Digital Still Camera Forensics

Kevin Cohen

***Abstract*—In many instances computer forensic practitioners come across digital photographs within an investigation. Examples include law enforcement examining computers for child pornography, intellectual property disputes involving proprietary digital images, or file recovery for individuals who have lost personal photos due to accidental deletion or electronic media corruption. More technical examples include locating steganographic images and digital rights management.**

Photographic images residing on electronic media most likely originated from digital still cameras ("DSC"). To identify the origin of a DSC image, a forensic examiner should understand the characteristics of a DSC image.

This paper describes some of the characteristics associated with DSC images. Knowledge allows a forensic practitioner to isolate specific attributes within a DSC image to identify exact replicas, derivates, or additional images that fall within a particular group set.

This paper further outlines some open source software that can be used to extract images based upon the characteristics of a DSC image.

***Index Terms*—Digital Still Camera, JPEG Metadata, JFIF Metadata, DCF Metadata, Exif Metadata, Linux Forensics, Recover pictures, Recover digital photos, Digital photo forensics, DSC, Jpeg header, Exif header, USB, DSC images**

I. A Brief History of the Digital Still Image

THE file extension .jpg is most commonly referred to as the JPEG file format - "a lossy[1] compression method that takes advantage of the limitation of the human eye." (ITU-T T81)

In 1983, the International Organization for Standardization ("ISO") started to look into methods for adding high resolution graphics and pictures onto computers (at that time text based terminals). Three years later, the Joint Photographic Experts Group ("JPEG") was formed by the International Telegraph and Telephone Consultative Committee ("CCITT") and ISO/IEC to develop a standard procedure for encoding grayscale and color images. JPEG published "Information Technology - Digital Compression and Coding of Continuous-tone Still images - Requirements and guidelines," otherwise known as the JPEG specifications for encoding and decoding compressed images, which is referenced as ISO 10918-1 or ITU-T T.81 (ITU stands for International Telecommunication Union, which is an agency within the United Nations. CCITT is a part of the ITU).

In 1992, Eric Hamilton wrote "JPEG File Interchange Format" ("JFIF") which calls for some metadata information regarding the JPEG images to be shared outside of the entropy encoded data segments. This additional information allows JPEG images to be shared and recognized amongst foreign applications.

In 1996, "Digital Still Camera Image File Format Standard" (Exif: Exchangeable image file format) was published, which is referenced as EXIF Format ISO 12234-1 [2] by Japan Electronic Industry Development Association ("JEIDA")

Also in 1996, a competing file format standard was published by Eastman Kodak, "FlashPix Format Specification." Though FlashPix was adopted by Eastman Kodak in collaboration with Hewlett-Packard, Microsoft, and Live Picture, it did not gain wide acceptance with some of the other manufacturers of DSC's.

In 1997, Canon published "CIFF Specification on Image Data File" to specify its camera image file format ("CIFF"). CIFF was intended to become a standard file system to be adopted for removable media used to store images by DSCs, such as SmartMedia, Compact Flash Disks, Memory Sticks, or San Disks.

In 1998, JEIDA wrote "Design rule for Camera File system" ("DCF"), a FAT based file system which is now the generally accepted format and replaced the CIFF architecture.

In 2002, Japan Electronics and Information Technology Industries Association ("JEITA"), formerly JEIDA, published "Exchangeable image file format for digital still cameras: Exif Version 2.2."

Prior to the development of inexpensive DSCs, scanners were used on a much larger scale. In 1986, the Aldus Corporation wrote the TIFF specification. TIFF is the standard for scanning an image without compression. Much of the Exif metadata comes from the document "TIFF Revision 6.0" published by Aldus in 1992. There are other digital image standards such as JPEG2000, JPEG-LS, and SPIFF; however, for the purpose of this paper, we will not be describing them.

As of the writing of this article, Exif is the most current standard for DSCs. Exif incorporates the best features of JPEG, JFIF, TIFF, and DCF while allowing for backwards compatibility of FlashPix, CIFF, and other proprietary formats.

A. Analysis of metadata in a .JPG Image

The term metadata refers to data about data. The most common interpretation of metadata is the information found within the directory entries of a FAT file system that describes the cluster chains associated with file data, or more recently the Master File Table ("MFT") Entry describing a data portion of a file within an NTFS partition. The MFT Entry describes the creation time, modified time, accessed time, written time, file name, file extension, file size, location of file data, owner, file permissions, and other attributes associated with the file data.

[1]Lossy - The opposite of lossy is lossless. The term lossy is used to express slight degradation from the original image. Since the human eye does not notice sharp changes in brightness of color, lossy compression can alter images with little or no visual effect.

Hex	Symbol	Marker Name	Description
FFD8	SOI	Start of Image	Start of compressed data
FFE1	APP1	Application Segment 1	Exif attribute information
FFE2	APP2	Application Segment 2	Exif extended data
FFDB	DQT	Define Quantization Table	Quantization table definition
FFC4	DHT	Define Huffman Table	Huffman table definition
FFDD	DRI	Define Restart Interoperability	Restart Interoperability definition
FFC0	SOF	Start of Frame	Parameters relating to frame
FFDA	SOS	Start of Scan	Parameters relating to components
FFD9	EOI	End of Image	End of compressed data

TABLE I
JPEG MARKER CODE ASSIGNMENTS USED IN EXIF (JEITA CP-3451)

Originally JPEG was developed for the compression techniques used to allow compression of 10:1 to 20:1 without visible loss. The entropy encoded compression and decompression was to be performed within a single application and thus no extra information about the data was necessary.

For extensibility, Exif includes many additional attributes that describe the entropy encoded data within the data portion of the file. Information found within the data portion of an Exif file includes date and Time picture was taken, make & model of camera, artist or owner of camera, F-stop, aperture and ISO settings, as well as many other camera settings including GPS information where the picture was taken.

B. JPEG Metadata

JPEG uses either Huffman coding or arithmetic coding compression to create a lossy representation of an image. The encoded data is separated into entropy encoded segments of the JPEG file. These entropy segments are separated by two byte markers.

A marker consists of two bytes, with the first byte being the hexadecimal value FF. Table 1 is a list of marker code assignments that can be used to identify segments within DSC images that originated from the JPEG image standard. Some markers to become familiar with are the start of image ("SOI") and end of image ("EOI").

Hex value **FF D8** represents the SOI of a JPEG image.
Hex value **FF D9** represents the EOI of a JPEG image.

If a file conforms to the JPEG standards, it will contain a header signature that starts with the SOI marker FFD8 and ends with a footer of the EOI marker FFD9.

Entropy encoded data segments sometimes have compression values that equate to FF. In such an instance, to distinguish the encoded data from a segment marker, the FF byte will be followed by another byte equal to 0 or hexadecimal representation of 00. Thus, if you see the two byte hexadecimal value FF 00 within a JPEG image, you should realize it is part of an entropy encoded segment and not a marker.

Thumbnails within Exif files may use a JFIF header and footer.

C. JFIF Metadata

JPEG File Interchange Format adds some additional information to the JPEG image which enables foreign applications to read and view the contents of a compressed JPEG image.

JFIF appends an APP0 segment directly after the SOI. The APP0 segment provides the information relating to version number, x and y pixel density, pixel aspect ratio, and an optional thumbnail.

If a file conforms to the JFIF standards, then it will contain a header signature that starts with the FF D8 FF E0 xx xx 4A 46 49 46 00 and ends with a footer of FF D9.

Hexadecimal representation ASCII representation
FF D8 FF E0 xx xx 4A 46 49 46 00 **ÿØÿá..JFIF**.
FF D8 represents the SOI
FF EO represents the APP0
xx xx is an unknown two bytes representing the length of the APP0 segment
4A 46 49 46 00 represents a signature JFIF terminated by 00

D. DCF Metadata

Design rule for Camera File system ("DCF") is a FAT 16 based file system. The metadata associated with DCF can be found within the data entry portion, specifically the naming convention.

DCF uses only upper case alpha numeric characters and the underscore symbol (A-Z, 0-9, _).

DCIM (DCF Camera Images) is the DCF image root directory.

Directories within the root directory consist of eight characters. The first three characters are digits equaling 100 or higher. The last five characters are upper case alpha characters. An example of a DCF compliant directory name would be 101ABCDE.

File names consist of 8 characters. The first four characters are alpha (including the _), and the last four are numerical 0001 or greater. An example of a DCF compliant file name is ABCD0001.JPG.

In Windows this particular file might look like:
D:\DCMI\101ABCDE\ABCD0001.JPG

E. Exif Metadata

Exchangeable image file format has APP1 and APP2 application segments and uses some metadata tags associated with the TIFF - 6.0 tags.

The Image File Directory ("IFD") is where the metadata is stored within Exif file. IFD is split into three sections, Exif IFD, GPS IFD, and Interoperability IFD.

The 0th IFD has tags associated with the primary image. The 1th IFD has tags associated with the thumbnail.

Exif files contain distinctive headers that start with the SOI and are followed by APP1. If a file conforms to the Exif standards, it will contain a header signature that starts with the FF D8 followed by APP1 FF E1 xx xx 45 78 69 66 00 and ends with a footer of FF D9.

TABLE II
METADATA TAGS USED IN EXIF (JEITA CP-3451)

Interoperability IFD
Attached Information Relating to Interoperability
InteroperabilityIndex

Hexadecimal representation ASCII representation
FF D8 FF E1 xx xx 45 78 69 66 00 **ÿØÿá..Exif**.
FF D8 represents the SOI
FF E1 represents the APP1
xx xx is an unknown two bytes representing the length of the APP1 segment
45 78 69 66 00 represents a signature JFIF terminated by 00

APP1 appears at the beginning of an Exif file. APP1 contains the thumbnail and cannot be more that 64 Kb in size.

Exif files allow for an optional APP2 which contains FlashPix extensions and has a signature FF E2 xx xx 45 78 69 66 00

Hexadecimal representation ASCII representation
FF E2 xx xx 45 78 69 66 00 **ÿØÿâ..FPXR**.

II. FORENSIC ANALYSIS ON DIGITAL STILL CAMERAS

A. Preservation

Prior to analysis, an examiner should understand that overwriting data could corrupt images contained within a device. An examiner should adhere the same level of precaution as if the data were part a criminal investigation. Meaning, it is the investigator's responsibility to preserve the evidence and maintain a chain of custody. To accomplish this task, the investigator should obtain write protection, bit by bit imaging software, and a method of demonstrating that neither the original data nor its copy was altered or overwritten during the copy process. To achieve these goals, this paper will focus on some open source Linux tools; however, many other commercial or non-commercial products are available.

Typically, DSCs utilize some form of removable media which store the digital images or other file data such as SmartMedia, CompactFlash, Memory Stick, Secure Digital ("SD"), or xD-Picture. Images on flash memory cards need to be DCF compliant for cameras to identify the DSC pictures. Extra directories and files may exist (even other partitions) on the flash media that are not visible to the DSC.

Some of the newer digital cameras have external cables that attach the camera to a computer. These attachments do not allow for the DSC to be read as a TTY terminal device for forensic copying. Thus, a memory card reader is necessary. It is possible to purchase an inexpensive universal memory card reader that reads most common cards. These memory card readers are typically attached to the computer via a USB port.

1) Write Block Protection: Some of the newer flash media have write blocking switches. If an evidence device has a write blocking switch, it should be activated for any duplication or investigative process. Unfortunately, most flash media do not have a write block switch, and thus other means are necessary for protecting valuable data from being overwritten.

With IDE hard drives, it is possible to use DOS based imaging utilities such as EnCase for DOS that will allow for software write block protection. Unfortunately, USB drivers are not available for these DOS based tools, and an alternate method of imaging flash media is necessary. With Windows based imaging (other than DOS), write blocking hardware is

TABLE III
METADATA TAGS USED IN EXIF (JEITA CP-3451)

EXIF IFD
Tags Relating to Version
ExifVersion
FlashpixVersion
Tags Relating to Image Data Characteristics
ColorSpace
Tags Relating to Image Configuration
ComponentsConfiguration
CompressedBitsPerPixel
PixelXDimension
PixelYDimension
Tags Relating to User Information
MakerNote
UserComment
Tags Relating to Related File Information
RelatedSoundFile
Tags Relating to Date and Time
DateTimeOriginal
DateTimeDigitized
SubSecTime
SubSecTimeOriginal
SubSecTimeDigitized
Tags Relating to Picture-Taking Conditions
ExposureTime
Fnumber
ExposureProgram
SpectralSensitivity
ISOSpeedRatings
OECF
ShutterSpeedValue
ApertureValue
BrightnessVAlue
ExposureBiasValue
MaxApertureValue
SubjectDistance
MeteringMode
LightSource
Flash
FocalLength
SubjectArea
FlashEnergy
SjpacialFrequencyResponse
FocalPlaneXResolution
FocalPlaneYResolution
FocalPlaneResolutionUnit
SubjectLocation
ExposureIndex
SensingMethod
FileSource
SceneType
CFAPattern
CustomRendered
ExposureMode
WhiteBalance
DigitalZoomRatio
FocalLengthIn35mmFilm
SceneCaptureType
GainControl
Contrast
Saturation
Sharpness
DeviceSettingDescription
SubjectDistanceRange
Other Tags
ImageUniqueID

TABLE IV
METADATA TAGS USED IN EXIF (JEITA CP-3451)

TIFF
Tags relating to image data structure
ImageWidth
ImageLength
BitsPerSample
Compression
PhotometricInterpretation
Orientation
SamplesPerPixel
PlanarConfiguration
YCbCrSubSampling
YCbCrPositioning
XResolution
YResolution
Tags relating to recording offset
StripOffsets
RowsPerStrip
JPEGInterchangeFormat
JPEGInterchangeFormatLength
Tags relating to image data characteristics
TransferFunction
WhitePoint
PrimaryChromaticities
YCbCrCoefficients
ReferenceBlackWhite
Other Tags
DateTime
ImageDescription
Make
Model
Software
Artist
Copyright

TABLE V
METADATA TAGS USED IN EXIF (JEITA CP-3451)

GPS IFD
Tags Relating to GPS
GPSVersionID
GPSLatitudeRef
GPSLatitude
GPSLongitudeRef
GPSLongitude
GPSAltitude
GPSTimeStamp
GPSSatellites
GPSStatus
GPSMeasureMode
GPSDOP
GPSSpeedRef
GPSTrackRef
GPSTrackRef
GPSImgDirectionRef
GPSImgDirectionRef
GPSMapDatum
GPSDestLatitudeRef
GPSDestLatitude
GPSDestLongitudeRef
GPSDestLongitude
GPSDestBearingRef
GPSDestBearing
GPSDestDistanceRef
GPSDestDistanceRef
GPSProcessingMethod
GPSAreaInformation
GPSDateStamp
GPSDifferential

```
# fdisk -l

Disk /dev/hda: 100.0 GB, 100030242816 bytes
255 heads, 63 sectors/track, 12161 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/hda1   *           1        2485    19960731    7  HPFS/NTFS
/dev/hda2            2486        2498      104422+  83  Linux
/dev/hda3            2499        2629     1052257+  82  Linux swap / Solaris
/dev/hda4            2630       12161    76565790    5  Extended
/dev/hda5            2630       12161    76565758+  83  Linux

Disk /dev/sda: 1015 MB, 1015808000 bytes
32 heads, 63 sectors/track, 984 cylinders
Units = cylinders of 2016 * 512 = 1032192 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1               1         984      991747+   6  FAT16
#
```

Fig. 1. fdisk example

necessary for ensuring that the evidence is not overwritten. (Windows XP does have a registry key that can be added for write blocking. This method of protection may work; however, it is not recommended because the software write block is not initiated between the BIOS and the operating system.) A list of tested write blocking hardware can be found on the Computer Forensic Tool Testing website, part of the National Institute of Standards and Technology ("NIST"). http://www.cftt.nist.gov/hardware_write_block.htm

A less expensive technique is to utilize software write block protection of an operating system that has control over the read and write functions of its devices. Linux allows for such control. In fact, Guidance Software supports the use of Linux software write block protection for acquiring devices as described in its article "How to Acquire a Drive Safely" found on its website http://www.guidancesoftware.com/support/articles/acquire_safely.aspx.

In its article, Guidance Software suggests the use of a non-auto mount distribution of Linux. By attaching an external device or card reader to the USB port, the device will not be automatically mounted. Read calls can be manually initiated to the device without altering or adding additional information.

Setting up a Linux forensic laptop or workstation can be laborious, yet rewarding. An easier and faster method is to use a bootable Linux CD that is customized for forensics analysis. A forensic bootable Linux CD can be used on just about any computer, allowing it to become a forensic laptop or workstation. A bootable Linux CD flavor that was recommended by the SANS Institute is Helix, http://www.e-fense.com/helix. Helix is a bootable Linux CD that is customized for forensics and incident response based on the Knoppix Live Linux CD.

2) Acquisition and Verification using Linux: Within Linux there are a few programs that a forensic examiner should become familiar with during the acquisition stage of an investigation: man, fdisk, mount, dd, and md5sum.

The command "man" is used to view manual pages, similar to a help file. If any questions arise about any of the commands, type "man fdisk", "man mount", "man dd", or "man md5sum". To escape out of a man page, type "q" for quit.

The command "fdisk" is a partition table manipulator for Linux. The program fdisk is used to identify the naming convention of the suspect and target device.

By typing "fdisk -l", the program fdisk will display devices with their associated partition tables as seen in figure 1. The first SCSI device (small computer system interface) associated with a Linux operating system is /dev/sda. Since a USB device

is viewed as a SCSI device by Linux, this paper will be referring to /dev/sda as the memory card, and /dev/sda1 as the first partition on the memory card. In figure 1, /dev/sda can be identified as the DSC flash memory card, since DSC images are both DCF compliant by the partition /dev/sda1 system type FAT16 and of the relatively small size of 1015 MB.

If /dev/sda1 is imaged instead of /dev/sda, then only the first partition is copied and not the entire device. Also, note /dev/sda is not always the name given to the memory card. /dev/sda could have been given to another SCSI device such as a SATA hard drive or other attached device. Thus the memory card may have a different name such as /dev/sdb or /dev/sdc, etc. The only way to distinguish the naming convention is to interpret the output generated by fdisk.

The command "mount" is used to mount a partition to a directory placeholder. Assume that the target drive and partition is /dev/hdc1 (the device /dev/hdc1 is equivalent to the first partition of the Secondary master IDE drive). To attach the partition to the directory /mnt/hdc1 use the command:

```
#mount /dev/hdc1 /mnt/hdc1
```

Other options can be used with the mount command such as -r for read only. To obtain more information on mount, please refer to the man page.

The command "dd" is an extremely powerful copy program found on most versions of Unix or Linux. The command dd can copy a file or device to another file or device. The output generated by dd is extremely versatile as it is understood by most, if not all forensic examining tools, commercial or open source.

Figure 2 assumes the input device (the flash memory card) is /dev/sda, the output directory is /mnt/hdc1, and the target output file is media.img. The command dd uses "if" as the input file and "of" as the output file. Additional options such as "bs=512 conv=noerror,notrunc,sync" ensure that the byte count for read and write is in chunks of 512 bytes, and if errors occur while copying, the process continues and pads spaces where errors occur.

To demonstrate the integrity of the image, the command "md5sum" can be used to generate a hash value of the image file and device. To obtain the md5 hash value of media.img and output to a file meida.md5, type the command:

```
#md5sum media.img >media.md5
```

The command "md5sum media.img" will output the hash value of the image file media.img. The ">media.md5" will redirect the hash value output to a new file media.md5. Once the md5 hash value is obtained, verification can be performed on the flash media device by the command "md5sum /dev/sda". The results of the md5 hash for the flash media device should match the image file.

Some variants of dd have extended its capabilities to perform more efficient forensic imaging operations such as dcfldd, the Department of Defense computer forensics lab ("DCFL") version of dd; however a strong knowledge of dd is essential.

```
#dd if=/dev/sda of=/mnt/hdc1/media.img bs=512 conv=noerror,notrunc,sync
1984000+0 records in
1984000+0 records out
1015808000 bytes (1.0 GB) copied, 105.739 seconds, 9.6 MB/s
#
```

Fig. 2. dd example

B. Analysis

It is common to find DSC images on a computer hard drive. Images are transferred to a computer hard drive via a camera attachment, email, CD/DVD ROM, external hard drive, thumb drive, network connection, web browsing, or by other means including attaching a memory card or camera. Within the Analysis phase we attempt to figure the What, Where, When, Why, and How!

What digital images exist? Where did they originate from? When were the images created or when were the images transferred? Why were the images transferred, intentionally or unintentionally? How were the images transferred?

Some of these answers can be found through extension analysis, header analysis, registry analysis, carving, metadata analysis, and hash analysis.

1) Extension Analysis: An examiner should use forensic analysis software such as EnCase, FTK, ProDiscover or Filehound to perform a search for all files that contain the file extension(s) associated with the type of DSC image .jpg. Such software has the ability to locate and group directory metadata associated with DSC images contained in the FAT or MFT including metadata associated with deleted files.

2) Signature Analysis: Why would some DSC images not contain a .jpg extension? A user could have intentionally altered an extension to hide the file. Macintosh OS does not utilize file extensions the same way as Windows and does not require an extension. Programs may give images other extension naming conventions. In any instance, a signature analysis search should be conducted to locate all files with matching DSC image headers.

FF D8 is the header for both JPEG and Exif files. If a file starts with FF D8, most likely it is a JPEG or DSC image.

Other areas to consider for locating DSC images are within compressed files, encrypted files, and emails.

3) Registry Analysis: "A record of all the devices detected since the system was installed is recorded under the HKLM\SYSTEM\CurrentControlSet\Enum registry key." (Russinovich & Solomon, p598) Within Microsoft Windows 2003, 2000 or XP operating systems, this registry key will list specific camera devices or flash media information such as Manufacturer, Location information, and HardwareID that were attached.

4) Carving Exif Images and Thumbnails: Once all the digital images are found within the allocated clusters on a device, it is time to search for the unallocated clusters and swap space.

Within a partition, clusters are allocated or unallocated. Allocated clusters belong to a file. Unallocated clusters are no longer part of a file; however, they still may contain data.

The basic building block of a file is a cluster. The clusters comprising a file are typically written in consecutive clusters. If a file did exist on a device it most likely was written

```
# mmls media.img
DOS Partition Table
Offset Sector: 0
Units are in 512-byte sectors

     Slot    Start        End          Length       Description
00:  -----   0000000000   0000000000   0000000001   Primary Table (#0)
01:  -----   0000000001   0000000248   0000000248   Unallocated
02:  00:00   0000000249   0001983743   0001983495   DOS FAT16 (0x06)
03:  -----   0001983744   0001983999   0000000256   Unallocated
#
```

Fig. 3. mmls example

```
# dd if=/mnt/hdc1/media.img of=/mnt/hdc1/media1.img bs=512 skip=249
count=1983495 conv=noerror,notrunc,sync
1983495+0 records in
1983495+0 records out
1015549440 bytes (1.0 GB) copied, 68.8978 seconds, 14.7 MB/s
#
```

Fig. 4. dd example

consecutively. In some instances where a file is fragmented, the next cluster in its cluster chain is written to the next available cluster or set of clusters. Thus, if the beginning or SOI of a digital image file is found, the remaining clusters should be relatively close.

Carving is a technique used to extract segment(s) of data within a larger body of data. Rules for carving are normally based on the header and footer being the start and stop points to carve the data segments. Rules also could be based on a header and a fixed length. There are many different carving tools; however, this paper will discuss the open source tool foremost, originally developed by the United States Air Force Office of Special Investigations and the Center for Information Systems Security Studies and Research. Current versions of foremost can be found at http://foremost.sourceforge.net.

Unallocated clusters are a large body of data that is a great data set to examine. Forensic software programs should have the ability to export the unallocated clusters to a file. Sticking with open source, a good tool to extract the unallocated clusters from a partition is dls. The program dls is a tool that can be found within Sleuth Kit and is maintained by Brian Carrier at http://www.sleuthkit.org.

For dls to work it has to read a partition file not a device file. The tool mmls can be used to obtain the key information on how to extract a partition into a file. The program mmls is another tool developed by Carrier and found within Sleuth Kit. In the dd example from figure 2, the device is media.img.

The tool mmls describes the file partition(s) within an image file. Figure 3 describes the FAT16 partition within media.img starts at sector 249 and has a length of 1983495 sectors. To extract the FAT16 partition using dd, options bs, skip, and count should be used.

In figure 4, the byte count for read and write process is in chunks of 512 bytes (one sector). The copy process starts after skipping 249 sectors (0-248) and the copy process occurs for 1983495 sectors.

Another method of extracting the FAT16 partition into the media1.img file would have been to copy the first partition out of the original DSC flash memory card /dev/sda1 found from the "fdisk -l" command in figure 1. The command "dd if=/dev/sda1 of=media1.img img bs=512 conv=noerror,notrunc,sync" will create an identical file to the one created from command in figure 4, possessing the same md5 value.

```
jpg   y   2000000   \xff\xd8\xff\xe1\?\?\x45\x78\x69\x66\x00   \xff\xd9   REVERSE
```

Fig. 5. carving rule for Exif

The tool dls can be used to extract the unallocated clusters from media1.img into a file media1.dls by typing "dls media1.img >media1.dls"

Once the data set of unallocated clusters is available, the tool foremost can be used. The command to carve files from media1.dls is:

#foremost -v -q -c ourfile.conf -o TEMPOUT media1.dls

Where:

-v = verbose

-q = quick (Searches for the header in the beginning of a sector. Most files start in the beginning of a sector; however when searching a compressed file or a cache file, the header might not start at the beginning of a sector.)

-c = configuration file (this is a file that should be edited to search only what is desired. The default configuration file is foremost.conf. The file foremost.conf has too many file structures to search for that might confuse results.)

-o = output directory

The default configuration file foremost.conf can be located in the source directory of foremost, or it might have been copied to the /usr/local/etc/ directory during installation. The file foremost.conf is a good reference as it describes how to write rules for foremost.

Each foremost rule set is written on one line. Attributes of a foremost rule include the extension, the maximum size, header and footer. A foremost rule that will carve out Exif image files is:

This rule described in figure 5 should be added to the outfile.conf configuration file for the foremost example to work The configuration rule in figure 5 uses 2,000,000 bytes as the size, which is close to 2MB - a little larger than an Exif file should be. In the Exif header, \x is the representation for a hexadecimal character and ? is a wild card character. \?\? is the two byte value that represents the length.

The foremost rule also has a footer. However, if we leave the footer out, foremost will extract 2,000,000 bytes, or, if the footer is not found, foremost will extract 2,000,000 bytes.

The REVERSE value is necessary after the footer, as it implies to start looking for the footer value 2,000,000 bytes after the header and work backwards. The reason for using REVERSE is the footer for both the Exif File and the Thumbnail is FF D9. If the REVERSE value was left out, the carving tool would carve from the beginning of the Exif header to the end of the thumbnail image (APP1 contains the thumbnail, and APP1 is 64Kb or less).

Another method of extracting the entire Exif image would be to leave the footer out. If the footer is left out, the foremost carving tool will extract 2,000,000 bytes, a file larger than the Exif image. The extra data will be ignored by an image viewer and will appear as normal.

To only extract the thumbnail image, the foremost rule

would be:

jpg y 200000 \xff\xd8\xff\xe0\?\?\x4A\x46\x49\x46\x00\xff\xd9

There are at least two good reasons to examine thumbnail images. The thumbnail image might be different from the original image. Also, the thumbnail image is much smaller than an Exif file and might have a better chance for full recovery without corruption.

Besides unallocated clusters, swap files and cache files such as Linux swap partition, Windows swap file pagefile.sys or AOL cache file cache.db are good spots to carve for images.

5) Exif Metadata: Table 2 described the metadata that could be associated with a DSC image. The program exiftool is an open source tool written by Phil Harvey that examines the metadata of Exif files. The exiftool can be downloaded from http://search.cpan.org/dist/Image-ExifTool/ or http://www.sno.phy.queensu.ca/~phil/exiftool/.

Since exiftool is written cleanly and published in open source, it would be easy to develop queries specific to an investigation. The metadata from table 2 can be extracted from all of the Exif images found on a device or devices (if the metatag exists). Thus, information such as date and time picture was taken, and GPS location of where picture was taken, could be extracted for investigation.

6) Md5 hash sets: Within the field of child pornography or intellectual property, hash sets can be used to identify an exact match for an image. Unfortunately, if a DSC image is altered slightly from editing attributes, the hash value will differ. Also, when an image is decompressed and saved recompressed again without any alteration, the saved file will most likely result in a lossier image, generating a hash value differing from the original.

It may be better to perform md5 hash analysis on the thumbnail image as apposed to the entire image. Steven Murdoch demonstrates in his blog http://blogs.23.nu/disLEXia/stories/5751/ that some image rendering programs do not alter the thumbnails.

7) DCF Structures: DSCs contain file structures that are DFC compliant. When a DSC is attached to a computer, chances are the directory of the DSC camera is read and files are copied or linked. Programs such as image viewers, image manipulators or antivirus may document touched files within a temporary file, log file or RAM. Evidence of DCF files and directories could remain in areas such as log files, the swap file, or unallocated clusters. Keyword searches for DCF files and directory structures could reveal some information about DSCs and DSC images that were attached or transferred to a computer.

DCMI - root directory for DCF
###???? - Directories that are DCF compliant are eight characters. The first three are numbers and the last five are alpha characters A-Z.
????#### - Files that are DCF compliant are eight characters (before an extension - usually .jpg). The first four are alpha characters A-Z including _and the last four are numbers.

III. Conclusion

While carving, some of the DSC images may have corruption within the visual representation of the image. Since an image is written one horizontal line at a time from left to right, it would be possible to reconfigure a broken image by developing a program that incorporates a hex editor with a visual representation of the image. Like Wireshark does to the IP protocol, an editor could isolate Exif segments visually with the DSC image. This would allow an investigator to piece together a corrupt image by easily cutting and pasting fragmented data. This also would allow an investigator to easily cut identifying tags for hex based searches and for screen shots during trial.

A Stegonographic program, like any program, leaves a footprint. Many of the stego programs could be better identified within the metadata and/or a histogram analysis. There should be a repository of footprints left by stegonographic tools, such as the MD5 hash values of common files found within the National Software Reference Library at http://www.nsrl.nist.gov for investigators and researchers.

References

[1] Aldus Corporation, *TIFF Revision 6.0*, June 1992.
[2] Canon Inc., *CIFF Specification on Image Data File*, December 1997.
[3] CCITT T.81, *Information technology – Digital compression and coding of continuous-tone still images – Requirements and guidelines*, September 1992.
[4] C-Cube Microsystems, *JPEG File Interchange Format* Version 1.02, September 1, 1992.
[5] B. Carrier, *File System Forensic Analysis*. Pearson Education Inc., 2005.
[6] M. Dornseif. "EXIF Thumbnail in JPEG images," December 17 2004. [Online]. Available: http://blogs.23.nu/disLEXia/stories/5751/.
[7] P. Harvey. "Read, Write and Edit Meta Information!," [Online]. Available: http://www.sno.phy.queensu.ca/~phil/exiftool/.
[8] Guidance Software, "How to Acquire a Drive Safely ," 2007. [Online]. Available: http://www.guidancesoftware.com/support/articles/acquire_safely.aspx.
[9] JEITA JEIDA-49-2-1998, *Design rule for Camera File System*, Version 1.0, December 1998.
[10] JEITA JEIDA-49-1998, *Digital Still Camera Image File Format Standard (Exchangeable image file format for Digital Still Camera:Exif)*, December 1998.
[11] JEITA CP-3451, *Exchangeable image file format for digital still cameras:* Exif Version 2.2, April 2002.
[12] JHOVE, "JPEG-hul Module," May 05 2005. [Online]. Available: http://hul.harvard.edu/jhove/jpeg-hul.html.
[13] M.E. Russinovich and D.A. Solomon, *Windows Internals - Microsoft Windows Server 2003, Windows XP, and Windows 2000*. Microsoft Press, 2005.
[14] T. Tachibanaya, "Description of Exif file format," February 03 2001. [Online]. Available: http://park2.wakwak.com/~tsuruzoh/Computer/Digicams/exif-e.html.
[15] M. Wandel, "Exif Jpeg header and thumbnail manipulator program," January 11, 2007. [Online]. Available: http://www.sentex.net/~mwandel/jhead/.
[16] Exchangeable image file format, *Wikipedia the Free Encyclopedia*, [Online]. Available: http://en.wikipedia.org/wiki/Exchangeable_image_file_format.

Kevin Cohen Kevin Cohen is an international computer forensics consultant. He is president of Data Triage Technologies, LLC. (http://www.datatriage.com), a computer forensics and electronic discovery firm based in Los Angeles.

Mr. Cohen has worked in the field of computer forensics since 1998. He is qualified as a computer forensics expert by multiple courts, has been appointed as a neutral computer forensics expert, and has given expert testimony in civil cases involving billions of dollars.

Mr. Cohen obtained his Bachelors of Arts in economics at the University of Colorado. He is a Certified Information Systems Security Professional (CISSP), Certified Information Systems Auditor (CISA), Global Information Assurance Certification ("GIAC") Certified Intrusion Analyst (GCIA), GIAC Certified Forensic Analyst (GCFA), and EnCase Certified Examiner (EnCE).