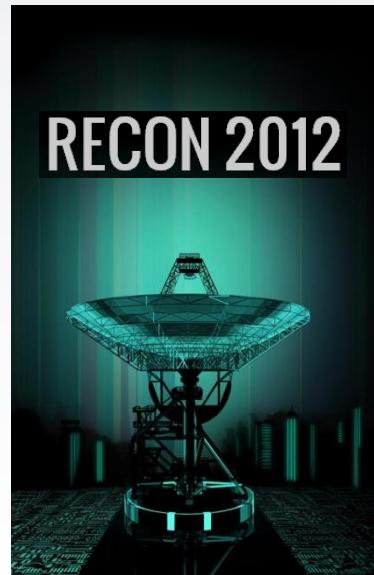


Bootkit Threats: In Depth Reverse Engineering & Defense

**Eugene Rodionov
Aleksandr Matrosov**



Outline of The Presentation

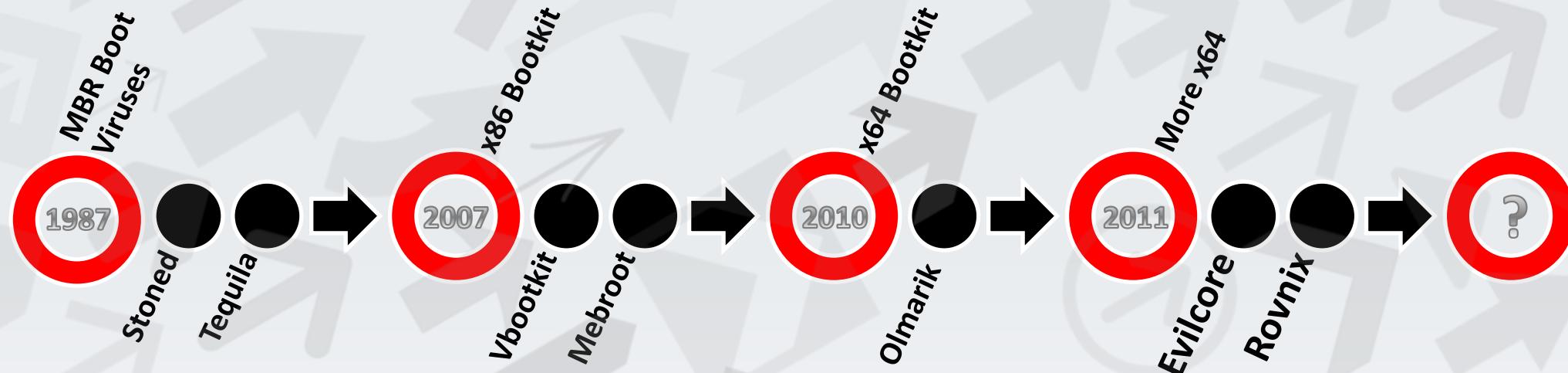
- ✓ Bootkit technology
 - ✓ Why? How?
- ✓ Bootkit design principles
 - ✓ Architecture
 - ✓ Analysis instrumentation
- ✓ Rovnix bootkit in-depth analysis
 - ✓ Infected VBR analysis
 - ✓ Infection strategy
- ✓ Bootkit remediation techniques



Bootkit technology



Bootkit evolution over time



- Bootkit PoC evolution:

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

- Bootkit Threats evolution:

- ✓ Win32/Mebroot (2007)
- ✓ Win32/Mebratix (2008)
- ✓ Win32/Mebroot v2 (2009)
- ✓ Win64/Olmarik (2010/11)
- ✓ Win64/Olmasco (2011)
- ✓ Win64/Rovnix (2011/2012)

Why?

Why there is a return to bootkit technology nowadays

- ✓ Microsoft kernel-mode code signing policy
 - loading unsigned kernel-mode driver
- ✓ High level of stealth
 - there are no malicious files in the file system
- ✓ High degree of survival
 - difficult to detect and remove
- ✓ Ability to disable security software
 - the malware is launched before security software

How?

Bootkits in the wild:

- Infecting:
 - ✓ MBR (Master Boot Record)
 - ✓ VBR (Volume Boot Record)

Proof of Concept Bootkits:

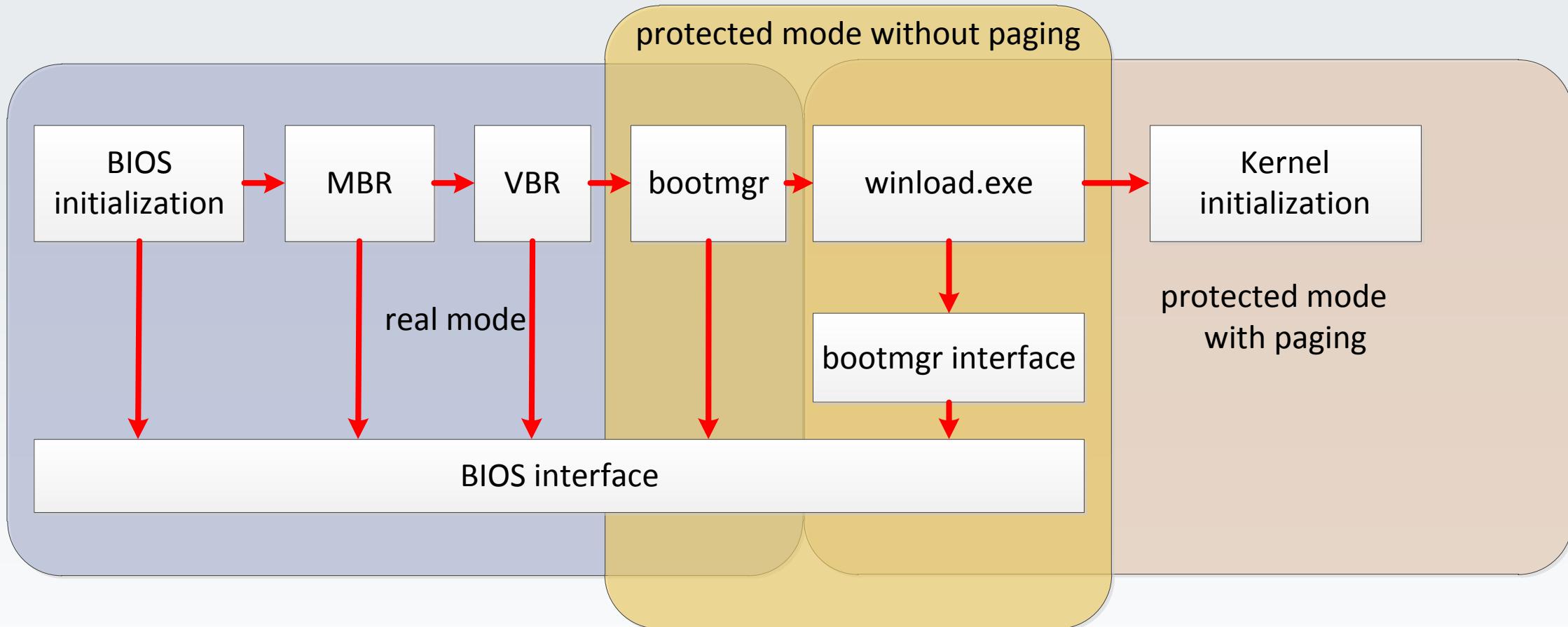
- Infecting UEFI

Bootkit design principles

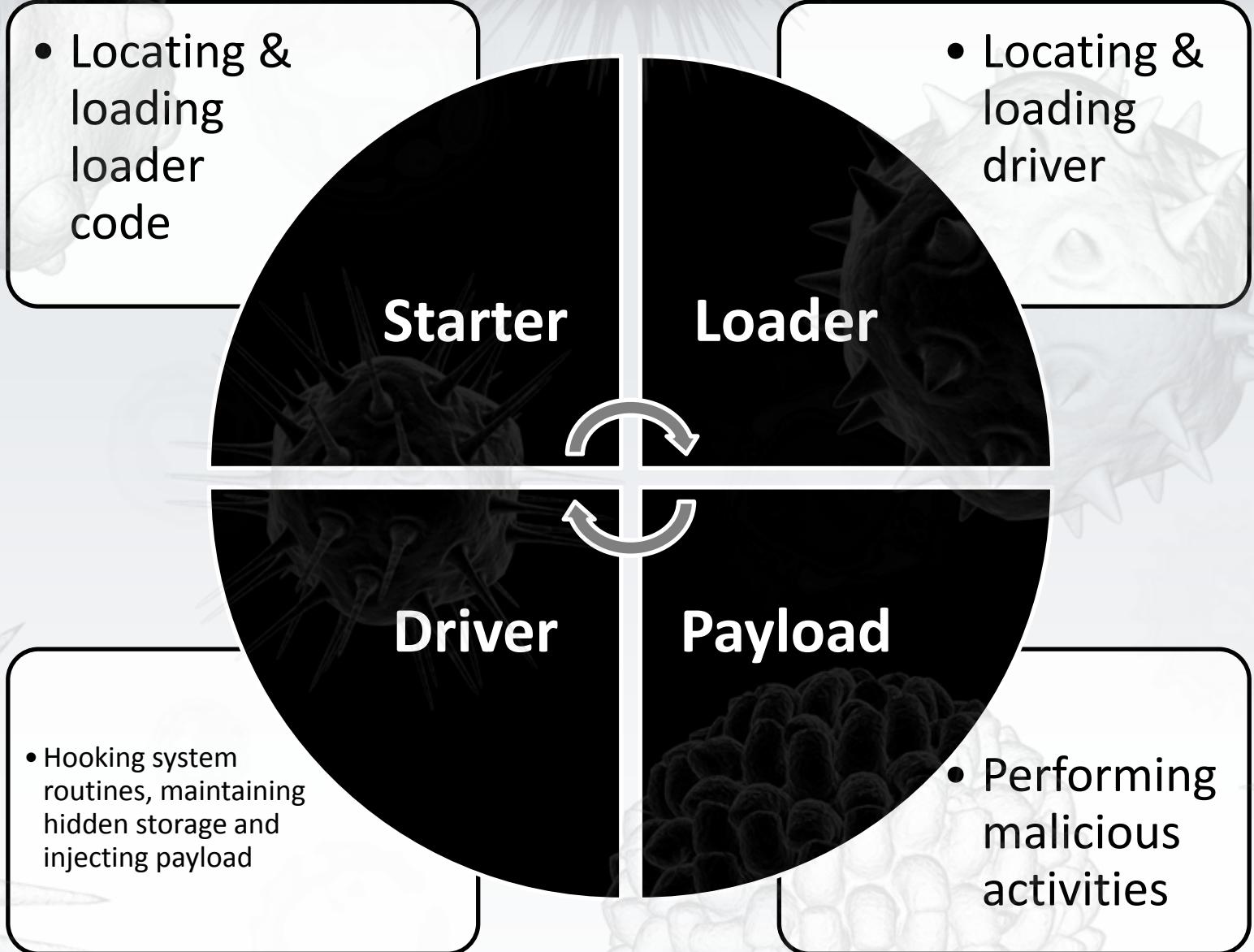


Boot process

Description of OS boot process:



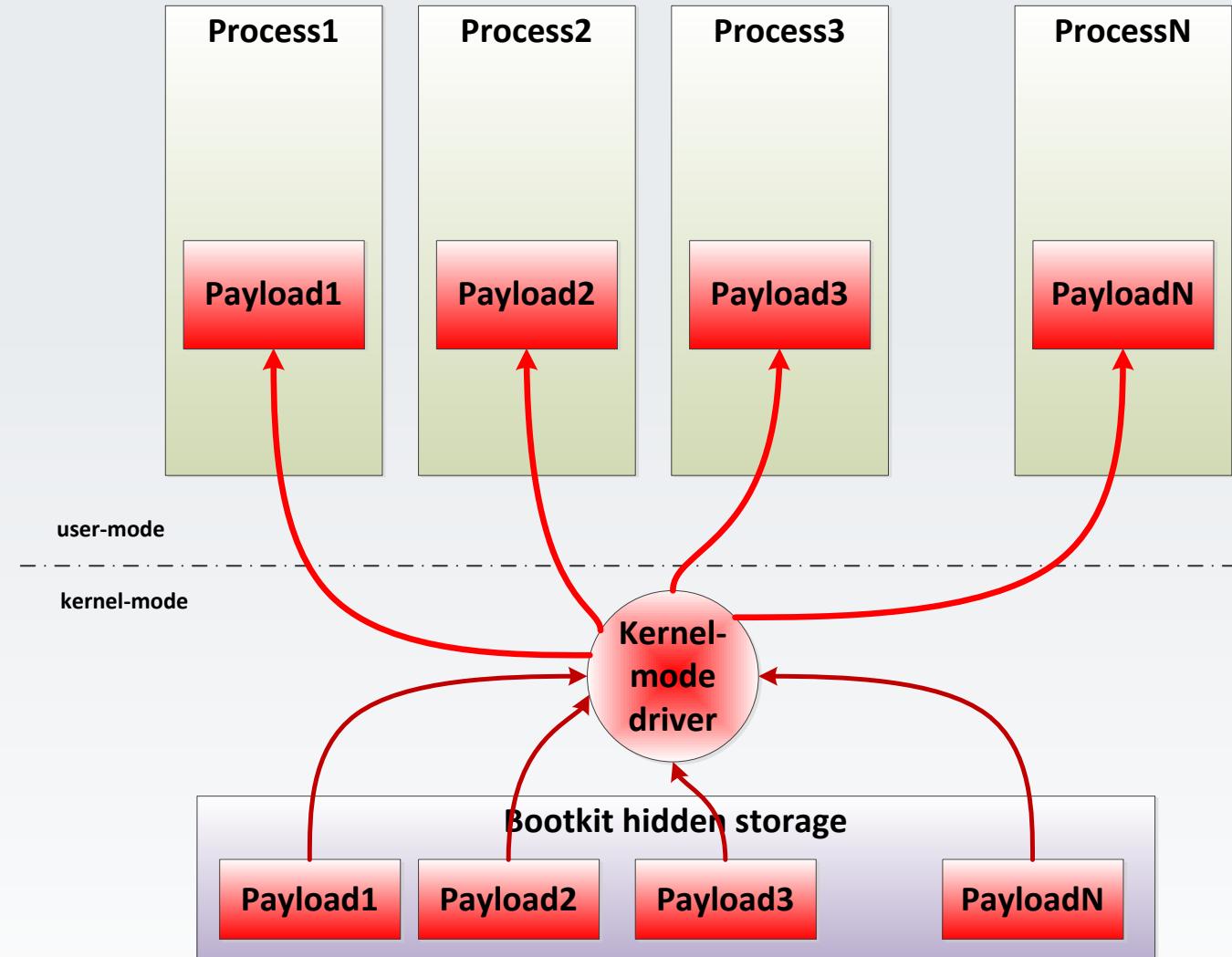
Bootkit Architecture



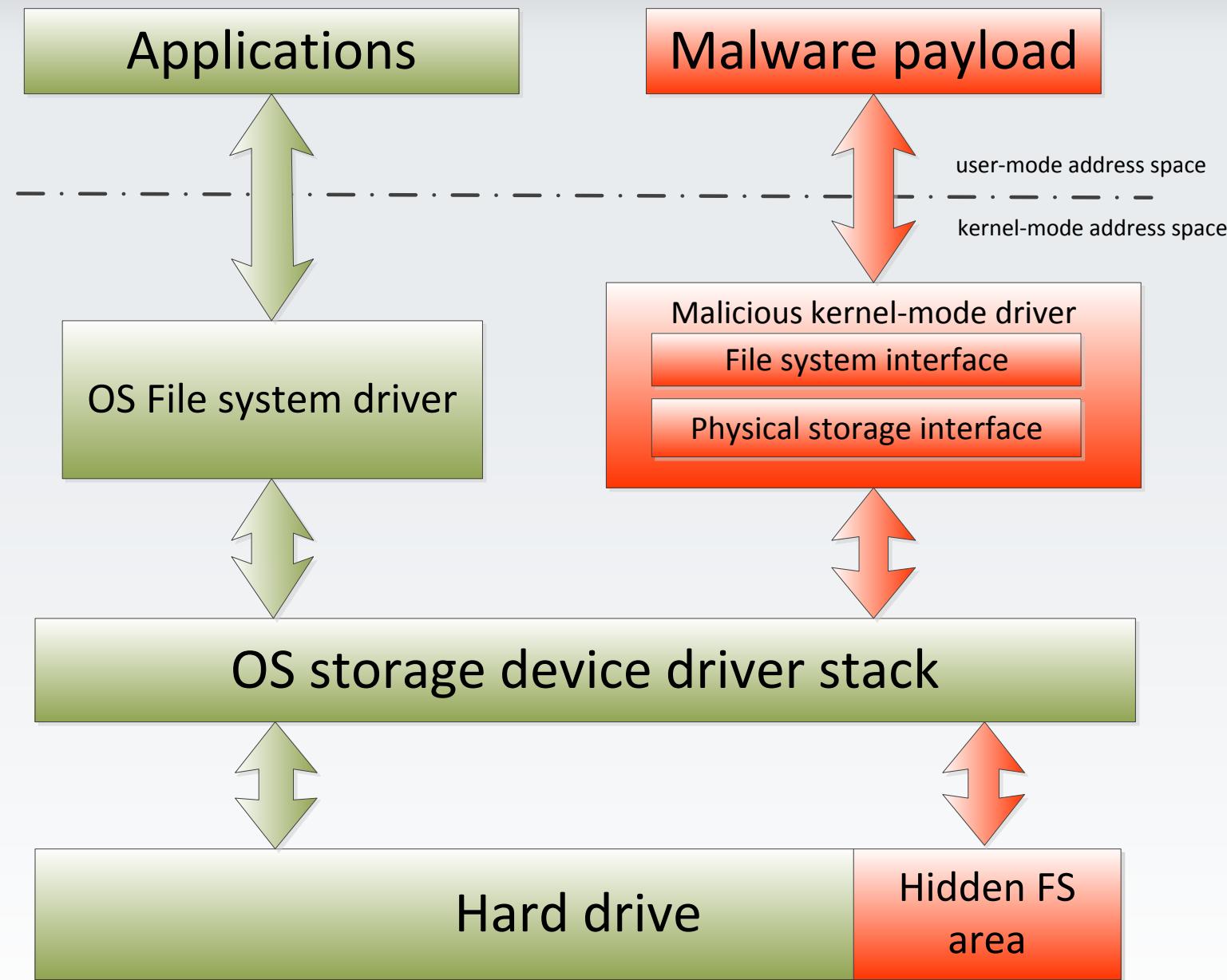
Injecting Payload

Injection approach

- ✓ APC routines
- ✓ Patching entry point of the executable



Hidden Storage Architecture



Bootkit Analysis Instrumentation



Debugging bootkit with Bochs

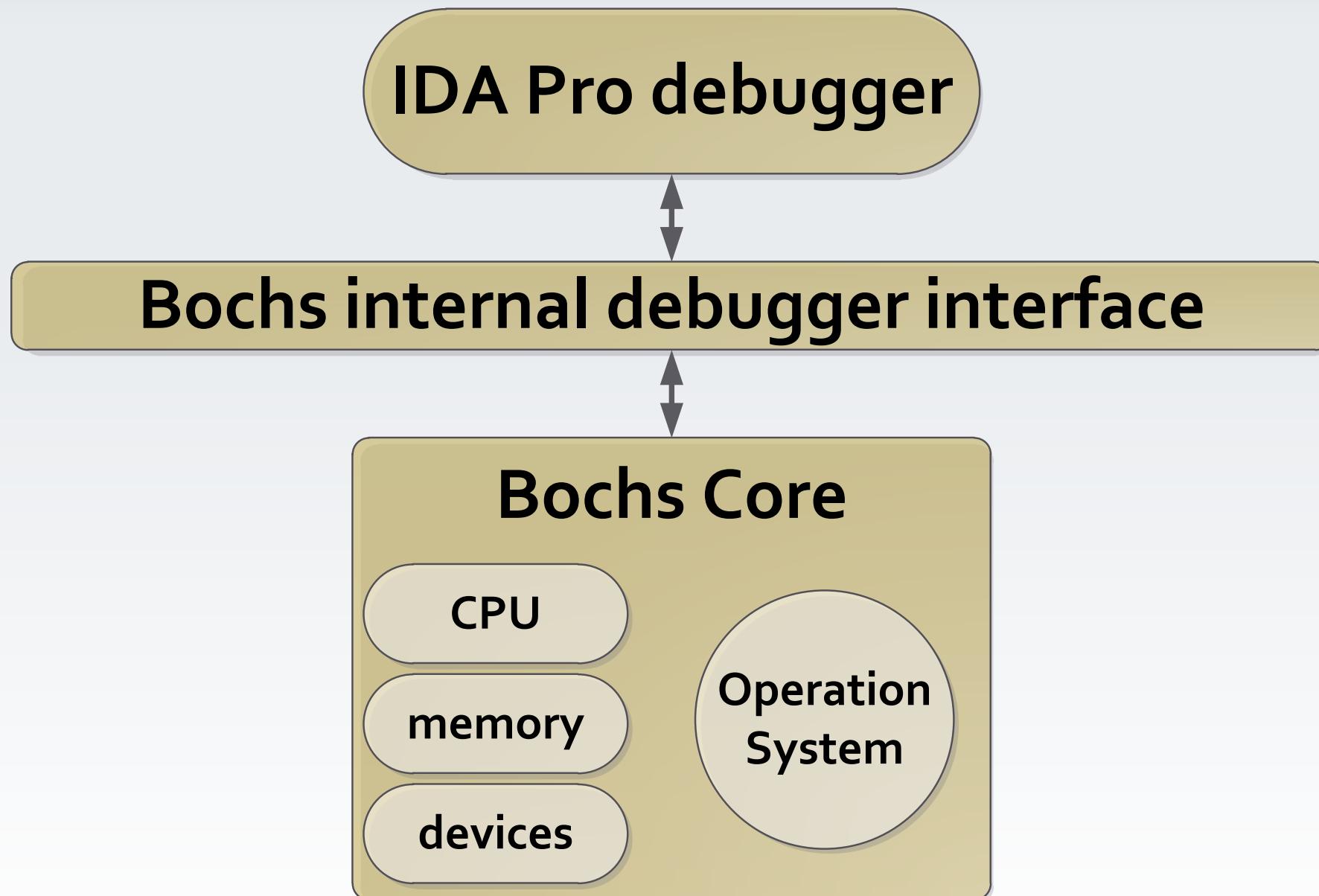
/configure --enable-debugger

```
Bochs for Windows - Console

=====
Bochs x86 Emulator 2.5.1
Built from SUN snapshot on January 6, 2012
Compiled on Jun  8 2012 at 14:28:34
=====

000000000000i[  ] Screen mode changed to
000000000000i[  ] reading configuration from D:\images\Win7EnterpriseSp1x64\bochsrc251.bxrc
000000000000e[  ] D:\images\Win7EnterpriseSp1x64\bochsrc251.bxrc:26: 'i440fxsupport' will be replaced by new 'pci' option.
000000000000e[  ] D:\images\Win7EnterpriseSp1x64\bochsrc251.bxrc:27: 'vga_update_interval' will be replaced by new 'vga: update_freq' option.
000000000000i[  ] Ignoring magic break points
Next at t=0
<0> [0x00000000fffffff0] f000:fff0 (unk. ctxt): jmp far f000:e05b      ; ea5be000f0
<bochs:1> help
h:help - show list of debugger commands
h:help command - show short command description
-- Debugger control --
  help, q|quit|exit, set, instrument, show, trace, trace-reg,
  trace-mem, u|disasm, ldsym, slist
-- Execution control --
  c|cont|continue, s|step, p|next, modebp, v|exitbp
-- Breakpoint management --
  vb|vbreak, lb|lbreak, pb|pbreak|b|break, sb, sha, blist,
  bpe, bpd, d|del|delete, watch, unwatch
-- CPU and memory contents --
  x, xp, setpmem, crc, info,
  r|reg|regs|registers, fp|fpu, mmx, sse, sreg, dreg, creg,
  page, set, ptime, print-stack, ?|calc
-- Working with bochs param tree --
  show "param", restore
<bochs:2>
```

Debugging bootkit with Bochs



LIVE DEMO



Rovnix Reverse Engineering



Ring0 bundle (Zerokit) for control million-strong botnet

Goto page 1, 2, 3, 4 Next

[Post Reply](#)

[darkode.com Forum Index](#) » [Projects](#)

[View previous topic](#)
[View next topic](#)

Ring0 bundle (Zerokit) for control million-strong botnet

Author	Message	QUOTE
ring0 Joined: 21 May 2011 Posts: 12 Rep: 1752	<p>Ring0 bundle (Zerokit) for control million-strong botnet</p> <p>I want to introduce new crazy ring0 bundle (Zerokit or 0kit) for control million-strong botnet.</p> <p>Breaking down all nowadays-existing firewall with full network blocking (bypassing in ring0).</p> <p>Existence of the bundle is not detected by any of the antivirus (the list http://www.matousec.com/projects/proactive-security-challenge/results.php), antirootkit-utilities (Tuluka, GMER, RKU, RootkitRevealer) also see nothing.</p>	

Features:

- Start of *.exe, *.dll (*.dll is in a pre-alpha stage) and shellcodes in a context of the chosen process.
- Start of files from a disk and from the memory* (start from memory is in a pre-alpha stage).
- Start of files with specified privileges: CurrentUser and NT SYSTEM/AUTHORITY.
- Granting the protected storehouse** for off-site (your) ring3-solutions for permanent existence in the system without need of crypt.
- Survivability of the bundle, down to a reinstallation of the system.
- All the components are stored outside of a file system and are invisible to OS.
- Intuitively clear interface of admin-panel.
- Protection against the abstraction of Admin Panel.
- Impossibility of detection of the bundle in the working system by any of known AV/rootkit scanner, owing to the use of author's technologies of concealment. The unique opportunity of detection exists only at loading with livecd or scanning of a disk from the other computer. Thus the opportunity of detection is also extremely improbable, as own algorithms of a mutation are used.

* Start of a file from the memory allows to bypass all modern proactive protection and AV-scanners, that is, there is no necessity to crypt a file.

** Protected storehouse is the original ciphered file system in which the certain quantity of files which will be started from the memory at each start of the OS can be stored.

The bundle consists of:

- **Bootkit.** It is responsible for the start of the basic modules at a stage of loading of OS.
- **Driver.** It is responsible for all infrastructure and implements componental business-logic on the basis of so-called mod (functional unit). That is, the driver is not a legacy driver (monolithic), and consists of the set of mods that allows to operate the bundle with maximum of flexibility, and to protect (hard to reverse), update and expand it.
- **Dropper.** At the current moment it breaks out all machines with the patches till January, 8th, 2011, except for XP x32/x64 where reloading is initiated. If the systems distinct from XP have latest updates reloading is initiated as well.
- User friendly Admin Panel.

Ring0 bundle (Zerokit) for control million-strong botnet

[Goto page 1, 2, 3, 4](#) [Next](#)

[Post Reply](#)

[darkode.com](#) Forum Index » Projects



Ring0 bundle (Zerokit) for control million-strong botnet

Goto page 1, 2, 3, 4 Next

[Post Reply](#)

[darkode.com](#) Forum Index » Projects

Author	Message
ring0	<p>Ring0 bundle (Zerokit) for control million-strong botnet</p> <p>I want to introduce new crazy ring0 bundle (Zerokit or Okit) for control million-strong botnet.</p>

Joined: 21 May 2011
Posts: 12
Rep: 1752

```
int      3
6 inc    edx
dec      ebx
sub    eax,044414F4C ;' DAOI'
inc    ebp
push   edx
```

otkit-utilities (Tuluka)

Field Name	Data Value	Description
Machine	014Ch	i386®
Number of Sections	0004h	
Time Date Stamp	4D5561A2h	11/02/2011 16:19:46
Pointer to Symbol Table	00000000h	
Number of Symbols	00000000h	
Size of Optional Header	00E0h	
Characteristics	0103h	
Magic	010Bh	PE32
Linker Version	0008h	8.0

```
movsd  
movsw  
pop edi  
pop esi  
pop ebp  
pushad
```

- **Bootkit.** It is responsible for the start of the basic modules at a stage of loading of OS.
- **Driver.** It is responsible for all infrastructure and implements componental business-logic on the basis of so-called mod (functional unit). That is, the driver is not a legacy driver (monolithic), and consists of the set of mods that allows to operate the bundle with maximum of flexibility, and to protect (hard to reverse), update and expand it.
- **Dropper.** At the current moment it brake out all machines with the patches till January, 8th, 2011, except for XP x32/x64 where reloading is initiated. If the systems distinct from XP have latest updates reloading is initiated as well.



Interesting Carberp sample (October 2011)

```
_IsWow64Process@4*
UBR*
\PHYSICALDRIVE0*
\PHYSICALDRIVE0*
BKSETUP: Payload of %u bytes successfully written at sector %x.
\Device\Hdddisk0\Partition%u*
\Device\Hdddisk0\Partition%u*
NTFS *
BKSETUP_>04x: BK setup dll version 2.1.
BKSETUP_>04x: Attached to a 32-bit process at 0x%.
BKSETUP_>04x: Detached from a 32-bit process.
<%08X->%04X->%04X->%04X->%08X>%04X>*
IsWow64Process*
KERNEL32.DLL*
open*
%lu.bat*
"%s"*
attrib -r -s -h%1
:klabel
del %1
if exist %1 goto klabel
del %0
Software\Classes\CLSID\*
runas*
BKSETUP: Failed generating program key name.
BKSETUP: Already installed.
BKSETUP: OS not supported.
BKSETUP: Not enough privileges to complete installation.
BKSETUP: No joined payload found.
BKSETUP: Installation failed because of unknown reason.
BKSETUP: Successfully installed.
BKSETUP: Version: 1.0
BKSETUP: Started as win32 process 0x%.
BKSETUP: Process 0x% finished with status %u.
BKSETUP: Version: 1.0
BKSETUP: Started as win32 process 0x%.
BKSETUP: Process 0x% finished with status %u
```



Interesting Carberp sample (October 2011)

Total bots: 2831

Sort
Status
Step
Alias
Other
Del

	ID	step	info	status	data
	TEST_BK_KIT_EXPLORER0D9493DFECAE8C4B0	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_KIT_EXPLORER08D7BD1230A905D00	6	BkInstall	FALSE	0000-00-00 00:00:00
	123213oob	1	infa	false	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV0F1B889AC4F21B5CA	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV09A01A1B010A8035A	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV0743BC19E94740	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV0DA631E2FA5B562AF	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV079943F8A64F9587B	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV09A01A1B010A8035A	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_MY_DRV07AA547C0940C1901	3	BkInstall0 GetLastError = 0	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_ORIG_DRV0B61FDB428F96A87B	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_ORIG_DRV0AE10F7A3602E42CB	6	BkInstall	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_ORIG_DRV06627C6A2AB3A2480	1	IsUserAdmin	FALSE	0000-00-00 00:00:00
	TEST_BK_EX_ORIG_DRV0623F20AD27008003	6	BkInstall	FALSE	0000-00-00 00:00:00

```
BKSETUP: Installation failed because of unknown reason.  
BKSETUP: Successfully installed.  
BKSETUP: Version: 1.0  
BKSETUP: Started as win32 process 0x2xx.  
BKSETUP: Process 0x2xx finished with status xu.  
BKSETUP: Version: 1.0  
BKSETUP: Started as win32 process 0x2xx  
BKSETUP: Process 0x2xx finished with status xu
```



Rovnix Kit Hidden File Systems Comparison

functionality	Rovnix.A	Carberp with bootkit	Rovnix.B
VBR modification	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
polymorphic VBR	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Malware driver storage	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Driver encryption algorithm	custom (ROR + XOR)	custom (ROR + XOR)	custom (ROR + XOR)
Hidden file system	<input checked="" type="checkbox"/>	FAT16 modification	FAT16 modification
File system encryption algorithm	<input checked="" type="checkbox"/>	RC6 modification	RC6 modification

Rovnix Architecture

Dropper

Infected VBR

Kernel-mode driver x86

Payload x86

Kernel-mode driver x64

Payload x64

Signature	Payload RVA	Decompressed size	Flags
00 00 00 00 00 00 00 00 2E 74 65 78 74 00 00 00			text..
76 29 00 00 00 10 00 00 00 2A 00 00 00 00 10 00 00			v) ..□...*...□..
00 00 00 00 00 00 00 00 00 00 00 20 00 00 60		
2E 72 64 61 74 61 00 00 5C 07 00 00 00 40 00 00			.rdata..\□..@..
00 08 00 00 00 40 00 00 00 00 00 00 00 00 00 00			.□...@..
00 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00		@..@.data..
08 00 00 00 00 00 00 00 00 00 00 50 00 00			□....P.....P..
00 00 00 00 00 00 00 00 00 00 00 40 00 00 C0		@..A..
2E 72 73 72 63 00 00 00 00 50 02 00 00 60 00 00			.rsrc...□..`..
00 46 02 00 00 60 00 00 00 00 00 00 00 00 00 00			.F□..`..
00 00 00 00 40 00 00 40 00 00 00 00 00 00 00 00		@..@..
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00		
Payload description block			
00 00 00 C0 00 00 00 00 00 00 00 00 00 00 00 00		
46 4A 10 00 00 64 00 00 52 09 00 00 00 21 00 00			EJ□..d..R....!
46 4A 10 00 00 6E 00 00 00 C4 00 00 03 28 00 00			EJ□..n...Д..□(..
46 4A 10 00 00 3C 01 00 00 52 01 00 0B 28 00 00			EJ□..<□..R□.□(..

Installation Into the System

Check administrative privileges

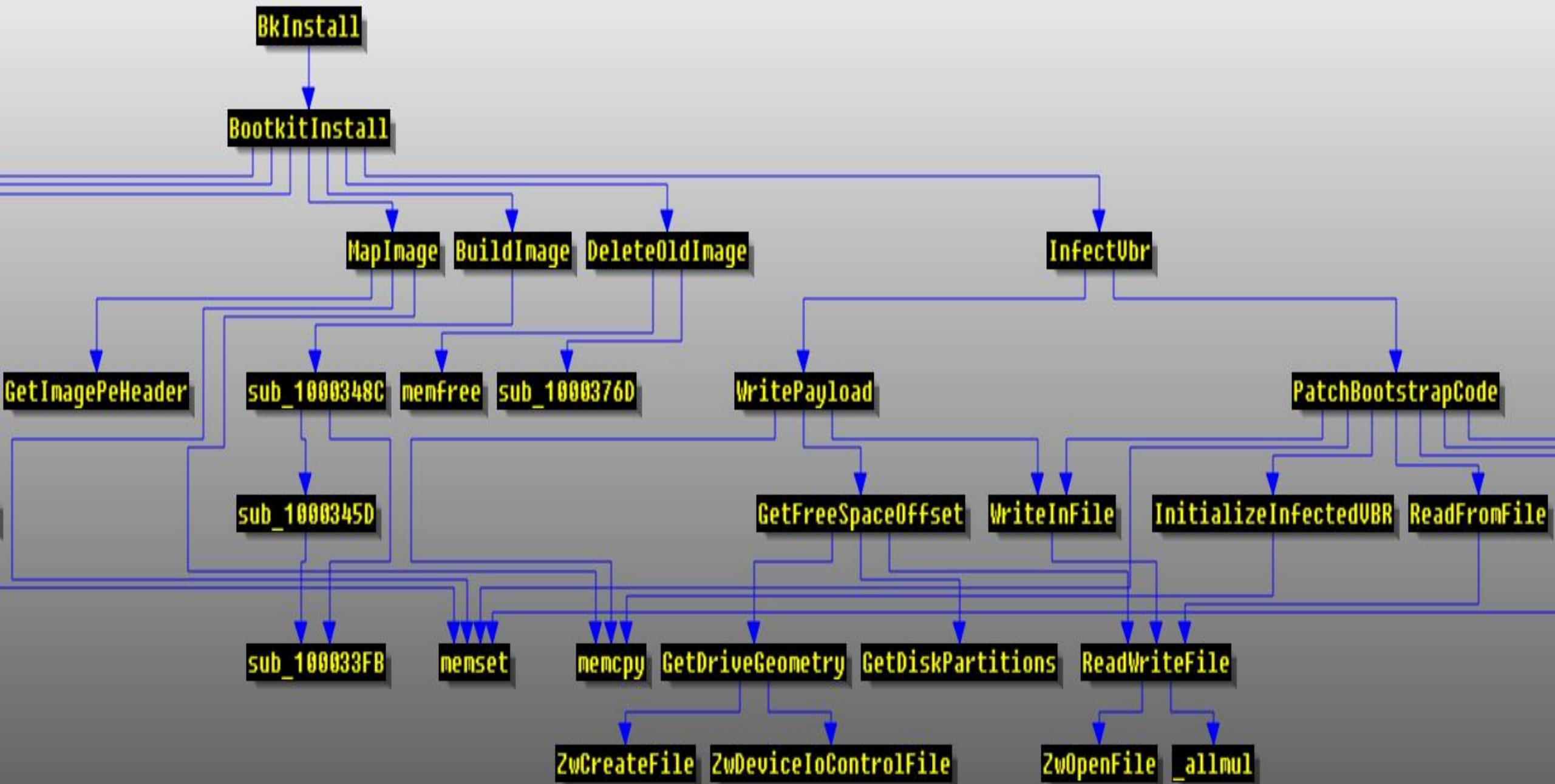
Check OS version

Locate free space on the hard drive to store kernel-mode driver & hidden FS image

Store the driver & hidden FS image in the located area.

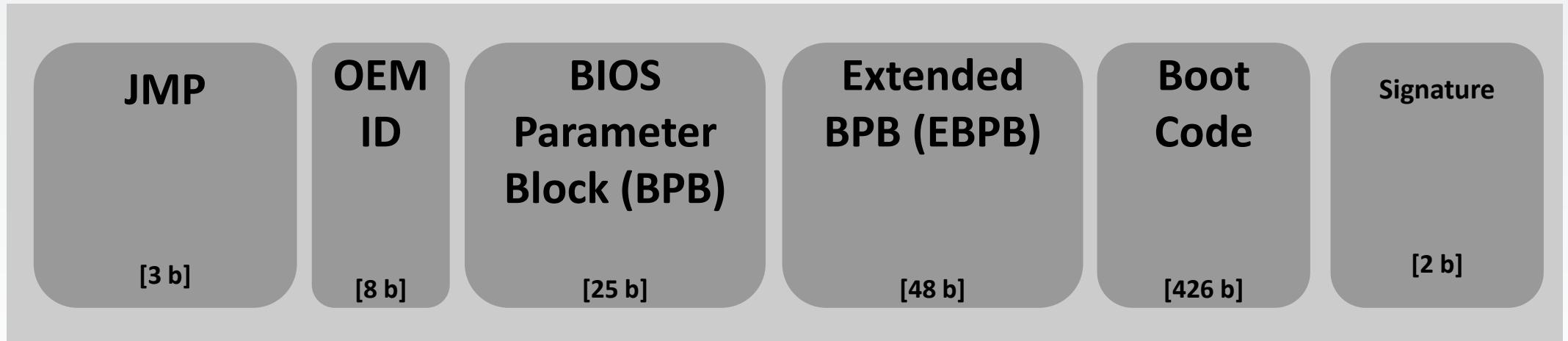
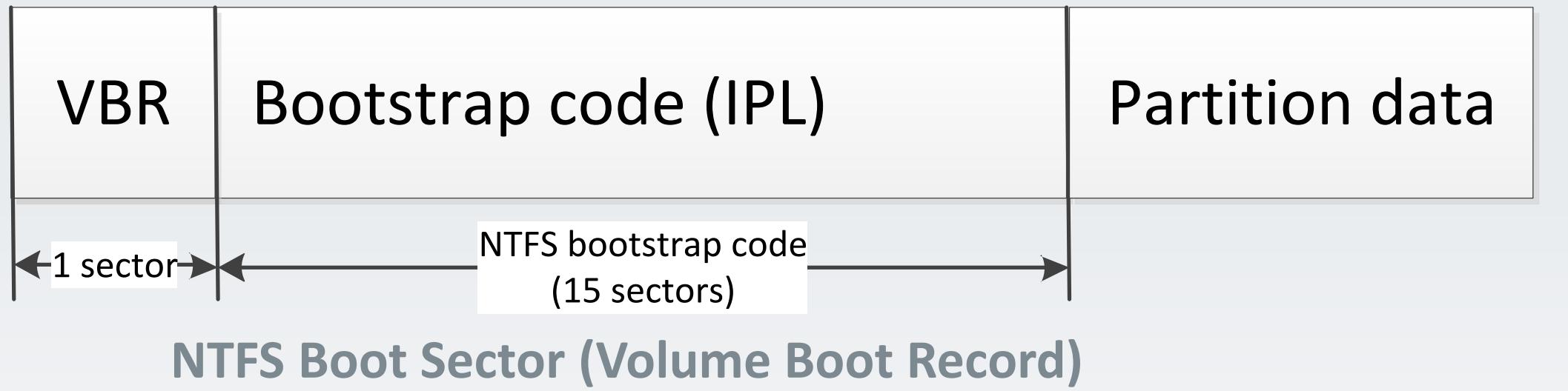
Overwrite bootstrap code of the active partition with malicious one

Callgraph of Bootkit Installation Routine

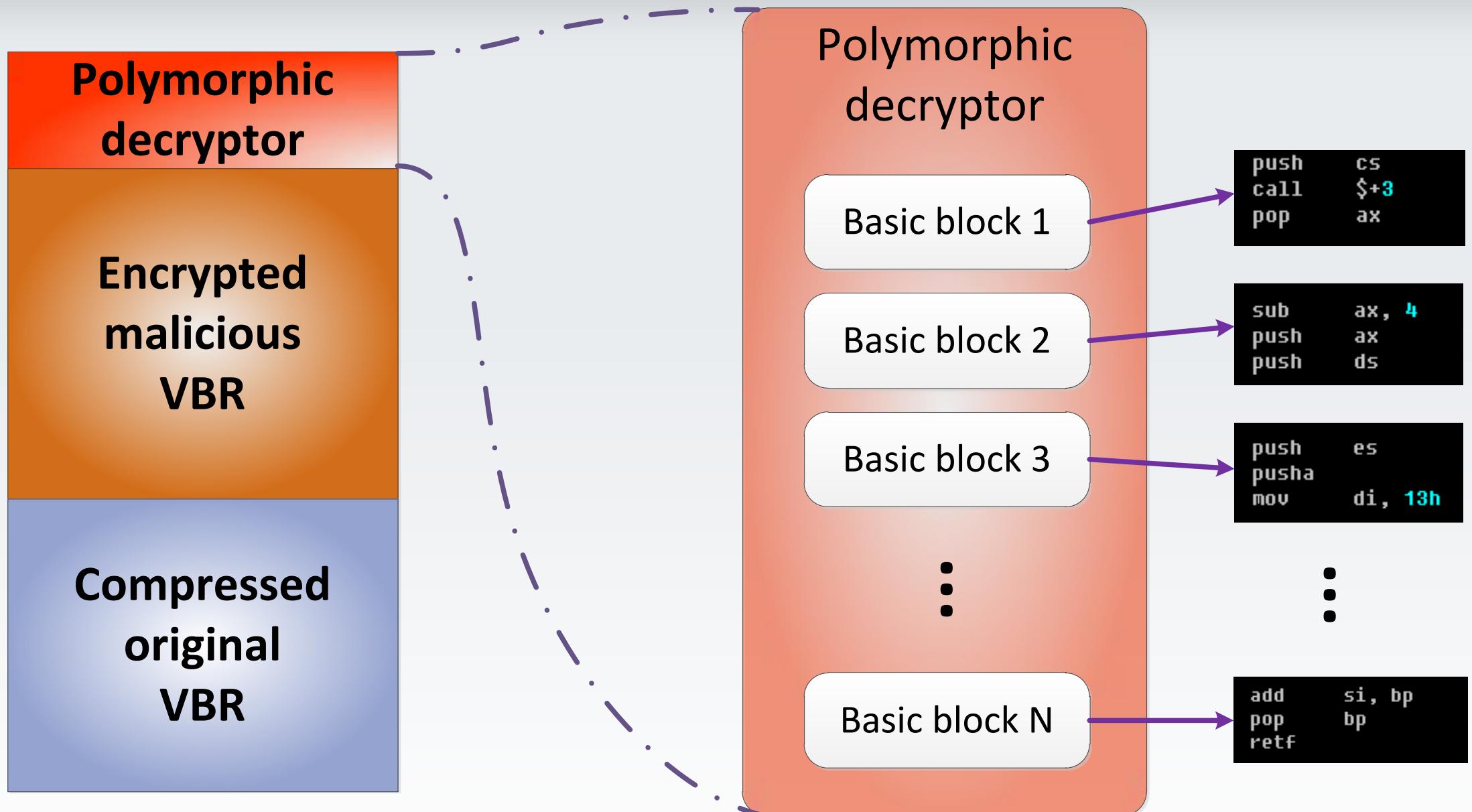


VBR Code Information

VBR is responsible for loading OS boot components (bootmgr, BCD, etc.).



Rovnix Polymorphic VBR



Rovnix Packer

Polymorphic
decryptor

Encrypted
malicious
VBR

Compressed
original
VBR

```
push    cs
call    $+3
pop    ax
jmp    short loc_4E
```

Basic Block 1

```
mov    cx, 469h
```

```
lodsw
xor    ax, dx
jmp    short loc_55
```

Basic Block 2

```
add    si, bp
pop    bp
retf
```

Basic Block N

```
add    ax, 68h ; 'h'
mov    si, ax
add    bp, ax
jmp    short loc_45
```

Basic Block 3

```
push    40h ; '@'
pop    ds
assume ds:nothing
mov    cx, [di]
sub    ecx, 3
mov    [di], cx
jmp    short loc_61
```

Basic Block 4

```
push    cs
call    $+3
pop    ax
```

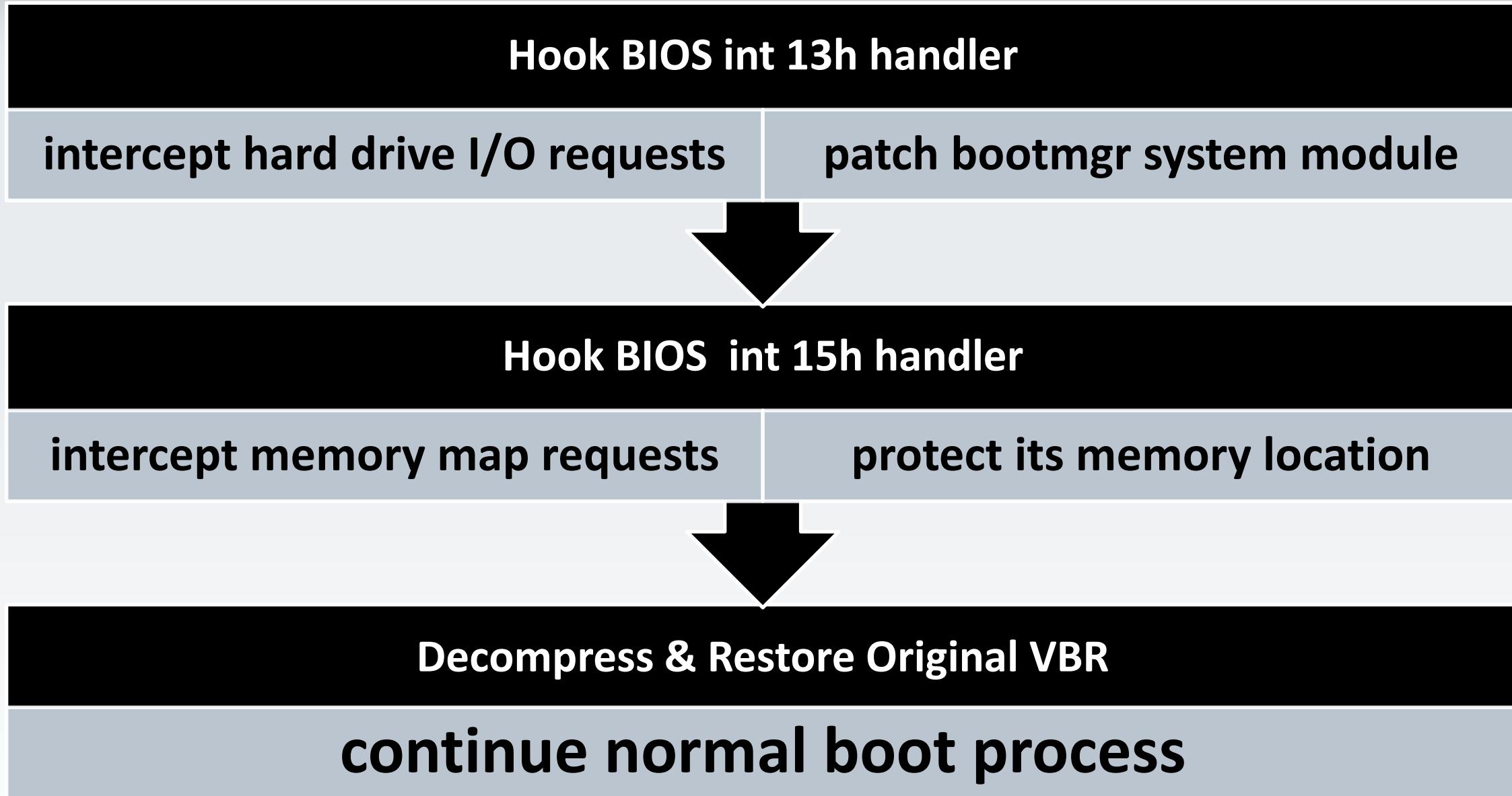
```
sub    ax, 4
push    ax
push    ds
```

```
push    es
pusha
mov    di, 13h
```

⋮
⋮

```
add    si, bp
pop    bp
retf
```

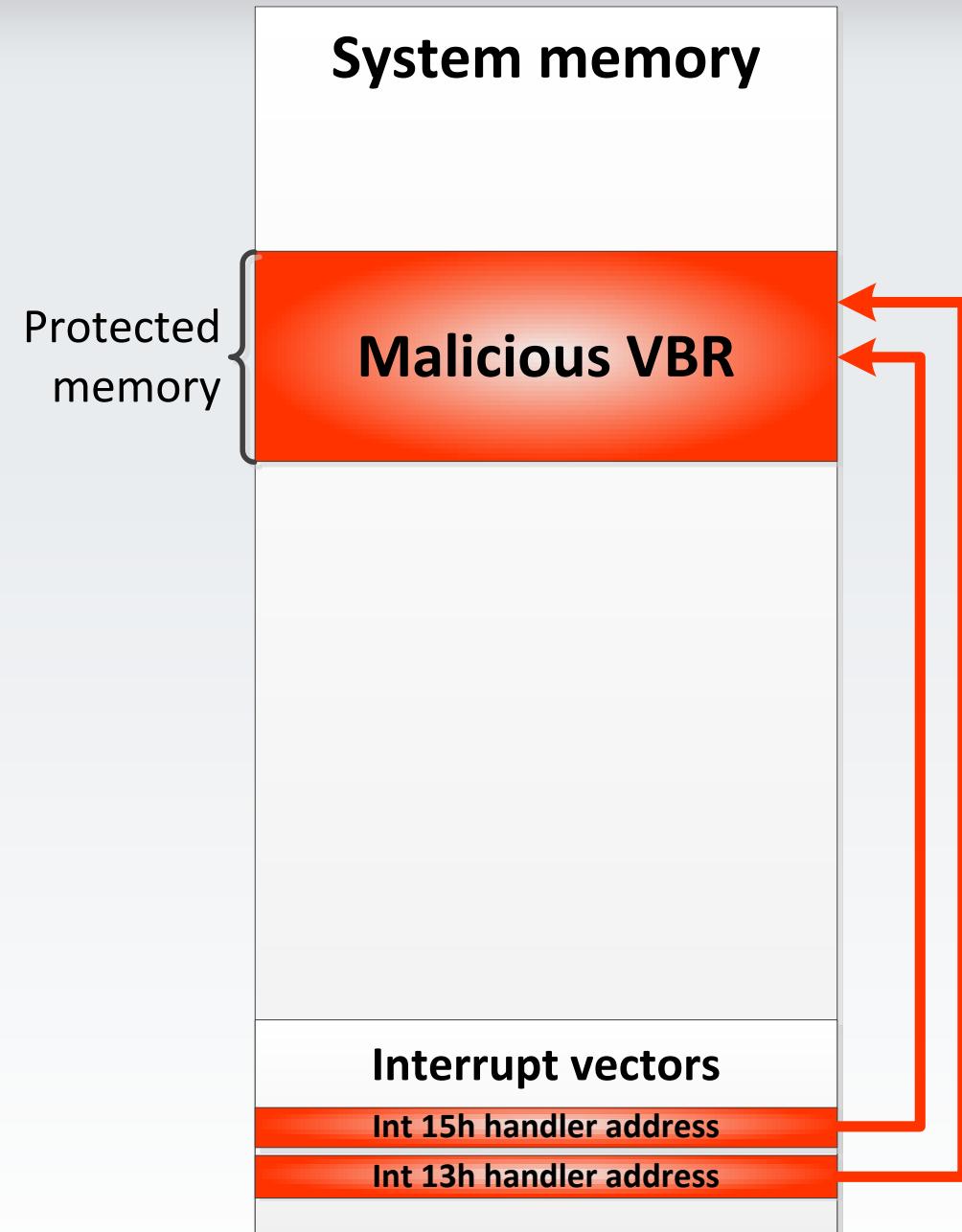
Decrypted VBR code



Hooking BIOS int 15h Handler

Used by operating system to query system address map.

Abused by malicious VBR to protect its memory region from allocation by OS



Surviving Execution Mode Switching

To be able to survive processor execution mode switching the malware:

- ✓ detects execution mode switching operation in bootmgr
- ✓ patches bootmgr right before switching into protected mode
- ✓ copies itself over the last half of IDT (which isn't used by OS)

Surviving Execution Mode Switching

To be able to switch

- ✓ detects
- ✓ patches
- ✓ copies

Execution mode

bootmgr
selected mode
(used by OS)

```
jb    short loc_2A2
pushf
pushad
cmp   ch, 42h ; 'B'
xchg  ax, cx
jnz   short loc_27F
mov   cx, [si+2]

loc_27F:
push  cs
pop   ds
shl   cx, 9
mov   di, bx
cld

loc_287:
mov   al, 0Fh      ; look for constant in bootmgr
repne scasb
jcxz short loc_29F
mov   eax, es:[di]
cmp   eax, 0DB87C022h ; mov cr0, eax
jnz   short loc_287
mov   bp, 168h
call  bp
; -----
db   1Fh          ; patch bootmgr
;

loc_29F:
popad
popf

loc_2A2:
pop   di
pop   bx
pop   es
pop   ds
retf  2
```

Surviving Execution Mode Switching

To be able to survive processor execution mode switching

- ✓ detects execution mode switches in bootmgr
- ✓ patches kernel to switch to protected mode
- ✓ copies its own code (that isn't used by OS)

```
enter 0, 0
mov ax, ds:word_1510
and ax, ax
jnz locret_86F
push large 0
popfd
mov ds:word_1510, 1
mov eax, ds:dword_1514
xor edx, edx
or edx, 1
and eax, 80000000h
jz loc_825
or edx, 80000000h

loc_825: ; CODE XREF: seg000:081A↑j
cli
lgdt fword ptr ds:byte_1500
lidt fword ptr ds:byte_1508
mov eax, cr0
or eax, edx      ; or eax 80000001h
mov cr0, eax      ; switch into PM
xchg bx, bx
nop
jmp short loc_83F
;

nop

loc_83F: ; CODE XREF: seg000:083C↑j
push 50h ; 'P'
push 845h
```

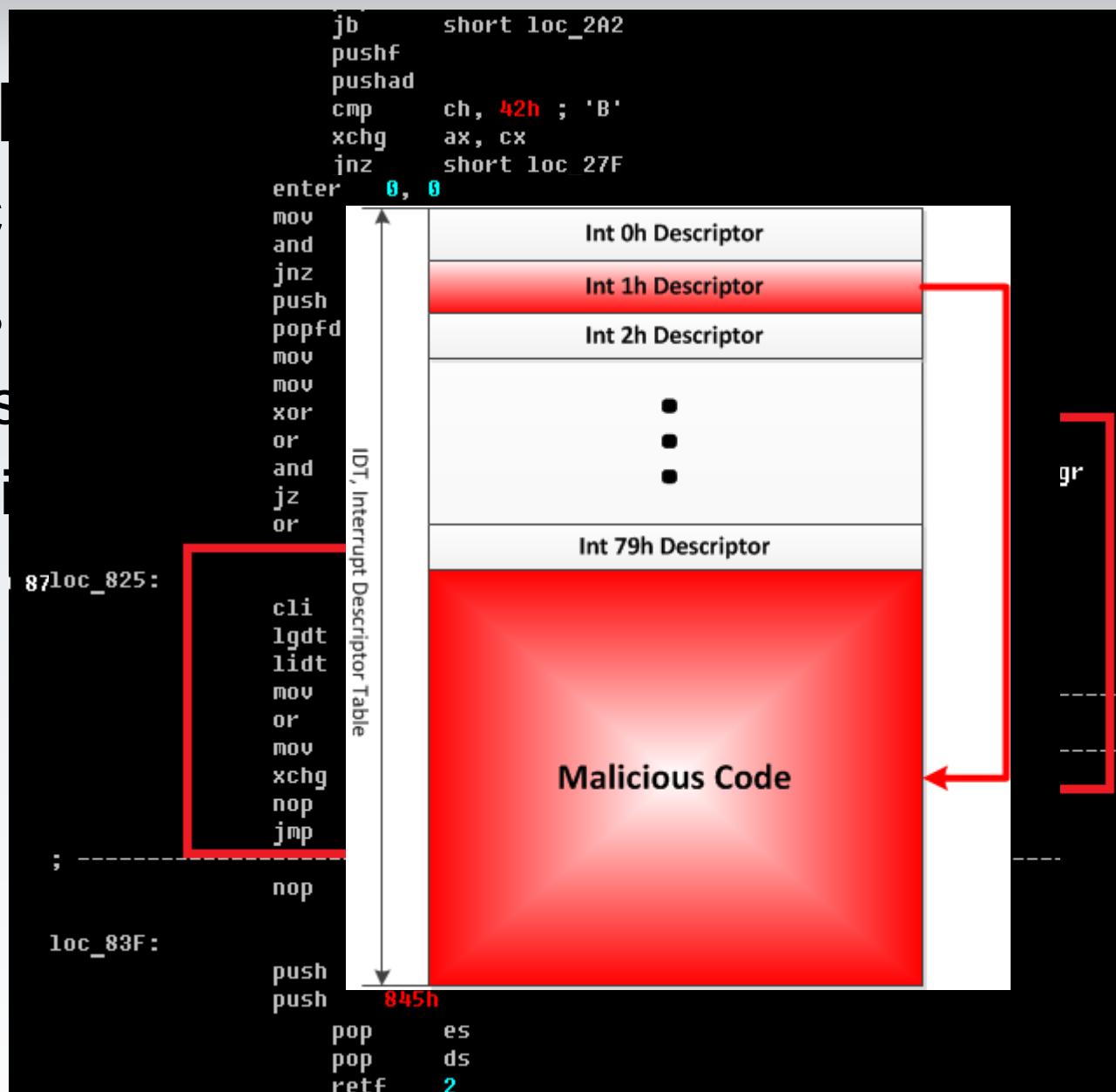
Surviving Execution Mode Switching

To be able to switch

- ✓ detects
- ✓ patches
- ✓ copies

Execution mode

bootmgr
selected mode
(used by OS)



Loading Kernel-mode Driver

To be able to load unsigned kernel-mode driver Rovnix:

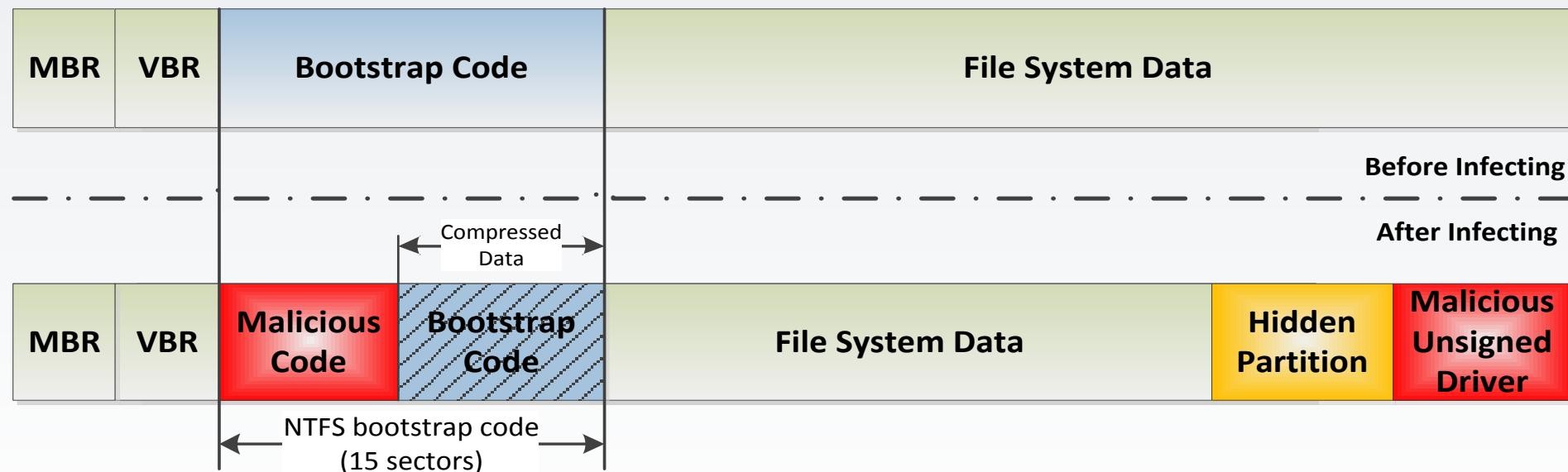
- Waits until kernel-mode memory manager is properly initialized:
 - ✓ Sets up hardware breakpoint
- Allocates memory buffer in kernel-mode address space to store the driver:
 - ✓ Calls *BIAAllocateAlignedDescriptor* system routine to allocate memory buffer
- Inserts corresponding structure in *BootDriverList* of *KeLoaderBlock*.
 - ✓ The driver receives control during boot start drivers initialization

LIVE DEMO



Hidden Storage Layout

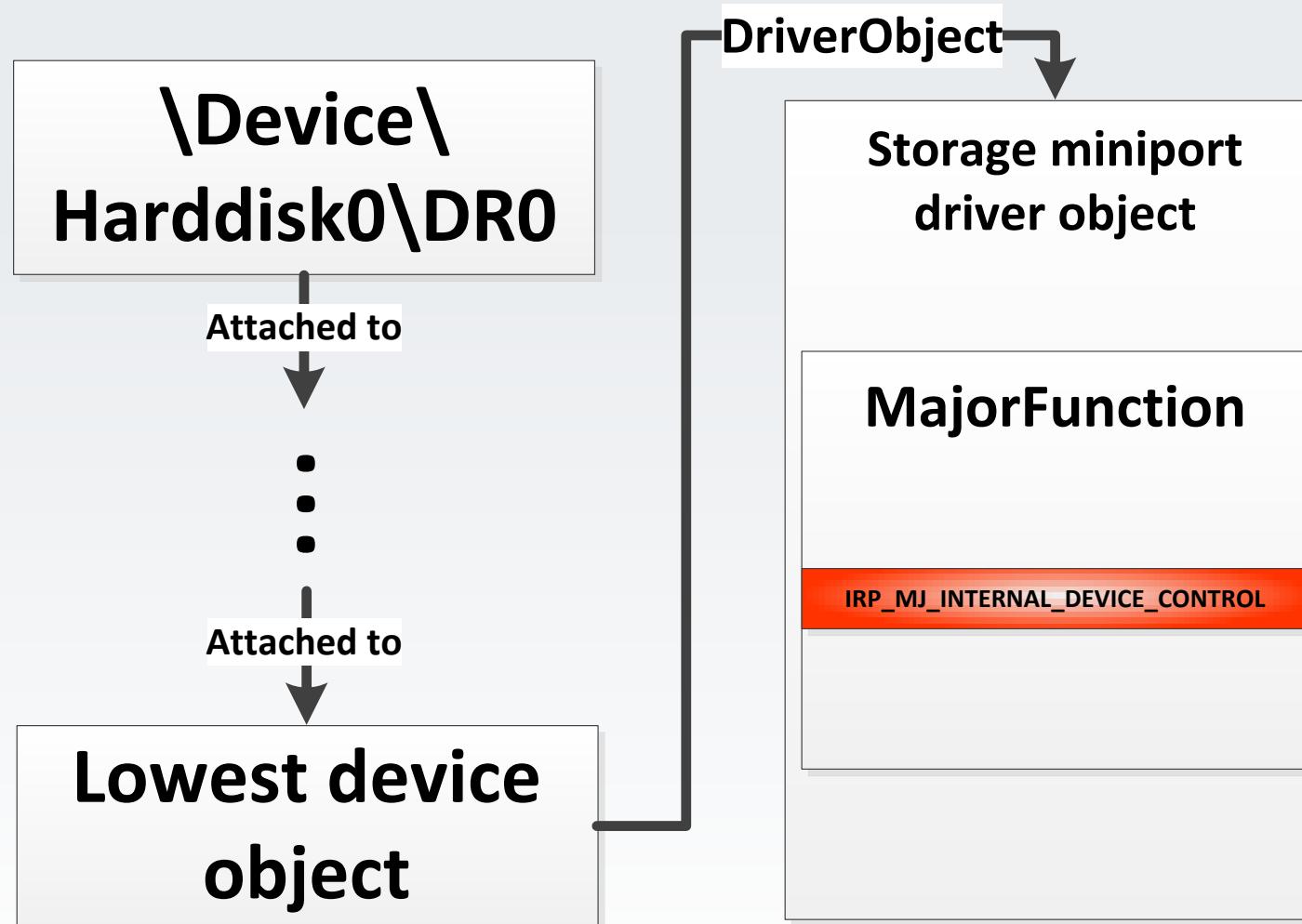
- Rovnix bootkit employs modification of FAT16 for hidden partition
- Hidden partition & kernel-mode driver are written either:
 - ✓ before first partition on the disk – if there is more than 2000 (1 Mb) free sectors
 - ✓ In the end of the hard drive otherwise



```
if ( StartPartitionLeast <= 0x7D0 )
{
    v12 = DiskGeometry->Cylinders.LowPart * DiskGeometry->TracksPerCylinder * DiskGeometry->SectorsPerTrack
        - EndPartition;
    if ( v12 <= 0x7D0 )
    {
        *(QWORD *)&a2->PartitionHiddenStart = (unsigned int)(DiskGeometry->Cylinders.LowPart
                                                * DiskGeometry->TracksPerCylinder
                                                * DiskGeometry->SectorsPerTrack)
                                                - 0x7D0i64;
        a2->PartitionHiddenSize = 2000;
        a2->BytesPerSector = DiskGeometry->BytesPerSector;
        result = 0;
    }
    else
    {
        a2->PartitionHiddenStart = EndPartition + 1;
        a2->PartitionHiddenXXX = 0;
        a2->PartitionHiddenSize = v12 - 1;
        a2->BytesPerSector = DiskGeometry->BytesPerSector;
        result = 0;
    }
}
else
{
    a2->PartitionHiddenXXX = 0;
    a2->PartitionHiddenStart = 1;
    a2->PartitionHiddenSize = StartPartitionLeast - 1;
    a2->BytesPerSector = DiskGeometry->BytesPerSector;
    result = 0;
}
return result;
}
```

Self-defense Mechanisms

To be able to protect VBR & Hidden file system Rovnix bootkit hooks IRP_MJ_INTERNAL_DEVICE_CONTROL handler:



Self-defense Mechanisms

```
int __stdcall NewIrpMjInternalHandler(unsigned int DeviceObject, PIRP _Irp)
{
    PDEVICE_OBJECT _DeviceObject; // ebx@1
    PIRP Irp; // esi@1
    UCHAR ScsiCommand; // al@2
    int Status; // edi@7
    unsigned __int64 Lba; // [sp+Ch] [bp-Ch]@2
    PVOID pTransferBuffer; // [sp+14h] [bp-4h]@2

    _DeviceObject = (PDEVICE_OBJECT)_DeviceObject;
    Irp = _Irp;
    if ( (PDEVICE_OBJECT)_DeviceObject != Dr0DeviceObject )
        return OriginalIrmMjInternalHandler(_DeviceObject, Irp);
    ScsiCommand = GetSrbParameters(_Irp, (_int64 *)&Lba, (int *)&DeviceObject, &pTransferBuffer, (DWORD *)&_Irp);
    if ( ScsiCommand == 0x2A || ScsiCommand == 0x3B )// SCSI write commands
    {
        if ( !CheckSrbParams(Lba, DeviceObject) )
            return OriginalIrmMjInternalHandler(_DeviceObject, Irp);
        Status = STATUS_ACCESS_DENIED; // return STATUS_ACCESS_DENIED
        Irp->IoStatus.Status = STATUS_ACCESS_DENIED;
        IoCompleteRequest(Irp, 0);
    }
    else // SCSI read commands
    {
        if ( ScsiCommand != 0x28 && ScsiCommand != 0x3C || !CheckSrbParams(Lba, DeviceObject) )
            return OriginalIrmMjInternalHandler(_DeviceObject, Irp);
        Status = SetCompletionRoutine(_DeviceObject, Irp, Lba, SHIDWORD(Lba), DeviceObject, (int)pTransferBuffer, (int)_Irp);
    }
    if ( Status == STATUS_REQUEST_NOT_ACCEPTED )
        return OriginalIrmMjInternalHandler(_DeviceObject, Irp);
    return Status;
}
```



Hidden File System Reader



ESET Hidden File System Reader

1.0.0.0 beta (Jun 9 2012 13:40:43)

Copyright (c) 1992-2012 ESET, spol. s r.o. All rights reserved.

Processing... Please wait.

"Rovnix.b_Driver" file system found:

- payload.sys

md5: 063E50BC2269F5D3858D53BB0C15527E

- vbr

md5: C1DD3EB02DA9FE9AF1C09E5EF0964451

"Rovnix.b_FS" file system found:

- BOOT.SYS

md5: 7FB1F36BFF3B6BE3FA4D7C1B4CCE5E61

File system(s) successfully exported!

Hidden File System Reader



ESET Hidden File System Reader

1.0.0.0 beta (Jun 20 2012 13:40:43)

final version will be released

"Rovnix.b_Driver" file system found:

- payload.sys
- vbr

"Rovnix.b_FS" file system found:

- BOOT.SYS

Processing...
at



C2269F5D3858D53BB0C15527E

02DA9FE9AF1C09E5EF0964451

md5: 7FB1F36BFF3B6BE3FA4D7C1B4CCE5E61

File system(s) successfully exported!



LIVE DEMO

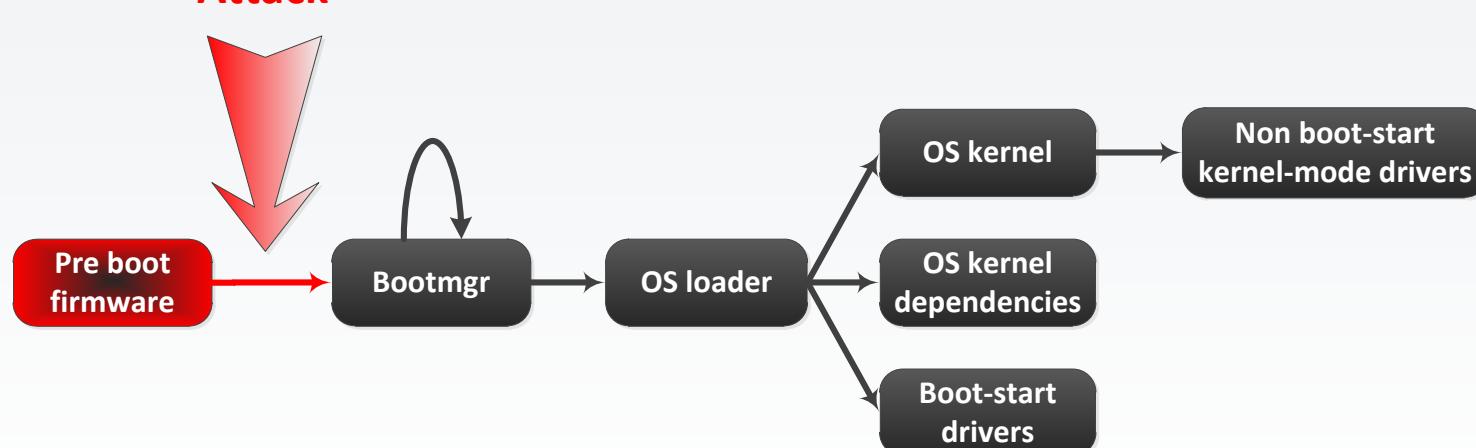
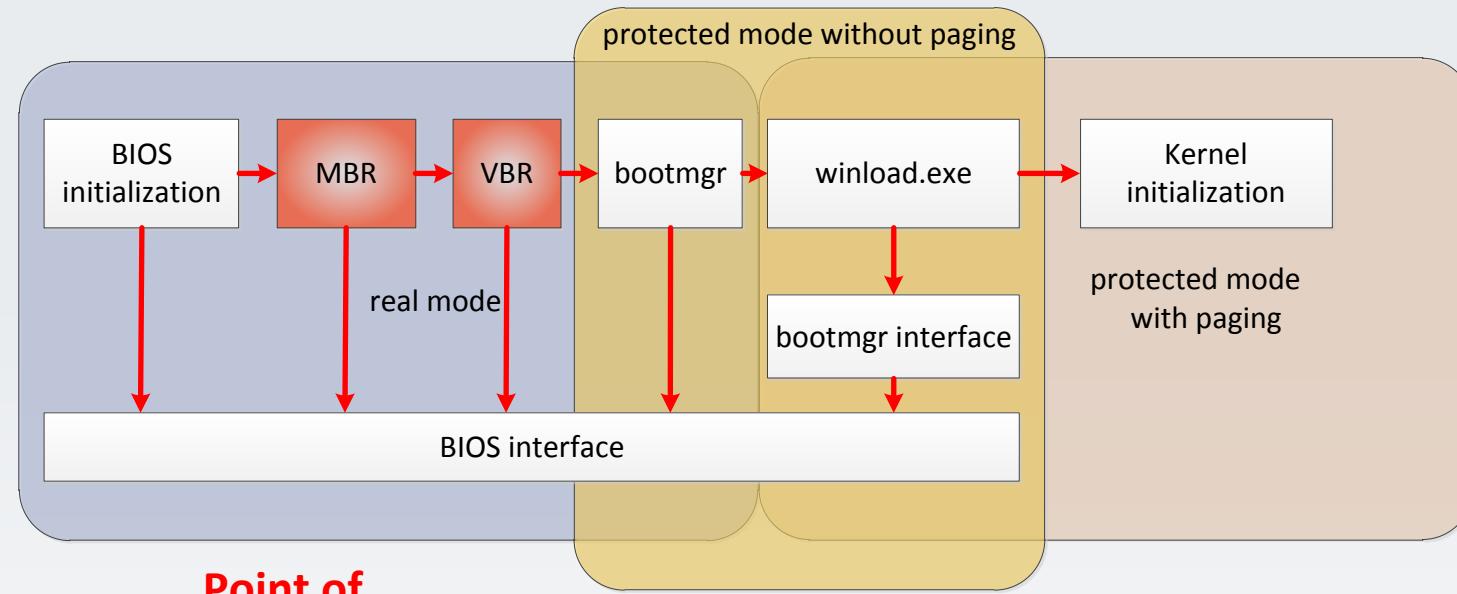


Bootkit countermeasures



Problem Description

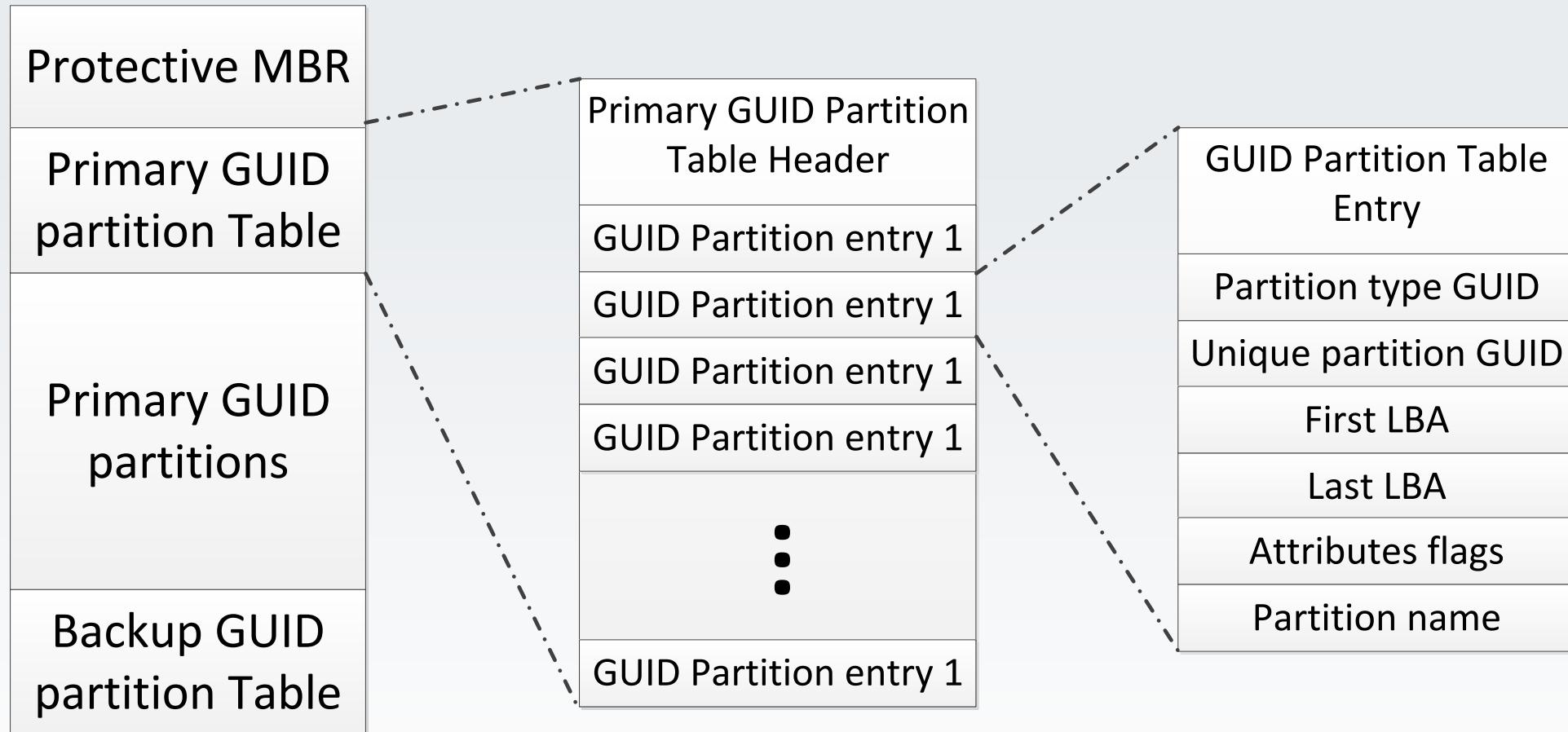
✓ Untrusted platform problem:



Bootkits & GPT Disks

There is no MBR & VBR code which is executed in GPT disks

- ✓ Bootkits in-the-wild aren't applicable to GPT disks



Bootkits & GPT Disks

UEFI Firmware

UEFI Boot Manager

Windows Boot Manager (bootmgr.efi)

Windows OS Loader (winload.efi)

OS Kernel (ntoskrnl.exe)

Windows 8 Security Features

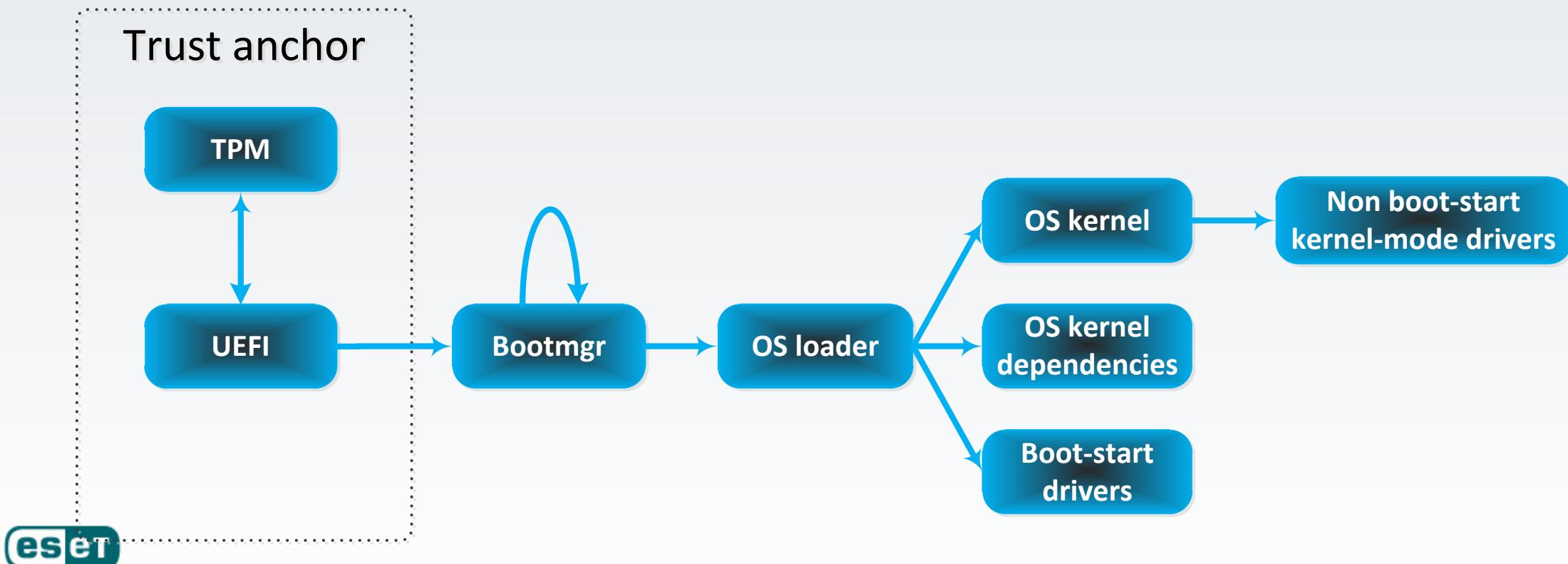
Security enhancements introduced in Windows 8:

- **Secure boot technology**
 - ✓ Employing UEFI secure boot in conjunction with TPM
- **Early anti-malware launch module**
 - ✓ Allows antimalware software start before any other third-party components

Secure Boot

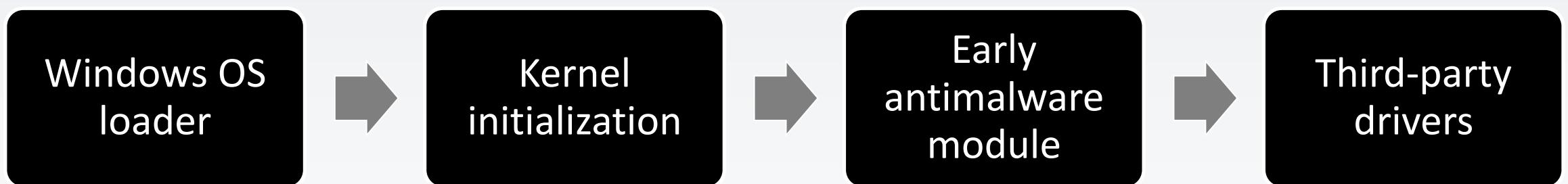
Secure boot prevents running an unknown OS loader:

- ✓ UEFI will verify OS loader
- ✓ The key for verification is stored inside TPM



Early antimalware launch module

Antimalware component receives control before any other third-party software at boot time.



Conclusion

- ✓ Bootkit technology allows malware to load unsigned kernel-mode driver and achieve high degree of stealth in the system
- ✓ The main target of bootkit infection are MBR & VBR
- ✓ Rovnix is a first known bootkit infecting VBR
- ✓ The most interesting features of the latest modification of Rovnix bootkit are:
 - ✓ Polymorphic infected VBR
 - ✓ Hidden Storage
- ✓ There are additional security features introduced in Windows 8 OS:
 - ✓ Early antimalware launch module
 - ✓ Secure Boot



References

- ✓ **Rovnix Reloaded: new step of evolution**

<http://blog.eset.com/2012/02/22/rovnix-reloaded-new-step-of-evolution>

- ✓ **TDL4 reloaded: Purple Haze all in my brain**

<http://blog.eset.com/2012/02/02/tdl4-reloaded-purple-haze-all-in-my-brain>

- ✓ **Bootkit Threat Evolution in 2011**

<http://blog.eset.com/2012/01/03/bootkit-threat-evolution-in-2011-2>

- ✓ **The Evolution of TDL: Conquering x64**

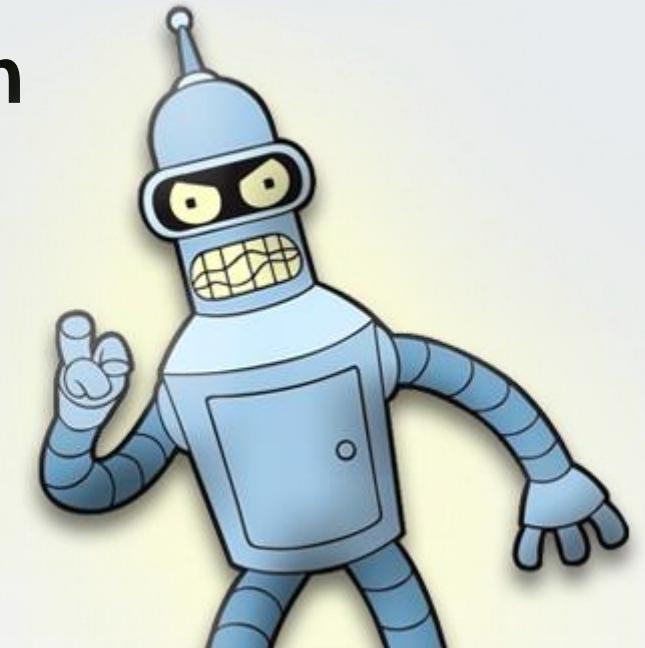
http://go.eset.com/us/resources/white-papers/The_Evolution_of_TDL.pdf

- ✓ **Modern bootkit trends: bypassing kernel-mode signing policy**

<http://www.virusbtn.com/conference/vb2011/abstracts/LastMinute1.xml>

- ✓ **King of Spam: Festi botnet analysis**

<http://blog.eset.com/2012/05/11/king-of-spam-festi-botnet-analysis>





www.zeronights.ru

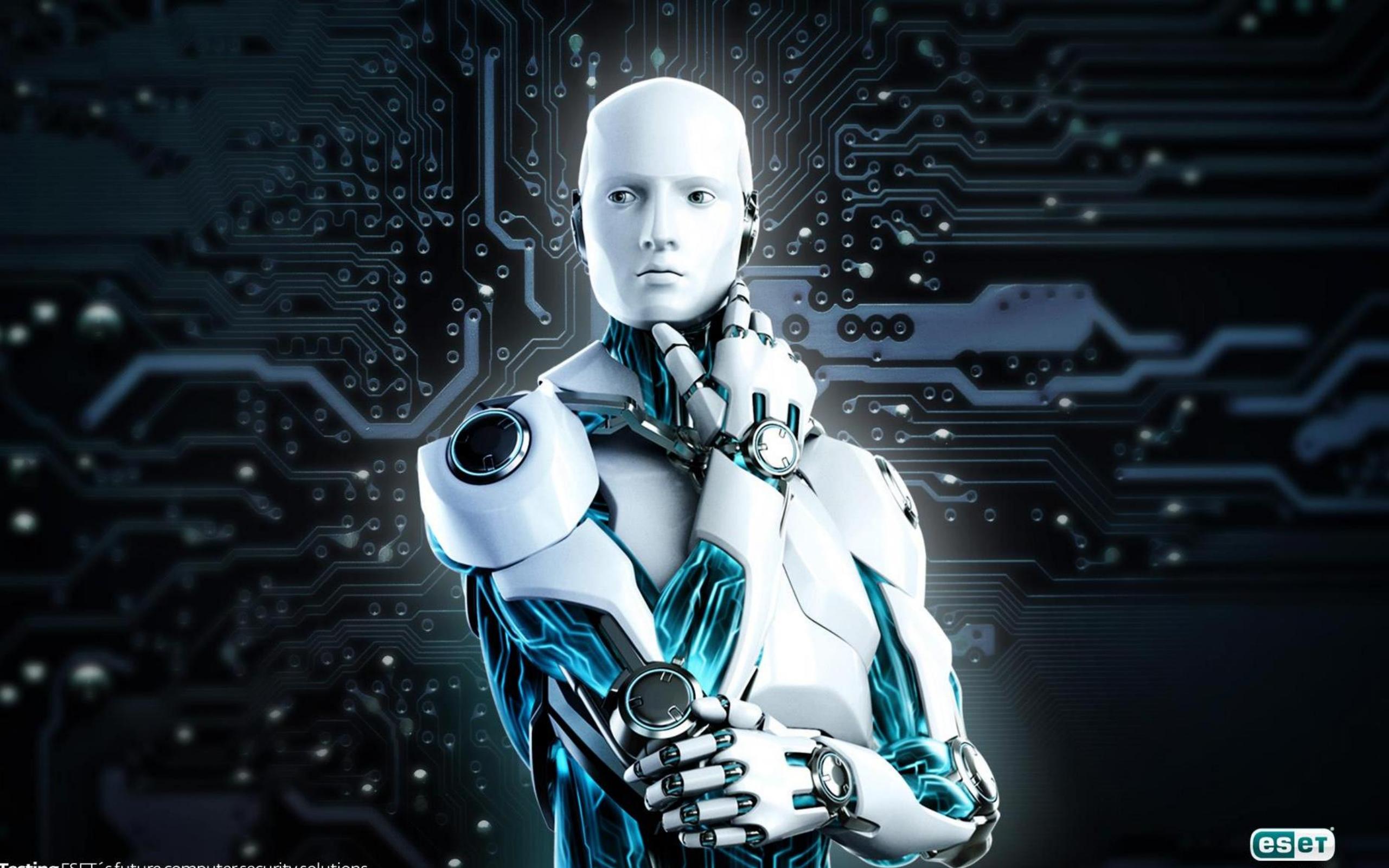


Moscow, Russia

19-20 November

cfp@zeronights.ru





eset

Testing ESET's future computer security solutions

Thank you for your attention!

Aleksandr Matrosov
matrosov@eset.sk
[@matrosov](https://twitter.com/matrosov)

Eugene Rodionov
rodionov@eset.sk
[@vxradius](https://twitter.com/vxradius)

