File System & its Forensic Analysis

Prof. Zaheed Shaikh





Objectives

- Investigative Process
 - Analysis Framework
- File Systems
 - FAT
 - NTFS
 - EXT3

Investigative process

Investigative process



Analysis Framework

- Brian Carrier, File System Forensic Analysis
- Data Categories provide a basic reference model
 - Good for comparing different file system types
 - Also allows us to understand how to search using various tool types

Categories

- File System Category
 - Content Category
 - Metadata Category
 - File Name Category
 - Application Category



Analysis by Category

- Categories are important as they allow us to filter and search for files
- For example, if we want to search for all images
 - GIF, JPEG
 - Search for all files ending in .gif or .jpg, OR their fileheader

File System Category

- All file systems have a general structure
- Informs of where to find data structures
 - Think of this as a map
- File system data resides on the first few sectors of the disk
- If corrupted, rebuilding by hand may need to occur
- Small amounts of data hiding can occur due to the sparse usage of the pre-allocated data structures



Content Category

- Actual content of the file
- Contains the majority of the actual data
- Usually organised into standard-sized containers
 - Clusters
 - Data Unit
- Allocation Strategy
 - First Available
 - Next Available
 - Best Fit

Analysis Techniques

- Data unit viewing
 - viewing content of the logical file system address using hex editor
 - dcat
- Logical file system-level searching
 - searches each data unit for know value
- Data unit Allocation Status
 - bitmap data structure of allocated data units (marked as "1") and unallocated data units (marked as "0")

Metadata Category

- Where the descriptive data resides
 - Last access time
 - Data units allocated to the file
- Provides pointers to the address of the Logical File Address
- MAC times
 - <u>M</u>odification, <u>A</u>ccess, <u>C</u>hange



Analysis Techniques

- Metadata Lookup
 - istat shows values from the metadata data structure
- Logical file viewing
 - icat shows the content data for a given metadata structure
 - -s option gives slack space
 - -r attempts to recover deleted files

Linking Data Units to Files

- ifind allows you to discover the rest of the data units allocated to a file
- E.g. if we find something interesting in data unit 3, and want to look at the rest of the file



File Searching

- Logical File Search
 - Looks at the file-address level, takes into account fragmentation
 - Logical File Address allocated units only



Slack Space

- Disks are block-based
- Example File1.txt
 - 700 byte file
 - Needs to allocate full data unit (2,048 bytes)
 - The remaining 1,348 would be slack
- Two interesting areas:
 - Between end of file and end of sector
 - Allocated Data Unit Sectors that contain no file content



Unallocated Metadata

- File names may be deleted, but metadata may still exist
- Examine the metadata as this may contain evidence
- The tool *ils* will list unallocated structures

Metadata Searching

- Allows time-based evaluation of activities
- Timelining events
- MAC
 - Modified
 - Access
 - Change
- Also owner ID and file permissions
- mactime tool in CAINE

File Name Category

- Includes names of files
- Allows a user to find a file by name, instead of its metadata entry
- When recovering files based on file names
 - We are still reliant on the metadata information
 - File names and metadata can get out of sync

File Name Analysis Techniques

- File Listings locate root directory and obtain list of files and corresponding metadata addresses
- File Name Searching file extension filtering and searching
 - ffind resolve metadata to file names
 - When evidence in a data unit is found, search for the metadata unit allocated, search for file name

Application Category

- Non-Essential Data file system jurnals
- Can be application specific data
- Search
 - grep
 - Data carving
 - File type sorting
 - Use file command to determine file types based on file signatures

- An operating system requires long term storage and retrieval
- A mechanism for storing files in hierarchy of files and directories
 - For example, a patient record filing system

- Data
 - Files
 - Directories
- Metadata
 - Time stamps (modify, access, create/change, delete)
 - Owner
 - Security properties
- Structures
 - Superblock/Master File Table/File Access Table
 - inodes/clusters
 - data

- More sophisticated data recovery requires deep knowledge of file system internals
- Structures that manage file system metadata
- Disk layout
- File deletion issues
- Many important file systems
 - DOS / Windows: FAT, FAT16, FAT32, NTFS
 - Unix: ext2, ext3, Reiser, JFS
 - Mac: MFS, HFS, HFS+

File Systems: FAT

- FAT12, FAT16, FAT32
 - different size of addressable cluseter
- Common format for floppy disks (remember those?)
- Limited time/date information for FAT files
 - Last write date/time is always available
 - Creation date/time is optional and may not be available
 - Last access DATE ONLY is optional and may not be available
- Short file names (8.3) on FAT12 and FAT16
- No security features
- Long names for FAT32

FAT: Short Filename Storage

"foo.bar"	->	" FOO	BAR″
"FOO.BAR"	->	`` FOO	BAR″
"Foo.Bar"	->	" FOO	BAR″
"foo"	->	`` FOO	"
"foo."	->	`` FOO	"
"PICKLE.A"	->	"PICKLE	A "
"prettybg.big"	->	"PRETTYB	GBIG″

- Note case is not significant
- "." between primary filename and extension is implied (not actually stored)
- Further, everything is space-padded

FAT: More Dir Entry Details

- Date format:
 - Bits 0–4: Day of month, valid value range 1-31 inclusive.
 - Bits 5–8: Month of year, 1 = January, valid value range 1–12 inclusive.
 - Bits 9–15: Count of years from 1980, valid value range 0–127 inclusive (1980–2107).
- Time Format:
 - A FAT directory entry time stamp is a 16-bit field that has a granularity of 2 seconds
 - Bits 0–4: 2-second count, valid value range 0–29 inclusive (0 58 seconds).
 - Bits 5–10: Minutes, valid value range 0–59 inclusive
 - Bits 11–15: Hours, valid value range 0–23 inclusive

FAT: Long Filenames

- Summary: a kludge to add support without changing short-name handling
- Up to 255 characters in pathname component
- Total pathname no longer than 260
- More supported characters
- Leading/trailing spaces ignored
- Internal spaces allowed
- Leading/embedded "." allowed
- Trailing "." are ignored
- Stored case-sensitive
- Processed case-insensitive (for compatibility)
- File created with short name (uses "~1", "~2", etc. suffix)

FAT Layout

Boot Rese Sector Ar	erved FAT Area	Root Directory	Data Area
------------------------	-------------------	----------------	-----------

	File System	Content	Metadata	File Name	Application
FAT	Boot Sector, FSINFO	Clusters, FAT	Directory Entries, FAT	Directory Entries	N/A

FAT32 Directory Structure

- An ordinary cluster chain
- Directory entry
 - 32 bytes for both files and directories
 - For deleted entries, first byte is set to 0xE5
 - First two entries in subdirectories are . and ..
 - Uses more than one entry to implement long filenames









FAT File Deletion

- First letter of the file is overwritten with 0xE5
- FAT pointers to allocation areas set to zero
 - Indicated that they are ready for re-use

Forensic Analysis - Examples

fsstat - display general details of a file system

FILE SYSTEM INFORMATION

File System Type: FAT16

OEM Name: MSDOS5.0 Volume ID: 0x3abd23ec Volume Label (Boot Sector): NO NAME Volume Label (Root Directory): File System Type Label: FAT16

Sectors before file system: 32

File System Layout (in sectors) Total Range: 0 - 2003935 * Reserved: 0 - 5 ** Boot Sector: 0 * FAT 0: 6 - 250 * FAT 1: 251 - 495 * Data Area: 496 - 2003935 ** Root Directory: 496 - 527 ** Cluster Area: 528 - 2003919 ** Non-clustered: 2003920 - 2003935

fsstat (continue)

METADATA INFORMATION

Range: 2 - 32055046 Root Directory: 2

CONTENT INFORMATION

Sector Size: 512 Cluster Size: 16384 Total Cluster Range: 2 - 62607

FAT CONTENTS (in sectors)

528-559 (32) -> EOF 560-591 (32) -> EOF 592-623 (32) -> EOF 624-687 (64) -> EOF 688-719 (32) -> EOF 720-751 (32) -> EOF **fls** - list file and directory names in a disk image

- d/d 4: Project 2010
- d/d 6: Report2010
- d/d 8: Other
- v/v 32055043: \$MBR
- v/v 32055044: \$FAT1
- v/v 32055045: \$FAT2
- d/d 32055046: \$OrphanFiles

istat - Display details of a meta-data structure (i.e. inode)

```
Directory Entry: 6
Allocated
File Attributes: Directory
Size: 16384
Name: REPORT~1
```

```
Directory Entry Times:Written:Thu Oct 20 06:36:44 2011Accessed:Thu Oct 20 00:00:00 2011Created:Thu Oct 20 06:36:42 2011
```

Sectors: 1488 1489 1490 1491 1492 1493 1494 1495

1496	1497	1498	1499	1500	1501	1502	1503
1504	1505	1506	1507	1508	1509	1510	1511
1512	1513	1514	1515	1516	1517	1518	1519

icat - Output the contents of a file based on its inode number

000000:	2e20	2020	2020	2020	2020	2010	0025	9534	• • • • •	. 4
0000010:	543f	543f	0000	9634	543f	2000	0000	0000	Т?Т?4Т?	
0000020:	2e2e	2020	2020	2020	2020	2010	0025	9534	•••	. 4
0000030:	543f	543f	0000	9634	543f	0000	0000	0000	T?T?4T?	
0000040:	422e	0064	006f	0063	0078	000f	0014	0000	Bd.o.c.x	
0000050:	ffff	ffff	ffff	ffff	ffff	0000	ffff	ffff		• •
0000060:	0143	0075	0073	0074	006f	000f	0014	6d00	.C.u.s.t.or	n.
0000070:	6500	7200	4400	6£00	6300	0000	5£00	3100	e.r.D.o.c	L.
0000080:	4355	5354	4f4d	7e31	444f	4320	0033	9534	CUSTOM~1DOC .3.	. 4
0000090:	543f	543f	0000	2231	543f	2100	5731	0000	T?T?"1T?!.W1	• •
00000a0:	4230	002e	0064	006f	0063	000f	00f4	7800	B0d.o.c	ζ.
00000b0:	0000	ffff	ffff	ffff	ffff	0000	ffff	ffff	•••••	• •
00000c0:	0143	0075	0073	0074	006f	000f	00f4	6d00	.C.u.s.t.or	n.
00000d0:	6500	7200	4400	6£00	6300	0000	5£00	3100	e.r.D.o.c	L.
00000e0:	4355	5354	4f4d	7e32	444f	4320	0038	9534	CUSTOM~2DOC .8.	. 4
00000f0:	543f	543f	0000	8533	543f	2200	2433	0000	T?T?3T?".\$3	• •
0000100:	4231	002e	0064	006f	0063	000f	00d4	7800	B1d.o.c	۲.
0000110:	0000	ffff	ffff	ffff	ffff	0000	ffff	ffff	•••••	• •
0000120:	0143	0075	0073	0074	006f	000f	00d4	6d00	.C.u.s.t.or	n.
0000130:	6500	7200	4400	6£00	6300	0000	5£00	3100	e.r.D.o.c	L.
0000140:	4355	5354	4f4d	7e33	444f	4320	0047	9534	CUSTOM~3DOC .G.	. 4
0000150:	543f	543f	0000	8133	543f	2300	4346	0000	T?T?3T?#.CF.	• •

FAT Allocation Table

Media	type	Partition state	Cluste	r 2 Clus	ster 3 C	luster 4	Cluster	5 Cluste	r 6
	7	\downarrow		4					
0000000:	f8ff	ffff	ffff	ffff	ffff	0600	ffff	ffff	
0000010:	ffff	ffff	ffff	ffff	ffff	ffff	ffff	ffff	
0000020:	ffff	ffff	ffff	ffff	ffff	ffff	ffff	ffff	
0000030:	1900	ffff	ffff	ffff	ffff	ffff	ffff	ffff	
0000040:	ffff	ffff	ffff	2400	ffff	ffff	ffff	ffff	\$
0000050:	ffff	ffff	ffff	ffff	ffff	ffff	ffff	ffff	
0000060:	ffff	ffff	ffff	ffff	ffff	ffff	3700	ffff	7
0000070:	ffff	ffff	ffff	ffff	ffff	ffff	ffff	4000	
0000080:	4100	ffff	4300	4400	4500	4600	4700	4800	AC.D.E.F.G.H.
0000090:	4900	ffff	4b00	4c00	4d00	ffff	4f00	ffff	IK.L.MO
00000a0:	5100	5200	5300	ffff	0000	0000	0000	0000	Q.R.S
00000b0:	0000	0000	0000	0000	0000	0000	0000	0000	
00000c0:	0000	0000	0000	0000	0000	0000	0000	0000	
00000d0:	0000	0000	0000	0000	0000	0000	0000	0000	
00000e0:	0000	0000	0000	0000	0000	0000	0000	0000	
00000f0:	0000	0000	0000	0000	0000	0000	0000	0000	
0000100:	0000	0000	0000	0000	0000	0000	0000	0000	