Hide’n Seek Revisited - Full Stealth Is Back

Kimmo Kasslin / 6th October 2005

F-SECURE®

BE SURE.

Introduction

Kimmo Kasslin

- Email: kimmo.kasslin@f-secure.com
- Researcher in F-Secure Security Labs
- Research is focused in the area of Windows rootkits and modern stealth malware
- Member of the team responsible for F-Secure BlackLight technology

Contents

Windows Rootkits

Stealth Malware

Hiding Techniques

Hidden Object Detection

Anti-Detection Techniques

Future Challenges

Conclusions

Windows Rootkits

Allow intruders to maintain access to the system

Operate in user mode or in kernel mode

Try to avoid detection by hiding e.g.

- Processes
- Files
- Registry keys
- Network connections

Stealth Malware – Past

In the era of DOS, stealth viruses were common

- 1986 – Brain
- 1990 – Frodo

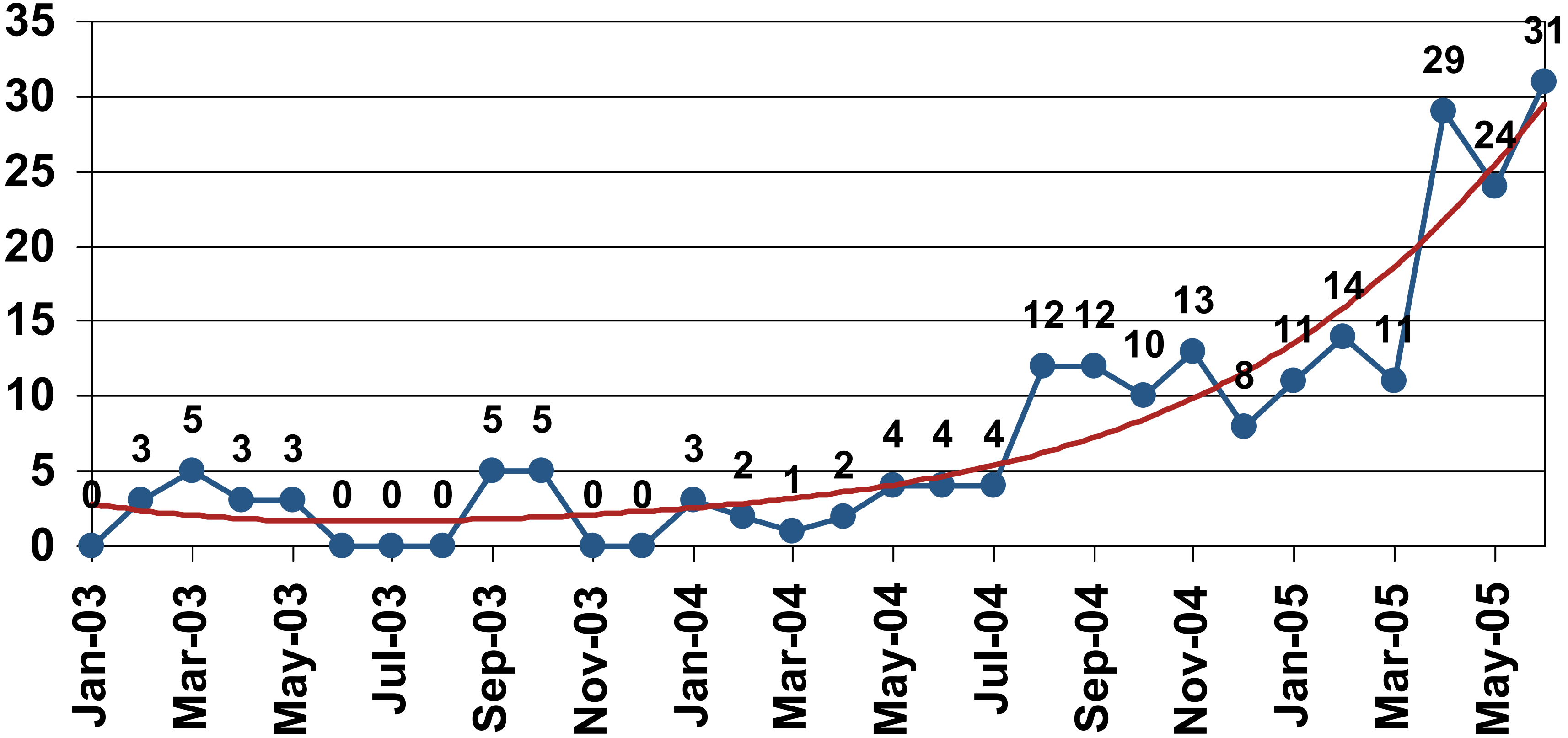
They started to disappear when Windows 95 became the dominant OS

Since then, their numbers remained low

- 1997 – Cabanas, first Windows NT virus

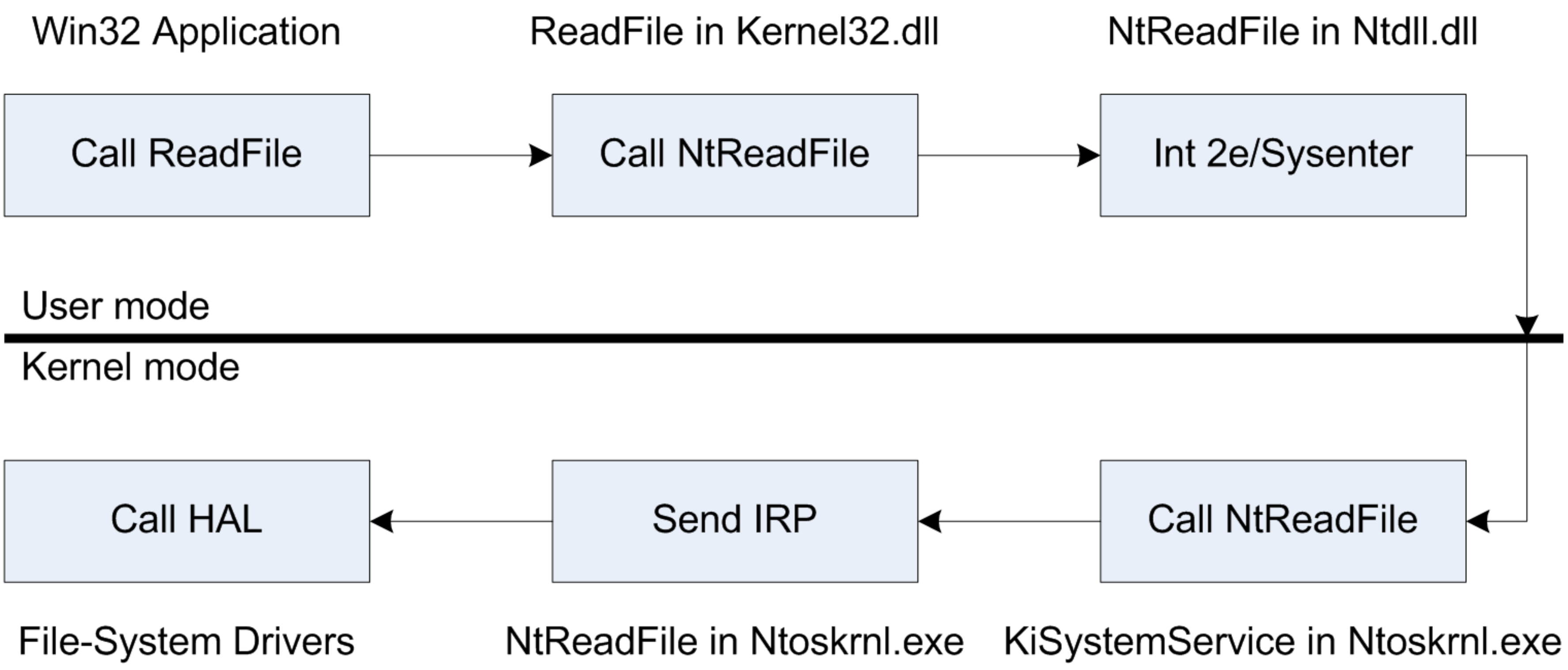
Stealth Malware - Present

Today, we are seeing increasing numbers of stealth malware



Source: Monastyrsky A.; Sapronov K.; Mashevsky Y. (2005). Kaspersky Lab

Hiding Techniques – Execution Path



Win32 Application

Call ReadFile

ReadFile in Kernel32.dll

Call NtReadFile

NtReadFile in Ntdll.dll

Int 2e/Sysenter

User mode

Kernel mode

Call NtReadFile

KiSystemService in Ntoskrnl.exe

Send IRP

NtReadFile in Ntoskrnl.exe

Call HAL

File-System Drivers

Hiding Techniques - Summary

Objects can be hidden through several means

- Inline hooking
- Import Address Table hooking
- Export Address Table hooking
- System Service Table hooking
- Interrupt Table hooking
- I/O Request Packet hooking
- Filter drivers
- Kernel object manipulation

Hacker Defender

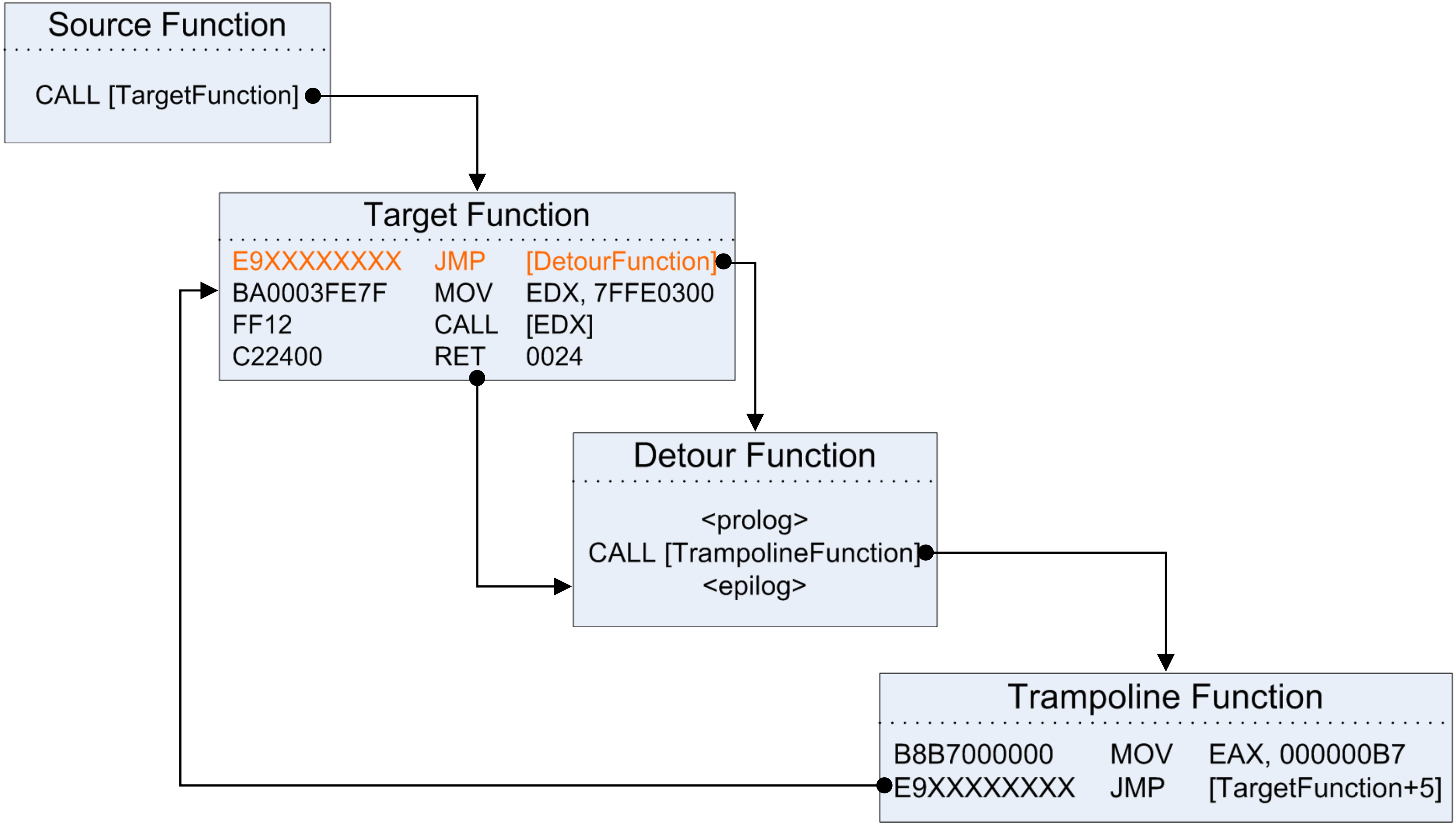
One of the most popular rootkits in the wild

- User-mode rootkit
- Feature rich
- Very stable and portable
- Under active development

Modifies the execution path of several Native and Windows API functions

- Inline hooking through direct memory patching

Inline Hooking à la Detours



Source Function

CALL [TargetFunction]

Target Function

```
E9XXXXXXXX   JMP    [DetourFunction]
BA0003FE7F   MOV    EDX, 7FFE0300
FF12         CALL   [EDX]
C22400       RET    0024
```

Detour Function

```
<prolog>
CALL [TrampolineFunction]
<epilog>
```

Trampoline Function

```
B8B7000000   MOV    EAX, 000000B7
E9XXXXXXXX   JMP    [TargetFunction+5]
```

Demo 1

F-SECURE®

BE SURE.

Hacker Defender - Hook Installation

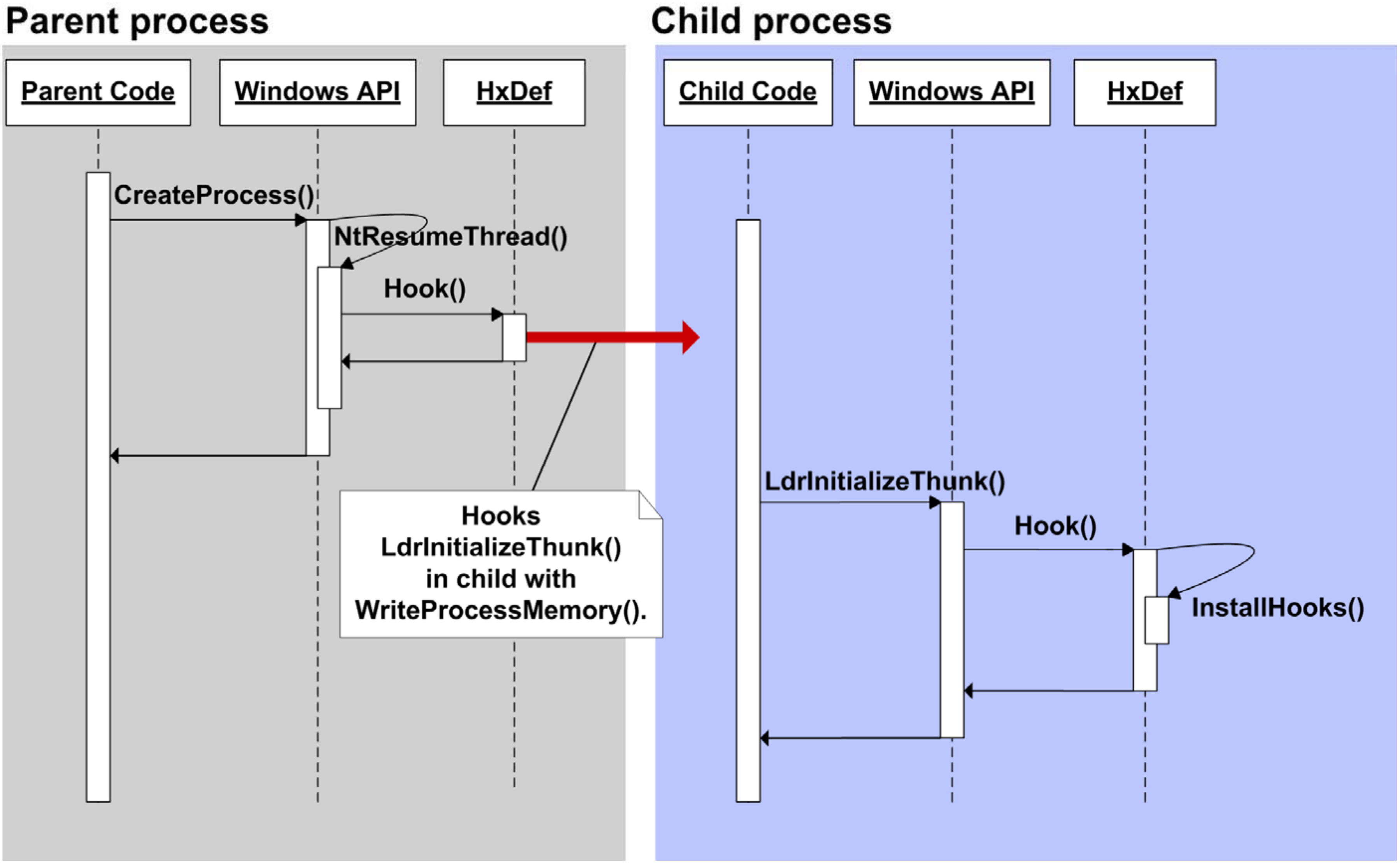
Installs user-mode hooks into every process

- WriteProcessMemory API function
- Requires debug privileges

New processes and dynamically loaded DLLs are patched through special hooks

- Ntdll!NtResumeThread of parent process
- Ntdll!LdrInitializeThunk of child process
- Ntdll!LdrLoadDll of child process

Hacker Defender – Hook Installation



Hidden Object Detection

One possible approach – “Cross-View Diff”

- Tainted view
- Trusted view

Challenges with this approach

- Collecting data for the trusted view
- Today, also collecting data for the tainted view

F-Secure BlackLight

- Stand-alone beta was released in March 2005
- Integrated into F-Secure Internet Security 2006

Anti-Detection Techniques

Successful detection requires that there is a difference between the two views

If the detector process can be identified by the rootkit, do not hide from it

- Filename
- Version information in image resources

Other approach is to only hide data from processes normally used by users

- Explorer, Task Manager, Process Explorer

Demo 2

F-SECURE®

BE SURE.

Golden Hacker Defender

Identifies detectors through binary signatures

- Our sample contains around 40 signatures

The signature is checked against the memory resident image when the first hook is executed

- Detection possible even if the binary is packed

If a match is found, a bit mask is set that defines which hooks will be disabled

In addition, modifies code in some images

- Defeats most of current anti-anti-detection measures

Demo 3

Future Challenges

Rootkits that do not need processes, files or registry keys

- ByShell

Rootkits that hide themselves even from kernel-mode memory scanning

- Shadow Walker

With kernel-mode rootkits only the imagination and skills of the developer are the limits

Conclusions

Stealth malware is back and kicking

- Hiding is based on rootkit techniques
- The most advanced techniques are still quite rare

Generic rootkit detection is feasible

- Cross-view diff based detectors can find majority of present stealth malware
- False positives are rare

Rootkits are evolving rapidly and will find ways to bypass detectors

- Direct attacks against the detectors

THANK YOU – QUESTIONS?

kimmo.kasslin@f-secure.com

F-SECURE®

BE SURE.