Forensic Images: For Your Viewing Pleasure

GIAC (GCFA) Gold Certification

Author: Sally Vandeven, sallyvdv@gmail.com
Advisor: Barbara Filkins

Accepted: September 15, 2014

Abstract

For a student just getting started in digital forensics, concepts regarding forensic images can be confusing. Terminology like images, clones, bit-stream copies and forensic images are often used incorrectly, further complicating the issue. This paper will attempt to clear up the confusion. We will present an instructive clarification of what a forensic image is as well as what it is not. In addition, we will provide a comprehensive look at the many different ways to access data on forensic images using mostly open source tools on both Windows and Linux platforms.

1. Introduction

Digital forensic investigations often involve creating and examining disk images. A disk image is a bit-for-bit copy of a full disk or a single partition from a disk. Because the contents of a disk are constantly changing on a running system, disk images are often created following an intrusion or incident to preserve the state of a disk at a particular point in time. The image is a static "snapshot" that can then be analyzed to determine the events surrounding an incident and it may also be used as evidence in a courtroom (Prosise & Mandia, 2003). Forensic examiners use imaging techniques to acquire data from a disk as opposed to copying files because an image contains every bit of data from the source disk and a copy operation will only retrieve currently accessible files. A ***forensic image*** will contain current files as well as ***slack space*** and ***unallocated space***. Relevant forensic artifacts such as, deleted files, deleted file fragments and hidden data may be found in slack and unallocated space.

There are many different terms used to refer to image files, which can confuse the student new to the field. In addition, there are many different tools available for creating and handling image files.

This paper will attempt to clarify the terminology and address some of the more common free tools and techniques used to acquire disk images as well those used for mounting and accessing the acquired images. We will focus primarily on acquiring disk images from Windows computers although many of the tools used to acquire and access the images run on the Linux platform. All tools described in this paper are free tools and many are also open source.

Disk images represent evidence that could potentially become part of a criminal or civil investigation; therefore, proper evidence handling techniques are important.[1]

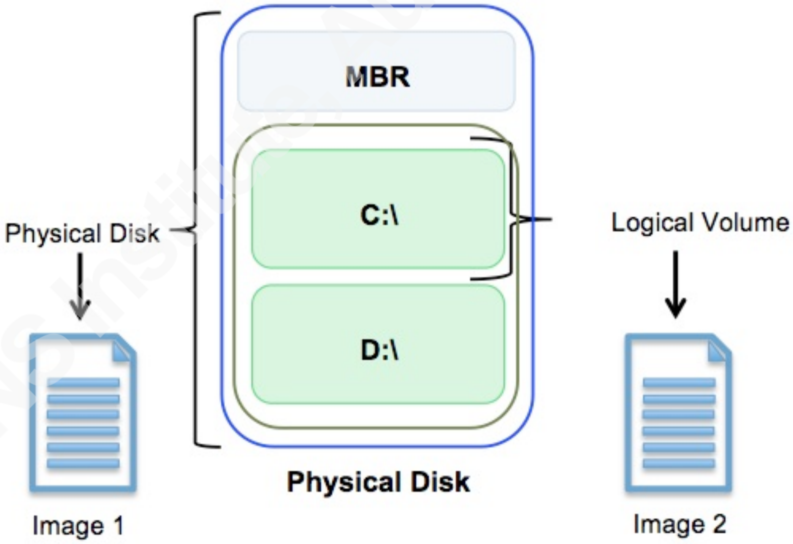
Appendix A contains a glossary of terms used throughout this paper. The first instance of every term found in the glossary will be italicized and bold.

[1] While out of scope for this paper, evidence-handling techniques are covered thoroughly in the book *Incident Response & Computer Forensics* (Prosise & Mandia, 2003).

2. Disk Image Sources

The physical disk shown in Figure 1 contains a ***master boot record*** (MBR) and a ***partition*** with two ***logical volumes***, each formatted with a ***file system***. The master boot record contains ***metadata*** about the layout of the disk including where the partitions begin and end. When all the data from a disk are duplicated exactly from the source disk and stored in a file, the resulting ***bit-for-bit copy*** is called an ***image*** file (Carrier, 2005, Ch. 3).[2] The source disk can be an entire ***physical disk*** that might contain one or more ***logical volumes*** as captured by Image 1 in Figure 1. The source of the image could also be a logical volume within a physical disk, such as Volume C:\ and Image 2. In both cases, an image is a file that will always contain an exact copy of the data that represents a snapshot of the source at the point in time when it was copied.

Figure 1: Logical volumes contain file systems within a physical disk



2.1. Files and Folders

Images can also be created from individual files and folders using many of the forensic imaging tools, however, the resulting images will not be considered forensic

[2] When all the data from a disk are duplicated exactly to another disk it is called a clone. Clones will be discussed in Section 3.1

images because it will only contain the data from the existing file and will not include unallocated space. This will be addressed in more detail in Section 4.4

2.2. Additional Challenges When Source is Solid State Drive

Solid-state drives (SSDs) present challenges to forensic investigators primarily because disk technologies called ***program and erase cycles*** and ***wear-leveling*** that cause unallocated space to get overwritten sooner than it would on mechanical disk drives during normal operation (Kumar & Vijayaraghavan, 2011). This causes quicker loss of potentially relevant data that could be extracted from unallocated space on solid-state drives. With respect to forensic imaging, however, SSDs and hard disk drives look the same logically to the forensic acquisition tools and can be imaged using the same methods as traditional disks (Weibe, 2013). An image created from an SSD drive will still represent a snapshot of all data on the SSD at the time of imaging.

3. Disk Images

3.1. Terminology

As described previously, when a logical volume or physical disk is copied bit-for-bit into a file, the resulting file contains a duplicate copy of the source disk and is called an image file. The terms forensic image, ***forensic duplicate*** and ***raw image*** are all used to refer to this bit-for-bit image file (Prosise & Mandia, 2003). This paper will use the term forensic image most frequently, as this seems to be the most common. If a forensic image is not compressed it will be the same size as the source disk or volume. Forensic images are frequently compressed to save space and additionally may be separated into multiple files called ***split images*** that are more manageable. In either case, when a compressed forensic image is uncompressed or split images are recombined, the result will be an exact duplicate of the source.

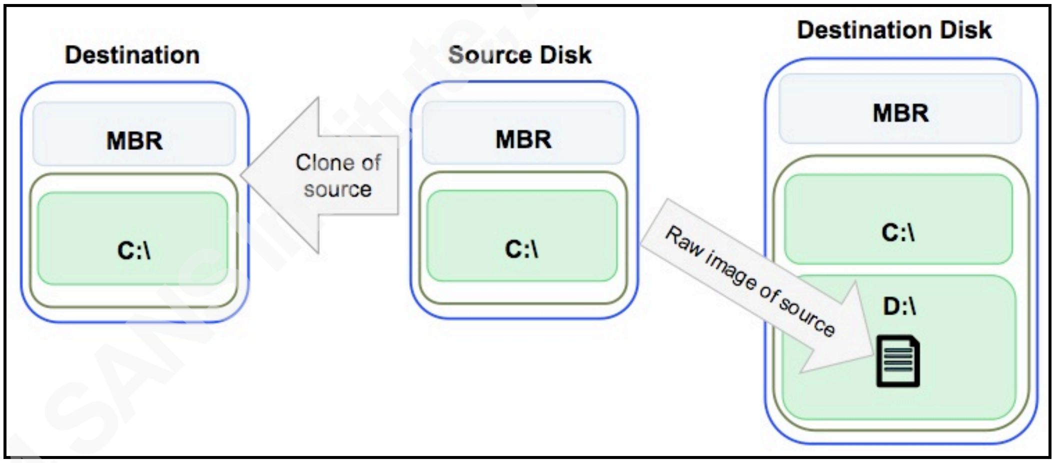
In forensics, ***embedded images*** are disk images that contain metadata about the image such as a timestamp when the image was created and a ***cryptographic hash*** acting as a ***fingerprint*** for the image (Carrier, 2005, Ch. 3).

3.1.1. Image vs. Clone

The terms image and ***clone*** are not synonymous. A forensic image is a duplicate of a physical drive that is written to a file. The file can be stored on an internal disk, on an external disk, on a network storage server or on optical storage, etc. A disk clone is a bit-for-bit copy of a physical disk directly on to another disk (Carrier, 2005). The original and the clone are identical and interchangeable[3]. Cloning is usually used to make a backup or duplicate of the operating system drive because it can be very quickly replaced in the case of drive failure.

Figure 2 compares a cloned disk to a disk image. On the left, the output from the cloning process is another entire disk that is an exact copy of the source, including the metadata in the MBR. On the right, the output of the imaging process is an image file, stored on another disk. Inside that image file is the exact copy of the source disk, including the metadata in the MBR.

Figure 2: A *clone* is a duplicate disk and an *image* is a duplicate of the data from the disk that is written to a file



Although a cloned disk could be used for forensic analysis, it presents other problems related to maintaining the forensic integrity of the data on the destination drive unless it is always mounted read-only, therefore, is not a common imaging technique

[3] Differences in disk geometry between the source disk and the cloned disk could present problems but this is out of scope for this paper.

(Carrier, 2005). Furthermore, because most forensic analysis tools expect to see forensic image files, the preferred data acquisition method is the creation of a forensic image file.

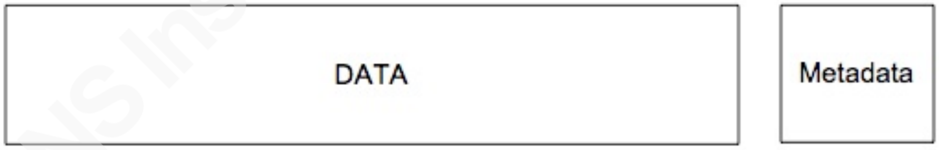
3.2. Image File Formats

The research done for this paper finds the two formats that are most widely used today are the Encase evidence file format (often called Expert Witness format or E01 images) and raw image file formats. The Expert Witness File format recently added a new format called Ex02, however, support for the new format is scarce as of this writing. In addition to these formats, there are several other image file formats found in the digital forensics literature; AFF, AFF4, Expert Witness File format, Logical Evidence File and SMART but there is little evidence of significant support for these currently. Therefore, this paper will only include discussion of tools and techniques that apply to raw images and E01 images.

3.2.1. Raw images

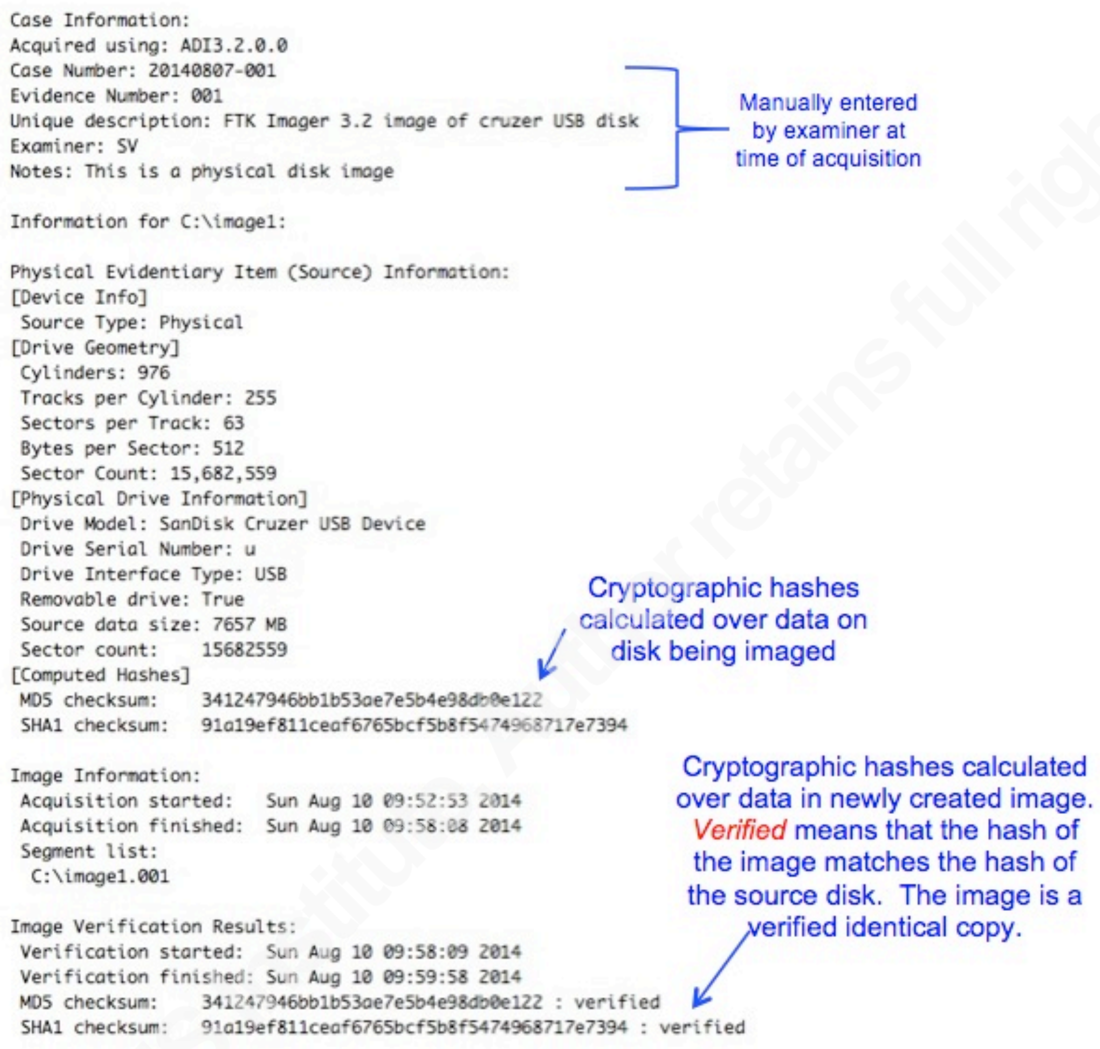
As described in Section 3.1, a raw image contains only the data from the source disk. There is no header or metadata included in the image file but some utilities may include a separate file with metadata as represented by Figure 3.

Figure 3: A raw image contains only source data. Metadata may be in separate file



FTK Imager is a free tool developed by The Access Data Group for creating disk images (Access Data, n.d.). An example of a metadata file associated with a raw image generated by Access Data's FTK Imager is shown in Figure 4. Image creation tools will be described in more detail in Section 4.

Figure 4: FTK Imager generates a metadata file associated when imaging a disk



```
Case Information:
Acquired using: ADI3.2.0.0
Case Number: 20140807-001
Evidence Number: 001
Unique description: FTK Imager 3.2 image of cruzer USB disk
Examiner: SV
Notes: This is a physical disk image

Information for C:\image1:

Physical Evidentiary Item (Source) Information:
[Device Info]
 Source Type: Physical
[Drive Geometry]
 Cylinders: 976
 Tracks per Cylinder: 255
 Sectors per Track: 63
 Bytes per Sector: 512
 Sector Count: 15,682,559
[Physical Drive Information]
 Drive Model: SanDisk Cruzer USB Device
 Drive Serial Number: u
 Drive Interface Type: USB
 Removable drive: True
 Source data size: 7657 MB
 Sector count:    15682559
[Computed Hashes]
 MD5 checksum:    341247946bb1b53ae7e5b4e98db0e122
 SHA1 checksum:   91a19ef811ceaf6765bcf5b8f5474968717e7394

Image Information:
 Acquisition started:   Sun Aug 10 09:52:53 2014
 Acquisition finished:  Sun Aug 10 09:58:08 2014
 Segment list:
  C:\image1.001

Image Verification Results:
 Verification started:  Sun Aug 10 09:58:09 2014
 Verification finished: Sun Aug 10 09:59:58 2014
 MD5 checksum:    341247946bb1b53ae7e5b4e98db0e122 : verified
 SHA1 checksum:   91a19ef811ceaf6765bcf5b8f5474968717e7394 : verified
```

Manually entered by examiner at time of acquisition

Cryptographic hashes calculated over data on disk being imaged

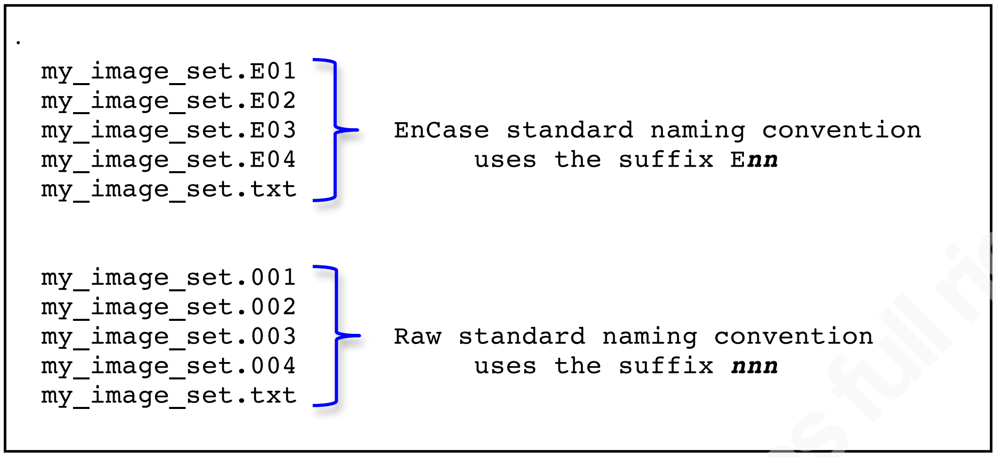
Cryptographic hashes calculated over data in newly created image. *Verified* means that the hash of the image matches the hash of the source disk. The image is a verified identical copy.

Raw images can be created by several different utilities and frequently will use the following file extensions:

`.dd    .raw    .img`

When FTK or EnCase create split images they default to a naming convention that assigns a numeric suffix to each chunk of the image as shown in Figure 5.

Figure 5 Naming conventions for split image file sets



```
.
my_image_set.E01
my_image_set.E02
my_image_set.E03      EnCase standard naming convention
my_image_set.E04           uses the suffix Enn
my_image_set.txt

my_image_set.001
my_image_set.002
my_image_set.003      Raw standard naming convention
my_image_set.004           uses the suffix nnn
my_image_set.txt
```

3.2.2. EnCase 6 evidence file images (E01)

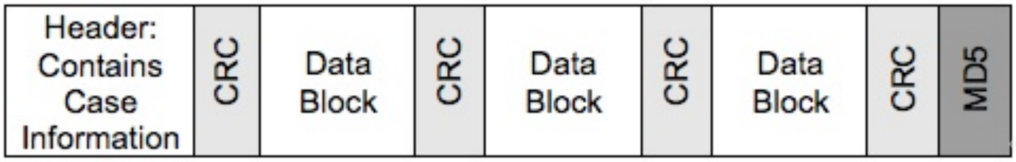
The popular commercial forensics suite, EnCase, developed a proprietary format called EnCase Evidence File format. EnCase Evidence Files use the file extension, E01, and are based on the Expert Witness Format (EWF) by ASR Data (Forensicswiki, 2012). These image files are commonly referred to as Expert Witness, E01 or EWF files.

E01 files have both a header and footer containing metadata about the image. The metadata includes the drive type, the version of EnCase that created the image, the source disk operating system, timestamps and a cryptographic hashes over the data portion of the image (Bunting, 2008, Ch. 5). E01 files also have a file integrity check calculation, called a ***Cyclical Redundancy Check*** (CRC), at 64 KB intervals[4] throughout the image. A CRC provides an integrity check over the previous block of data much like a cryptographic hash provides an integrity check over the entire image.

Figure 6 shows the logical layout of an EnCase formatted imaged. Most of the metadata is found in the header. The cryptographic hash, called an ***acquisition hash*** by EnCase, is calculated over the data blocks only and is appended at the end of the image.

[4] The default block size is 64KB but is configurable. A CRC is calculated after every block.

Figure 6: The format of EnCase evidence files. (Bunting, 2008, p. 180)



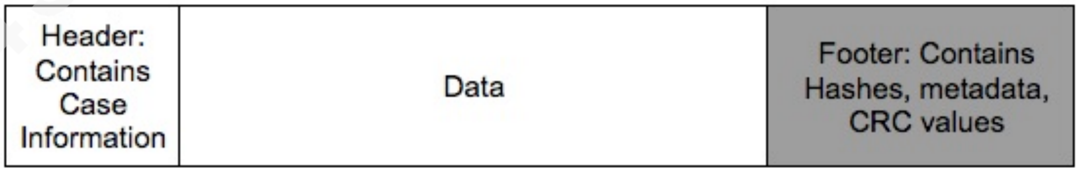
By default, E01 files are compressed at the block level using the zlib compression algorithm so that after a block of data is read from the source, a CRC is calculated and appended to the block. The block is then compressed and added to the image file. Performing frequent CRC checks over chunks of data provides a cross check of the integrity of the image with the MD5 or SHA-1 hash over the entire data stream (Bunting, 2008, Ch. 5).

3.2.3. EnCase 7 evidence file images (Ex01)

Access Data introduced a new format for its evidence files in version 7 of EnCase. One of the changes has to do with the algorithm used for data block compression, which changes the image file format.

Figure 7 shows a representation of the newest evidence file format for EnCase 7. Because it uses a different compression algorithm, bzip instead of zlib, CRCs are written to a table at the end of the file instead of after each data block (Bunting, 2012, Ch. 5). This is relevant because utilities that were able to read Encase 6 images must be adapted to accommodate the new file format.

Figure 7: The version 7 EnCase evidence file format


Header: Contains Case Information
Data
Footer: Contains Hashes, metadata, CRC values

The version 7 evidence files use the file extension Ex01 so the examiner can quickly identify the format and like Encase 6, are by default compressed but can be created without compression.

4. Creating Forensic Images

4.1. Dead vs. Live Imaging

In order to create a forensic image of an entire disk, best practice dictates that the imaging process should not alter any data on the disk and that all data, metadata and unallocated space be included. Traditionally, forensic investigators accomplish this by powering down the system and removing the disk (or disks) in order to connect it to a forensic workstation or hardware or software ***write-blocker*** to create the image. This is referred to as ***dead imaging***. A write-blocker, as its name implies, will prevent any data from being written to the disk, allowing read access only (Brown, 2010). Removing a disk from a running system prevents any further changes due to normal system operations or process and user interactions. Using a write-blocker during evidence acquisition preserves the integrity of the file metadata, such as timestamps that may be relevant to the investigation.

There are, however, times when a live system must be imaged without removing power (***live imaging***). For example, if the examiner determines that the disk in question uses a full disk encryption solution and does not have access to the decryption key then the unencrypted running volume should be imaged without powering off the system.

The responsible party may require that the system stay online for business reasons. Or if the data to be imaged resides in a remote location where physical access is limited or difficult then the only practical solution may be live imaging without removing power. Additionally, powering off a disk or system may tip-off an attacker or malicious process that it is being analyzed; causing a change in behavior that may further hinder the investigation.

4.2. Forensic Imaging Best Practices

When creating a disk image it is important to use proper evidence handling techniques in the event that the image is used as evidence in a courtroom (Prosise & Mandia, 2003), (SWGDE, 2013). In addition to using write-blockers to prevent alteration of the evidence, other important considerations facing the examiner are:

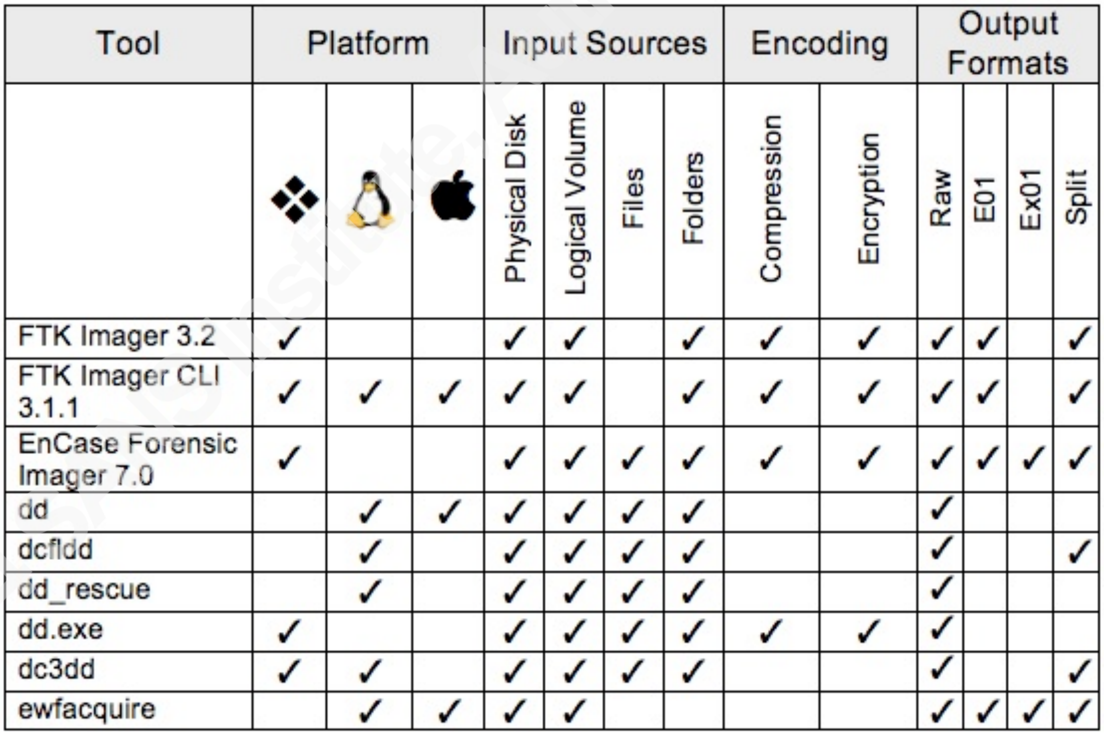
- Use of tested tools and techniques

- Check for disk encryption prior to removing power from disk
- Consideration of logical volume or physical drive acquisition
- Encrypting forensic images for confidentiality
- Establish proper chain of custody for the image

4.3. Image Creation Tools

Table 1 summarizes the common tools discussed in this paper that can be used to create image files along with the platforms on which each can be executed and what type of input it can take. Three of the imaging utilities provide native encryption capability. And finally, the table indicates the supported output formats.

Table 1: Image creation tools



Tool	Platform			Input Sources				Encoding		Output Formats			
	❖	🐧		Physical Disk	Logical Volume	Files	Folders	Compression	Encryption	Raw	E01	Ex01	Split
FTK Imager 3.2	✓			✓	✓		✓	✓	✓	✓	✓		✓
FTK Imager CLI 3.1.1	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓
EnCase Forensic Imager 7.0	✓			✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
dd		✓	✓	✓	✓	✓	✓			✓			
dcfldd		✓		✓	✓	✓	✓			✓			✓
dd_rescue		✓		✓	✓	✓	✓			✓			
dd.exe	✓			✓	✓	✓	✓	✓	✓	✓			
dc3dd	✓	✓		✓	✓	✓	✓			✓			✓
ewfacquire		✓	✓	✓	✓					✓	✓	✓	✓

The SANS Investigative Forensic Workstation (SIFT) is a collection of tools useful to forensics investigations, including the image creation tools *dd, dcfldd, dc3dd* and *ewfacquire* shown in Table 1. A SIFT workstation can be built on a Debian or

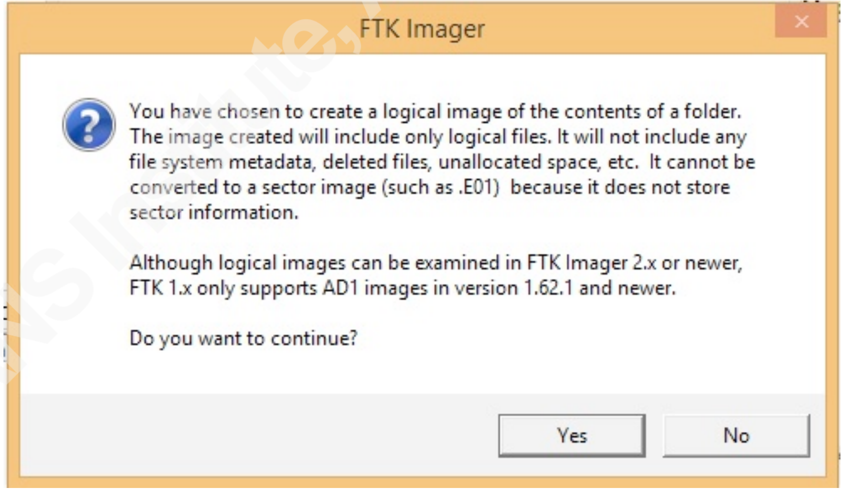
Ubuntu base using the installation instructions found at the URL https://github.com/sans-dfir/sift-bootstrap. It is distributed for free and regularly updated. (SANS, n.d.).

4.4. FTK Imager

The Access Data Group markets the Forensic Toolkit (FTK), a popular forensic analysis suite. Although FTK is a commercial product, Access Data provides the imaging tool, FTK Imager, for free. FTK Imager can be downloaded from http://www.accessdata.com/support/product-downloads. The free download also contains a detailed user manual.

As of this writing, the current version of FTK Imager, version 3.2.0, is able to create forensic images of physical disks, logical volumes and previously created images. It can create an image at the folder level as well but these images will not be considered forensic images because they will not contain slack or unallocated space as shown by the warning in Figure 8.

Figure 8: Message from FTK Imager when imaging a folder



You have chosen to create a logical image of the contents of a folder. The image created will include only logical files. It will not include any file system metadata, deleted files, unallocated space, etc. It cannot be converted to a sector image (such as .E01) because it does not store sector information.

Although logical images can be examined in FTK Imager 2.x or newer, FTK 1.x only supports AD1 images in version 1.62.1 and newer.

Do you want to continue?

Using the FTK graphical user interface, a user can create a forensic image of disk, volume or another image by selecting the “Create Disk Image…” option from the file menu. The wizard will walk the user through selecting the source disk, the output format, the image file destination, the image file name, encryption options, etc.

Another interesting feature of the FTK Imager is the ability to create a directory listing of all files within the image. After creating the image, if it can parse the file system, it will output a directory listing to a separate CSV file using the naming convention

```
<image name>.001.csv
```

The directory listing will contain the full path to the file, file name, size, timestamps and an "is deleted" indication.

FTK also offers encryption of images during the image creation process. The specific implementation is not documented publicly but the FTK Imager User Manual indicates that the image will be encrypted using the AES symmetric key encryption algorithm with either a 128, 192 or 256 bit key size[5]. The AES symmetric key is protected using the users choice of a passphrase or an asymmetric key; a selection that is part of the image creation dialogue.

The detailed steps with screenshots for completing the imaging operation are found in the FTKImager_UserGuide.pdf that is included in the FTK Imager download from the Access Data Group.

4.5. FTK Imager CLI

The FTK Imager also has command line versions for Windows, Linux and OS X. The current version running on all three platforms is version 3.1.1 from August 2012. The command line version has most of the same functionality as the Windows graphical version with a few exceptions. It cannot create a directory listing of the files in the image nor can it use an overflow location[6]. By default it will create a raw image but can be configured to create an image in E01 format by using the `–e01` option.

The testing done for this paper also concludes that the Windows command line imager version 3.1.1 is not able to properly verify images that have been created by FTK

[5] The user manual states that 128, 192 or 256 bit AES can be used but AccessData Group responded to this author's inquiry and reported that AES-256 is always used and is not configurable.

[6] Overflow location offers FTK a second location to write output files in the event that the primary destination disk cannot accept more data.

Imager using its built-in Access Data (AD) proprietary encryption. This functionality works properly for FTK CLI on Linux and OS X. The Windows FTK CLI is, however, able to properly verify images that have not been encrypted.

The command line utility does not come with a user manual but has built-in help that can be accessed in all versions with the command:

```
$ ftkimager  --help
```

All command line versions for FTK can be downloaded for free at http://www.accessdata.com/support/product-downloads.

4.6. Encase Forensic Imager

Like FTK, Encase Forensic from Guidance Software is a full-featured software package that can create images and perform analysis. Encase Forensic is a commercial tool but Guidance Software offers Encase Forensic Imager as a free product. The utility can be downloaded from https://www.guidancesoftware.com/products/Pages/Product-Forms/Forensic-Imager-download.aspx. The Encase Imager User's Guide is also available from the download site.

The Encase Forensic Imager can image physical drives, logical volumes, individual files or folder and other image files. It will only output to the Expert Witness formats E01 and Ex01. If the newer EnCase version 7 format, Ex01 is selected, the image can be compressed using the bzip algorithm and encrypted using the AES-256 encryption cipher (NIST, 2001) (Bunting, 2012, Ch. 5).

Guidance Software had an acquisition tool that ran on Linux platforms called Linen. According to a telephone support representative at Guidance Software, it has been discontinued in the newest version of Encase Forensic, version 7. Version 7 has instead expanded it capability to recognize Linux and Unix specific file systems with its commercial Encase Forensic suite that runs on the Windows platform.

4.7. Dd

The software utility *dd* was originally developed for Unix in the 1970's (Chessman, 1996). There are currently versions that run on Unix, Linux, OS X and

Windows. The version that has been ported to Windows is called dd.exe and is described in Section 4.7. *Dd* is a very simple utility that reads data from a source and writes it to a destination. In the Linux operating system, a ***raw device*** is a software mechanism that allows direct access to a disk, or "raw" access to the disk. The *dd* utility accesses the data on the disk as if it were "underneath" the file system without any regard to how that data is organized or what file system has been configured on the disk (Carrier, 2005, Ch. 3). In this way, *dd* is able to capture the files on the disk as well as the slack and unallocated space. The *dd* utility is included in OS X and all major Linux distributions by default. The *dd* man page contains syntax and usage examples.

4.8. Dd.exe

Dd.exe is one of the tools in the Forensic Acquisition Utilities suite written by George Garner. *Dd.exe* is a Windows implementation of the Linux *dd* utility. It functions very much like the Unix/Linux dd utility with some notable exceptions. Dd.exe can send output directly to a specified TCP or UDP port. This would be useful for network data acquisition for example. Dd could be made to do this as well by setting up a netcat relay for example, however, this is built into to dd.exe as a command line option. Dd.exe also has built-in support for compression (zlib) and encryption of the raw image output. Built-in help can be viewed using

```
C:\>  dd.exe  --help   or   C:\> dd.exe  /?
```

The utility can be downloaded for free at http://www.gmgsystemsinc.com/fau/

4.9. Dcfldd

A former member of the U.S. Department of Defense created Dcfldd in 2002. It is similar to *dd* but it has added features that are useful for forensic acquisitions. The added features include a hash comparison of the source and created image, a flexible naming convention for the split image files, a flexible naming convention for the metadata file containing the image information, a progress indicator during the imaging process and the ability to send output to multiple locations simultaneously.

Dcfldd is a Linux/Unix utility only and contains built-in help using the command

```
# dcfldd  --help     or     # man  dcfldd
```

The most recent update to *dcfldd* is from 2006 and can be downloaded from http://dcfldd.sourceforge.net/.

4.10. Dc3dd

Dc3dd is an imaging tool written in 2008 and is similar to *dcfldd*. It is a command line imager that creates raw disk images but has added features that make it useful in forensics. A graphical interface has been developed for *dc3dd* called *Automated Image and Restore* (AIR) and can be downloaded at http://sourceforge.net/projects/air-imager/. The main differences between *dcfldd* and *dc3dd* are related to current changes to the standard *dd* command. *Dcfldd* was a new project using the underlying *dd* code from 2002. *Dc3dd* provides additional functionality to the current *dd* command so that updates to *dd* automatically apply to *dc3dd*. Like *dcfldd*, *dc3dd* provides hash comparisons, flexible-naming conventions for split image files and metadata file, a progress indicator and output to multiple files/locations. The utility can be built and run on Linux and OS X systems. *Dc3dd* has extensive built-in help using the command

```
# dc3dd  --help
```

The latest version of *dc3dd* can be downloaded from http://sourceforge.net/projects/dc3dd/

4.11. Dd_rescue

Dd_rescue was written by Kurt Garloff and was initially released in 1999. It is still being updated currently. As of this writing, the most recent update was in August of 2014. The project page that includes download links is at http://www.gnu.org/software/ddrescue/ddrescue.html.

Dd_rescue is not a derivative of the *dd* command but it functions similarly. It was written to help recover data from files or entire disks when the disks are encountered errors that cause a regular *dd* copy operation to fail. Disks are always read from or written to in chunks called sectors. When a bad sector is encountered, *dd_rescue* will skip it and move on to the next sector. When there are no errors detected, *dd_rescue* will use a large block size such as 16KB for faster copying. When it detects errors it will reduce the block size to the lowest possible for that hardware, typically 512 bytes. By reducing the block size it will reduce the amount of data skipped when bad sectors are encountered. *dd_rescue* also allows the user to copy a disk in the reverse direction in order to maximize the chances of accessing bad sectors.

The user manual for dd_rescue is at http://www.gnu.org/software/ddrescue/manual/ddrescue_manual.html and the built-in help can also be viewed with

```
# ddrescue –help  or  # man  ddrescue
```

4.12. Libewf and Ewfacquire

Joachim Metz has published an open source library of utilities called *libewf* that will read and write EnCase version 6 and 7 evidence files (Metz, n.d.). As of this writing, the library supports E01 (compressed and uncompressed) and Ex01 (uncompressed only). The library has multiple utilities but of interest here is the utility to create images, *ewfacquire*.

Ewfacquire will acquire data and write to a single file or split files in an EnCase Evidence file format, E01. It also supports the newest version 7, Ex01 uncompressed format. It can acquire data from physical and logical disks and files. *Ewfacquire* is useful to convert images to other formats, for example, an Ex01 image can be converted to an E01 image to use with tools that may not yet support the newer Ex01 format such as Volatility (Levy, 2014).

The utility has built-in help that can be accessed with the command

```
$ ewfacquire  --help  or  $  man  ewfacquire
```

Ewfacquire and the other utilities included with the libewf suite can be downloaded from https://code.google.com/p/libewf/.

5. Accessing Forensic Images for Analysis

An analyst can either mount the image to see a file system view of the data or she can analyze the image file directly using a variety of tools typically tailored for forensics, such as FTK, EnCase and others. The access method chosen depends on what data the analyst is looking for.

Some of the reasons that one may choose to mount an image file and interact with the data via the file system include:

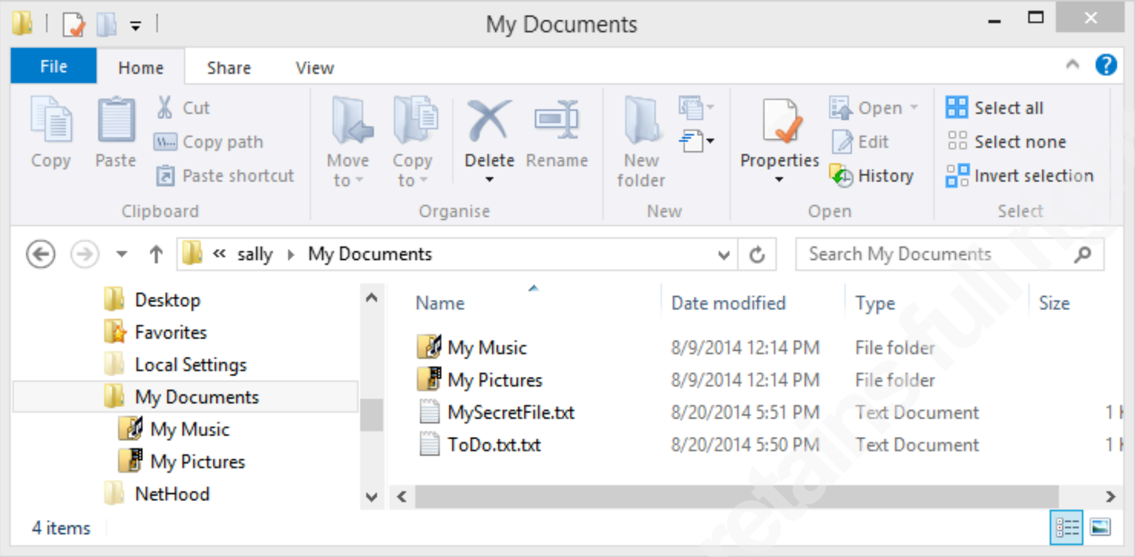
- To run anti-virus software against files
- To view files using their native applications
- To browse the file system with Windows Explorer as the user did
- To facilitate running scripted parsing and searching from Windows or Linux
- To easily export files
- If not interested in unallocated space

Some of the reasons that one may choose to analyze an image file directly include:

- To perform keyword searches that will also search in unallocated space.
- To extract deleted files or file fragments

Figure 9 shows an example of a mounted image as viewed through Windows Explorer. It represents an image of a Windows XP system disk. The folder sally\My Documents shows the presence of two text files. In contrast, when the same image is viewed using a forensic tool that examines all the files as well as the unallocated space, one can see the presence of additional artifacts. The additional artifacts are deleted files that are now in unallocated space. The unallocated space will eventually be allocated to a new file and the data overwritten but until then the data remain in place.

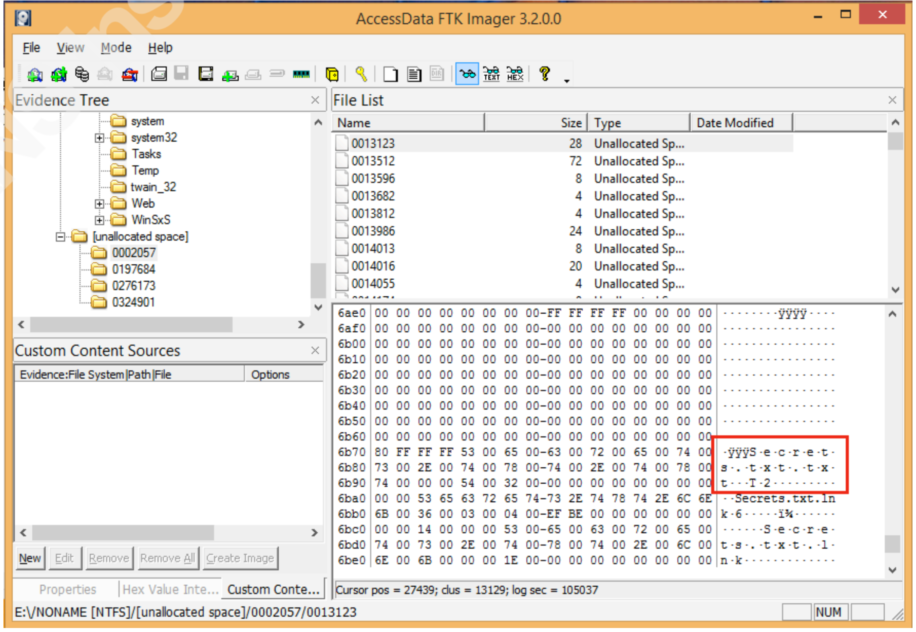
Figure 9: A forensic image mounted and viewed using Windows File Explorer.



Name	Date modified	Type	Size
My Music	8/9/2014 12:14 PM	File folder	
My Pictures	8/9/2014 12:14 PM	File folder	
MySecretFile.txt	8/20/2014 5:51 PM	Text Document	1
ToDo.txt.txt	8/20/2014 5:50 PM	Text Document	1

Figure 10 shows the view of the same file system when the image file is not mounted but is examined with a forensic analysis tool that is designed to access all data on the disk including deleted files in unallocated space. In this example, reference to a file called secrets.txt.txt is found in unallocated space on the disk image. Because this record has not yet been overwritten we know that file existed at one time.

Figure 10: A forensic image file is examined using a forensic analysis tool



Name	Size	Type	Date Modified
0013123	28	Unallocated Sp...	
0013512	72	Unallocated Sp...	
0013596	8	Unallocated Sp...	
0013682	4	Unallocated Sp...	
0013812	4	Unallocated Sp...	
0013986	24	Unallocated Sp...	
0014013	8	Unallocated Sp...	
0014016	20	Unallocated Sp...	
0014055	4	Unallocated Sp...	

```
6ae0 00 00 00 00 00 00 00 00-FF FF FF FF 00 00 00 00  ········ÿÿÿÿ····
6af0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b10 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b20 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b30 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b40 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b50 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ················
6b60 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
6b70 80 FF FF FF 53 00 65 00-63 00 72 00 65 00 74 00  ·ÿÿÿS·e·c·r·e·t·
6b80 73 00 2E 00 74 00 78 00-74 00 2E 00 74 00 78 00  s·.·t·x·t·.·t·x·
6b90 74 00 00 00 54 00 32 00-00 00 00 00 00 00 00 00  t···T·2·········
6ba0 00 00 53 65 63 72 65 74-73 2E 74 78 74 2E 6C 6E  ··Secrets.txt.ln
6bb0 6B 00 36 00 03 00 04 00-EF BE 00 00 00 00 00 00  k·6·····ï¾······
6bc0 00 00 14 00 00 00 53 00-65 00 63 00 72 00 65 00  ······S·e·c·r·e·
6bd0 74 00 73 00 2E 00 74 00-78 00 74 00 2E 00 6C 00  t·s·.·t·x·t·.·l·
6be0 6E 00 6B 00 00 00 1E 00-00 00 00 00 00 00 00 00  n·k·············
```

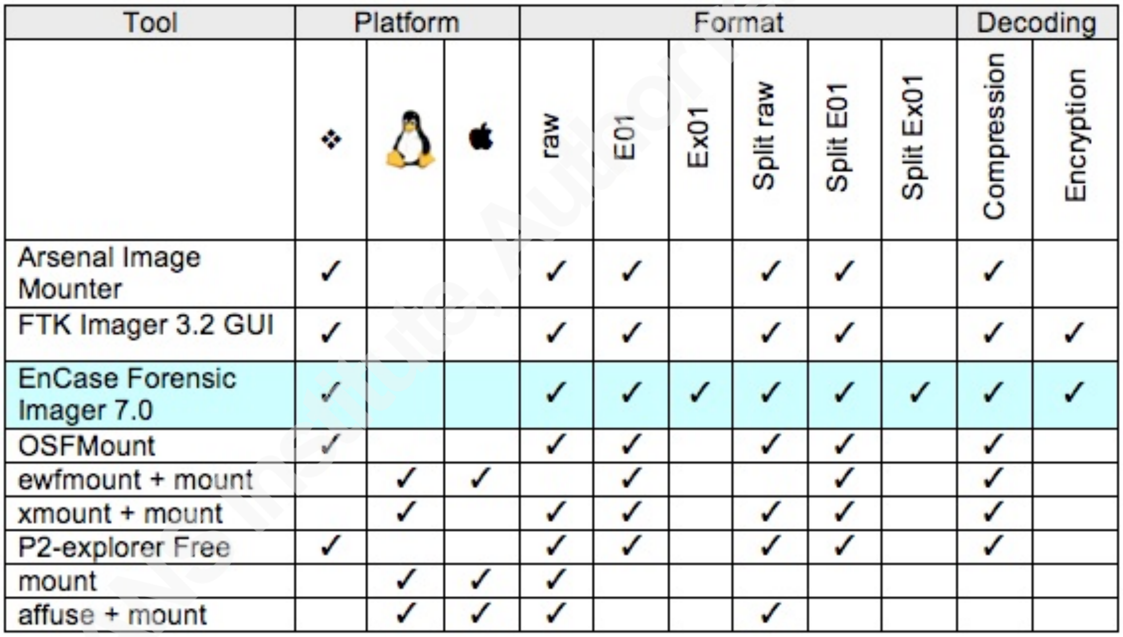
Cursor pos = 27439; clus = 13129; log sec = 105037

E:\/NONAME [NTFS]/[unallocated space]/0002057/0013123

5.1. Mounting Image Files

Table 2 lists common tools for mounting image files and the formats and features supported by each. The EnCase Forensic Imager, highlighted in blue, provides a simple preview of an image within that utility only. It does not perform a file system mount that is accessible by Windows File Explorer or any other tools. It is included in the table for completeness because it is the only tool that can mount Ex01 images. Each of the tools in Table 2 was tested in a lab and the results for each are discussed below. The SANS SIFT Workstation version 3 includes the utilities: *mount, ewfmount, xmo*unt and *affuse*.

Table 2: Common tools used to mount forensic images for file system access



Tool	Platform			Format						Decoding	
	❖	🐧		raw	E01	Ex01	Split raw	Split E01	Split Ex01	Compression	Encryption
Arsenal Image Mounter	✓			✓	✓		✓	✓		✓	
FTK Imager 3.2 GUI	✓			✓	✓		✓	✓		✓	✓
EnCase Forensic Imager 7.0	✓			✓	✓	✓	✓	✓	✓	✓	✓
OSFMount	✓			✓	✓		✓	✓		✓	
ewfmount + mount		✓	✓		✓			✓		✓	
xmount + mount		✓		✓	✓		✓	✓		✓	
P2-explorer Free	✓			✓	✓		✓	✓		✓	
mount		✓	✓	✓							
affuse + mount		✓	✓	✓			✓				

5.2. Windows Utilities to Mount a Forensic Image

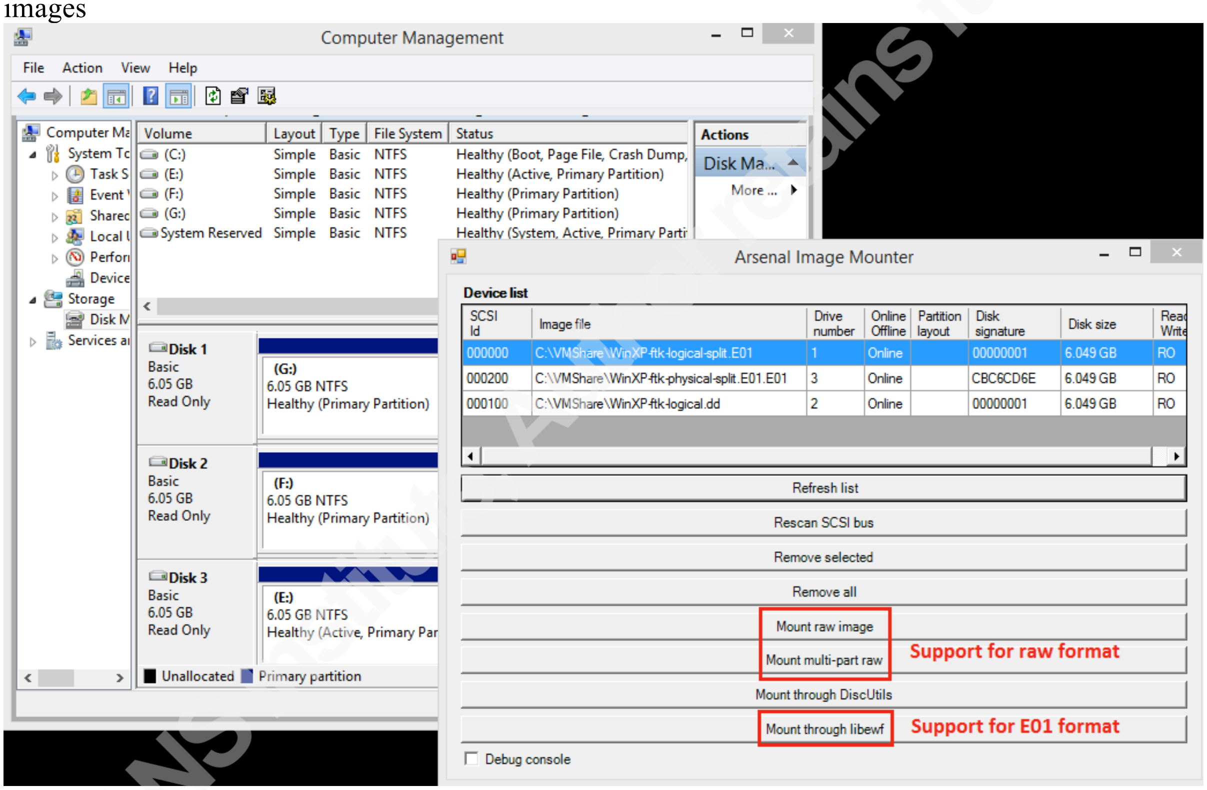
5.2.1. Arsenal Image Mounter

Arsenal Image Mounter (AIM) is a free tool developed by Arsenal Recon (Arsenal Recon, 2014). It can mount raw, split raw, E01 and split E01 logical and physical disk images. It does not support Ex01 images. It is a Windows application that installs quickly and easily. By default it mounts images read only. In addition to

Windows file systems, AIM supports images of HFS+, UFS and EXT3 file systems. Although there is no built-in help, the application is very easy to use.

Figure 11 Shows the Image Mounter tool from Arsenal Recon. Three images are mounted and the associated drives are detected by the Windows Disk Manager.

Figure 11: Arsenal Recon Image Mounter tool runs on Windows and mounts raw, E01 and split images



Volume	Layout	Type	File System	Status
(C:)	Simple	Basic	NTFS	Healthy (Boot, Page File, Crash Dump,
(E:)	Simple	Basic	NTFS	Healthy (Active, Primary Partition)
(F:)	Simple	Basic	NTFS	Healthy (Primary Partition)
(G:)	Simple	Basic	NTFS	Healthy (Primary Partition)
System Reserved	Simple	Basic	NTFS	Healthy (System, Active, Primary Parti

Disk 1
Basic
6.05 GB
Read Only

(G:)
6.05 GB NTFS
Healthy (Primary Partition)

Disk 2
Basic
6.05 GB
Read Only

(F:)
6.05 GB NTFS
Healthy (Primary Partition)

Disk 3
Basic
6.05 GB
Read Only

(E:)
6.05 GB NTFS
Healthy (Active, Primary Par

■ Unallocated ■ Primary partition

SCSI Id	Image file	Drive number	Online Offline	Partition layout	Disk signature	Disk size	Read Write
000000	C:\VMShare\WinXP-ftk-logical-split.E01	1	Online		00000001	6.049 GB	RO
000200	C:\VMShare\WinXP-ftk-physical-split.E01.E01	3	Online		CBC6CD6E	6.049 GB	RO
000100	C:\VMShare\WinXP-ftk-logical.dd	2	Online		00000001	6.049 GB	RO

Support for raw format

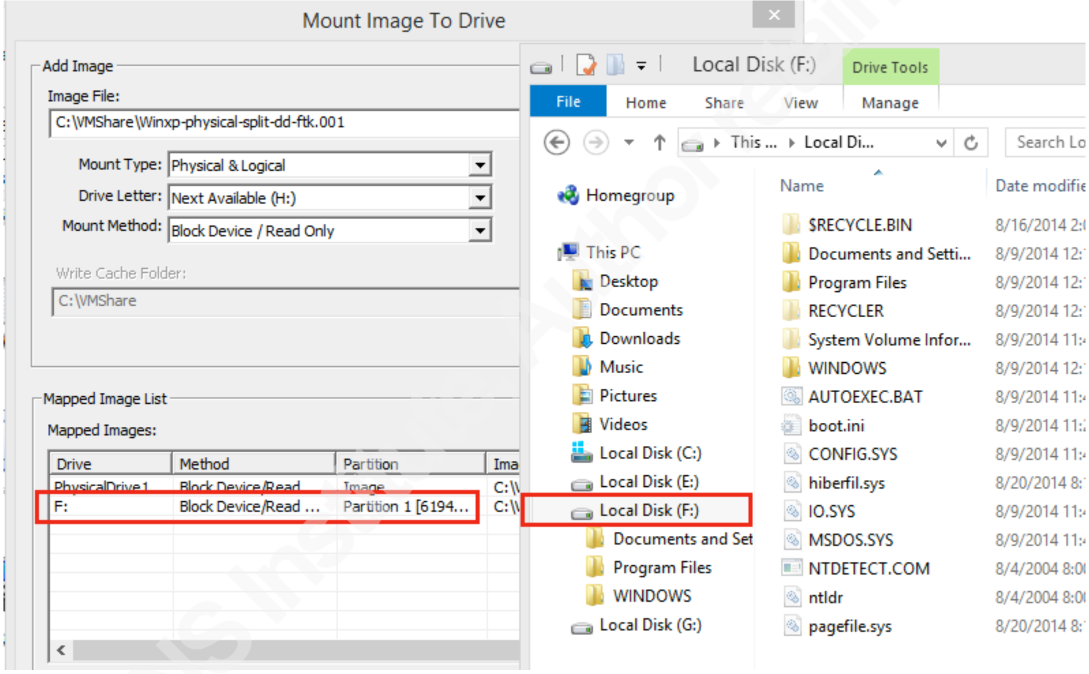
Support for E01 format

5.2.2. FTK Imager 3.2 with graphical user interface

The FTK Imager is a free utility from the Access Data group. In addition to creating disk images as described in Section 3.4, it will mount physical and logical disks and disk images. It supports mounting of raw images and E01 images but does not include support for the newer EnCase Ex01 image format. FTK Imager is a graphical tool that runs on Windows only. Included in the Imager is an image mounting utility. The graphical user interface contains a built-in help utility. FTK Image mounting utility will recognize and mount the following file systems: FAT 12, FAT16, FAT32, Ext2FS, Ext3FS, Ext4FS, ReiserFS3, exFAT, NTFS, HFS, HFS+, CDFS and VXFS according to the FTK Imager user guide.

Figure 12 shows the file system from an image mounted with the FTK image mounting utility. By default, the files are mounted with read-only access to protect them from being changed. The FTK mounting utility does provide the option to mount the files with write access.

Figure 12: FTK Image mounting utility mounts a logical volume that is accessible with the Windows file explorer.



Mount Image To Drive

Add Image

Image File: C:\VMShare\Winxp-physical-split-dd-ftk.001

Mount Type: Physical & Logical

Drive Letter: Next Available (H:)

Mount Method: Block Device / Read Only

Write Cache Folder: C:\VMShare

Mapped Image List

Mapped Images:

Drive	Method	Partition	Ima
PhysicalDrive1	Block Device/Read	Image	C:\
F:	Block Device/Read ...	Partition 1 [6194...	C:\

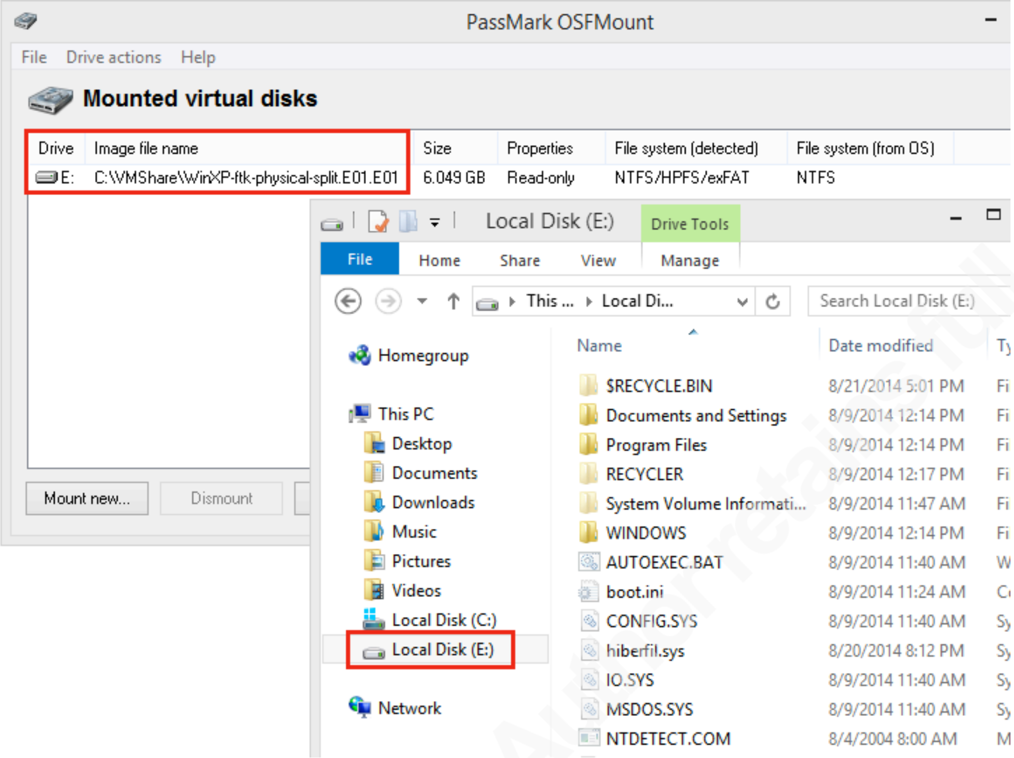
Local Disk (F:)

Name	Date modifie
$RECYCLE.BIN	8/16/2014 2:
Documents and Setti...	8/9/2014 12:
Program Files	8/9/2014 12:
RECYCLER	8/9/2014 12:
System Volume Infor...	8/9/2014 11:
WINDOWS	8/9/2014 12:
AUTOEXEC.BAT	8/9/2014 11:
boot.ini	8/9/2014 11:
CONFIG.SYS	8/9/2014 11:
hiberfil.sys	8/20/2014 8:
IO.SYS	8/9/2014 11:
MSDOS.SYS	8/9/2014 11:
NTDETECT.COM	8/4/2004 8:0
ntldr	8/4/2004 8:0
pagefile.sys	8/20/2014 8:

5.2.3. OSFMount version 1.5

PassMark Software has a free image mounting utility called OSFMount. It is a Windows utility with a graphical user interface (PassMark Software, n.d.). It can mount logical and physical disk images in the raw or E01 formats. It can also successfully mount split images in either format. Figure 13 shows the OSFMount graphical user interface along with a Windows Explorer window that has browsed to the mounted volume. OSFMount mounts images read-only by default but can be configured to allow write access. The built-in help utility can be accessed via the menu bar in the application.

Figure 13: OSFMount mounts a physical drive and attaches the logical volume to mount point E:



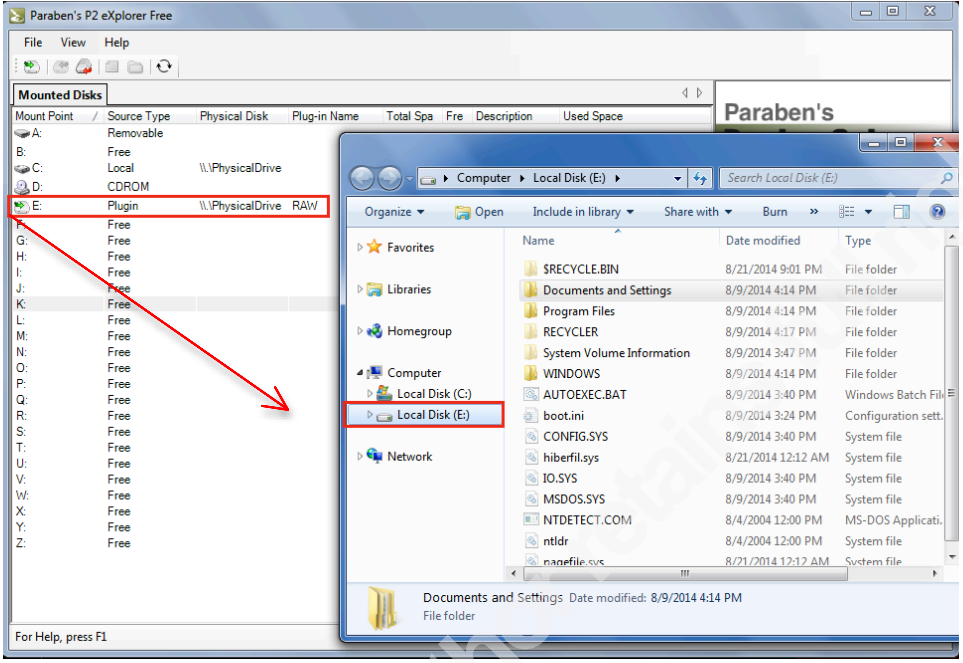
Drive	Image file name	Size	Properties	File system (detected)	File system (from OS)
E:	C:\VMShare\WinXP-ftk-physical-split.E01.E01	6.049 GB	Read-only	NTFS/HPFS/exFAT	NTFS

Name	Date modified	Ty
$RECYCLE.BIN	8/21/2014 5:01 PM	Fi
Documents and Settings	8/9/2014 12:14 PM	Fi
Program Files	8/9/2014 12:14 PM	Fi
RECYCLER	8/9/2014 12:17 PM	Fi
System Volume Informati...	8/9/2014 11:47 AM	Fi
WINDOWS	8/9/2014 12:14 PM	Fi
AUTOEXEC.BAT	8/9/2014 11:40 AM	W
boot.ini	8/9/2014 11:24 AM	C
CONFIG.SYS	8/9/2014 11:40 AM	Sy
hiberfil.sys	8/20/2014 8:12 PM	Sy
IO.SYS	8/9/2014 11:40 AM	Sy
MSDOS.SYS	8/9/2014 11:40 AM	Sy
NTDETECT.COM	8/4/2004 8:00 AM	M

5.2.4. P2-eXplorer Free version 4.0

The Paraben Corporation distributes a free version of its P2-Xplorer tool that will mount both raw and E01 images. It runs on 32 bit Windows systems only and provides a graphical interface. It is able to mount single or split images in either raw or E01 format. It will not, however, mount the newer Encase Ex01 format (Paraben Corp, n.d.). If the image file represents a physical disk P2-eXplorer can detect a logical volume within the image. The user is not required to provide an offset to the start of the file system. P2-eXplorer will attach the logical volume to the chosen mount point represented by a drive letter. Figure 14 shows a disk image mounted in P2-eXplorer along with the Windows file explorer view of the mounted file system. The utility provides built-in help.

Figure 14: A disk image, mounted using P2-eXplorer and viewed with File Explorer



Mount Point	Source Type	Physical Disk	Plug-in Name	Total Spa	Fre	Description	Used Space
A:	Removable						
B:	Free						
C:	Local	\\.\PhysicalDrive					
D:	CDROM						
E:	Plugin	\\.\PhysicalDrive	RAW				
F:	Free						
G:	Free						
H:	Free						
I:	Free						
J:	Free						
K:	Free						
L:	Free						
M:	Free						
N:	Free						
O:	Free						
P:	Free						
Q:	Free						
R:	Free						
S:	Free						
T:	Free						
U:	Free						
V:	Free						
W:	Free						
X:	Free						
Y:	Free						
Z:	Free						

Name	Date modified	Type
$RECYCLE.BIN	8/21/2014 9:01 PM	File folder
Documents and Settings	8/9/2014 4:14 PM	File folder
Program Files	8/9/2014 4:14 PM	File folder
RECYCLER	8/9/2014 4:17 PM	File folder
System Volume Information	8/9/2014 3:47 PM	File folder
WINDOWS	8/9/2014 4:14 PM	File folder
AUTOEXEC.BAT	8/9/2014 3:40 PM	Windows Batch Fil
boot.ini	8/9/2014 3:24 PM	Configuration sett.
CONFIG.SYS	8/9/2014 3:40 PM	System file
hiberfil.sys	8/21/2014 12:12 AM	System file
IO.SYS	8/9/2014 3:40 PM	System file
MSDOS.SYS	8/9/2014 3:40 PM	System file
NTDETECT.COM	8/4/2004 12:00 PM	MS-DOS Applicati.
ntldr	8/4/2004 12:00 PM	System file
pagefile.sys	8/21/2014 12:12 AM	System file

Documents and Settings Date modified: 8/9/2014 4:14 PM
File folder

5.3. Linux Utilities to Mount a Forensic Image

5.3.1. Ewfmount

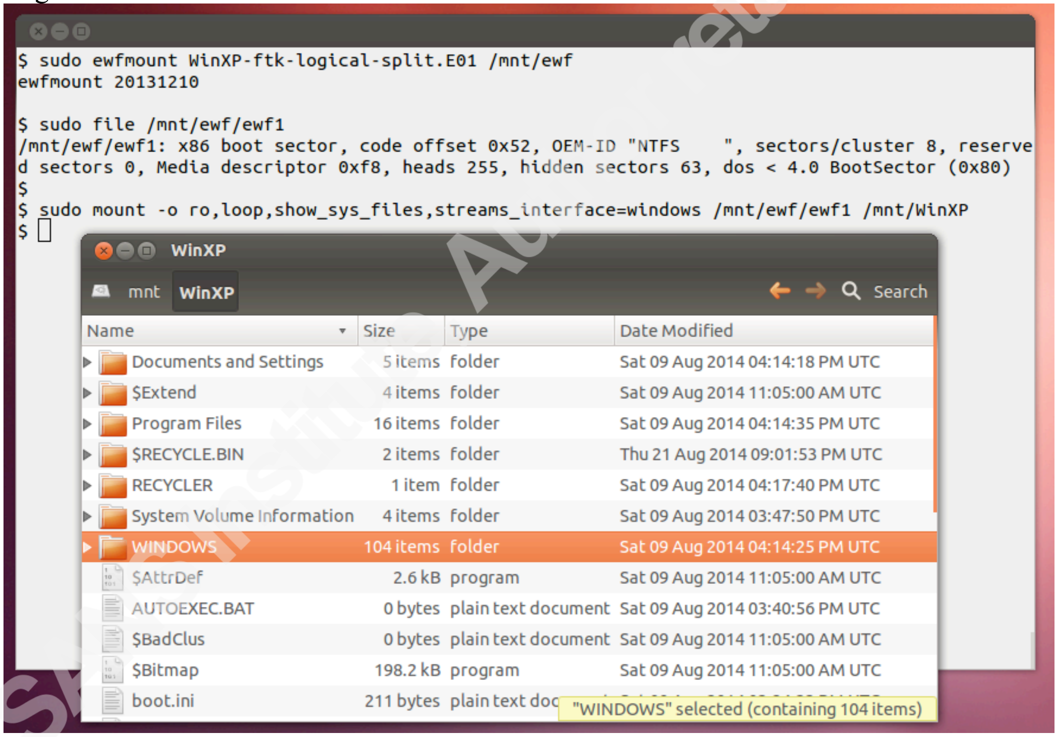
The Libewf library introduced in Section 4.12 also contains the image mounting utility *ewfmount*. It is a command line utility that runs on Linux and OSX and takes as input E01 files only. Libewf tools can also run on Windows operating systems if built using Cygwin[7]. *Ewfmount* does not yet have full support for the newer EnCase format, Ex01.

As detailed in Section 3.2.2 E01 files are embedded images that contain metadata and may also be compressed. *Ewfmount* is able to create a virtual file system by performing a preliminary mount of an E01 embedded image file or file set. This preliminary mount provides a view of the embedded image after removing the metadata and combining the file set so that it appears as a single raw image. A secondary mount using the Linux mount command is then able to assign a file system mount point to the virtual image created by *ewfmount* so that the contents of the partition can be accessed normally.

[7] Cygwin is a utility that allows a user to port a Linux application to Windows and can be downloaded from https://www.cygwin.com/

Figure 15 shows how an E01 file set is mounted using the *ewfmount* and *mount* commands. First, the *ewfmount* command attaches the uncompressed, combined file set to mount point /mnt/ewf. The Linux *file* utility recognizes the file "ewf1" as a partition containing an NTFS formatted file system. The Linux *mount* command attaches the NTFS partition to mount point /mnt/WinXP. The options used in the *mount* command serve to mount the image read only (ro), treat the file like a ***block device*** (loop), show otherwise hidden system files (show_sys_files), and show all data streams associated with each file (streams_interface=windows).

Figure 15: Use the *ewfmount* command to prepare an E01 file set for attaching to a mount point with regular *mount* command



```
$ sudo ewfmount WinXP-ftk-logical-split.E01 /mnt/ewf
ewfmount 20131210

$ sudo file /mnt/ewf/ewf1
/mnt/ewf/ewf1: x86 boot sector, code offset 0x52, OEM-ID "NTFS    ", sectors/cluster 8, reserve
d sectors 0, Media descriptor 0xf8, heads 255, hidden sectors 63, dos < 4.0 BootSector (0x80)
$
$ sudo mount -o ro,loop,show_sys_files,streams_interface=windows /mnt/ewf/ewf1 /mnt/WinXP
$ 
```

Name	Size	Type	Date Modified
Documents and Settings	5 items	folder	Sat 09 Aug 2014 04:14:18 PM UTC
$Extend	4 items	folder	Sat 09 Aug 2014 11:05:00 AM UTC
Program Files	16 items	folder	Sat 09 Aug 2014 04:14:35 PM UTC
$RECYCLE.BIN	2 items	folder	Thu 21 Aug 2014 09:01:53 PM UTC
RECYCLER	1 item	folder	Sat 09 Aug 2014 04:17:40 PM UTC
System Volume Information	4 items	folder	Sat 09 Aug 2014 03:47:50 PM UTC
WINDOWS	104 items	folder	Sat 09 Aug 2014 04:14:25 PM UTC
$AttrDef	2.6 kB	program	Sat 09 Aug 2014 11:05:00 AM UTC
AUTOEXEC.BAT	0 bytes	plain text document	Sat 09 Aug 2014 03:40:56 PM UTC
$BadClus	0 bytes	plain text document	Sat 09 Aug 2014 11:05:00 AM UTC
$Bitmap	198.2 kB	program	Sat 09 Aug 2014 11:05:00 AM UTC
boot.ini	211 bytes	plain text doc	

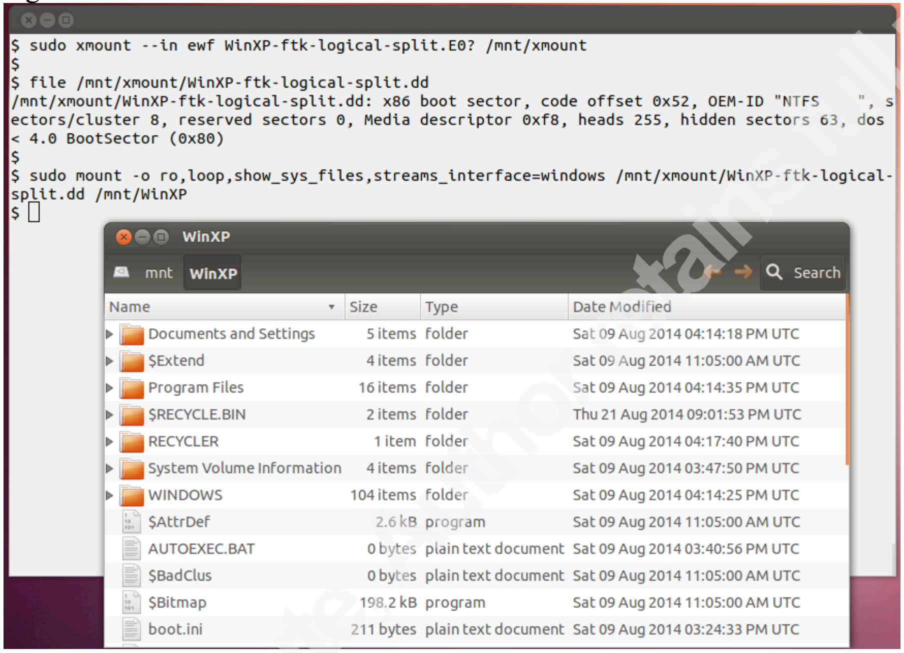
"WINDOWS" selected (containing 104 items)

The files in the NTFS partition can be viewed either with a graphical file explorer or using command line tools. The *ewfmount* command expects all of the files in the E01 file set to be located in the same directory. Only the first file in the set, E01, is passed as an argument to *ewfmount*.

5.3.2. Xmount

Xmount is another command line utility that runs on Linux and OS X (Gillen, n.d.). It functions like *ewfmount* by creating a virtual file system using an image file set that can be attached to a mount point using the *mount* command. *Xmount* can take image files in either the raw or E01 format as input[8]. Figure 16 illustrates how to use the *xmount* command to mount an E01 file set.

Figure 16: Use the *xmount* command to prepare a raw or E01 file set for attaching to a mount point with regular *mount* command



```
$ sudo xmount --in ewf WinXP-ftk-logical-split.E0? /mnt/xmount
$
$ file /mnt/xmount/WinXP-ftk-logical-split.dd
/mnt/xmount/WinXP-ftk-logical-split.dd: x86 boot sector, code offset 0x52, OEM-ID "NTFS    ", s
ectors/cluster 8, reserved sectors 0, Media descriptor 0xf8, heads 255, hidden sectors 63, dos
< 4.0 BootSector (0x80)
$
$ sudo mount -o ro,loop,show_sys_files,streams_interface=windows /mnt/xmount/WinXP-ftk-logical-
split.dd /mnt/WinXP
$ 
```

Name	Size	Type	Date Modified
Documents and Settings	5 items	folder	Sat 09 Aug 2014 04:14:18 PM UTC
$Extend	4 items	folder	Sat 09 Aug 2014 11:05:00 AM UTC
Program Files	16 items	folder	Sat 09 Aug 2014 04:14:35 PM UTC
$RECYCLE.BIN	2 items	folder	Thu 21 Aug 2014 09:01:53 PM UTC
RECYCLER	1 item	folder	Sat 09 Aug 2014 04:17:40 PM UTC
System Volume Information	4 items	folder	Sat 09 Aug 2014 03:47:50 PM UTC
WINDOWS	104 items	folder	Sat 09 Aug 2014 04:14:25 PM UTC
$AttrDef	2.6 kB	program	Sat 09 Aug 2014 11:05:00 AM UTC
AUTOEXEC.BAT	0 bytes	plain text document	Sat 09 Aug 2014 03:40:56 PM UTC
$BadClus	0 bytes	plain text document	Sat 09 Aug 2014 11:05:00 AM UTC
$Bitmap	198.2 kB	program	Sat 09 Aug 2014 11:05:00 AM UTC
boot.ini	211 bytes	plain text document	Sat 09 Aug 2014 03:24:33 PM UTC

The parameter "- -in ewf" is used to specify the input file format. When mounting a raw formatted file set use the parameter "- -in dd". *Xmount* must be given all files in the split image set. This can be done with wildcard characters as shown is Figure 16 or by specifying each file separated by spaces. The last argument passed to *xmount* should be the desired mount point.

Xmount mounts the virtual file system read only by default. *xmount* help is available using any of the following:

```
$ man  xmount   or   $ info xmount    or    $ xmount  -h
```

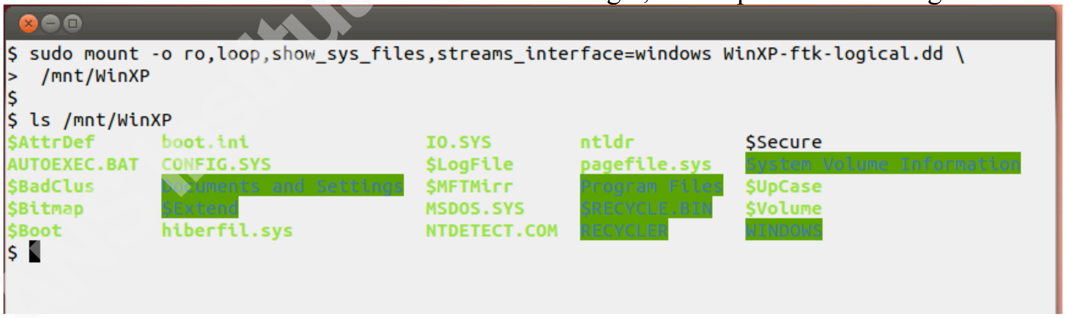
[8] Xmount version 0.6.0 is required to mount split raw image files. As of 9/1/2014 the SIFT 3.0 Workstation runs xmount version 0.4.5.

5.3.3. Mount

The Linux *mount* command can be used alone to mount a single image or if used together with the affuse utility can mount split images. The Linux or OS X *mount* command can mount a single uncompressed raw image files without the need for initially creating a virtual file system. This is because a single raw image file contains no metadata. Instead it contains a complete exact bit-for-bit copy of a file system partition. To access the file system contained in the raw image it must first be attached to a mount point using the *mount* command.

Figure 17 illustrates how the Linux *mount* command can be used to mount a single, uncompressed raw image of a logical volume. The logical volume contains only a formatted partition. In contrast, an image file created from an entire physical disk will also contain a partition table describing the partitions currently formatted on the disk. When mounting a partition from an image of a physical disk it is necessary to give an additional parameter to the *mount* command.

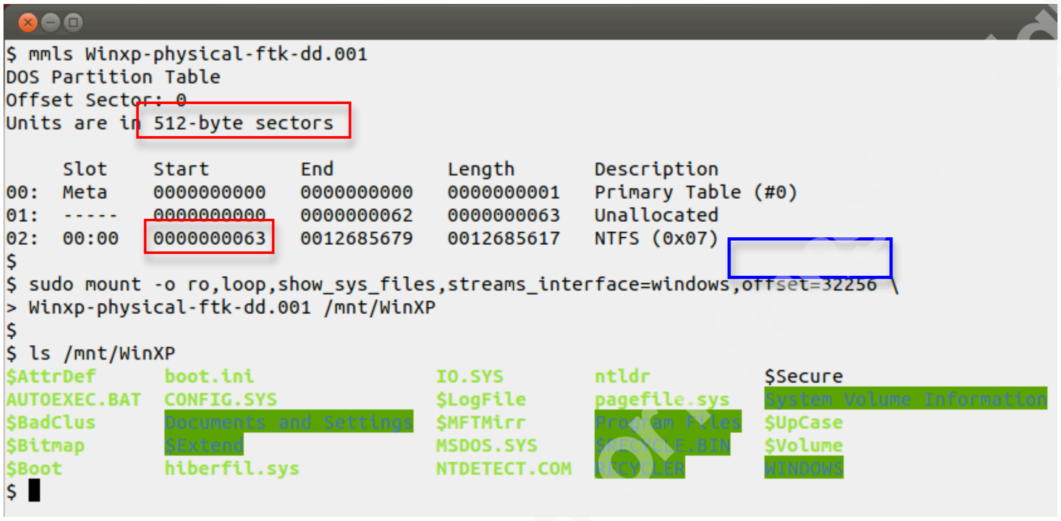
Figure 17: The Linux mount command can mount a single, uncompressed raw image



```
$ sudo mount -o ro,loop,show_sys_files,streams_interface=windows WinXP-ftk-logical.dd \
>  /mnt/WinXP
$
$ ls /mnt/WinXP
$AttrDef      boot.ini                IO.SYS        ntldr          $Secure
AUTOEXEC.BAT  CONFIG.SYS              $LogFile      pagefile.sys   System Volume Information
$BadClus      Documents and Settings  $MFTMirr      Program Files  $UpCase
$Bitmap       $Extend                 MSDOS.SYS     $RECYCLE.BIN   $Volume
$Boot         hiberfil.sys            NTDETECT.COM  RECYCLER       WINDOWS
$ 
```

Figure 18 illustrates how to use the “offset” parameter to direct the *mount* command to the starting location of the desired partition within the image. To determine the offset value required, use the Sleuthkit utility *mmls*. The Sleuthkit is a collection of command line tools directly related to examining disk images (Carrier, n.d.). The *mmls* utility gives a “media management listing” or partition table listing of an image (Carrier, 2005, Ch. 4). The NTFS formatted partition begins at the 63rd sector on the disk. *Mmls* displays the sector size as 512 bytes. Therefore, the offset value to pass to the *mount* command is 512 x 63, or 32256 bytes.

Figure 18: Use the offset parameter to specify where the file system begins on the disk

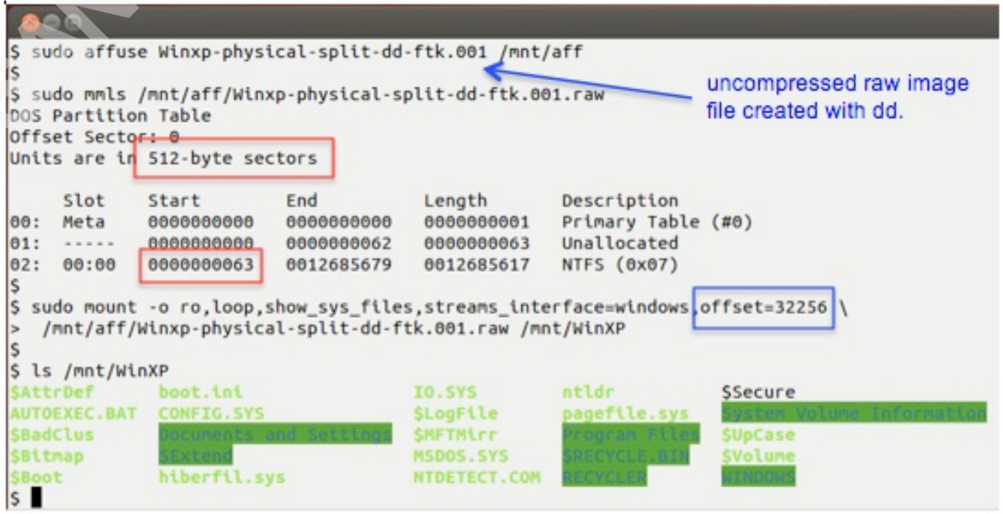


```
$ mmls Winxp-physical-ftk-dd.001
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot     Start        End          Length       Description
00:  Meta     0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----    0000000000   0000000062   0000000063   Unallocated
02:  00:00    0000000063   0012685679   0012685617   NTFS (0x07)
$
$ sudo mount -o ro,loop,show_sys_files,streams_interface=windows,offset=32256 \
> Winxp-physical-ftk-dd.001 /mnt/WinXP
$
$ ls /mnt/WinXP
$AttrDef      boot.ini                IO.SYS        ntldr          $Secure
AUTOEXEC.BAT  CONFIG.SYS              $LogFile      pagefile.sys   System Volume Information
$BadClus      Documents and Settings  $MFTMirr      Program Files  $UpCase
$Bitmap       $Extend                 MSDOS.SYS     $RECYCLE.BIN   $Volume
$Boot         hiberfil.sys            NTDETECT.COM  RECYCLER       WINDOWS
$ 
```

Figure 19 illustrates how to use the *affuse* command together with the *mount* command to mount a split raw image file set. *Affuse* is a utility that runs on Linux or OS X and is part of a library of disk utilities by Simson Garfinkel (Garfinkel, 2013). *Ewfmount* and *xmount* can create a virtual file system from E01 split images. Both *xmount* and *affuse* are able to create a virtual file system from raw split images. The Linux *mount* command attaches the virtual file system to the mount point "/mnt/WinXP" as shown in Figure 19.

Figure 19: Affuse utility will create a virtual file system of split raw images



```
$ sudo affuse Winxp-physical-split-dd-ftk.001 /mnt/aff
$
$ sudo mmls /mnt/aff/Winxp-physical-split-dd-ftk.001.raw
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot     Start        End          Length       Description
00:  Meta     0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----    0000000000   0000000062   0000000063   Unallocated
02:  00:00    0000000063   0012685679   0012685617   NTFS (0x07)
$
$ sudo mount -o ro,loop,show_sys_files,streams_interface=windows,offset=32256 \
>  /mnt/aff/Winxp-physical-split-dd-ftk.001.raw /mnt/WinXP
$
$ ls /mnt/WinXP
$AttrDef      boot.ini                IO.SYS        ntldr          $Secure
AUTOEXEC.BAT  CONFIG.SYS              $LogFile      pagefile.sys   System Volume Information
$BadClus      Documents and Settings  $MFTMirr      Program Files  $UpCase
$Bitmap       $Extend                 MSDOS.SYS     $RECYCLE.BIN   $Volume
$Boot         hiberfil.sys            NTDETECT.COM  RECYCLER       WINDOWS
$ 
```

uncompressed raw image file created with dd.

6. Conclusion

This paper included discussion of what a forensic image is and why it is useful to forensic analysts. It described common tools to create forensic images, as well as, common tools to access the images either by viewing the image file directly or by performing a file system mount to access the files. The method chosen depends on the target data. When using analysis applications or utilities that interface with operating system API's, a forensic image can be attached to a mount point and accessed "normally" as a file system. For example, if an investigator would like to anti-virus tools against the files from a Windows image, she could mount the image and point the anti-virus tools to the files to be scanned using Windows File Explorer.

When an investigator needs to see all data in the image or data that is normally not accessible using native file viewing applications, he can examine data within the image file, without mounting the file system so that every bit from the disk is accessible, including data that has been marked as unallocated by the file system. The SANS SIFT Workstation is rich with tools to accomplish this type of access.

References

Access Data Group. (n.d.). Computer forensics software for digital investigations. Retrieved from http://www.accessdata.com/solutions/digital-forensics/ftk

Arsenal Recon. (2014). Arsenal image mounter. Retrieved from http://arsenalrecon.com/apps/image-mounter/

Brown, C. L. T. (2010). *Computer evidence: collection and preservation.* 2nd ed. Boston, MA: Charles River Media/Cengage Learning.

Bunting, S. (2008). *EnCE: The official EnCase certified examiner study guide* (2nd ed.). Indianapolis, Indiana: Wiley Publishing, Inc.

Bunting, S. (2012). *EnCase computer forensics: the official EnCE : EnCase certified examiner study guide* (3rd ed.). Indianapolis, Ind.: Wiley.

Carrier, B. (2005). *File system forensic analysis.* Boston, Mass.: Addison-Wesley.

Carrier, B. (n.d.). The Sleuth Kit. Retrieved from http://www.sleuthkit.org/sleuthkit/index.php

Chessman, S. (1996, December 1). *dd*. Retrieved August 5, 2014, from http://www.linuxjournal.com/article/1320

Forensics file formats - forensicsWiki. (2012, July 21). Retrieved August 7, 2014, from http://www.forensicswiki.org/wiki/Category:Forensics_File_Formats

Garfinkel, S. (2013, November 16). AFFLIBv3. Retrieved from https://github.com/simsong/AFFLIBv3

Gillen, D. (n.d.). Xmount. Retrieved from https://www.pinguin.lu/xmount

Kumar, S., & Vijayaraghavan, R. (2011, October). Solid state drive (SSD) FAQ. Retrieved from http://www.dell.com/downloads/global/products/pvaul/en/Solid-State-Drive-FAQ-us.pdf

Levy, J. (2014, April 23). *EWFAddressSpace*. Retrieved August 20, 2014, from https://code.google.com/p/volatility/wiki/EWFAddressSpace

Metz, J. (n.d.). *Libewf and tooling to access the Expert Witness compression format (EWF)*. Retrieved August 9, 2014, from https://code.google.com/p/libewf/

NIST. (2001, November 26). *Announcing the advanced encryption standard*. Retrieved August 17, 2014, from http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

Paraben Corp. (n.d.). *Paraben - P2 eXplorer Pro*. Retrieved from https://www.paraben.com/p2-explorer.html

PassMark Software. (n.d.). *OSFMount - Mount CD and Ddisk images in Windows*. Retrieved from http://www.osforensics.com/tools/mount-disk-images.html

Prosise, C., Mandia, K. (2003). *Incident response and computer forensics.* 2nd ed. New York, New York: McGraw-Hill/Osborne.

SANS. (n.d.). SANS SIFT Kit/Workstation: Investigative forensic toolkit download. Retrieved from http://digital-forensics.sans.org/community/downloads

SWGDE. (2013, September 14). *SWGDE Best practice for computer forensics*. Retrieved August 13, 2014, from https://swgde.org/documents/Current%20Documents/2014-09-05%20SWGDE%20Best%20Practices%20for%20Computer%20Forensics%20V3-1

Weibe, J. (2013, May 28). *Forensic insight into solid state drives*. Retrieved August 8, 2014, from http://www.dfinews.com/articles/2013/05/forensic-insight-solid-state-drives

Appendix A
Glossary

Acquisition hash - The term used by EnCase for the cryptographic hash computed for an EnCase evidence image.

Bit-for-bit copy - An exact replica of the bits from either a logical volume or a physical drive. When the copy is made to a file it is called a forensic image file. When the copy is made to another disk it is called a clone.

Bit-stream image - Same as bit-for-bit copy.

Bitlocker - Full disk encryption solution from Microsoft. Encrypts Logical Volumes.

Block device – a device or file that allows data to be written to or read from in blocks of data at random locations. This is in contrast to a character device like a magnetic tape where data can only be accessed sequentially.

Clone An exact replica or bit-for-bit copy of one disk on to another disk.

Cryptographic hash - A cryptographic hash refers to a mathematical function that takes any amount of data as input and performs a mathematical calculation over that data resulting in a fixed length output. In digital forensics, the hash functions used are usually MD5 or SHA-1 and the output is a 128 or 256 bit value respectively. The output, often called a *message digest*, represents a unique identifier for the input. A cryptographic hash function is considered irreversible allowing the message digest to act like an identifier or *fingerprint* of the input data. When a copy is made of a forensic image, the message digests from each can be compared to determine if the files are exact duplicates. If the image files are exactly the same, their cryptographic hashes will also be identical.

Cyclic Redundancy Check (CRC) - A CRC is an error checking mechanism often used to compare two sets of data in order to detect errors during transmission. It is less compute intensive than MD5 or SHA-1 hash algorithms but still provides a statistically sufficient validation that no transmission errors have occurred when the CRC of the source data block matches the CRC of the destination data block.

Dead imaging – When a disk is removed from a computer system and reattached via a write-blocker to another system that can acquire the contents of the disk it is called dead imaging. It is considered "dead" because even though power is applied to the disk, the write-blocker will prevent any data on the disk from being changed. In addition, image acquisition software accessing a disk attached in this way is able to access all space on the drive including unallocated space. This provides a forensically sound method of disk acquisition.

Disk Image File - A file containing an exact copy of a physical disk or logical volume. Tools like dd, dcfldd and FTK Imager can create disk image files. ISO images are disk images of data CDs or DVDs.

Embedded image - disk images that contain metadata about the image such as a timestamp when the image was created and a ***cryptographic hash***. E01 (Expert Witness) images are embedded images that contain an acquisition hash, CRC calculations after each data block and case information.

File system - A file system consists of files and the metadata used to describe the organization of the files as well as the data in the files. Metadata may include items like filename, directory structure, timestamps, length of file, owner ID, permissions and other attributes. Common file systems are ReFS, NTFS, FAT32, exFAT, HFS+, ext2, ext3.

Fingerprint - An identifier for a particular forensic image. See *cryptographic hash*.

Forensic duplicate - Same as Forensic image.

Forensic image - A copy of a logical volume or physical disk that has been copied bit-for-bit so that it includes all data and metadata. A forensic image will normally include metadata (often in a separate file) about the image such as when it was acquired, with which tool and the cryptographic hash used for verification of the image.

Formatting - Formatting a hard disk means preparing it for a particular file system. A hard drive must first be partitioned before it can be formatted.

Image File - An image file is a

ISO image - ISO images are exact copies or images of data CDs or data DVDs. ISO is short for ISO 9660, which is a file system standard used by the ISO organization, an internationally recognized standards organization.

Live imaging – When a disk cannot be removed from a running system and its contents acquired during normal operation, the contents of the disk may change during or after the live-imaging process because the disk will not likely be mounted read-only. If live imaging takes place on a Windows system, the acquired image will not contain unallocated space. Live imaging performed from a Linux system does allow access to unallocated space.

Logical volume - A partition that has been formatted with a file system makes up a logical volume.

MBR - Master Boot Record. The first sector of a physical disk contains the MBR. The MBR identifies up to four partitions on a disk by giving the starting location, length of partition and the file system type it contains.

Memory Image File - A file containing an exact copy of a computer's physical memory. Since the contents of memory are volatile and not static, a memory image is like a snapshot from a specific point in time.

Metadata- Metadata is data describing other data. For example, the metadata in a Microsoft WORD document will contain the filename, the size of the file, permissions associated with the file etc. This is not data that is visible within the document. It is data about the document.

Mirror image - Same as clone.

Mount Point - A directory on a file system to which another file system can be attached for accessing. It defines the "door" to that file system. For example, on a Windows system the mount point for the primary file system that stores the operating system is typically C:\ so that files on that disk can be accessed with a path that starts at the mount point C:\. When a USB drive is plugged in it is normally mounted using the next available mount point D:\ or E:\. A mount point can be any empty directory in both Windows and Linux operating systems.

Mounting an image - A file system within an image file can only be viewed if the image file is made accessible by logically attaching it or *mounting* it to a ***mount point*** on a computer system. If the image represents a logical volume the process of mounting the image is straightforward. On Linux systems, the mount command will attach or associate the logical volume with a mount point. There are free applications available on Windows systems to mount images of logical volumes. When the image represents an entire physical disk then the image also contains metadata at the beginning of the disk, like the MBR. In this case, mounting the image requires an additional step in order to determine where the logical volume beings.

Partition - A section of a physical disk that can be formatted with a file system.

Physical disk - Physical disk is another name for hard disk or hard disk drive (HDD) that stores digital data. Physical disks are normally partitioned and then formatted with file systems in order to store data.

Program and erase cycle – Also called a p/e cycle. It is a technology used in solid-state disks to write data. Before a data can be written or "programmed" to the disk it must be erased. Each erase and write operation is called a p/e cycle. All solid-state drives become unusable after some number of p/e cycles. Because of the wear from p/e cycles, a technology called wear leveling (see wear-leveling entry in this glossary) is used to spread the p/e cycles more evenly over the drive resulting in an overall increased lifetime.

Raw device - In Unix and Linux operating systems, disk drives can be accessed via a software construct called a raw device. This allows direct access to the disk drive and bypasses any operating prescribed caching or buffering. Raw devices are useful in forensics because it allows precise amounts of data to be read and written as opposed to only chunks of a specific size.

Raw image - A raw image is an exact bit-for-bit copy of a disk into a file. A raw image does not contain metadata and is not compressed. Some tools that create raw images will create a separate file containing metadata about the image like when it was acquired, the name of the tool that generated the image file and the cryptographic hash used to fingerprint the image for later verification.

Slack space - explain in detail including a graphic

Split images – forensic images that are split into multiple pieces of a predetermined size. This provides flexibility with storage of forensic images as disk drive capacity grows.

Unallocated space - explain in detail including a graphic

Wear leveling - The microchips that store data in solid-state storage devices have a limited number of writes that can occur before the chips will begin to falter and provide unreliable results. In order to maximize the reliable lifetime of the storage a process called wear leveling is used to evenly spread out the data across the drive. This is accomplished by moving data around continuously to prevent excessive operations to the same location on the disk. See also ***program and erase cycle***.

Write-blocker – A write-blocker is used to connect a disk drive to a computer system for the purpose of imaging the disk while assuring that all write operations to the disk are blocked. During a forensic investigation the disk is considered evidence so whenever possible changes to the evidence are prevented. Specific directions for using a write-blocker are manufacturer specific.