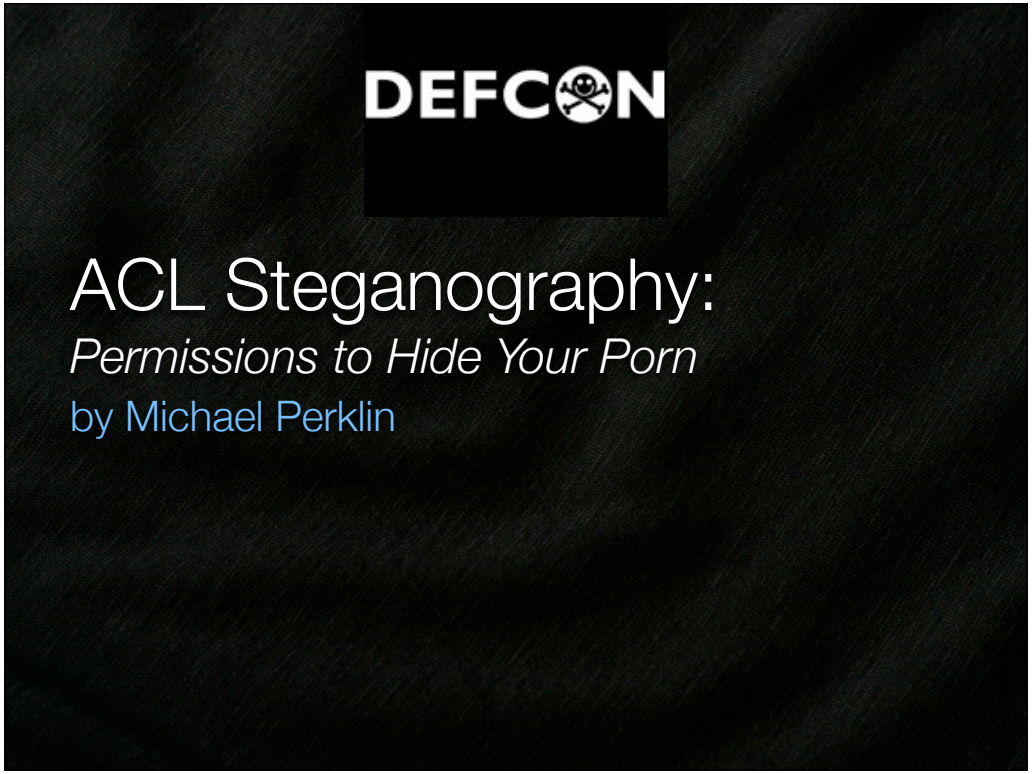

DEFCON
ACL Steganography:
Permissions to Hide Your Porn
by Michael Perklin

Michael Perklin

- Corporate Investigator
- Digital Forensic Examiner
- Security Professional
- eDiscovery Administrator

- Computer Geek + Legal Support hybrid

Michael Perklin

- Diploma in Computer Science Technology
- Bachelor's Degree in Information Systems Security
- Master's Degree in Information Assurance
- EnCase Certified Examiner (EnCE)
- AccessData Certified Examiner (ACE)
- Certified Information Systems Security Professional (CISSP)

In This Talk...

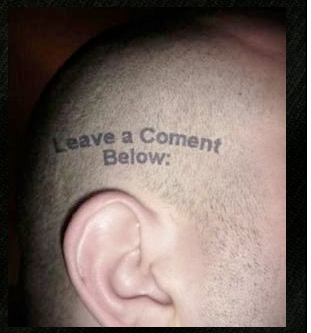
- What is Steganography?
  - Historical examples of physical and digital forms
  - How do they work?
- Identifying a “Lowest Common Denominator”
- ACL Steganography - a new scheme

What Is Steganography?

- Greek origin and means "concealed writing"
  - steganos (στεγανός) meaning "covered or protected"
  - graphei (γραφή) meaning "writing"
  - The term was first coined in 1499, but there are many earlier examples
- Basically, hiding something in plain sight

Classical Example: Tattoo

- Tattoo under hair
  - Encoder tattoos a slave's scalp
  - Decoder shaves the messenger's hair
- The message must be delayed to allow time for hair regrowth

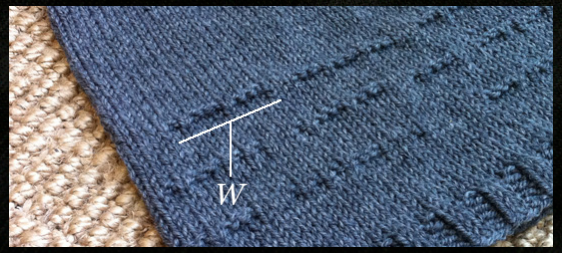


Tattoos Are Permanent

- Oops

Classical Example: Morse

- Stitch morse code into a sweater/jacket worn by a messenger
- Messenger hand-delivers one message while actually delivering two

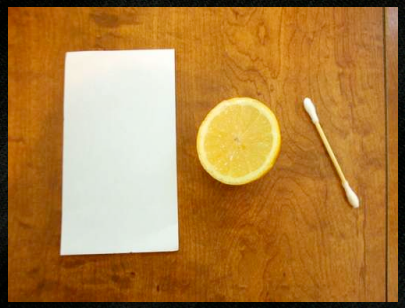



× THIS WORK WAS ×
× DONE BY ×
× MAJOR ×
× A.T.CASDAGLI. ×
× № 3311. ×
× WHILE IN ×
× CAPTIVITY AT ×
DOSSEL-WARBURG
× GERMANY. ×
× DECEMBER 1941. ×

Classical Example: Invisible Ink

- Write secrets with lemon juice
- Allow to dry

- Decode with heat
  (candle, match, hair dryer, iron)



Decode With Heat

Digital Example: Photos

- Files can be encoded as colour information embedded in a photo
- Most common type of digital steganography
- Based on the fact that only super-humans can tell the difference between Chartreuse and Lemon

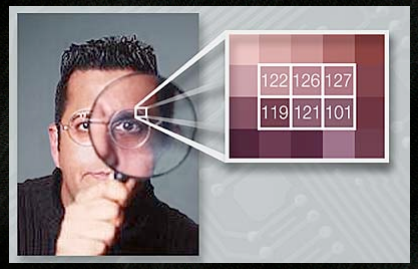


Photo Steganography

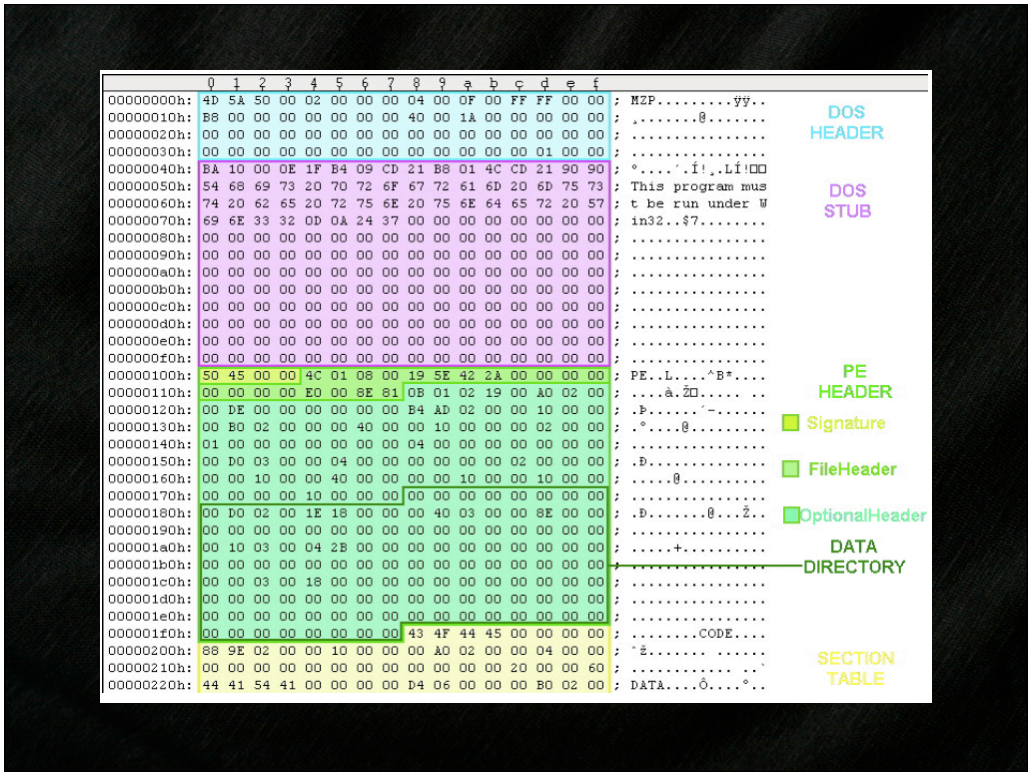
- Each pixel is assigned a colour with an RGB colour code
- The last bit of this 8-bit code is overwritten with encoded data
- #DFFF00 is chartreuse
- #DFFF01 is.... one of the yellows
- 8 adjacent pixels with 8 slightly-adjusted colours allows 1 byte of encoded information

Audio Steganography

- Same principle as photographic steganography, but with audio
- Humans can't easily tell the difference between 400hz and 401hz, especially if the note isn't sustained
- Alter each frame of audio with 1 bit of encoded information

Digital Example: x86 Ops

- Information can be encoded in x86 op codes
  - NOP - No Operation
  - ADD / SUB - Addition and Subtraction
- PE files (standard .exe programs) have many other areas that can hold arbitrary data


DOS HEADER
DOS STUB
PE HEADER
Signature
FileHeader
OptionalHeader
DATA DIRECTORY
SECTION TABLE

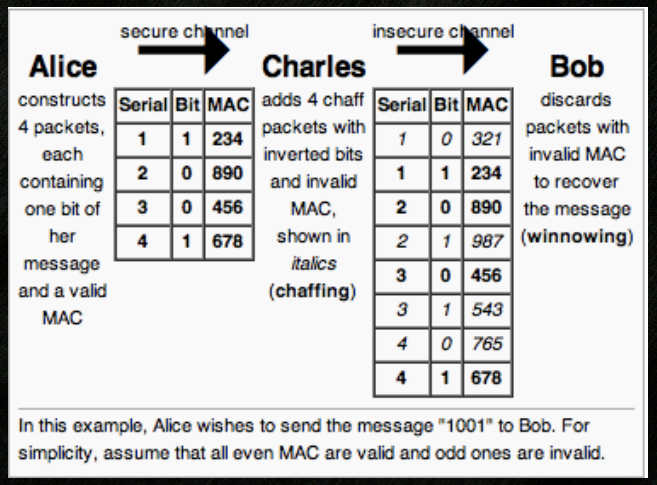
Digital Example: Chaffing and Winnowing

- Conceived by Ron Rivest in 1998 (the R in RSA, as well as RC4 and others)
- Not quite steganography
- Not quite encryption
- Has properties of both stego and encryption

Chaffing and Winnowing

- Sender issues ‘real’ messages and ‘chaff’ messages
- Listeners don’t know which messages are real
- Real chunks of the message include a parity value
  - Message Authentication Code (MAC)
- Receiver calculates MACs on every packet
  - Discards packets whose MACs aren’t valid
  - Reassembles all packets with valid MACs

Chaffing and Winnowing



Serial	Bit	MAC
1	1	234
2	0	890
3	0	456
4	1	678

Serial	Bit	MAC
1	*0*	*321*
1	1	234
2	0	890
2	*1*	*987*
3	0	456
3	*1*	*543*
4	*0*	*765*
4	1	678

Courtesy: Wikimedia Commons

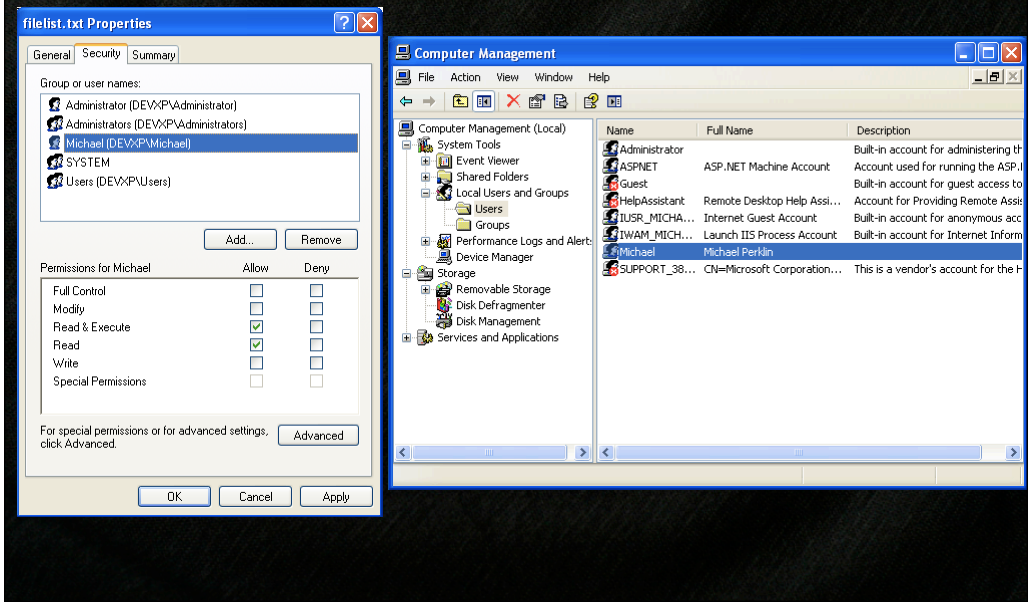
Steganography Breakdown

- All types of steganography require three things:
  - A medium of arbitrary information
  - A key or legend for encoding information
  - A way to differentiate ‘encoded’ and ‘medium’ info

ACL Steganography

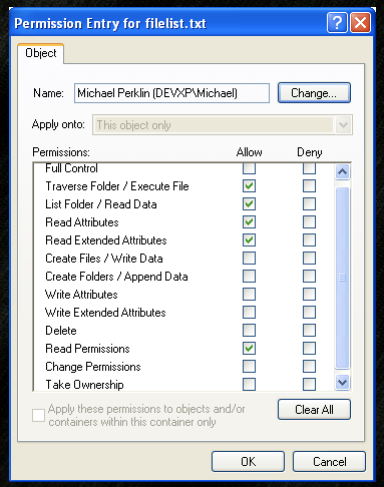
- A way to encode files as Access Control Entries within Access Control Lists of files stored on an NTFS volume
  - Medium: All files on an NTFS volume
  - Key: Security Identifiers in ACEs
  - Differentiator: ACEs with an unlikely combination of permissions

Background: NTFS Security



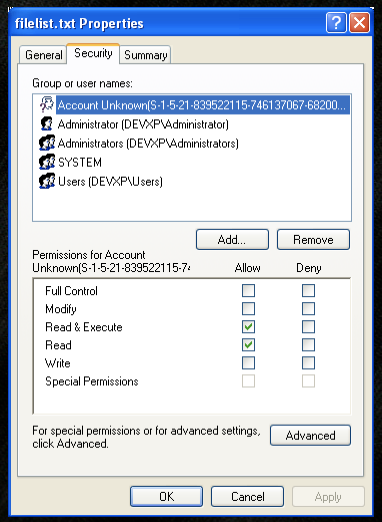
NTFS Permissions

- Entries correspond to system users
- There are 22 unique permissions available, stored in a 32-bit field
- Many more granular permissions exist than “Read, Write, Execute”



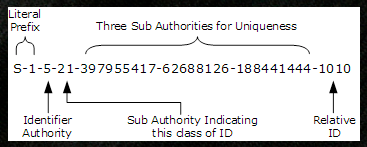
NTFS Permissions

- Permission entries are stored using Security Identifier (S-ID)
- If the user is removed, the OS can't look up the friendly name
- Photo shows same file after "Michael" is removed from OS



NTFS Security Identifiers

- Maximum Size: 68-bytes
- 1st byte is the revision
  (Always 1)
- 2nd byte is the count of SubAuthorities in this SID
  (Maximum 15 SubAuthorities per SID)
- 6 bytes used for the Identifier Authority
  (Always 000004)
- 60 bytes store the content of the SubAuthorities and the Relative ID



Acronym Review (AR)

- Access Control List (ACL)
  - A list of Access Control Entries
- Access Control Entry (ACE)
  - A permission rule (allow or deny) pertaining to a SID
- Security Identifier (SID)
  - A unique identifier for a user or group of a Windows system

ACL Steganography

- (photo of file with 60byte chunks)
- A file is split up into 60-byte chunks
- Each chunk becomes a SID
- ACEs are created with “Allow” permissions for each of these SIDs
- ACEs are added to the ACLs of multiple files

Demonstration

- A folder full of files
- A filelist.txt with these files
- A .tc volume with cool stuff in it
- Encoding the volume
- Showing the ACEs on the files
- Decoding the volume

ACLEncoding Details

- Two bits are set for all ACLEncoded entries:
  - Synchronize + ReadPermissions
  - Synchronize cannot be set within the Windows UI
- The 9 least significant bits are used as a counter from 0-512
  - These bits correspond to the permissions: ReadData, CreateFile, AppendData, ReadExtendedAttribute, WriteExtendedAttribute, ExecuteFile, Traverse, DeleteSubdirectoriesAndFiles, ReadAttributes

ACLEncode Details

- The FileList becomes a kind of symmetric key between the encoder and decoder
- The list identifies:
  - Which files have ACLEncoded entries
  - The order in which those entries are encoded

Limitations

- An ACL can be no bigger than 64kB per file
- Maximum ACE size is 76 bytes (68 for SID + 8 byte header)
- This produces a theoretical maximum of 862 ACEs per file
- I've imposed a limit of 512 entries per file
  - This leaves room for legitimate permissions

Limitations

- The largest possible file to be encoded:
  - NumFilesInList * 512 * 60bytes
  - or about 30kB per file
- Need to store a larger file? Use a longer file list.

$SECURE File Limitation

- The $SECURE file is a hidden file on every NTFS volume
- All ACLs for all files are stored in this one file

- Each time a new SID is encountered, it's added to this file
  - This way, future permission operations for that user can use the existing reference without duplicating it

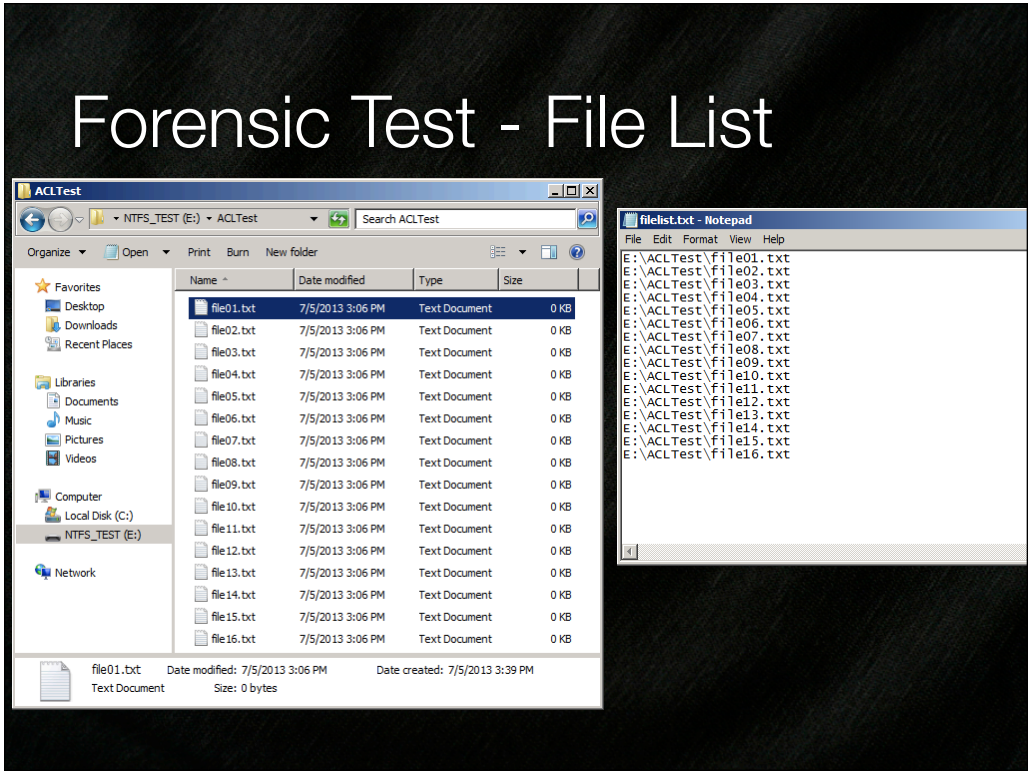
$SECURE File Limitation

- NTFS does *NOT* remove old/unused SIDs from the $SECURE file
- The $SECURE file is designed only to grow in size and never shrink
- This means, every ACLEncoded chunk from every run of ACLEncode will persist here forever

A Forensic Review

- I conducted a test:
  - 2GB USB Key, formatted as NTFS
  - AccessData FTK 4.0.2.33
  - Guidance EnCase Forensic 6.19.6

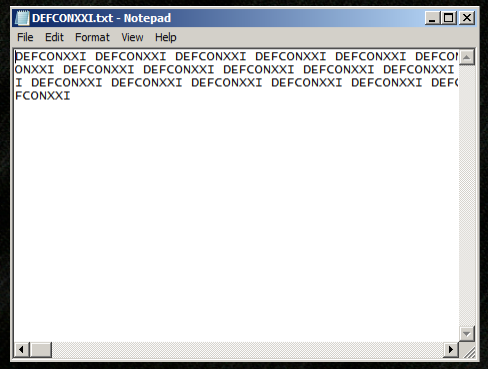
Forensic Test - File List

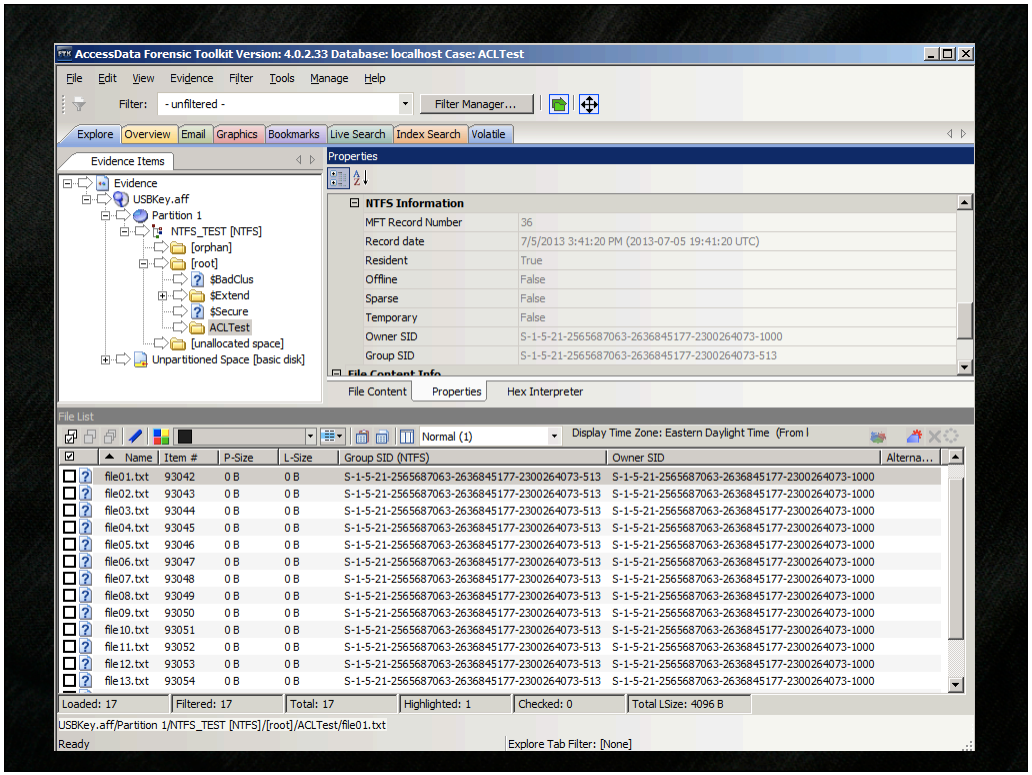


I created these files for the test
I could have used any file already on the system

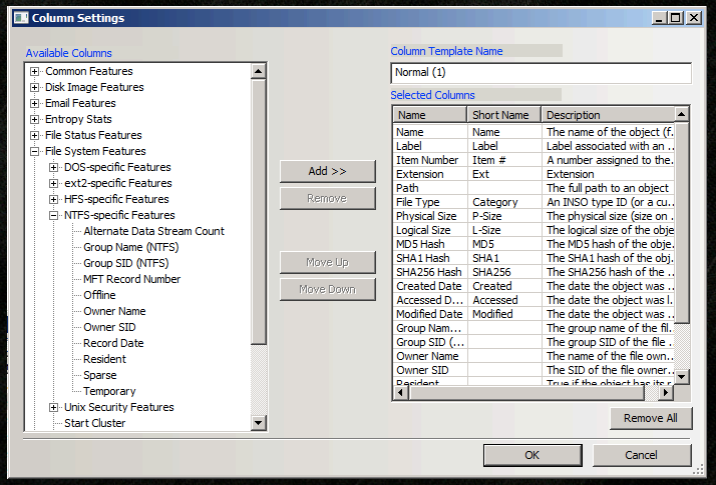
Forensic Test - Input File

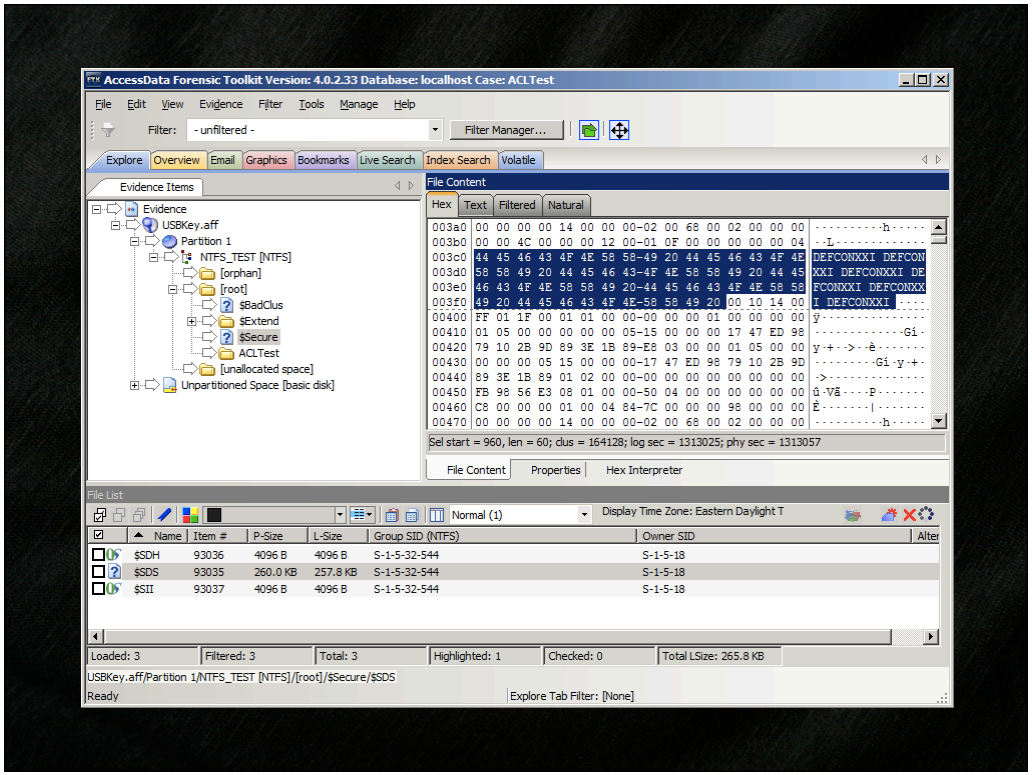
- DEFCONXXI repeated



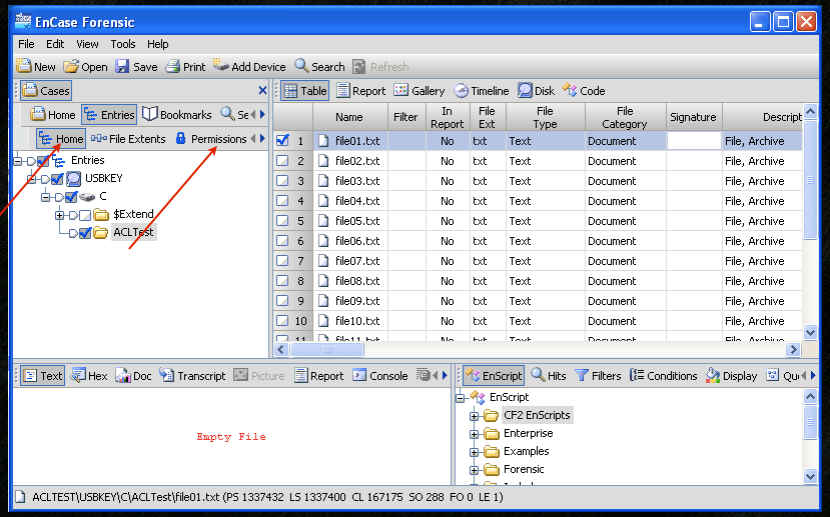

AccessData Forensic Toolkit Version: 4.0.2.33 Database: localhost Case: ACLTest
File Edit View Evidence Filter Tools Manage Help
Filter: - unfiltered - Filter Manager...
Explore Overview Email Graphics Bookmarks Live Search Index Search Volatile
Evidence Items
Evidence
USBKey.aff
Partition 1
NTFS_TEST [NTFS]
[orphan]
[root]
$BadClus
$Extend
$Secure
ACLTest
[unallocated space]
Unpartitioned Space [basic disk]
Properties
NTFS Information
MFT Record Number 36
Record date 7/5/2013 3:41:20 PM (2013-07-05 19:41:20 UTC)
Resident True
Offline False
Sparse False
Temporary False
Owner SID S-1-5-21-2565687063-2636845177-2300264073-1000
Group SID S-1-5-21-2565687063-2636845177-2300264073-513
File Content Properties Hex Interpreter
File List
Normal (1)
Display Time Zone: Eastern Daylight Time (From I
Name Item # P-Size L-Size Group SID (NTFS) Owner SID Alterna...
file01.txt 93042 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file02.txt 93043 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file03.txt 93044 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file04.txt 93045 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file05.txt 93046 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file06.txt 93047 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file07.txt 93048 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file08.txt 93049 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file09.txt 93050 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file10.txt 93051 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file11.txt 93052 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file12.txt 93053 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
file13.txt 93054 0 B 0 B S-1-5-21-2565687063-2636845177-2300264073-513 S-1-5-21-2565687063-2636845177-2300264073-1000
Loaded: 17 Filtered: 17 Total: 17 Highlighted: 1 Checked: 0 Total LSize: 4096 B
USBKey.aff/Partition 1/NTFS_TEST [NTFS]/[root]/ACLTest/file01.txt
Ready
Explore Tab Filter: [None]

Forensic Test - FTK4

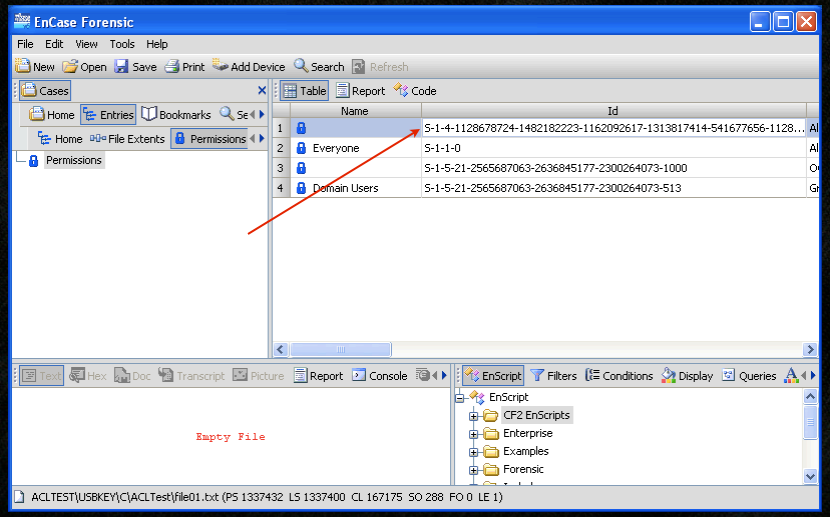



AccessData Forensic Toolkit Version: 4.0.2.33 Database: localhost Case: ACLTest
File Edit View Evidence Filter Tools Manage Help
Filter: - unfiltered - Filter Manager...
Explore Overview Email Graphics Bookmarks Live Search Index Search Volatile
Evidence Items
Evidence
USBKey.aff
Partition 1
NTFS_TEST [NTFS]
[orphan]
[root]
$BadClus
$Extend
$Secure
ACLTest
[unallocated space]
Unpartitioned Space [basic disk]
File Content
Hex Text Filtered Natural
003a0 00 00 00 00 14 00 00 00-02 00 68 00 02 00 00 00
003b0 00 00 4C 00 00 00 12 00-01 0F 00 00 00 00 00 04
003c0 44 45 46 43 4F 4E 58 58-49 20 44 45 46 43 4F 4E DEFCONXXI DEFCON
003d0 58 58 49 20 44 45 46 43-4F 4E 58 58 49 20 44 45 XXI DEFCONXXI DE
003e0 46 43 4F 4E 58 58 49 20-44 45 46 43 4F 4E 58 58 FCONXXI DEFCONXX
003f0 49 20 44 45 46 43 4F 4E-58 58 49 20 00 10 14 00 I DEFCONXXI
00400 FF 01 1F 00 01 01 00 00-00 00 00 01 00 00 00 00
00410 01 05 00 00 00 00 00 05-15 00 00 00 17 47 ED 98
00420 79 10 2B 9D 89 3E 1B 89-E8 03 00 00 01 05 00 00
00430 00 00 00 05 15 00 00 00-17 47 ED 98 79 10 2B 9D
00440 89 3E 1B 89 01 02 00 00-00 00 00 00 00 00 00 00
00450 FB 98 56 E3 08 01 00 00-50 04 00 00 00 00 00 00
00460 C8 00 00 00 01 00 04 84-7C 00 00 00 98 00 00 00
00470 00 00 00 00 14 00 00 00-02 00 68 00 02 00 00 00
Sel start = 960, len = 60; clus = 164128; log sec = 1313025; phy sec = 1313057
File Content Properties Hex Interpreter
File List
Normal (1)
Display Time Zone: Eastern Daylight T
Name Item # P-Size L-Size Group SID (NTFS) Owner SID
$SDH 93036 4096 B 4096 B S-1-5-32-544 S-1-5-18
$SDS 93035 260.0 KB 257.8 KB S-1-5-32-544 S-1-5-18
$SII 93037 4096 B 4096 B S-1-5-32-544 S-1-5-18
Loaded: 3 Filtered: 3 Total: 3 Highlighted: 1 Checked: 0 Total LSize: 265.8 KB
USBKey.aff/Partition 1/NTFS_TEST [NTFS]/[root]/$Secure/$SDS
Ready
Explore Tab Filter: [None]

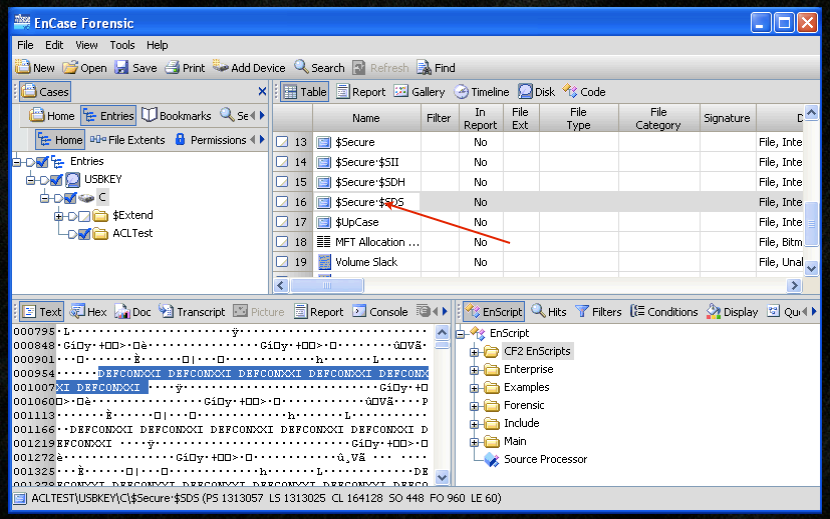
Forensic Test - EnCase 6



Forensic Test - EnCase 6



Forensic Test - EnCase 6



Forensic Detection of ACLEncoding

- Detection of ACLEncoded entries is a manual process
  - (using the most popular forensic tools)
- Detection can be automated with the creation of EnScripts (EnCase's scripting language) and other purpose-built tools
- Unfortunately not enough time to go over these today

Questions and Answers

- If you have questions, see me in the Q&A room for Track 1

- Thanks to Josh, Nick, Joel, Reesh, my family, my friends, my colleagues, and my employer for providing me the time for this research
- Thanks Eugene for seeding the thought in my mind of “How can you hide data on a drive without detection?”

ACLEncode

- Source code Available for download:
- http://www.perklin.ca/~defcon21/ACLEncode.zip

Latest version of Slides

- The latest version of these slides are available online:
- http://www.perklin.ca/~defcon21/aclsteganography.pdf

- This latest version will be available on the DEFCON site soon

References

- http://msdn.microsoft.com/en-us/library/gg465313.aspx
- http://stackoverflow.com/questions/1140528/what-is-the-maximum-length-of-a-sid-in-sddl-format
- http://technet.microsoft.com/en-us/library/cc962011.aspx
- http://msdn.microsoft.com/en-CA/library/ms229078(v=vs.85).aspx
- https://github.com/mosa/Mono-Class-Libraries/blob/master/mcs/class/corlib/System.Security.AccessControl/FileSystemRights.cs
- http://msdn.microsoft.com/en-us/library/system.security.accesscontrol.filesystemrights.aspx
- http://www.ntfs.com/ntfs-permissions-access-entries.htm
- http://www.ntfs.com/ntfs-permissions-security-descriptor.htm
- http://support.microsoft.com/kb/279682