



Windows Memory Forensics and Direct Kernel Object Manipulation

Jesse Kornblum

Outline

- Introduction
- The Kernel
- Direct Kernel Object Manipulation
- Standard DKOM
- Devious DKOM
- Better Magic
- Relations Between Kernel Objects
- Questions

Introduction

- Computer Forensics Research Guru
 - md5deep, hashdeep, fuzzy hashing (ssdeep), foremost, etc
 - AFOSI, DoJ, ManTech
- Kyrus Technology

Introduction

- Direct Kernel Object Manipulation (DKOM)
- Powerful technique for p0wning a computer
 - or crashing it
- Memory forensics should be able help us
 - but can be subverted too
- But we shall prevail

The Kernel

- The kernel must maintain lots of data
 - Processes
 - Threads
 - File handles
 - Network connections
 - Interrupts
 - Really everything on the system
- All stored in kernel data structures

How it's Supposed to Work

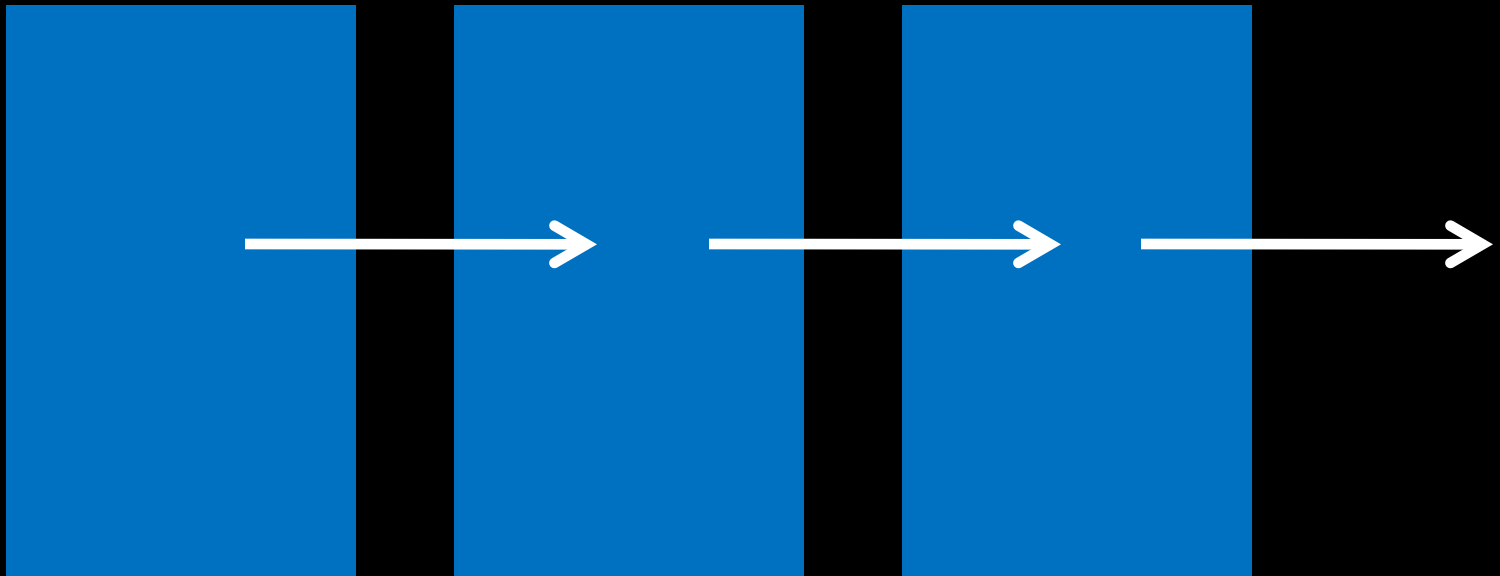
- Structures are modified by API functions
- Several different levels of API functions
 - CreateProcess
 - NtCreateProcess
 - ZwCreateProcess
 - And many more!
- These functions provide
 - Sanity checking
 - Memory allocation
 - Data initialization

Direct Kernel Object Manipulation

- Modify data structures without using API functions
- Must be done by code running in ring zero
 - Also called kernel mode
 - But not userland programs
- Can be done by drivers
 - This is why drivers can cause crashes
- Code injected into the kernel process

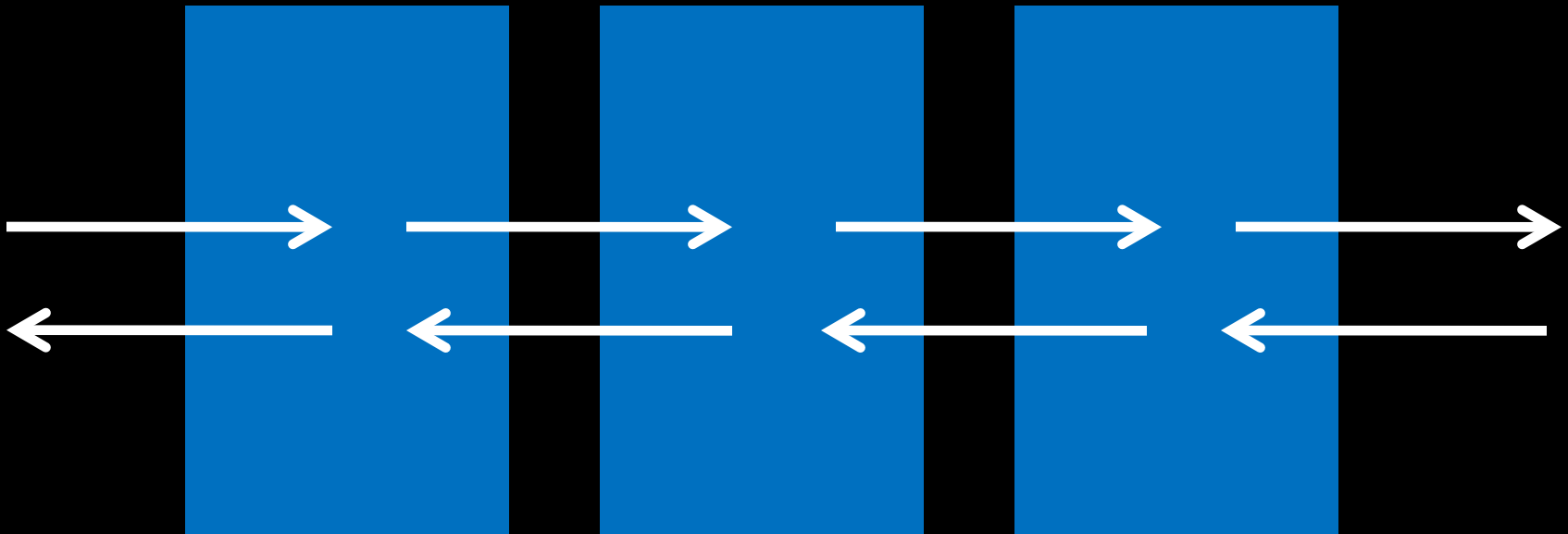
The Kernel

- Lots of lists
- Linked lists
- Each item points to the next item in the list

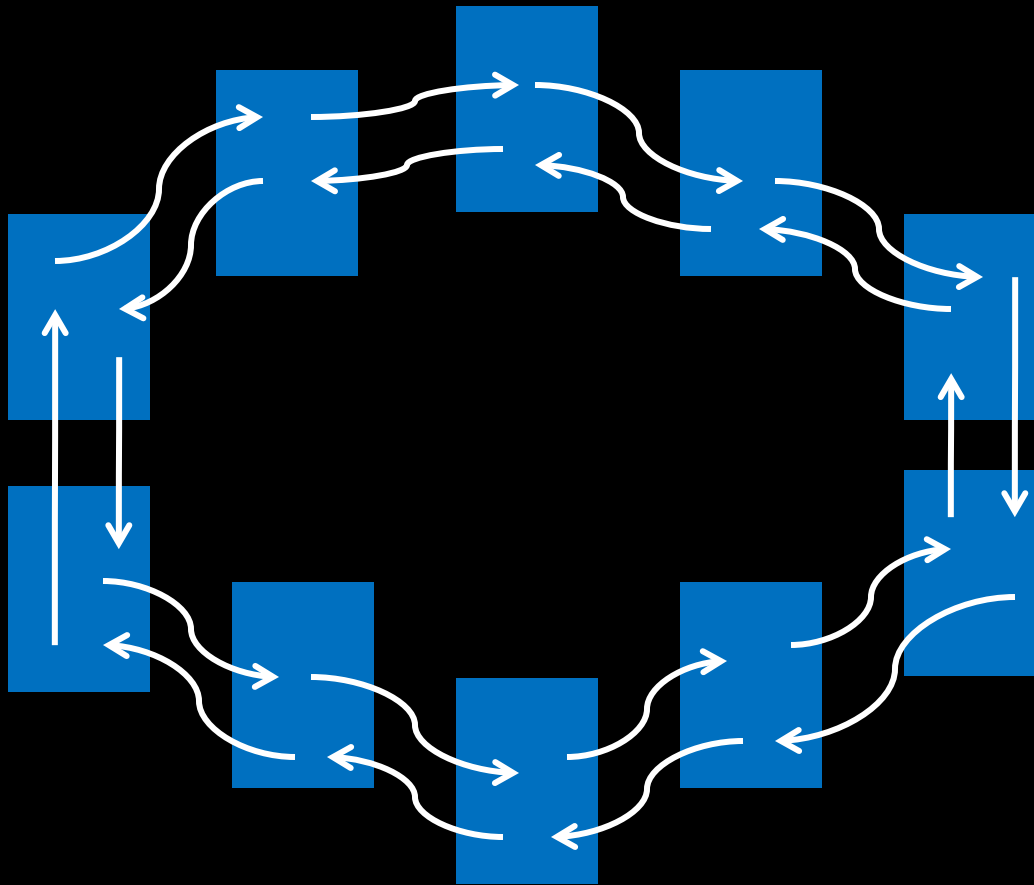


The Kernel

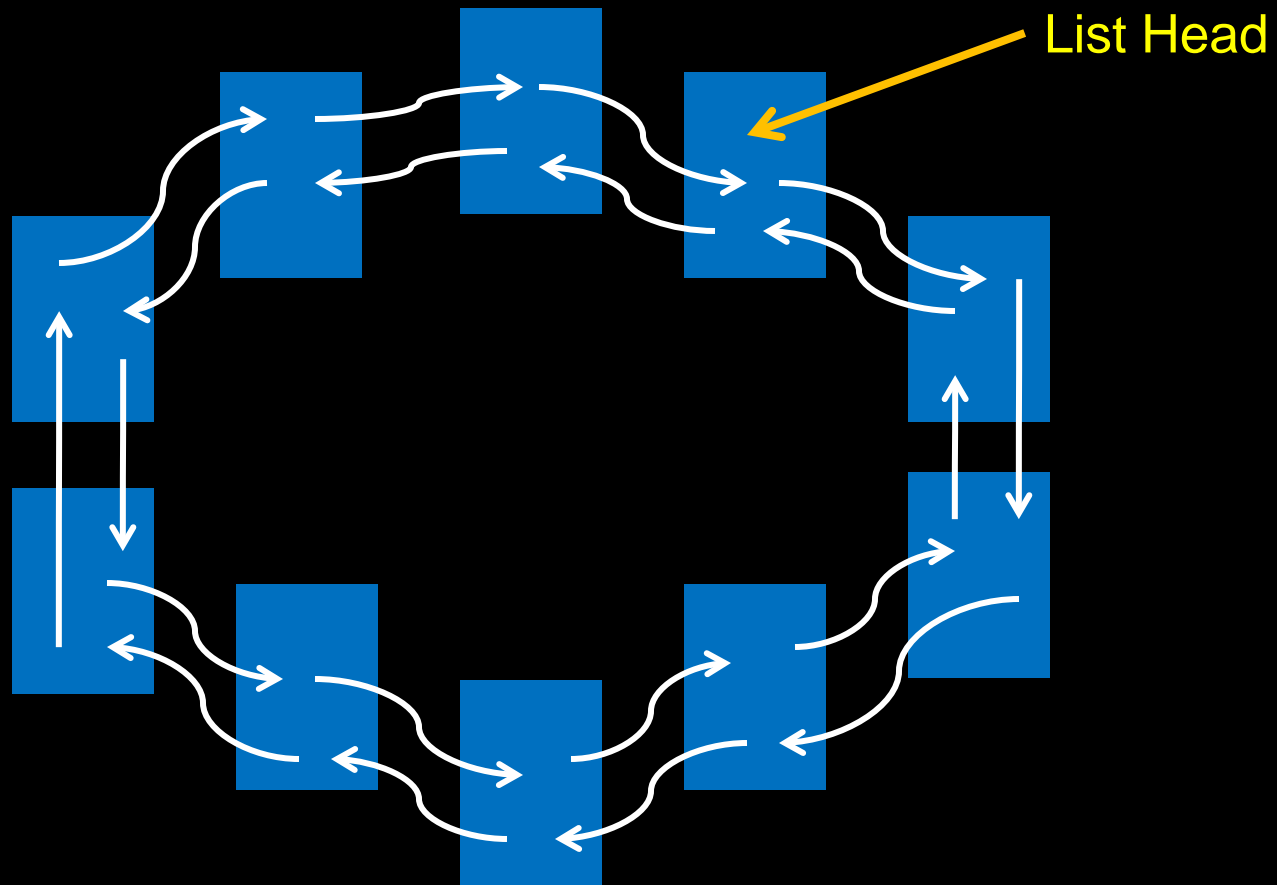
- Doubly linked lists
- Each item points to the next and previous items in the list



How it's Supposed to Work

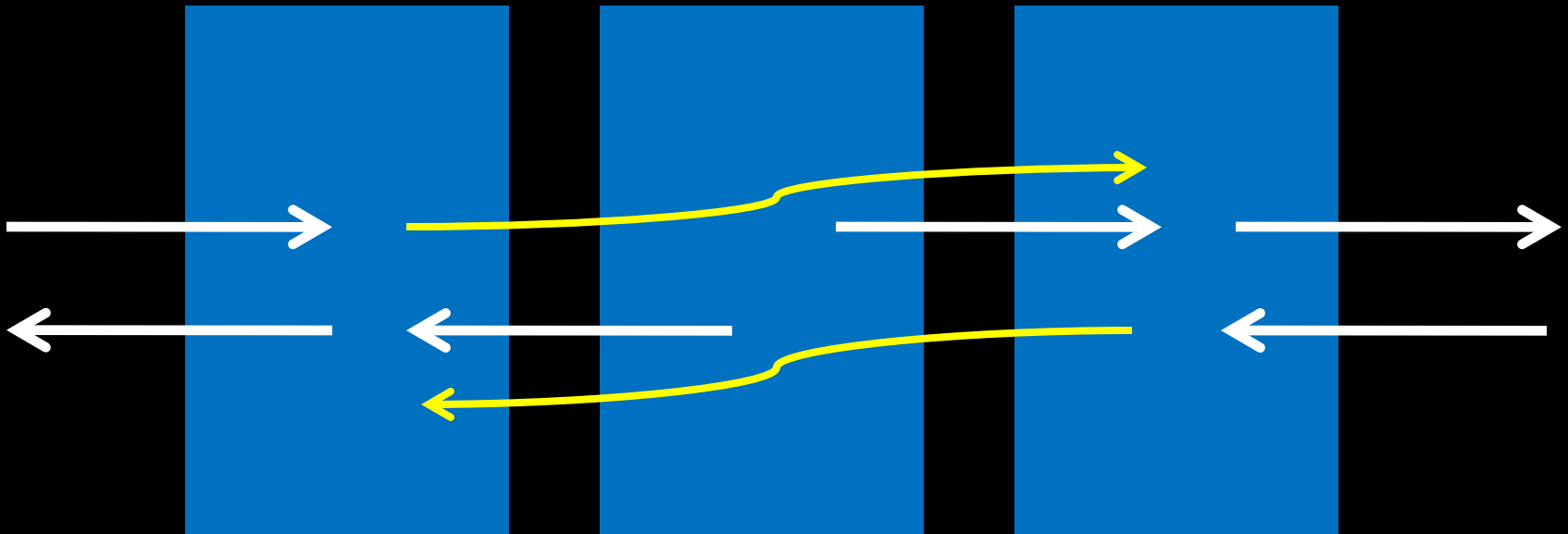


How it's Supposed to Work

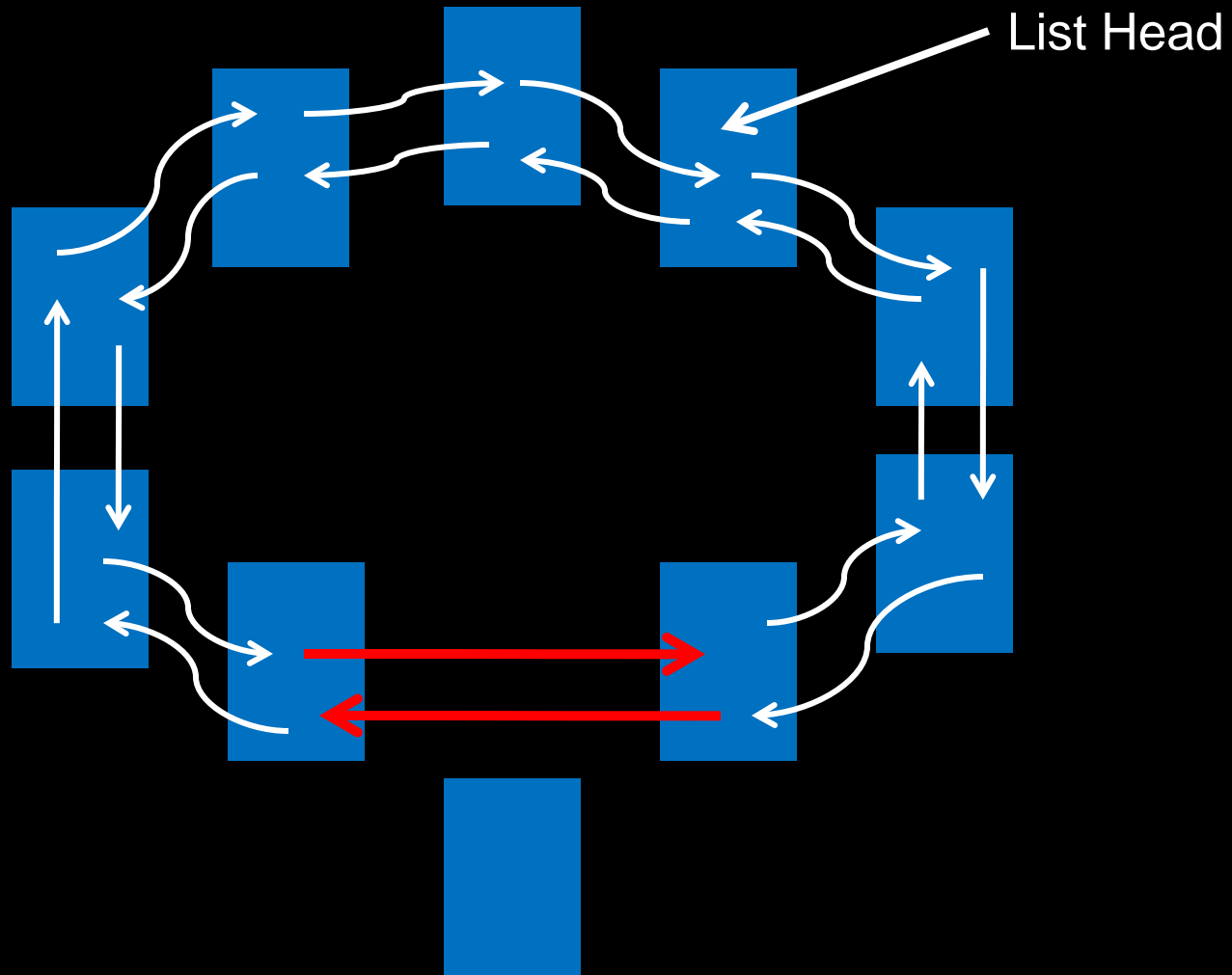


DKOM Example

- Unlink a process to hide it
- Adjust forward and back links to skip an item



Standard DKOM



Detecting Standard DKOM

- High-low analysis
 - Follow process links, record all processes
 - Brute force search for processes
- Compare the results
- Any process that shows up in one list but not the other is suspicious

α β γ δ ε ζ η θ κ λ π σ φ ψ
α β γ δ ε ζ η θ κ λ σ φ ψ

Devious DKOM

- How do you do a brute force search?
- Most modern tools looks for a magic value
- Magic values may not be required
- Some can be replaced with arbitrary values
 - System still runs

Process Structures

- Execute Process structure
 - EPROCESS
- Consists of several substructures
- Lives in pool memory
- Starts with a POOL_HEADER
 - You don't need to know what this is
 - Contains values set by kernel
 - But not referenced while running



Image courtesy Flickr user leozaza and licensed under the Creative Commons

Devious DKOM

- On Windows XP the POOL_HEADER starts with
50 72 6f e3 (“Proã” in ASCII)
- Can be replaced with, for example
00 00 00 00

Devious DKOM Demo

- Using Volatility Framework
 - <https://www.volatilesystems.com/default/volatility>

Devious DKOM Demo

- Not picking on Volatility
 - All existing tools use magic values

Detecting Devious DKOM

- Two approaches
 - Get better magic
 - Detect using something else

Better Magic



- Better Magic Through Fuzzing™
- Fuzzing means inputting random data and seeing what happens
- Use automated tools to only report the interesting inputs

Image courtesy Flickr user LaMenta3 and licensed under the Creative Commons

Better Magic

- Fuzzing to find magic values
 - Fire up virtual machine and start a process
 - Pause VM
 - Change EPROCESS values at random
 - Resume VM
 - Record if change made the process or machine crash
 - Repeat
- Do mathy stuff to generate rules for which values cannot be changed without a crash
- Full citation at the end,
http://www.cc.gatech.edu/~brendan/ccs09_siggen.pdf

Better Magic

- Examples from EPROCESS
- Pcb.ReadyListHead
 - List Head of threads ready to execute
 - $val \& 0x80000000 == 0x80000000$ AND $val \% 0x8 == 0$
- Peb
 - Address of Process Environment Block
 - $val == 0$ OR
 - $(val \& 0x7ffd0000 == 0x7ffd0000$ AND $val \% 0x1000 == 0)$

Problems with Better Magic

- These rules are for 32-bit Windows XP Service Pack 2 only
- Fuzzing must be repeated for each configuration
- Rules will be different for each configuration
 - Especially 64-bit systems

Detecting Devious DKOM

- Two approaches
 - Get better magic
 - Detect using something else

Kernel Objects

- Use inherent organization of the kernel
- The kernel is massive
 - Lots of structures to choose from
- Particularly focus on the connections between these objects

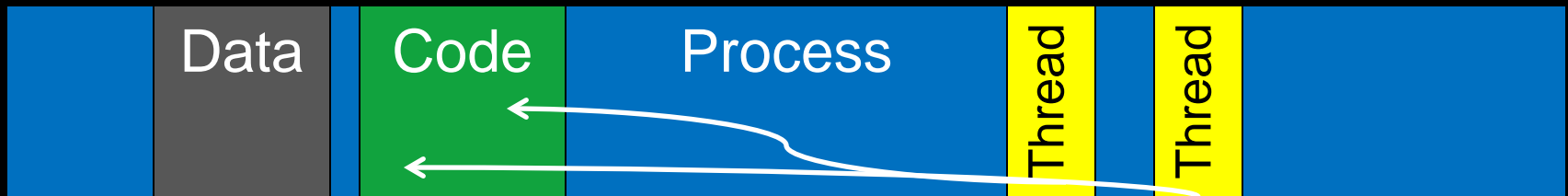
Processes

- A process is a container
 - Holds threads, handles, DLLs, and many other structures
- Let's talk about threads
 - Threads are paths of execution
 - Have a stack
 - Work off common code base
 - Can interact with other threads
- Every process starts with one thread
 - Can start more threads
- Could have a process with no threads, but it wouldn't do anything

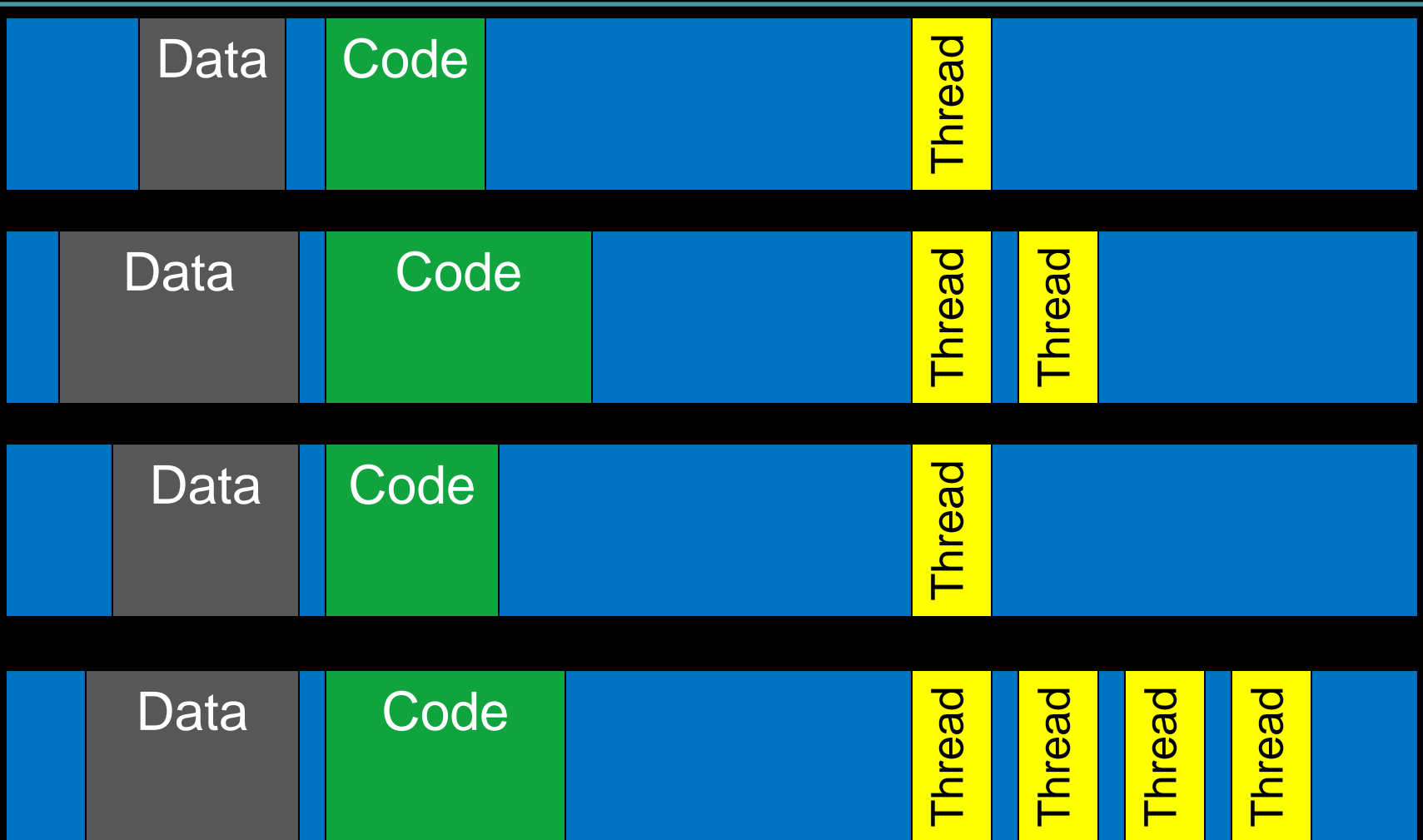
Threads



Threads



Threads



The Kernel

- The Kernel is just another process on the system
 - Starts first
 - Gets to talk to the hardware
 - Schedules threads
- Tells hardware to transfer execution to a thread for a given time
- When finished, hardware interrupts the thread
 - Allow it to store its data gracefully
- Return control to kernel

The Kernel



Image Copyright © 1999 Twentieth Century Fox

Why Manage Thread Scheduling?

- Some threads are higher priority
 - Video playback
- Some are lower priority
 - Prefetching content
 - Indexing service
- Threads can also be interrupted by hardware
 - Key press
 - Network packet received
- Thread currently executing may not handle the event

The Kernel

Thread

Thread

Thread

Thread

Thread

Thread

Thread

Thread

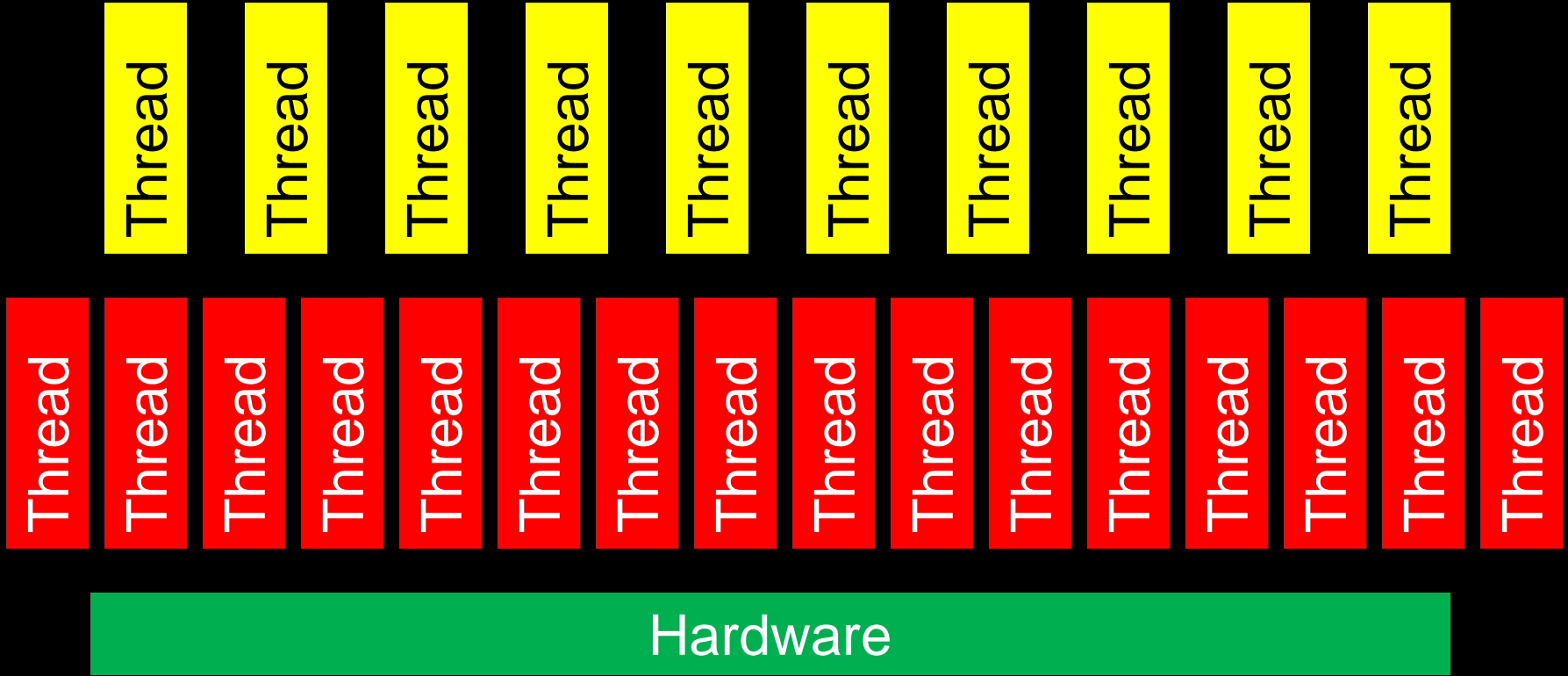
Thread

Thread

Kernel

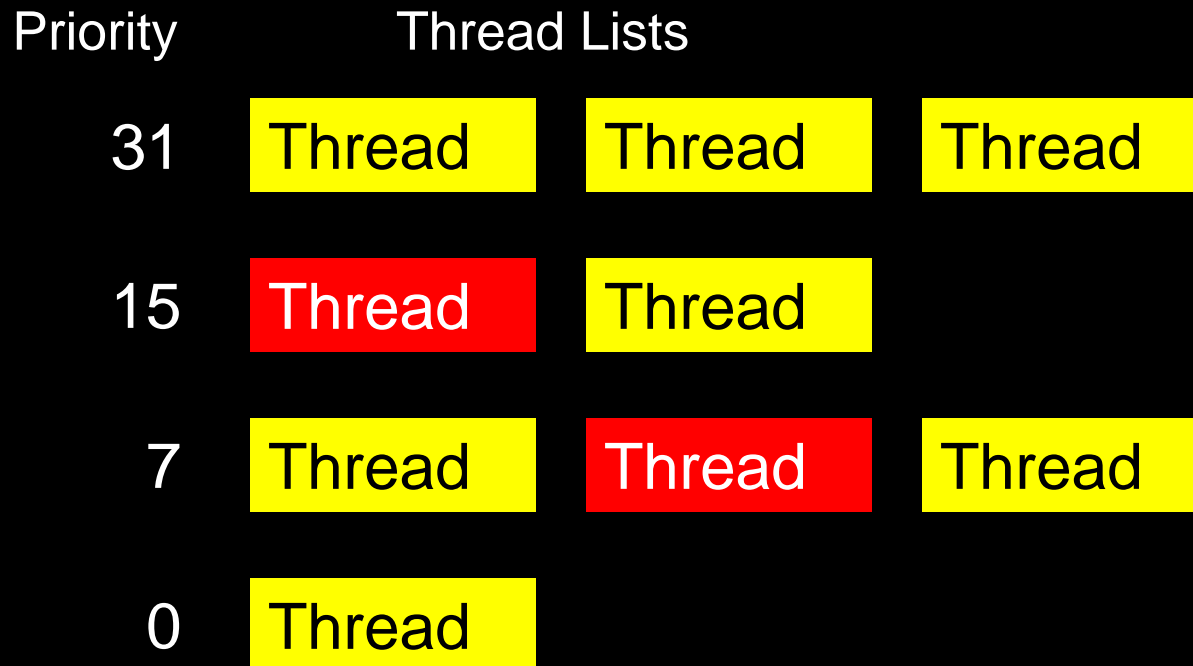
Hardware

The Kernel



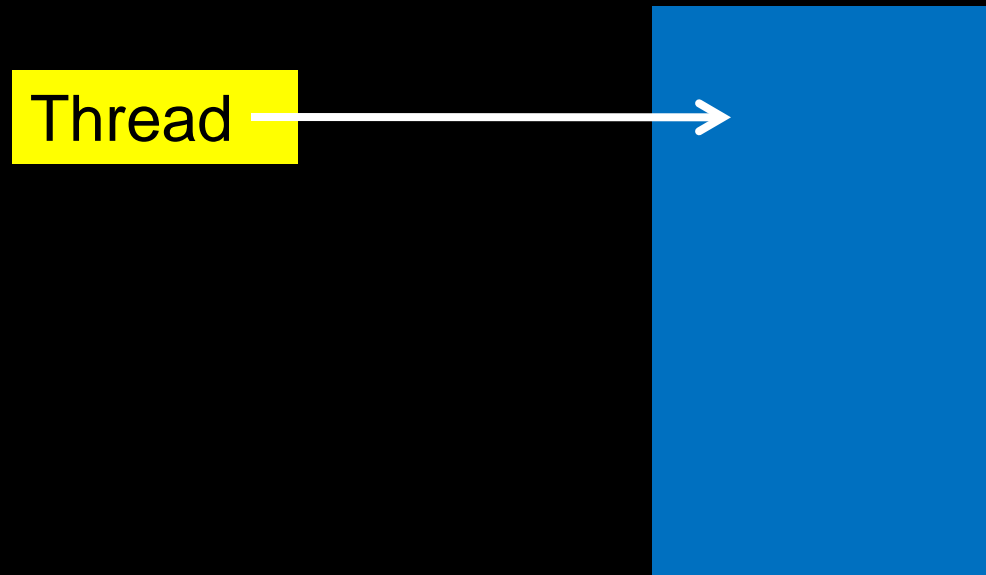
Windows Scheduler

- Structure used by Windows to schedule threads
- Organized by priority
- One doubly linked list for each priority level



Windows Scheduler

- Lists of threads
- Each points to an ETHREAD
- Each ETHREAD points to its EPROCESS

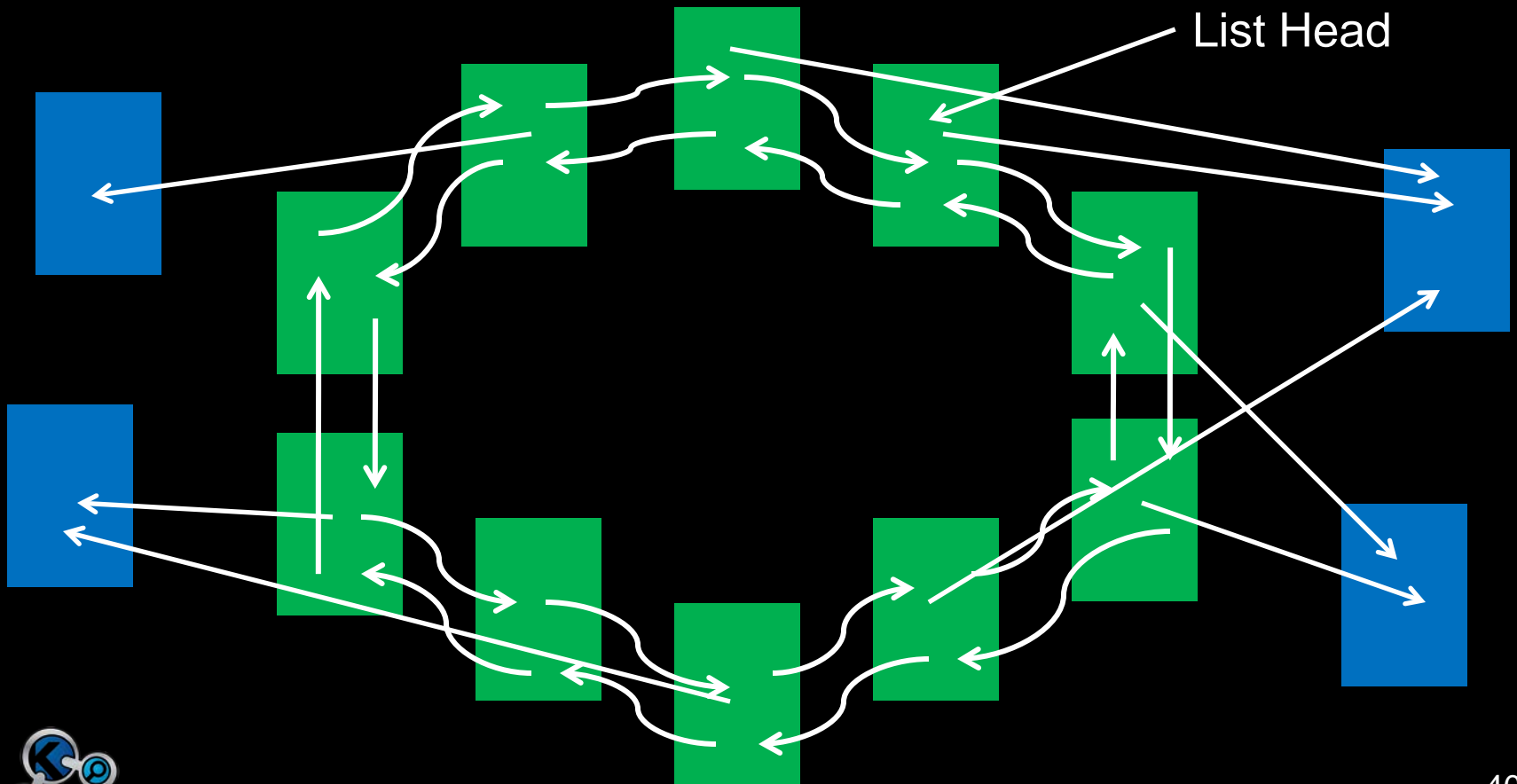


The Rootkit Paradox

- Rootkits want to run
- Rootkits don't want to be seen
- But to have the former, they must violate the latter
- Full paper
<http://www.utica.edu/academic/institutes/ecii/publications/articles/EF E2FC4D-0B11-BC08-AD2958256F5E68F1.pdf>

But wait, there's more!

- File handles also point to processes
- Kernel maintains list of handles

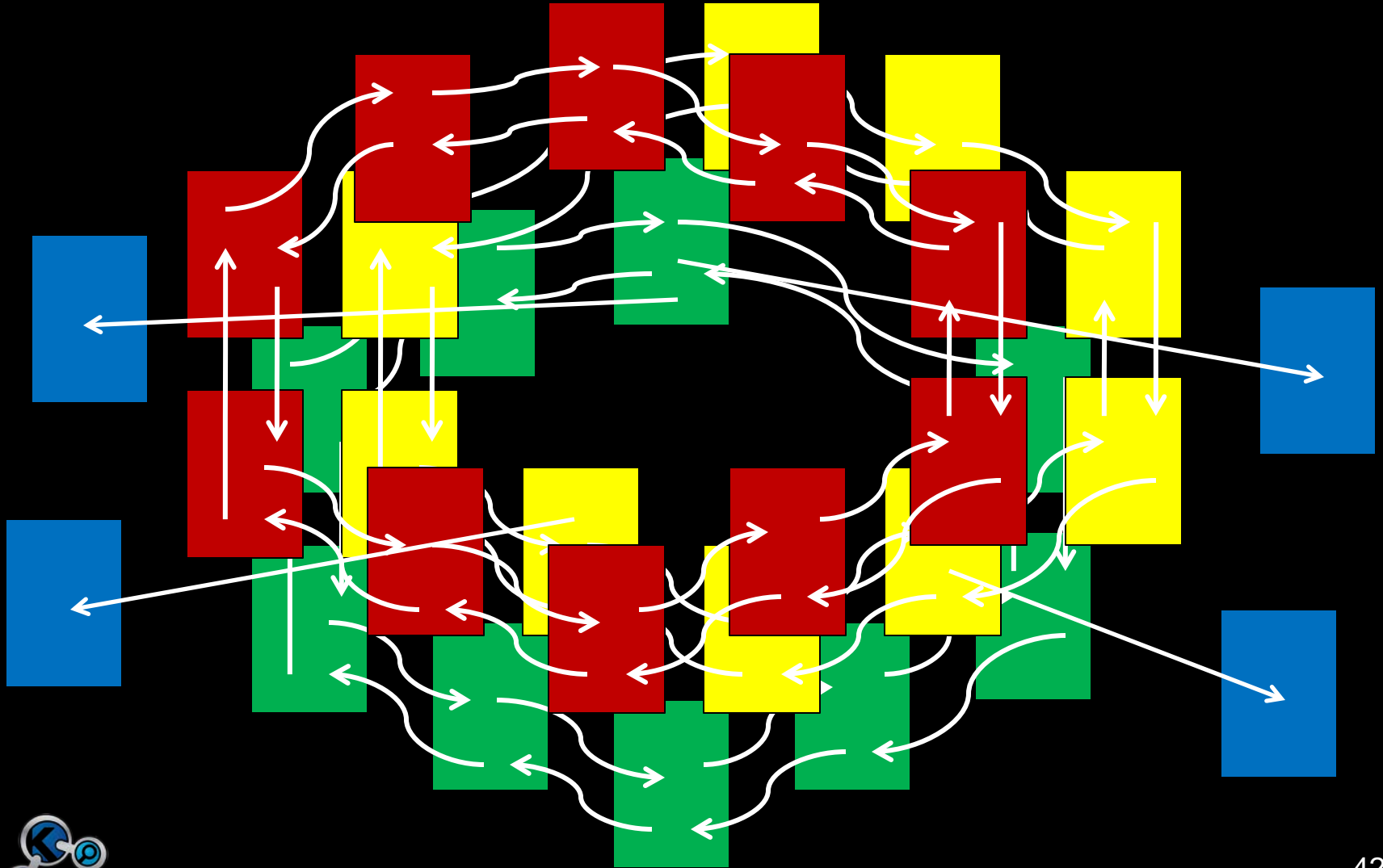


But wait, there's more!

- Processes point to threads
- Network connections point to processes
- And on and on and on...

- For an attacker to hide, they have to update everything
- We just have to validate everything
 - Any inconsistency means we win

But wait, there's more!



Coming Soon

- Unfortunately, no tools use either better magic or kernel objects
 - Yet
- Should be coming to AccessData tools in the near future

Outline

- Introduction
- The Kernel
- Direct Kernel Object Manipulation
- Standard DKOM
- Devious DKOM
- Better Magic
- Relations Between Kernel Objects
- Questions

References

- Brendan Dolan-Gavitt, Abhinav Srivasta, Patrick Traynor, and Jonathon Giffin, Robust Signatures for Kernel Data Structures. Proceedings of the ACM Conference on Computer and Communications Security (CCS), November 2009, http://www.cc.gatech.edu/~brendan/ccs09_siggen.pdf
- Jesse Kornblum, Exploiting the Rootkit Paradox with Windows Memory Analysis, International Journal of Digital Evidence, Fall 2006, <http://www.utica.edu/academic/institutes/ecii/publications/articles/EF E2FC4D-0B11-BC08-AD2958256F5E68F1.pdf>

Questions?

Jesse Kornblum
jesse.kornblum@kyrus-tech.com

