

# ***Windows Memory Analysis***

Jesse Kornblum

# Overview

---

- Why Memory Analysis
- Windows without Windows
- Gathering Information
- Parsing the Processes
- The Rootkit Paradox
- Address Translation
- Recovering Executables
- Fuzzy Hashing
- Getting memory images
- Analysis Tools

# Introduction

---



# Why

---

- Data that exists nowhere else
- Encryption Keys
  - BitLocker, PGP Whole Disk Encryption, etc.
- What was happening on the system
  - Running programs, open (unsaved) documents
  - Unpacked contents of packed programs
- What was really happening on the system
  - Not the sanitized (lying) version from the OS
  - Hidden programs, rootkits, injected code
- What was really happening on the system
  - What was running ten minutes ago

# Why You

- Anybody can play!
- Ed Felten and Princeton University
- Matt Suiche and Nicholas Ruff in France



# Why Windows?

---



# Open Source OS

---

- Open Source is Easier
  - You have the blueprints!
- Open Source is Harder
  - Everything changes with each recompilation

# Windows without Windows

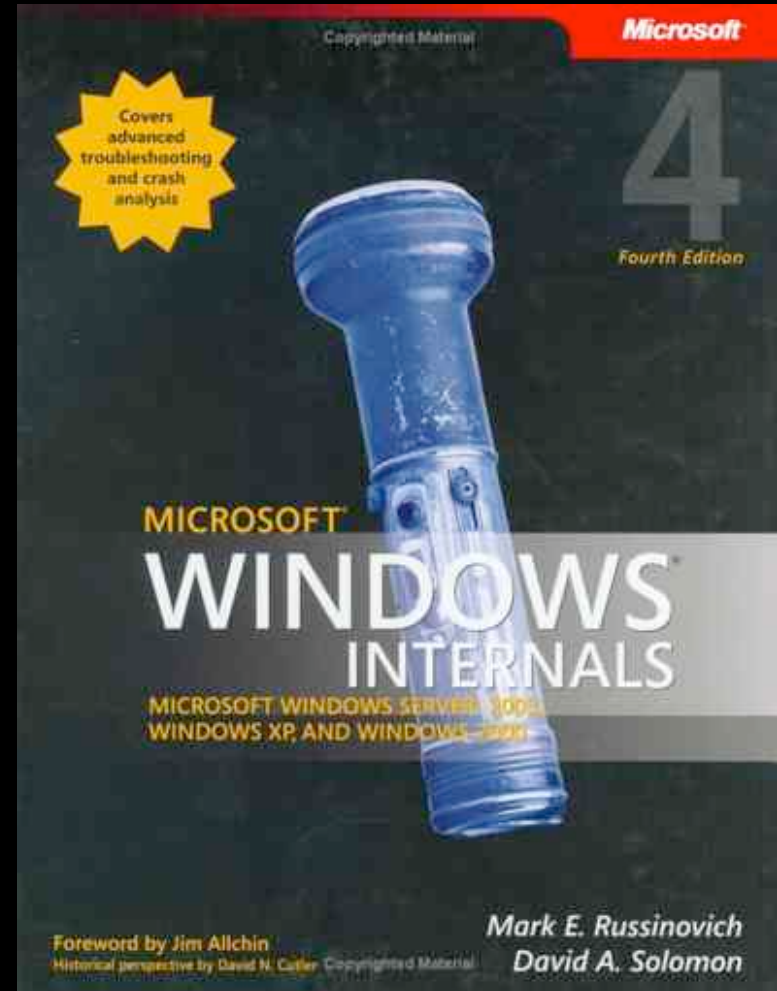
---

- We must do everything the OS would do
- Windows is complex
  - Minor understatement
- Thousands of structures
  - Many change between versions
  - Mostly undocumented
- Hacks on top of short cuts on top of optimizations on top of...



# Windows without Windows

- *Microsoft Windows Internals*
  - Russinovich and Solomon
  - Yes, the SysInternals guy
  - Fifth edition in 2009?



# Windows without Windows

---

- Other Resources
  - Intel
  - Microsoft Documentation
  - Windows Research Kernel
  - Malware authors

# Getting Started

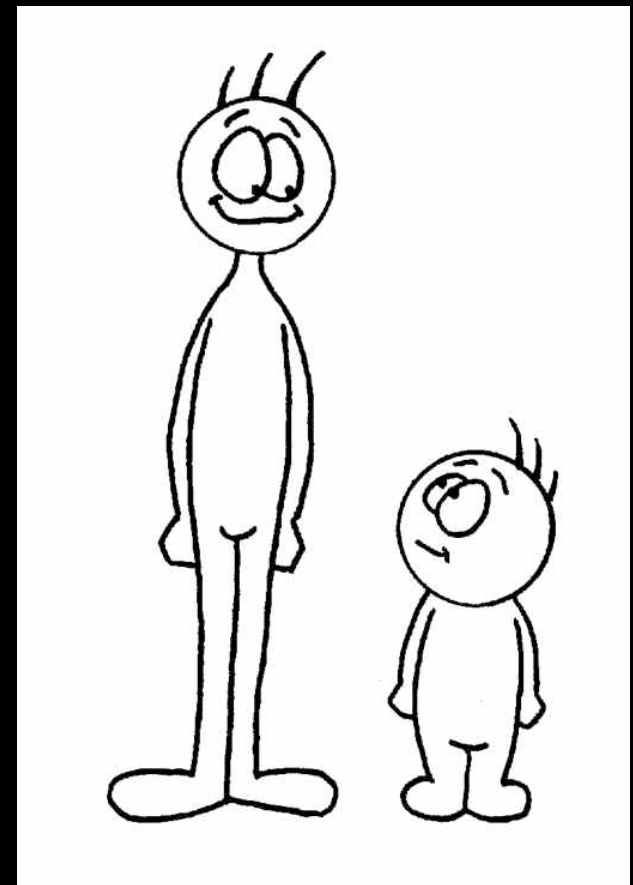
---

- Brute force search through image
  - Processes
  - Application specific information
  - etc etc
- ptfinder by Schuster is a great start
  - See DFRWS 2006 paper
- Volatility Framework can help you explore

# High/Low Comparison

---

- Compare Windows structures to brute force search results
  - Similar to rootkit detectors

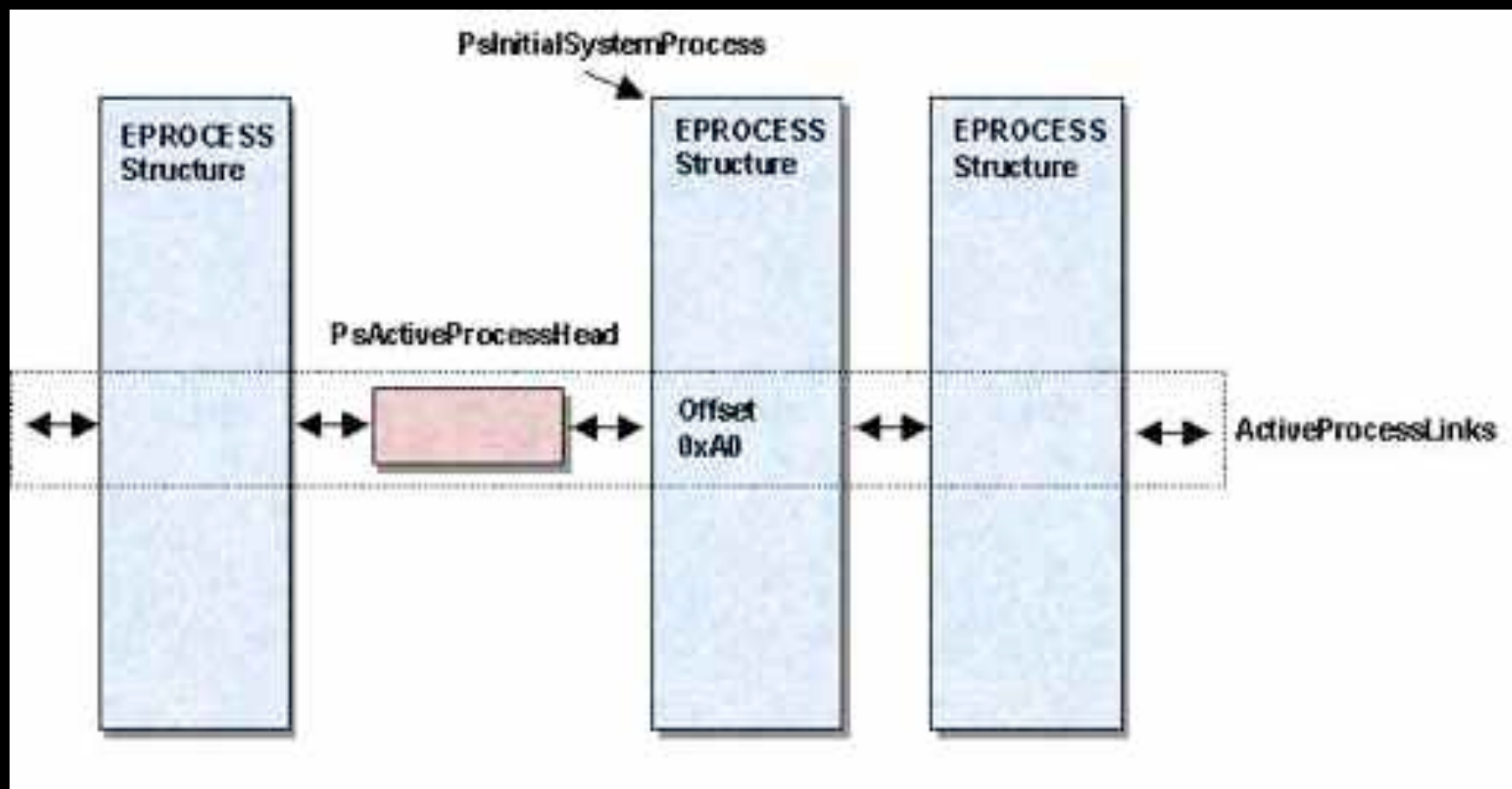


# EProcess

---

- Executive Process structure
  - Each userland process has one
- My starting point
  - Your mileage may vary
  - Feel free to use something else
- They are doubly linked with all other active processes
  - What the OS thought was running

# Active Process Links



# Suspicious Processes

---

- Linked Process is in memory image but not OS list
  - Hidden process
  - API Hooking
  - Hacker Defender rootkit, et al.
- Linked Process in OS list but not memory
  - Phantom process
  - Never seen, but could happen
  - “Yes, your anti-virus program is running! Really!”

# Suspicious Processes

---

- Find both running and terminated processes
- Terminated means it ended
  - Yes, Windows really records this
  - Including the time it ended
- Unlinked running process means either
  - Unlinked by DKOM
  - Leftover from previous boot
- Compare start time versus boot time



# Leftover from Previous Boot?

---



- Sure!
- RAM is not wiped at boot
- It used to be wiped
  - RAM test at startup
  - Took too much time, so now disabled by default
- See "Lest We Remember: Cold Boot Attacks on Encryption Keys" by Felten et al for more.

# Process Information

---

- Full name and path
- Command line arguments
- Process ID number (PID)
- Parent PID
- Current directory
- Window Title
- Handles
  - Files, devices, drivers
- List of loaded modules
  - DLLs

# Process Information

---

- Suspicious program names
  - UMGR32.EXE



- Suspicious command lines
  - C:\TEMP\backdoor.exe -r 63.161.169.137 -stealth
  - X:\h4x0r-Toolkit\notepad.exe
  - c:\windows\system32\cmd.exe

# Process Information

---

- Most system processes have well defined parents
  - cmd.exe should not be the parent of lsass.exe
- Most user processes are started by Explorer.exe
- It's suspicious when they're not
  - Maybe started from a command prompt
  - Orphaned processes
- Some system processes should never start programs
  - lsass.exe should not start cmd.exe



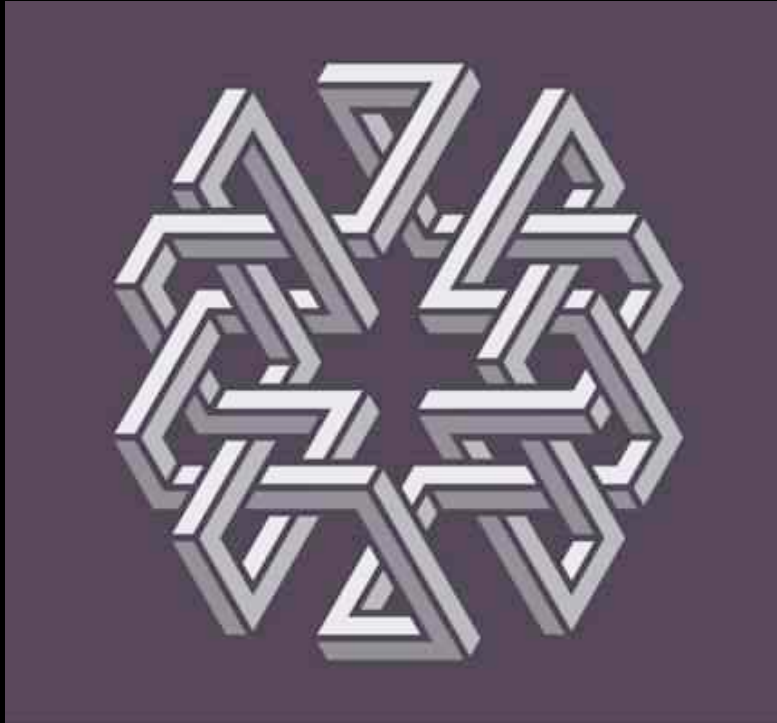
# Process Information

---

- List of DLLs for each process
  - We get the name, path, and size of each
- What is notepad.exe doing with wsock32.dll?
- What is iexplore doing with c:\temp\wsock32.dll?
- What if there is no path information?
  - Injected code!
  - First part of the rootkit paradox
  - If they supply fake data, where is it on the disk?

# The Rootkit Paradox

---



- The more a rootkit tries to hide, the easier it is to see
- All rootkits obey two rules
  - n They don't want to be seen
  - n They want to execute
- We will see you in the operating system or in memory analysis, your choice

# The Rootkit Paradox

---

- Anything can be subverted
  - Especially memory acquisition
- Normally, errors don't happen
- If rootkit creates error, it's just indicated its presence!
  - The lack of data is in the indicator
- For more, see Rootkit Paradox paper in Int'l Journal of Digital Evidence

# Recovering Executables

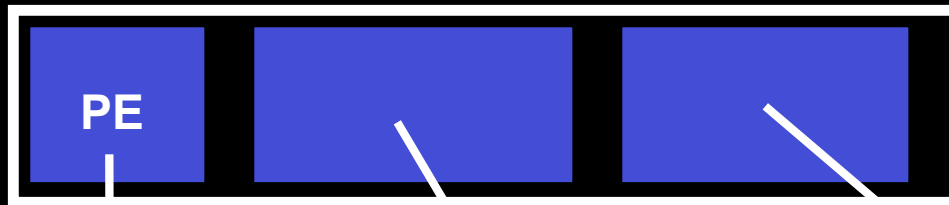
---

- We have pointers to recover each module
- Recreate files (mostly) as they existed on the disk
- First page of module has PE header
  - Names, sizes, and locations of sections
  - Locations both in memory and on disk
- Can read sections from memory and write back to disk

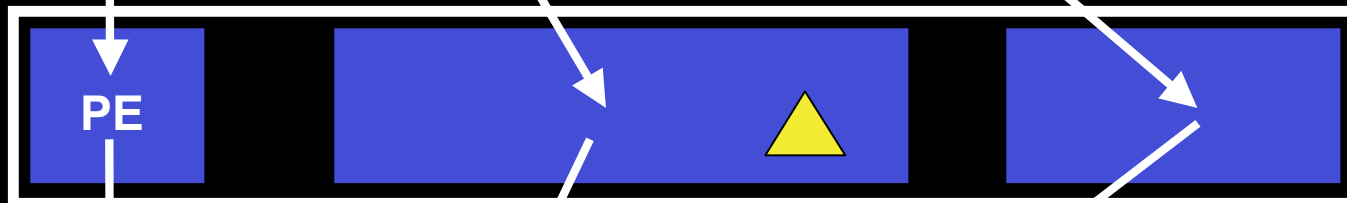


# Recovering Executables

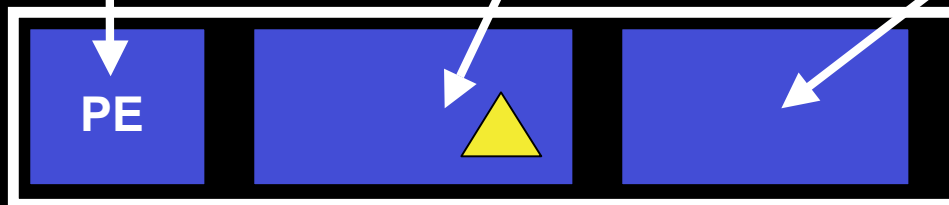
On Disk



In Memory



Recovered



# Recovering Executables

---

- Recovered programs not identical to versions on disk
  - Resource mappings change
  - Program variables
- If only we had a tool to match slightly different versions of a file back to the originals...

# Disclaimer

---



- I didn't invent this math
- Originally Dr. Andrew Tridgell
  - Samba
  - rsync was part of his thesis
  - Modified slightly for spamsum
    - Spam detector in his “junk code” folder
- User report that rsync confuses similar Word documents

# MD5 Explained

---

How MD5 (roughly) works:

1. Start with an initial state
2. Look at fixed size block of input
  - Do mathy stuff with current state and block
  - Get new state
3. Advance to next block of input
4. Repeat steps 2 and 3 until out of input blocks
5. Ending state is the hash

# MD5 Explained

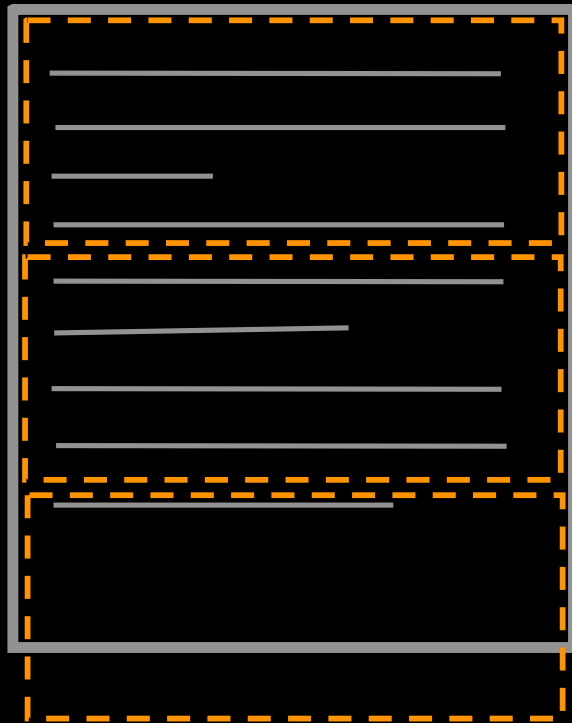
---

- If you change one bit in the middle, you change the next state
- Which ends up changing the end result
- Is this a good thing or a bad thing?



# Piecewise Hashing

- Developed for integrity during imaging
- Divide input into fixed sized sections and hash separately
- Insert or delete changes all subsequent hashes



**3b152e0baa367a8038373f6df**



**40c39f174a8756a2c266849b**



**fdb05977978a8bc69ecc46ec**

# Rolling Hash

---

- It would be nice to set boundaries such that
  - Insertions and deletions are contained within a block

# Rolling Hash

- A different kind of hash function
- Produces a pseudorandom output for every position in a file
  - Depends only on last few bytes
  - Lots of academic work on these
  - Just mathy tricks

F o u r s c o r e -> 83,742,221

F o u r s c o r e -> 5

F o u r s c o r e -> 90,281



# Rolling Hash

---

To update state (c,x,y,z>window) for a byte d:

$y = y - x$

$y = y + \text{size} * d$

$x = x + d$

$x = x - \text{window}[c \bmod \text{size}]$

$\text{window}[c \bmod \text{size}] = d$

$c = c + 1$

$z = z \ll 5$

$z = z \text{ XOR } d$

return (x + y + z)

# Rolling Hash

---

- We use the rolling hash to generate block boundaries
- Select some values as trigger points
- When we hit a trigger point, end the block
- Example
  - Excerpt from "The Raven" by Edgar Allan Poe
  - Triggers on ood and ore

# Rolling Hash

---

Deep into the darkness peering, long I stood there, wondering, fearing  
Doubting, dreaming dreams no mortals ever dared to dream before;  
But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word,  
Lenore?, This I whispered, and an echo murmured back the word,  
"Lenore!" Merely this, and nothing more

# Rolling Hash

---

Deep into the darkness peering, long I st~~oo~~d there, wondering, fearing  
Doubting, dreaming dreams no mortals ever dared to dream bef~~or~~e;  
But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word,  
Len~~or~~e?, This I whispered, and an echo murmured back the word,  
"Len~~or~~e!" Merely this, and nothing mor~~e~~

# Rolling Hash

---

Deep into the darkness peering, long I st**ood**

there, wondering, fearing Doubting, dreaming dreams no mortals ever  
dared to dream bef**ore**

; But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word, Len**ore**

?, This I whispered, and an echo murmured back the word, "Len**ore**

!" Merely this, and nothing m**ore**

# Rolling Hash

---

- How do we choose the triggers?
  - Chosen randomly, before reading the file
  - Based on the size of the input file
  - Really just a set of numbers
  - Has nothing to do with type of input data

# Fuzzy Hashing

---

- Combine Rolling Hash with a Traditional Hash
- Use Fowler/Noll/Vo (FNV) hash
  - That's what Tridgell did
  - Faster and less complex than MD5
  - We're only using a small part of the result
- Start reading file, compute Rolling and Traditional Hashes
- When Rolling Hash triggers
  - Record LSB of Traditional Hash value
- When finished, combine LSBs to make signature

# Rolling Hash

---

Deep into the darkness peering, long I st**ood**

there, wondering, fearing Doubting, dreaming dreams no mortals ever  
dared to dream bef**ore**

; But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word, Len**ore**

?, This I whispered, and an echo murmured back the word, "Len**ore**

!" Merely this, and nothing m**ore**



# Rolling Hash

Deep into the darkness peering, long I st**ood**

28163

there, wondering, fearing Doubting, dreaming dreams no mortals ever  
dared to dream bef**ore**

491522

; But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word, Len**ore**

57

?, This I whispered, and an echo murmured back the word, "Len**ore**

!" Merely this, and nothing m**ore**

145410213

738210

# Rolling Hash

Deep into the darkness peering, long I stood

28163

there, wondering, fearing I AM THE LIZARD KING! Doubting, dreaming  
dreams no mortals ever dared to dream before

82910

; But the silence was unbroken, and the stillness gave no token,  
And the only word there spoken was the whispered word, Lenore

57

?, This I whispered, and an echo murmured back the word, "Lenore

!" Merely this, and nothing more

145410213

738210

# Matching

---

Signature 1: 3 2 7 3 0

Signature 2: 3 0 7 3 0

- Edit Distance
  - Number of insertions, modifications and deletions to turn Signature 1 into Signature 2.
  - For the example above, the edit distance is one.
- Signatures (and thus files) match when the ratio of the edit distance to the length is small

# Demonstration

---

**WARNING:  
EXPLICIT IMAGERY**

# Demonstration

---



# Demonstration

---

## Corrupted File



**Known kitty porn**



**MATCH**

# Demonstration

---

## Different File



**Known kitty porn**



**No match**

# Demonstration

---

## File Header



**Known kitty porn**



**MATCH**



# Demonstration

---

**File Footer**



**Known kitty porn**

**MATCH**

# Demonstration

---

## File Footer



**Known kitty porn**



**MATCH**

# Issues

- Does not work for similar looking graphics
  - For that use imgSeek  
<http://www.imgseek.net/>
- Unable to handle cropping, resizing, and other edits
- Confused by many small changes throughout input
- Computationally intensive
  - 7-10 times slower than MD5
- No way to sort signatures
  - Must compare each input to all known signatures



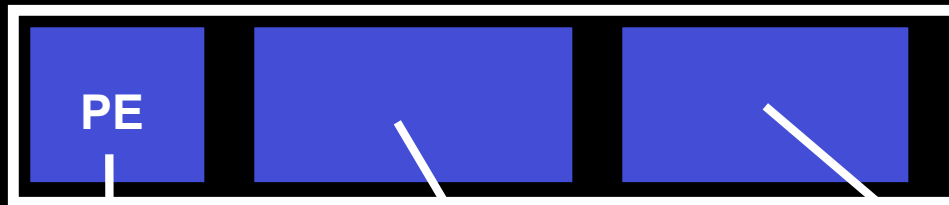
# Fuzzy Hashing

---

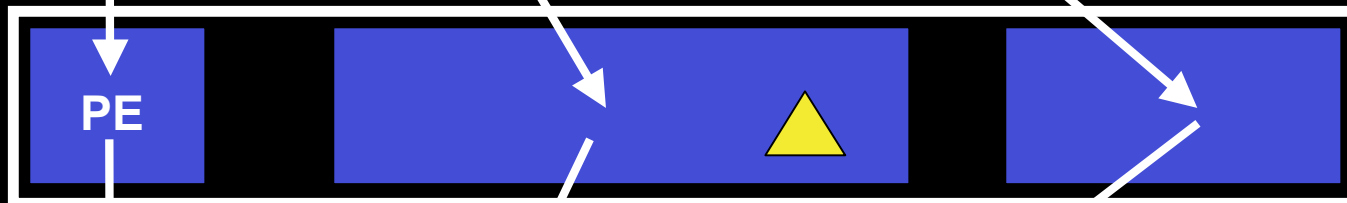
- Matches similar but not identical bitstreams
  - Great for corrupted or partial documents
  - Also great for source code reuse
- Freely available
  - <http://ssdeep.sf.net/>
  - Windows, Windows GUI, \*nix, and OS X
- Academic paper
  - <http://dfrws.org/2006/proceedings/12-Kornblum.pdf>

# Recovering Executables

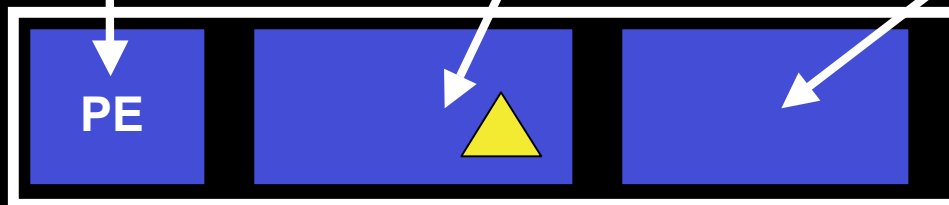
On Disk



In Memory



Recovered



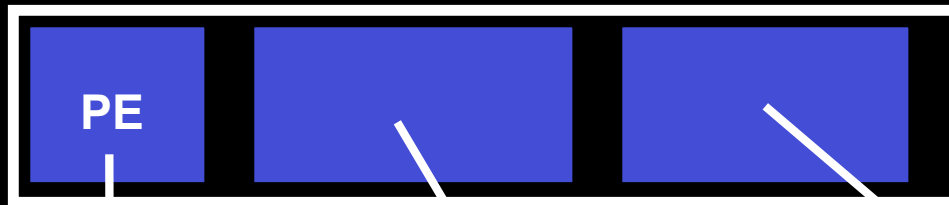
# Recovering Executables

---

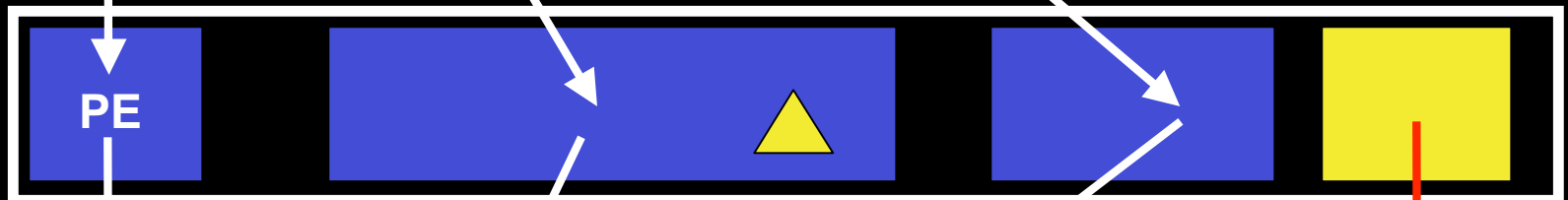
- Packed programs
  - Self decompressing or decrypting
- Look at section names
  - Normal values are things like .code, .data, .text
  - Well known packing program UPX
- Also check for sections that are zero bytes on disk
  - Data is decompressed on load into these sections
  - Indicate packed program
- Some packed programs are “ok”
  - Skype

# Recovering Executables

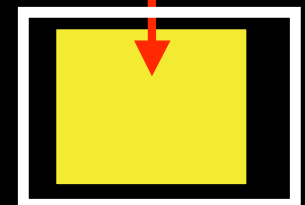
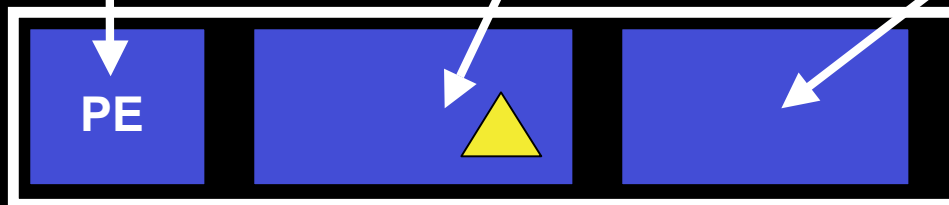
On Disk



In Memory



Recovered



# Getting Memory Images

---

- Hardware debugger
  - Best, but hard to do
  - Especially for incident response
- Suspend virtual machine
- Imaging programs
  - mdd, win32dd, kntdd, winen
  - Not like a disk image
  - Data changed during acquisition and by acquisition



# Getting Memory Images

---

- Hibernation Files
  - Sandman library can decompress it
  - Probably not a good idea for incident response
- Crashdump files
  - Andreas Schuster can show you how to convert
  - Probably not a good idea for incident response
- Load a second OS
  - Body Snatcher, Schatz, DFRWS 2007

# Getting Memory Images

---

- DFRWS Memory Images
  - <http://dfrws.org/2005/challenge/>
  - Two Windows 2000 Service Pack 1 images
  - Malware on both
- Digital Forensics Tool Testing
  - <http://dfft.sourceforge.net/test13/>
  - Five images: Windows 2000, XP, 2003, Vista Beta 2
  - Boomer is multiprocessor

# Analysis Tools

---

- No major commercial support (yet)
- kntdd and kntlist
  - Expensive as all get out
- Volatility by AAron Walters and company
  - Free, open source python based
  - Integrated into pyflag, if you'd like
  - <https://www.volatilesystems.com/>

# Analysis Tools

---

- The rest of the field:
  - Andreas Schuster - ptfinder
    - Great blog
  - Chris Betz - Memparser - <http://sourceforge.net/projects/memparser>
  - Harlan Carvey has some Perl scripts
  - Mariusz Burdach - WMFT
  - Joe Stewart - pmodump (TRUMAN)

# Analysis Tools

---

- Your name here...

# Conclusion

---

- Why Memory Analysis
- Windows without Windows
- Gathering Information
- Parsing the Processes
- The Rootkit Paradox
- Address Translation
- Recovering Executables
- Fuzzy Hashing
- Getting memory images
- Analysis Tools

# Questions

---



Jesse Kornblum

[jesse.kornblum@mantech.com](mailto:jesse.kornblum@mantech.com)

**ManTech**  
International Corporation ®