File System Analysis Using TSK & Autopsy

Carrier, Chapter 8

© 2022, D. Nelson, W. Lidinsky

Introduction and Perspective

Carrier created a file system reference model

- *Goal: Organize and standardize the analysis of file systems*
- *Carrier seems to have created this model while thinking mostly of Unix and Linux types of file systems*
  - Consequently, it's a struggle to fit NTFS and especially FAT into this model
- *TSK: the TSK tools for analyzing file systems are organized around this reference model*
- *It helps to know something about this reference model*
  - In order to understand and analyze Unix and Linux FSs
  - In order to more easily use TSK's file system analysis tools

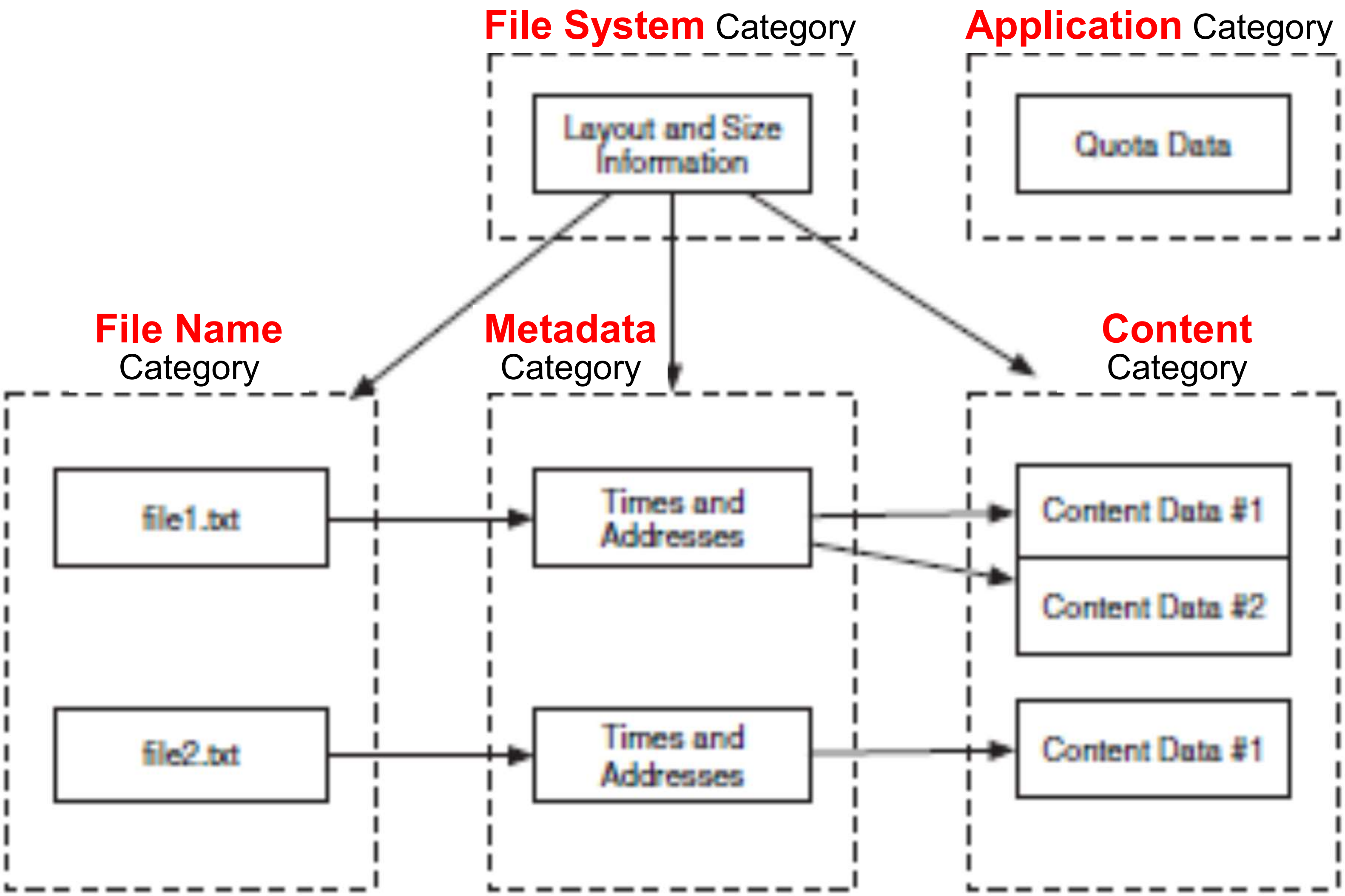
Carrier's Reference Model

Divides a file system into five categories

- *File System*
- *Content*
- *Metadata*
- *File Name*
- *Application*

Many (but not all) TSK tools are based on these categories

Carrier's Reference Model



File System Category

Defines the structure of the entire file system; its layout

Locations of parts and subparts of the file system

Size or max. size of each of the parts and subparts

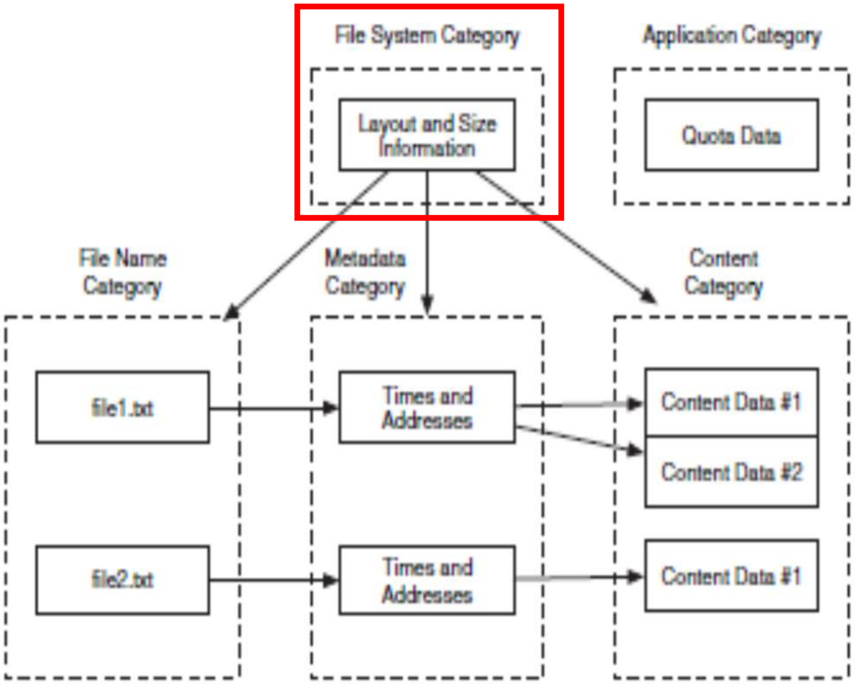
Performance configuration information perhaps

Examples

ExtX: Superblock, Group Descriptor

NTFS: $Boot, $Volume, $AttrDef

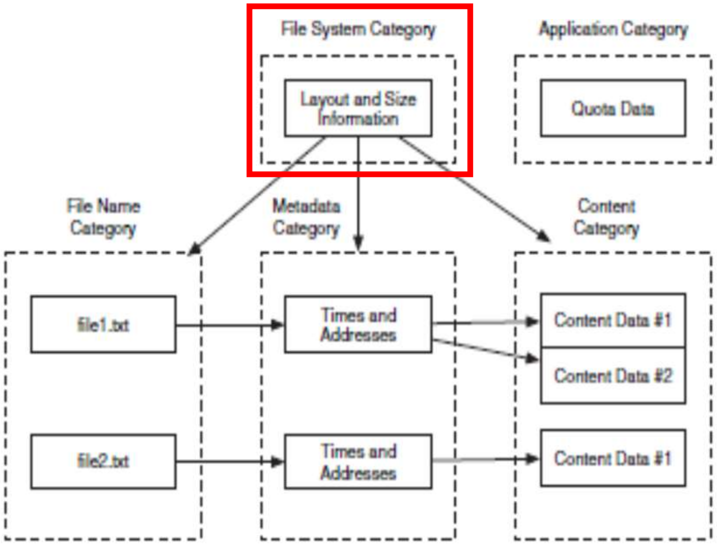
FAT: Boot sector



TSK Tools by Category
File System Category

File System (1 tool, "f" prefix)

- `fsstat` *: Displays the boot sector, superblock or other info about the file system*
  - Specific to the particular file system
  - Output differs depending upon the file system



Content Category

Contains the actual content of the files

Usually organized as groups of fixed size containers

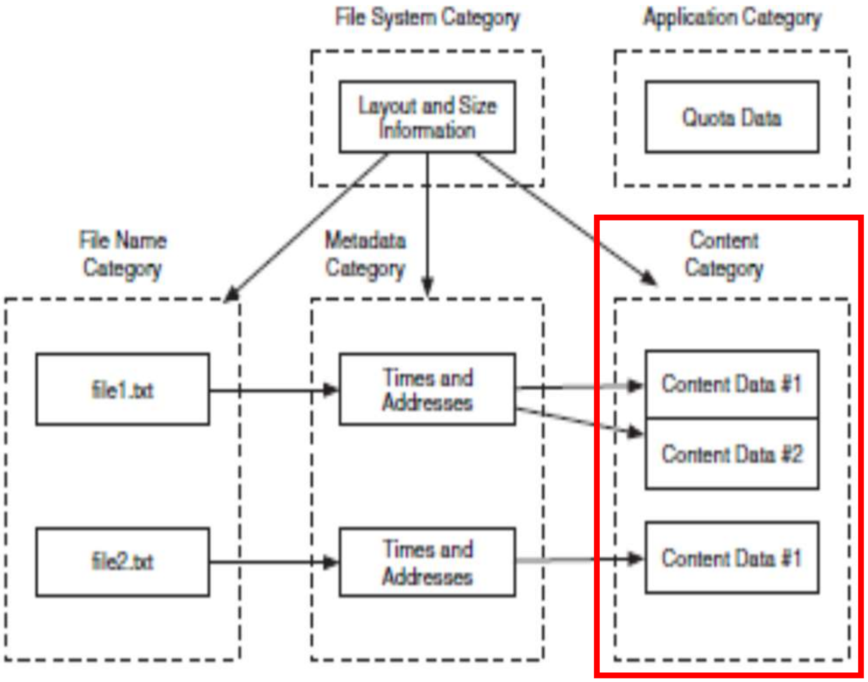
Examples

NTFS: Clusters, $Bitmap

ExtX: Blocks, Block Bitmap

FAT: Clusters, FAT

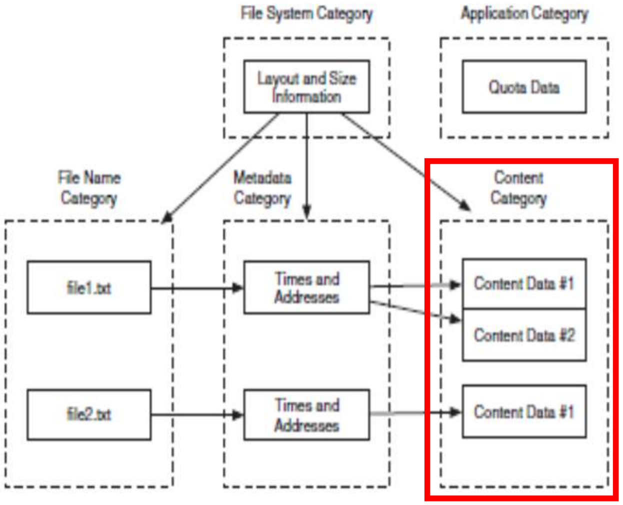
Carrier's term: "data units"



TSK Tools by Category
Content Category

Content (4 tools, "blk" prefix; previously was "d' prefix)

- **`blkls`** : *Lists the contents of data units*
  - Options for data unit listing: allocated, unallocated, offset location, slack etc.
- **`blkcalc`** : *Works with* `blkls`. *Determines the original data unit location of data found in the output of* `blkls`
- **`blkstat`** : *Displays allocation status of specified data units*
- **`blkcat`** : *Displays contents or stats of a specified data unit*
  - Options for hex, ascii. Sort of like xxd for specified data unit



Metadata Category

Contains the metadata for a file

Examples

Locations in the file system of each file

Size of the file

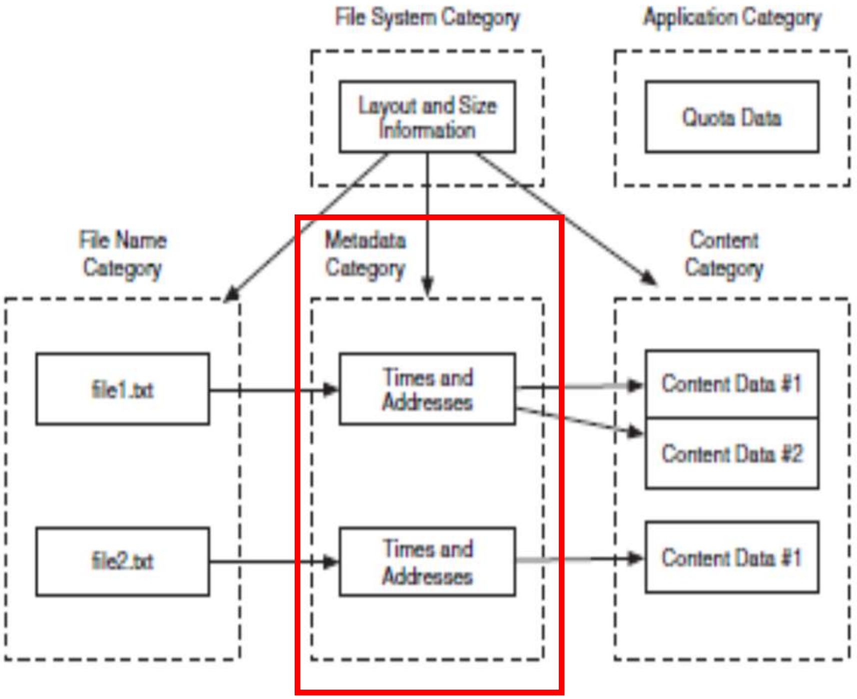
Times & Dates

Created, last read, last modified

FAT: Root directory entries

NTFS: Several MFT entries

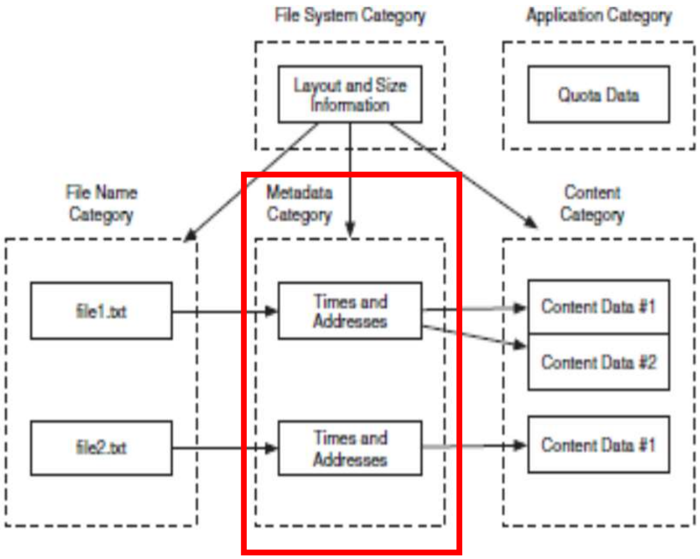
ExtX: inode, inode bitmap



TSK Tools by Category
Metadata Category

Metadata (4 tools, "i" prefix)

- **`istat`** : *Displays the details of a specific metadata entry*
  - e.g., Details of a NTFS MFT entry or a ExtX inode
- **`ils`** : *Lists metadata entries, either unallocated, orphan or all entries*
  - e.g., All MFT entries or all inodes
- **`ifind`** : *Finds the metadata entry that allocated a specific data unit*
  - If a cluster contains interesting stuff, **`ifind`** locates the MFT entry
- **`icat`** : *Lists the contents of a data unit based upon cluster or inode number*



File Name Category

Entry contains the name of the file and pointers to location of metadata

Usually organized as a directory of file names & pointers

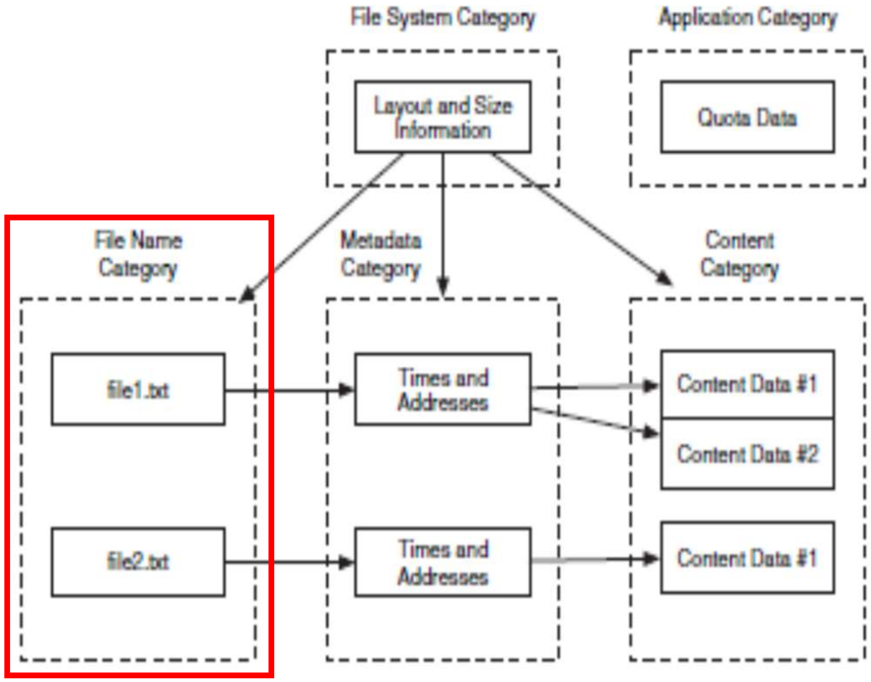
Examples

FAT: Root Directory entries

But the Root Dir. entries also contains the metadata

ExtX: Directory entries

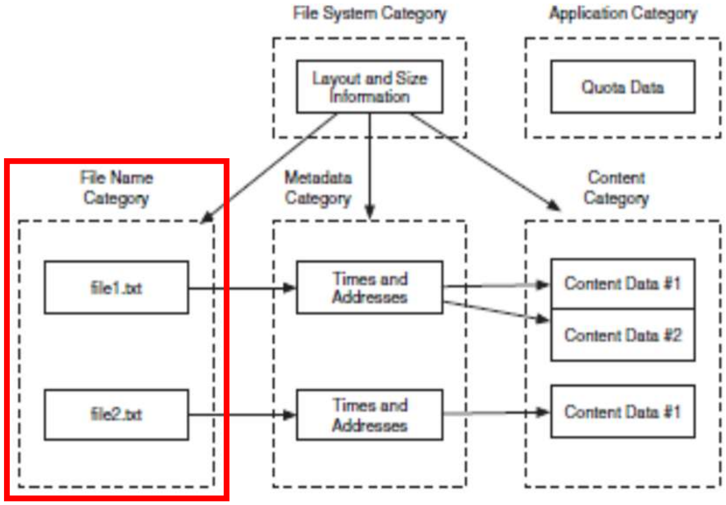
NTFS: $FILENAME, IDX_ROOT, $BITMAP, etc.



TSK Tools by Category
File Name Category

File Name (2 tools, "f" prefix)

- `ffind` : *Displays the name of a file or directory for a specific cluster or inode*
- `fls` : *Lists the file and/or directory names*
  - Recently deleted files and directories can be listed



Application Category

Sort of ad hoc information or information that could be located outside the file system but is part of a specific file system

Examples

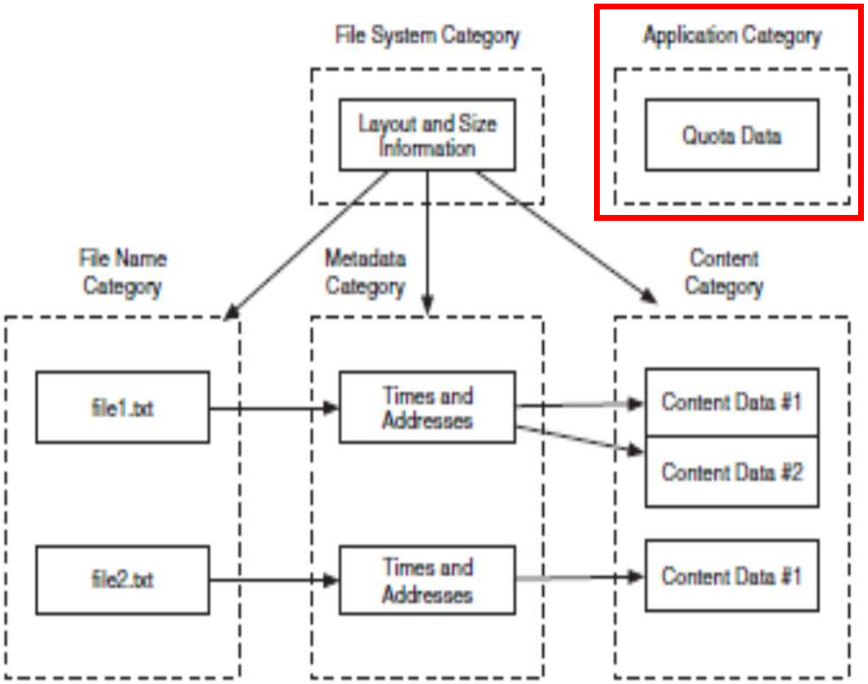
User quotas

File system journals

FAT: None

ExtX Journal

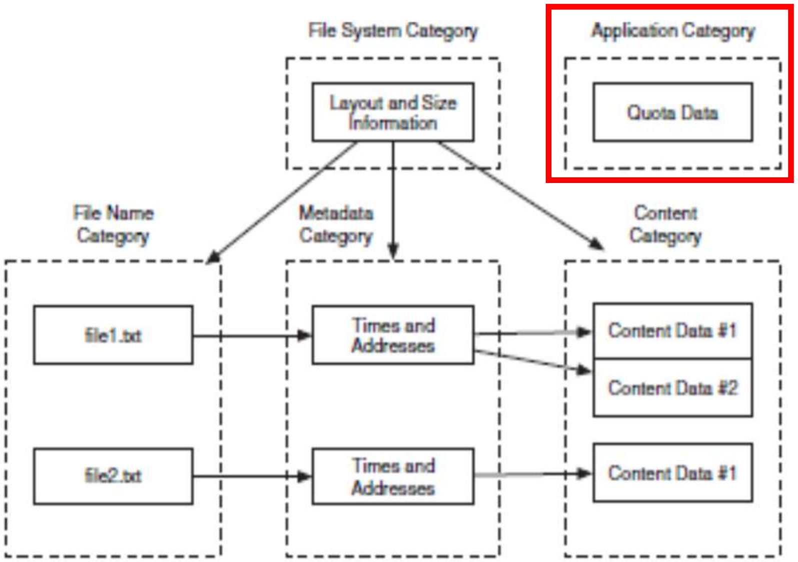
NTFS: Journal, File system encryption, etc.



TSK Tools by Category
Application Category

Application (2 tools, "j" prefix)

- *Work only with Ext3 and some other journaling FSs*
- `jls` : *Lists the contents of the file system journal*
- `jcat` : *Displays the contents of a specific journal entry*



Other TSK Tools

No Category

`hfind` : Looks up known files based upon hash value

Inputs: md5sum of unknown

NIST National Software Reference Library (NSRL)

Output: *The known file, if it is in the NSRL Library*

`sigfind` : Searches for a specific hex string in an image file

`sorter` : Sorts files in a file system image into file signature value types

`mactime` : Uses **`fls`** & **`ils`** tool output to generate a timeline of file activity

Sidebar: Data Carving

Searches for signatures in unknown data units that correspond to the beginning and end of known file types

Often is used on unallocated data units in order to recover files that do not have metadata structures pointing to them

e.g., MSWord, jpeg, PDF and many other file types have known beginning and ending structures

Other Non-TSK Carving Tools
Application Category

`file` : Can identify the structure on many unknown files

- *Based upon a self-contained database of signature values*
- *Sort of a lightweight carving tool*

`lazarus` : Processes an entire file system image, executing **`file`** on each sector

- *Contiguous sectors having the same signature values are grouped*
- *Lists each sector or group and its signature value*

Other Non-TSK Carving Tools

Application Category

`foremost` : Heavy duty carving based upon signatures

- *Analyzes entire file system (raw or image)*
- *Signatures contain*
  - Known header info
  - Header case sensitivity
  - Known footer info
  - Max. file size
  - Usual file name extensions

Data Structures Associated With Each Reference Model Category

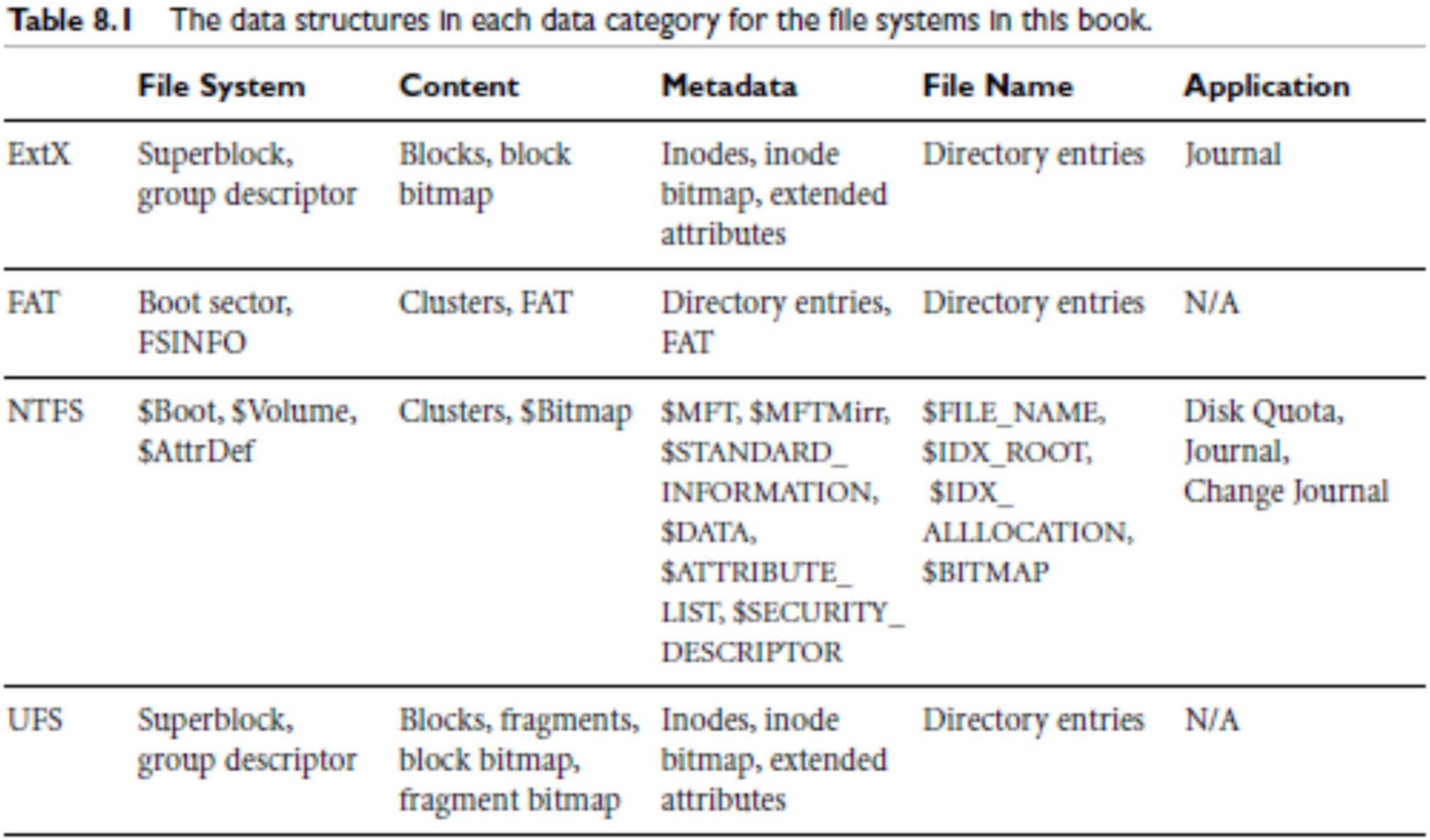


Table 8.1 The data structures in each data category for the file systems in this book.

	File System	Content	Metadata	File Name	Application
ExtX	Superblock, group descriptor	Blocks, block bitmap	Inodes, inode bitmap, extended attributes	Directory entries	Journal
FAT	Boot sector, FSINFO	Clusters, FAT	Directory entries, FAT	Directory entries	N/A
NTFS	$Boot, $Volume, $AttrDef	Clusters, $Bitmap	$MFT, $MFTMirr, $STANDARD_ INFORMATION, $DATA, $ATTRIBUTE_ LIST, $SECURITY_ DESCRIPTOR	$FILE_NAME, $IDX_ROOT, $IDX_ ALLLOCATION, $BITMAP	Disk Quota, Journal, Change Journal
UFS	Superblock, group descriptor	Blocks, fragments, block bitmap, fragment bitmap	Inodes, inode bitmap, extended attributes	Directory entries	N/A

Category Used Based Upon Analysis Need

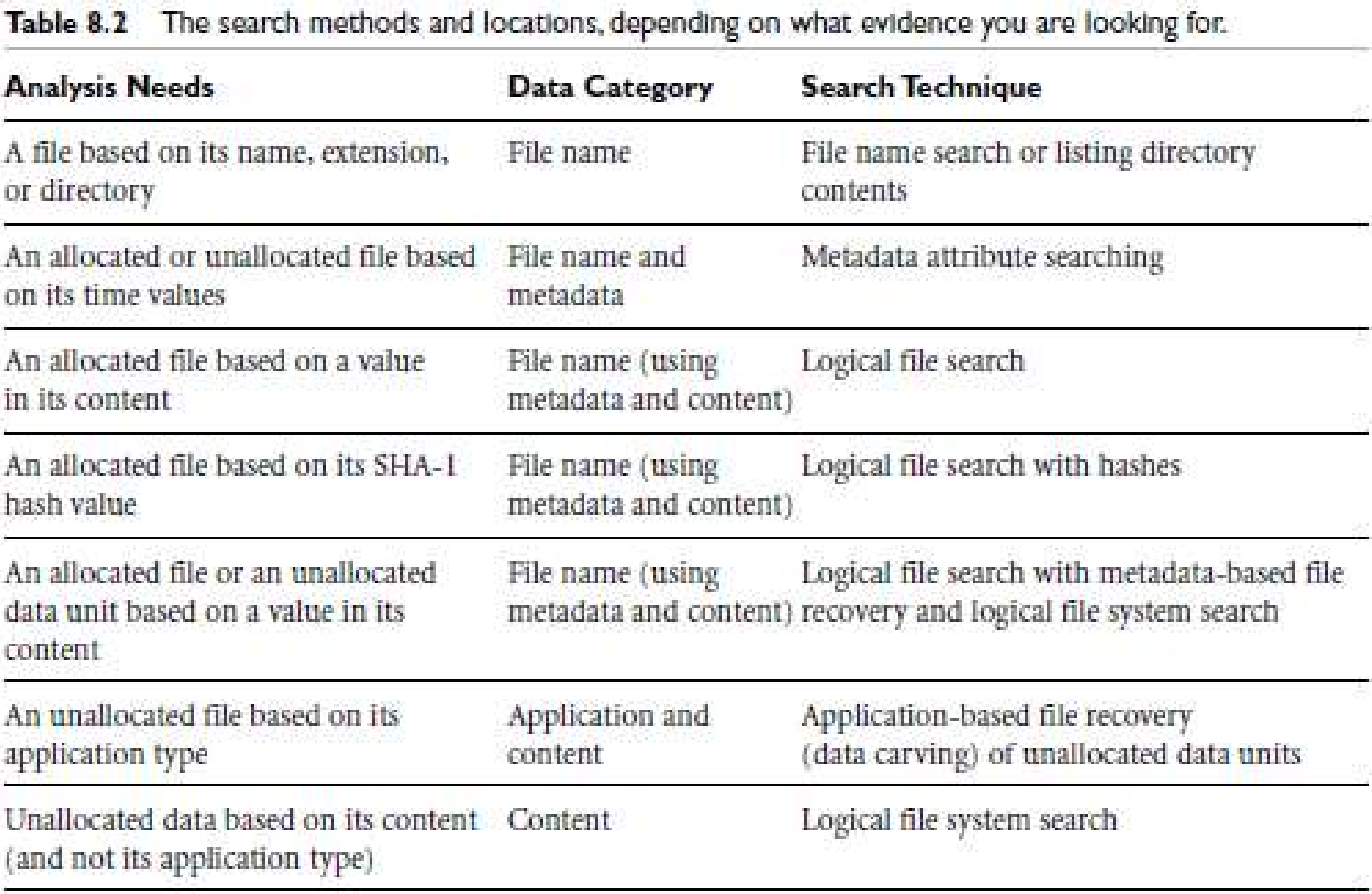


Table 8.2 The search methods and locations, depending on what evidence you are looking for.

Analysis Needs	Data Category	Search Technique
A file based on its name, extension, or directory	File name	File name search or listing directory contents
An allocated or unallocated file based on its time values	File name and metadata	Metadata attribute searching
An allocated file based on a value in its content	File name (using metadata and content)	Logical file search
An allocated file based on its SHA-1 hash value	File name (using metadata and content)	Logical file search with hashes
An allocated file or an unallocated data unit based on a value in its content	File name (using metadata and content)	Logical file search with metadata-based file recovery and logical file system search
An unallocated file based on its application type	Application and content	Application-based file recovery (data carving) of unallocated data units
Unallocated data based on its content (and not its application type)	Content	Logical file system search

Autopsy

Forensic software using TSK and other tools

Original Autopsy
A "sort of" GUI for TSK

Brian Carrier originally created a sort of GUI that used TSK tools

- *Named* **Autopsy Forensic Browser (AFB)**
- *The* **AFB** *interface was originally a web browser*
- *Original ran on most browsers*
  - Natively used TSK-based scripts on Linux and WinXP sp ≥ 2 and after

Original ***AFB*** was a set of scripts that were used via a web browser

- *It used the TSK tools plus some Linux distro tools plus other software tools to analyze an drive, partition or file image*

Helped organize a forensic analysis

It's free.

Evolution of Autopsy

Eventually AFB was rewritten as a native application that would run on Linux, Windows and OS X

Uses TSK tools, Linux distro tools, and other software tools to analyze a drive, partition or file

Continues to be free.

Comments About Autopsy

- AFB helps organize a forensic analysis
  - *Don't need to know much about TSK and other command line tools*
    - However, in certain situations, such knowledge is useful
- Good at timeline analysis
  - *Describes what files were created, modified, accessed and changed in an orderly way*
- Good at searching and displaying context around search terms in readable format
- Rendering is marginal
  - *Design approach: use external viewer*
- As new TSK and other free cyber forensic tools become available, they have been selectively integrated into Autopsy

Autopsy

Autopsy is the most widely-used open-source forensic suite in use today

- *Excellent support for the key forensic tools available*
- *Extensible*
- *Customizable*
- *Interoperable*
- *Updated frequently*
- *High marks from cyber forensic specialists*

We'll be discussing and use Autopsy a great deal in the weeks to come