Mac Forensic Tools Using The Sleuth Kit

Rob Joyce
rob@atc-nycorp.com

Sleuth Kit and Open Source Forensics Conference
June 9, 2010

ATC-NY
Architecture Technology Corporation

Topics

- Current state of Mac OS X & HFS+ support in Sleuth Kit
- How Mac Marshal™ makes use of Sleuth Kit
- Possible directions for future Sleuth Kit work

Current state of Mac OS X and HFS+ support in Sleuth Kit

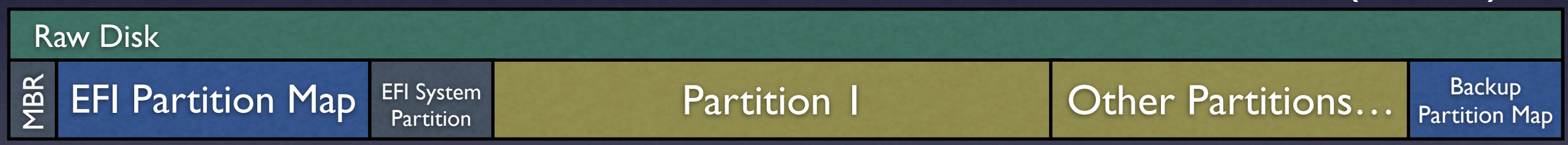
Mac OS X Support

- Running Sleuth Kit
  - Sleuth Kit builds and runs normally on OS X machines, both PowerPC and Intel, 32- and 64-bit
  - Optional libraries (libewf, AFF, etc.) build normally too
- Examining Mac disks
  - Mac Partition Tables
  - Mac File System (HFS+)

Partition Table Support

- PowerPC Mac boot disks: Apple Partition Map

- Intel Mac boot disks: GUID Partition Table (GPT)



- Sleuth Kit supports both natively
  - Sleuth Kit v3.1.0 simplifies analysis of GPT disks: no longer need to add '`-t gpt`' hint to distinguish from traditional MBR disks

File System Support

- HFS+ is the dominant Mac OS X file system
  - Legacy HFS (System 8 and older) is not supported by Sleuth Kit
  - Sleuth Kit can read HFS+ file systems wrapped in an HFS compatibility layer (still occasionally done on external disks)
- HFS+ in Sleuth Kit (re-)enabled in v3.1.0
  - HFS+ support had languished in the 2.0x days and was disabled
  - Much work by Brian Carrier (Basis), Rob Joyce and Judson Powers (ATC-NY) to re-write for new APIs and fix bugs/limitations
  - Sleuth Kit also supports HFSX, the variant used on iPhones

HFS+ Support

- Certain SK features don't yet work for HFS+
  - Deleted files
  - Journal information
- Certain HFS+ features aren't yet exposed in SK
  - Resource forks
  - HFS+ extended attributes
  - Hard links (both file and *directory* links, used by Time Machine)

Snow Leopard HFS+

- Mac OS X 10.6 added *transparent file compression* to HFS+
  - Generally only used at install time for system files, but there are user-land tools to generate compressed files
- File system analysis tools (including Sleuth Kit) don't yet support this file system-level compression

Compression Example

```
root# cd /var/db/dslocal/nodes/Default/groups/

root# ls -li staff.plist
7981 -rw-------  1 root  wheel  771 Jul  2  2009 staff.plist

root# /tmp/sleuthkit-3.1.2/icat /dev/rdisk0s2 7981

root# cat staff.plist
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
   ...
```

How Compression Works

- File's data fork is *empty*
  - Forensic tools such as Sleuth Kit *will show an empty file*
- HFS+ extended attribute *com.apple.decmpfs* has a compression header that declares:
  - What kind of compression is used (uncompressed or ZLib)
  - Where the compressed data is stored (in the same extended attribute or in the resource fork)

How Mac Marshal makes use of Sleuth Kit

Mac Marshal

A forensic tool to analyze Mac disk images, for use in

- Triage Phase

  Discovering the lay of the land of a Mac hard drive, focusing an investigation

- Analysis Phase

  Extracting usage information left by the operating system and common Mac OS applications

www.macmarshal.com

Mac Marshal+Sleuth Kit

MAC MARSHAL UI (JAVA)

Safari Usage Analysis | Mail Usage Analysis | iChat Usage Analysis | ...

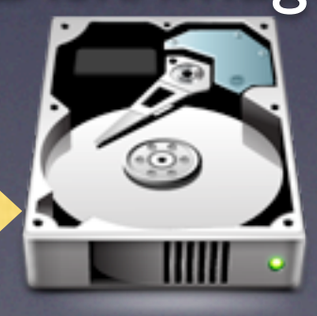
Disk Triage | Application/OS Analysis

Disk Processing Modules | File Extraction & Parsing

SLEUTH KIT

Sleuth Kit Commands | Custom Cmds

libtsk



Icon from the Crystal Project, www.everaldo.com

Stock Sleuth Kit Tools

- Tools used during triage
  - `mmls` - partition table information for triage
  - `img_stat` - disk image info during triage
  - `blkcat` - read raw blocks from the disk image
  - `fsstat` - file system information during triage
- Tools used during application/OS analysis
  - `ifind` - look up inode number for path
  - `fls` - get directory contents
  - `istat` - read metadata about an inode
  - `icat` - read contents of a file

Custom SK Tools

- Modifications to standard tools
  - `istat` - also print file path (if available) on HFS+
  - `icat` - also accept paths by name, optional output file name, progress reporting
- New tools
  - `atc_fsearch` - find files matching specified criteria
  - `atc_icp` - recursively copy a directory hierarchy (for efficiency reasons)

Using SK as a Back End

- Why?
  - Support for multiple image formats, partitioning schemes, etc.
  - Visibility of data not provided by native file system drivers
  - File system corruption won't crash the kernel
- Linking with libtsk directly
  - Fastest execution; only need to read the catalog info once
- Using Sleuth Kit command-line tools
  - Slower; catalog is read on each invocation
  - Allows practitioners to reproduce intermediate results
  - Subject to output format changes with new SK versions

Possible directions for future Sleuth Kit work

HFS+ Improvements

- Implement HFS+-specific aspects of
  - Deleted files
  - Journal information
  - Resource forks (likely with same API as NTFS streams)
  - HFS+ extended attributes
- Snow Leopard's compressed files
- Hard links (both file and *directory* links)

App Integration Ideas

- More easily parseable output
  - Especially when file names contain special characters (e.g., \n)
- Progress reporting monitoring for slow operations
- Speed
  - Front-ends may have to call Sleuth Kit tools multiple times
  - Increasing the image I/O cache from 4 to 128 64K buffers helps a lot with HFS+ (so much of the catalog can be cached)

Questions?