



Defeating x64: Modern Trends of Kernel-Mode Rootkits

Aleksandr Matrosov

Eugene Rodionov



Who we are?

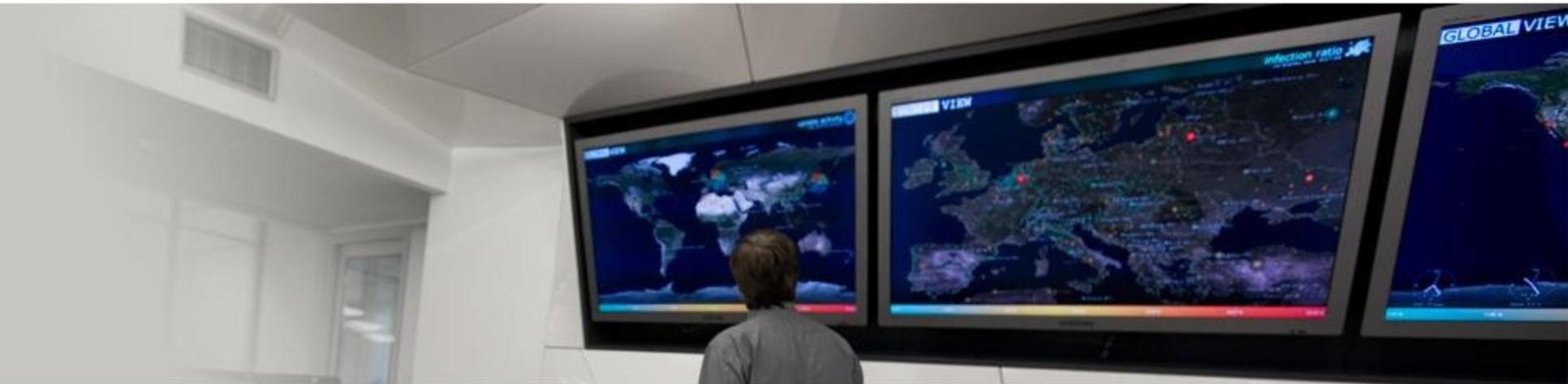
- **Malware researchers at ESET**
 - **rootkits analysis**
 - **development of cleaning tools**
 - **tracking new rootkit techniques**
 - **investigation of cybercrime groups**



<http://www.joineset.com/>

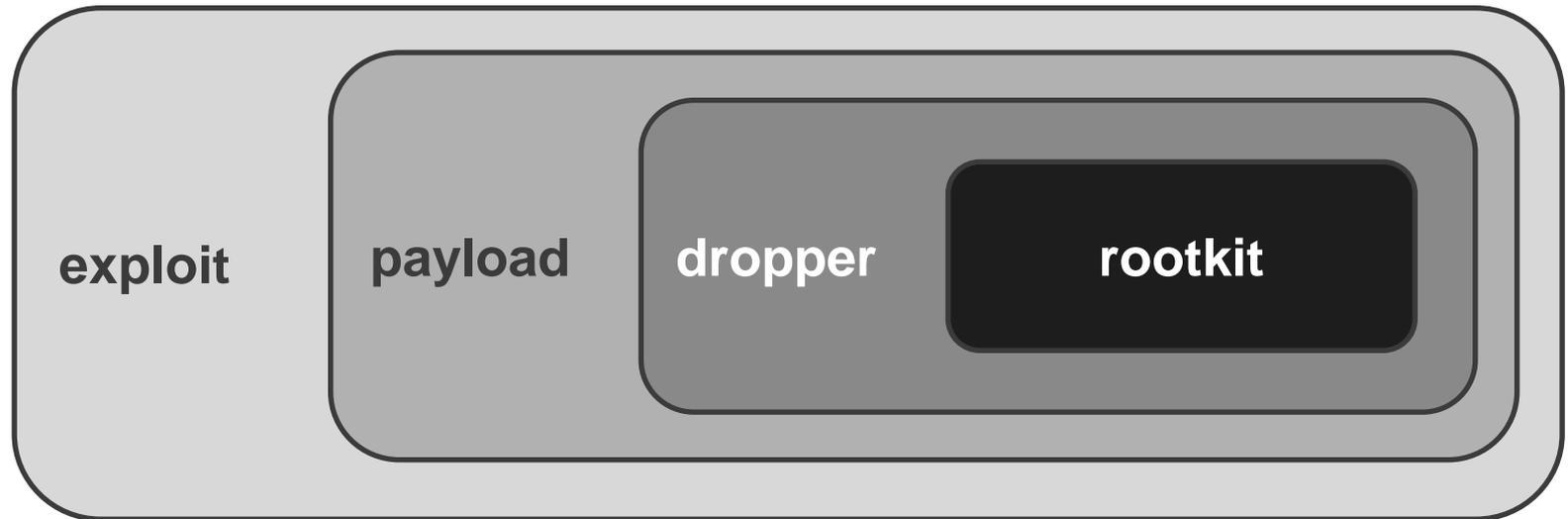
Agenda

- ✓ **Evolution of payloads and rootkits**
- ✓ **Bypassing code integrity checks**
- ✓ **Attacking Windows Bootloader**
- ✓ **Modern Bootkit details:**
 - **Win64/Olmarik**
 - **Win64/Rovnix**
- ✓ **How to debug bootkit with Bochs emulator**
- ✓ **HiddenFsReader as a forensic tool**

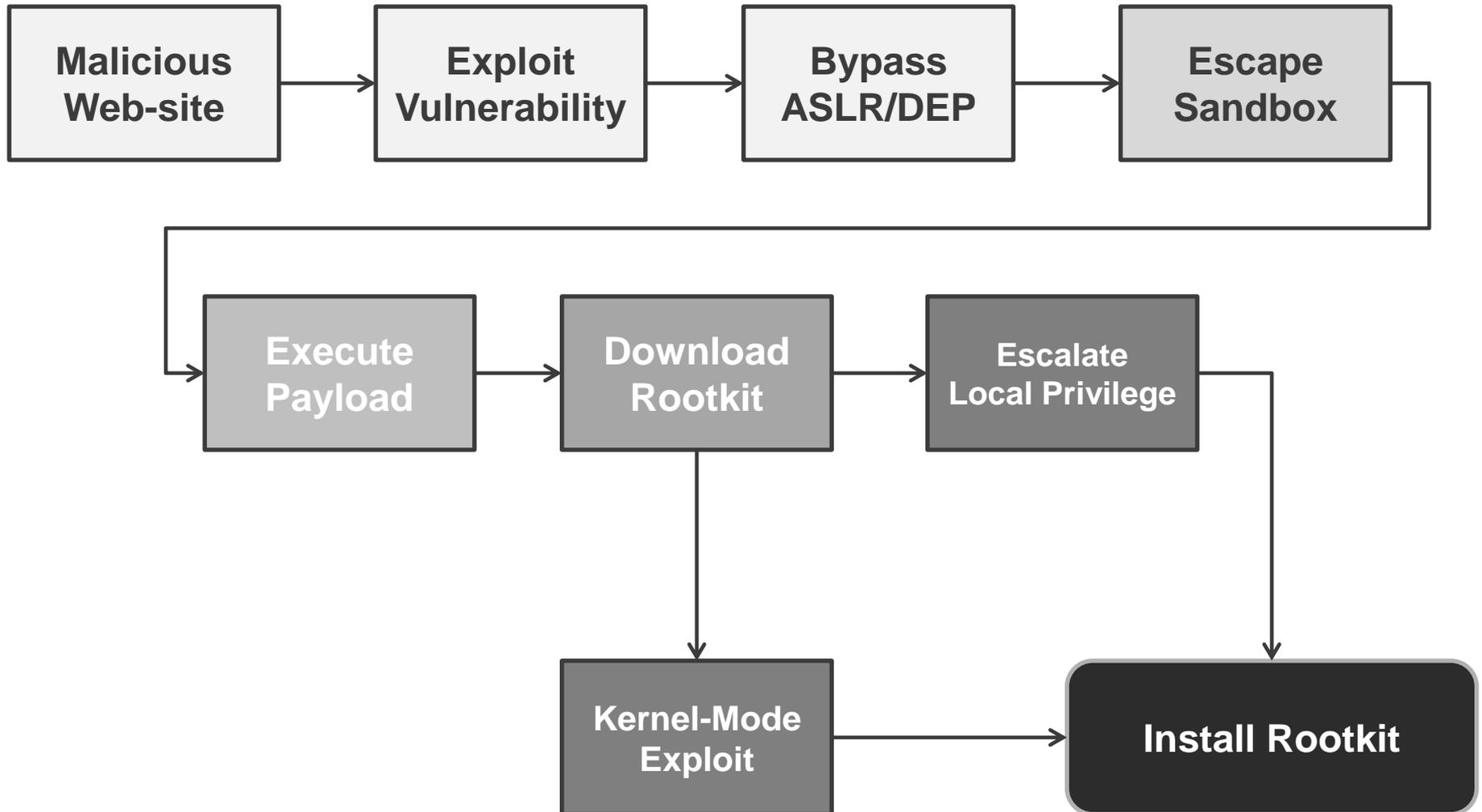


Evolution of Rootkits

Evolution of Rootkit Installation



Evolution of Rootkit Installation



Evolution of Rootkit Features

x86

Dropper

bypassing HIPS/AV

privilege escalation

installing rootkit
driver

Rootkit

self-defense

surviving reboot

injecting payload

User mode

Kernel mode

Evolution of Rootkit Features

x64

Dropper

bypassing HIPS/AV

privilege escalation

installing rootkit
driver

User mode

Rootkit

self-defense

surviving reboot

bypassing signature
check

bypassing
MS PatchGuard

injecting payload

Kernel mode

Obstacles for 64-bit Rootkits

- **Kernel-Mode Code Signing Policy:**
 - ✓ It is “difficult” to load unsigned kernel-mode driver
- **Kernel-Mode Patch Protection (Patch Guard):**
 - ✓ **SSDT (System Service Dispatch Table)**
 - ✓ **IDT (Interrupt Descriptor Table)**
 - ✓ **GDT (Global Descriptor Table)**
 - ✓ **MSRs (Model Specific Registers)**



Bypassing Code Integrity Checks

Types of Integrity Checks

- **PnP Device Installation Signing Requirements**

- **Kernel-Mode Code Signing Policy**

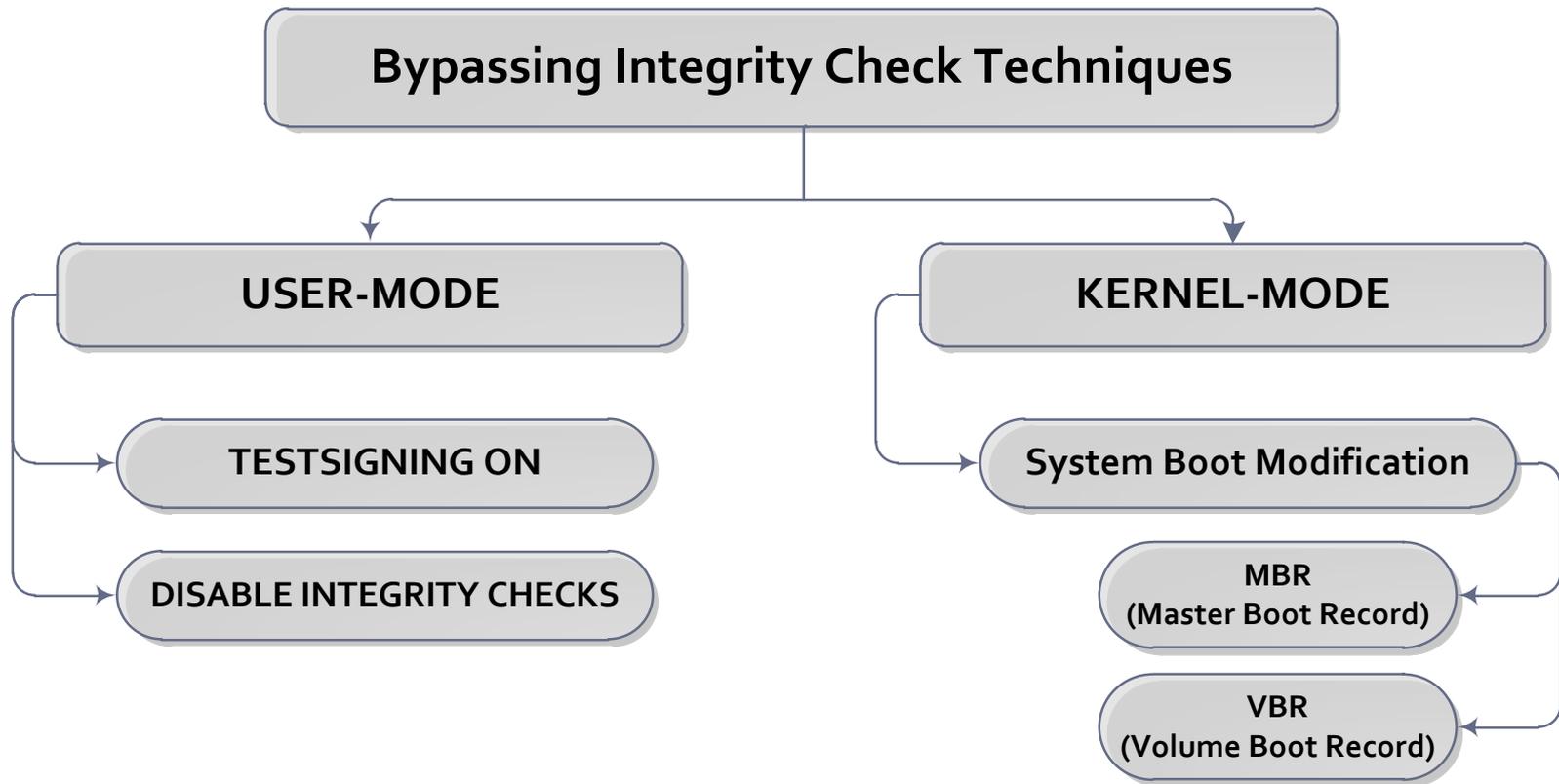
 - ✓ **Enforced on 64-bit version of Windows Vista and later versions**

	64-bit Windows Vista and later	32-bit Windows Vista and later
Boot-start driver	✓	✓
Non boot-start PnP driver	✓	✗
Non boot-start, non-PnP driver	✓	✗ (except stream protected media drivers)

Subverting KMCSP

- **Abusing vulnerable signed legitimate kernel-mode driver**
- **Switching off kernel-mode code signing checks by altering BCD data:**
 - ✓ **abusing WinPe Mode**
 - ✓ **disabling signing check**
 - ✓ **enabling test signing**
- **Patching Bootmgr and OS loader**

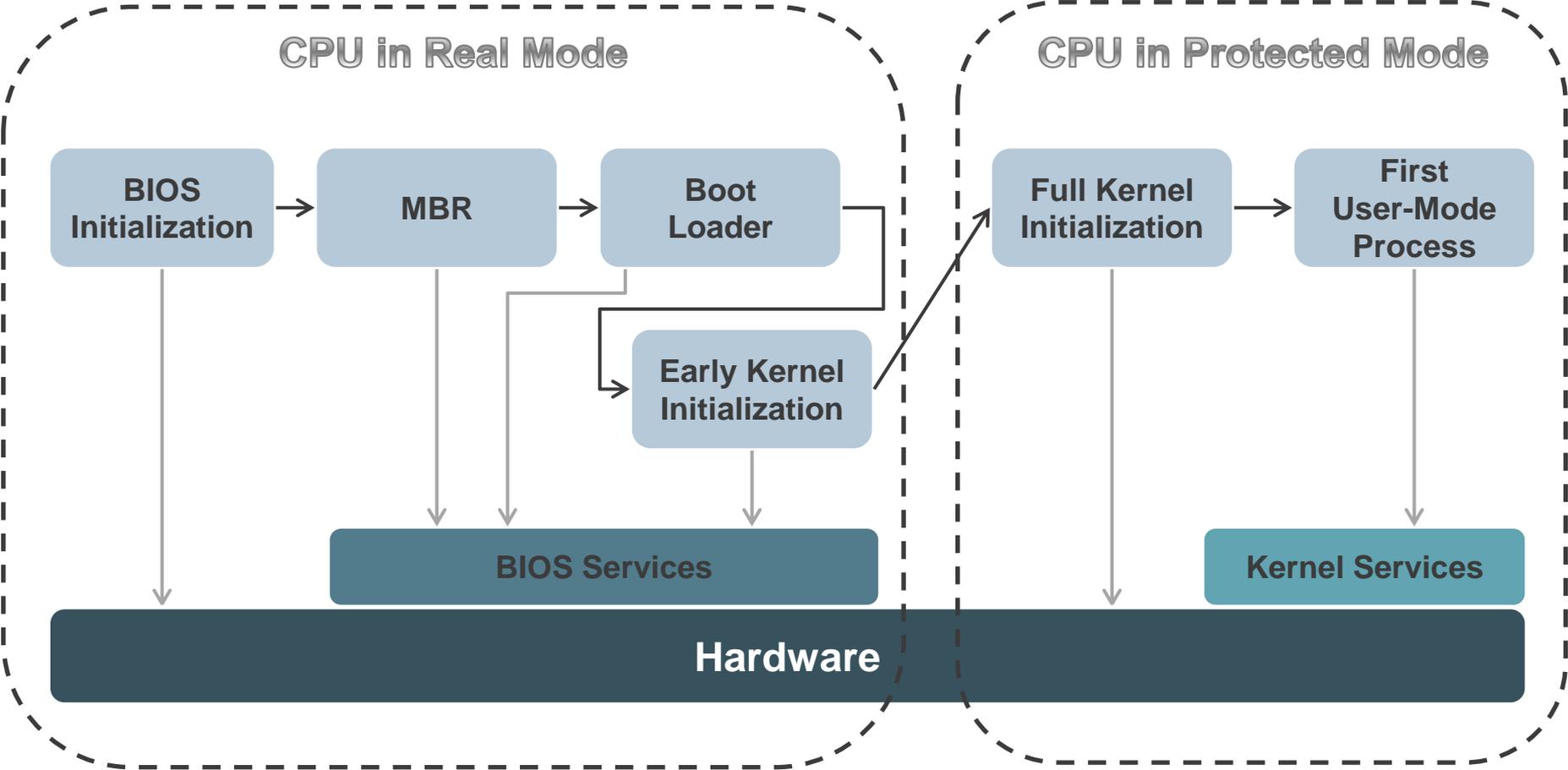
Bypassing Integrity Checks



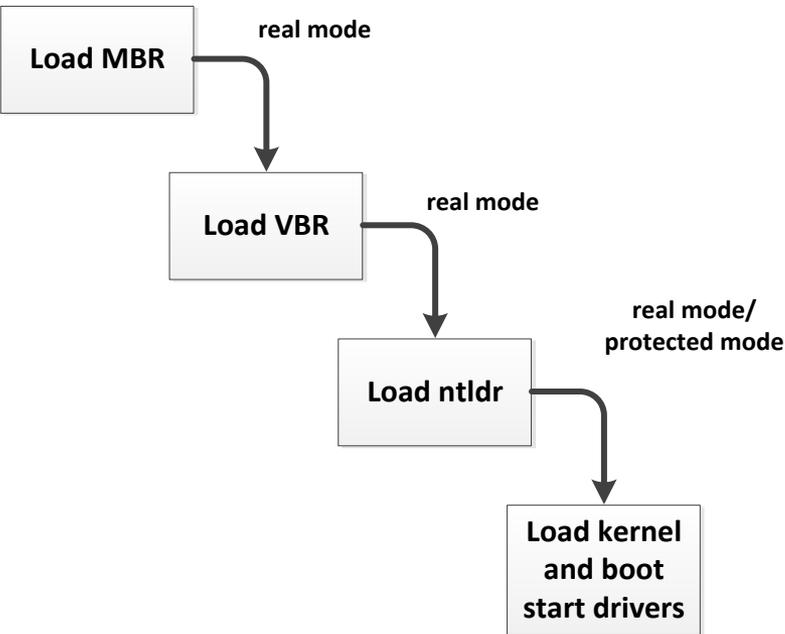


Attacking Windows Bootloader

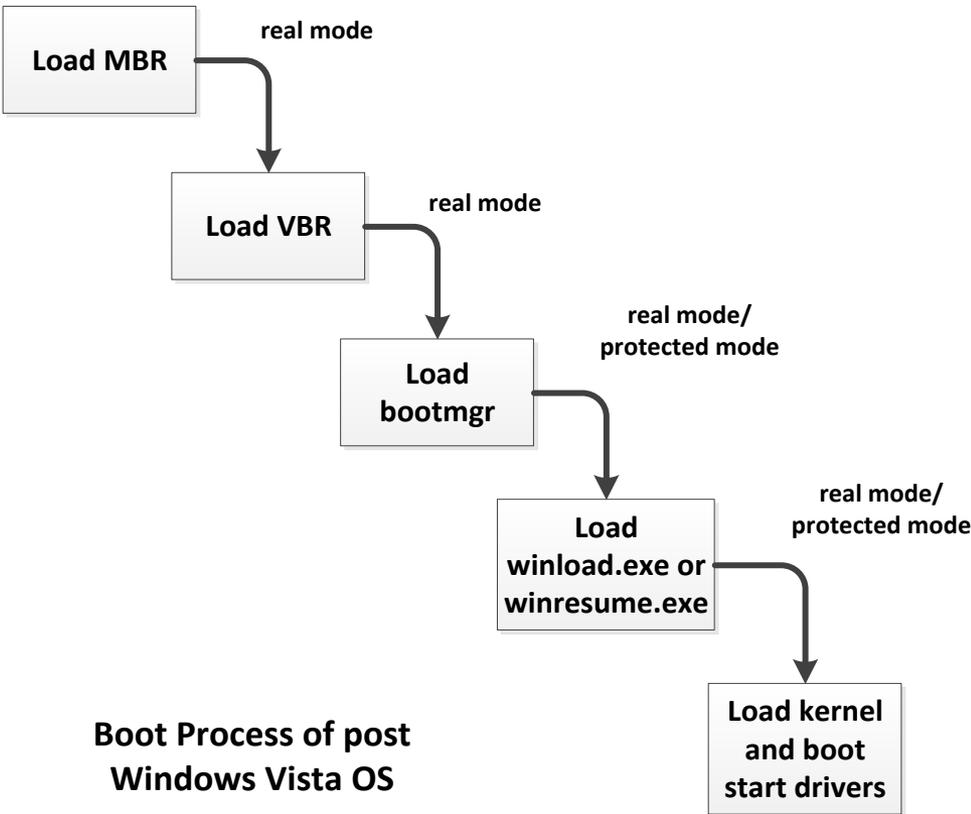
Boot Process



Boot Process of Windows OS



Boot Process of pre Windows Vista OS

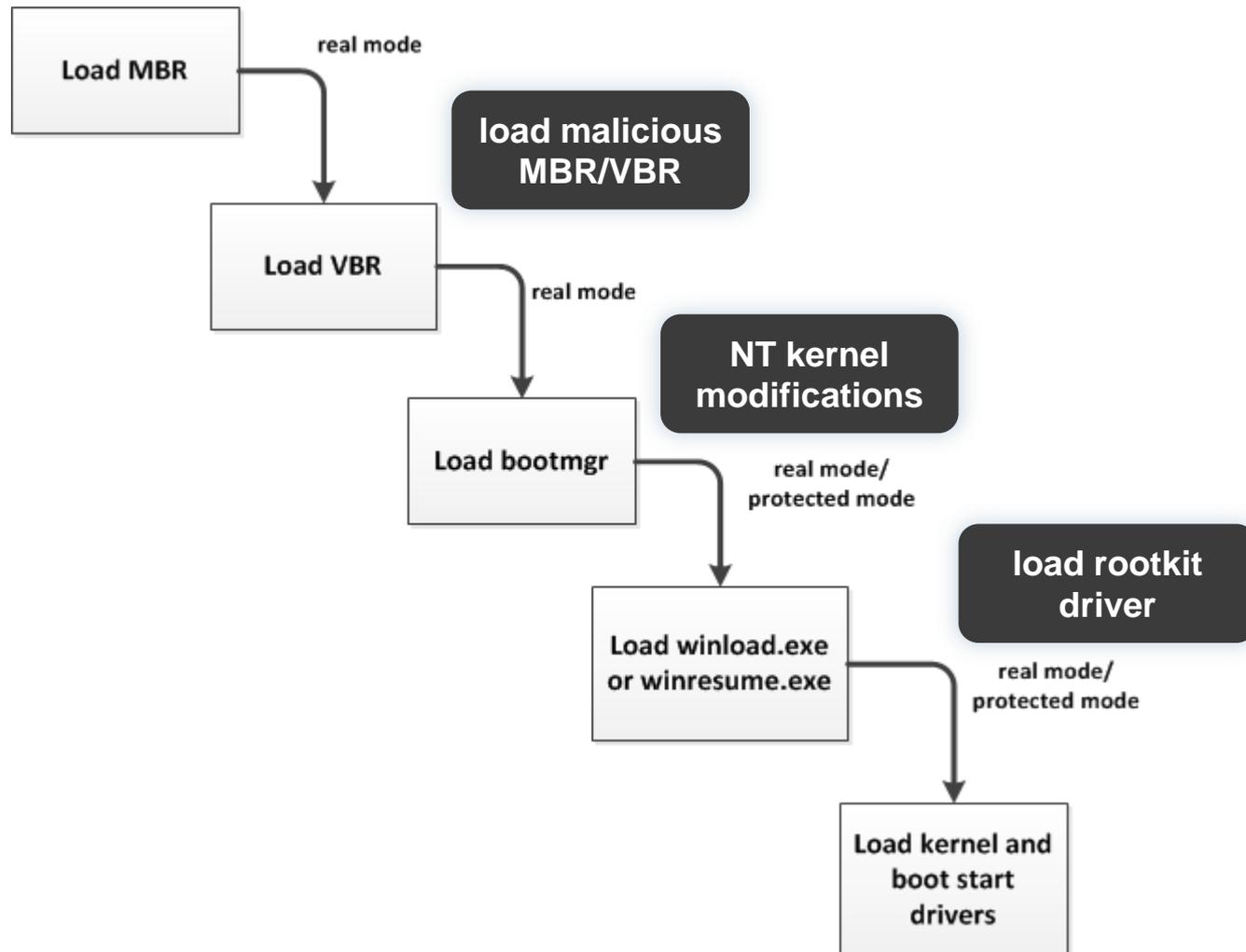


Boot Process of post Windows Vista OS

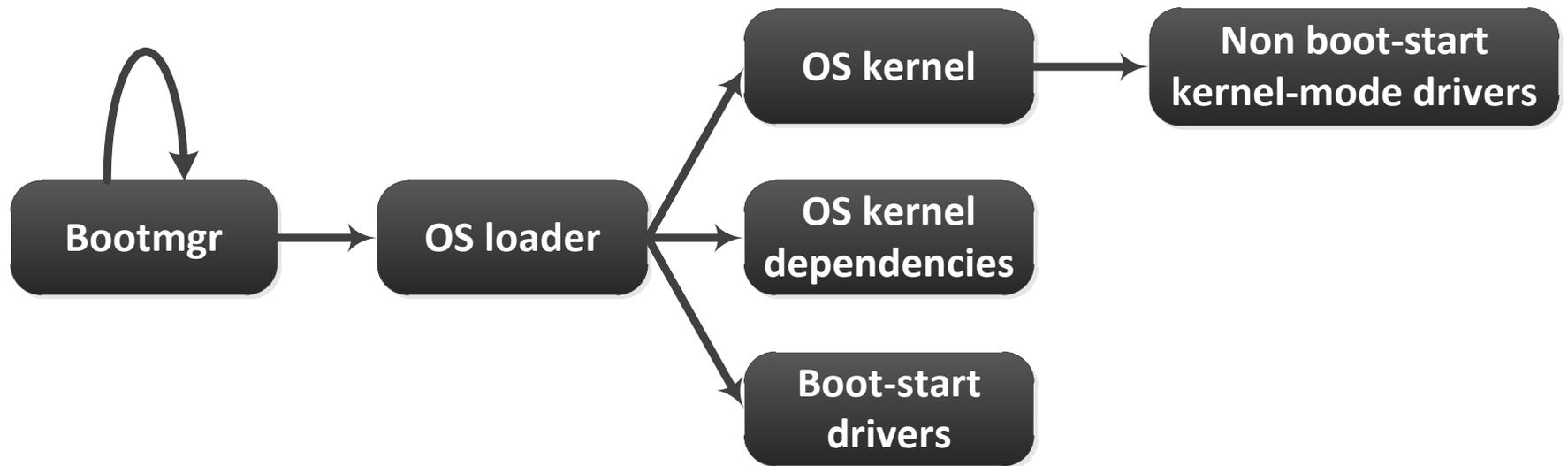
MBR – Master Boot Record

VBR – Volume Boot Record

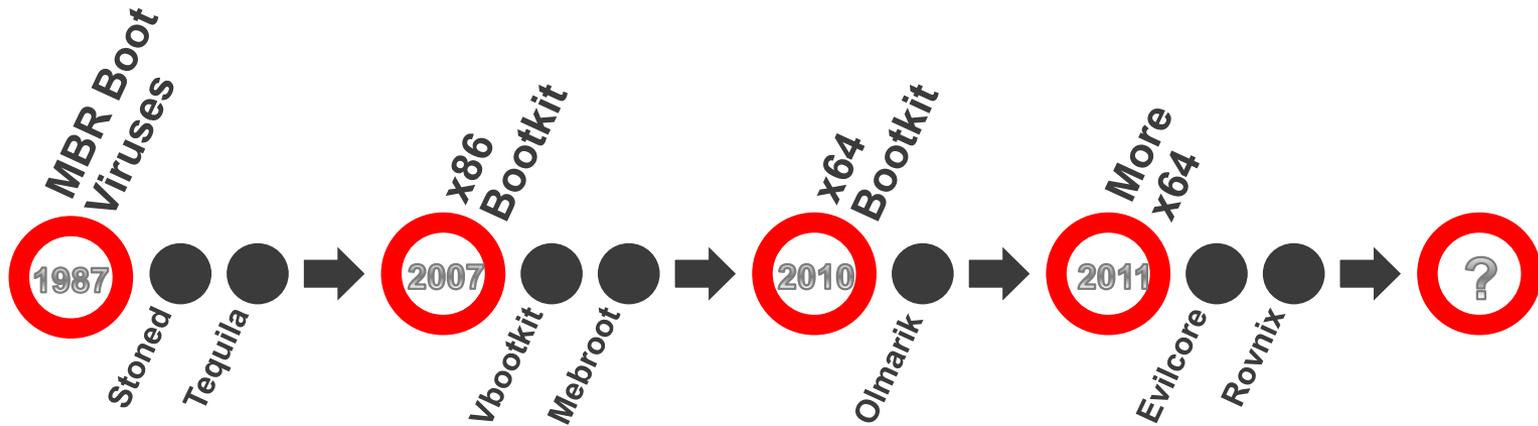
Boot Process with Bootkit Infection



Code Integrity Check



Evolution of Bootkits



○ Bootkit PoC evolution:

- ✓ eEye Bootroot (2005)
- ✓ Vbootkit (2007)
- ✓ Vbootkit v2 (2009)
- ✓ Stoned Bootkit (2009)
- ✓ Evilcore x64 (2011)

○ Bootkit Threats evolution:

- ✓ Win32/Mebrook (2007)
- ✓ Win32/Mebratix (2008)
- ✓ Win32/Mebrook v2 (2009)
- ✓ Win64/Olmarik (2010/11)
- ✓ Win64/Rovnix (2011)



Win64/Olmarik

wanted

◆◆◆ DEAD OR ALIVE ◆◆◆

DOGMA MILLIONS

Главная Регистрация FAQ Топ10 Соппорт ru ▼

Присоединяйся СЕЙЧАС!

Логин:
Пароль:
 помечать меня
 забыли пароль?

60-70% От дохода
3-5% С рефералов

Наши преимущества

- Лучший выхлоп среди аналогичных решений
- Стабильные выплаты
- Надежность сотрудничества
- Индивидуальный подход
- Дружественный саппорт
- Активное совершенствование конвертации

Дополнительная информация

Успешно конвертируем следующие страны: US, CA, AU, GB, DE, FR. Увеличена долговечность работы и выхлоп с каждого инсталла. Мы готовы предложить индивидуальные рейты и условия оплаты постоянным партнерам. Вы можете использовать собственные лендинги для сбора веб трафика.

Стандартные условия

Вы получаете 60% от общего дохода с инсталлов.
Вы получаете 3% от дохода привлеченных Вами мастеров.
Стабильные выплаты 2 раза в месяц, 1-го и 16-го числа.
Большой выбор способов оплаты - WebMoney, Enpay, Банковской перевод, Enaspay, PayPal и

Новости

29-03-2010
Пересчет статьи
Уважаемые партнеры! Для улучшения качества работы партнерки, проводится техническое переоснащение текущих серверов, а так же добавление новых. В следствии этого некоторое время возможна разбегность баланса с текущим заработком. Данная погрешность будет ликвидирована по окончании технических работ. Ожидаемый срок завершения до 1.04.2010

20-01-2010
возросшие инсталлы
В данный момент так как мы исправили проблему отстукв к нам возвращается часть старых инсталлов и поэтому, если вы видите что у вас прибавилось инсталлы сверх нормы - это доходят старые сделанные вами когда то инсталлы.

wanted

DEAD OR ALIVE

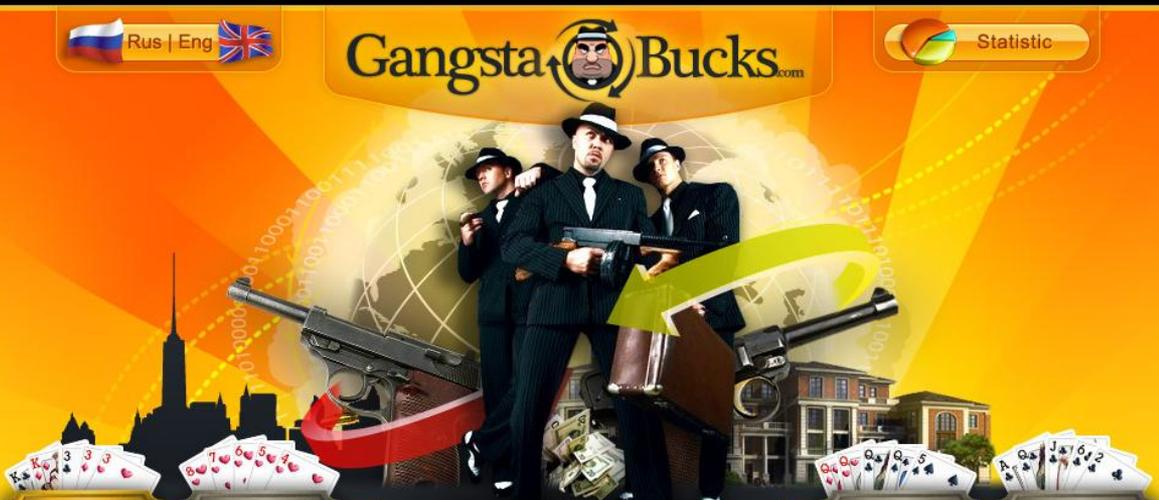


Главная Регистрация FAQ Топ10 Соппорт ru ▼

Rus | Eng

Gangsta Bucks.com

Statistic



Home Conditions Registration Tariffs Contacts

Наши преимущества

- Лучший выхлоп среди аналогичных решений
- Стабильные выплаты
- Надежность сотрудничества
- Индивидуальный подход
- Дружественный саппорт
- Активное совершенствование конвертации

Стандартные условия

Вы получаете 60% от общего дохода с инсталлов.
Вы получаете 3% от дохода привлеченных Вами мастеров.
Стабильные выплаты 2 раза в месяц, 1-го и 16-го числа.
Большой выбор способов оплаты - WebMoney, Енакс, Банковской перевод, Енакспорт, PayPal и

Дополнительная информация

Успешно конвертируем следующие страны: US, CA, AU, GB, DE, FR. Увеличена долговечность работы и выхлоп с каждого инсталла. Мы готовы предложить индивидуальные реиты и условия оплаты постоянным партнерам. Вы можете использовать собственные лендинги для слива веб траффика.

возросшие инсталлы

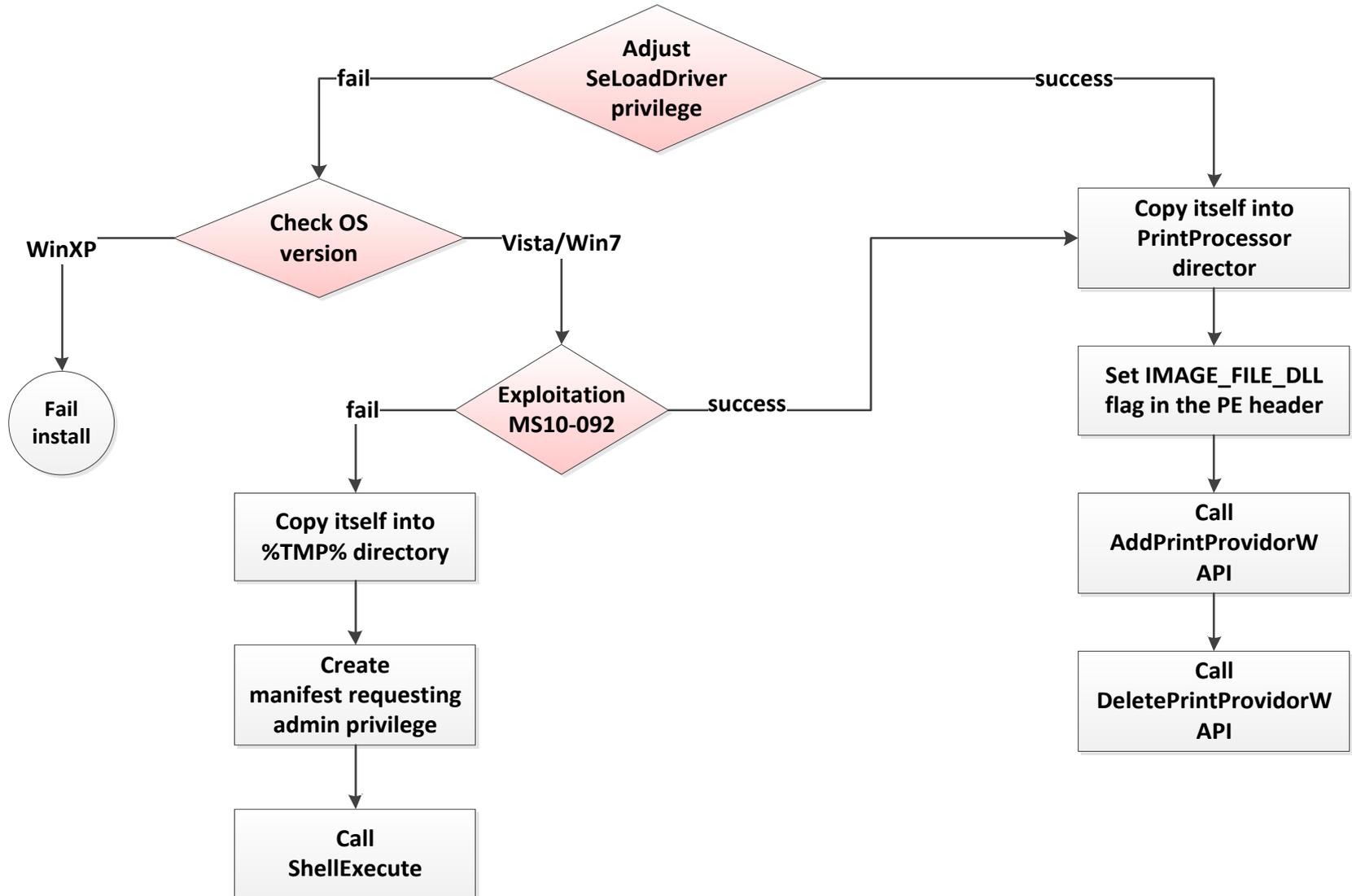
В данный момент так как мы исправили проблему отстукв к нам возвращается часть старых инсталлов и поэтому, если вы видите что у вас прибавились инсталлы сверх нормы - это доходят старые сделанные вами когда то инсталлы.

20-01-2010

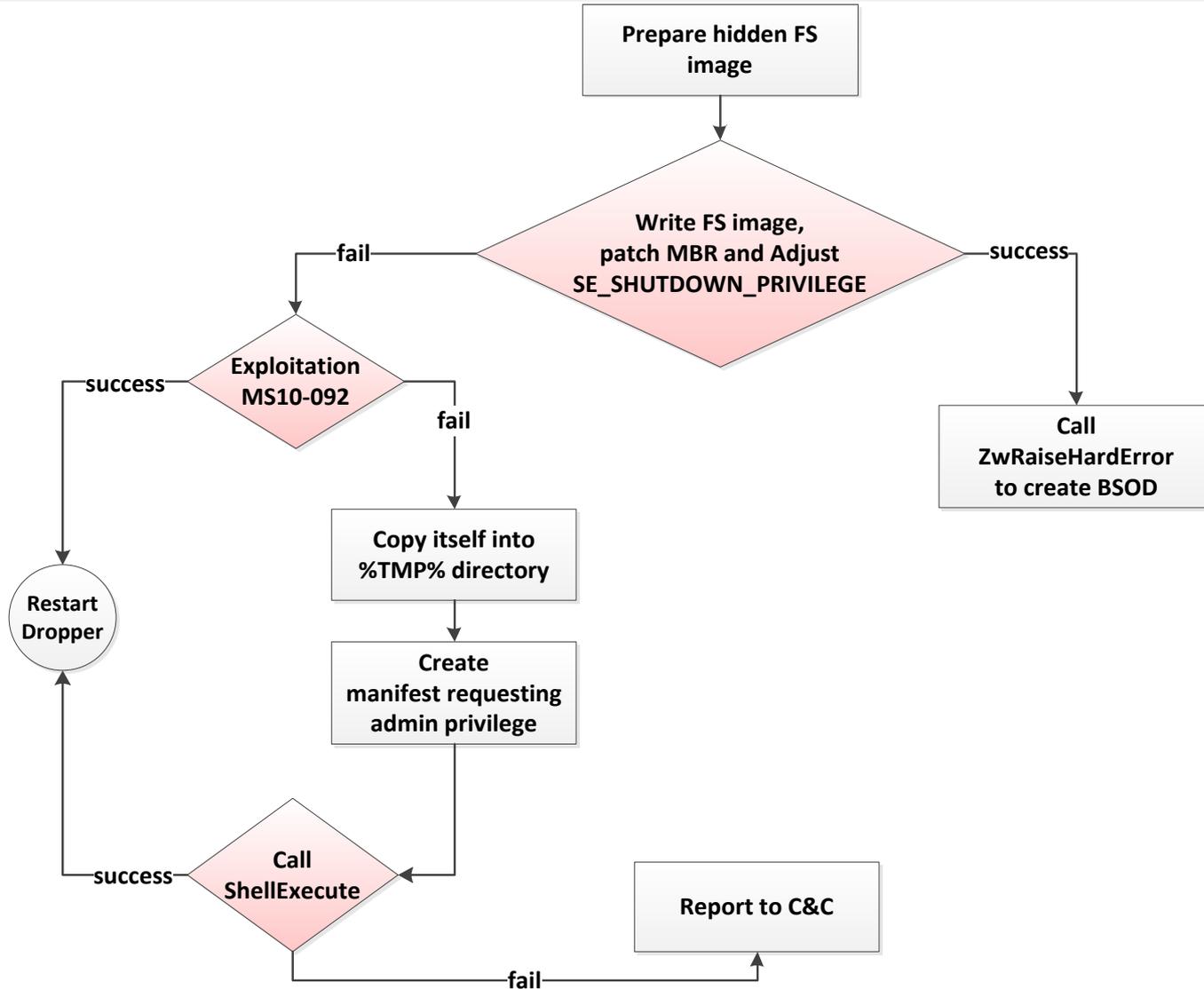


Installation on x86 vs. x64

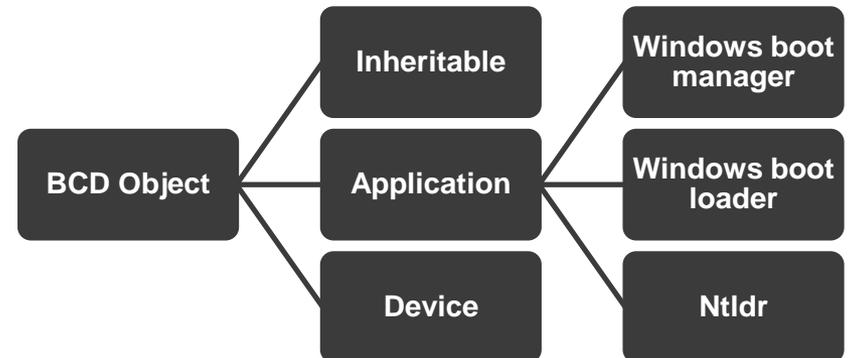
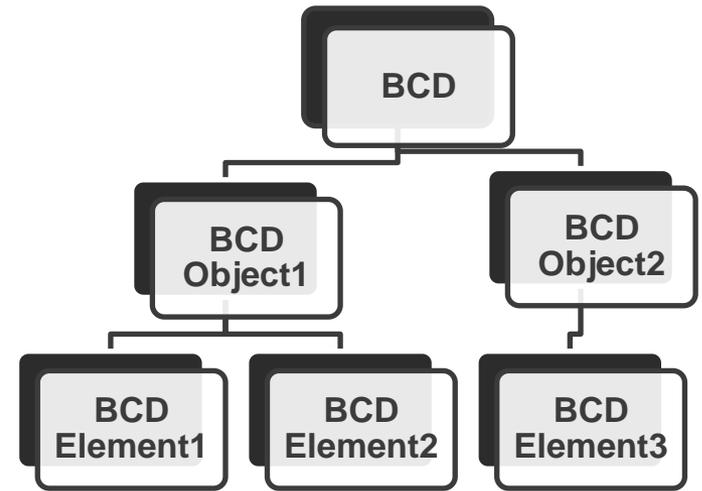
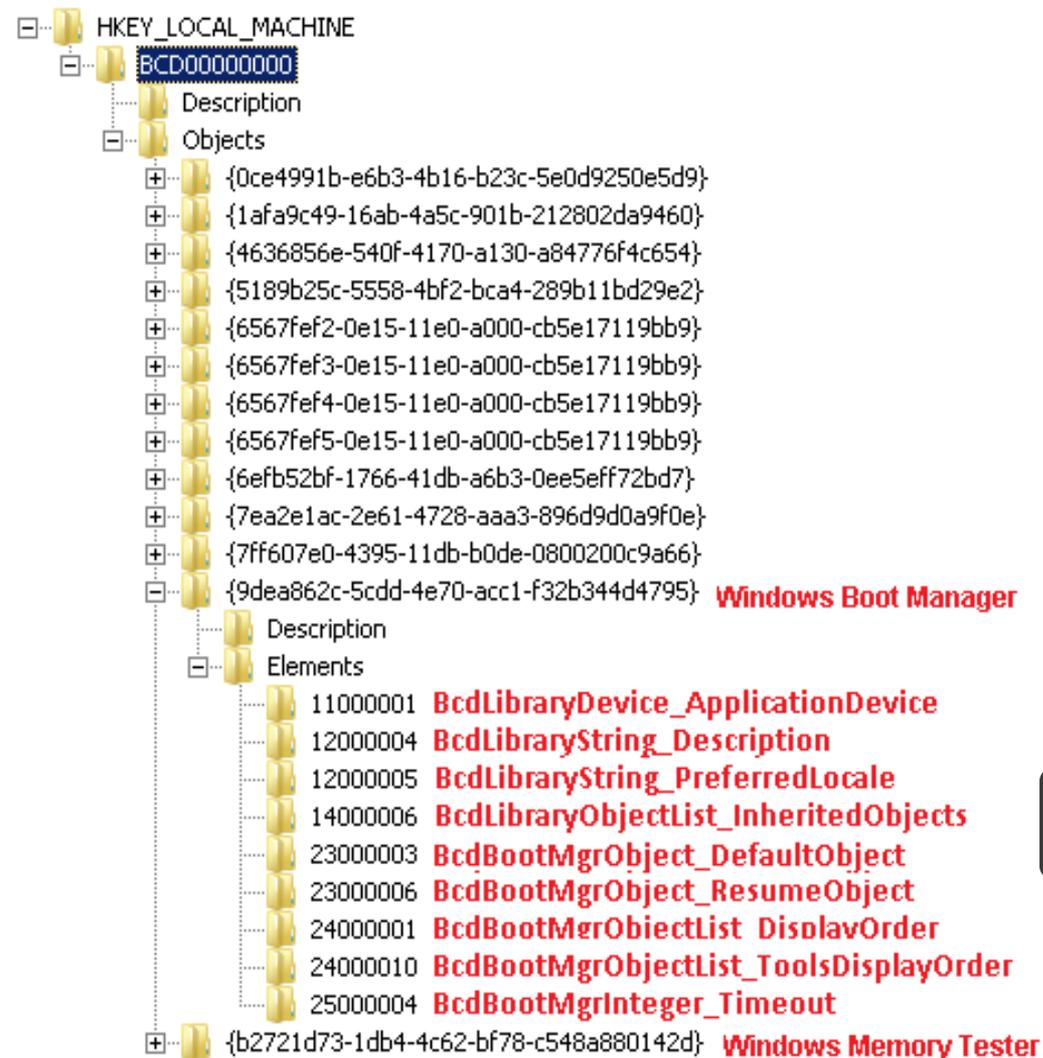
TDL4 Installation on x86



TDL4 Installation on x64



Boot Configuration Data (BCD)



BCD Elements determining KMCSP (before KB2506014)

BCD option	Description
BcdLibraryBoolean_DisableIntegrityCheck (0x16000020)	disables kernel-mode code integrity checks
BcdOSLoaderBoolean_WinPEMode (0x26000022)	instructs kernel to be loaded in preinstallation mode, disabling kernel-mode code integrity checks as a byproduct
BcdLibraryBoolean_AllowPrereleaseSignatures (0x16000049)	enables test signing

BCD Elements determining KMCSP (before KB2506014)

```
; BYTE __stdcall B1ImgQueryCodeIntegrityBootOptions(BYTE *a2, BYTE *pOption, BYTE *AllowPreReleaseSign)
_B1ImgQueryCodeIntegrityBootOptions@12 proc near
    mov     edi, edi
    push   ebp
    mov    ebp, esp
    push   ecx
    push   esi
    mov    esi, [edx+14h]
    lea   eax, [ebp+var_1]
    push   eax
    push   BcdLibraryBoolean_DisableIntegrityChecks
    push   esi
    call   _B1GetBootOptionBoolean@12 ; B1GetBootOptionBoolean(x,x,x)
    test   eax, eax
    jge   short loc_428742
    mov    [ebp+var_1], 0

loc_428742:
    test   byte ptr [edx], 4
    jz    short loc_428764
    cmp    [ebp+var_1], 0
    jnz   short loc_428764
    lea   eax, [ebp+var_1]
    push   eax
    push   BcdLibraryBoolean_WinPEEnabled
    push   esi
    call   _B1GetBootOptionBoolean@12 ; B1GetBootOptionBoolean(x,x,x)
    test   eax, eax
    jge   short loc_428764
    mov    [ebp+var_1], 0

loc_428764:
    ; CODE XREF: B1ImgQueryCodeIntegrityBootOptions(x,x,x)+1B1j
    ; B1ImgQueryCodeIntegrityBootOptions(x,x,x)+2A1j ...
    mov    eax, [ebp+pOption]
    mov    cl, [ebp+var_1]
    mov    [eax], cl
    lea   eax, [ebp+var_1]
    push   eax
    push   BcdLibraryBoolean_AllowPrereleaseSignatures
    push   esi
    call   _B1GetBootOptionBoolean@12 ; B1GetBootOptionBoolean(x,x,x)
```

BCD c

BcdLibra
(0x16000

BcdOSL
(0x26000

BcdLibra
(0x16000

tegrity

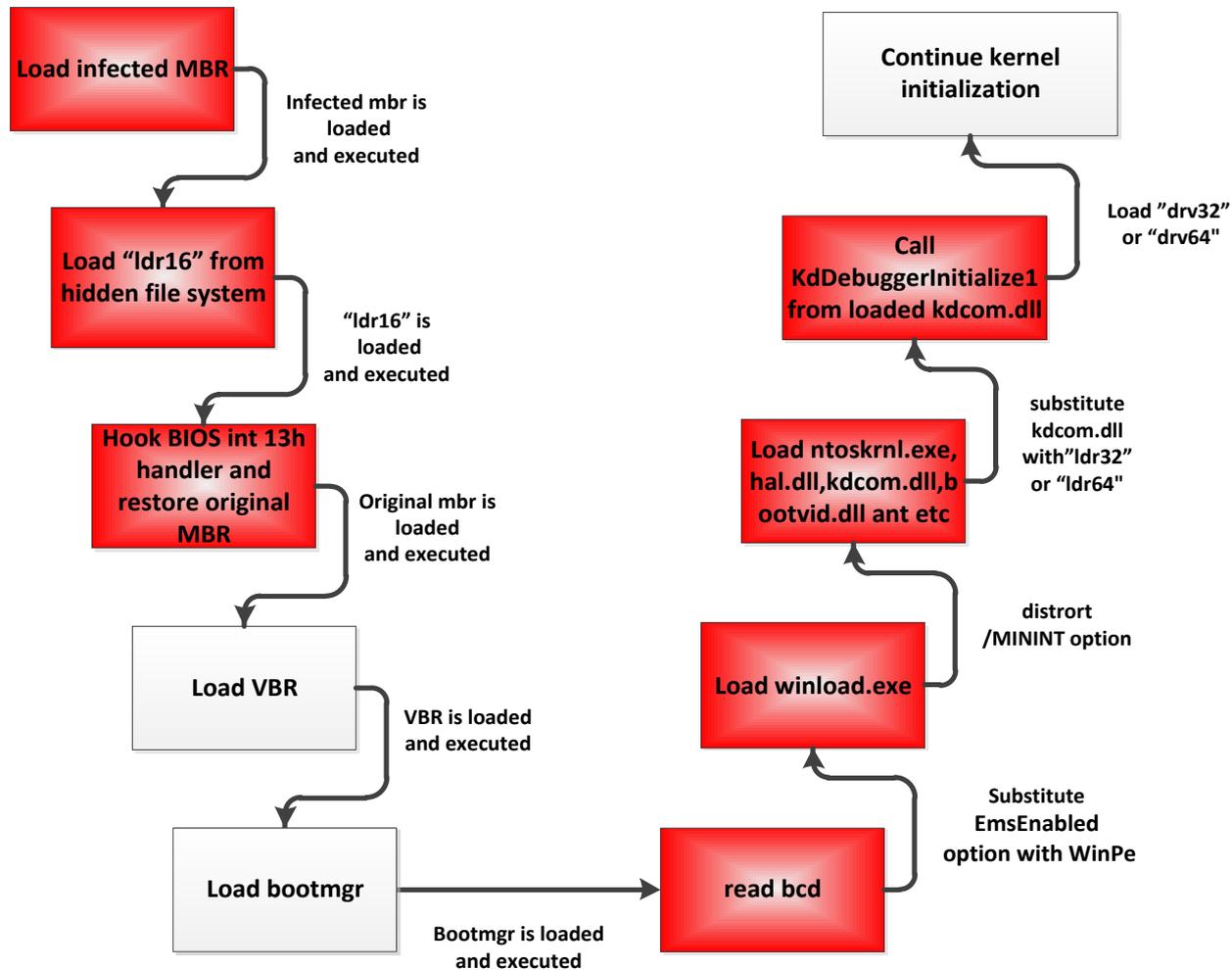
n
g
necks

Abusing Win PE mode: TDL4 modules

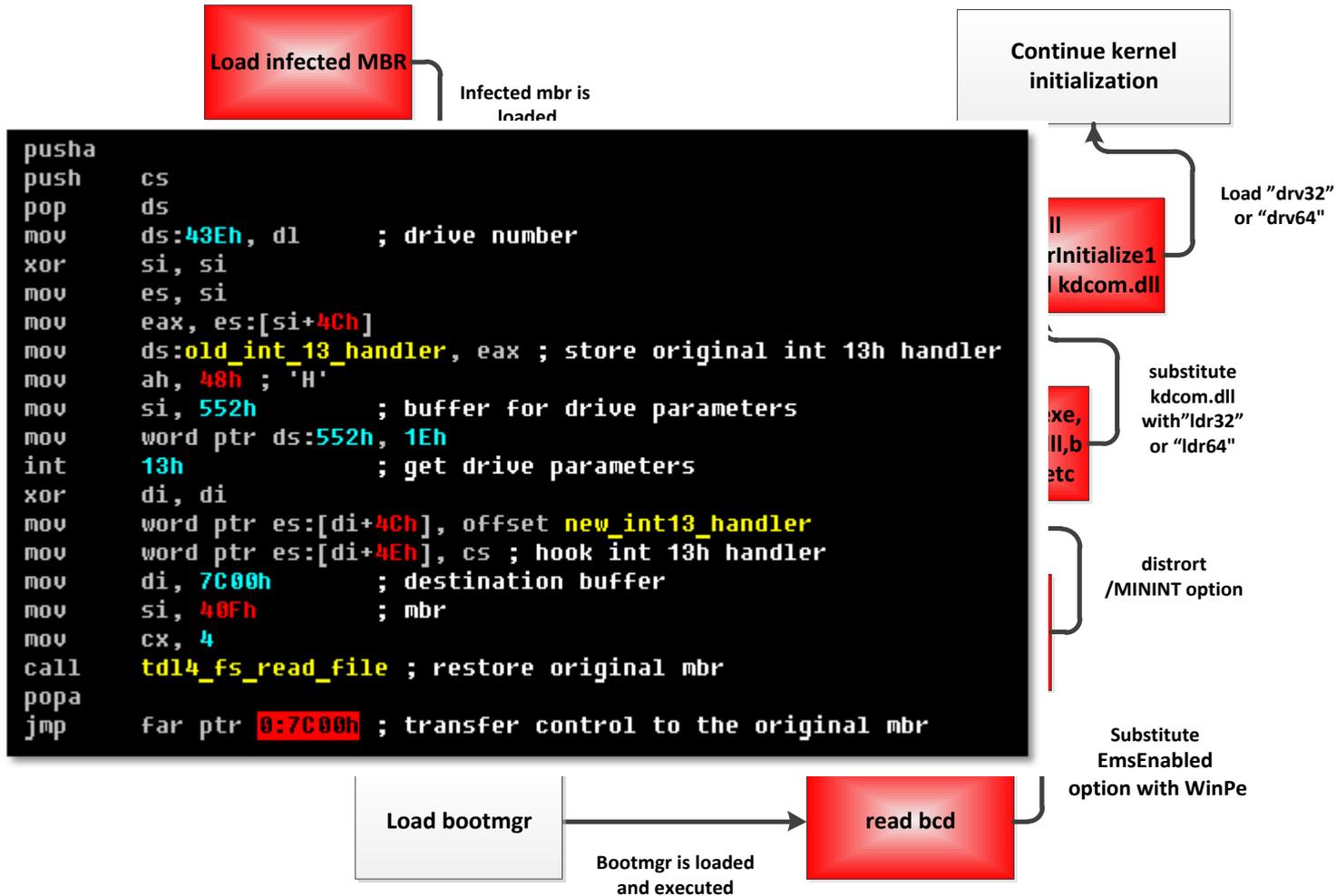
Module name	Description
mbr (infected)	infected MBR loads <i>ldr16</i> module and restores original MBR in memory
ldr16	hooks 13h interrupt to disable KMCSP and substitute <i>kdcom.dll</i> with <i>ldr32</i> or <i>ldr64</i>
ldr32	reads TDL4's kernel-mode driver from hidden file system and maps it into kernel-mode address space
ldr64	implementation of <i>ldr32</i> module functionality for 64-bit OS

int 13h – service provided by BIOS to communicate with IDE HDD controller

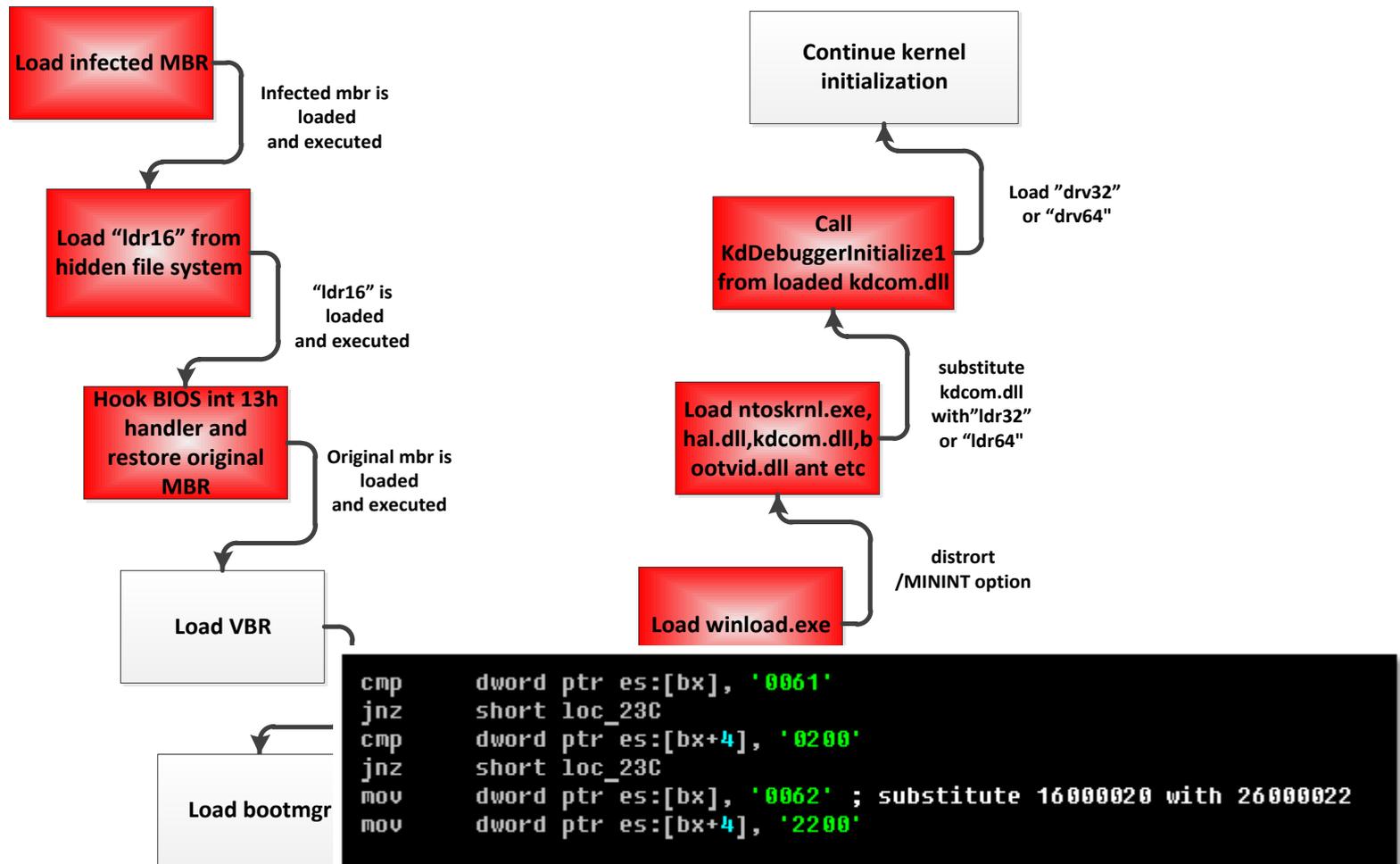
Abusing Win PE mode: Workflow



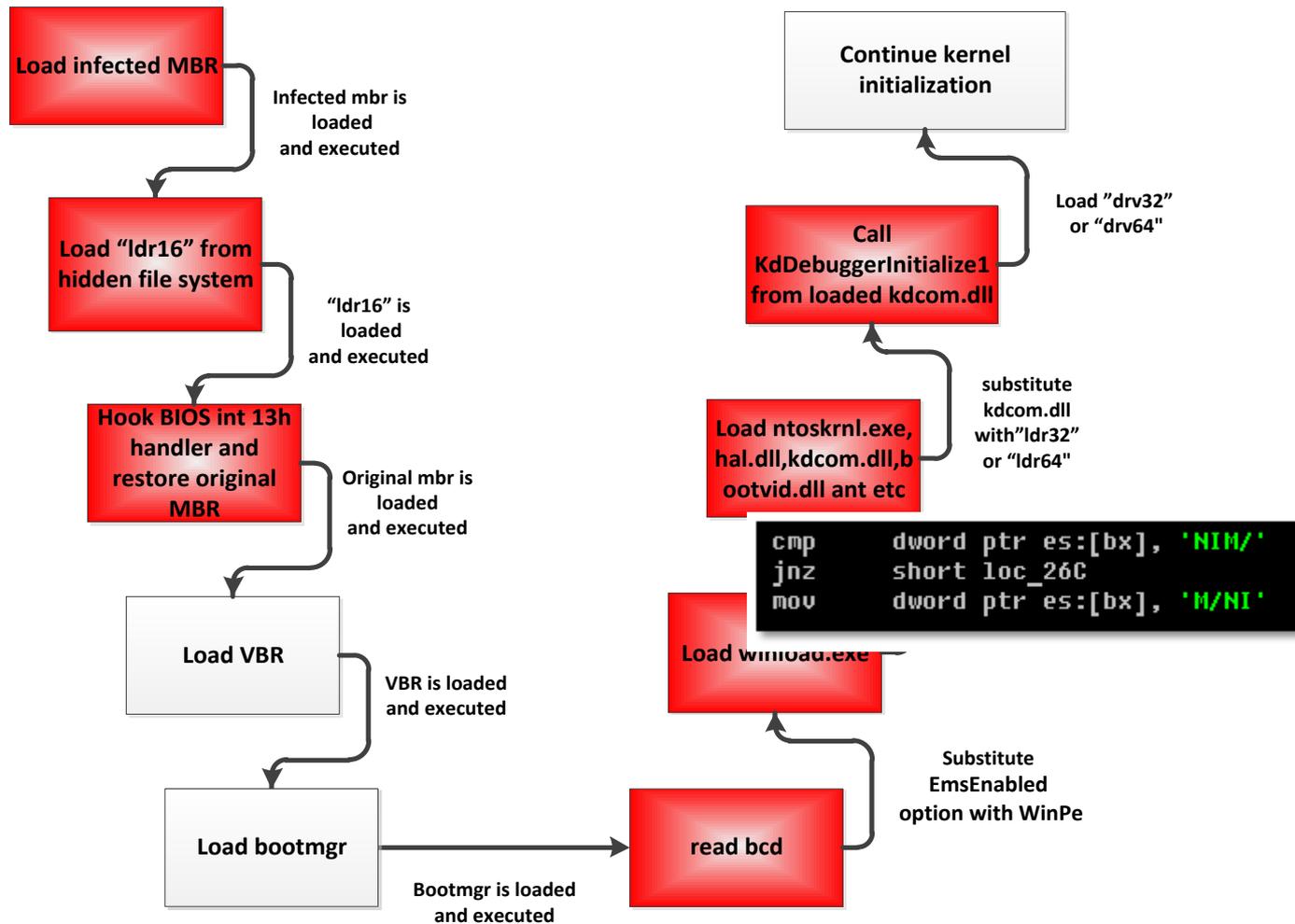
Abusing Win PE mode: Workflow



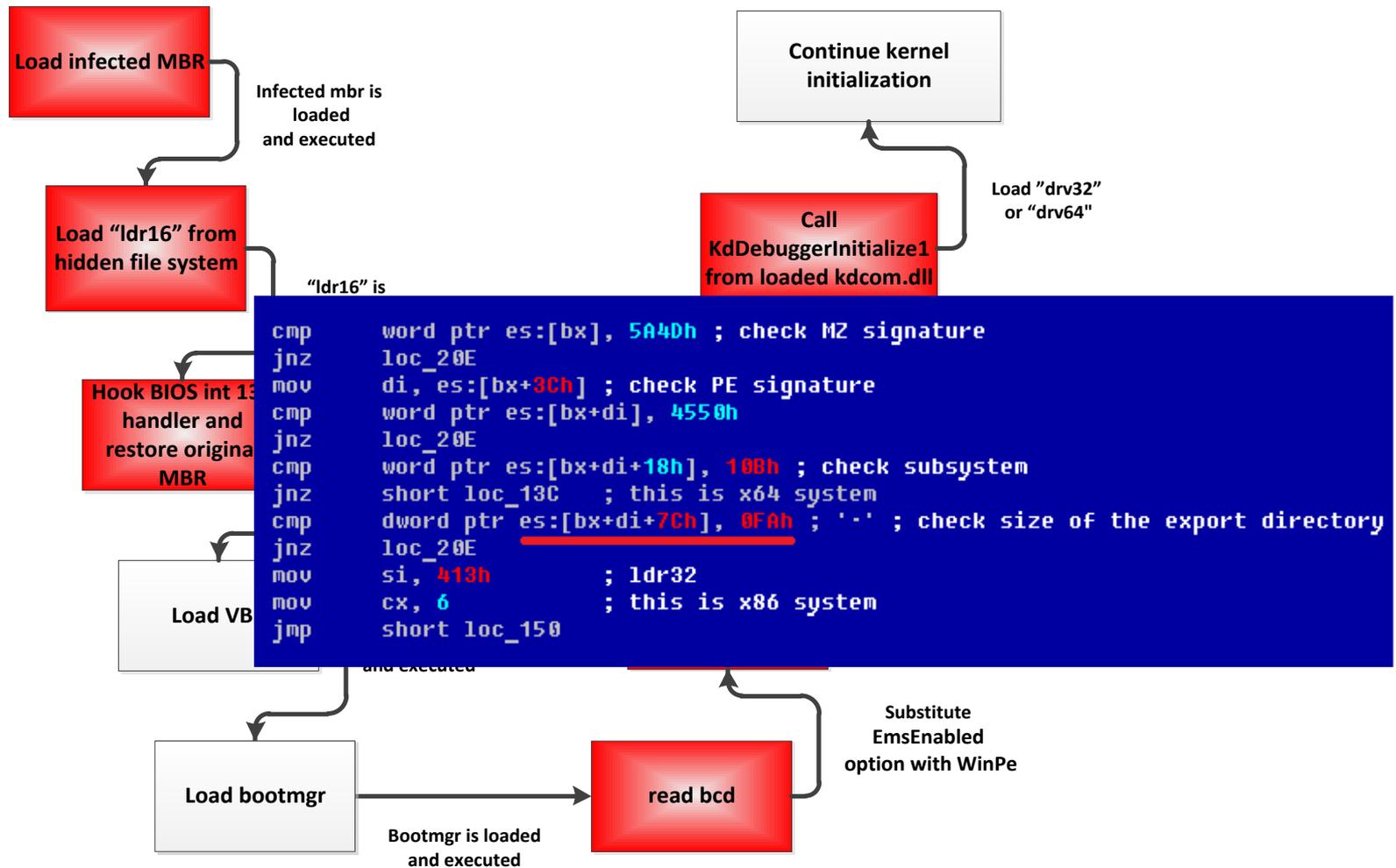
Abusing Win PE mode: Workflow



Abusing Win PE mode: Workflow



Abusing Win PE mode: Workflow



MS Patch (KB2506014)

- ***BcdOsLoaderBoolean_WinPEMode*** option no longer influences kernel-mode code signing policy
- **Size of the export directory of *kdcom.dll* has been changed**

MS Patch (KB2506014)

- *BcdOsLoad* influences
- Size of the changed

```
BLImgQueryCodeIntegrityBootOptions proc near
mov     [rsp+arg_8], rbx
push   rdi
sub    rsp, 20h
mov    r11, [rcx+18h]
mov    rbx, r8
mov    r10, rdx
lea    r8, [rsp+28h+arg_0]
mov    edx, BcdLibraryBoolean_DisableIntegrityCheck
mov    rcx, r11
call   BIGetBootOptionBoolean
movzx  r9d, [rsp+28h+arg_0]
xor    edi, edi
cmp    eax, edi
lea    r8, [rsp+28h+arg_0]
mov    edx, BcdLibraryBoolean_AllowPrereleaseSignatures
cmovl  r9d, edi
mov    rcx, r11
mov    [rsp+28h+arg_0], r9b
mov    [r10], r9b
call   BIGetBootOptionBoolean
movzx  ecx, [rsp+28h+arg_0]
cmp    eax, edi
cmovl  ecx, edi
mov    [rbx], cl
mov    rbx, [rsp+28h+arg_8]
add    rsp, 20h
pop    rdi
retn
BLImgQueryCodeIntegrityBootOptions endp
```

on no longer
cy
has been

MS Patch (KB2506014)

○ *BcdOsLoad*

influences

○ Size of the

changed

```
B!ImgQueryCodeIntegrityBootOptions proc near
```

```
mov     [rsp+arg_8], rbx
push   rdi
sub     rsp, 20h
mov     r11, [rcx+18h]
mov     rbx, r8
mov     r10, rdx
lea     r8, [rsp+28h+arg_0]
mov     edx, BcdLibraryBoolean_DisableIntegrityCheck
mov     rcx, r11
```

Ordinal	Function RVA	Name Ordinal	Name RVA	Name
N/A	00001E3C	00001E7A	00001E60	00001EE6
(nFunctions)	Dword	Word	Dword	szAnsi
00000001	00001014	0000	0000608C	KdD0Transition
00000002	00001014	0001	0000609B	KdD3Transition
00000003	00001020	0002	000060AA	KdDebuggerInitialize0
00000004	00001104	0003	000060C0	KdDebuggerInitialize1
00000005	00001228	0004	000060D6	KdReceivePacket
00000006	00001008	0005	000060E6	KdReserved0
00000007	00001158	0006	000060F2	KdRestore
00000008	00001144	0007	000060FC	KdSave
00000009	00001608	0008	00006103	KdSendPacket

```
add     rsp, 20h
pop     rdi
retn
```

```
B!ImgQueryCodeIntegrityBootOptions endp
```

on no longer

cy

has been

Bypassing KMCSP: Another Attempt

Patch Bootmgr and OS loader (*winload.exe*) to disable KMCSP:

```
                                ; CODE XREF: seg000:0330fj
cmp     byte ptr es:[bx], 0BFh ; '-'
jnz     short loc_354
cmp     dword ptr es:[bx+1], 0C0000428h ; mov edi, C0000428h
jnz     short loc_354
mov     dword ptr es:[bx+1], 0C428h ;     mov edi, 0000C428h
```

Bypassing KMCSP: Another Attempt

Patch Bootmgr and OS loader (*winload.exe*) to disable KMCSP:

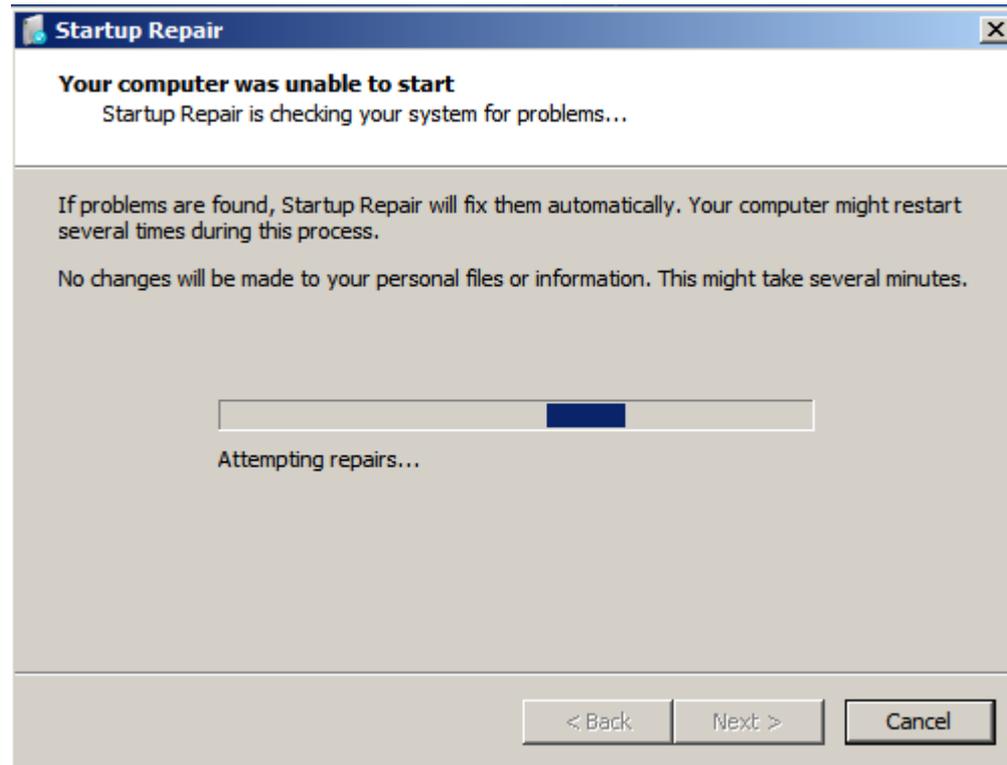
```
; __int32 __stdcall I_CheckImageHashInCatalog(struct _CRYPTOAPI_BLOB *, unsigned __int8 *const )
?I_CheckImageHashInCatalog@@@YAJPEAU_CRYPTOAPI_BLOB@@@QEAE@Z proc near
    ; CODE XREF: MinCrypL_CheckImageHash+2C1p
    ; MinCrypL_CheckImageHash+521p
    ; DATA XREF: ...
```

```
var_88      = dword ptr -88h
var_80      = qword ptr -80h
var_78      = byte ptr -78h
var_68      = dword ptr -68h
var_28      = dword ptr -28h
Source2     = qword ptr -20h
var_18      = byte ptr -18h
arg_0       = dword ptr  8
arg_8       = qword ptr 10h
arg_10      = qword ptr 18h
```

```
48 89 5C 24 10      mov     [rsp+arg_8], rbx
48 89 6C 24 18      mov     [rsp+arg_10], rbp
56                push   rsi
57                push   rdi
41 54              push   r12
48 81 EC 90 00 00 00 sub     rsp, 90h
8B 19              mov     ebx, [rcx]
48 8B 69 08        mov     rbp, [rcx+8]
4C 8B E2           mov     r12, rdx
85 DB            test   ebx, ebx
BF 28 04 00 C0     mov     edi, 0C0000428h ; STATUS_INVALID_IMAGE_HASH
```

Bypassing KMCSP: Result

Bootmgr fails to verify OS loader's integrity



Bypassing KMCSP: Result

Bootmgr fails to verify OS loader's integrity

```
A problem has been detected and windows has been shut down to prevent damage to your computer.
```

```
PAGE_FAULT_IN_NONPAGED_AREA
```

```
If this is the first time you've seen this stop error screen, restart your computer. If this screen appears again, follow these steps:
```

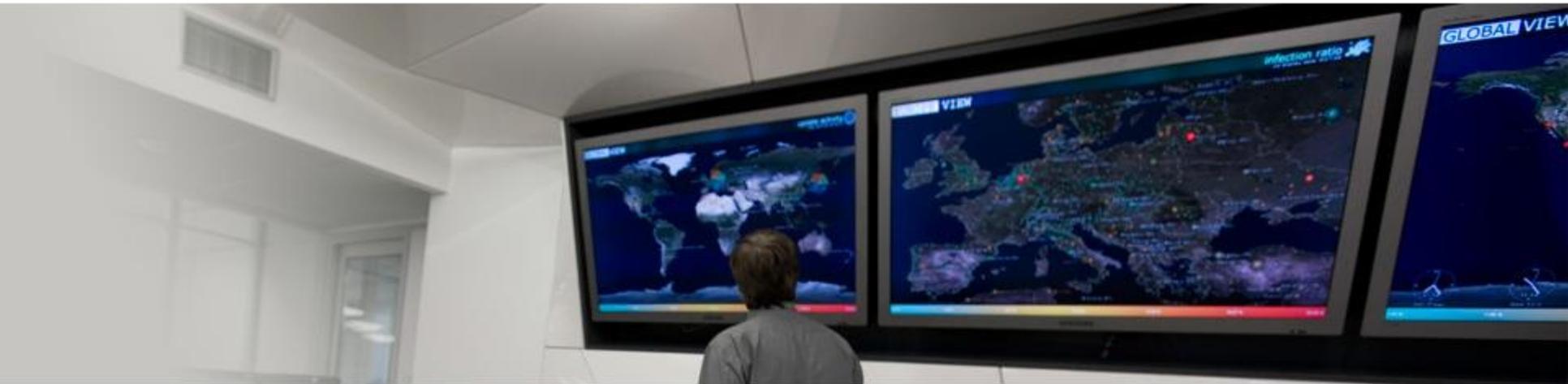
```
Check to make sure any new hardware or software is properly installed. If this is a new installation, ask your hardware or software manufacturer for any windows updates you might need.
```

```
If problems continue, disable or remove any newly installed hardware or software. Disable BIOS memory options such as caching or shadowing. If you need to use safe mode to remove or disable components, restart your computer, press F8 to select Advanced Startup Options, and then select safe Mode.
```

```
Technical information:
```

```
*** STOP: 0x00000050 (0xC1DB09A7,0x00000000,0x8050FD6A,0x00000000)
```

**MS10-015
kills TDL3**

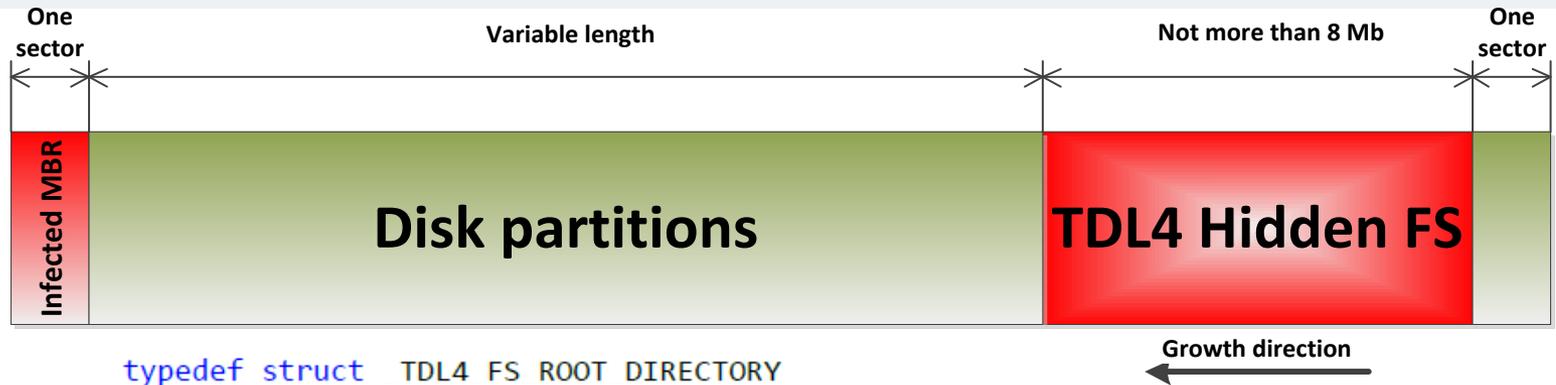


TDL4 Hidden File Systems

TDL's Hidden Storage

- Reserve space in the end of the hard drive (not visible at file system level analysis)
- Encrypted contents (stream cipher: RC4, XOR-ing)
- Implemented as a hidden volume in the system
- Can be accessed by standard APIs (*CreateFile*, *ReadFile*, *WriteFile*, *SetFilePointer*, *CloseHandle*)

TDL4 File System Layout



```
typedef struct _TDL4_FS_ROOT_DIRECTORY
{
    // Signature of the block
    // DC - root directory
    WORD Signature;
    // Set to zero
    DWORD Reserved;
    // Array of entries corresponding to files in FS
    TDL4_FS_FILE_ENTRY FileTable[15];
}TDL4_FS_ROOT_DIRECTORY, *PTDL4_FS_ROOT_DIRECTORY;

typedef struct _TDL4_FS_FILE_ENTRY
{
    // File name - null terminated string
    char FileName[16];
    // Offset from beginning of the file system to file
    DWORD FileBlockOffset;
    // Reserved
    DWORD dwFileSize;
    // Time and Date of file creation
    FILETIME CreateTime;
}TDL4_FS_FILE_ENTRY, *PTDL4_FS_FILE_ENTRY;
```



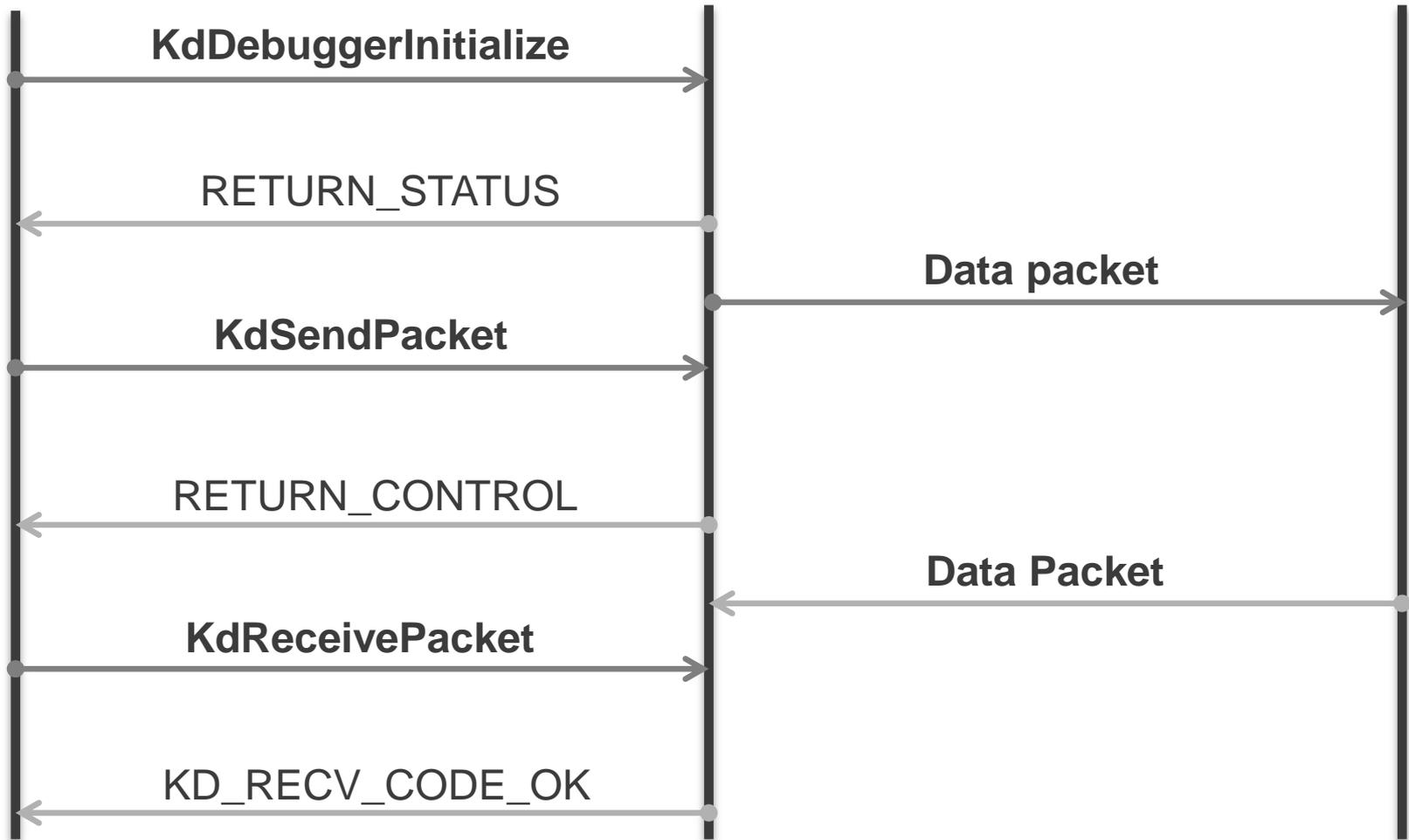
Debugging Bootkit with WinDbg

WinDbg and kdcom.dll

NTOSKRNL

KDCOM.DLL

WinDbg



TDL4 and kdcom.dll

original routine

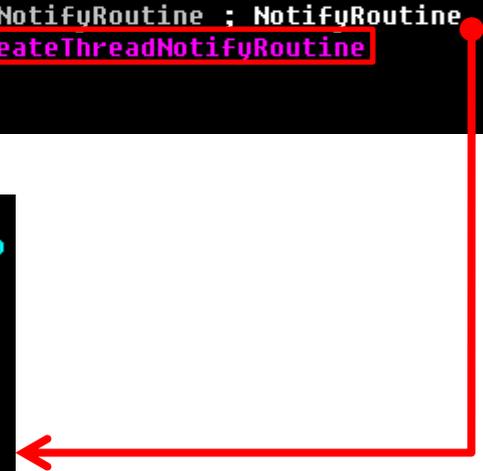
```
; __stdcall KdDebuggerInitialize1(x)
    public _KdDebuggerInitialize1@4
_KdDebuggerInitialize1@4 proc near      ; DATA XREF: .edata:off_80011328↓o
    call    _KdCompInitialize1@0 ; KdCompInitialize1()
    xor     eax, eax
    retn    4
_KdDebuggerInitialize1@4 endp
```

modified routine

```
    public KdDebuggerInitialize1
KdDebuggerInitialize1 proc near      ; DATA XREF: .text:off_10001058↑o
    push   offset NotifyRoutine ; NotifyRoutine
    call   PsSetCreateThreadNotifyRoutine
    retn   4
KdDebuggerInitialize1 endp
```

```
; void __stdcall NotifyRoutine(HANDLE, HANDLE, BOOLEAN)
NotifyRoutine proc near              ; DATA XREF: CallbackRoutine+1CE↑o
                                      ; KdDebuggerInitialize1↓o
    cmp     dword_100017F0, 0
    jnz    short locret_1000179D
    push   offset DriverEntry
    push   0
    call   IoCreateDriver
    xor     ecx, ecx
    test   eax, eax
    setns  cl
    mov    dword_100017F0, ecx

locret_1000179D:                      ; CODE XREF: NotifyRoutine+7↑j
    retn   0Ch
NotifyRoutine endp
```



TDL4 and kdcom.dll

original export table

Name	Address	Ordinal
KdD0Transition	80010386	1
KdD3Transition	80010386	2
KdDebuggerInitialize0	800103A6	3
KdDebuggerInitialize1	8001044C	4
KdReceivePacket	80010F4C	5
KdRestore	80010460	6
KdSave	80010456	7
KdSendPacket	800111B2	8
HalInitSystem(x,x)	80010CE6	

modified export table

Name	Address	Ordinal
KdD0Transition	1000171A	1
KdD3Transition	10001724	2
KdDebuggerInitialize0	100017A0	3
KdDebuggerInitialize1	100017AC	4
KdReceivePacket	100017DC	5
KdRestore	100017C6	6
KdSave	100017BA	7
KdSendPacket	100017D2	8
DriverEntry	1000172E	

```
; void __stdcall NotifyRoutine(HANDLE, HANDLE, BOOLEAN)
NotifyRoutine proc near
    ; DATA XREF: CallbackRoutine+1CE↑o
    ; KdDebuggerInitialize1↓o
    cmp     dword_100017F0, 0
    jnz    short locret_1000179D
    push   offset DriverEntry
    push   0
    call   InCreateDriver
    xor    ecx, ecx
    test   eax, eax
    setns  cl
    mov    dword_100017F0, ecx

locret_1000179D:
    ; CODE XREF: NotifyRoutine+7↑j
    retn   0Ch
NotifyRoutine endp
```

How to Debug TDL4 with WinDbg

- Patch *ldr16* to disable *kdcom.dll* substitution
- Reboot the system and attach to it with WinDbg
- Manually load *drv32/drv64*

“TDL4 Analysis Paper: a brief introduction and How to Debug It”, Andrea Allievi
http://www.aall86.altervista.org/TDLRootkit/TDL4_Analysis_Paper.pdf

Debugging Bootkits with Bochs

The screenshot displays the IDA Pro interface with the Bochs debugger. The main window shows assembly code for the boot sector, with the instruction pointer (EIP) pointing to the start of the code. A secondary window shows the Bochs BIOS output, including the version information and the boot failure message: "Boot failed: could not read the boot disk". The right-hand side of the interface shows the General registers window, with EIP set to 00007C00 and pointing to BOOT_SECTOR:start. The bottom of the interface shows the Hex View-1 window, displaying the memory contents at address 10000:1FEFFFF0.

```
IDA View-EIP:
00007C00 ;org 7C00h
00007C00 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
00007C00
00007C00 public start
00007C00
00007C00
00007C02
00007C04
00007C07
00007C09
00007C09
00007C0B
00007C0B
00007C0E
00007C11
00007C14
00007C15
00007C17
00007C18
00007C1B
00007C1B
00007C1C
00007C1D
00007C1E
00007C1F
00007C20
00007C21
00007C22
00007C23
00007C24
00007C25
00007C26
00007C27
UNKNOWN 00007C00: CTRL + 3rd button enables mouse    IPS: 49406868    A:    B:    NUM    CAPS    SCRL    HD-0-M    CD-1-M
```

Bochs for Windows - Display

Plex86/Bochs VGABios 0.6c 08 Apr 2009
This VGA/VBE Bios is released under the GNU LGPL

Please visit :
. <http://bochs.sourceforge.net>
. <http://www.nongnu.org/vgabios>

Bochs VBE Display Adapter enabled

Bochs BIOS - build: 02/03/01
\$Revision: 1.257 \$ \$Date: 2001/03/02 14:58:00 \$
Options: apmbios pcibios

ata0 master: Generic 12.0
ata1 master: Generic 12.0

Press F12 for boot menu.

Booting from CD-Rom...
CDROM boot failure code : 0003
Boot failed: could not read the boot disk

Booting from Hard Disk...

General registers

EAX	0000AA55	debug001:A455	OF	0
EBX	00000000	IUTABLE:0000	DF	0
ECX	00000000	IUTABLE:0000	IF	0
EDX	00000000	IUTABLE:0000	TF	0
ESI	000E0000	ROMEXT:10000	SF	1
EDI	0000F770	debug001:F170	ZF	0
EBP	00000000	IUTABLE:0000	AF	0
ESP	0000FFD6	debug001:F9D6	PF	0
EIP	00007C00	BOOT_SECTOR:start	CF	0

Modules

Path	Base	Size
BOCHS_DISKIMAGE_LDR		

Threads

Decimal	Hex	State
1	1	Ready

Hex View-1

```
10000:1FEFFFF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

Stack view

```
0060:F9D6 F000AB00
0060:F9DA 00000002 IUTABLE:0002
0060:F9DE 00000000 IUTABLE:0000
UNKNOWN 0000FFD6: debug001:F9D6
```

DEMO

<http://www.youtube.com/watch?v=sT6N7Dr-G6s>

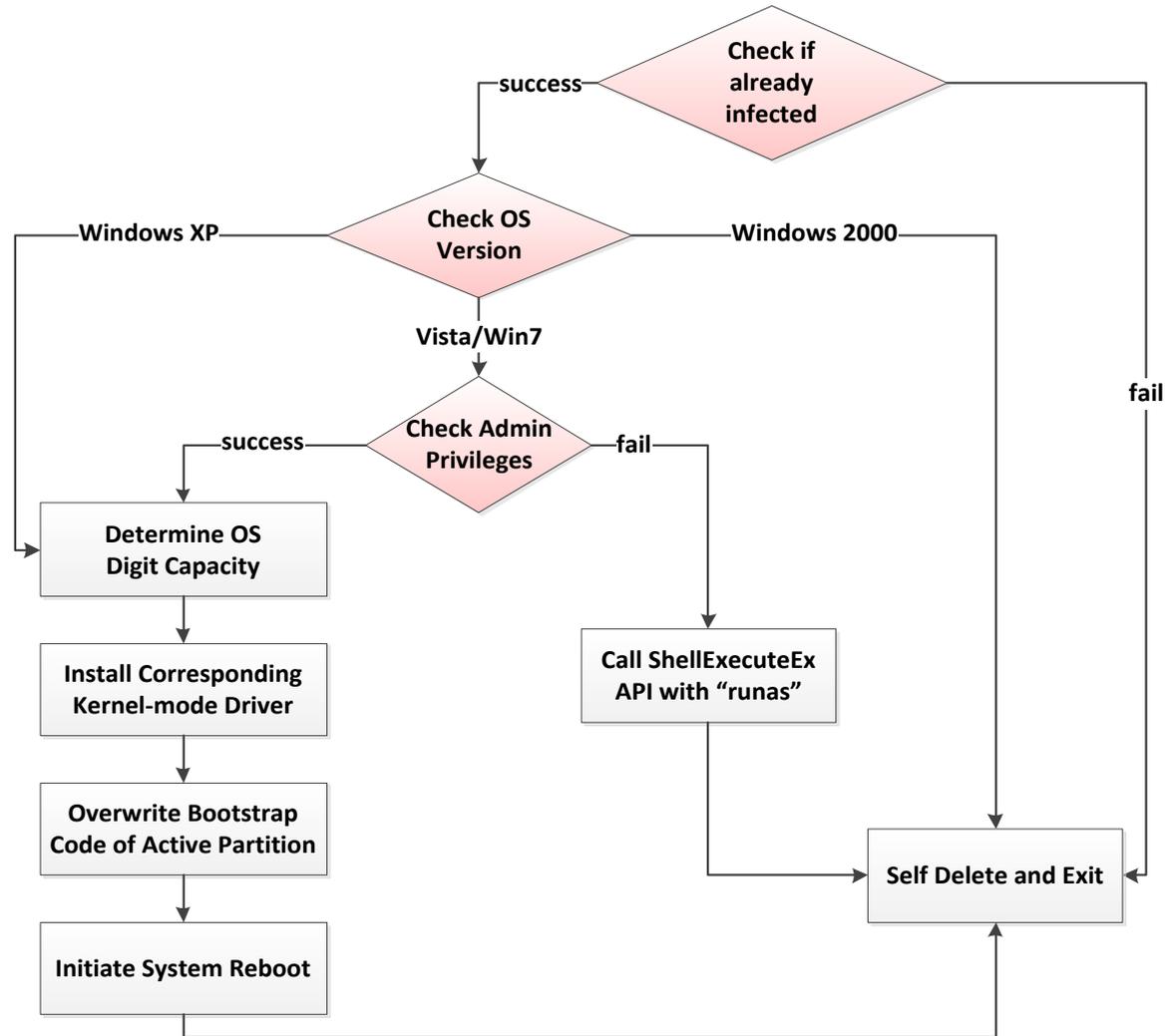




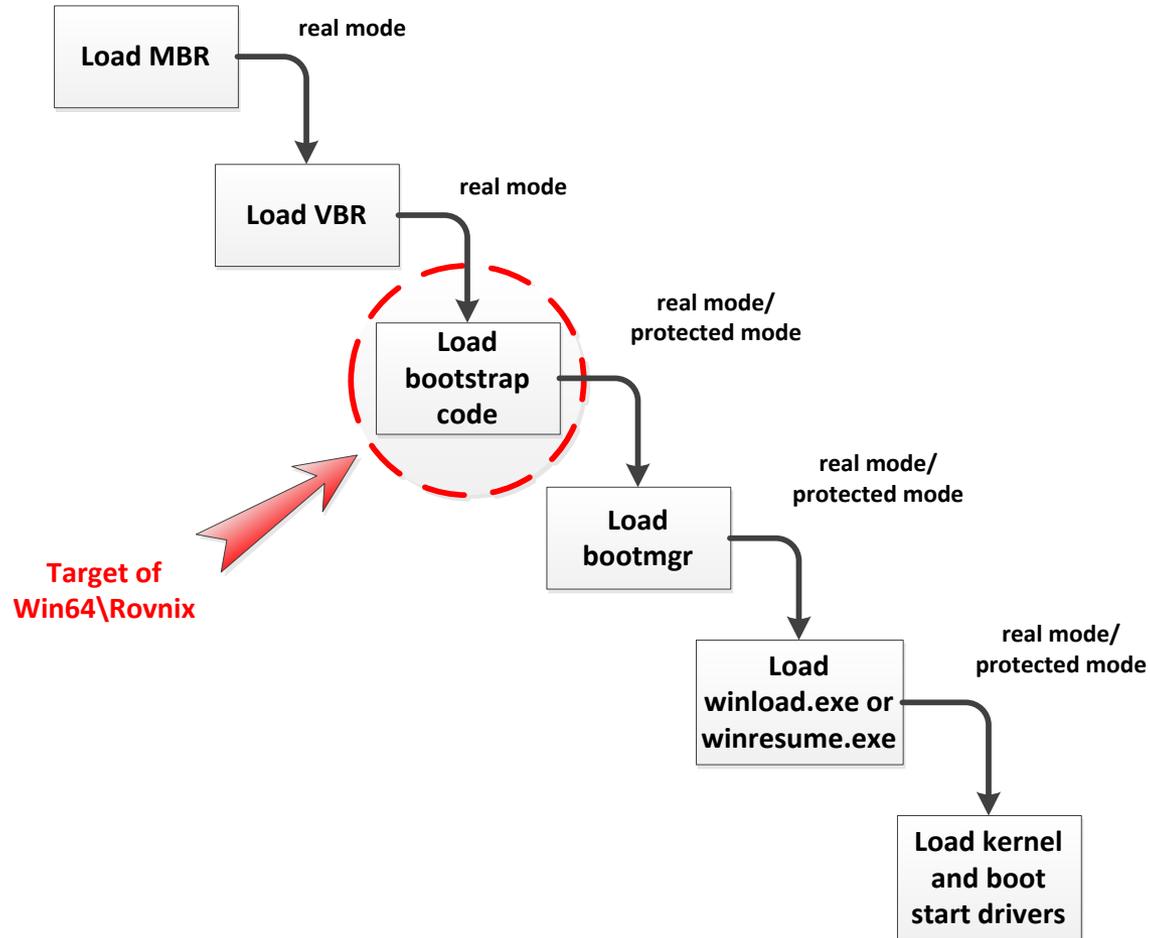
Win64/Rovnix



Win64/Rovnix: Installation

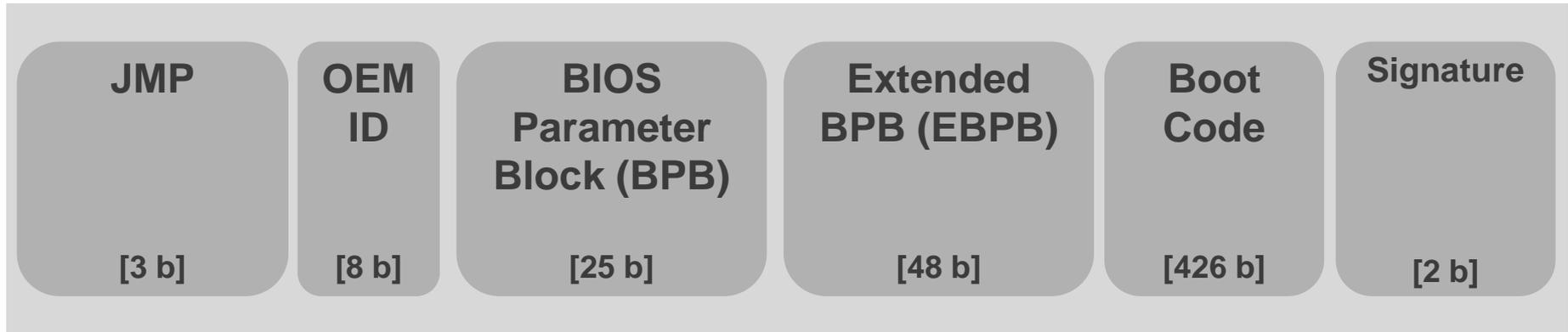


Win64/Rovnix: Bootkit Overview



NTFS Bootstrap Code

NTFS Boot Sector (Volume Boot Record)



NTFS Bootstrap Code

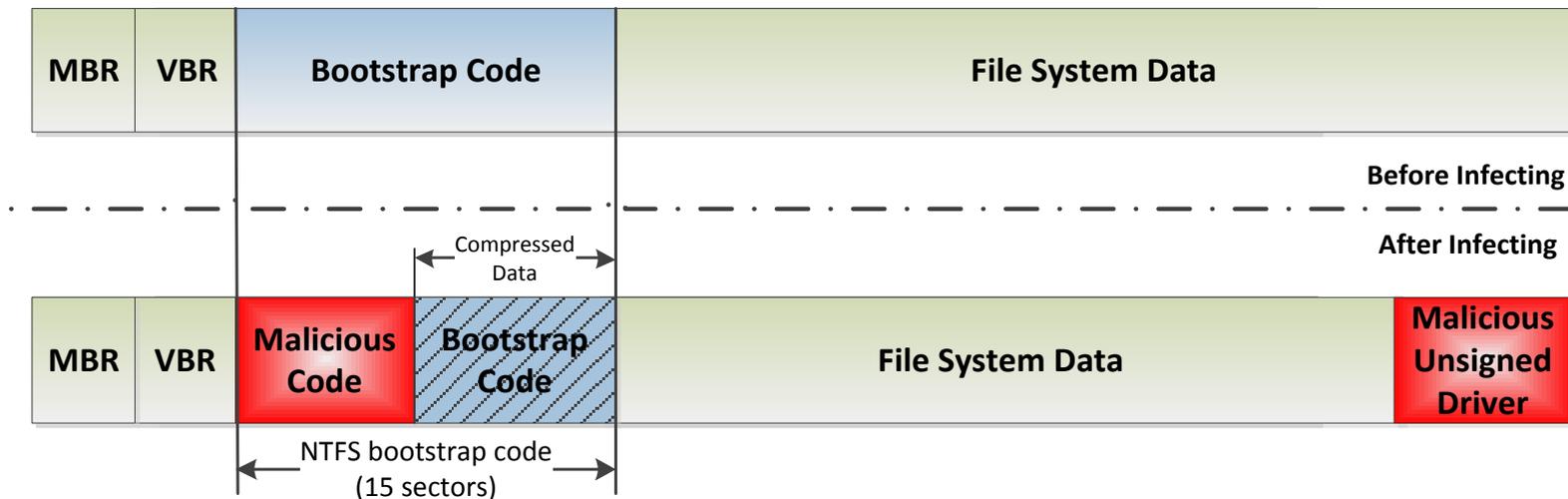
```
0000h: EB 52 90 4E 54 46 53 20 20 20 20 00 02 08 00 00 eR.NTFS .....
0010h: 00 00 00 00 00 F8 00 00 3F 00 FF 00 3F 00 00 00 .....ø.?.ÿ.?...
0020h: 00 00 00 00 80 00 80 00 D8 A6 3F 01 00 00 00 00 ...e.€.ø!?......
0030h: 00 00 0C 00 00 00 00 00 6D FA 13 00 00 00 00 00 .....mü.....
0040h: F6 00 00 00 01 00 00 00 29 1A 63 28 2F 63 28 CA ö.....).c(/c(È
0050h: 00 00 00 00 FA 33 C0 8E D0 BC 00 7C FB B8 C0 07 ...ú3ÄžBw.¡ù.À.
0060h: 8E D8 E8 16 00 B8 00 0D 8E C0 33 DB C6 06 0E 00 ž0è...žÄsÓz...
0070h: 10 E8 53 00 68 00 0D 68 6A 02 CB 8A 16 24 00 B4 .ès.h..hj.ÈŠ.$.'
0080h: 08 CD 13 73 05 B9 FF FF 8A F1 66 0F B6 C6 40 66 .i.s.*ÿÿšnf.ŵ@f
0090h: 0F B6 D1 80 E2 3F F7 E2 86 CD C0 ED (06) 41 66 0F .ŷÑeá?+áfÁí.ĥf.
00A0h: B7 C9 66 F7 E1 66 A3 20 00 C3 B4 41 BB AA 55 8A .Ěf+áfĚ.Ā.A»UŠ
00B0h: 16 24 00 CD 13 72 0F 81 FB 55 AA 75 09 F6 C1 01 .$.í.r..ŝU*u.šĀ.
00C0h: 74 04 FE 06 14 00 C3 66 60 1E 06 66 A1 10 00 66 t.p...Āf`..fj...f
00D0h: 03 06 1C 00 66 3B 06 20 00 0F 82 3A 00 1E 66 6A ....f. ....:..fj
00E0h: 00 66 50 06 53 66 68 10 00 01 00 80 3E 14 00 00 .fP.Sfh....€>...
00F0h: 0F 85 0C 00 E8 B3 FF 80 3E 14 00 00 0F 84 61 00 ....è*ÿe>.....a.
0100h: B4 42 8A 16 24 00 16 1F 8B F4 CD 13 66 58 5B 07 'BŠ.$...<ší.fX[.
0110h: 66 58 66 58 1F EB 2D 66 33 D2 66 0F B7 0E 18 00 fXfX.è-fš0f....
0120h: 66 F7 F1 FE C2 8A CA 66 8B D0 66 C1 EA 10 F7 36 f+ĥpĀŠĚf<ĥfĀè.+6
0130h: 1A 00 86 D6 8A 16 24 00 8A E8 C0 E4 06 0A CC B8 .+ŝŠ.$.ŠèĀĀ..Ī.
0140h: 01 02 CD 13 0F 82 19 00 8C C0 05 20 00 8E C0 66 ..í...ġĀ. .žĀf
0150h: FF 06 10 00 FF 0E 0E 00 0F 85 6F FF 07 1F 66 61 ŷ...ÿ...ŝoÿ..fa
0160h: C3 A0 F8 01 E8 09 00 A0 FB 01 E8 03 00 FB EB FE Ā ø.è.. ŷ.è..ŷĕp
0170h: B4 01 8B F0 AC 3C 00 74 09 B4 0E BB 07 00 CD 10 ';<š-<t.'...Ī.
0180h: EB F2 C3 0D 0A 41 20 64 69 73 6B 20 72 65 61 64 èšĀ..A disk read
0190h: 20 65 72 72 6F 72 20 6F 63 63 75 72 72 65 64 00 error occurred.
01A0h: 0D 0A 4E 54 4C 44 52 20 69 73 20 6D 69 73 73 69 ..NTLDR is missi
01B0h: 6E 67 00 0D 0A 4E 54 4C 44 52 20 69 73 20 63 6F ng...NTLDR is co
01C0h: 6D 70 72 65 73 73 65 64 00 0D 0A 50 72 65 73 73 mpressed...Press
01D0h: 20 43 74 72 6C 2B 41 6C 74 2B 44 65 6C 20 74 6F Ctrl+Alt+Del to
01E0h: 20 72 65 73 74 61 72 74 0D 0A 00 00 00 00 00 00 restart.....
01F0h: 00 00 00 00 00 00 00 00 83 A0 B3 C9 00 00 55 AA .....f *È..U+
0200h: 05 00 4E 00 54 00 4C 00 44 00 52 00 04 00 24 00 ..N.T.L.D.R...$.
0210h: 49 00 33 00 30 00 00 E0 00 00 00 30 00 00 00 00 I.3.0..à...0....
0220h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0230h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0240h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0250h: 00 00 00 00 00 00 EB 12 90 90 00 00 00 00 00 .....Ē.....
0260h: 00 00 00 00 00 00 00 00 00 00 0E E8 00 00 58 2D .....è...X-
0270h: 04 00 50 1E 06 60 BF 13 00 6A 40 1F 8B 0D 49 49 ..P..ž..j@.<.II
0280h: 89 0D C1 E1 06 0E 1F 8B DC BD FF 05 50 51 05 4D %.Āá...<Ůšÿ.PQ.M
0290h: 00 8B F0 03 E8 33 FF 07 FC B9 D1 07 F3 A4 5F 06 .<š.èÿÿ.u*Ň.šġ..
02A0h: 1E 07 B1 18 FF D5 02 B9 FF 05 2B C5 03 C1 BE D2 ..±.ÿŌ..ÿ.+.Ā.ĀšŌ
02B0h: 01 50 51 03 F5 5D CB B1 05 FF D5 02 16 07 96 33 .PQ.š]Ě±.ÿŌ...-3
02C0h: C9 83 EC 08 8B FC 66 A5 66 A5 AD BF 13 00 8B D0 Ěfi.<úfÿÿÿ-ž...<Ě
02D0h: 6A 40 1F 8B 0D D1 E8 40 2B C8 89 0D C1 E1 06 51 jŝ.<.Ňè@+Ě%.ĀĀ.C
02E0h: 6A 00 52 6A 10 0E 1F 89 4C F6 16 1F 8B F4 B2 80 j.Rj...%Lš...<š*È
02F0h: B4 42 CD 13 83 C4 10 C3 53 B2 80 66 33 DB B9 00 .Bf.rĀ.Āššfĥĥ±:
```

```
typedef struct BOOTSTRAP_CODE {
    BYTE code_1[8] <fgcolor=cGreen>;
    BYTE IPL[32] <fgcolor=cAqua>;
    BYTE IPL_EXT[7] <fgcolor=cYellow>;
    BYTE code_2[387] <fgcolor=cGreen>;
    BYTE endOfSector[2] <fgcolor=cWhite>;
};
```

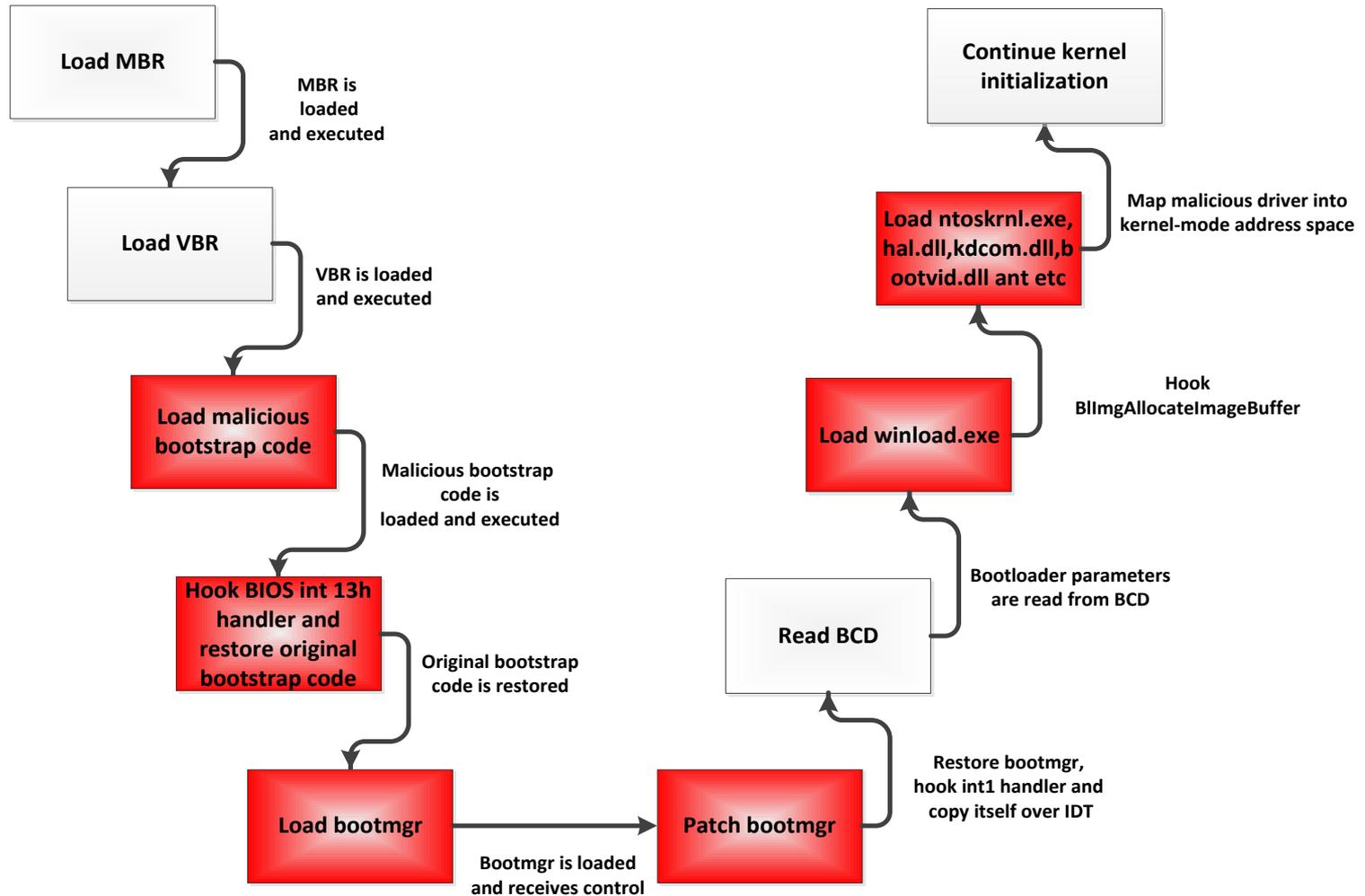
```
typedef struct _BOOTSECTOR {
    BYTE jmp[3] <fgcolor=cGreen>;
    BYTE oemID[8] <fgcolor=cWhite>;
    BIOS_PARAMETR_BLOCK bpb <fgcolor=cYellow>;
    EXT_BIOS_PARAMETR_BLOCK ebpb <fgcolor=cPurple>;
    BOOTSTRAP_CODE bootstrap;
    NTLDR_CODE ntldr;
};
```

Win64/Rovnix: Infected Partition Layout

- Win64/Rovnix overwrites bootstrap code of the active partition
- The malicious driver is written either:
 - ✓ before active partition, in case there is enough space
 - ✓ in the end of the hard drive, otherwise

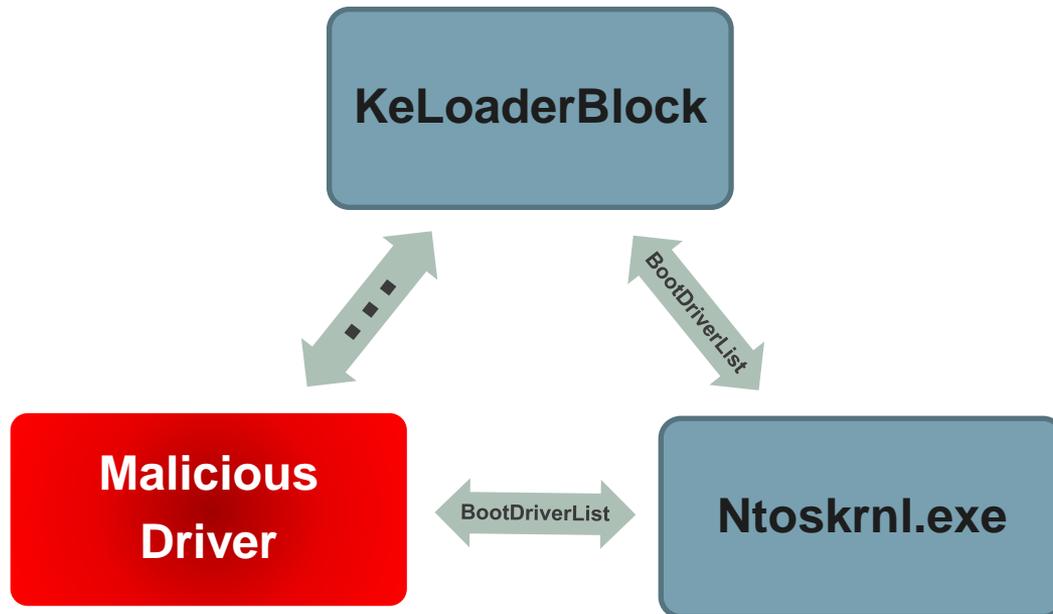


Win64/Rovnix: Bootkit Details



Win64/Rovnix: Loading Unsigned Driver

- Insert malicious driver in *BootDriverList* of *KeLoaderBlock* structure
- When kernel receives control it calls entry point of each module in the *BootDriverList*



Win64/Rovnix: Abusing Debugging Facilities

Win64/Rovnix:

- **hooks Int 1h**
 - ✓ tracing
 - ✓ handles hardware breakpoints (DR0-DR7)
- **overwrites the last half of IDT (*Interrupt Descriptor Table*)**
 - ✓ is not used by OS

As a result the malware is able to:

- ✓ set up hooks without patching bootloader components
- ✓ retain control after switching into protected mode

Win64/Rovnix: Abusing Debugging Facilities

Win64/Rovnix:

- hooks Int 1h

 - ✓ tracing

 - ✓ handles hardware

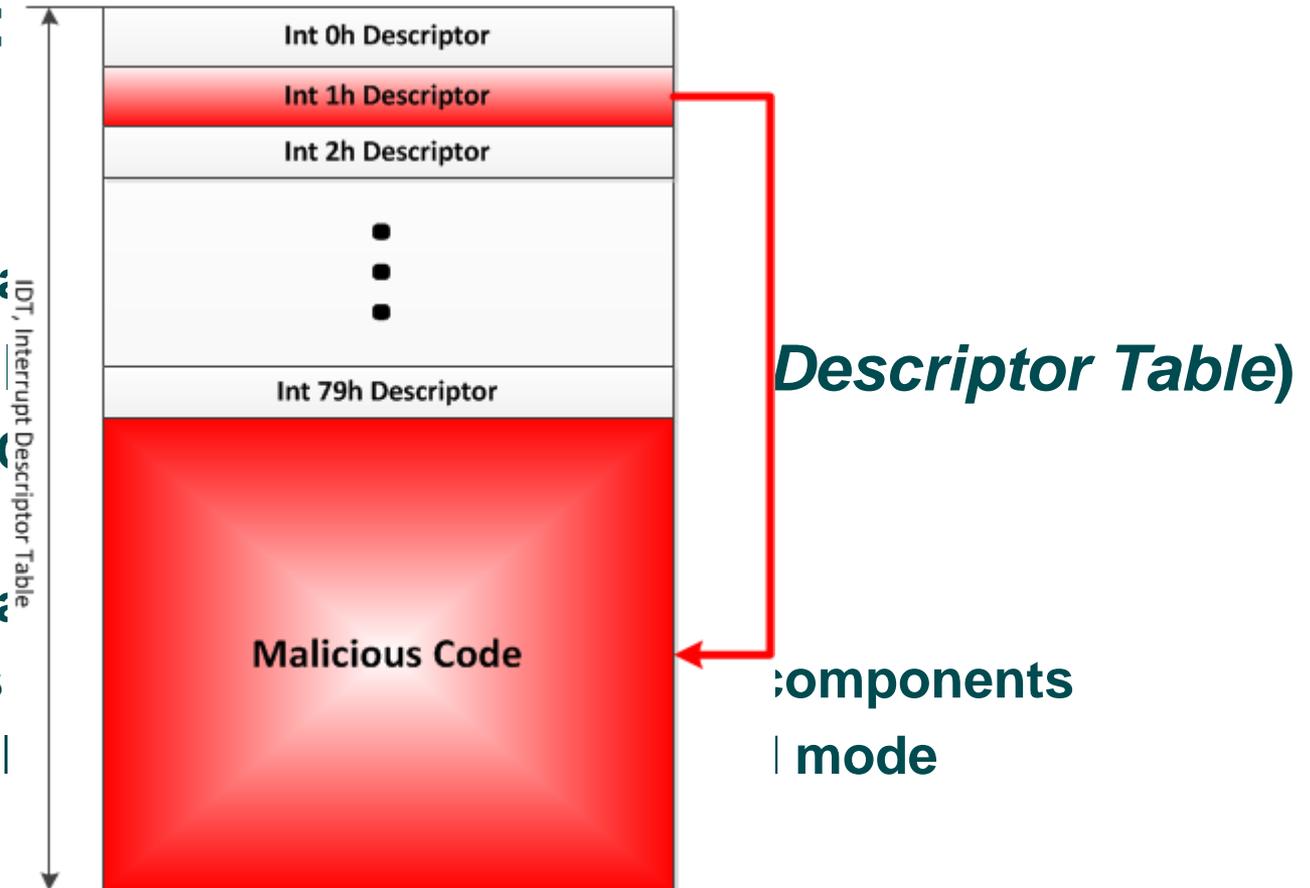
- overwrites the

 - ✓ is not used by (

As a result the malware

 - ✓ set up hooks

 - ✓ retain control



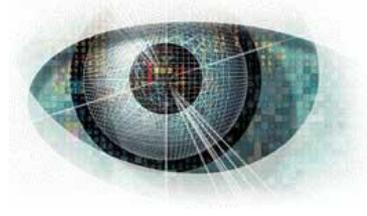
DEMO



Olmarik vs Rovnix

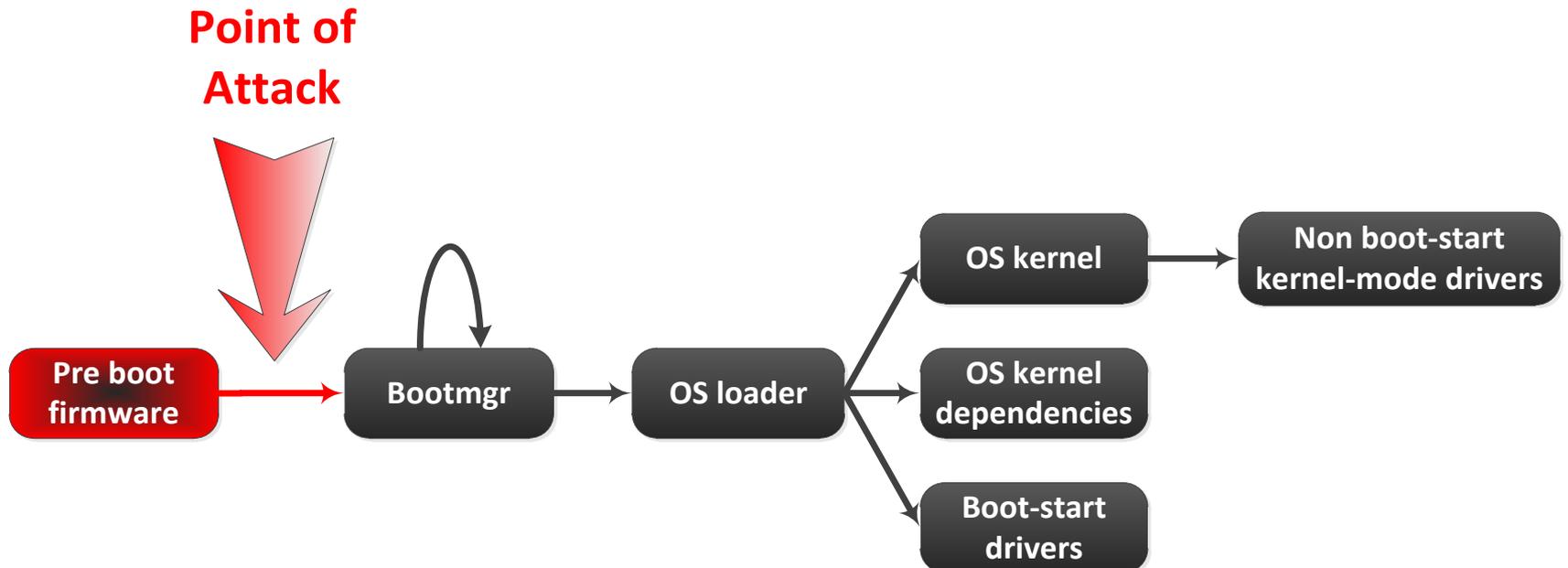
Characteristics	Win64/Olmarik	Win64/Rovnix
Privilege escalation	MS10-092	<input checked="" type="checkbox"/>
Reboot technique	<i>ZwRaiseHardError</i> API	<i>ExitWindowsEx</i> API
MBR/VBR infection	MBR	VBR (bootstrap code)
Loading driver	<i>ZwCreateDriver</i> API	Inserting into boot driver list of <i>KeLoaderBlock</i> structure
Payload injection	<i>KeInitializeApc/KeInstertQueueApc</i> APIs	<i>KeInitializeApc/KeInstertQueueApc</i> APIs
Self-defense	Kernel-mode hooks, MBR monitoring	<input checked="" type="checkbox"/>
Number of modules	10	2
Stability of code	■ ■ ■ ■ ■	■ ■ ■ ■ ■
Threat complexity	■ ■ ■ ■ ■	■ ■ ■ ■ ■

What Facilitates the Attack Vector?



- **Untrusted platform problem**

- ✓ BIOS controls boot process, but who controls it?
- ✓ The trust anchor is below point of attack





HiddenFsReader as a Forensic Tool

HiddenFsReader as a Forensic Tool

Retrieves content of the malware hidden file system.

Supported malware: TDL3/TDL3+,TDL4;

ZeroAccess (will be added soon)



```
C:\>TdlFsReader.exe
Contents of IDL file system:
  cfg.ini MD5: B8C8B1B5C01EBF2F48760F2E06C402E6
  mbr MD5: AF1EC9B9C5CE1D74D3D9CA3BBE0FA941
  bckfg.tmp MD5: 6AD76461EEB59A1D77529B595D3672ED
  cmd.dll MD5: 4DADED6C7EFF7230D68AE48B02A847FA
  ldr16 MD5: 4FB9748189F6688ADA9A51A5901406FA
  ldr32 MD5: C078E5EA19F853AC0830D2F6088F7161
  ldr64 MD5: ADEB890D564913F11301C648A5FB6220
  drv64 MD5: 87A462D034192EDD60ACE835E91B930B
  cmd64.dll MD5: BC3B9FB8EAFD440D43B76DC12FB445C7
  drv32 MD5: 1EF0E0C765DA7F727E1FB8FF38D02FF1
```

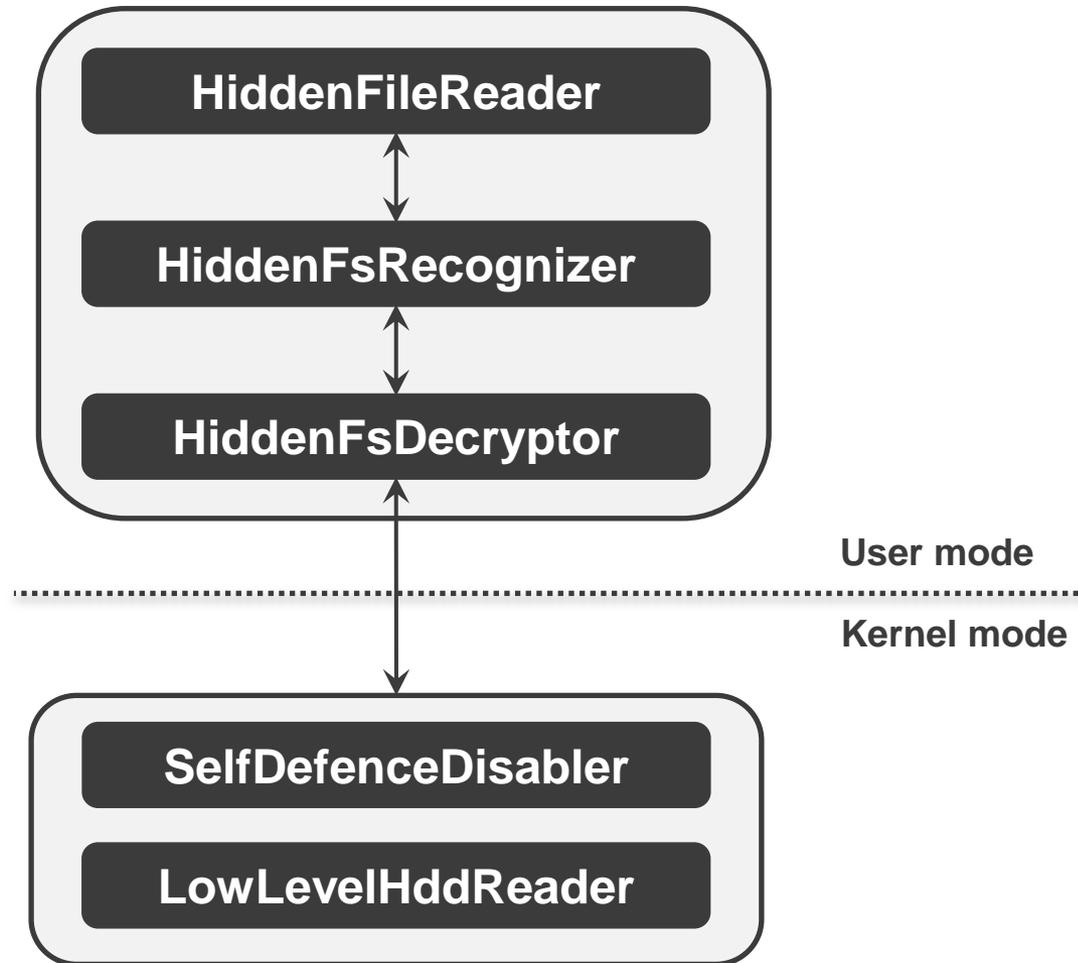
<http://eset.ru/tools/TdlFsReader.exe>

DEMO

<http://www.youtube.com/watch?v=iRpp6vn2DAE>



HiddenFsReader (HFR) Architecture



Conclusion

- ✓ **The bootkit technique allows malware to bypass KMCSPP**
- ✓ **Return to old-school techniques of infecting MBR**
- ✓ **Win64/Olmarik (TDL4) is the first widely spread rootkit targeting Win x64**
- ✓ **Win64/Rovnix relies on debugging facilities of the platform to subvert KMCSPP**
- ✓ **The only possible way of debugging bootkits is to use emulators (Bochs, QEMU)**
- ✓ **The untrusted platform facilitates bootkit techniques**
- ✓ **HiddenFsReader is shared amongst malware researchers**

References

✓ **“The Evolution of TDL: Conquering x64”**

http://www.eset.com/us/resources/white-papers/The_Evolution_of_TDL.pdf

✓ **“Defeating x64: The Evolution of the TDL Rootkit”**

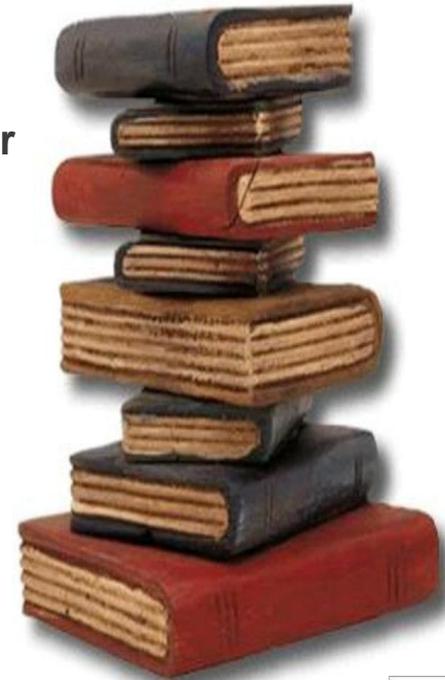
<http://www.eset.com/us/resources/white-papers/TDL4-CONFidence-2011.pdf>

✓ **“Hasta La Vista, Bootkit: Exploiting the VBR”**

<http://blog.eset.com/2011/08/23/hasta-la-vista-bootkit-exploiting-the-vbr>

✓ **Follow ESET Threat Blog**

<http://blog.eset.com>



Questions



Thank you for your attention ;)

Aleksandr Matrosov

matrosov@eset.sk

@matrosov

Eugene Rodionov

rodionov@eset.sk

@vxradius

