

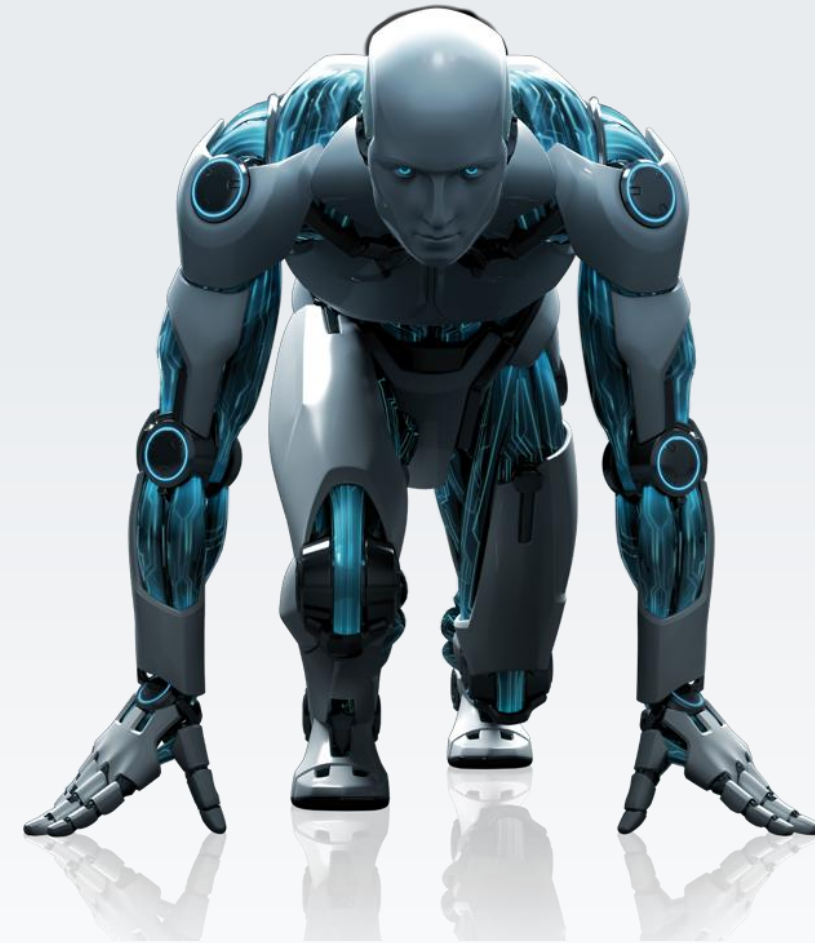
Defeating Anti-forensics in Contemporary Complex Threats

Eugene Rodionov
Aleksandr Matrosov

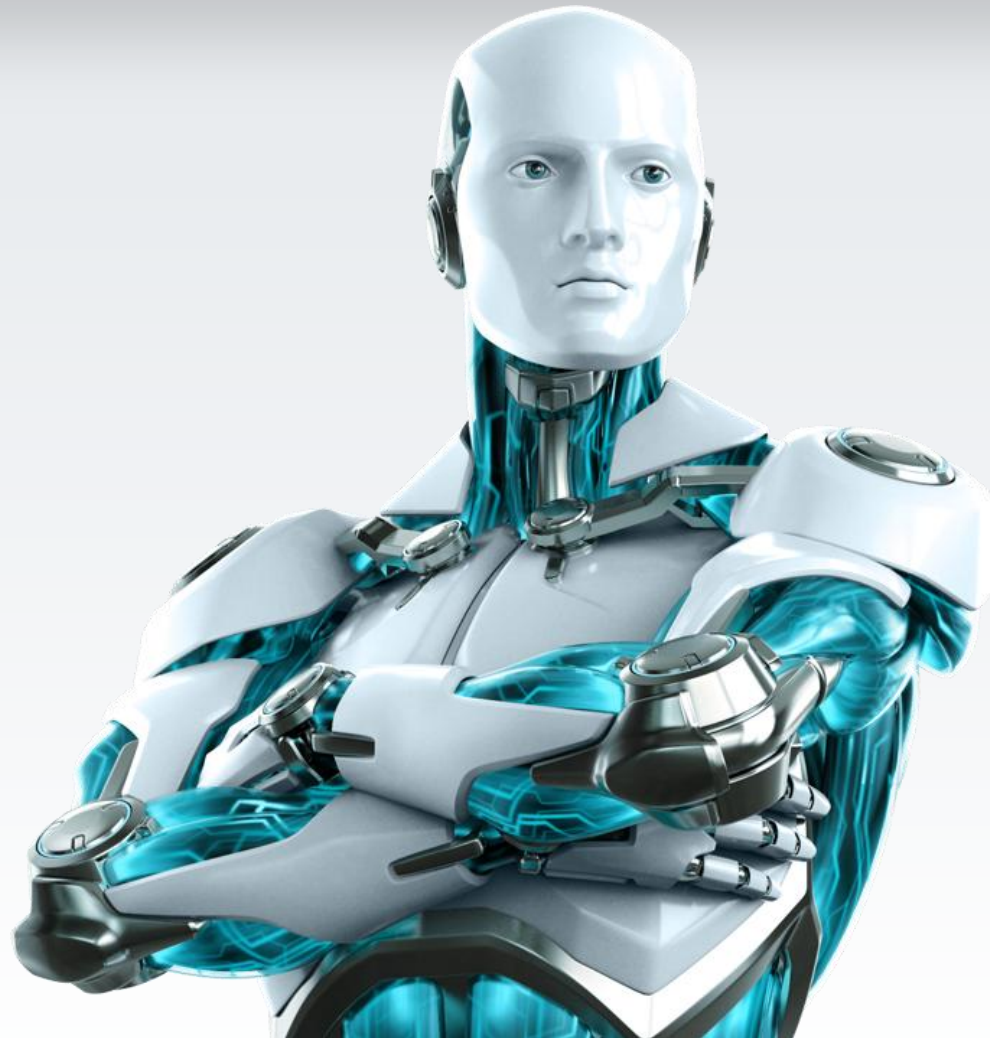


Outline of The Presentation

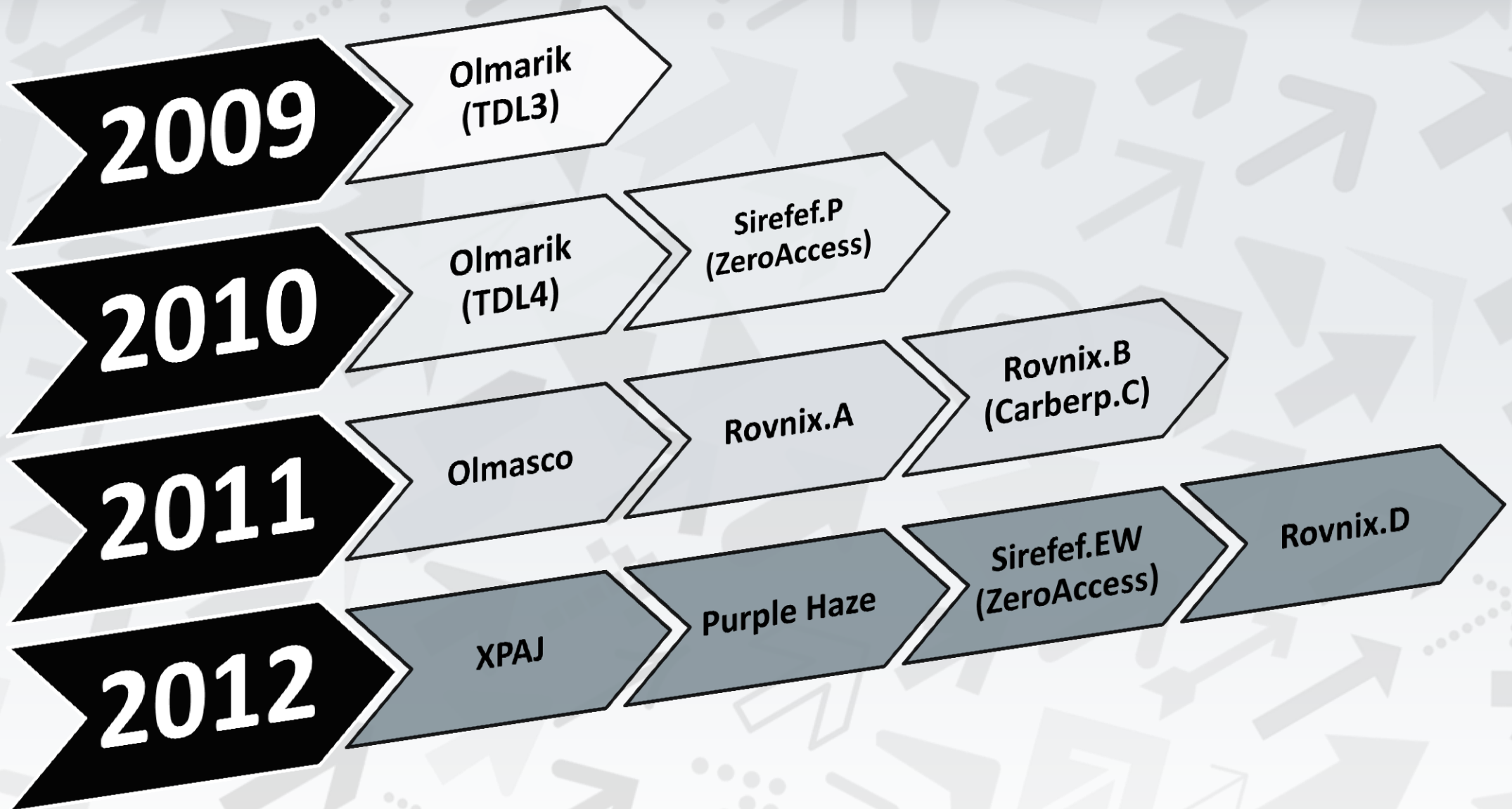
- Hidden file systems
 - ✓ Why?
 - ✓ Anti-Forensics approaches
- Hidden storage design principles
 - ✓ Architecture
 - ✓ Hidden file systems in the wild
 - ✓ TDL3/TDL4/Olmasco/Rovnix/Sirefef ...
- HiddenFsReader forensic tool



Hidden File Systems



Hidden File Systems Evolution Over Time



Why?

Why do modern complex threats need a hidden FS?

- ✓ Secure storage of components and configuration file(s)
 - bypass standard forensic methods for extraction
- ✓ High level of stealth
 - there are no malicious files in the file system to detect
- ✓ High degree of survivability
 - difficult to detect and remove
- ✓ Ability to disable and bypass security software
 - the malware is launched before security software

Anti-Forensics

Bypass typical forensic techniques

➤ Problems:

- ✓ Malicious files are not stored in the file system (difficult to extract)
- ✓ Hidden storage cannot be decrypted without malware analysis
- ✓ Typical forensic tools do not work out of the box

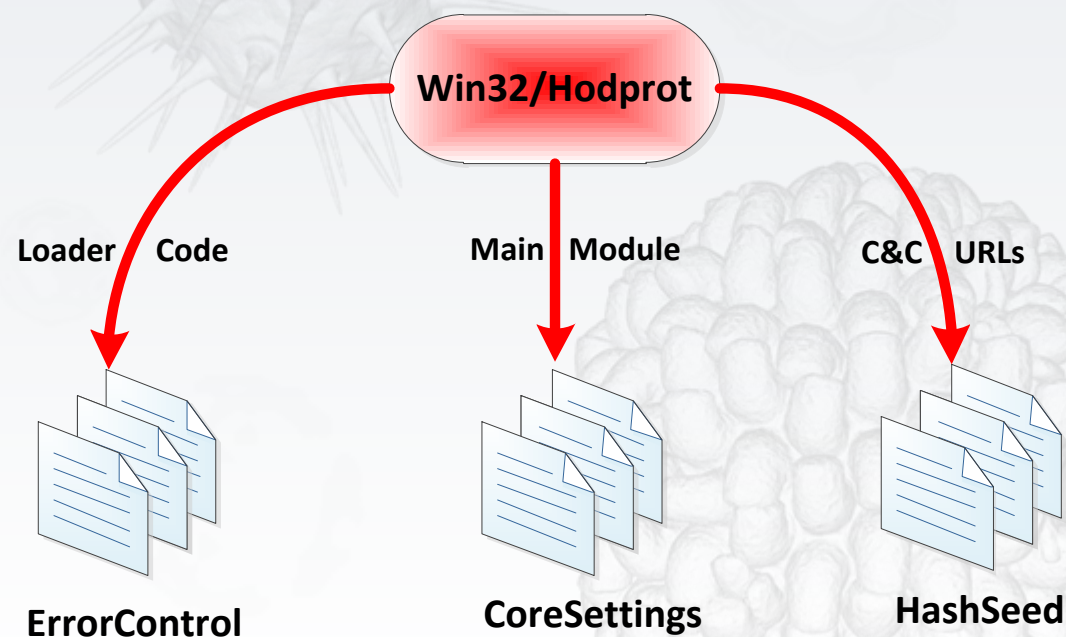
➤ Solutions:

- ✓ Malware analysis and reconstruction algorithms
- ✓ Development of custom tools for extracting hidden content






Anti-Forensics: Hodprot

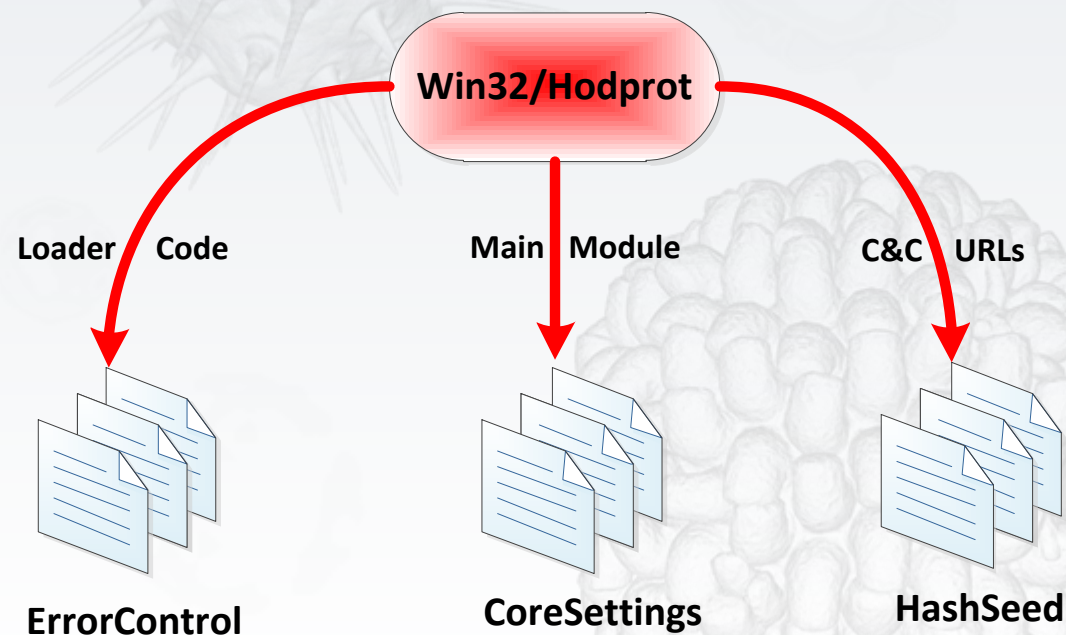
One of the most frequently used droppers in 2011 for delivering banking trojans in Russia: Carberp, Sheldor, RDPdoor.

Relies on system registry to keep its modules and payload
HKLM\SOFTWARE\Settings

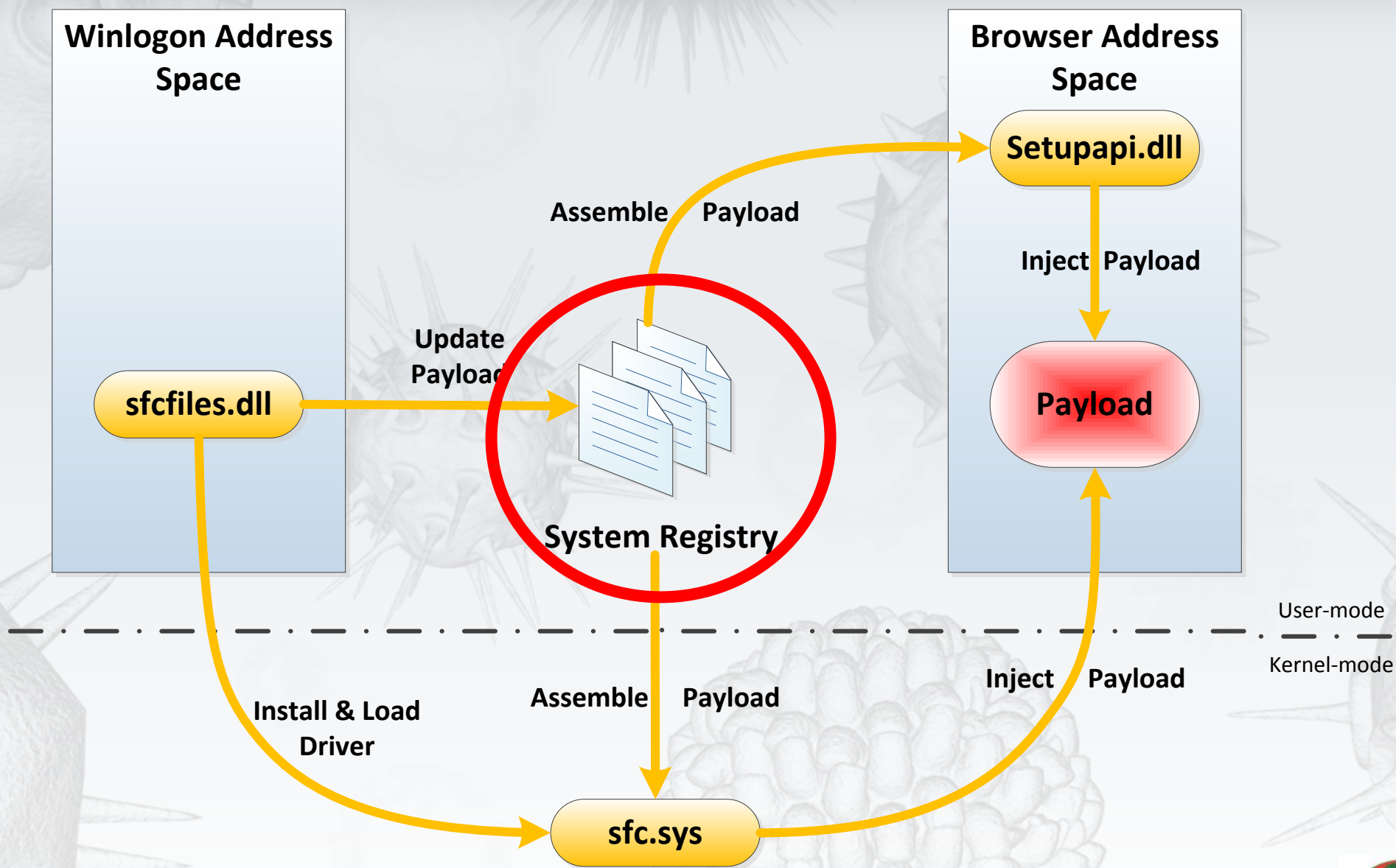


Anti-Forensics: Hodprot

One of banking	 CoreSettings	REG_BINARY	3d 1b 4a a6 4d a5 8d 43 34 a5 43 99 e4 b2 33 46 63 a5 8c 33 38 86 83 b9
	 CryptoHash	REG_BINARY	3d 1b 4a a6 4d a5 8d 43 34 a5 43 99 e4 b2 33 46 63 a5 8c 33 38 86 83 b9
	 DriveSettings	REG_BINARY	3d 1b 4a a6 4d a5 8d 43 34 a5 43 99 e4 b2 33 46 63 a5 8c 33 38 86 83 b9
Relies HKLM	 ErrorControl	REG_BINARY	ab a3 52 84 c9 87 63 61 12 e1 61 64 1b 4d 33 46 8f 5a 91 63 b1 66 1b 43
	 HashSeed	REG_BINARY	cb a2 31 c8 0e 57 ad 9a 4b 26 32 19 c0 a7 d3 79 0e 06 3f 8b 62 e2 51 b8



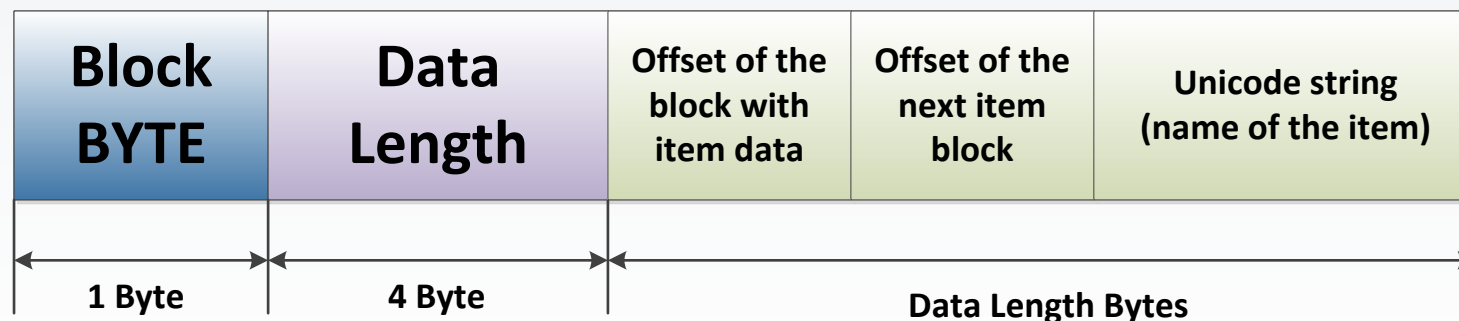
Anti-Forensics: Hodprot



Flame configuration data storage

Configuration data are stored in a resource of the Flame main module

Configuration data are encrypted with custom algorithm and compressed with *DEFLATE* algorithm



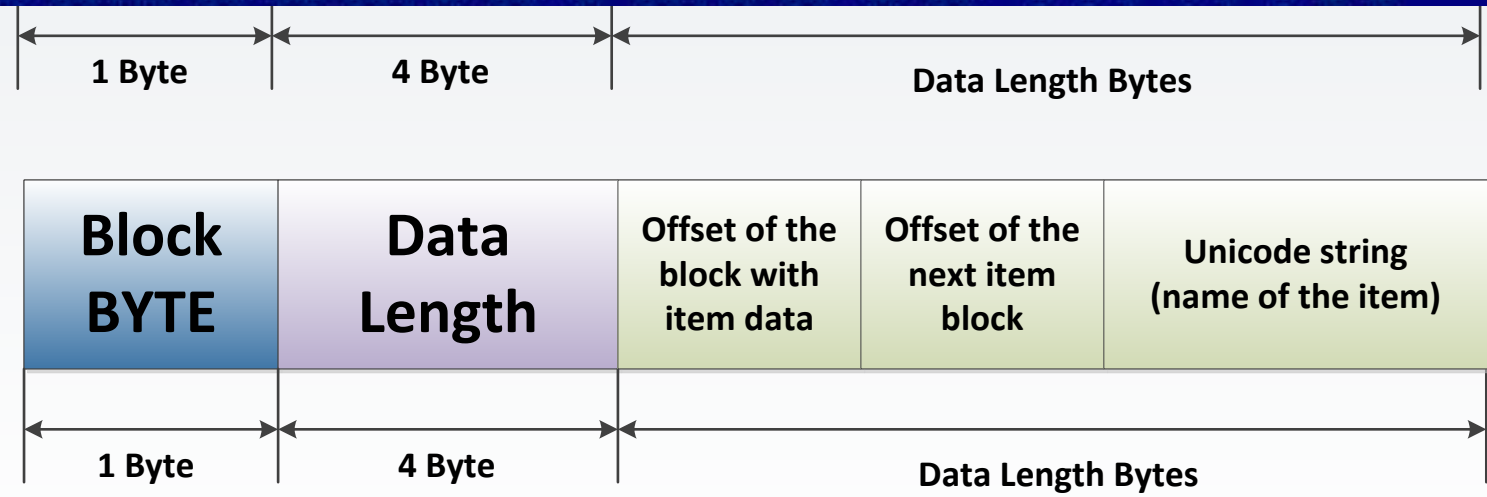
Flame configuration data storage

```

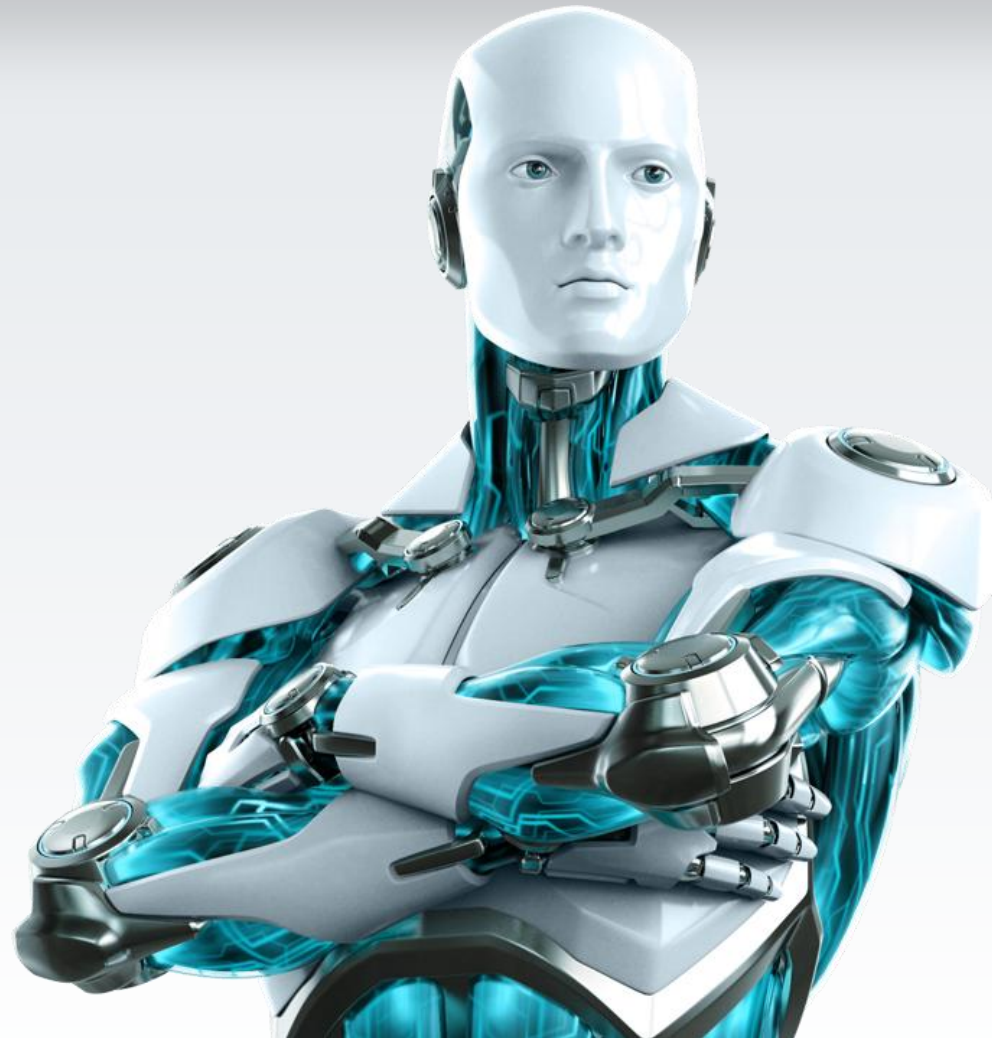
0038A6C0: 00 4F 00 5F 00 44 00 45 00 4C 00 45 00 54 00 45 0 _ D E L E T E
0038A6D0: 00 2E 00 73 00 69 00 7A 00 65 00 7C 85 AB 0C 06 . s i z e ! E A
0038A6E0: 04 00 00 00 00 00 00 00 DE 63 59 11 03 54 00 00 *8 D *8 I R T S
0038A6F0: 00 DF A6 38 00 84 A6 38 00 FF FE 52 00 54 00 53 . M E D I A _ S
0038A700: 00 2E 00 4D 00 45 00 44 00 47 00 41 00 5F 00 53 E T U P . F I L
0038A710: 00 45 00 54 00 55 00 50 00 2E 00 46 00 49 00 4C E S _ T O _ D E
0038A720: 00 45 00 53 00 5F 00 54 00 4F 00 5F 00 44 00 45 L E T E . F I P
0038A730: 00 4C 00 45 00 54 00 45 00 2E 00 66 00 69 00 72 s t < # $ T 0 0 0 0 0 0 0
0038A740: 00 73 00 74 00 7B 23 15 C2 AE AE AE AE AE AE AE 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0038A750: AE AF AF AF 06 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0038A760: 11 03 52 00 00 00 54 A7 38 00 EC A6 38 00 FF FE R T S . M E D I
0038A770: 52 00 54 00 53 00 2E 00 4D 00 45 00 44 00 47 00 A _ S E T U P .
0038A780: 41 00 5F 00 53 00 45 00 54 00 55 00 50 00 2E 00 F I L E S _ T O
0038A790: 46 00 49 00 4C 00 45 00 53 00 5F 00 54 00 4F 00 _ D E L E T E .
0038A7A0: 5F 00 44 00 45 00 4C 00 45 00 54 00 45 00 2E 00 l a s t w o r d i
0038A7B0: 6C 00 61 00 73 00 74 00 EE 70 FH F5 06 04 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0038A7C0: 00 01 00 00 00 BB 04 E5 A9 03 52 00 00 00 BC 07 @ 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0038A7D0: 38 00 61 A7 38 00 FF FE 52 00 54 00 53 00 2E 00 8 a 3 8 I R T S .
0038A7E0: 4D 00 45 00 44 00 49 00 41 00 5F 00 53 00 45 00 M E D I A _ S E
0038A7F0: 54 00 55 00 50 00 2E 00 46 00 49 00 4C 00 45 00 T U P . F I L E
    
```

block byte
 data length
 offset to item data
 offset to next item

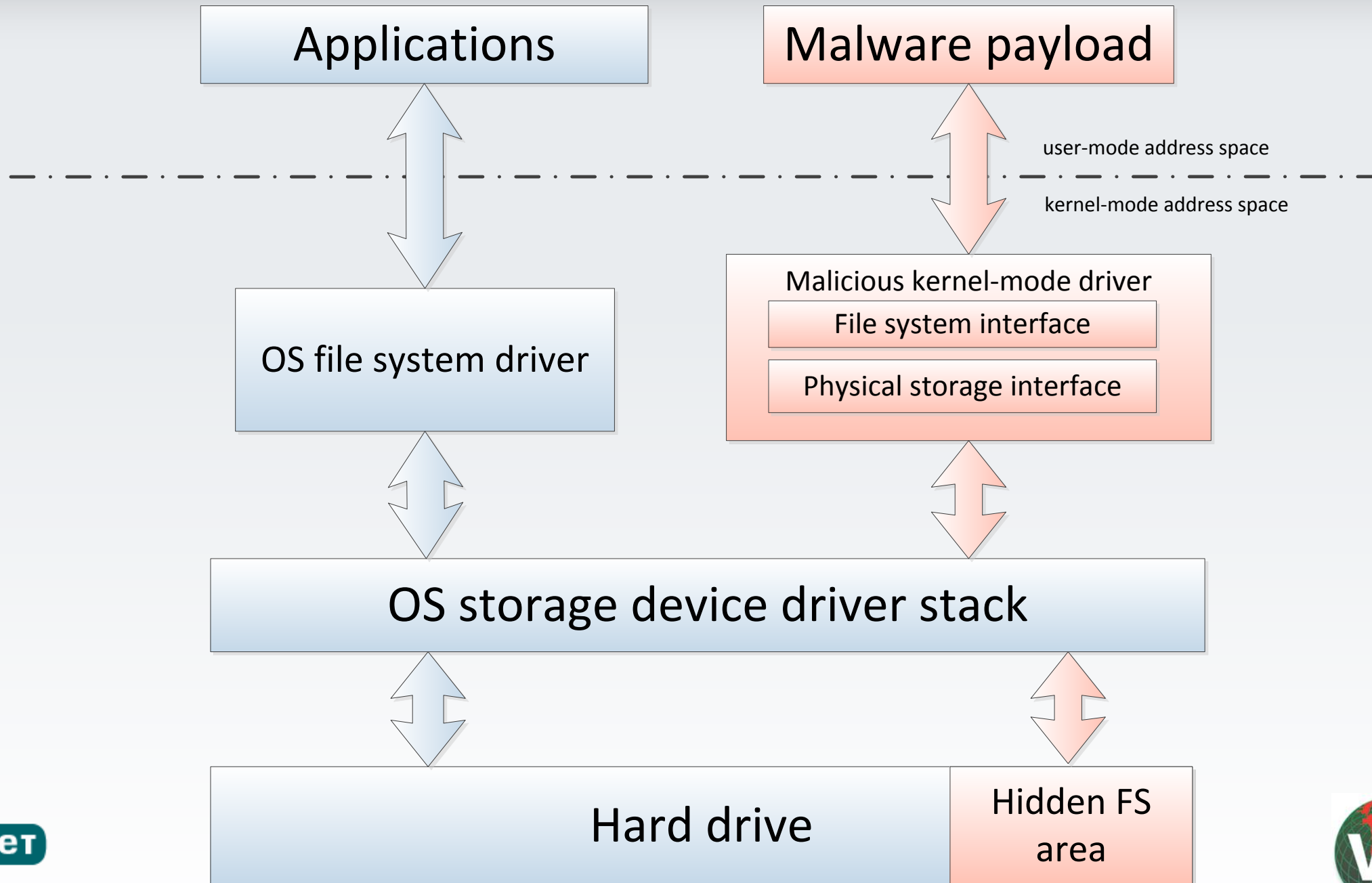
 item name



Hidden storage design principles



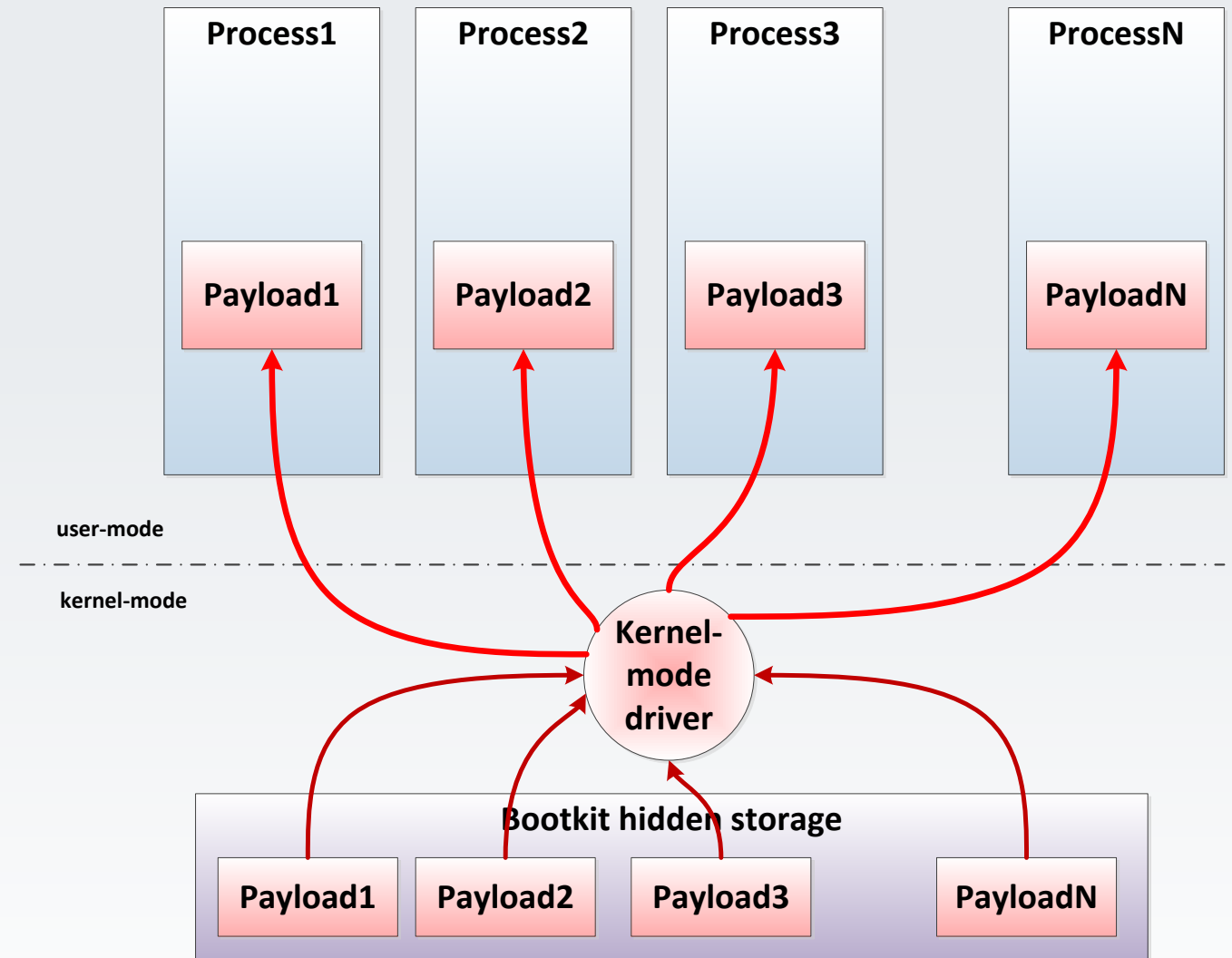
Anti-Forensics: Hidden Storage Architecture



Injecting Payload

Injection approach

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



Hidden file systems in the wild



Olmarik (TDL4)

First widely spread bootkit targeting Microsoft Windows x64 platform

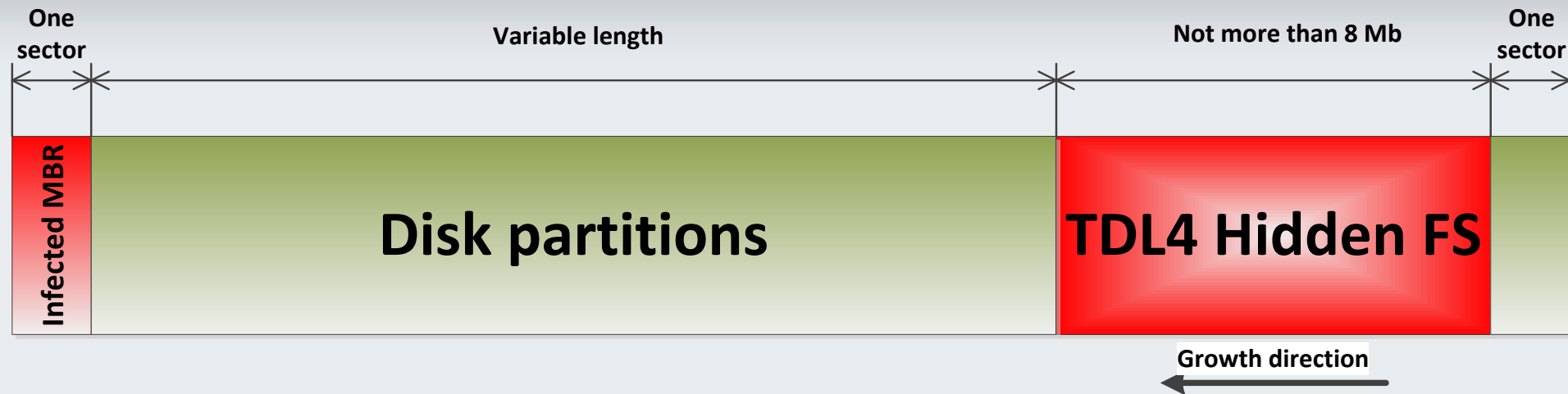
Infects MBR of the bootable hard drive to receive control before OS

Abuses Windows PE (Preinstallation Environment) boot mode to disable kernel-mode code signing policy

Keeps payload in the hidden file system

Hijacks pointer to the driver object of the lowest device object in storage device stack

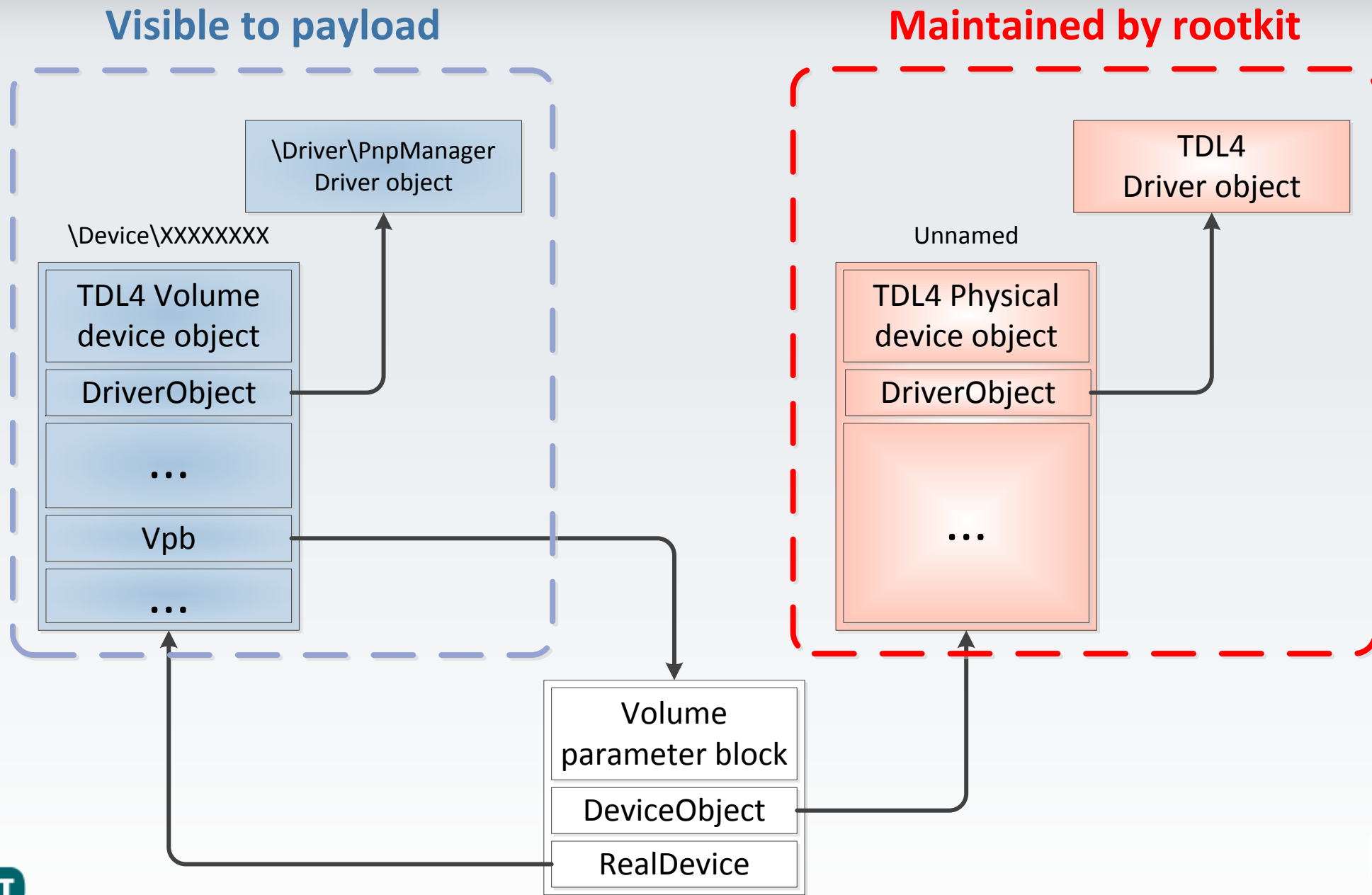
Olmarik (TDL4): Hidden File System Layout



```
typedef struct _TDL4_FS_ROOT_DIRECTORY
{
    // Signature of the block
    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;
```

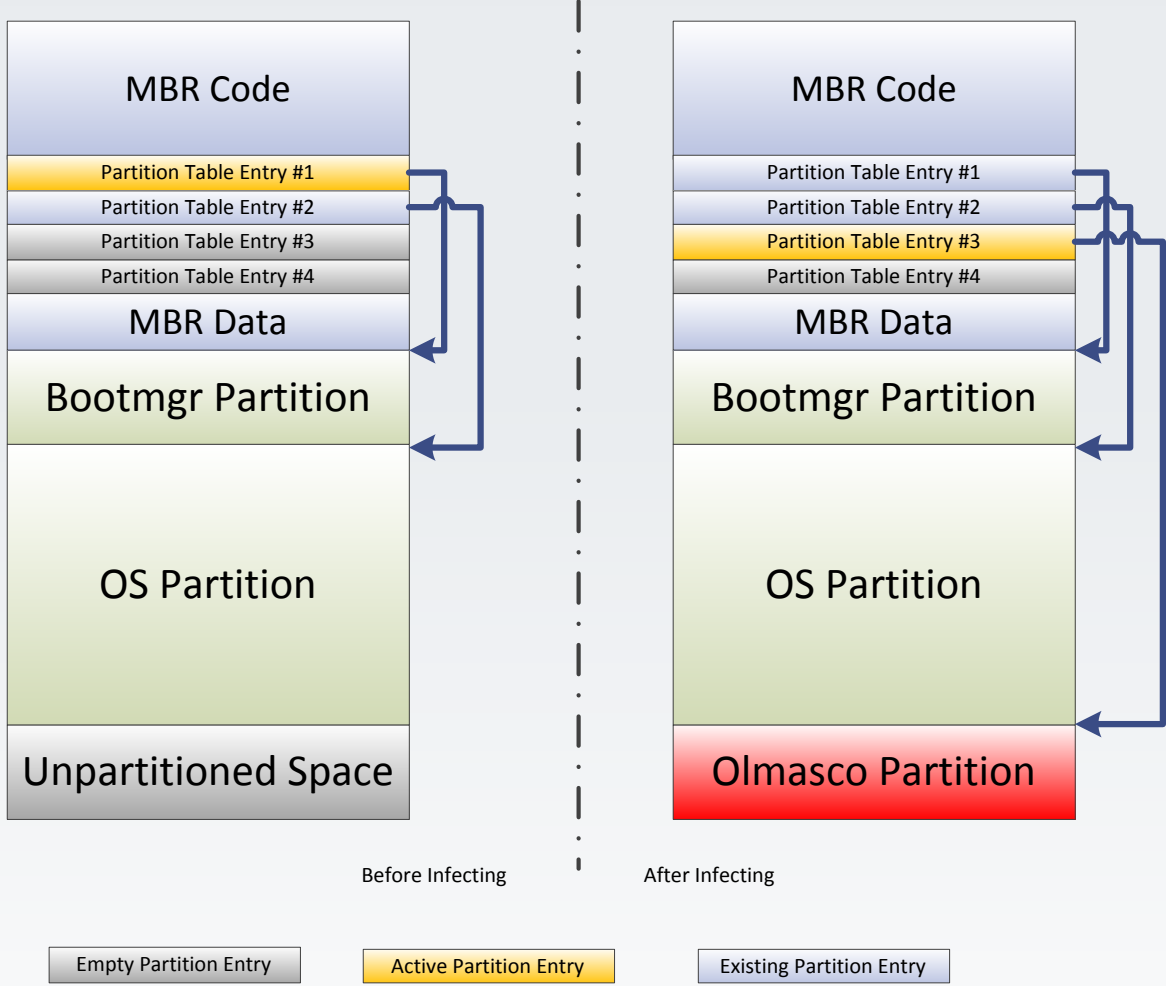
Olmarik (TDL4): Hidden Storage Implementation



Olmasco (MaxSS)

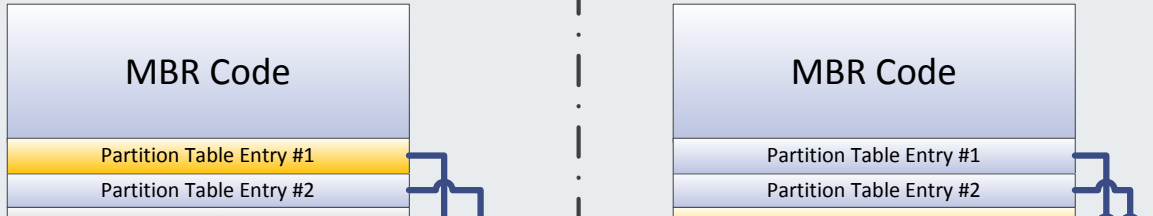
Employs the same approach for disabling kernel-mode signing policy as TDL4 bootkit

Modifies partition table of the bootable hard drive to create malicious partition and mark it active



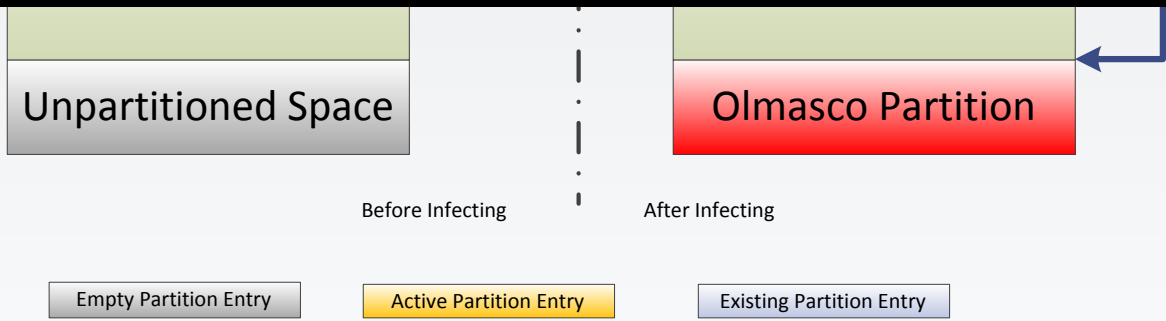
Olmasco (MaxSS)

Employs the same approach for



First partition	00212000	0C13DF07	00000800	00032000	!	---	---	---	---
Second partition (OS)	0C14DF00	FFFFFFE07	00032800	00FCC800	-	---		(---
Third partition (Olmasco), <u>Active</u>	FFFFFFE80	FFFFFFE1B	00FFF000	00000FB0	A!	-		E	---
Fourth partition (Empty)	00000000	00000000	00000000	00000000	---	---	---	---	---
			Partition LBA	Partition Size					

Modifies partition table of the bootable hard drive to create malicious partition and mark it active



Olmasco (MaxSS): Hidden File System Layout

```
typedef struct _OLMASCO_FS_ROOT_DIRECTORY
{
    // Signature of the block
    // DC - root directory
    DWORD Signature;
    // Set to zero
    DWORD Reserved1;
    // Set to zero
    DWORD Reserved2;
    // Set to zero
    DWORD Reserved3;
    // Size of the file system cluster
    DWORD ClusterSize;
    // Size of file table in clusters
    DWORD SizeOfSysTableInClusters;
    // Size of file table in bytes
    DWORD SizeOfSysTableInBytes;
    // Checksum of file table
    DWORD SysTableCRC32;
    // Array of entries corresponding to files in FS
    OLMASCO_FS_FILE_ENTRY FileTable[];
}OLMASCO_FS_ROOT_DIRECTORY, *POLMASCO_FS_ROOT_DIRECTORY;
```

```
typedef struct _OLMASCO_FS_FILE_ENTRY
{
    // File name - null terminated string
    char FileName[16];
    // Offset from beginning of the file system to file
    DWORD OffsetInClusters;
    // Size of the file in clusters
    DWORD SizeInClusters;
    // Size of the file in bytes
    DWORD SizeInBytes;
    // Checksum
    DWORD Crc32;
}OLMASCO_FS_FILE_ENTRY, *POLMASCO_FS_FILE_ENTRY;
```

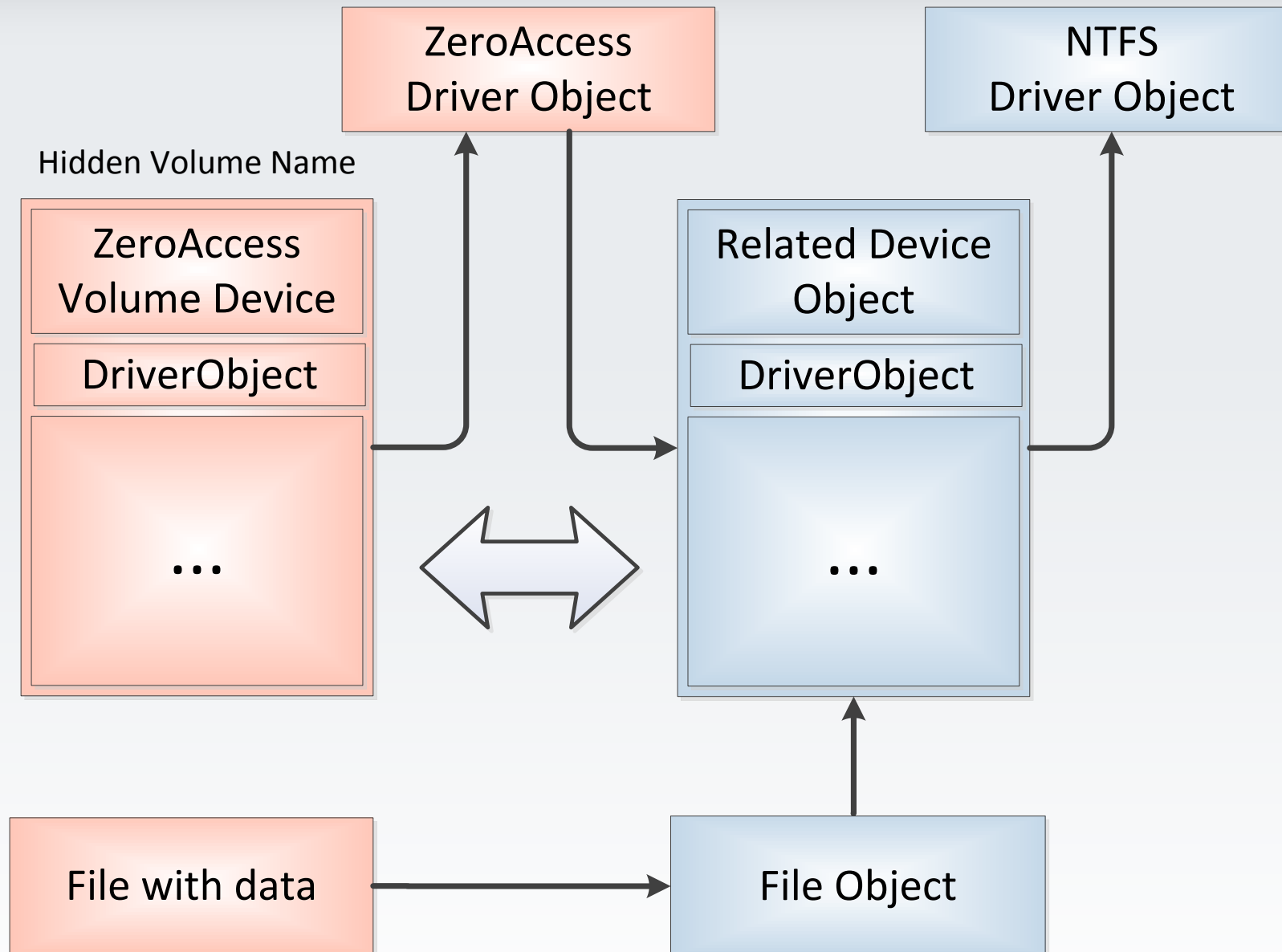
Sirefef (ZeroAccess)

Mounts a file containing payload & configuration data as NTFS volume with transparent encryption

VS.

Keeps payload & configuration data encrypted in “*C:\windows\\$NtUninstallKB35373\$*” directory

ZeroAccess: Hidden Storage Implementation



Rovnix (Cidox)

First known bootkit which infects VBR (Volume Boot Record) with polymorphic malicious bootstrap code

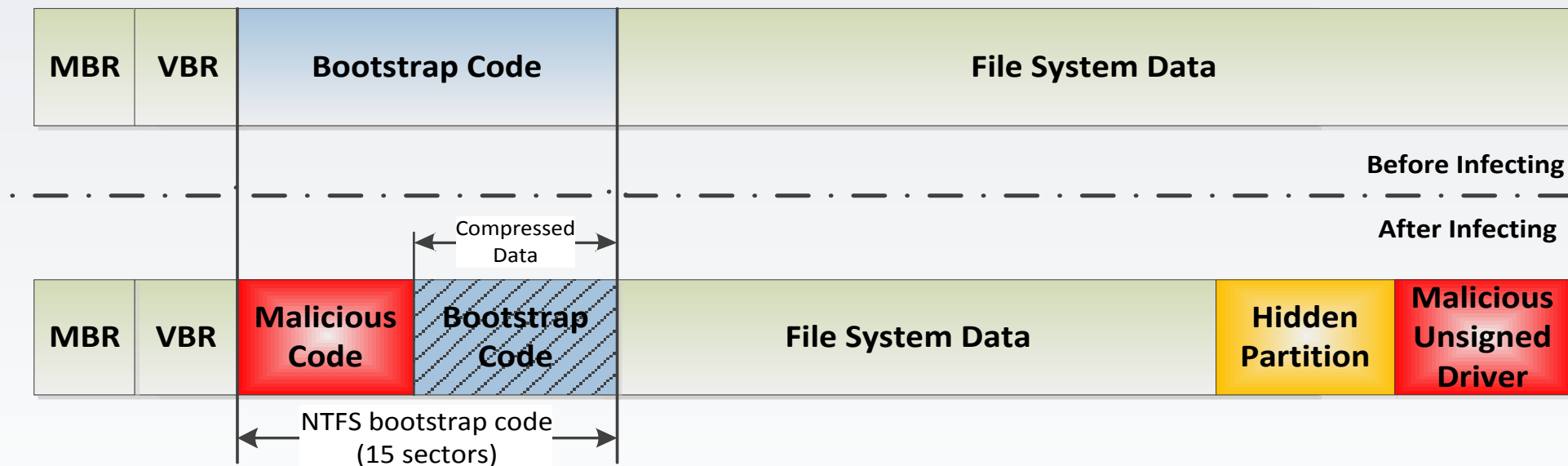
It utilizes debugging facilities of the hardware (debugging registers) to persists among processor execution mode switching and set up hooks

Rovnix bootkit employs modification of FAT16 for hidden partition

Rovnix bootkit was used in Carberp banking trojan

Rovnix (Cidox): Hidden Storage Layout

- Hidden partition & kernel-mode driver are written either:
 - ✓ before first partition on the disk – if there are more than 2,000 (1 MB) free sectors
 - ✓ otherwise, at the end of the hard drive



Goblin (XPAJ)

Employs advanced bootkit techniques to load unsigned malicious kernel-mode driver on 64-bit version of Windows OS

Combines bootkit techniques present in Olmarik (TDL4) & Rovnix

Bypasses PatchGuard by means of modifying kernel before integrity enforcement service is started

Goblin hidden implements hidden file system in a similar way to Olmarik (TDL4)

Hidden File Systems Comparison

functionality	Olmarik (TDL4)	Sirefef (ZeroAccess)	Rovnix (Cidox)	Goblin (XPAJ)	Olmasco (MaxSS)
MBR modification	☑	☑	☒	☑	☑
VBR modification	☒	☒	☑	☒	☒
Hidden file system type	Custom	NTFS	FAT16 modification	Custom (TDL4 based)	Custom
encryption algorithm	XOR/RC4	RC4	Custom (XOR+ROL)	☒	RC6 modification
compression algorithm	☒	☒	aPlib	aPlib	☒

HiddenFsReader forensic tool



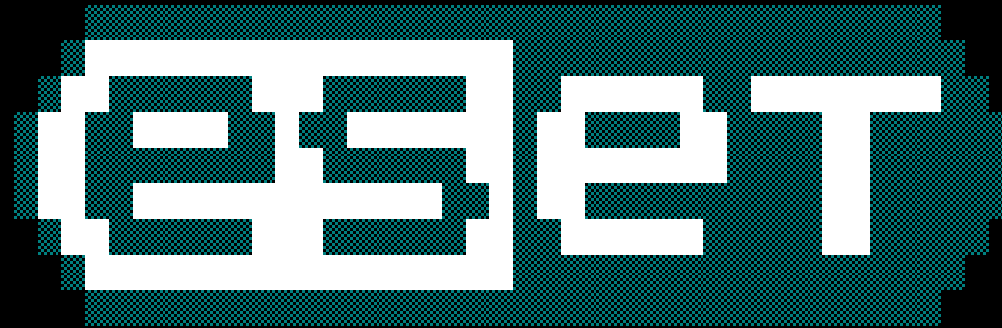
HiddenFsReader

HiddenFsReader: The evolution of TdFsReader

- Supports the following hidden file systems:
 - ✓ TDL3 and modifications
 - ✓ TDL4 and modifications
 - ✓ Olmasco
 - ✓ Rovnix.A
 - ✓ PCKE
 - ✓ Sirefef (ZeroAccess)
 - ✓ Goblin (XPAJ)
 - ✓ Flame (dump decrypted resource section)

HiddenFsReader - free public tool

Hidden File System Reader



ESET Hidden File System Reader

1.0.2.2 beta <Sep 20 2012 13:07:27>

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

Processing... Please wait.

"Rovnix.b_Driver" file system found:

- payload.sys

md5: BC6D411047E078DF3BA24FEC80645556

- vbr

md5: 446E4C3EA59D2FB1EBB7B9A4FF2D7244

File system(s) successfully exported!

Hidden File System Reader



ESET Hidden File System Reader

1.0.2.2 beta <Sep 20 2012 13:07:27>

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

Processing... Please wait.

"XPAJ.B" file system found:

"Rovnix.b_1

- payload.s

- vbr

- noname_0

- noname_1

- noname_2

- noname_3

md5: D47ADC36DE840C145E45BE529FD6AC60

md5: 9EDD2D4A3E8F1055BF189E45A4B9AA55

md5: 8C28EFB8E179C177C0D2ED9895F7B900

md5: 2E4406035F0F23B566D5FDBB437E6F66

FEC80645556

7A4FF2D7244

File system(s) successfully exported!



Hidden File System Reader



ESET Hidden File System Reader

1.0.2.2 beta <Sep 20 2012 13:07:27>

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

Processing... Please wait.

"TDL4_PH" file system found:

- phm	md5: DF09785A37B0197496A1C45A8292FAA6
- phs	md5: 7591CFFC80CE754D591EE5CE5C260786
- ph.dll	md5: 68D5C59C4E554A04514E157C31F38EF9
- phdata	md5: 205E4B7ACF1DE985BF25B7D7F3032040
- phld	md5: 53FC3109DB25895A1EA379040D4F43D3
- phln	md5: F33A3C5FA8C6E16PCE3F0E321471C7E3
- phd	md5: 2493C1EB48F036F35965034BD2847393

File system(s) successfully exported!

"Rovnix.l

- payload

- vbr

80645556

FF2D7244



2012

DALLAS

26 - 28 September 2012

HiddenFsReader: using scenarios

Incident response

- Dump and decrypt hidden file system
- Dump MBR/VBR or any range of sectors

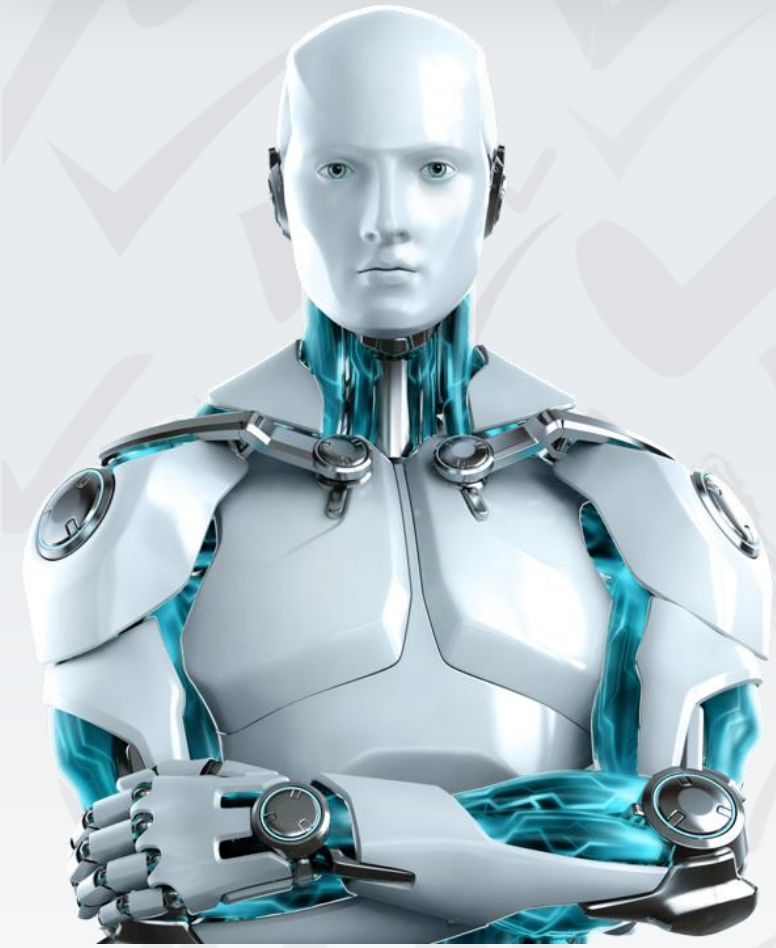
Threat analysis and monitoring

- Quick dump of payload from hidden file system
- Developing botnet monitoring tracker

LIVE DEMO



HiddenFsReader: Free public forensic tool



Try to use it right now ;)

eset.com/download/utilities/detail/family/173/

Download



eset



2012
DALLAS
26 - 28 September 2012

Conclusion

- Implementing techniques to counteracting forensic analysis is a common feature of complex threats
- The most well-known threats implementing hidden file systems:
 - ✓ Olmarik (TDL4)
 - ✓ Olamsco (MaxSS)
 - ✓ Rovnix (Cidox)
 - ✓ Goblin (XPAJ)
 - ✓ Sirefef (ZeroAccess)
- HiddenFsReader is a tool that makes it possible to retrieve contents of the most widely spread hidden file systems

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**

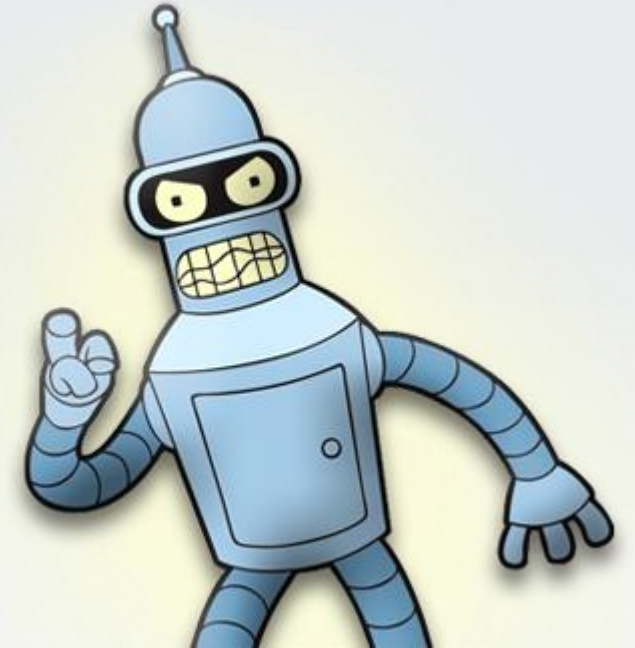
<http://blog.eset.com/2012/01/03/bootkit-threat-evolution-in-2011-2>

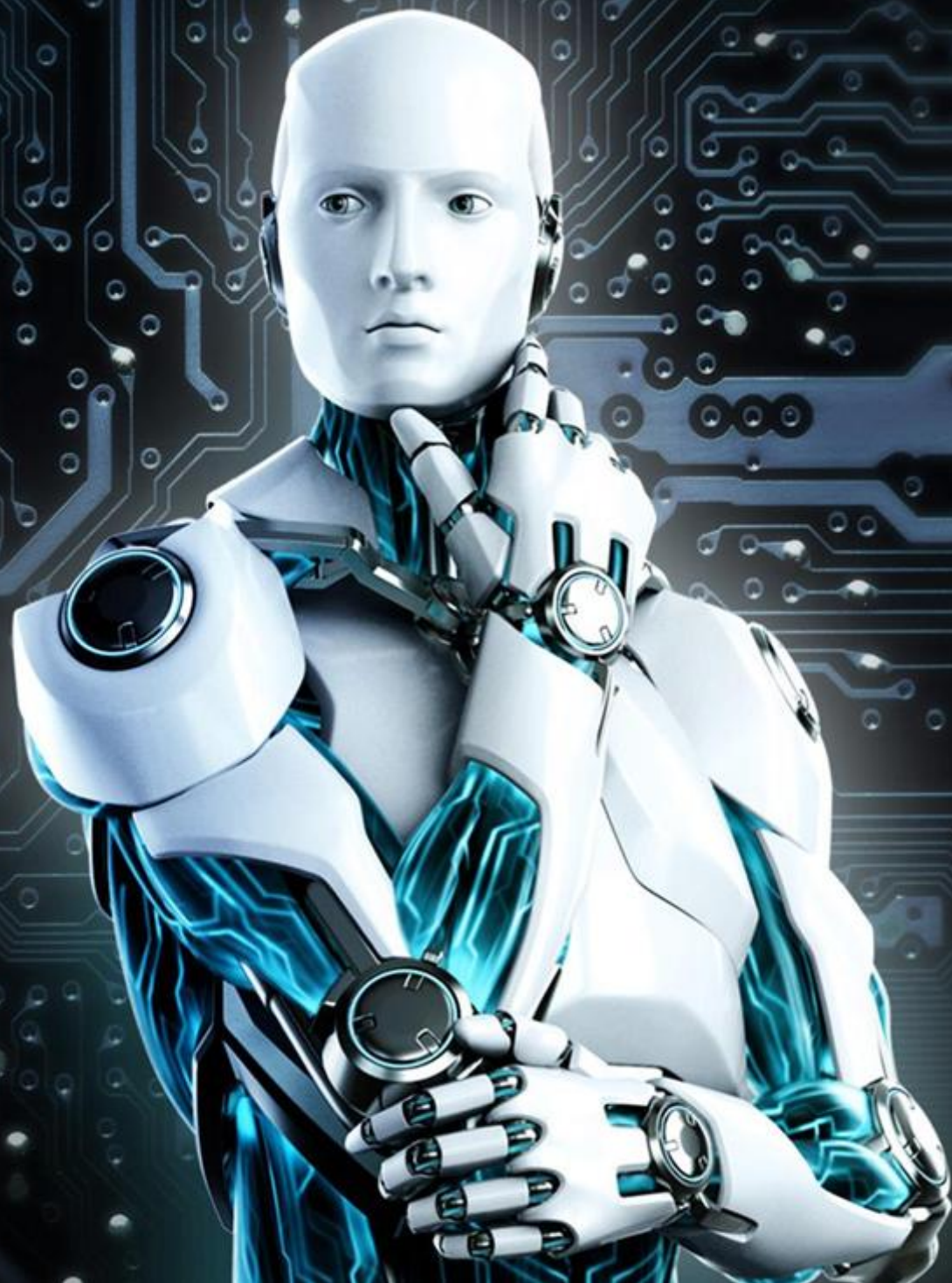
✓ **Rovnix.D: the code injection story**

<http://blog.eset.com/2012/07/27/rovnix-d-the-code-injection-story>

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

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





Thank you for your attention!

Aleksandr Matrosov
matrosov@eset.sk
@matrosov

Eugene Rodionov
rodionov@eset.sk
@vxradius



2012
DALLAS 
26 - 28 September 2012