Forensic Analysis of Spoliation and Other Discovery Violations Part I of a 2 Part Series: Macintosh Examinations

By Steve Bunting

When the police investigate a crime and they execute a search warrant for digital evidence, the charged party usually isn't aware that the police are coming with warrant in hand. In essence, the search of the digital media is often achieved by surprise and the suspect has little or no time to dispose of evidence. Even if the defendant had some prior warning and subsequently deleted or secreted digital evidence, from a practical sense, there's no crime or penalty for doing so. Furthermore, the criminal defendant enjoys the right not to self-incriminate.

In matters involving litigation, the rules are much different. Once a party to potential litigation becomes aware of the reasonable possibility or likelihood of litigation, a duty attaches to all parties to preserve all potential evidence, including digital evidence. Even counsel for both parties has an obligation to instruct their clients and to ensure that preservation of evidence. In the Zubulake case[1], the court stated that counsel has an affirmative duty to monitor their client's compliance with evidence preservation obligations. Thus obligated, the parties carefully preserve all evidence and provide that evidence to the other party when the discovery process begins, or at least, that's how it is supposed to work in an ideal world when everyone follows the rules.

Understandably, it must be a tough pill to swallow for a potential litigant to look at their digital media, know it contains information that harms their position, and then preserve it so it can be handed over to the opposing party to be used against them. And so the thought crosses their mind that it would be much better to either exclude that media from discovery or to destroy or alter the incriminating portions of the media so it can be safely turned over during discovery. Spoliation of evidence is the intentional, reckless, or negligent withholding, hiding, altering, fabricating, or destroying of evidence relevant to a legal proceeding. Thus withholding, deleting, or hiding evidence are forms of spoliation. More specifically, referencing Black's Law Dictionary in its ruling, an Arkansas court defined spoliation as "the intentional destruction of evidence and when established, [the] fact finder may draw [an] inference that [the] evidence destroyed was unfavorable to [the] party responsible for its spoliation."[2] Thus spoliation carries with it a very specific penalty in that the aggrieved party may legally infer the destroyed evidence was unfavorable, which often has a devastating impact on the party who destroyed the evidence.

Once the litigant heads down this path, the slope becomes treacherous and slippery. In addition to spoliation of evidence, often they are signing sworn declarations attesting to accuracy and completeness of the discovery materials, which in turn can form the basis for perjury and the case can quickly evolve into a criminal matter. Of course, spoliation requires proof, but once that proof is forensically established, things start to snowball and the penalties are usually worse than the outcome would have been otherwise, if the rules had been followed, which is, of course, the intent of the law and the rules in the first place. So let's turn our attention now to various forms of proof that can be used to establish spoliation. Part 1, this part, will discuss the artifacts found on the OS X or Macintosh operating system. Certainly, we can't cover all facets of such an examination, but will discuss some of the more common artifacts of interest in a spoliation case.

Before we get technical, let's look at the matter in a physical world. Let's assume that we are walking through snow covered terrain and we wish to hide our tracks. The simple way would be to cut an evergreen bough and use it to whisk away and obliterate those tracks. From a practical aspect, the whisking away can leave a pattern that is observable. The tracks may be gone, but the trace artifacts left by the branch are present. When we reach the end of our trail and are complete, we have eliminated the tracks, but are still left holding the branch in our hands. We can toss it or hide it, perhaps, but it can still likely be found in part or in whole. And let's not forget that somewhere there is evidence of where we cut this evergreen bough. This creates an interesting physical scenario and helps us view the process in the digital world.

In the digital world, the process is somewhat similar. Let's say the party uses an OS X built-in tool to wipe those tracks, which is "Erase Free Space", found in Disk Utility. This tool will zero out or wipe data in the free spaces of the selected media. Such an act, done after the duty to preserve attaches, is clearly an act intended to destroy potential evidence and certainly fits the definition of spoliation. Figure 1, below, shows an area in the unallocated clusters that contains data, or footprints in the snow, before any wiping has occurred. The data, or tracks in the snow, are plain to see.

After the data has been wiped, as shown below in Figure 2, the data is gone, as one would expect. It is important to note, though, that in its place is a pattern of zeros, all zeros in fact. This pattern is analogous to the pattern left by the evergreen bough in our physical example.

Normally, one expects to see considerable data in the unallocated spaces in a normally functioning computer system. Files are continually being deleted, both by the system and by the user. When files are deleted, only the pointers to that data are changed. The data is not deleted when a file is deleted. The space occupying the data is marked as

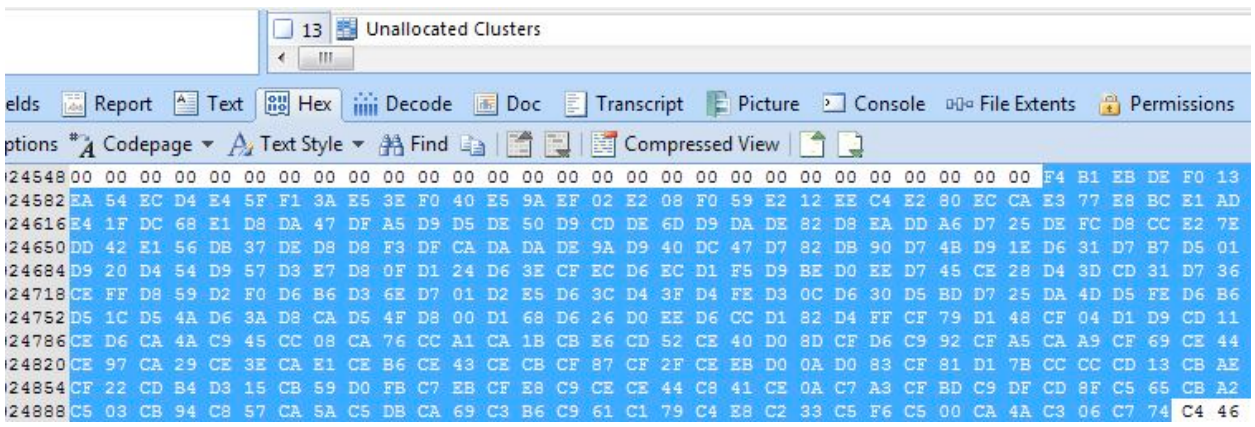

Figure 1 - Data found, starting at offset 24,559, in the unallocated clusters, before any wiping.

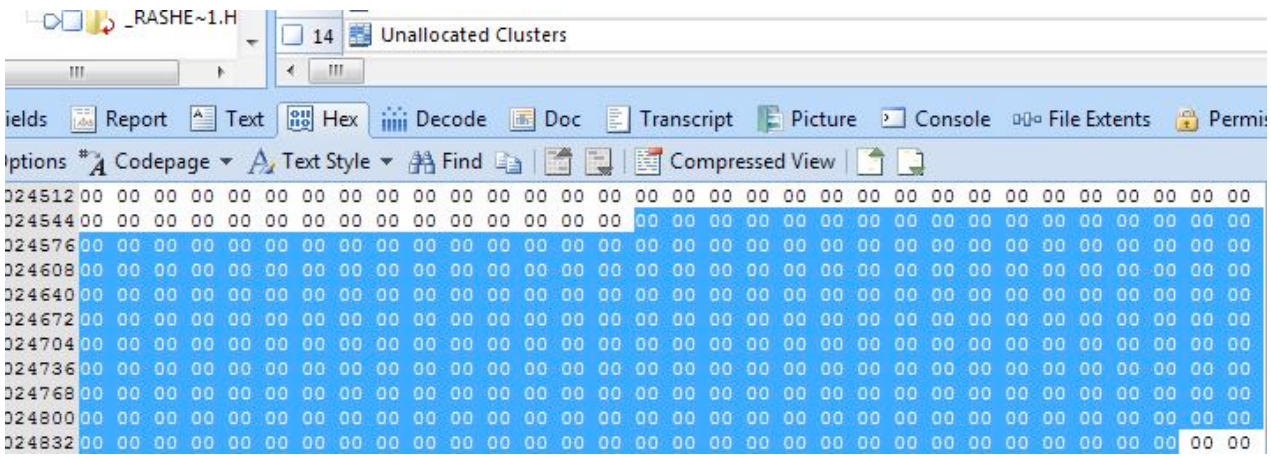

Figure 2 - Same exact location in the unallocated clusters after secure erase has occurred.

available and can be overwritten, but normally, one finds considerable data in these spaces. To find mostly zeros in the unallocated spaces of an active computer system is suspicious.

Now that we've seen the pattern left by the tool, let's see if there's evidence of the tool itself and perhaps some incriminating metadata. After all, a branch was used to wipe the tracks in the snow, let's see if there's evidence of that branch.

On a Macintosh operating system, OS X, there's no need to purchase a tool, as an excellent one is available for free. It is located, as previously mentioned, in the Disk Utility. If there's one thing for certain with OS X, it is a logging beast, which is to say that since OS X is based on BSD Unix, robust logging is hereditary. Believe it or not, Disk Utility has its own log and since many spoliation activities (formatting, erasing, securely erasing, encrypting disks, etc.) are carried out within the Disk Utility toolset, this log is a goldmine when examining spoliation issues, as nearly all Disk Utility actions are recorded here.

There are many ways to examine this log, including most any text editor, as it is a pure text-based log. My preference is to use Console, which is the OS X native utility for viewing and searching logs. There is a Disk Utility log for each user, which establishes individual accountability on multi-user systems, assuming each user has and uses their own account. The log is located at: `~/Library/Logs/DiskUtility.Log`.

With this log open, it is often wise to peruse through it and observe the activity. This log often covers long periods of time as it exclusive to Disk Utility. For example, my machine is used almost daily when I'm in my lab and Disk Utility is used frequently. As of this writing, there is activity in this log covering a 2 ½ year span.

In particular, you want to observe carefully the timeline the moment the duty to preserve attaches, or thereabouts. This is often referred to as the OS moment (Oh Shoot moment), when the party realizes they are in trouble and the urge to circle the wagons occurs. If suddenly, at that juncture, you see a lot of drives being formatted, securely wiped, and so forth, you have hit spoliation pay dirt.

To see if there was any erasing or erasing of free space (two different functions), filtering in Console for the string "erase" is a good starting point. Figure 3, below, shows different types of erasing activity on the system under review.

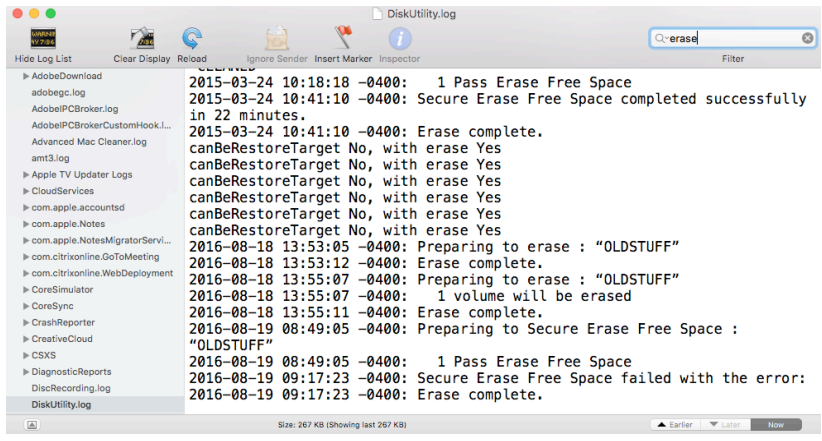

Figure 3 – Using Console, `DiskUtility.log` file filtered for 'erase".

To distinguish between the two different erasing activities, if one erases a partition, they are replacing it with another partition. In that case, the security options default to "fast" or no wiping with zeros. In this case, data still can be recovered from that media. If one erases free space (Secure Erase Free Space), then all data in the free spaces are replaced with zeros and that data is, for all practical purposes, gone. Regardless, either can be evidence of spoliation and are captured in this log.

Another good review item for this log is to filter on the string "format" or "formatting". See Figure 4, below. You will then see all volumes that have been formatted with Disk Utility. You may see drives formatted shortly after preservation has attached, which can establish spoliation. You may also see drives formatted that have been withheld from discovery and by their very names appear relevant. For example, spoliation is all too common in digital rights litigation, where a defendant is accused of illegally downloading copyrighted materials.

If a drive, in such a case, were discovered in this log to have been formatted and named "`MyMovieLibrary`", such would be most significant where a device by that name had not been disclosed in discovery. Figure 4 shows a log filtered for "formatting". In it, one entry of particular import has been circled in red, which indicates a volume has been formatted and named "`TimeMachineMacPro4TB`". Such a discovery is

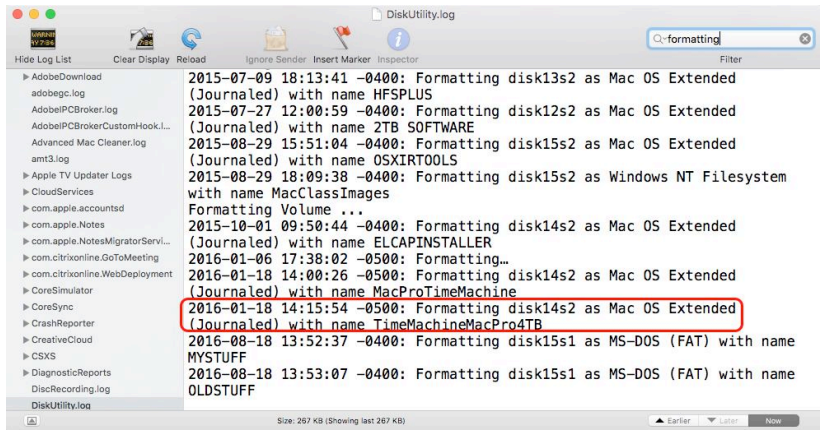

Figure 4 - DiskUtility.log file viewed in Console and filtered for "formatting".

incredibly important in every case, but even more so when spoliation is the issue. If a party is trying to hide evidence, it is not likely they have disclosed their TimeMachine or backup drive, which provides a backup of nearly all of their data. How far it goes back will vary but, from this log, you have just discovered its existence and the spoliation case has become much stronger. Naturally, you'll need to seek production of this drive from the party.

Whether you've discovered a volume named Time Machine in the Disk Utility log or not, you'll still want to see if Time Machine is running, when it last backed up, and the name of the volume to which it is writing. This information is contained in the file /var/log/system.log. Again, you can view it in the application Console. To view Time Machine information, filter for the string "backupd" (backup daemon). If Time Machine is running, you'll find entries regardless of whether the backup disk is mounted or not. If mounted, you see references to it as shown in Figure 5 below (see volume name highlighted in blue). If not, it will mention the volume by name that couldn't be found. Either way, you will

know of its existence. If it exists and has not been disclosed, your spoliation case is again supported by this finding.

If you have a Time Machine drive, produced initially or produced later as a result of your discovering its existence, you'll want to compare what is currently on the party's machine with content of the Time Machine drive between the "OS moment" and when the Macintosh computer system was produced. It is during this period when the data starts to "disappear".

It may be that the party claims they don't have the Time Machine drive for whatever reason. That's an issue for the court, but that doesn't necessarily stop your examination of Time Machine. Time Machine has an obscure feature about which little is mentioned in forensic circles. I was doing some testing with Scott Pearson in April 2013 in Manila when we encountered a hidden file in the root named ".MobileBackups" coupled with a mounted volume by the same name. All of this is hidden from the regular user. Upon exploring this a little more, we discovered that whenever Time Machine can't backup to a drive that is not present, it maintains the Time Machine function by writing temporary Time Machine data to this hidden file. It is stored in the same format as a regular Time Machine drive, as shown in Figures 6 and 7.

Time Machine makes extensive use of link files. Where a file hasn't changed, there's a link file

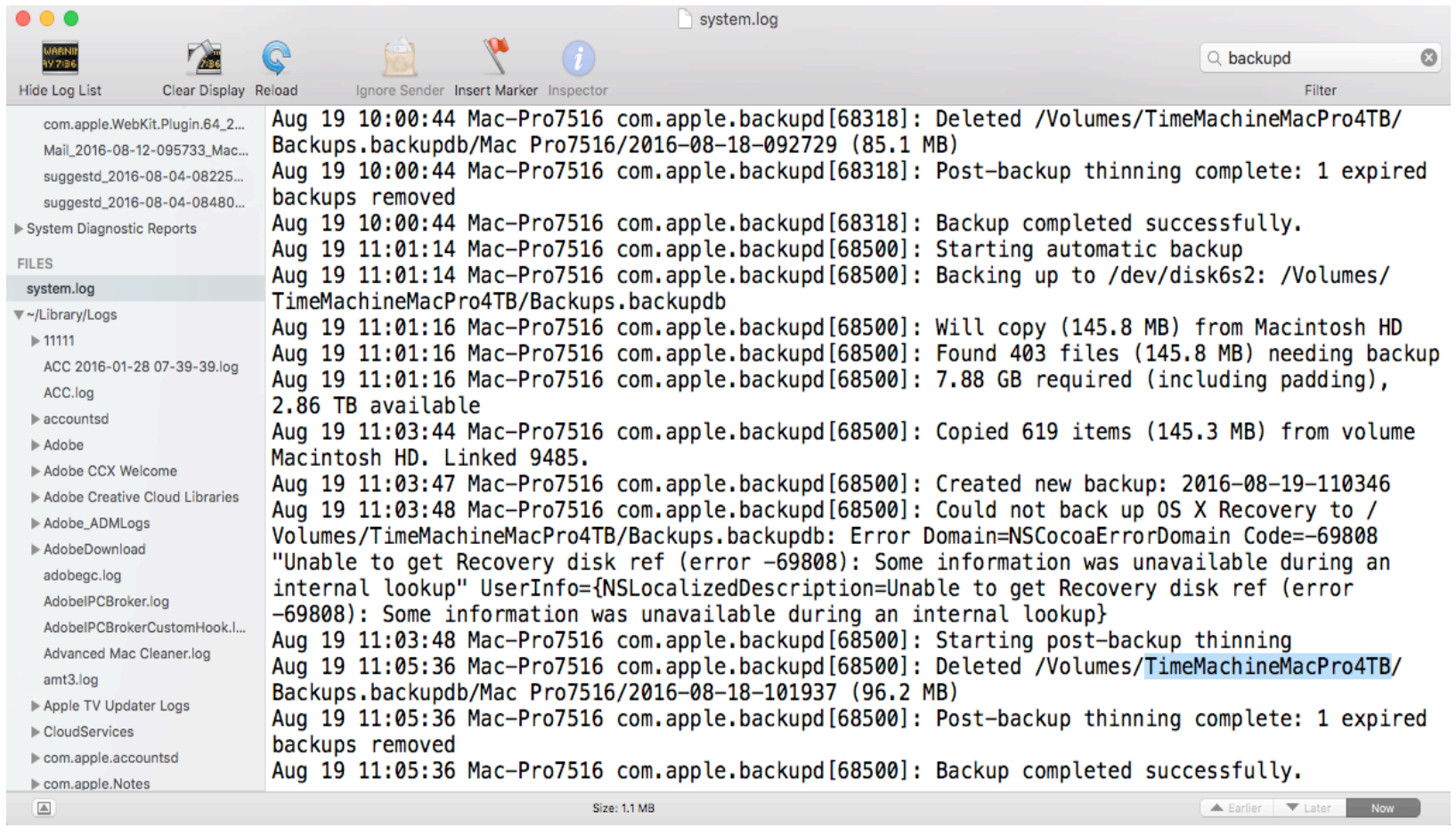

Figure 5 - `/var/log/system.log` filtered for string 'backupd' to reveal Time Machine activity

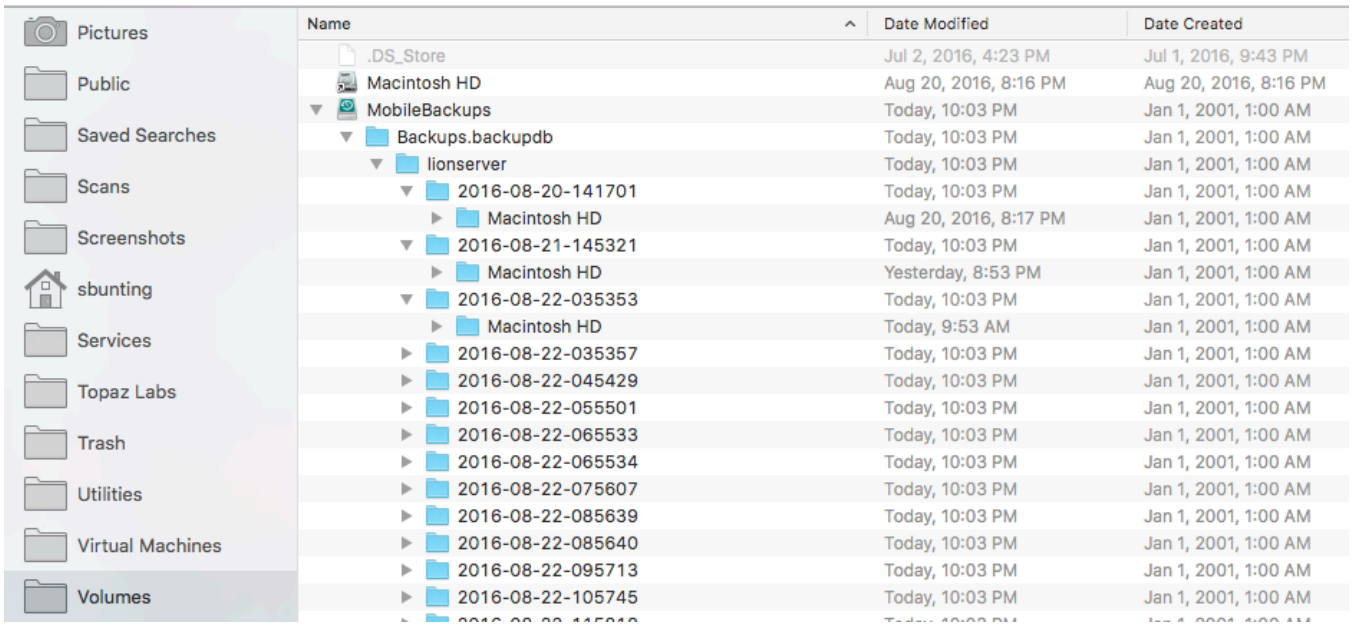

Figure 6 - Hidden Time Machine file in the root that is mounted and appears in /Volumes. It uses the same format as a regular Time Machine drive.

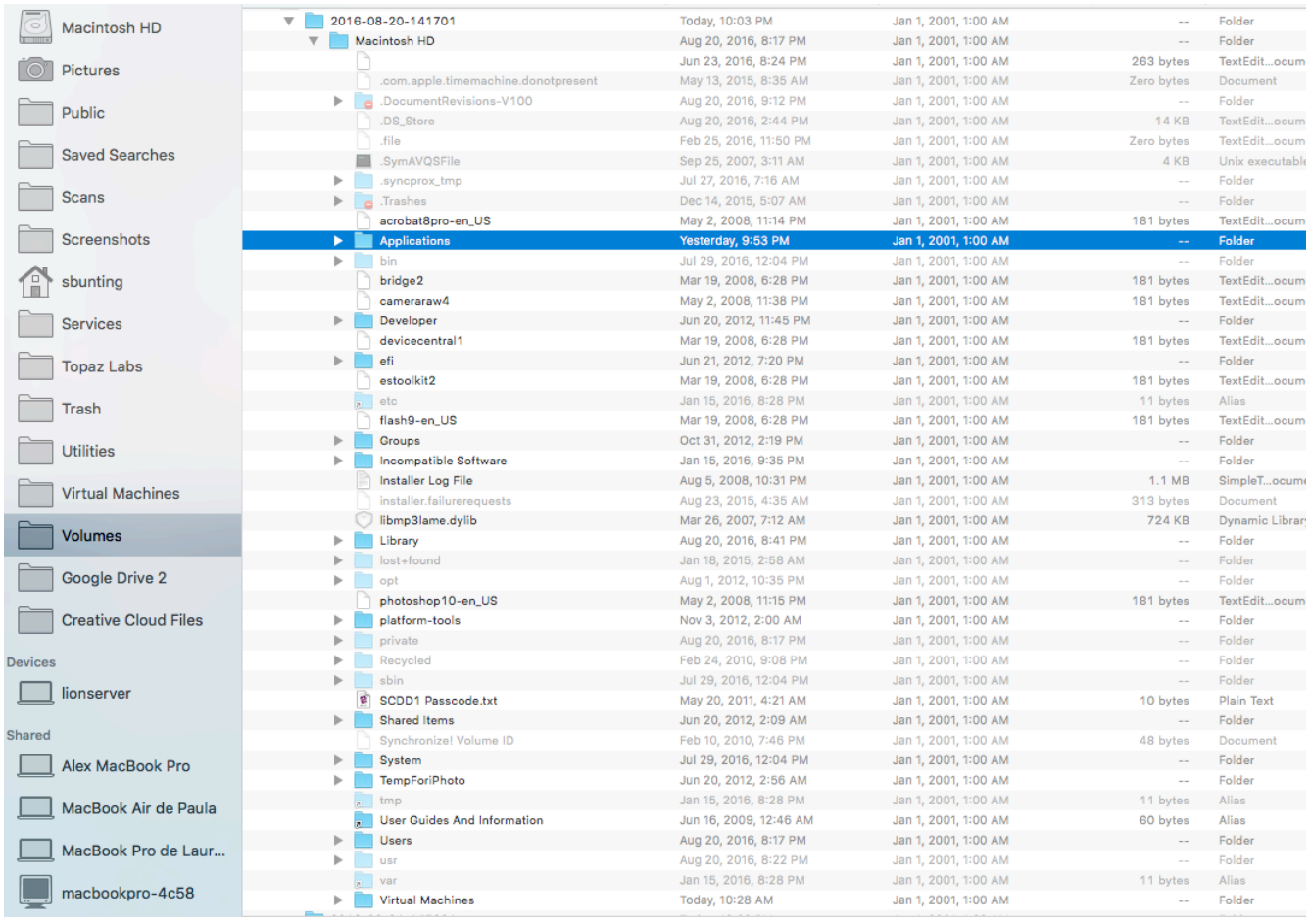

Figure 7 - Inside each folder bearing a timestamp name, there's a complete directory structure of the entire drive.

pointing back in time to where it actually exists when it was last changed. It takes some getting used to. EnCase has an excellent parser for stepping through these files. The point here is not to make you an expert on examining Time Machine files, but instead to point out their criticality in spoliation cases and to point out the existence of this hidden Time Machine, which can be a virtual gold mine in any spoliation case. Very few forensic examiners know about the hidden copy of Time Machine and even fewer users,

so likely it will not be touched by a user trying to hide his or her tracks. As with any Time Machine examination, you should compare what you find in this temporary Time Machine with the periods before and after the duty to preserve attaches, especially as production for discovery commences.

Another under-exploited resource for spoliation examinations is the Macintosh Quick Look Thumbnail Cache. This database and cache image storage supports the Quick Look function in Finder and provides the cached thumbnail images that you see when you open a folder in Finder. Thus the act of opening a folder creates a thumbnail of that file's content. With Windows, only images have cached thumbnails. Mac supports other formats and thus you can expect to see cached thumbnails of documents and images.

The path to this file is deep and obscure and buried below `/private/var/folders`.

The bottom level folder is named "`com.apple.QuickLook.thumbnailcache`".

Between these two, you will find randomly named GUID folders, so it is easiest to filter your forensic tool to locate the string :

"`com.apple.QuickLook.thumbnailcache`".

Therein you will find a set of SQLite database files (index.sqlite index.sqlite-wal & index.sqllite.shm) along with the images themselves in the thumbnails.data file, as shown below in Figure 8.

When you review the index.sqlite database, make certain to copy out the write-ahead-log (wal) and shared memory (shm) files. Data is first written to the wal file and later committed. Were you to read the database file alone, you would miss data contained in the 'wal' file! The database contains a series of related tables. One table, shown in Figure 9, is 'files' and contains the path and file names for which cached thumbnails have been created.

The 'files' table is linked or related to the 'thumbnails' table, which is shown in Figure 10. For each entry in the 'files' table, this table tracks the 'last_hit_date' and hit counts. Further, this table

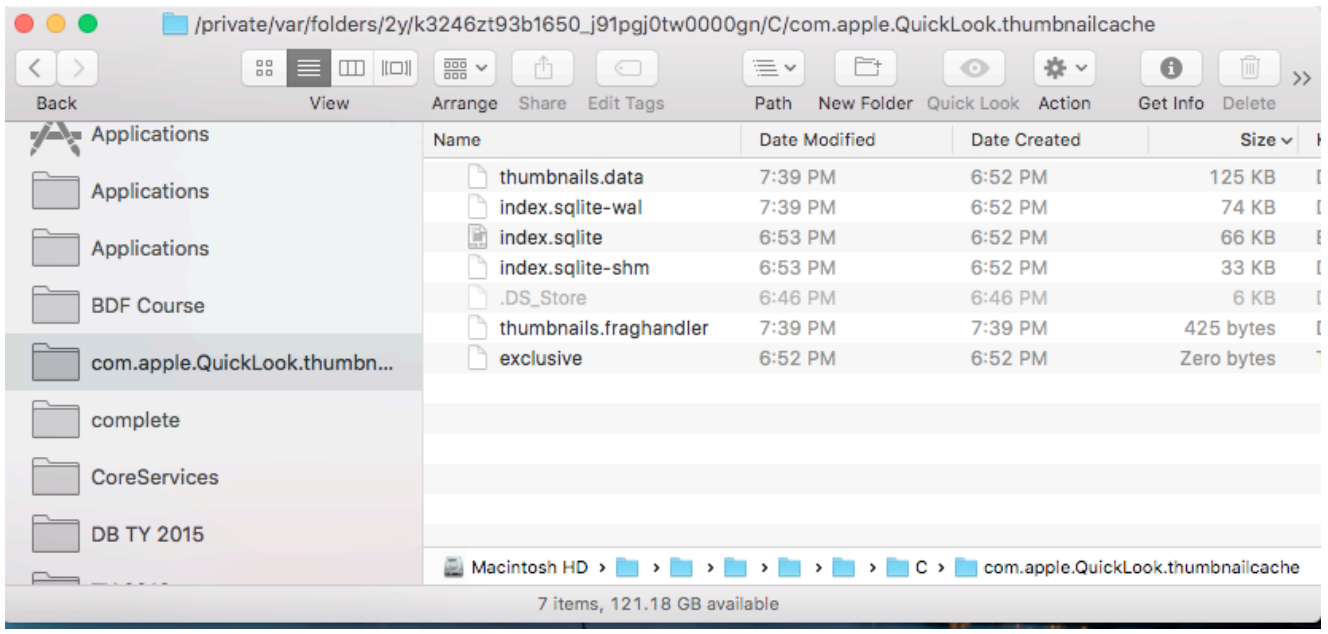

Figure 8 - Contents of the Quick Look Thumbnail Cache folder

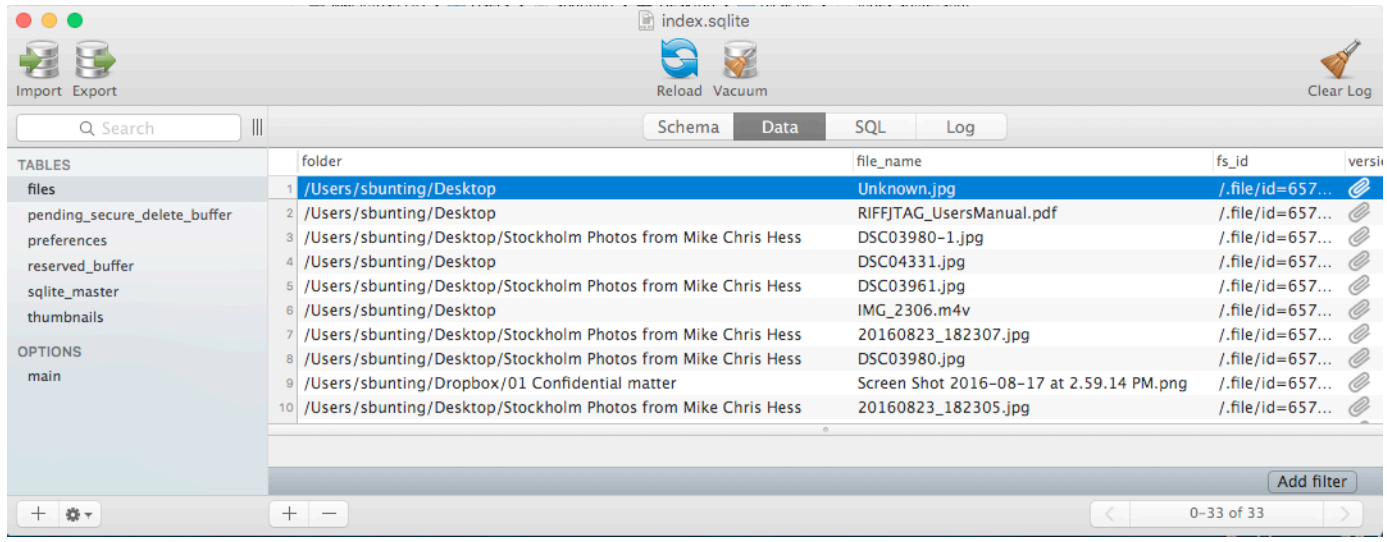

*Figure 9 - The 'files' table is displayed in **Base**, revealing file names and paths for which thumbnail caches have been created.*

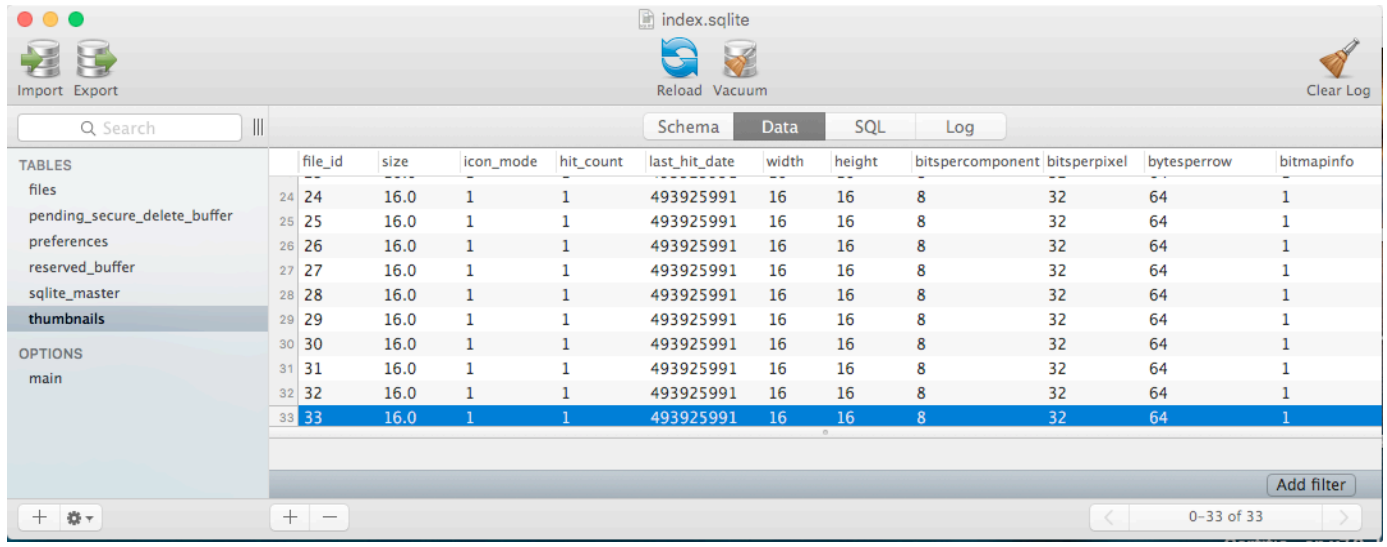

*Figure 10 - The 'thumbnails' table is displayed, in **Base** in which the last hit date, hit count, and pointers to the thumbnail cache are contained.*

contains the pointers to the actual thumb image in the thumbnails.data file.

While you can manually step through these entries, it is no fun and very time consuming. Simon Key, from Guidance Software, has written an excellent EnScript for EnCase 7 / 8 that parses this information and pulls the images out as well, as shown in Figure 11. Simon has also posted an explanation of how this feature works and of its forensic import. This blog can be found at: http://encase-forensic-blog.guidancesoftware.com/2014/05/examination-of-mac-os-x-quick-look.html

The import of this data to a spoliation examination is quite simple. You can see files that likely once existed and are no longer present. In the above image, Figure 11, if this were a digital rights case and the defendant claimed to have never downloaded the above song, imagine the impact of

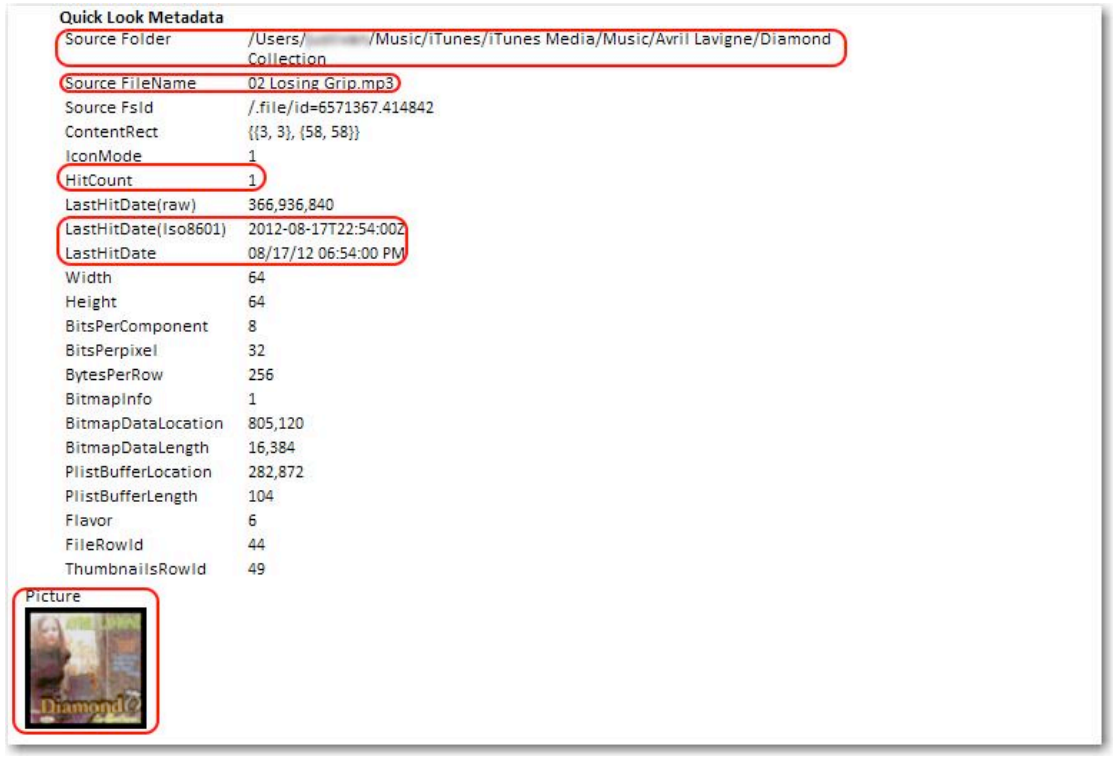

Figure 11 - EnCase 7 Quick Look Thumbnail Cache Parser results. All this Quick Look Metadata, including the image itself, is parsed and presented in this view. Note the file name, path, and last hit timestamps.

this thumbnail cache entry on that claim. The last_hit_date will often reflect when the user was last reviewing the files, which is often after preservation attaches and before production of discovery. File names are often indicative of their contents and can establish relevance, which is supported by a thumbnail cache created by its contents. When such relevant appearing files are no longer present and the timestamp points to that critical "OS moment", you have found evidence of potential spoliation. You also have a file name to search for its presence in other critical areas, including the Time Machine.

Finding deleted files on a Macintosh, aside from those found in Time Machine, is a challenging endeavor. When files are deleted on a Mac, unlike Windows NTFS or FAT file systems, the Catalog's B-Tree structure very efficiency removes file metadata shortly after a deletion. Such leaves carving as the primary means of recovering deleted files on a Macintosh system. There are EnScripts that parse the Journal file, from which some file recoveries can be made. When files are carved, sometimes they contain internal metadata that can establish identity and timestamps in addition to their contents, any of which could be used as evidence of spoliation. Such data can also be used in conjunction with other findings, such as Quick Look Thumbnail Cache, for example.

Going back to covering our tracks in the snow, we should look also for evidence of the evergreen branch, which is to say in a digital world that we are looking for programs that are used to clean, remove, or destroy data and artifacts. As tools or programs come and go, it is often a good practice to search

Google for something like "OS X Evidence Cleaner". In this manner, you'll see the most popular tools for removing evidence. You can fashion search strings to search the case for the presence of such tools, either installed or when the user was searching for such tools.

An excellent tool for processing Macintosh systems is Sumuri's Recon. Recon allows you to select processing modules or plugins for various types of examinations. Once selected, the modules are run and results returned in the "Result Viewer", as shown below in Figure 12 below. Figure 12 shows the Result Viewer with the list of all installed applications displayed. In this case, we have selected the evidence cleaning tool known as "CCleaner". It shows when it was installed. If such coincided with the critical "OS Moment", you are showing the user installed a tool to remove evidence at a time when evidence was supposed to be preserved.

While this view is important, the Advanced Analysis section contains more information concerning this tool, as shown in Figure 13. You can see the settings for this tool, which is to say, which artifacts or evidence it is configured to clean or remove. Since this tool was used, it is a good idea to install and test the tool that was used so that you can observe first

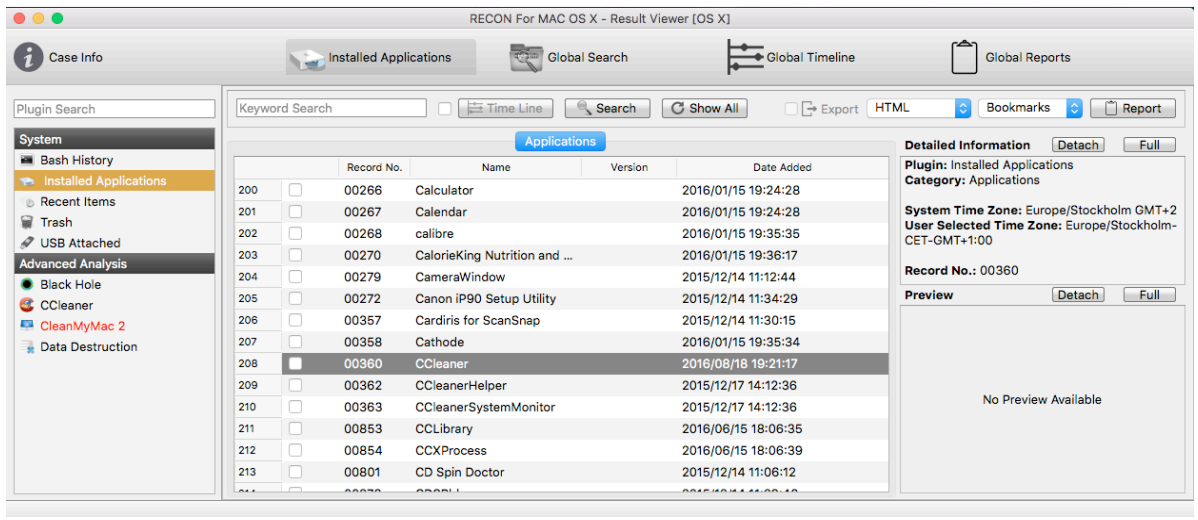

Figure 12 - Result Viewer showing installed applications, specifically 'CCleaner'

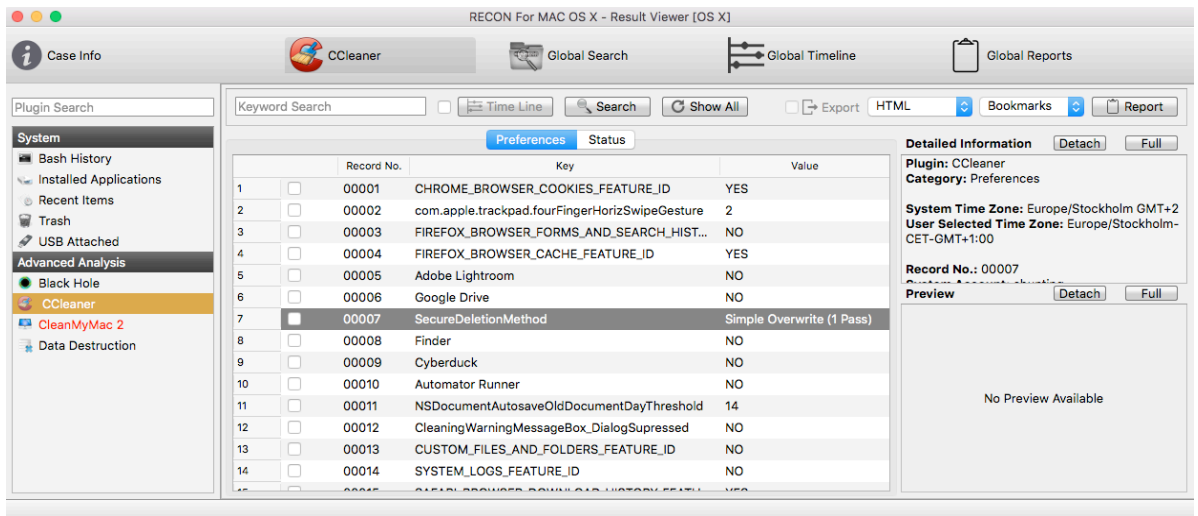

Figure 13 - CCLeaner details are displayed in the Advanced Analysis section. There are 223 settings or properties that are parsed. Only 14 are seen here.

hand its behavior, default settings, and any patterns or temporary files it may leave behind.

Figure 14, shown below, shows the USB devices that have been attached. These are tracked in the system log. You can find them manually by searching for the string USBMSC if you wish. Recon does a nice job of parsing and presenting them. You can sort by time in Recon and if you see USB devices attached during the period around the "OS moment", you should consider that data may have been moved elsewhere. You should seek to make those devices part of discovery if they are not already included, as you will have the make, model, and serial number of the USB device by which to identify it.

The Trash Can, shown in Figure 15 below, is where deleted files are initially sent when deleted. They are not really deleted until the Trash Can is emptied. While one would think that someone trying to destroy evidence would do a good job, they just might forget to empty it. It has happened in the past, because some users are just not that computer savvy, thus one should always check the obvious places by checking the Trash Can. You might get lucky. If you do, it will also show the timestamp for when it was deleted or 'added to the Trash Can'.

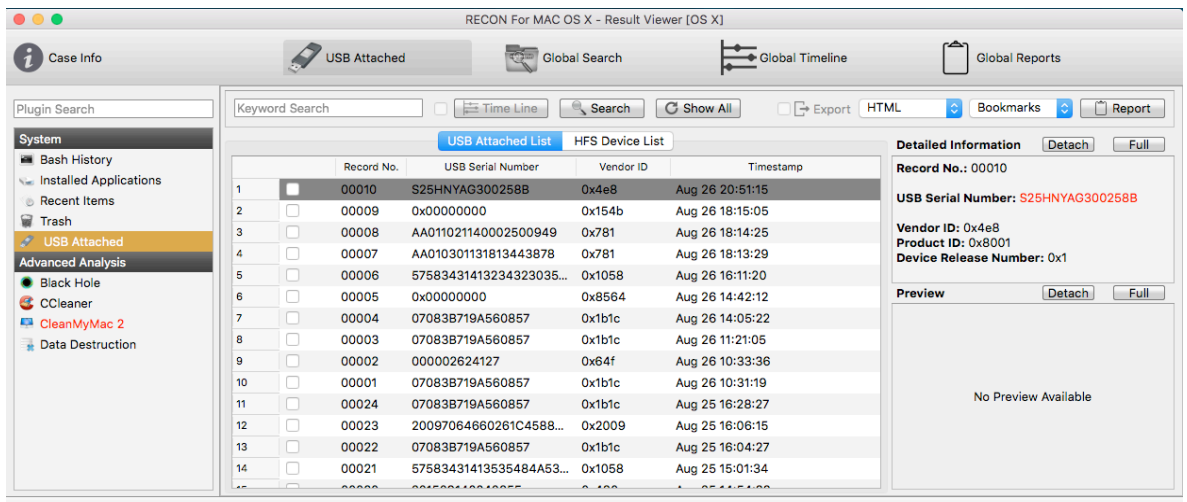

Figure 14 - This view lists USB devices that have been attached. There is a timestamp along with the make, model, and serial number of the device.

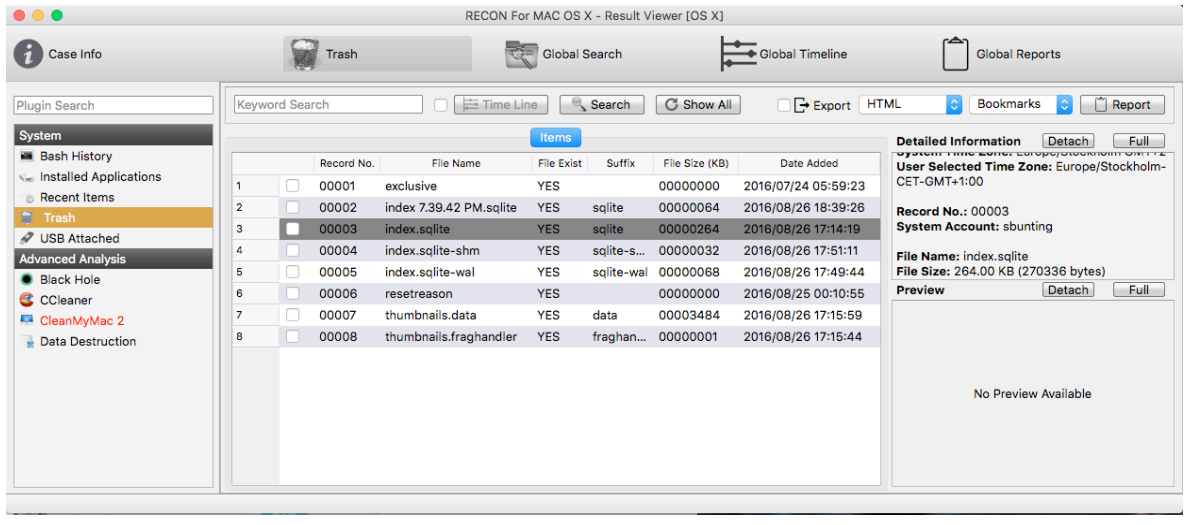

Figure 15 - The Trash Can could contain folders or files that the user deleted and simply forget to empty the Trash Can. It does happen!

The OS X operating system, just like its cousin Windows, stores a vast collection of recent items. Nearly all are in "plist" or property list files, which serve much the same function in OS X as does the registry in Windows. The number of plist files that are parsed and represented in this view is quite impressive. Some of system plist files are retained by application-specific plist files. Regardless, they are all aggregated in the Recent Items view in Recon. Figure 16, below, shows Recent Hosts and Recent Servers to which the computer in question was connected over the network. The import in spoliation cases is that here you will find remote or networked computers that should be and have not been made part of discovery.

Under the Advanced Analysis section, you will find a category for Data Destruction. While we've already discussed it earlier, this module searches for strings in the Disk Utility log relating to data destruction, as shown in Figure 17 below.

When using automated tools, one can save time and methodically carry out a large number of specialized tasks. These tools can point to areas in need of more in-depth analysis. One must remember, however, that they are no substitute for a knowledgeable examiner. They assist the examiner only. The examiner must know the tool and its limitations. The

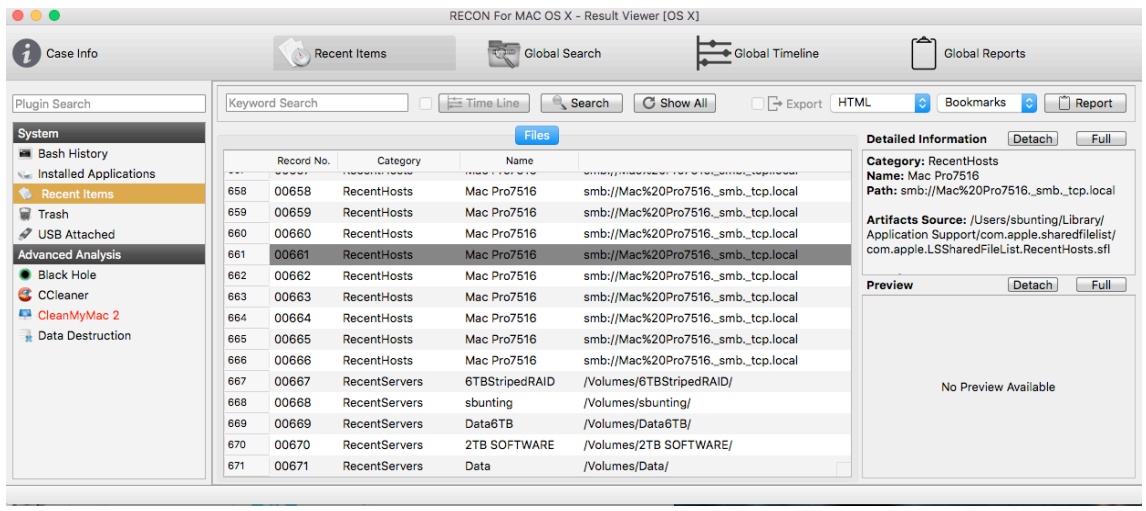

Figure 16 - Shown is the Recent Items view in which RecentHosts and RecentServers are listed.

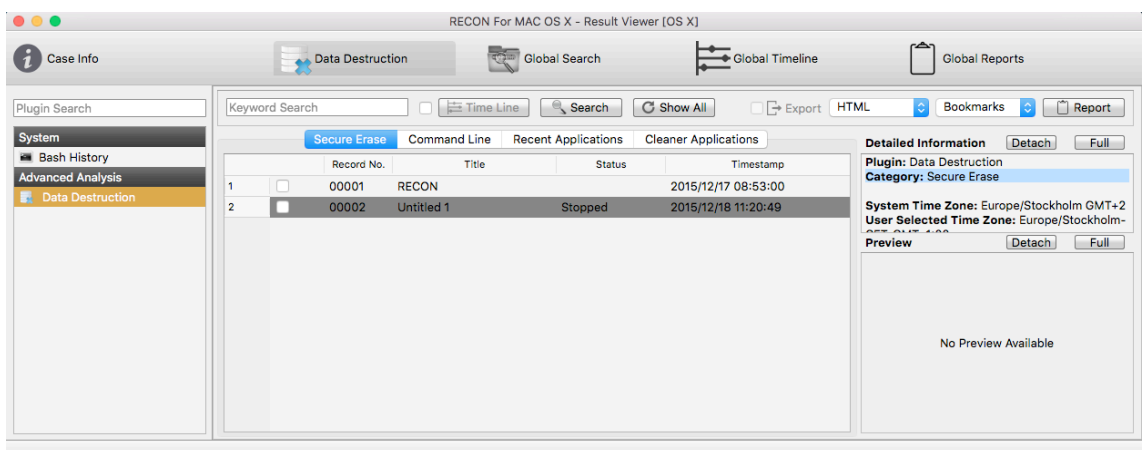

Figure 17 - Data Destruction looks for strings in the Disk Utility log pertaining to secure wiping, etc.

examiner must fill in the gaps; otherwise, important data can be missed. The Data Destruction module tests the Disk Utility log for activity carried out by the GUI (Graphical User Interface) known as the Disk Utility. The GUI isn't the only means of data destruction built into OS X. Many commands issued in Terminal can be used to delete or destroy data.

For example, prior to the El Capitan release of OS X, there used to be a feature to allow the Trash Can to do a secure erase. Due to a limitation in that feature, Apple could not guarantee that this feature could do what it appeared to do, which is to guarantee a file or folder is securely erased. Accordingly, that feature is not available in El Capitan. Despite such, one can effectively carry out the same task in Terminal, by issuing the "srm" command, which is the secure version of the "rm" command, which means to remove or delete a file or folder. The 'srm' command, in its default mode, will make 35 passes as it overwrites the data. That is DOD-grade wiping on steroids, so don't doubt for one minute its effectiveness in destroying data.

So where does one find evidence of commands used in Terminal? The answer lies in the history file, which exists for each user. To view this file in Recon, look to the Bash History view, as shown in Figure 18 below. By searching for 'rm' we find, in addition to some extraneous hits, one "rm" and one "srm" command in this case.

In OS X, there are no timestamps for each action in the history file, however, since the commands appear in the sequence executed, one can sometimes impute a range when something occurred by examining other items before and after the command in question. Sometimes you can see a command and find the results of that command and obtain timestamps in that manner.

Let's consider one more Terminal command that destroys data. Recall earlier we discussed how to "Erase Free Space" using the GUI, Disk Utility. If that occurred using the GUI, we would expect an entry in the DiskUtility.log file. Starting with the El Capitan release of OS X, Disk Utility was stripped of much of

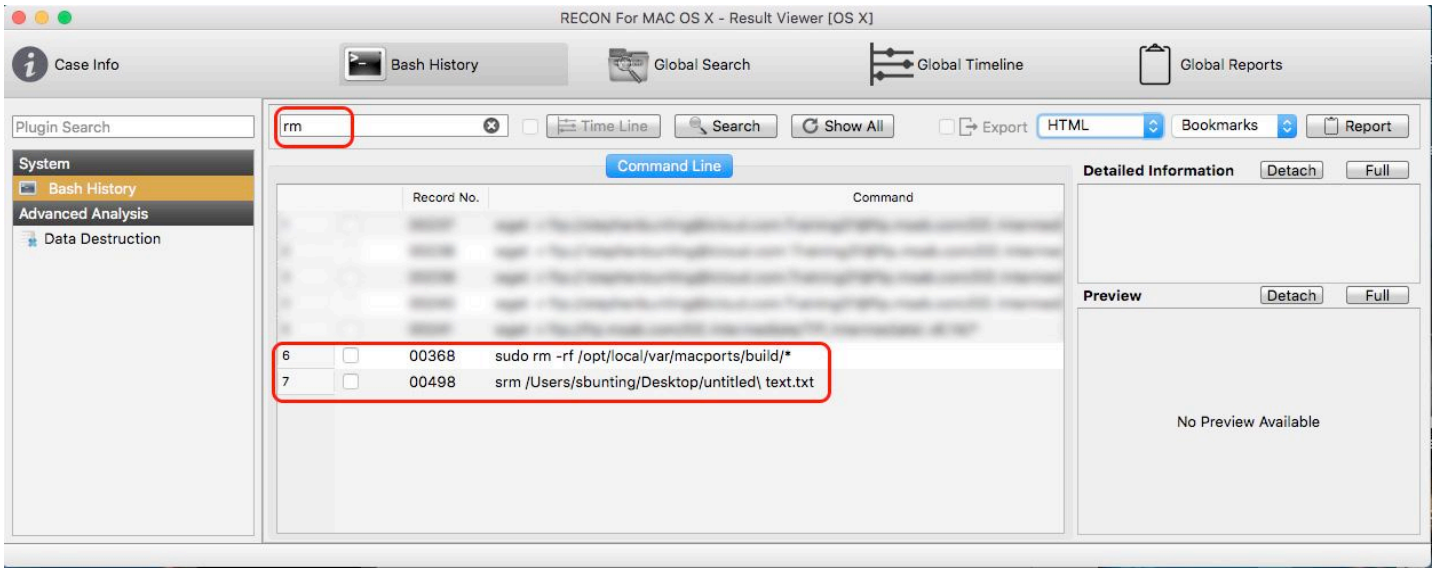

Figure 18 - An 'rm' and a 'srm' command found in the Terminal bash history file.

its former functionality. To wipe free space now, a terminal command will get the job done. That command is `diskutil secureErase freespace LEVEL /Volumes/DRIVE`, where `DRIVE` is replaced with the name of the mounted volume and `LEVEL` is replaced with a number from 0 to 4, which means:

- 0 writes zeroes to the disk once
- 1 writes a series of random numbers
- 2 writes zeroes 7 times
- 3 writes zeroes 35 times
- 4 writes zeroes 3 times

This command would also appear in the history file and one should search for the string "secureErase" to locate this activity. The command "diskutil" is followed by many 'verbs' that are associated with various data destroying activities. Below is a list of strings for which you should search in the history file for other forms of possible spoliation:

- eraseDisk
- eraseVolume
- reformat
- eraseOptical
- zeroDisk
- randomDisk
- secureErase
- partitionDisk

Alternatively, you may just want to search for "disktutil", which will return all activity involving "diskutil", which is shown below in Figure 19. You'll have, perhaps, more data to review, but you will be more thorough with that methodology.

Conducting a spoliation examination is a specialized exam for sure. The focus is on the various means of data destruction that are built into the operating system and also third-party tools. In addition, you are also making many before and after comparisons to show that data once present was deleted. As you have seen, the tools can be found as well as the patterns and artifacts they leave behind. It is very difficult to interact with a computer without leaving behind trace evidence. Spoliation is no different. There is usually plenty of evidence for the skilled examiner to establish spoliation when it occurs. This is not a complete treatise on spoliation, nor could it be. It is, however, a road map to follow that will uncover most of the more common spoliation activities. As always, one must be alert to the unusual and to always be inquisitive.

```
Mac-Pro7516:~ sbunting$ history |grep -i diskutil
  510  sudo diskutil secureErase freespace 1 /Volumes/NO\ NAME
  516  history |grep -i diskutil
Mac-Pro7516:~ sbunting$
```

Figure 19 - From the OS X terminal interface, history is shown, but piped to a grep command that is filtering for the string 'diskutil'. This returns all entries with 'diskutil'.

Part 2 of this 2-part series will focus on spoliation evidence found on the Windows operating system. The principles are largely the same, but the two systems are very different and hence the artifacts left behind.

References:

1. Zubulake v. UBS Warburg, LLC, 229 F.R.D. 422 (S.D.N.Y. 2004)
2. 342 Ark at 146, 27 S.W.3d at 388 (quoting BLACK'S LAW DICTIONARY 1401 (6th ed. 1990) (content in brackets inserted by the court)).

About the author: Steve Bunting

Steve Bunting, the author, is one of the pioneers in the field of digital forensics with over 17 years in the field. He spent ten of those years in digital forensics during his 35-year law enforcement career and seven of those years in support of the private sector.

He has been a presenter at several seminars and workshops, the author of numerous "white papers", the principal author of EnCase Computer Forensics - The Official EnCE: EnCase Certified Examiner Study Guide, 3rd Edition, the co-author of Mastering Windows Network Forensics and Investigation , the author of EnCase Computer Forensics—The Official EnCE: EnCase Certified Examiner Study Guide, 2nd Edition, the co-author of Mastering Windows Network Forensics and Investigation 2nd Edition, the author of EnCase Computer Forensics—The Official EnCE: EnCase Certified Examiner Study Guide, 3rd Edition (all published by Wiley).