Protected Process Light will be Protected – MemoryRanger Fills the Gap Again

Slide 1 Hello	
Hi everyone!	
Thanks for inviting me. I am happy to be here.	Protected Process Light is not Protected -
Today I will be talking about undetected kernel attacks on one of Windows mechanisms called Protected	MemoryRanger Fills the Gap Again
Process Light and how to avoid this kind of attacks.	Igor Korkin Independent Researcher
	2021
I think this topic is of crucial importance for all experts dealing with application security.	
Slide 2 "WhoamI" (fast: speed=3)	WHOAMI PhD speaker at the ADFSL BlackHat, HITB IEEE SPW
A few words about me. I've been exploring / ou 'es/ OS security for more than 10 years.	OS Security Researcher:
I've been <u>curious</u> about applying both theoretical and practical expertise to discover new attacks on the OS	Rootkits, Anti-rootkits and EDRs Memory Forensics for user- and kernel- modes
kernel and find the ways to prevent them. You can find all the information in my blog.	Bare-Metal Hypervisors against Attacks on Kernel Memory Fan of cross-disciplinary research — igorkorkin.blogspot.com
	• Love traveling and powerlifting — $\textcircled{0}$ igor korkin
Slide 3 "WhoamI" (fast: speed=2)	RCENDA • Protected Process Light (PPL) — Internals and Attacks
Today, I will be showing you my analysis of a Windows mechanism called Protected Process Light.	
We' <u>ll be seeing</u> its algorithm and its vulnerabilities.	
Slide 4 "Agenda: PeePeeL"	AGENDA
You know {pause}, a great amount of sensitive data {pause} is located in process memory.	
Providing <u>data protection at run time</u> is always a challenge.	<u>.</u>
	Users secrets are stored in process memory AGENDA
Slide 5 "Agenda: PeePeeL disabling"	ROLLIUR
Protected Process Light is a Windows mechanism designed	Protected Process Light (resolved)
	PPL is enabled for the process to protect its memory

Slide 6 "Agenda: PeePeeL disabling"	AGENDA
to protect to guard sensitive data against malicious attacks.	Protected Proces light (enabled) Attackers are trying to steal the secrets, but PFL blocks their access
Slide 7 "Agenda: PeePeeL disabling"	Protected Process Light (enabled)
For example, protected processes cannot be dumped or terminated by non-protected processes.	Thanks to PPL non-protected processes cannot do the following: Acress to the protected process memory inject code into the protected processes Triminate protected processes Attackers are trying to steal the secrets, but PPL blocks their access
Slide 8 "Agenda: PeePeeL disabling"	AGENDA
But attackers can disable PeePeeL for the target processes and	Protected Process Light (masked) Attackers can disable PPL
Slide 9 "Agenda: PeePeeL disabling"	AGENDA
steal users' secrets {pause}.	Protected Process Light Committee of Protected Process Light (manked) (mank
Slide 10 "Agenda: PeePeeL illegal enabling"	PROCESS MEMORY: ATTACKS & PROTECTION
At the same time, attackers are always looking for the ways to protect their own malware from being detected.	Protected Process Light is disabled for malways Attachers want to protect malways processes
Slide 11 "Agenda: PeePeeL illegal enabling"	PROCESS MEMORY: ATTACKS & PROTECTION
PeePeeL seems very promising for attackers, but there are no PeePeeL functions to enable protection for the third-party apps.	Presented Process Light is disable for subserve Comments of the Comments of th
Slide 12 "Agenda: PeePeeL illegal enabling"	PROCESS MEMORY: ATTACKS & PROTECTION
But attackers can illegally enable PeePeeL for malware processes to protect their apps from being detected.	Presented Present Light is disable for natures Attackers can enable FPI to protect their malware processes

Slide 13 "Agenda: PeePeeL disabling and illegal enabling"	AGENDA - Protected Process light (PFL) — Algorithm and Attacks
Both these attacks can be implemented	Example 10 Page 10 Pag
Slide 14 "Agenda: PeePeeL disabling and illegal enabling"	RCENDR Protected Process Light (PPL) — Algorithm and Attacks
by modifying kernel memory, without triggering Windows security features.	
However, {pause} these attacks can be blocked	Name appeared to
	Attackers are abusing PPL by patching kernel data
Slide 15 "Agenda: PeePeeL disabling and illegal enabling"	ACENDA Protected Process Light (PPL) — Algorithm and Attacks
by my MemoryRanger {pause}.	
Slide 16 "Agenda: MemoryRanger protects PeePeeL"	AGENDA Protected Process Light (PPL) — Internals and Attacks
MemoryRanger is {very slow} the solution, {very slow} the tool, {very slow} the utility,	* * * * * * * * * * * * * * * * * * *
which I designed to prevent attacks on kernel memory and	MemoryBanger blocks attacks on PPLs
we {p} will be seeing how my MemoryRanger can successfully block all the attacks on PeePeeL.	(School) (School) (School)
Slide 17 "Agenda: MemoryRanger protects PeePeeL"	Windows 10 Windows 11
Windows experts have developed various process protection mechanisms.	Windows 8
	Microsoft Windows OSes
Slide 18 "Windows Process Protection Mechanisms" Episode 1	
{pause-music}	Episode 1: Windows Features to Protect Process Memory

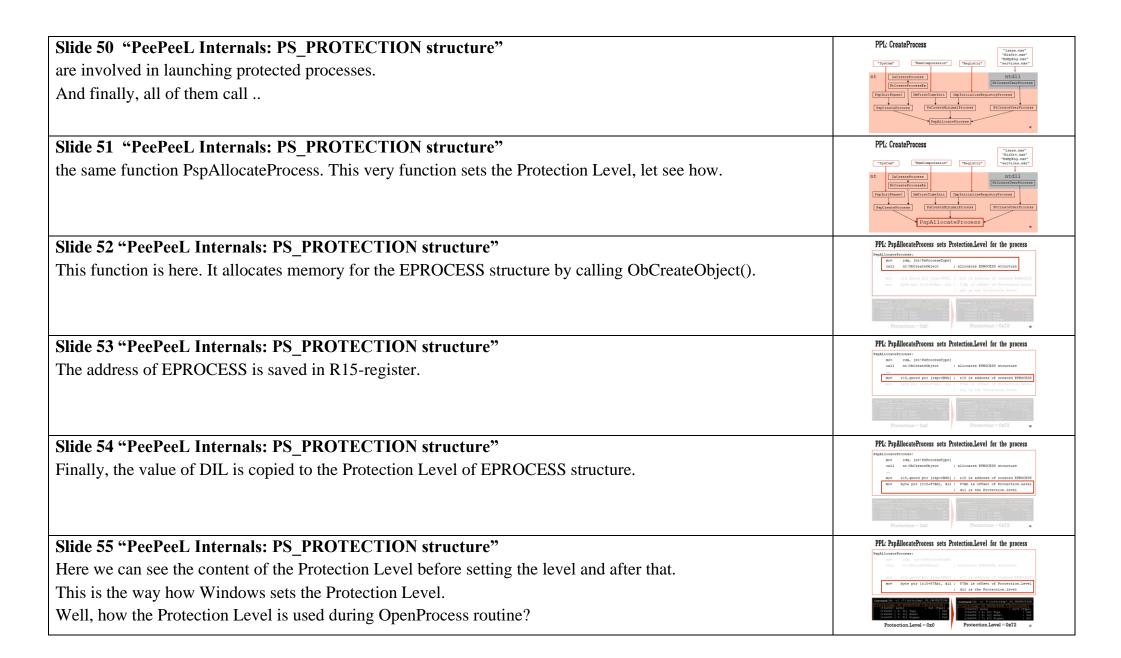
Slide 19 "Security Reference Monitor"	WINDOWS FEATURES TO PROTECT PROCESS MEMORY 1. Security Reference Member (8884)
One of the essential Windows components is	Protected Process Light (PFK) Applications Telestion Windows Resource Protection (WEP, STC) Season Delation and Secure Desktop Windows Memory Management (Virtual semory and Exclave APIs) Mandacty Integrity Centrol (GIC) User Instructor Protection (GIC) Limits are Protection (GIC) Limits Protection (GIC) Limits (GIC
Slide 20 "Security Reference Monitor"	WINDOWS FEATURES TO PROTECT PROCESS MEMORY; SRM 1. Security Reference Monitor (SBM2) 2. Protected Process Light (PPH)
Security Reference Monitor, which performs access check.	AppContainer Indiation Windows Resources Protection (WRF, SFC) Session Deletion and Secure Dealton Windows Memory Management (Virtual namery and Exclave AFIs)
Let's see how it works.	T. Mandanny Integrity Control (IGC) 1. Was Instructive Principles Institute (IGC) 2. Danison Principles Institute (IGC) 2. Danison Provinced Indoo (IGC) 8. Included User Mode (IGC) enabled by Hyper V
Slide 21 "Security Reference Monitor Principles"	Malware App calls OpenProcess() to Access Secret Data
OpenProcess() routine <u>calls</u> {p} Security Reference Monitor to <u>check</u> whether a process has enough privilege	Opposition (A)
to access memory of another process.	
Slide 22 "Security Reference Monitor Principles"	SRM checks access rights using Token and Security Descriptor
• Each process has its own Access Token, which identifies the privileges.	Generation of the control of the con
Slide 23 "Security Reference Monitor Principles"	SRM checks access rights using Token and Security Descriptor
The Security Descriptor stores the process security attributes.	OpenStreet of St. St. State State Street on State St. State St. St. State St. St. State St.
Security Reference Monitor performs access check by comparing	Security Interesse Meanine (EDI) Security Interesse Security Intere
Slide 24 "Security Reference Monitor Principles"	SRM checks access rights using Token and Security Descriptor
Access Token and Security Descriptor.	Oppositions of the control description of the co
Slide 25 "Security Reference Monitor Principles does not restrict processes with the debug privilege"	SRM allows full access for process with SeDebugPrivilege
But for the process running with a debug privilege,	Opportunities (p) SIM Colors according to ground and principal to the colors according to the colors
Security Reference Monitor always allows <u>fúll</u> access without any checks.	Becardy Informer Meaning (SA)
	*

	T
Slide 26 "SRM does not restrict processes running with administrative privileges"	Malware with SeDebugPrivilege can steal sensitive data
Attackers can enable the debug privilege by using a Windows API routine.	Bill allow name for ground make the state of
Therefore, the <u>malware</u> process {p} <u>can get áccess</u> to the sensitive data.	(a) Beauty Salaraca Measing (DM)
That's {p} the way how Security Reference Monitor works.	Problem: Malware app with enabled SelbebugPrivilege can access memory of all processes without any restrictions.
Slide 27 "PeePeeL mechanism: Stop Access Private Data"	How to protect data from apps running with debug privilege? SRM always allows full access for process with enabled
How to fill this /gæp/ gap with data protection and prevent the access attempts to the critical process memory?	Open Process (%) Data is teaked seattle data
Windows experts had faced this problem, and they introduced one more security mechanism.	RT/Process Message Security Reference Message Mes
Slide 28 "Episode 2: PPL Overview"	
{Music}-{Pause}	
	Episode 2 PPL Overview
Slide 29 "PeePeeL mechanism: Stop Access Private Data"	WINDOWS FEATURES TO PROTECT PROCESS MEMORY: PPL
Slide 29 "PeePeeL mechanism: Stop Access Private Data" It is called Protected Processes Light, or PeePeeL.	WINDOWS FEATURES TO PROTECT PROCESS MEMORY: PPL 1. Security References Monitory (2004). 2. Protected Process Light (PPL) 3. Applications Resources Production (VVPE, SFC) 6. Security Delations and Security Resources Performed (PPL) 6. Security Delations and Security Resources Performed (PPL) 7. Monitority Resources (ORC) 8. Use Institute Protection (ORC) 8. Lincitated User Mode (URA) smalled by Hyper-V
	Becaute Reference Meniner (BRM)
It is called Protected Processes Light, or PeePeeL.	1. Becurity Reference Monitor (SRM) 2. Protected Process Lipid (FFE) 3. AppContainer Isolation 4. Windows Resource Protection (WRF, SFC) 5. Section Obstaints and Secure Devictory 6. Windows Memory Management (Windam sensory and Enclave AFEs) 7. Mandatory Integrity Contain (DIGC) 1. User Instale Protection (URG) 8. Educated Protected Model (GRM) 8. Educated Protected Model (GRM)
It is called Protected Processes Light, or PeePeeL. Slide 30 "PeePeeL mechanism: Stop Access Private Data"	Recurry Reference Measure (RRM) Protected Process Light (PFE) 3. Applications Industrial Windows Resource Pointed (In (WE) EFC) 3. Sequel Delation and Secure Deskrip Windows Resource Pointed (In
It is called Protected Processes Light, or PeePeeL. Slide 30 "PeePeeL mechanism: Stop Access Private Data"	Recurry Reference Measure (RRM) Protected Process Light (PFE) 3. Applications Industrial Windows Resource Pointed (In (WE) EFC) 3. Sequel Delation and Secure Deskrip Windows Resource Pointed (In
It is called Protected Processes Light, or PeePeeL. Slide 30 "PeePeeL mechanism: Stop Access Private Data" PeePeeL adds	1. Security Reference Montage (SIDA) 2. Protection Process Light (PTA) 3. App/Container Industria 4. Wildows Resource Protection (WIR-SPC) 6. Session O Industria and Secure Dealtop 6. Windows Memory Management (Virtual sensory and Exclave AFts) 7. Kanadasary Sinegary Content (OEC) 6. Dealtows Theorem (Mod (SIDA) 7. Exclaved Travello (Mod (SIDA) 8. Inclained User Mode (UEA) sealthed by Hyper-V 7. PFL restricts no-FFL appr raming with debug privilege PFL restricts on-FFL appr raming with debug privilege

Slide 32 "PeePeeL mechanism: Stop Access Private Data"	
OS <u>marks</u> some apps as protected or PeePeeL processes	PPS, centricus non-PPS, appa running with debug privilege **********************************
Slide 33 "PeePeeL mechanism: Stop Access Private Data"	PFs restricts non-PFs and remains with debut privilege
while other apps are marked as not protected.	Special State Control Prince Control
Slide 34 "PeePeeL mechanism: Stop Access Private Data"	PPL restricts non-PPL apps running with debug privilege
Now, any non-protected process <u>es</u> cann <u>ó</u> t get access to the data of protected one.	An access is blocked Combresses (A) Professes Ranges Security Reference Monitor (SRM) Protected Process Light (PPL) Non-protected process cannot access the protected one
Slide 35 "PeePeeL mechanism: Stop Access Private Data"	PPL restricts non-PPL apps running with debug privilege
The illegal access is blocked. It seems that the data protection can be performed by PeePeeL, but let's analyze its algorithm carefully.	An access is blocked Constructed from the first that the first th
Slide 36 "Episode 3 PPL Internals: PPL Data and PPL Code"	
{Pause-music}	Episode 3 PPL Internals: PPL Data and PPL Code
Slide 37 "PeePeeL Internals: New Fields in EPROCESS"	PPL: a new Protection field in EPROCESS
As you know <u>e-each</u> Windows process is represented by a kernel structure called EPROCESS.	Use mode Erem to de EPROCESS

PPL: a new Protection field in EPROCESS Slide 38 "PeePeeL Internals: New Fields in EPROCESS" It includes information about the process, like process ID, full name, process privileges and other related structures. To /sə pɔ:t/ support PeePeeL the EPROCESS structure has been updated. Slide 39 "PeePeeL Internals: New Fields in EPROCESS" PPL: a new Protection field in EPROCESS A new field named Protection has been added. Slide 40 "PeePeeL Internals: New Fields in EPROCESS" The protection level is stored in a PS_PROTECTION structure, which is here. All the information is stored in a single byte in the two parts. UCHAR Signer : 4; Slide 41 "PeePeeL Internals: PS PROTECTION structure" Protection Level = Signer A signer of protected process and a type. PPL: a new Protection field in EPROCESS Slide 42 "PeePeeL Internals: PS PROTECTION structure" Signer can have 9 different values, while type just three. 1 - Authenticos The protection level is defined by a combination of these two fields. 8 - Windows 6 - Win Tob Slide 43 "PeePeeL Internals: PS PROTECTION structure" EXAMPLES OF PROTECTION LEVEL Various system processes have different Protection Level values. 0x31 For example, LSASS process is running with the Protection value 41.

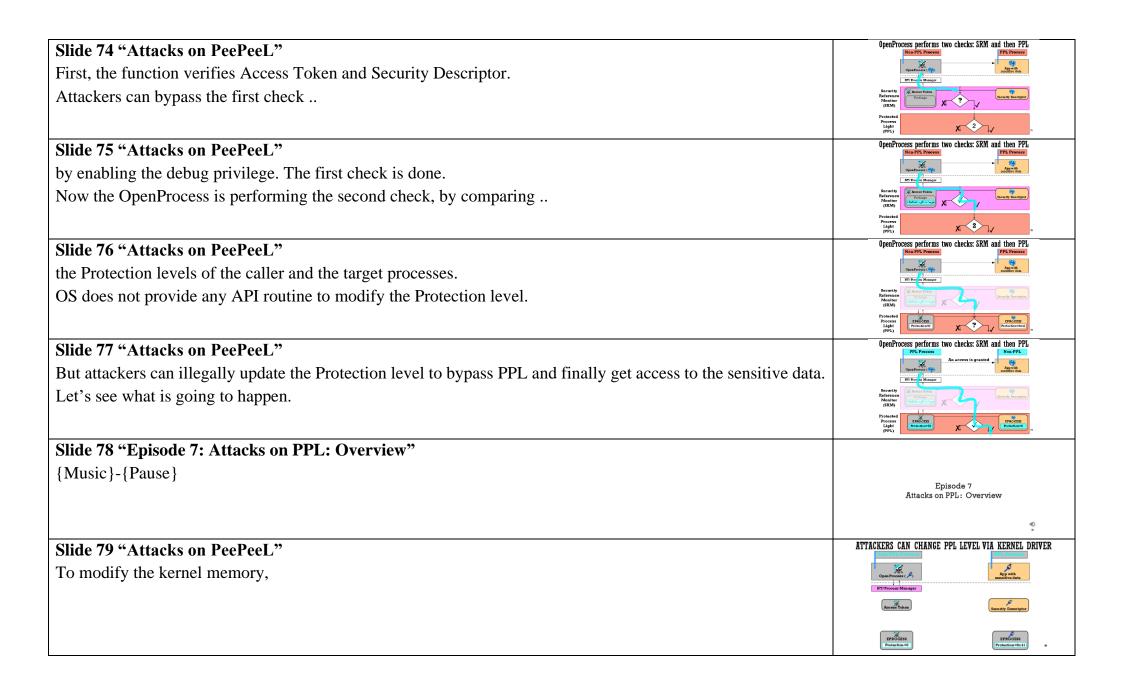
SLIDE 44 "PeePeeL Internals: PS_PROTECTION structure"	EXAMPLES OF PROTECTION LEVEL
As we know, each protection level is a combination of the Signer and Type.	Process Protection Level Signer Type
Slide 45 "PeePeeL Internals: PS_PROTECTION structure"	EXAMPLES OF PROTECTION LEVEL Process name Protection Level Signer Type
Briefly speaking, processes with higher protection levels are more protected.	Nis6rv 0x31
They cannot be accessed by processes with lower protection levels.	SgrmBroker 0x82 6 (WinTeb) 2 (Full)
How does Windows initiate the Protection level for new launched processes?	Nasirv — Microsoft Network Bealtime Inspection Service. IASS — Local Security Authority Subsystem Service. Symmitotics— System Guards Nutrino Memorit Robot. • • • • • • • • • • • • • • • • • • •
Slide 46 "Episode 4: PPL Code: Creating Protected Processes"	
{Pause-music}	Episode 3 continue PPL Code
Slide 47 "PeePeeL Internals: PS_PROTECTION structure"	PPL: CreateProcess
The following protected processes are launched during Windows startup.	"Paystes" "WestCompression" "Registry" "Helpfide, eet" "service.eee"
Slide 48 "PeePeeL Internals: PS_PROTECTION structure"	PPL: CreateProcess
Which OS functions are involved to launch them?	**Pyrnor*
Slide 49 "PeePeeL Internals: PS_PROTECTION structure"	PPL: CreateProcess
We can see that various API functions	Typecar "Tendingression" "Replaty" "Retition.css" It Directors Directors



Slide 56 "Episode 5: PPL Code: Opening Protected Processes"	
{Pause}-{sound}	Episode 5 PPL Code: Opening Process
Slide 57 "PeePeeL Internals: New values" One app is trying to open a protected process.	PPL: OpenProcess OpenProcess **********************************
Slide 58 "PeePeeL Internals: New values" Which OS functions are involved to open a process?	PPL: OpenProcess OpenProcess **Janant - see* **Tanant
Slide 59 "PeePeeL Internals: New values" The control goes to the NtOpenProcess, from ntdll.	PPL: OpenProcess Tomas App handle = OpenProcess(pid) "" "ARRA-REA" "RELETY-REA" "RELETY-REA" "RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REA" RELETY-REAT RELET
Slide 60 "PeePeeL Internals: New values" We have a list of function calls. And finally, the control goes to the function RtlTestProtectedAccess.	PPL: OpenProcess Data App
Slide 61 "PeePeeL Internals: New values" This very function checks the process protection permissions, let's see how.	PPL: OpenProcess → RtiTestProtectedAccess Tomackey handle = OpenProcess(pid)

Slide 62 "PeePeeL Internals: New values"	PPL: RtiTestProtectedAccess() bool ReflectProcectedAccess((R) PROTECTION CallesProt, PS PROTECTION TargesProt)
This function is here.	(if (TaugedPool.Type == 0) return true;
	if (GallerProt.Type < TargetProt.Type) secure false,
	auto CallerDMaak: #BilProtectedAccess(CallerProt.Signer). DominateMaak; auto TargotMaak: = (1 < TargotProt.Signer); roturn (CallerDMaak: & TargotMaak);
Slide 63 "PeePeeL Internals: New values"	PPL: RtiTestProtectedAccess() boot RtitlestProtectedAccess(Ps. PROTECTION CalletProt, Ps. PROTECTION TargetProt)
The function has two <u>input</u> parameters – Protection values for the caller and the target processes.	if (TargedFrot Type == 0) return true;
	if (CalletProt.Type < TargetProt.Type) return false;
	<pre>auto Caller/Dhatak = Rifforcierchd/cosus[Caller/Frot Signer] DominateMask; auto TargetMask = [1 << TargetProt Signer); return (Caller/Dhask & TargetMask);</pre>
Slide 64 "PeePeeL Internals: New values"	PPL: RtiTestProtectedAccess()
First, it checks the target protection level. If it is zero the access is granted.	bool RithestPostectedAccess(PS_PROTECTION GallerProt, TargesProt) {
For example, Mimikatz. You know it is a software tool that can retrieve the credentials.	if (CalleerPot.Type < TargetFrot.Type) rotum false;
Mimikatz uses this feature to <u>á</u> ccess LSASS memory.	auto CallerDMaak = RilProtectedAccess[CallerProt Signer] DominateMask; autoTargetMask = (1 << TargetProt Signer); return (CallerDMask & TargetMask);
If the target process is protected, the control goes to the second check.	,
Slide 65 "PeePeeL Internals: New values"	PPL: RtiTestProtectedAccess() bool RtiTestProtectedAccess(PS_PROTECTION CallerProt, TargetProt)
Now the function is comparing the Protection types. If the Target type is bigger, it blocks the access.	if (TargedFrot.Type == 0) return true;
If the Caller type is big enough the control goes to the final check.	if (CallerProt.Type - TargetProt.Type) return false; auto Caller DMask = RBProtectedAccess(CallerProt Signer) DominateMask.
	autoTargetMask = (1 << TargetPot Signer); return (CaberDMask & TargetMask); }
Slide 66 "PeePeeL Internals: New values"	PPI: RiffestProtected&ccess() bool RiffustProtected&ccess(%_PROTECTION_CalberProt, TargetProt)
Now, the function is checking whether a caller dominates the target using an especial array called	i (TargelFrot.Type == 0) return true,
RtlProtectedAccess.	if (CallerProt Type < TargetProt Type) return false;
	auto CallertDMaak = RitPricenced.Roceas(CallerProt.Signer) DominateMaak auto TargetMaak = (1 << TargetProt.Signer); return (CallertDMaak & TargetDtAak);
Slide 67 "PeePeeL Internals: New values"	RtiProtectedAccess Array
This data array is one more structure created for PeePeeL mechanism.	
The part of this array is here. For each Signer type, the array includes the field called	3 Antimatware 108 4 Lea 110
"DominateMask". / AEAEA /. This field indicates the privilege for each Signer type.	5 Windows 19e 6 WinTCE 17e 7 WinSystem Ife 8 SuperApp 0

Slide 68 "PeePeeL Internals: New values"	RtiProtectedAccess Array Index Signer DominateMask Bit Explanation
Let me explain.	
Slide 69 "PeePeeL Internals: New values"	RtlProtectedAccess Array
For example, /óthentic code signer/ Authenticode Signer has a DominateMask equals 2.	Bigner Demonstrate of Departments Bigner Demonstrate of Departments Bigner Demonstrate of Demonstr
It means that bit 1 {pause} is enabled, which corresponds to the Authenticode.	1 10080-008
Therefore, processes with /othentic code signer/ Authenticode signer can access only /othentic code processes/	
Authenticode processes.	8 SignerApp 0 a/s
Slide 70 "PeePeeL Internals: New values"	RtiProtectedAccess Array
We can see that LSA has DominateMask equals 110.	
Now bits 3 and 8 are enabled. Therefore, LSA processes can access LSA and SignerApp processes.	4 Les 110 07654 2210 -41-Les 100 1000 -41-Les -4-ReperApp -6 WertClb 17e 10111110 -4-ReperApp -7 Wildpanes 16 ReperApp -7 Wildpanes 16 ReperApp -7 Wildpanes 16 ReperApp -7 Re
Slide 71 "PeePeeL Internals: New values"	RtlProtectedAccess Array
WinSystem is a very interesting case. Processes with this signer level can access all processes because all bits are set. Malware can use this information to access any processes, without regarding their protection levels. Let's see how it can happen.	Authorization
Slide 72 "Episode 6: SRM and PPL are playing together and losing"	
{Pause}-{Music}	Episode 6 SRM and PPL – malware avoids both
Slide 73 "Attacks on PeePeeL"	OpenProcess performs two checks: SRM and then PPL
To access the protected process the malware has to bypass two {pause} security access checks.	Recurity Reference Menture (BEN) Protected Process



	AMBRANDO GEN GUENAD DOL LUNDI ME VIDADI DOLUDO
Slide 80 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
an attacker has to use a kernel driver.	Cymn Processes (A)
	Account Token Remed Driver Security Descriptor
	EPROCESS Protection to
Slide 81 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
The attacker can reset or clear the Protection field for the target process, which disables PPL to this process.	Committee of the Control of the Cont
	EPROCESS Protection #0 Protection #0
Slide 82 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
After that, the attacker can access the process memory easily, because PeePeeL has been disabled.	Opus Process (P) NT-Process Manager Mill Process Manager Remail Driver Respirator distance of distalled by the country Descriptor Respirator to distance of distalled by the country Descriptor of
	EPRÓCESS Protection 10
Slide 83 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
At the same time, the attacker can illegally enable PeePeeL for his own malicious process by setting the Protection level.	Open Fromm () Appointment of the Market of
	Account Token Kernel Driver Security Descriptor
	Set value to enhalo FPL for following EPRÖCESS Protection 100 Protection 100 110 110 110 110 110 110 110
Slide 84 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
Now, PeePeeL is protecting the <u>malware</u> process.	Open Primes (A) Open Transa (A) NY Prevents Manager Manager
	Security Descriptor Set value to enable FPI for Malmans EPROCESS FYDEROMS FYDEROM

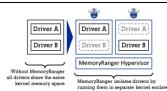
Slide 85 "Attacks on PeePeeL"	ATTACKERS CAN CHANGE PPL LEVEL VIA KERNEL DRIVER
Both these kernel data modifications never trigger any security features, like PatchGuard.	Open Processe () age with sandiffice data
Probably, Windows developers didn't take into account this attack on PeePeeL.	NT/Froom Manager
But they should've. (But they should have taken them.)	Reveal Driver Entryline to enable PPL for Malway PPL for Malway
Let's see some examples of the attacks on PeePeeL.	PROCES PROCES
Slide 86 "Attacks on PeePeeL"	Attacks on PPL Drivers Can Modify Protection Byte
To perform this kind of attacks, intruders have to use kernel drivers.	Fances Control
Slide 87 "Attacks on PeePeeL"	Attacks on PPA Drivers Can Modify Protection Byte
They can exploit vulnerable drivers;	Tensor Parket Control of Control
Slide 88 "Attacks on PeePeeL"	Attacks on FPL Drivers Can Modify Protections Byte (€ Name) ▼ Name
They can use specially crafted drivers;	The state of the s
Slide 89 "Attacks on PeePeeL"	Attacks on PPL Drivers Cas Modify Protection Byte
They can apply even a general-purpose hacker's toolkit, such as Blackbone and even Mimikatz.	TORONO CONTRACTOR CONT
Let me show you an example with Mimikatz.	organ in the control of the control
Slide 90 "Episode 7: Mimikatz disables PPL for LSASS to dump NTLM hashes"	
{Music}-{Pause}	Episode 7
	Mimikatz disables PPL for LSASS to dump NTLM hashes
Slide 91 "Mimikatz"	Minister can gather credentials from Windows *** *** *** *** *** *** ***
You know, Mimikatz is an open-source toolkit designed to make some experiments with Windows Security.	
Slide 92 "Mimikatz"	Mimitatz can gather credentials from Windows **Recycl Windows District ***********************************
An attacker can use Mimikatz to extract users' credentials from LSASS twice.	225.
Slide 93 "Mimikatz"	Minibatz can gather credentials from Windows **Compared to the compared to th

First, an attacker adds the debug privilege for Mimikatz App and after that, he tries to extract the password	
hashes. We will see if he gains the access.	
Slide 94 "Mimikatz"	Minimate can gather credentials from Windows **Recognitive Contractive To Contra
If he fails, he will load a driver,	The state of the s
Slide 95 "Mimikatz"	Minikatz can gather credentials from Windows
which disables PeePeeL.	To the second se
Slide 96 "Mimikatz"	Himitate on gether credentials from Windows House I those thouse / The Company of the Albane Recognition of the Albane Recogni
After that, he repeats the commands to extract hashes,	Control Contro
Slide 97 "Mimikatz"	Ministra can enther credentials from Windows Manage
and finally, he uses a special tool called hashcat to recover user's passwords from the leaked hash.	The state of the s
Let's see how it can happen.	Commence of the second of the
Slide 98 "Demo 1: Attack on PeePeeL"	
Let's check the Windows version. We've got the final one.	
An attacker is launching Mimikatz app. This app is loaded.	
He is copying the commands to add privilege and extract hashes.	
And look – he has failed. PeePeeL prevents illegal access to the protected process.	Mimikatz disables PPL to dump NTLM hashes
	minimate disables 111 to dump Willin adoles
$Okay\{p\},$	
but, the attacker is trying again {p}	The online version is here—
using more commands in order to disable PeePeeL and extract hashes.	https://www.yositube.com/embed/88g4PgtuDZc?vq=hd1440 =
Look – the hash has been gained.	
The attacker is copying the gained hash to the hashcat's config file.	
Now, the hashcat is ready to start.	
The attacker is launching cmd to load hashcat.	

The hashcat is starting to crack the password hash.	
Let's wait until the hash is cracked. It usually takes a while, and in this case, it is about {p} seven minutes.	
The password "honeypot" has been retrieved.	
Is the password correct? The attacker is checking it.	
Oh no! The gained password is correct.	
The OS is in danger.	
Slide 99 "Nutshell: MemoryRanger Prevent PeePeeL disabling"	How to prevent PPL disabling? Just restrict access to the Protection field
Is it possible to prevent this kind of attacks? Yes, it is /itis/.	The state of the s
	U. T.
Slide 100 "Episode 8: MemoryRanger blocks Mimikatz"	
{Music}-{Pause}	
{\text{viusic} - {\text{rause}}	Episode 8 MemoryRanger blocks Mimikatz
	Methoty Manger Micros Militade
Slide 101 "Nutshell: MemoryRanger Prevent PeePeeL disabling"	KERNEL DRIVERS SHARE THE SAME MEMORY SPACE
Currently, all kernel drivers share the same memory space.	The Current situation
If we are able to prevent illegal access to the Protection field of ERPOCESS structure, we can protect PeePeeL.	Larger Margarian Control Contr
The ke-e-y feature of PPL is that only Windows kernel needs to access the Protection field, {pause}	Zandia PFS
all other access attempts can and must be {1 sec} blocked.	
Slide 102 "Nutshell: MemoryRanger Prevent PeePeeL disabling"	Here to restrict access to the Postercials diale?
How to prevent this access? I suppose that my MemoryRanger {pause for the sound on the next slide}	
Slide 103 "MemoryRanger"	MemoryRanger
is the solution to control access in kernel memory.	

Slide 104 "MemoryRanger"

My MemoryRanger, it is a <u>hypervisor-básed</u> {small-pause} <u>software</u> {pause} designed to block kernel-mode attacks. It can /træp/ trap the loading of new drivers and move them to the {slow} isolated kernel/'enklervs/ énclaves in run-time with different memory access restrictions.



Slide 105 "MemoryRanger blocks PPL disabling"

To prevent any illegal access to the sensitive data,

The Current schadine

Slide 106 "MemoryRanger blocks PPL disabling"

My MemoryRanger must be loaded first.



Slide 107 "MemoryRanger blocks PPL disabling"

It allocates the default énclave for the OS and previously loaded drivers.



Slide 108 "MemoryRanger blocks PPL disabling"

MemoryRanger can trap the loading of Mimikatz driver and move it to the separate énclave.

This enclave includes only Mimikatz driver and the limited number of OS drivers.

The Protection field of LSASS will be excluded from this enclave.

Slide 109 "MemoryRanger blocks PPL disabling"

This <u>sche-e-me</u> helps to prevent disabling PeePeeL mechanism by trapping and blocking illegal access to the Protection field.

Let's see how it works.



Slide 110 "Demo2: MemoryRanger prevents Hijacking Handle Table"

Let's check the Windows version. We've got the final one.

The MemoryRanger hypervisor is loaded first.

Now, an attacker is **launching** Mimikatz App. This app is loaded.

The attacker is copying the commands to add privilege and extract hashes.

And look – He has failed. PeePeeL prevents illegal access to the protected process.

MemoryRanger Blocks Mimikatz and prevents disabling of PPL



The online version is here –

Okay{p}, but the attacker is trying again {p} using more commands to disable PeePeeL and extract hashes. Look – He gets nothing. He has failed again. MemoryRanger has prevented disabling of PeePeeL. Thanks to MemoryRanger the OS and user's data are protected. Slide 111 "Episode 9: Malware escalates its own PPL to attack protected processes" MemoryRanger Prevents Disabling of PPL for LSASS MemoryRanger prevents modifying of Protection Level for LSASS. Well, but what about escalating PPL level for malware? Slide 112 "Episode 10: Malware escalates its own PPL to attack protected processes" {music}-{pause} Slide 113 "MemoryRanger" As we know the OS runs the process as protected only if its image file has a special digital certificate. Here is the certificate for LSASS process and corresponding Protection Level. Slide 114 "MemoryRanger" Malware doesn't have this certificate, and it doesn't care. Malware can escalate the Protection level in run time to access the protected processes, let's see how. Slide 115 "MemoryRanger" Here we have a malware app and three system processes

Slide 116 "MemoryRanger"

The attacker wants to dump these processes, but PPL prevents these attempts.

Malware can escalate its PPL to dump Protected Apps | Market | Ma

Slide 117 "MemoryRanger"

Here we have corresponding EPROCESS structures.

Malware can escalate its PPL to dump Protected Apps | Street | St

Slide 118 "MemoryRanger"

Malware can load its driver to modify the Protection level for its own malware process and dump protected memory. Let's see if it causes a PatchGuard reaction.



Slide 119 "Demo3: Attacker's App Escalates Privileges"

Let's check the Windows version. We've got a newest Windows 11.

An attacker is **launching** its app, which loads a driver.

Process Hacker is launched in order to see the Protection Levels.

Windows built-in processes are protected by PPL.

But attacker's app is not protected, which is expected.

Malware escalates its PPL to dump Protected Processes

The online version is here https://www.voutube.com/embed/777777777777772vg=hd14

The attacker is dumping **Realtime Inspection Service**, and he fails.

OS prevents illegal access.

The attacker is setting debug privilege and setting the Protection Level, which is 31 (thirty one).

Let's check the Protection Levels again and relaunch the Process Hacker.

Now the attacker App is protected. The attacker is dumping Realtime Inspection Service again, and he succeeds, the dump file is here. The attacker is focusing on LSASS. The attacker is dumping this process, and he fails. OS prevents illegal access. The attacker is escalating the Protection Level using 62 (sixty-two) value. Let's check the Protection Levels and relaunch the Process Hacker once more. The attacker's app is protected with increased /encreSt/ Protection Level. The attacker is dumping LSASS again, and he succeeds, the dump file is here. The attacker is focusing on SgrmBroker. The attacker is dumping this process, and he succeeds again, the dump file is here. Let's wait for Patch Guard reaction, which is designed to prevent any illegal memory modifications. We've been waiting for 10 hours it is quite a long time and ... Look! Nothing has happened. The OS has not been crushed. The OS is in danger. Malware can escalate its PPL to dump Protected Apps Slide 120 "MemoryRanger" We have seen that attacker has successfully dumped protected processes without triggering Windows security features. Let me show you how MemoryRanger can prevent this attack. Slide 121 "Episode 10: MemoryRanger blocks modifying PPL" {music}-{pause} MemoryRanger blocks modifying PPL

Slide 122 "MemoryRanger"	
MemoryRanger is launched first	
Slide 123 "MemoryRanger"	
and it moves all EPROCESS structures into the separate enclave.	
MemoryRanger can trap the launching of the malware app	
Slide 124 "MemoryRanger"	
and moves its EPROCESS into this enclave.	
MemoryRanger intercepts the launching of the malware driver	
Slide 125 "MemoryRanger"	
and moves it to the separate enclave.	
Let me show how this scheme helps to block PPL escalation.	
Slide 126 "Demo4: MemoryRanger Prevents Attacker's App from Modifying the PPL Level"	
Let's check the Windows version. We've got a newest Windows 11.	
First, we launch a DebugView to see the kernel debug prints.	
MemoryRanger hyper <u>v</u> isor is loaded.	
An attacker is <u>launching</u> its app, which loads a driver.	MemoryRanger prevents escalation of PPL
MemoryRanger traps creation of attacker's app and restricts memory access to its Protection Level.	
Memory Ranger also traps the loading of attacker's driver and isolates this driver.	The online version is here— https://www.voutube.com/embed/222222222222222222222222222222222222
	mpe) www.pontuo-com entition assessments. (4-mit 140 - a
Process Hacker is launched in order to see the Protection Levels.	
Windows built-in processes are protected by PPL.	
Attacker's app is non-protected.	
The attacker is dumping Realtime Inspection Service , and he fails.	
OS prevents illegal access.	

The attacker is setting debug privilege and is setting the Protection Level, which is 31 (thirty one).

Let's check the Protection Level again and relaunch the Process Hacker.

The attacker's app is still non-protected.

Let's see the debug output.

MemoryRanger traps an access attempt to the sensitive memory and {pause} redirects it to the fake page.

The attacker is dumping Realtime Inspection Service once more.

He dumps nothing. He fails again.

OS prevents illegal access.

The attacker is dumping LSASS and he fails.

OS prevents illegal access again.

The attacker is escalating the Protection Level using 62 (sixty-two) value.

Let's check the Protection Levels and relaunch the Process Hacker once more.

The attacker's app is still non-protected.

MemoryRanger traps this illegal access and blocks it as well.

The attacker is trying to dump LSASS again. He dumps nothing. He fails again.

The attacker is dumping SgrmBroker and he fails once more.

The OS is protected!

Thanks to the MemoryRanger the Protected Processes are actually Protected now.

Slide 127 "MemoryRanger"	MemoryRanger blocks modifying PPL
We've just seen that MemoryRanger can successfully block both attacks on PPL: disabling PPL and escalating	MELTONICO MATERIA MATE
PPL level.	Particular Par
Now let's move on to the MemoryRanger architecture.	
Slide 128 "Episode 11: Architecture and Customization of MemoryRanger"	
{Pause}-{Music}	Episode 9
	Architecture and Customization of MemoryRanger
Slide 129 "MemoryRanger overview"	SUPERIOR STARK THICK
MemoryRanger processes the following four events:	
Slide 130 "MemoryRanger overview"	MONEY RACES ACCITOURS OF PARTIES OF PART
loading new drivers	
Slide 131 "MemoryRanger overview"	MEMORY ABADES ARCHITATURE Of (Assessed Assessed) On Assessed Ass
launching new processes	
Slide 132 "MemoryRanger overview"	NEXORE ANGER AGAINTATIVE *** \(\text{ \te\
calling kernel API functions and	
Slide 133 "MemoryRanger overview"	MONEY SACCE ACCEPTANCE (4) [Territory Investment Inv
processing memory access violations, which occur due to the access to the memory with restricted access.	
Slide 134 "MemoryRanger overview"	MEMORY SANCE ANCHITOTURE 10 Service Servi
After loading, MemoryRanger allocates a default enclave for the OS and previously loaded drivers.	The state of the s
Slide 135 "MemoryRanger overview"	
• MemoryRanger traps loading drivers and creates a separate memory énclave for each of them. These	
enclaves have different memory access configurations. The information about each memory	
configuration is saved into ISOLATE_MEM_ENCLAVE structure.	

Slide 136 "MemoryRanger overview"	MEMORY RANGES, RECURSORY (sealing and sealing and seal
 MemoryRanger can also trap launching new users' apps. It helps to locate sensitive kernel data and 	CONTRACTOR
block access to them by modifying PROTECTED_MEMORY structures.	(MANUAL STATE OF STAT
Slide 137 "MemoryRanger overview"	MEMORY RANGER ARCHITOTURE
MemoryRanger can trap any kernel API routine. DdiMon component provides these hidden hooks using	(a) Particularly (b) Particularly (b) Annual Marian
EeePeeTee. It helps to locate sensitive data, allocated dynamically.	ENTER AND
Any access to the restricted memory areas causes EPT violations.	
Slide 138 "MemoryRanger overview"	
MemoryMonRWX is designed to process all these violations. To process execute violations MemoryMonRWX	MEMORY RANCE RECEIVATION (ii) Annual Annual
changes enclaves, so another driver continues execution inside its own enclave.	The same of the sa
Read or write violations mean that a driver is <u>á</u> ccessing restricted memory data. <u>This case</u> is redirected to the	(MATERIAL MATERIAL MA
Memory Access Policy (MAP), which	
Slide 139 "MemoryRanger overview"	MEMORY RANGE RECEIVATIVE or Language Contracting Contracting Contracting of Security Systems or Language Contracting Contracting Contracting of Security Systems
decides whether to block or allow this access.	The state of the s
MemoryRanger is a proof-of-concept which can monitor and block access to the kernel data and code.	CONTRACT OF STATE OF
Slide 140 "MemoryRanger: Previous Research"	MEMORYRANGER PREVIOUS RESEARCH
Here are some of my research projects. That demonstrate various types of MemoryRanger customization to	Marie Andrews Company of the Company
prevent different types of kernel attacks.	dilli Donto ur Diginar Leiner Elevaria (Leiner Leiner Leiner Leiner Elevaria (Leiner Elevaria (Le
	n 2000 (100 de de tres no 2000) (100 de de tres no 2000) (100 de t
Slide 141 "MemoryRanger: Customized"	MennsyRanger can Protect PPL - 2012 Direct - 1010 Hypersons
To protect PPL MemoryRanger has been customized in the following way:	
Slide 142 "MemoryRanger: Customized"	MemoryRanger can Protect PPI. - ten Show:
MemoryRanger's driver {p} locates the address of the Protection field of EPROCESS.	- MD Rywroac
Slide 143 "MemoryRanger: Customized"	NemeryRanger can Protect PPL
and traps loading of kernel drivers.	months on antimose of minimum tons of all ADAG (MINICAS) MINIMUM TONS (MINIMUM TONS) MINIMUM TONS (MINIMUM TONS)
Slide 144 "MemoryRanger: Customized"	MemoryRanger can Pottest PPU - 1887 - Done - I man to Antonion Chromismo fine of LEAAA (1890-0288) - I may to Antonion Chromismo fine of LEAAA (1890-0288)
MemoryRanger's hypervisor provides various memory access restrictions for the default enclave and	Discontinuous United States
	•

Slide 145 "MemoryRanger: Customized"	MemoryRanger can Protect PPL - 1000 Entree: - Inners a substant of Protection that of Mandal SPROCESS
for the enclaves allocated for the kernel drivers.	* reprivate private *********************************
Slide 146 "MemoryRanger: Customized"	Menosykasper can Protect PPL
MemoryRanger's hyper <u>v</u> isor blocks illegal access to the Protection field.	- teams a solation or Francisco size (ASAM SPECIA) - SEE SEE SEE SEE SEE SEE SEE SEE SEE SE
This is the way how MemoryRanger protects PeePeeL.	Sith Stypersons: The State of
Slide 147 "Conclusion"	
Let me /ˈriːkap/ ré-cáp very briefly on what we have discussed so far.	CONCLISION
First of all, Windows Security Model is based on User's Access Tokens and Object's Security Descriptors.	VANA-GARDEN 1. Washards Southery Woods date on menters upon meaning with delivery periodicys. 2. Promoted From the Ight PRING primer a memory of OR and AFF promotes. 3. Annahom and disability of this Southering ways. 4. Annahom and disability in this Southering ways.
This model does not restrict processes running with the debug privilege.	- person the national by Equility resident (EE) - man and multiple and Cold and Est pressures by Equiphing Smithag (MR. 4. Memory Regard blacks sensite an Nersol data sainfalling sensite as TEG.
To fill this gap a new mechanism called Protected Process Light has been released. But it can be disabled just	
by modifying a byte in kernel memory.	
Slide 148 "Thank you"	Think your
I have presented my MemoryRanger, which can prevent this kind of attacks by restricting memory access to the	lyer Kerken spec korken@geneil.com All the detach my hore spechorlen.hoppyse.com
Protection fields.	T See See 1
Thank you!	