let's try a different approach
 - Bait and Switch aka Time of Check, Time of Use (TOCTOU)

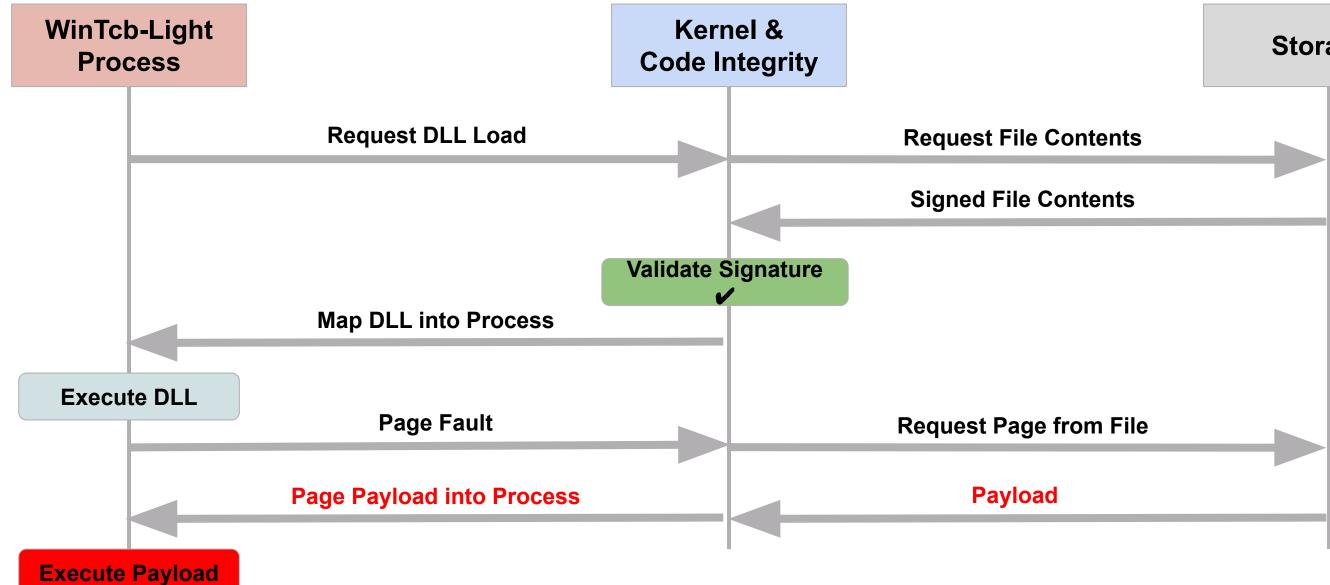
Pro	cess Monitor - S	Sysinte	ernals: www.s	sysinternals.com		
<u>File</u>	<u>E</u> dit E <u>v</u> ent F	i <u>l</u> ter	<u>T</u> ools <u>O</u> p	tions <u>H</u> elp		
	3 🖸 🗖	Ŵ	YZ	1 💿 品 🕹 🤈 🖊 📑	-	
Time	Process Name	PID	Operation	Path	Result	Detail
10:42:	services.exe	824	🐂 ReadFile	C:\Windows\System32\EventAggregation.c	III SUCCESS	Offset: 73,728, Length: 4,096, I/O Flags: Non-cache, Paging I/O, Synchronous
10:42:	services.exe	824	ReadFile	C:\Windows\System32\EventAggregation.c	III SUCCESS	Offset: 53,248, Length: 16,384, I/O Flags: Non-cached, Paging I/O, Synchronous
10:42:	services.exe	824	🐂 ReadFile	C:\Windows\System32\EventAggregation.c		Offset: 4,096, Length: 4,096, I/O Flags: Non-cached Paging I/O, Synchronous P
10:42:	services.exe	824	ReadFile	C:\Windows\System32\EventAggregation.c	III SUCCESS	Offset: 4,096, Length: 32,768, I/O Flags: Non-cachee, Paging I/O, Synchronous I
10:42:	services.exe	824	ReadFile	C:\Windows\System32\EventAggregation.c	III SUCCESS	Offset: 40,960, Length: 4,096, I/O Flags: Non-cached, Paging I/O, Synchronous I
10:42:	services.exe	824	🐂 ReadFile	C:\Windows\System32\EventAggregation.c	III SUCCESS	Offset: 36,864, Length: 12,288, I/O Flags: Non-cached, Paging I/O, Synchronous
Showing	g 6 of 99,45 <mark>4</mark> ever	nts (0.	0060%)	Backed by virtual memory		







CI TOCTOU: Planning the Attack







Storage



CI TOCTOU: Page Hashes

• Page hashes present in services.exe but not EventAggregation.dll

C:\Windows\System32>signtool verify /v /ph services.exe | grep -A10 "Page hash" Page hashes: 0x00000000 973911F5DEABEFCF45A87E948DE1DF57DBE1C6C22D12559F2754862CEC5BB516 0x00000400 40048953BD60329AC1486A957A4EEC5D3A14ABC4E0E359BBAD063097495C3AB9 0x00001400 DA4D752F6C5EAA717CD127E8C4D4491F1D87CD2D73E2B7F38BC8A01336FE76E4 0x00002400 A8A85175F216A21BF270A65CCF26CD623E95FC88DA08FE8747606A16710A655F 0x00003400 A95303468AB638FD630C643265C819C14224805B954CAE98701D9428A2C6C1E9 0x00004400 161A709452170F6659EE9639432402D4E3454A31F5F0F5AAA3E3E29D9E5249C1 0x00005400 B8ADD3652917342812D22A573E75AFC85060321E30F15D7895058C2152847750 0x00006400 0755C750AD27D96B7F2D68D83216C4183B625E6CF768DF98F1C4F62F340D1D1C 0x00007400 A3A63DA8FB35B218BA9E3F116789D81D84CEB35B4F3FFE1D5A1003E5A9DA07AA 0x00008400 3A24070DDBE08C046B1A4BC2B183CF41D35EF623C0E0E2F3058E1EDE3BCD2C87

C:\Windows\System32>signtool verify /a /v /ph EventAggregation.dll | grep -A10 "Page hash" SignTool Warning: No page hashes are present.





CI TOCTOU: Hunting for Local Paging

- Start simple run services.exe as WinTcb-PPL
 - \circ X No file reads, and no paging I/O

Process Monitor - Sy	vsinternals: www.sysinterna	als.com		
<u>File E</u> dit E <u>v</u> ent Fil	ter <u>T</u> ools <u>O</u> ptions <u>H</u>	<u>l</u> elp		
	💼 🍸 💋 🎯	사 🕹 🖉 🗸 📜 📰 🔁 🗞	\mathbf{N}	
Time Process Name	PID Operation	Path	Result	Detail
1:35:5 Transferrences.exe	5892 coad Image	C:\Windows\System32\services.exe	SUCCESS	Image Base: 0x7ff69b540000, Image Size: 0xb6000
1:35:5 Tervices.exe	5892 🐂 Create File	C:\Windows\System32\EventAggregation.dl	SUCCESS	Desired Access: Read Attributes, Disposition: Open,
1:35:5 Tervices.exe	5892 📻 QueryBasicInform	nationFile C:\Windows\System32\EventAggregation.dl	I SUCCESS	Creation Time: 5/6/2022 10:19:20 PM, LastAccess Ti
1:35:5 Tervices.exe	5892 🐂 CloseFile	C:\Windows\System32\EventAggregation.dl	SUCCESS	
1:35:5 Tervices.exe	5892 🐂 Create File	C:\Windows\System32\EventAggregation.dl	I SUCCESS	Desired Access: Read Data/List Directory, Execute/
1:35:5 Tervices.exe	5892 🐂 CreateFileMappin	g C:\Windows\System32\EventAggregation.dl	FILE LOCKED WITH ONLY READERS	SyncType: SyncTypeCreateSection, PageProtection
1:35:5 Tervices.exe	5892 📻 Create File Mappin	g C:\Windows\System32\EventAggregation.dl	I SUCCESS	SyncType: SyncTypeOther
1:35:5 Tervices.exe	5892 CLoad Image	C:\Windows\System32\EventAggregation.dl	I SUCCESS	Image Base: 0x7fffda250000, Image Size: 0x16000
1:35:5 Tervices.exe	5892 📻 CloseFile	C:\Windows\System32\EventAggregation.dl	I SUCCESS	
Showing 9 of 150,614 even	nts (0.0059%)	Backed by virtual memory		

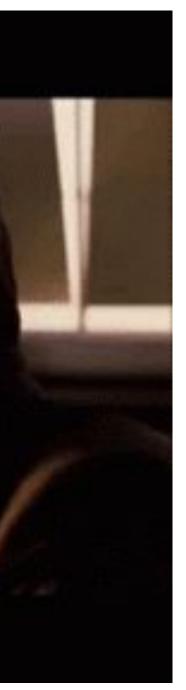


	_	×
0		
0 n, Options: Open Repa Time: 3/24/2023 1:35		
e/Traverse, Synchronia on: PAGE_EXECUTE_		
)		
		115



WE NEED TO GO DEEPER







CI TOCTOU: Hunting for Remote Paging

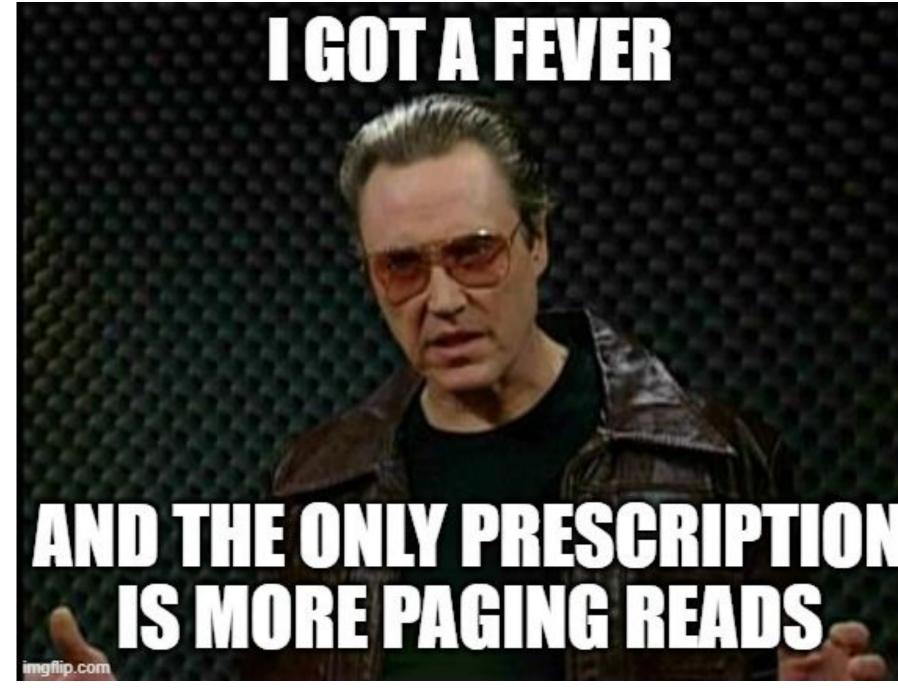
- What about SMB? Replace EventAggregation.dll with a symlink to loopback SMB
- We can see a paging operation over SMB

Process	s Monitor - Sysinte	ernals: www.sysinternals.co	m		
<u>F</u> ile <u>E</u> dit	E <u>v</u> ent Fi <u>l</u> ter	<u>T</u> ools <u>O</u> ptions <u>H</u> elp			
6	🖸 🗔 🗖	700 🗛	🦻 🔎 🖊 📑 🔚 🖵 📽 🗛		
Time of D	Process Name	PID Operation	Path	Result	Detail
AN ALL AND AN ALL AND A	services.exe	8180 🐂 CreateFile 8180 🐂 CloseFile	C:\Windows\System32\EventAggregation.dll C:\Windows\System32\EventAggregation.dll	SUCCESS SUCCESS	Desired Access: Read Attributes, Disposition: Op
	services.exe	8180 CreateFile	C:\Windows\System32\EventAggregation.dll	REPARSE	Desired Access: Read Data/List Directory, Exec
	services.exe	8180 Recreate File 8180 Recreate File Mapping	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak \\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS FILE LOCKED WI.	Desired Access: Read Data/List Directory, Exec Sync Type: Sync TypeCreate Section, PageProtect
3:21:19.76	services.exe	8180 📻 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 0, Length: 90,112, I/O Flags: Non-cached
3:21:19.77	services.exe services.exe services.exe	8180 ReadFile 8180 QueryEAFile 8180 SetEAFile	C:\Windows\System32\EventAggregation.dll.bak \\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak \\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS ACCESS DENIED ACCESS DENIED	Offset: 0, Length: 90,112, I/O Flags: Non-cache
	services.exe	8180 ReateFileMapping 8180 CreateFileMapping	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak \\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS SUCCESS	SyncType: SyncTypeOther Image Base: 0x7ffdf0450000, Image Size: 0x160
	services.exe	8180 CloseFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	image base. uk muro450000, image Size. uk roo
Showing 12	of 25 events (48%)) Back	ed by virtual memory		



	7 <u>~~</u>		×
Open, Options: Op	pen For Ba	ackup, Op	en R
ecute/Traverse, S ecute/Traverse, S ection: PAGE_E	Synchroniz	e, Dispos	ition:
ed, Paging I/O, S	COLUMN 2 NOT THE OWNER WATER OF THE OWNER		The second se
ed			
5000			
(201) S			-11









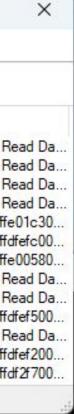
CI TOCTOU: Oplock Candidates

- Can we slow down process launch to allow time for paging?
- What about an opportunistic lock (oplock)?
 - Non-cooperative NTFS/SMB file locking mechanism 0
- Let's look for a CreateFile operation that we can interrupt

Proc	cess Monitor - S	ysintern	als: ww	w.sysinternals.	com		- 0
<u>File</u>	dit E <u>v</u> ent Fi	i <u>l</u> ter <u>T</u> o	ools	Options <u>H</u> el	p		
	3 🖸 🗔		V I	2 💿 🖥	ቴ 😼 🔎 🖊 📑 📰 🖾 🖾		
Time	Process Name	PID	TID	Operation	Path	Result	Detail
1:32:1	services.exe	7132	9652	🐂 CreateFile	C:\Windows\System32\rpcrt4.dll	SUCCESS	Desired Access: R
1:32:1	services.exe	7132		CreateFile	C:\Windows\System32\dpapi.dll	SUCCESS	Desired Access: R
1:32:1	services.exe	7132	9596	📻 Create File	C:\Windows\System32\EventAggregation.dll	REPARSE	Desired Access: R
1:32:1	services.exe	7132	8564	🐂 Create File	C:\Windows\System32\sechost.dll	SUCCESS	Desired Access: R
1:32:1	services.exe	7132	9652	C Load Image	C:\Windows\System32\rpcrt4.dll	SUCCESS	Image Base: 0x7ffe
1:32:1	services.exe	7132	8472	C Load Image	C:\Windows\System32\dpapi.dll	SUCCESS	Image Base: 0x7ffc
1:32:1	services.exe	7132	8564	C Load Image	C:\Windows\System32\sechost.dll	SUCCESS	Image Base: 0x7ffe
1:32:1	services.exe	7132	9596	🐂 Create File	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Desired Access: R
1:32:1	services.exe	7132	9652	📻 CreateFile	C:\Windows\System32\cfgmgr32.dll	SUCCESS	Desired Access: R
1:32:1	services.exe	7132	9652	C Load Image	C:\Windows\System32\cfgmgr32.dll	SUCCESS	Image Base: 0x7ffc
1:32:1	services.exe	7132	8472	📻 CreateFile	C:\Windows\System32\devobj.dll	SUCCESS	Desired Access: R
1:32:1	services.exe	7132	8472	C Load Image	C:\Windows\System32\devobj.dll	SUCCESS	Image Base: 0x7ffc
1:32:1	services.exe	7132	9596	C Load Image	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Image Base: 0x7ffc
Showing	13 of 16 events	(81%)		Ba	cked by virtual memory		









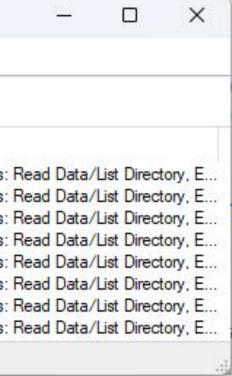
CI TOCTOU: Oplock Results

- Set an oplock on devobj.dll and launch services.exe
- IRP has no result operation is still pending

Process	Monitor - Sysinte	ernals: www.sy	sinternals.com	m			
<u>File</u> <u>E</u> dit	E <u>v</u> ent Fi <u>l</u> ter	<u>T</u> ools <u>O</u> pti	ons <u>H</u> elp				
d 📙	🖸 🗔 💼	7 💋	◎ Å	🦻 P 🖊 🔡 🖬	🖓 🖍		
Time of D	Process Name	PID Operat	ion	Path		Result	Detail
3:36:13.47	services.exe	3552 🐂 Crea	teFile	C:\Windows\System32\kernel32.dll		SUCCESS	Desired Access:
3:36:13.48	services.exe	3552 🐂 Crea	teFile	C:\Windows\System32\rpcrt4.dll		SUCCESS	Desired Access:
3:36:13.48	services.exe	3552 📻 Crea	teFile	C:\Windows\System32\dpapi.dll		SUCCESS	Desired Access:
3:36:13.48	services.exe	3552 📻 Crea	teFile	C:\Windows\System32\EventAggre	gation.dll	REPARSE	Desired Access:
3:36:13.48	services.exe	3552 🐂 Crea	teFile	C:\Windows\System32\sechost.dll		SUCCESS	Desired Access:
3:36:13.48	services.exe	3552 🐂 Crea	teFile	C:\Windows\System32\cfgmgr32.dl	í	SUCCESS	Desired Access:
3:36:13.48	services.exe	3552 📻 Crea	teFile	C:\Windows\System32\devobj.dll			Desired Access:
3:36:13.49	services.exe	3552 🐂 Crea	te File	\\127.0.0.1\C\$\Windows\System3	2\EventAggregation.dll.bak	SUCCESS	Desired Access:
Showing all	8 events		Backe	ed by virtual memory			









CI TOCTOU: Oplock Results

3: kd > k*** Stack trace for last set context - .thread/.cxr resets it # Child-SP Call Site RetAddr 00 ffffb88e`ac5be210 fffff807`7e4cb6c5 nt!KiSwapContext+0x76 01 ffffb88e`ac5be350 fffff807`7e4ccae7 nt!KiSwapThread+0xb05 02 ffffb88e`ac5be4a0 fffff807`7e4cf106 nt!KiCommitThreadWait+0x137 03 ffffb88e`ac5be550 fffff807`7e95be2c nt!KeWaitForSingleObject+0x256 nt!FsRtlCancellableWaitForMultipleObjects+0xcc 04 ffffb88e`ac5be8f0 fffff807`7e95bae7 05 ffffb88e`ac5be960 fffff807`822c16c8 nt!FsRtlCancellableWaitForSingleObject+0x27 Ntfs!NtfsWaitForOplockCompletionEvent+0x24 06 ffffb88e`ac5be9a0 fffff807`82256222 07 ffffb88e`ac5be9e0 fffff807`7e4d00a5 Ntfs!NtfsEsdCreate+0x272 08 ffffb88e`ac5bec60 fffff807`813d9f5b nt!IofCallDriver+0x55 09 ffffb88e`ac5beca0 fffff807`8140eff3 FLTMGR!FltpLegacyProcessingAfterPreCallbacksCompleted+0x15b 0a ffffb88e`ac5bed10 fffff807`7e4d00a5 FLTMGR!FltpCreate+0x323 0b ffffb88e`ac5bedc0 fffff807`7e8e2979 nt!IofCallDriver+0x55 0c ffffb88e`ac5bee00 fffff807`7e8de4f1 nt!IopParseDevice+0x8c9 0d ffffb88e`ac5befd0 fffff807`7e8dd4d2 nt!ObpLookupObjectName+0xae1 0e ffffb88e`ac5bf170 fffff807`7e8c1cf9 nt!ObOpenObjectByNameEx+0x1f2 nt!IopCreateFile+0x439 0f ffffb88e`ac5bf2a0 fffff807`7e8bdfc8 nt!NtOpenFile+0x58 10 ffffb88e`ac5bf360 fffff807`7e63e1e8 nt!KiSystemServiceCopyEnd+0x28 11 ffffb88e`ac5bf3f0 00007fff`dd26f2b4 ntdll!NtOpenFile+0x14 12 00000083 4a77f0f8 00007fff dd1e064c ntdll!LdrpMapDllNtFileName+0xe8 13 0000083 4a77f100 00007fff dd1e0bb8 ntdll!LdrpMapDllSearchPath+0x1d0 14 00000083 4a77f200 00007fff dd1e0f80 ntdll!LdrpProcessWork+0x148 15 00000083 4a77f460 00007fff dd1e0dbb 16 00000083 4a77f4b0 00007fff dd23236a ntdll!LdrpWorkCallback+0xbb ntdll!TppWorkpExecuteCallback+0x13a 17 00000083 4a77f4e0 00007fff dd205976 ntdll!TppWorkerThread+0x8f6 18 0000083 4a77f530 00007fff dcf626bd KERNEL 32!BaseThreadInitThunk+0x1d 19 00000083 4a77f810 00007fff dd22a9f8 1a 00000083 4a77f840 00000000 0000000 ntdll!RtlUserThreadStart+0x28







CI TOCTOU: Forcing Paging

- Where do we go from here?
- We have a frozen WinTcb PPL process. We want it to page-in code over the network.
- Can we page it out using EmptyWorkingSet?
 - X Requires PROCESS_SET_QUOTA, which we can't get
- What about paging out the whole OS?
 - Empty system working set and standby lists Ο
 - NtSetSystemInformation(SystemMemoryListInformation)*
 - Requires SeProfileSingleProcessPrivilege, which Admins have

* https://github.com/elastic/Silhouette/blob/main/2023-01%20Silhouette%20Shmoocon%20Presentation.pd;



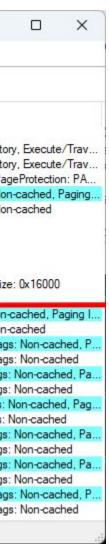




CI TOCTOU: Paged Reads

Process Monitor - Sysinternals <u>File Edit Event Filter Tool</u>				
0 🗄 🚺 🗟 💼 🛛		> 🖊 📑 🚍 🕏 🗞 🚹		
Relative Time Process Name	PID Operation	Path	Result	Detail
00:00:17.0969135 services.exe	508 📻 CloseFile	C:\Windows\System32\EventAggregation.dll	SUCCESS	
00:00:17.0971696 services.exe	508 CreateFile	C:\Windows\System32\EventAggregation.dll	REPARSE	Desired Access: Read Data/List Director
00:00:17.1006151 services.exe	508 CreateFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Desired Access: Read Data/List Director
00:00:17.1017799 services.exe	508 TreateFileMapping	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	FILE LOCKED WI	SyncType: SyncTypeCreateSection, Pag
00:00:17.1018504 services.exe	508 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 0, Length: 90,112, I/O Flags: Non
00:00:17.1018615 = services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 0, Length: 90,112, I/O Flags: Non
00:00:17.1027614 🔳 services.exe	508 🐂 Query EAFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak		
00:00:17.1057495 = services.exe	508 🐂 Query EAFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak		
00:00:17.1058253 🔳 services.exe	508 📻 SetEAFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	ACCESS DENIED	
0:00:17.1064822 🔳 services.exe	508 🐂 Create File Mapping	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	SyncType: SyncTypeOther
00:00:17.1066262 🔳 services.exe	508 c [®] Load Image	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Image Base: 0x7ffdf0450000, Image Size
00:00:17.1066861 = services.exe	508 📻 CloseFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	
00:00:51.1699161 = services.exe	508 📻 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 0, Length: 4,096, I/O Flags: Non-
00:00:51.1701225 services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 0, Length: 4,096, I/O Flags: Non-
00:00:51.1702783 services.exe	508 📻 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 53,248, Length: 16,384, I/O Flags
00:00:51.1702900 = services.exe	508 🐂 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 53,248, Length: 16,384, I/O Flags
00:00:51.1715041 services.exe	508 📻 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 73,728, Length: 4,096, I/O Flags:
00:00:51.1715132 = services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 73,728, Length: 4,096, I/O Flags:
00:00:51.1715971 services.exe	508 📊 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 4,096, Length: 4,096, I/O Flags: 1
00:00:51.1716052 services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 4,096, Length: 4,096, I/O Flags: N
00:00:51.1716701 services.exe	508 📻 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 4,096, Length: 32,768, I/O Flags:
00:00:51.1716803 = services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 4,096, Length: 32,768, I/O Flags:
00:00:51.1717834 services.exe	508 📊 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 40,960, Length: 4,096, I/O Flags:
00:00:51.1717897 = services.exe	508 🐂 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 40,960, Length: 4,096, I/O Flags:
00:00:51.1718510 services.exe	508 🐂 ReadFile	\\127.0.0.1\C\$\Windows\System32\EventAggregation.dll.bak		Offset: 36,864, Length: 12,288, I/O Flags
00:00:51.1718630 = services.exe	508 📷 ReadFile	C:\Windows\System32\EventAggregation.dll.bak	SUCCESS	Offset: 36,864, Length: 12,288, I/O Flags
showing 26 of 43 events (60%)	Backed by virtual	memory		







CI TOCTOU: Delivering the Payload

- Now that we can reliably force page faults, let's try to inject some code
 - Disable the local SMB server (LanManServer service) and reboot a.
 - b. Run local SMB server that serves two versions of EventAggregation.dll
 - First, serve original DLL for CI verification
 - Later, patch in special sauce over DIIMain for subsequent requests

```
# This payload requires a kernel debugger to view
# If you use this payload, type this in WinDbg afterwards:
# db @rip; dx @$curprocess->Name; dx @$curprocess->KernelObject->Protection
$Payload = "CC" + ("90" * 15) + ("CAFECODE" * 64)
```







CI TOCTOU: Code Execution

		30000003 (first chance)
0033:00007fff`addb1	1550 CC	int 3
5: kd> db @rip		
00007fff`addb1550	cc 90 90 90 90 90	0 90 90-90 90 90 90 90 90 90 90
00007fff`addb1560	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb1570	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb1580	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb1590	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb15a0	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb15b0	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
00007fff`addb15c0	ca fe c0 de ca fe	e c0 de-ca fe c0 de ca fe c0 de
5: kd> dx @\$curprod	cess->Name	
@\$curprocess->Name	: services.exe	
Length	: Øxc	
5: kd> dx @\$curprod	cess->KernelObject-	->Protection
@\$curprocess->Kerne	elObject->Protectio	Lon [Type: PS PROTECTION]
[+0x000] Level	: 0x61	I [Type: unsigned char]
		: 0x1 [Type: unsigned char]
		: 0x0 [Type: unsigned char]
		: 0x6 [Type: unsigned char]





CI TOCTOU: Removing the Reboot

- LanManServer configuration change is noisy. Can we remove the reboot?
 - a. X SMB port fixed. LanManServer takes it early in boot. No way to release it
 - b. X WebDAV file is read once at mapping and cached locally
- Cloud Filter API
 - a. Available by default in Client SKUs of 1709+
 - Create small/empty placeholder files marked with reparse tags b.
 - When read requests come, minifilter drive detects reparse tags and calls UM callback to request data C.
 - UM callback provides the requested file contents d.
 - You decide what bytes to serve to the client in your rehydration callback
 - e. Simple-to-use usermode API
 - No COM
 - No special signing requirements
 - James Forshaws provided working sample code. q.





CI TOCOTU: Putting it All Together

Final attack flow:

- Use CloudFilter to create an empty placeholder file with a callback function we control а.
- Redirect EventAggregation.dll to our placeholder through loopback SMB via symbolic link b.
- Set oplock on devobj.dll to interrupt process initialization C.
- Run the target PPL d.
- The target PPL attempts to load EventAggregation.dll e.
- CloudFilter callback fires to rehydrate placeholder f.
 - Serve up original EventAggregation.dll for CI verification
 - Page everything out by emptying working set and standby lists
 - Release oplock
- g. The PPL resumes and leads to paging reads over SMB, which are forwarded to the placeholder
- CloudFilter callback fires to rehydrate placeholder h.
 - Serve up patched copy of EventAggregation.dll
- The PPL executes our PIC payload inside services.exe as WinTcb-Light, which dumps the process of your choosing
- This is PPLFault







• DEMO





Why Stop at LSASS? ANGRYORCHARD

- Released in July 2022 by Austin Hudson when Microsoft patched PPLdump
- Exploits PPLdump bug to achieve code execution in CSRSS (WinTcb PPL) Exploits bug in NtUserHardErrorControl to perform arbitrary kernel decrement a. Only exploitable within CSRSS
- Decrement KTHREAD.PreviousMode from UserMode (1) to KernelMode (0)
 - KernelMode disables most memory and security access checks on the system а.
 - b. GodMode syscalls treat you like a kernel worker thread
 - Examples: C.
 - hSystemProcess = OpenProcess(4, PROCESS_ALL_ACCESS)
 - WriteProcessMemory(SomeKernelAddress)
 - NtOpenSection(\Device\PhysicalMemory, SECTION_ALL_ACCESS)

SecIdiot/ ANGRYORCHARD

A kernel exploit leveraging NtUserHardErrorControl to elevate a thread to KernelMode and achieve arbitrary kernel R/W & more. $\odot 0$ 公 92 ¥ 24 83 1 Contributor Issues Stars Forks







()



Exploit Chain Demo - GodFault

• DEMO





Mitigations - Windows

- Root of problem is a TOCTOU where signature validation is decoupled from paging
- If only Windows had some way to validate the hashes of pages...

C:\Windows\System32>signtool verify /v /ph "C:\Windows\System32\ntdll.dll" | grep -A10 "Page hash" Page hashes: 0x00000000 63A2FF4AF0FE4AB373879E79C5B6AD71F87921D7785A76DCDA9EA7251D6A5CEB 0x00000400 387F45BEE453C35ED971806041D0A9D71A30DFA5590E05435ABB2D099849C64B 0x00001400 441CDBF430CAFB55AEAB82A0767D81422145245E772FAE5855777F52D5E0D20D

0x00002400 8381A86212C8DDC6048C49523DFDA8416169FFF7BFD141A58FFBAB4A8296BFC9 0x00003400 3A55CF1DE8C04DAAF6DA6D4216C020A9766D5E2EE346A00019B7D2B84BA6FDF4 0x00004400 D0BC1D0FE6C5C4B0206ED7E34107BA16D75C21D884E00410B27FBCC94D68AF3A 0x00005400 BD6AAF7C53EFEB822A16A016C915D907E9A7C4A9A45FB9AF461A9F05D059E365 0x00006400 A620F0E12B712862CCC7BED40134A3978169F523EA52C36424E3BC52BC6EAF57 0x00007400 16C9D10265D816C9EA3790BEA07C3A58DCAE6164ABC1794EF0471D5268805647 0x00008400 D2F08CE6751388ED85E2453994C01392558DDAD7E496D0C5B0D941BBBB36FE4F







Mitigations - AV Vendors

- AntiMalware vendors can't
 - a. Modify the memory manager to require page hashes for all images loaded into PPL
 - b. Re-sign Microsoft binaries with PPL certs to add page hashes
- AntiMalware vendors can still break the PPLFault exploit chain





Mitigation - NoFault

NoRemoteImages

- Exploit mitigation to prevent loading of DLLs from network locations (SMB, WebDAV, etc) a.
- b. Originally introduced with EMET. Later integrated directly into Windows
- Set-ProcessMitigation PowerShell cmdlet
 - a. Persists key in registry
 - b. Useless against attacker who controls registry
- NoFault.sys
 - Enables NoRemoteImages policy early in process lifecycle via process creation callback a.







• DEMO





Disclosure Timeline

• Timeline

- 2022-09-22 Reported PPLFault and GodFault to MSRC as VULN-074311
- 2022-10-21 MSRC case closed without action
- 2023-02-28 I publicly announced this BlackHat talk on Twitter
- 2023-03-01 Windows Defender team reached out to me via Twitter
- Exploits still functional against:
 - Windows 11 22H2 22621.1702 (May 2023)
 - Windows 11 Insider Canary 25346.1001 (April 2023)





Conclusions / Black Hat Sound Bytes

- Defending against administrators is hard
 - Lots of power and attack surface
- Little things add up
 - Non-Elevated => Admin (UAC bypass) is not a security boundary 0
 - Admin => PPL is not a security boundary
 - PPL => Kernel RW is not a security boundary
 - Transitively: Non-Elevated => Kernel RW is not a security boundary
- When MSRC doesn't care, the Defender team still might
- Public tooling get bugs fixed
 - It required "active abuse" to force Microsoft's hand on the DefineDosDevice vulnerability



James Forshaw @tiraniddo · Jul 26, 2022

Replying to @PhilipTsukerman

It is impressed it got fixed, but then it seems "active abuse" is usually the point where something will be fixed if at all. No one cared I documented the weakness until there was turnkey tooling on github :-)







Conclusions: Patching

ppp

Available for: iPhone 8 and later, iPad Pro (all models), iPad Air 3rd generation and later, iPad 5th generation and later, iPad mini 5th generation and later

Impact: An app with **root privileges** may be able to execute arbitrary code with kernel privileges

Description: A use after free issue was addressed with improved memory management.

CVE-2022-42829: an anonymous researcher

ppp

Available for: iPhone 8 and later, iPad Pro (all models), iPad Air 3rd generation and later, iPad 5th generation and later, iPad mini 5th generation and later

Impact: An app with **root privileges** may be able to execute arbitrary code with kernel privileges

Description: The issue was addressed with improved memory handling.

CVE-2022-42830: an anonymous researcher

ppp

Available for: iPhone 8 and later, iPad Pro (all models), iPad Air 3rd generation and later, iPad 5th generation and later, iPad mini 5th generation and later

Impact: An app with **root privileges** may be able to execute arbitrary code with kernel privileges

Description: A race condition was addressed with improved locking.

CVE-2022-42831: an anonymous researcher

CVE-2022-42832: an anonymous researcher









- Gabriel Landau at Elastic Security Labs
- Twitter: @GabrielLandau
- PoC code: https://github.com/gabriellandau/PPLFault



