

Malware Profiling and Rootkit Detection on Windows

Matt Conover

Email: shok1234@msn.com



 X'con 2005

Agenda

- ◆ Basics about Rootkits
- ◆ Current rootkit detection
- ◆ Rootkit techniques
- ◆ A better way...



Introduction

- ❖ A constant battle—cat and mouse game
- ❖ The detection/profiling mechanisms have changed little over the years
- ❖ Malware/Rootkits are increasingly sophisticated in evasion



Rootkit Introduction

- ◆ Rootkit first appeared on Windows in 1999 (NTRootkit, Hoglund):
 - ◆ Different agenda than viruses
 - ◆ Non-destructive information gatherers
 - ◆ Usually running in the kernel (easier to hide)



Rootkit Detection

- ❖ At the beginning of this year, there was almost no commercial products
- ❖ Rootkit detection has suddenly become popular. F-Secure, Microsoft, etc. have all released products.
- ❖ Rootkit products will not be very useful unless they adapt as quickly as the rootkits
- ❖ Rootkit evasion techniques are advancing much faster than rootkit detection



Rootkit Detection

- ◆ Three current detection mechanisms:
 - ◆ Anti-virus software approach
 - ◆ HIPS (Host Intrusion Prevention Systems)
 - ◆ Execution Path Analysis (EPA)
 - ◆ The newcomer: Differential testing



Rootkit Detection Anti-Virus

- ❖ Very effective at preventing use of known rootkits
- ❖ New signatures are made as new variants and rootkits come out
- ❖ Detects the rootkit's fingerprint before it has a chance to run



Rootkit Detection

X'con 2005

Problems with Anti-Virus

- ◆ Few rootkits are observed in the wild
 - ◆ This gives them a low priority
- ◆ Rootkits are evasive and non-destructive
 - ◆ Few samples of rootkits are sent to AV companies
- ◆ Too late...
 - ◆ Rootkits will just unhook antivirus (usually a filter driver over the file system)
 - ◆ Then when an AV definition comes out, it is too late J



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Rootkit Detection Host IPS

- ◆ Two layers of defense
- ◆ Tries to prevent exploitation of the machine (stop buffer overflows, RLIBC attacks, etc.)
- ◆ If hackers get past that defense, then try to block the hacker from getting into the kernel



Rootkit Detection

Problems with Host IPS

- ◆ Many weaknesses outlined in a Phrack 62 (Butler)
- ◆ API hooks are easy to evade
- ◆ Most HIPS cover only those that are likely to be used by an exploit
- ◆ Hard to cover all ways a rootkit can be introduced:
 - ◆ Crazylord evaded a rootkit detector by using a symbolic link `\Device\PhysicalMemory`
 - ◆ Defenseless against use of new kernel privilege escalation vulnerabilities



Rootkit Detection

Execution Path Analysis (EPA)

- ◆ Discussed at BlackHat Las Vegas 2003 by Joanna Rutkowski
 - ◆ An old idea now applied specifically to rootkits
- ◆ Uses instruction trapping to profile system calls
 - ◆ Goes through a learning period when the system is known to be clean
 - ◆ Remembers the instruction counts or code paths of the system calls
- ◆ Detects rootkit when the execution path of a system call differs



Rootkit Detection

Problems with the EPA

- ❖ Large performance degradation tracing through all system calls
- ❖ Difficult to implement correctly (many ways to disable):
 - ❖ Overwriting the trap handler in the IDT
 - ❖ Overwriting EFLAGS.TF in the TSS
 - ❖ Overwriting EFLAGS.TF via POPF



Rootkit Detection Differential

- ◆ Query same information from top locations:
 - ◆ First use user-mode APIs
 - ◆ Then use low-level methods (looking at the registry file, NTFS directly, etc.)
- ◆ If these differ, something is hiding information



Rootkit Detection

Problems with Differential

X'con 2005

- ❖ Was quickly defeated (see rootkit.com)
- ❖ They are easy targets for rootkits
- ❖ These methods are too basic
- ❖ Rootkits can make special cases to handle these tools



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Rootkit Technologies Introduction

- ◆ User-mode rootkits (not covered here)
 - ◆ Hide in other processes
 - ◆ Keyboard sniffing
 - ◆ May be "diskless" (AV cannot detect)
 - ◆ Metasploit, CANVAS, and CORE IMPACT are all diskless
 - ◆ Won't be discussed in this presentation

- ◆ Kernel-mode rootkits
 - ◆ Coming up next...



Rootkit Technologies Introduction

- ◆ First, get into kernel-mode
- ◆ Second, hook into kernel
- ◆ Third, try to become permanent



Rootkit Technologies

Getting into Kernel #1

X'con 2005

- ◆ Using ZwSetSystemInformation or ZwLoadDriver
 - ◆ Enable SeLoadDriverPrivilege
 - ◆ The problem is that it will be pageable (as Hوجلund/Butler note)
 - ◆ But there is a magic trick: MmResetDriverPaging J
- ◆ Service Control Manager (the normal way)
 - ◆ No special tricks required
 - ◆ This will require creating a registry key
- ◆ Both require a physical file be present
- ◆ Makes the rootkit an easy target for antivirus detection



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Rootkit Technologies

Getting into Kernel #2

- ◆ Use a kernel-mode exploit.. some examples:
 - ◆ LPC (local): 原创 (eyas)
 - ◆ Norton Antivirus (local): s.k. chong
 - ◆ SymDNS (remote): barnaby jack



Rootkit Technologies

Getting into Kernel #3

X'con 2005

- ❖ Install Ring3->Ring0 call gate from user mode
 - ❖ See paper by crazylord
 - ❖ No disk access (AV can't detect)
 - ❖ Less complicated than kernel-mode exploits
 - ❖ Modify x86 GDT directly from user mode
 - ❖ May not work for newer versions of Windows



Rootkit Technologies

Hooking into the Kernel

X'con 2005

- ◆ Once your code is running the kernel, now what?
- ◆ Hooking system call table
 - ◆ Used to either add new system calls or hide information like files, registry keys, etc.
- ◆ Hooking interrupt handlers
- ◆ Manipulate page tables entries (executable, no readable)
- ◆ Hooking driver dispatch tables
- ◆ Add filter drivers



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Better Method...

Introduction

- ❖ Can be used to detect rootkits
- ❖ Can be used to monitor system activity (helpful to profile malware)



Better Method...

X'con 2005

Windows Executive Objects

- ❖ Windows uses “executive objects”
 - ❖ Controlled by an Object Manager
 - ❖ Handles are all indirect references to objects
 - ❖ Everything is an object



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Better Method...

X'con 2005

Windows Executive Objects

- ◇ Memory sections
- ◇ LPC ports
- ◇ I/O completion
- ◇ WMI
- ◇ Desktops
- ◇ Mutexes
- ◇ Events
- ◇ Semaphores
- ◇ I/O Controllers
- ◇ Files
- ◇ Registry keys
- ◇ Devices
- ◇ Drivers
- ◇ Processes
- ◇ Threads
- ◇ Jobs
- ◇ Sockets
- ◇ Security tokens

These are all objects!



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Better Method...

Windows Executive Objects

- ◆ How does the Object Manager track so many types of objects?
- ◆ It doesn't "memorize" all these executive object types
- ◆ Instead, executive object types are registered dynamically
- ◆ There are set of callbacks for each object type, and it is responsible for opening, creating, securing, and closing that object type



Better Method...

Example

 X'con 2005

- ◆ During system initialization, IoInitSystem() registers the FILE_OBJECT type
- ◆ Later you call NtCreateFile() to create a new file:
- ◆ This calls ObCreateObject(name, FILE_OBJECT)
- ◆ The Object Manager calls the Open callback with the mode set to Create routine registered for the FILE_OBJECT
- ◆ If the Open callback returns successfully, then the handle is returned to NtCreateFile



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Better Method...

- ◆ We can replace the callbacks for all object types we're interested in
- ◆ If we're interested in finding out every time a process or file is opened:
 - ◆ Find the FILE_OBJECT object type and replace the Open callback
 - ◆ Find the EPROCESS object type and replace the Open callback



Better Method...

- ◆ In the callback, we analyze the event and then call the original callback
- ◆ If we're just profiling:
 - ◆ We record the event and allow it to pass
- ◆ If we're doing rootkit detection:
 - ◆ We check if there are any matching signatures
 - ◆ If a signature matches, we execute the signature action (e.g., report, block, etc.)



Better Method...



Benefits

- ◆ Saves on performance big time
 - ◆ Can be isolated to specific object types, specific processes, or just the kernel
 - ◆ Attempts to open an object that doesn't exist don't even reach the Open callback (thus no overhead)
 - ◆ Attempts to create an object when the caller doesn't have adequate permission doesn't even reach the Open callback (thus no overhead)
- ◆ New possibilities!
 - ◆ Able to monitor almost all aspects of the systems behavior
- ◆ Remember, almost everything is an object!



Better Method...

How To

 X'con 2005

- ◆ If we want to profile malware:
 - ◆ Start the malware in a suspended state
 - ◆ Monitor all object types
 - ◆ Apply it only to the malware process
- ◆ If we want to detect rootkits:
 - ◆ Process signatures and only monitor the object types that has a matching signature
 - ◆ Apply it only to kernel mode (e.g., ignore user-mode processes)



XFOCUS TEAM

BEIJING.CHINA

2002-2005



Profiling: Demo



Detecting Driver Loading

- ◆ Creating section in kernel with SEC_IMAGE flag



Detecting Filters

- ◆ Detect creation of I/O completion port from kernel
- ◆ Enumerate DEVICE_OBJECTs



Detecting Dispatch Hooks

- ◆ Enumerate DRIVER_OBJECTs



Detecting Hidden Rootkits

- ◆ Some of the things rootkits do make them easier to detect J

Return address into non-readable page

Return address into non-paged memory pool



Removing a Rootkit...

- ❖ Still experimental...
- ❖ Creates instability to remove rootkit (unknown hooks)
- ❖ Replace all rootkit code with INT 3 (breakpoint)
- ❖ Add INT3 handler
- ❖ If the return address is to a suspicious place, add INT 3 to that page also
- ❖ After we no longer see any new pages for a while, replace INT 3 with NOP



Self-Preservation

- ◆ Since this is a cat and mouse game, rootkits will improve to hide from this method
- ◆ We need to protect ourselves from when the rootkit authors begin specifically targeting this detection mechanism
- ◆ Thus, we need to take whatever self-preservation mechanisms we can to stay in control



Self-Preservation: Step 1

- ◆ Prevent a rootkit from being loaded in the first place
 - ◆ Disable access to
 \Device\PhysicalMemory
 - ◆ Disable driver loading methods
- ◆ Limitations:
 - ◆ The attacker will use a new kernel privilege escalation vulnerability, and get past this step



Self-Preservation: Step 2

- ◆ Prevent a rootkit from making itself permanent
 - ◆ Disable any attempt to create HKLM\SYSTEM\CurrentControlSet*\Type with type 0 or 1 (change to 4 for disabled)
 - ◆ Disable any attempt to modify an existing an HKLM\SYSTEM\CurrentControlSet*\Type
- ◆ Limitations:
 - ◆ The rootkit may physically patch hal.dll, ntoskrnl.exe, etc.
 - ◆ Be wary of accessing the registry keys through symbolic links



Self-Preservation: Step 3

- ◆ Ensure no driver except FAT/NTFS loads before us
 - ◆ Install ourselves at the beginning of the "Boot Bus Extender"
 - ◆ Prevent any changes to HKLM\SYSTEM\CurrentControlSet\GroupOrderList



Self-Preservation: Step 4

- ◆ Ensure no one changed the object type callbacks
 - ◆ Keep a thread in an infinite loop watching the hooked callbacks every few 100 milliseconds or so
 - ◆ Restore callbacks if they are changed and report an attack
 - ◆ Find out where the callbacks pointed to (this lets us know who did it)
 - ◆ If it is not a known system driver, unload it



Summary

- ◆ Presented a method of observing system behavior
 - ◆ User-mode and kernel-mode
- ◆ Presented a method to block certain behaviors
 - ◆ Signature language can be used to detect known rootkits
- ◆ Presented self-preservation methods
 - ◆ Needed if new rootkits come out that aren't recognized
- ◆ In the end, this is just a step in the cat and mouse game



Acknowledgements

- ❖ Many kung fu masters for Windows kernel-mode exploitation and rootkits:

Joanna Rutkowska, Jamie Butler, flashsky,
S.K. Chong,

Barnaby Jack, Greg Hoglund, Derek Soder,
crazylord



STKIT– Shok Toolkit J

- ◆ Remember this URL...
- ◆ Remember this URL...
- ◆ Remember this URL...
- ◆ Remember this URL...

<http://www.cybertech.net/~sh0ksh0k>

Will not be publicly announced, so you must remember
Code will be put there in the next 2 weeks



The End

- ◆ Thanks for listening J
- ◆ Send email to [shok1234 msn.com](mailto:shok1234@msn.com)

