available at www.sciencedirect.com

journal homepage: www.elsevier.com/locate/diin

MEGA: A tool for Mac OS X operating system and application forensics

Robert A. Joyce[*], *Judson Powers*, *Frank Adelstein*

ATC-NY, 33 Thornwood Dr. Suite 500, Ithaca, NY 14850, United States

Keywords:
Mac OS X
Computer forensics
Spotlight
Disk image analysis
Application analysis

ABSTRACT

Computer forensic tools for Apple Mac hardware have traditionally focused on low-level file system details. Mac OS X and common applications on the Mac platform provide an abundance of information about the user's activities in configuration files, caches, and logs. We are developing MEGA, an extensible tool suite for the analysis of files on Mac OS X disk images. MEGA provides simple access to Spotlight metadata maintained by the operating system, yielding efficient file content search and exposing metadata such as digital camera make and model. It can also help investigators to assess FileVault encrypted home directories. MEGA support tools are under development to interpret files written by common Mac OS applications such as Safari, Mail, and iTunes.

1. Introduction

The increasing popularity of Apple Macintosh hardware, particularly that using Intel x86-compatible processors, provides new challenges and data-gathering opportunities for forensic examiners. Various estimates of Mac market share hover between 5 and 14%, and that number is growing quarter by quarter (AppleInsider, 2008). The New York State Computer Crime Unit sees approximately 5–10% Macs, almost exclusively Mac OS X, in their investigations. Some niche markets have even greater penetration; Princeton University reports that 45% of new computer purchases on campus in 2006 were Macs (Daily Princetonian, 2006).

Yet forensics on Mac OS X-based systems is an immature field with relatively few tools available. Those tools that do exist focus mainly on low-level file system analysis and provide little interpretation of the files or insight into user activities (at least not without substantial effort by the investigator). The complexity of Mac-based investigations is compounded by the fact modern Macs often run multiple operating systems (such as Microsoft Windows), either dual-boot via Apple's Boot Camp software or in a virtual machine such as Parallels or VMware Fusion.

With the assistance of the National Institute of Justice (NIJ), we are developing an extensible Macintosh Evidence Gathering and Analysis (MEGA) tool suite that allows investigators to graphically assess and collect data on dual-boot Mac systems, and to gather and analyze forensically relevant data specific to Mac OS X and common applications on the platform.

Section 2 describes existing forensic research and tools available for Mac OS X. Section 3 summarizes the design and implementation approach taken for MEGA. The Mac "Spotlight" search index, maintained by Mac OS X 10.4 and later, can dramatically speed investigators' search for particular files; the acquisition and forensic implications of this metadata are discussed in Section 4. In addition to Spotlight queries, MEGA can also handle FileVault encrypted home directories, which are discussed in Section 5. Finally, Section 6 describes other features of MEGA that are currently under development, as well as future directions for Mac OS X forensic research.

[*] *Corresponding author.*
E-mail addresses: rob@atc-nycorp.com (R.A. Joyce), jpowers@atc-nycorp.com (J. Powers), fadelstein@atc-nycorp.com (F. Adelstein).

1742-2876/$ – see front matter © 2008 Digital Forensic Research Workshop. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2008.05.011

2. Related work

Existing forensic tools with explicit support for Mac OS X are few, and typically focus on low-level file system forensics. Guidance Software's EnCase, AccessData's FTK, and some versions of the open source Sleuth Kit can read Apple HFS+-formatted disk images; EnCase can also perform a limited snapshot of live OS X machines. BlackBag's Mac Forensics Software and subrosasoft's MacForensicsLab also focus on file-based data recovery and extraction of files of evidentiary value (BlackBag Technologies Inc.; Subrosasoft's MacForensicsLab). Cyber Security Technologies's live forensic tool, OnLineDFS, supports Mac OS X but does not take advantage of Mac-specific forensic data (OnLineDFS). In addition, with OS X's Unix underpinnings, Unix-based tools and scripts allow an expert examiner to gather generic information about files and processes.

None of the existing tools addresses dual-boot Windows/OS X machines, made possible with Apple's Boot Camp software, nor do they handle the increasingly popular Windows/Linux virtual machines under OS X on Intel, such as Parallels and VMware Fusion. Further, no existing tool addresses the new, forensically critical data available in OS X: Spotlight searches and caches, Safari web browser information, encrypted home directories, iPods associated with a machine, and so on.

Recent research, such as that of Kubasiak (2007, 2008) and MAC OS, has explored the forensic implications of data written by the Mac OS X operating system and common applications. Much more data of forensic relevance is documented in reference materials by Apple Computer Inc. (2006) and Singh (2006). Recent work has also shown that dictionary attacks are effective in decrypting user directories encrypted with Apple's FileVault feature (Appelbaum and Weinmann, 2006). Finally, the limited existing research on iPod forensics indicates that details of the machine ''associated'' with an iPod are maintained on the iPod itself, in addition to any address book entries or calendar items automatically synced by iTunes (Marsico and Rogers, 2005; Slay and Przbilla, 2007). This data could prove a vital link between a music player found on the scene and a computer used at home.

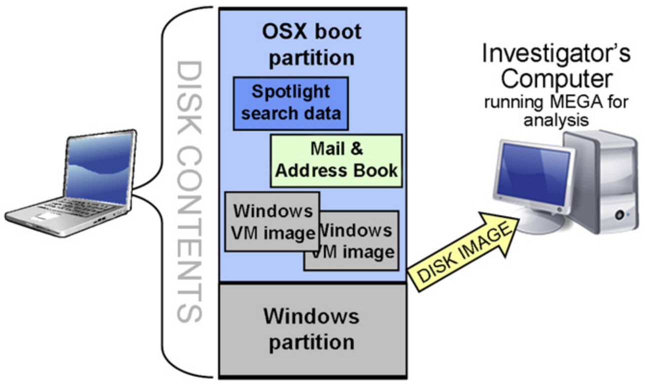


Fig. 1 – Analysis of a dual-boot Mac OS system with Windows virtual machines.

3. Approach

MEGA extracts and analyzes OS X-specific forensic information from a seized disk image, as shown in Fig. 1. (MEGA could also operate in a ''live'' forensics setting – executing directly on the running machine to be analyzed – but our initial attention is on after-the-fact analysis.) With a focus on interpreting and analyzing files written by the operating system and common OS X applications, MEGA will provide insight into user activities that is not possible – or is substantially more tedious to obtain – with low-level disk image tools.

MEGA uses the open source Sleuth Kit tools to extract partition information and files, allowing MEGA to read dd (raw), EnCase, and FTK format disk images (Carrier). Most of MEGA's extraction and analysis work is performed by executing command-line tools – both The Sleuth Kit and our own custom tools. MEGA performs its tasks in a forensically sound, reproducible manner, including fully documenting the investigative process with its own audit log. It can also produce RTF, PDF, or HTML formatted reports of the data gathered, along with investigators' notes.

The initial ''triage'' mode in MEGA allows an investigator to quickly assess the operating system(s) installed on a Mac OS X disk image or machine – both bootable partitions and virtual machine images within the file systems. This information allows the forensic examiner to quickly pinpoint the disk partitions most likely of interest and to apply operating system-specific tools to those partitions. Fig. 2 shows an example of the triage results for a machine with two Windows virtual machines.

Once triage is complete, MEGA employs command-line analysis tools on files automatically extracted from the disk image, then presents the results graphically. MEGA is designed to be extensible, allowing new analysis tools to be created and added dynamically (e.g., to extract ''recent addresses'' from a new type of e-mail client).

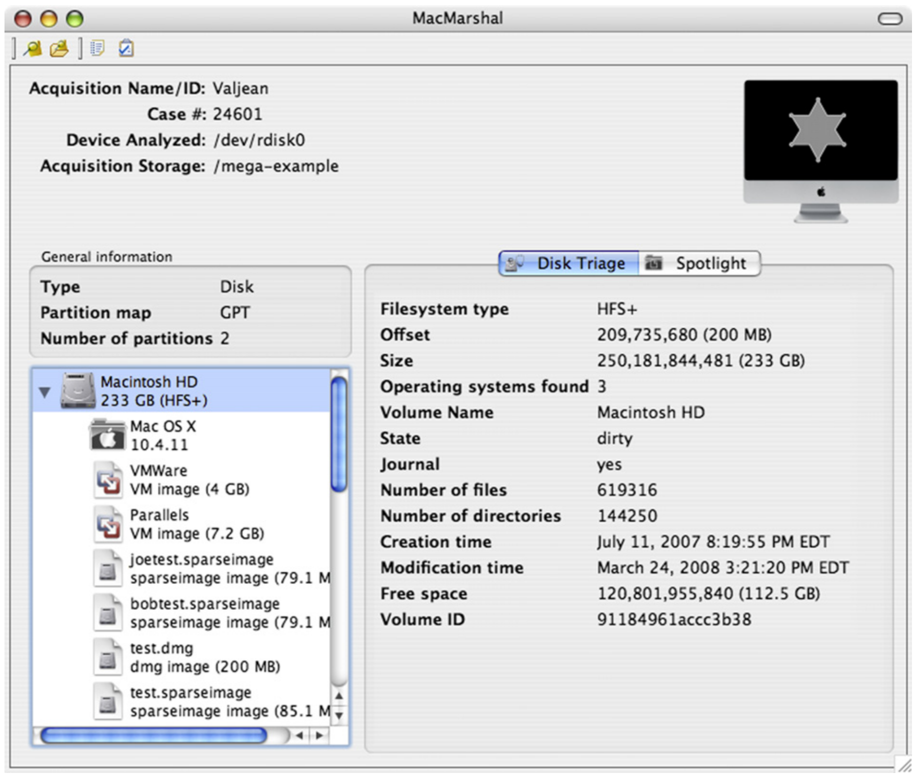
Machine-wide, MEGA can make use of Spotlight indexes written by the operating system; this feature is described in Section 4. MEGA can also detect and analyze user home directories encrypted using the Mac OS X FileVault feature, as will be discussed in Section 5.

4. Spotlight file metadata

Spotlight is the metadata indexing system introduced in Mac OS X 10.4. At the highest level, Spotlight is responsible for acquiring, storing, indexing, and performing searches on file metadata (Apple Computer Inc., 2007a). By default, the Spotlight indexer runs continuously in the background, tracking modified files and updating the metadata index in real-time.

For investigators, Spotlight metadata should be treated as advisory only, used in conjunction with other search technologies; the absence of a file from the Spotlight index is not exculpatory evidence. Yet the Spotlight index is invaluable in speeding searches for files based on sophisticated content or metadata criteria.

Spotlight encapsulates two other capabilities: searching for files based on file system metadata, such as filename, size, and modification date; and indexing and searching the content of



Acquisition Name/ID: Valjean
Case #: 24601
Device Analyzed: /dev/rdisk0
Acquisition Storage: /mega-example

Type	Disk
Partition map	GPT
Number of partitions	2

Macintosh HD
233 GB (HFS+)
Mac OS X
10.4.11
VMWare
VM image (4 GB)
Parallels
VM image (7.2 GB)
joetest.sparseimage
sparseimage image (79.1 M
bobtest.sparseimage
sparseimage image (79.1 M
test.dmg
dmg image (200 MB)
test.sparseimage
sparseimage image (85.1 M

Filesystem type	HFS+
Offset	209,735,680 (200 MB)
Size	250,181,844,481 (233 GB)
Operating systems found	3
Volume Name	Macintosh HD
State	dirty
Journal	yes
Number of files	619316
Number of directories	144250
Creation time	July 11, 2007 8:19:55 PM EDT
Modification time	March 24, 2008 3:21:20 PM EDT
Free space	120,801,955,840 (112.5 GB)
Volume ID	91184961accc3b38

Fig. 2 – Example of MEGA triage results with a single-boot machine containing two virtual machines.

text-based files, a feature formerly available through Search Kit in OS X 10.3. The behavior of these two capabilities differs slightly from the behavior of other Spotlight metadata, despite being integrated into Spotlight.

When a file is created or modified, it is processed by the appropriate metadata importer, which generates metadata from the file. This metadata is then stored and indexed in the Spotlight database, located at the root of the volume containing the indexed file. There are few restrictions on what data can be generated by a metadata importer. However, there are a number of common metadata attributes used by different importers. The Mac OS includes metadata importers for many common file types, including common video, audio, and image formats.

Apple applications such as Mail, Safari, iTunes, and iCal ensure that their files have useful Spotlight metadata. They also store their data in such a way that individual entities, such as Web bookmarks, e-mail messages, and address book entries, appear in Spotlight. Other applications may not be designed with Spotlight in mind. For example, bookmarks in the Firefox web browser do not appear as individual entities in Spotlight (but the content of the Firefox `bookmarks.html` file *will* be indexed, so bookmarked Firefox URLs can still be found via Spotlight).

Not all files are indexed by Spotlight. Indexing can be disabled for a specified folder or volume, and Spotlight can be disabled altogether. Invisible files (such as `.DS_Store`, which keeps Finder icon-placement information) are never indexed by Spotlight. Further, files of unknown data type are not content-indexed by Spotlight, though their file metadata is indexed. (Spotlight is plug-in based and knows about many file content formats by default, including Microsoft Word files, Apple Pages and Keynote documents, common image formats, and e-mail mailbox files from a number of client programs. Other file types can be supported for forensic analysis if the database is rebuilt, as will be discussed in Section 4.2.1.)

The metadata for one or many currently mounted volumes can be searched or queried using Spotlight. The search capability present in the Finder, for example, is provided by Spotlight. The access permissions of the application (or user) executing the search are honored; Spotlight does not include inaccessible items in the search results. The query capabilities are used in the Finder's *Get Info* command to display file-type-specific information about the file.

Spotlight metadata does not exist for files that have not been indexed. So, for example, searching for images using *kMDItemContentTypeTree* on a volume that is not indexed will return no results, whether or not there are images on the volume. However, file system metadata always exists for all files and folders. Spotlight will search this metadata even if a file, folder, or volume is not indexed. Since this information is not indexed, the search is considerably slower.

Spotlight also allows a user to perform searches against the text content of files (for example, searching for PDF documents containing the word "MEGA").

4.1. *Common metadata*

Each item indexed by Spotlight has a list of metadata attributes. A metadata attribute consists of a name and a value. These are provided by the individual metadata importers, but there are a number of agreed-upon attribute names (Apple Computer Inc., 2007b). Many of these contain forensically useful information. A list of names and brief descriptions of all metadata attributes on the current system can be obtained using `mdimport-A`.

The type of an indexed item is stored as a Uniform Type Identifier (UTI) in the *kMDItemContentType* attribute. A JPEG image, for example, has a content type of ''public.jpeg''. A description of the file type is stored in *kMDItemKind*; for a JPEG file, this is ''JPEG Image''. The *kMDItemContentTypeTree* attribute for an item contains the content type attribute itself, its parent content types, its grandparent content types, etc. This allows searching for broader types of files. Rather than searching for ''JPEG images and PNG images and GIF images and ...'', the investigator can search for files whose content type tree contains ''public.image''. There are many useful UTIs, including public.message, public.contact, public.vcard, public.archive, public.disk-image, public.image, public.movie, and public.audio. Note that the content type and content type tree are not searched against unless specifically requested.

Image files can contain the *kMDItemAcquisitionMake* and *kMDItemAcquisitionModel* attributes, which describe the manufacturer and model, respectively, of the device used to capture the image. For images captured with a digital camera, these are set to the make and model of the digital camera (e.g., ''NIKON CORPORATION'' and ''NIKON D70'', respectively, for a Nikon D70 camera). An investigator can thus search for all images on a volume taken with a known make or model of camera. He or she can also search for images produced by any digital camera at all (as opposed to images produced by some other means, such as drawing software). Other generic image properties, such as the height (*kMDItemPixelHeight*) and width (*kMDItemPixelWidth*) of the image are stored as well.

The *kMDItemWhereFroms* attribute describes where the file was obtained. Files downloaded using the Safari web browser – including cache files – store the URL of the file in this attribute. (Other web browsers may or may not do the same; Firefox, for example, does not. In addition, images ''saved'' by drag-and-drop into other applications will likely not have a *kMDItemWhereFroms* attribute.) The source URL information can be vital in linking a found contraband image (e.g., from a skin tone detection tool) with a potential distributor on the Internet.

Searches against the text content of files are performed using the *kMDItemTextContent* metadata attribute. Unlike other metadata attributes, it is not possible to query the ''text content'' metadata of a specified file; the text content may only be searched. Spotlight searches do not, by default, search against the text content of files – it must be specifically requested. Search results that are produced by matching against the text content attribute also have an additional relevance attribute indicating the ''strength'' of the content match. Search results that are produced by matching against other metadata attributes have no relevance attribute.

Although undocumented by Apple, Spotlight also maintains a *kMDItemUsedDates* attribute. This is similar to the last-access-time maintained by the file system, except that it is a list of *all* dates (not times) on which the file was accessed. In our experience, Mac OS X 10.4 does not update this attribute reliably, but 10.5 consistently does. Because of its undocumented status, the lack of an entry in *kMDItemUsedDates* should not be used as evidence of an unread file, but where date entries appear, they can be very valuable to examiners and give an idea of *how often* a given file was used.

4.2. *Accessing and using metadata*

Spotlight searches against any mounted volume can be executed with the *mdfind* command-line tool. This tool uses the documented Spotlight query format, which allows for powerful Boolean expressions (Apple Computer Inc., 2006). The metadata for a file can be queried with the *mdls* command-line tool. For un-indexed volumes or un-indexed files on indexed volumes, only file system metadata will be searched.

For example, one could search for all images taken with a Nikon D70 digital camera this year on the volume `/Volumes/Suspect` with

```
mdfind -onlyin /Volumes/Suspect
  ``(kMDItemContentTypeTree = = *) &&
  (kMDItemAcquisitionModel = = 'NIKON D70') &&
  (kMDItemContentCreationDate >=
  $time.this_year)''
```

Such a query could be critical in finding images taken with a camera at the crime scene, or for quickly tracking contraband images on many suspect machines.

On un-indexed volumes, or for un-indexed files on indexed volumes, only file system metadata will be searched. This means that `(kMDItemFSName == *.html && kMDItemFSContentChangeDate >= $time.this_year)` will work whether the volume is indexed or not. On indexed volumes, this search will execute much faster.

Searches against any metadata attribute, such as `(* ==``*penguin*''c)`, apply to the metadata attributes available for the particular file being tested. On an indexed file, this will match if any metadata attribute contains the text ''penguin'' (using case-insensitive matching). On an un-indexed file, this will only match if the filename (the only file system metadata attribute that is a string) contains ''penguin''. (Note that the ''w'' flag applied to strings allows for smart word-boundary matching and is generally preferred to surrounding a search string in asterisks. However, the *kMDItemFSName* attribute does not appear to treat this flag in the expected fashion, so asterisks must be used if file system name is a search criteria.)

4.2.1. *Preserving forensic integrity for non-indexed file systems*

If a disk image is un-indexed, there is reason to believe the index has been tampered with, or an investigator wants to rebuild the index to include additional files or metadata importers, the disk image can be mounted with a shadow file. A shadow file allows a disk image to be mounted read–write without modifying the original disk image (changes are written to the shadow file). The Spotlight index can then be deleted and rebuilt (or built, if it was missing) without changing the source image. Note that the new Spotlight index will use the metadata importers present on the investigator's system, which may be different from those present on the original system.

Normally a disk image should be mounted read-only using Apple's command-line tool: `hdiutil attach -readonly /path/to/image.dmg`. The image can instead be mounted with a shadow file: `hdiutil attach -shadow /path/to/shadow /path/to/image.dmg`. If the shadow file does not exist, a new, empty file will be created. The Spotlight database for this volume can then be rebuilt using *mdutil*.

4.3. Spotlight searches in MEGA

Making use of the operating system-generated index can dramatically speed case work: with searches executing in seconds, files of interest can be quickly found, or leads ruled out, in real-time. As discussed in the previous section, even un-indexed volumes (or volumes whose Spotlight index is un-trusted) can be indexed for forensic use: simply mount the volume read-only with a shadow file to contain the index.

MEGA includes a Spotlight search tool and GUI interface that allows the user to execute arbitrary searches, using the Spotlight query language, on individual volumes. The tool also supports simple string searches. This emulates the Finder's search behavior by searching for the specified string in any metadata attribute as well as searching against document text content for the string. Each search result includes a full list of its metadata attributes, including the special path and relevance attributes, as shown in Fig. 3.

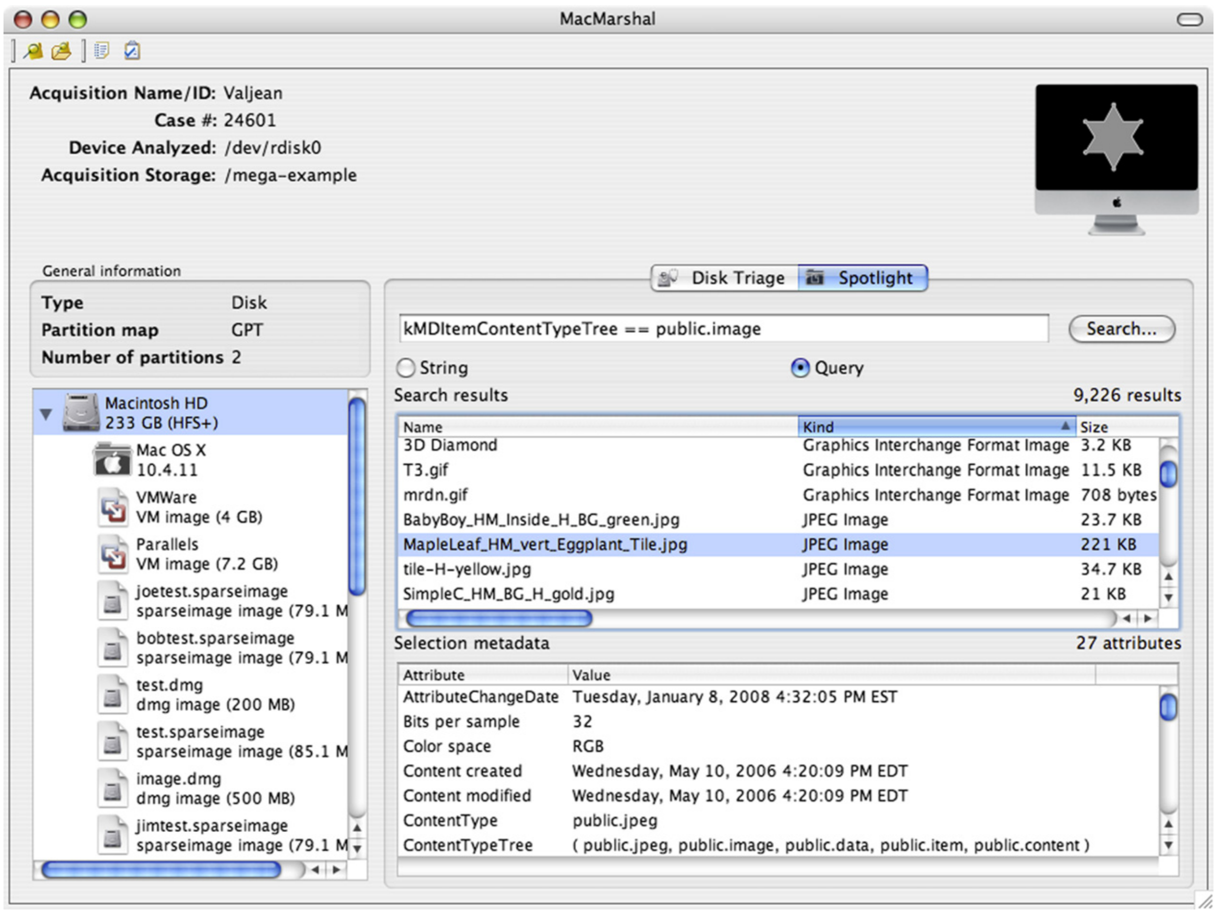
In order to answer the query, MEGA executes a custom command-line tool (`spquery`), which is similar to the built-in `mdfind` but offers more detailed output. As for all back-end tools in MEGA, the raw `spquery` output is also saved for auditing purposes.

For indexed files, executing Spotlight searches is very fast; even searches against files' text content takes only a few seconds. Listing the metadata attributes for every search result is more time-consuming – on the order of 1 s per 100 search results. Searching un-indexed file systems is slower, and cannot make use of derived attributes based on file content, but still appears to be faster than other available methods – 30 s for a disk containing approximately 70,000 files. Future versions of MEGA will give the investigator the option of automatically building a shadow index for un-indexed file systems.

5. FileVault encrypted user directories

FileVault, first appearing in Mac OS 10.3, enables individual users to encrypt their home directories, rendering the contents more difficult to access.

FileVault's capabilities are supplied by the Apple Disk Images framework. The Disk Images framework and its cryptographic backend, Apple's Common Data Security Architecture, provide the potential for future flexibility. FileVault has



Acquisition Name/ID: Valjean
Case #: 24601
Device Analyzed: /dev/rdisk0
Acquisition Storage: /mega-example

Type	Disk
Partition map	GPT
Number of partitions	2

Macintosh HD 233 GB (HFS+)
Mac OS X 10.4.11
VMWare VM image (4 GB)
Parallels VM image (7.2 GB)
joetest.sparseimage sparseimage image (79.1 M
bobtest.sparseimage sparseimage image (79.1 M
test.dmg dmg image (200 MB)
test.sparseimage sparseimage image (85.1 M
image.dmg dmg image (500 MB)
jimtest.sparseimage sparseimage image (79.1 M

kMDItemContentTypeTree == public.image

String [✓ Query]

Search results 9,226 results

Name	Kind	Size
3D Diamond	Graphics Interchange Format Image	3.2 KB
T3.gif	Graphics Interchange Format Image	11.5 KB
mrdn.gif	Graphics Interchange Format Image	708 bytes
BabyBoy_HM_Inside_H_BG_green.jpg	JPEG Image	23.7 KB
MapleLeaf_HM_vert_Eggplant_Tile.jpg	JPEG Image	221 KB
tile-H-yellow.jpg	JPEG Image	34.7 KB
SimpleC_HM_BG_H_gold.jpg	JPEG Image	21 KB

Selection metadata 27 attributes

Attribute	Value
AttributeChangeDate	Tuesday, January 8, 2008 4:32:05 PM EST
Bits per sample	32
Color space	RGB
Content created	Wednesday, May 10, 2006 4:20:09 PM EDT
Content modified	Wednesday, May 10, 2006 4:20:09 PM EDT
ContentType	public.jpeg
ContentTypeTree	(public.jpeg, public.image, public.data, public.item, public.content)

Fig. 3 – MEGA Spotlight search interface.

relatively little configuration flexibility, and has been reverse-engineered (Appelbaum and Weinmann, 2006).

In Mac OS 10.3 and 10.4, FileVault uses sparse images. Sparse images are a convenient mechanism for storing very large disk images without storing the contents of unused space, allowing the user to add data to his home directory without pre-allocating a large disk image, running out of space in his home directory, or requiring a time-consuming image rebuild. A sparse image consists of a header describing the layout of the disk and blocks (called stripes) of used disk space. Mac OS 10.5 uses sparsebundles, structured directory where the header and each stripe are stored in separate files.

When FileVault is enabled on a user account, the contents of the user's home directory are copied into a new disk image and deleted. A user account with FileVault enabled is readily identifiable, as its home directory contains only a single sparseimage file or sparsebundle directory.

The data in FileVault images is encrypted with 128-bit AES encryption. This data encryption key is stored in a header in the sparseimage file (or the "token" file in a sparsebundle). This header is encrypted with 3DES-EDE. The key to decrypt the header is generated from the user's login password (using 1000-round PBKDF2-HMAC-SHA1). Further, the data encryption key is stored a second time in the header, this time using the FileVault Master recovery keychain (using 1024-bit RSA public-key encryption). This is to allow the administrator to decrypt the user's home directory if the user forgets his password.

So, FileVault is not as secure as simple 128-bit AES. Any means of obtaining the user's login password or the FileVault Master recovery keychain will allow access to the FileVault image. Brute-force password cracking against either the disk image header or the Mac OS login credentials will recover the user's login password. The key-generation function is a weaker version of that used in Wi-Fi Protected Access (WPA); tools already exist to attack these keys, and one has been modified for FileVault. The security of Mac OS X login credentials has changed repeatedly to increase security, but as of Mac OS 10.4, none of the Mac OS X password-storage mechanisms were nearly as secure as that used by FileVault.

FileVault cannot protect any data outside of the user's home directory. In particular, this includes the virtual memory files and the "safe sleep" file – a copy of memory generated when a laptop goes to sleep. Virtual memory can be encrypted, but often this option is disabled. The safe sleep file is encrypted, but the decryption key is stored in the file. On a live system, passwords and decryption keys may be obtained from system memory. If the user is currently logged in, the FileVault disk image is mounted; not only is the decryption key in memory, but the user's home directory can be accessed freely until the user logs out.

MEGA currently finds FileVault-encrypted home directories, along with other disk images, during disk triage. Support for the sparsebundle format will be added. These images can be extracted for further analysis, decrypted if the investigator determines the password, and mounted and searched with Spotlight. MEGA also includes a modified version of *vfcrack* (Appelbaum and Weinmann, 2006), which enables dictionary-based brute-force password cracking of many encrypted disk images.

6. Other OS X-specific forensic data

MEGA will soon be expanded to provide a broad suite of Mac OS X-specific tools. First, we will add support for extracting data from system-wide services including the Address Book, spell checker, and NetInfo database. OS X's Address Book is commonly used to store contact information because of its tight integration with other Mac applications, especially Mail. The Address Book is also automatically synchronized to external devices such as iPods or supported mobile phones. (Savvy users may even associate portrait images with address book entries.) Extracting such system-wide contacts from a suspect machine could yield valuable leads in a case.

Similarly, a single spell checking service is provided at the operating system level and shared by most Mac applications. When the spell checker encounters proper names and other unknown words, it offers to "learn" them – i.e., add them to the per-user custom dictionary. This custom dictionary may well contain items – including proper names and locations – of significant evidentiary value in a case. (These words might also yield potential passwords for decrypting FileVault accounts.) MEGA will be able to extract and present these custom entries.

Mac OS ships with a number of applications that are the de facto standards for Internet access and other tasks. We will extend MEGA to gather evidence from the most common Mac-only applications, including Safari, Apple Mail, and iChat. Safari, Apple's web browser application, caches web pages and stores recent URLs in a history similar to most other browsers. Safari also stores the 10 most recent Google search terms entered in its "search" bar, items likely of high value in many cases. In addition, Safari has a web form-filling feature that goes beyond the typical "auto fill" (such as in Internet Explorer): it attempts to match known user data to new forms. For instance, when making reservations at a hotel site that has never been visited before, the user may start typing his first and last names; Safari will then "guess" at proper values for the Address, City, State, and other fields based on past form history. MEGA will be able to extract this form history, as well as Safari URL history, Google search history, and cached HTML and image files. (Through OS X 10.4, Safari used a sequence of cache files to store this information; more recent versions of Safari use an SQLite database file, readable with standard, public domain SQLite tools.)

Apple's Mail software, like many e-mail clients, keeps track of recently used e-mail addresses to simplify the composition of new messages. These recent addresses (both recipients and senders) will likely be valuable as evidence, and will be acquired by MEGA. MEGA will also be able to copy any mail stored in Apple Mail mailboxes.

Next, we will add support for extracting user information, buddy lists, and any saved instant messenger logs from Apple's iChat IM application. These logs and transcripts (including files transmitted) are typically stored in the user's Documents/iChats directory.

iPhoto, Apple's default photo management software, not only stores images but also keeps track of metadata (key words, ratings, comments, etc.) for each catalogued photograph. MEGA will be able to acquire photos by date, keyword, or other criteria, and will use third party previewers (or the investigator's machine itself) to view the images.

iTunes is the vector by which music is synchronized between an iPod and the target Mac. While the music files on a suspect machine may be of interest to an investigation (and can be acquired if needed), what is likely of more interest is tying a specific iPod to the Mac. All revisions of iPod hardware behave like standard USB drives, with FAT32 or HFS+ file systems, using hidden and renamed files to thwart casual copying of music from an iPod to the computer. Non-music data is also stored in the iPod file system. This data includes (Marsico and Rogers, 2005; Slay and Przbilla, 2007):

- A serial number, recorded in the iPod's SysInfo file. When the iPod is synchronized with a Mac or Windows machine via iTunes, this serial number is recorded by iTunes (e.g., in the Windows registry) and can be later used to tie the specific iPod to that specific machine.
- Address book entries, which are automatically copied to the iPod by iTunes (unless the user actively disables this feature – something casual users are unlikely to do).
- Calendar entries (appointments) which, like contact lists, are often inadvertently synchronized to the user's iPod.
- Photos or videos: newer iPods are capable of displaying images and playing videos; contraband images or videos, or ones containing evidentiary links, may have been placed on the iPod.
- Arbitrary files copied to the iPod by the user (e.g., for backup purposes).

The iPod cannot be ignored during an investigation because it can hold data of evidentiary value that may be critical to the examiner's case. More importantly, the iPod's synchronization data can prove valuable as well, linking the portable device with the computer to which it was last connected or providing further leads.

MEGA's extensibility is what allows it to handle this myriad of application- and OS-generated forensic data in a uniform way. It also affords an opportunity to handle emerging software and revisions of Mac OS X as they become available, thereby encoding the best practices of extracting useful forensic information from the files written by the OS and applications.

Existing forensic tools do not gather such data specific to Mac OS X, thereby ignoring a vital source of potential evidence. As Macs gain market share, the number of OS X machines seen by forensic specialists will only increase. By giving insight into the user's activities and exploiting OS X-specific features, rather than just file system-level details, MEGA fills a vital role for law enforcement and corporate compliance when Macs are involved.

Acknowledgements

This project was supported by Award No. 2007-DN-BX-K020 awarded by the National Institute of Justice, Office of Justice Programs, U.S. Department of Justice. The opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the Department of Justice.

REFERENCES

Appelbaum Jacob, Weinmann Ralf-Philipp. Unlocking FileVault: an analysis of Apple's disk encryption system. In: 23rd chaos communication congress, Berlin; December 2006. http://events.ccc.de/congress/2006/Fahrplan/attachments/1244-23C3VileFault.pdf, with code at http://events.ccc.de/congress/2006/Fahrplan/attachments/1245-vilefault-23c3.tar.gz.

AppleInsider. Apple snags 14 percent of US-based PC retail sales in February, http://www.appleinsider.com/articles/08/03/17/apple_snags_14_percent_of_us_based_pc_retail_sales_in_february.html; March 17, 2008.

Apple Computer Inc.. Secrets of the GPT. Technical Note TN2166, http://developer.apple.com/technotes/tn2006/tn2166.html; November 2006a.

Apple Computer Inc., Spotlight query programming guide. March 2006b. http://developer.apple.com/documentation/Carbon/Conceptual/SpotlightQuery/SpotlightQuery.html.

Apple Computer Inc.. Introduction to Spotlight, http://developer.apple.com/documentation/Carbon/Conceptual/MetadataIntro/MetadataIntro.html; May 2007a.

Apple Computer Inc. Spotlight metadata attributes reference, http://developer.apple.com/documentation/Carbon/Reference/MetadataAttributesRef/MetadataAttrRef.html; May 2007b.

BlackBag Technologies Inc. Macintosh forensic software. http://www.blackbagtech.com/software_mfs.html.

Carrier Brian. The Sleuth Kit. http://www.sleuthkit.org/.

Cyber Security Technologies's OnLine Digital Forensic Suite. http://www.onlinedfs.com/.

Daily Princetonian, http://www.dailyprincetonian.com/archives/2006/10/12/arts/16162.shtml; October 12, 2006.

Kubasiak Ryan. MacOS X forensics guide, http://www.macosxforensics.com/page1/files/MacForensics.pdf; May 29, 2007.

Kubasiak Ryan. Mac OS Forensics web site, http://www.macosxforensics.com/; 2008.

Marsico Christopher V, Rogers Marcus K. iPod forensics. CERIAS tech report 2005-13, Purdue University; 2005. https://www.cerias.purdue.edu/tools_and_resources/bibtex_archive/archive/2005-13.pdf.

Singh Amit. Mac OS X internals: a systems approach. Addison Wesley; 2006.

Slay J, Przibilla A. iPod forensics: forensically sound examination of an Apple iPod. In: Proceedings 40th annual Hawaii international conference on system sciences; January 2007.

Subrosasoft's MacForensicsLab. http://www.macforensicslab.com/.

Dr. Robert A. Joyce is the Technical Director for Information Management at ATC-NY. His research interests include distributed information storage and transformation, computer forensics, image and video processing, network and media security, visualization and design, and human-computer interaction. Since joining ATC-NY in 2002, he has led several research and development efforts in the area of information management and has made significant contributions to many other projects within the organization. He was a substantial contributor to the development of the OnLine Digital Forensic Suite, a live forensics tool, as well as P2P Marshal, a peer-to-peer usage analysis tool.

Judson Powers is a Computer Scientist at ATC-NY. Since joining in 2007, he has contributed to P2P Marshal, a tool to analyze peer-to-peer file sharing usage, and to a peer-to-peer file sharing forensics training course. He holds an M.S. in physics from Cornell University.

Dr. Frank Adelstein is the Technical Director of Computer Security at ATC-NY, and provides oversight and guidance to projects at ATC-NY relating to computer security. His areas of expertise include digital forensics, intrusion detection, networking, and wireless systems. He has co-authored a book on mobile and pervasive computing. He received his GIAC Certified Forensic Analyst certification in 2004. A recent research focus is in the area of live forensics. He was the Principal Investigator on projects that resulted in the OnLine Digital Forensic Suite(TM), a live forensics tool, and P2P Marshal, a file sharing analysis tool. He is the vice-chair of the Digital Forensics Research Workshop.