Forensic Data Carving

Digambar Povar and V.K. Bhadran

Center for Development of Advanced Computing, Trivandrum,
Ministry of Communications and Information Technology, Govt. of India
{paward,bhadran}@cdactvm.in

Abstract. File or data carving is a term used in the field of Cyber forensics. Cyber forensics is the process of acquisition, authentication, analysis and documentation of evidence extracted from and/or contained in a computer system, computer network and digital media. Extracting data (file) out of undifferentiated blocks (raw data) is called as carving. Identifying and recovering files based on analysis of file formats is known as file carving. In Cyber Forensics, carving is a helpful technique in finding hidden or deleted files from digital media. A file can be hidden in areas like lost clusters, unallocated clusters and slack space of the disk or digital media. To use this method of extraction, a file should have a standard file signature called a file header (start of the file). A search is performed to locate the file header and continued until the file footer (end of the file) is reached. The data between these two points will be extracted and analyzed to validate the file. The extraction algorithm uses different methods of carving depending on the file formats.

Keywords: Cyber Forensics, Data Carving, Slack Space, Lost and Unallocated Clusters.

1 Introduction

Computer forensics deals with the important problem of recovering files from digital media for which no file system information available [1][2]. The file system information can be unavailable for several reasons. First, digital media can be formatted to destroy the file system. Second, the file of interest may have been deleted such that the file system indexes no longer refer to the file content. Third, digital media may contain an unknown file system. Last, a file can be hidden in areas like lost clusters, unallocated clusters and slack space. In all the above cases, the file content is usually unchanged until the clusters belonging to the file are overwritten with other files. The process of file recovery from digital media by locating file signatures (header, footer) and extracting data between these end points is known as file carving. A file on digital media can start at cluster, sector, or at any byte (only in case of embedded files). To optimize the search process of locating the header signature, it is sufficient to search for the first few bytes of every cluster or sector. In case of embedded files, a search has to be performed byte by byte. The Boyer-Moore string search algorithm is used to perform a byte by byte search. The Boyer-Moore searching algorithm, described in R. S. Boyer and J. S. Moore's 1977 paper "*A Fast String Searching Algorithm*" [3] is among the best ways known for finding a sub string in a search space.

Using their method it is possible to search a data space for a known pattern without having to examine all the characters in the search space. One or more of the file carving methods proposed by Simson Garfinkel and Joachim Metz [14] are used to carve files depending on the file types. Our Forensic Data Carving method uses the following methods proposed by them.

- ❖ Header/Footer carving
- ❖ Header/Embedded length carving
- ❖ File structure based carving
- ❖ Carving with validation and
- ❖ Header/Maximum file size carving

Each of these methods with suitable example will be described in following sections. Files such as digital images (jpeg, gif, bmp, png), html, zip, compound documents (doc, ppt, excel, thumbs.db), pdf, video (avi, dat, mp4, mov, wmv, 3gp) can be carved using the above mentioned methods. We produced a tool using the methods discussed above. This tool is developed keeping two challenges in mind. First, carving files from hidden areas of the digital media when the file system exists. Second, carving files from any raw image that does or does not have a file system. To carve files from only hidden areas, we implemented a system that is available as a module for Cyber-Check V4.0 (a disk analysis tool). This module provides in-place (or zero storage) carving [14] facility from lost clusters, unallocated clusters and disk slack. It also supports carving files that are embedded into other files such as picture files embedded into documents and thumbs.db which contain picture thumbnails. To carve files from any raw image, we also created a stand-alone tool that can connect to external storage to carve files of interest.

2 An Approach to Minimize Search Time

Traditional forensic data carving tools available today search the header signature of a file in whole forensic image of a given digital media even if the file system exists for the media. To minimize the search space, it is sufficient to search header signatures in lost clusters, unallocated clusters and slack space of the disk that has a file system. Additionally, it is not necessary to search for the header signature of a file that is not embedded into another file in an entire cluster or sector. Any file that is newly created would be allocated a few fresh clusters or sectors and its header signature will be available in the first few bytes of first cluster or sector. Therefore, to minimize the header signature search time we suggest the following search options. Search the header signature in the:

- ❖ Option1: First few bytes of cluster
- ❖ Option2: First few bytes of sector
- ❖ Option3: Throughout the sector

All the above search options can be used for the digital media forensic image with a supported file system. It is important to note that sometimes, using the Option 1 search may lead to missing key evidence. For example, suppose a hard disk is formatted using a FAT32 file system and its cluster size is 2 sectors as shown in following Figure 1.

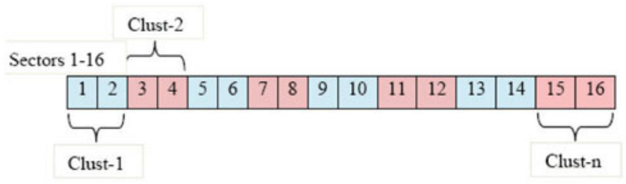


Fig. 1. Media representation with 2 sectors per cluster

If a file is stored in the second cluster, its header signature is stored in first few bytes of the second cluster or third sector. If the same hard disk is formatted and its cluster size is 4 sectors as shown in Figure 2, this search option misses the file starting at sector 3. This problem may be resolved by using the second search option (Option 2), i.e., beginning of the sector. In both cases above, first few bytes in the sense, length of the file header signature.

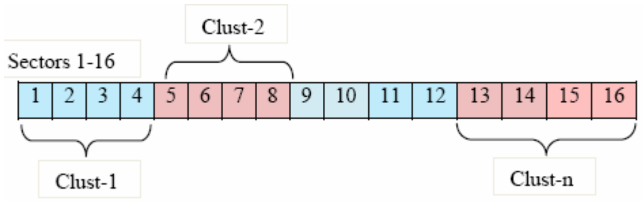


Fig. 2. Media representation with 4 sectors per cluster

It is also possible that a file can be embedded into another file i.e. picture files are embedded into document file and thumbs.db. In such a case both Options 1 and 2 cannot find the header signature, and Option 3 may be used. A search is performed to locate the file header signature throughout the sector.

To carve out files within a known file (for example document and thumbs.db file), a header signature search is performed only within that file which results in a faster carving process.

In case where digital media does not support a file system, Options 2 and 3 can be used.

3 Lost and Unallocated Clusters, Slack Space

A cluster is defined as a logical unit of file storage on a hard disk. Lost clusters are the clusters which are allocated to a file but are not having reference in the file allocation table. Lost clusters can result from files not being closed properly, from shutting down a computer without first closing an application (power failure) or from ejecting a storage medium, such as a floppy disk, from the disk drive while the drive is reading or writing.

Unallocated clusters are the clusters that are not allocated to any files on the hard disk according to the File Management System. These clusters can result from deleting a file or formatting the partition (or entire disk). While deleting a file or formatting the hard disk, data that is there in the clusters remains unchanged. If the resulted cluster in this case contains data, it is called a used unallocated cluster. If the data is not available, it is referred to as an unused unallocated cluster. A computer expert can identify and hide valuable information in these unallocated clusters.

Unused space in a disk cluster is defined as slack. The DOS and Windows file systems have fixed-size clusters. Even if the actual data being stored requires less storage than the cluster size, an entire cluster is reserved for the file. The unused space is called *slack space.* Slack space of the disk is mainly classified as file slack, partition slack, and disk slack. File slack and partition slack are always less than cluster size. Therefore, it is difficult for someone to hide an entire file in these slack spaces. Disk slack is slightly different from these two. There is no concept of cluster in disk slack. The space that is occupied by the set of sectors that are not coming as part of any partition of the disk is called disk slack. It can occur as a result of deleting a partition or leaving some part of a disk without partition.

4 Header/Footer Carving

This method of carving files is used when a file has defined header and footer. Jpeg, gif, png, html, pdf etc., may fall under this category. A file type can have more than one header and or footer, an example of that is an html file. In such a case searching for header and or footer should be performed repeatedly. Unique file headers and footers of various files that are supported by our tool are shown in the following table.

Table 1. Header, footer signatures

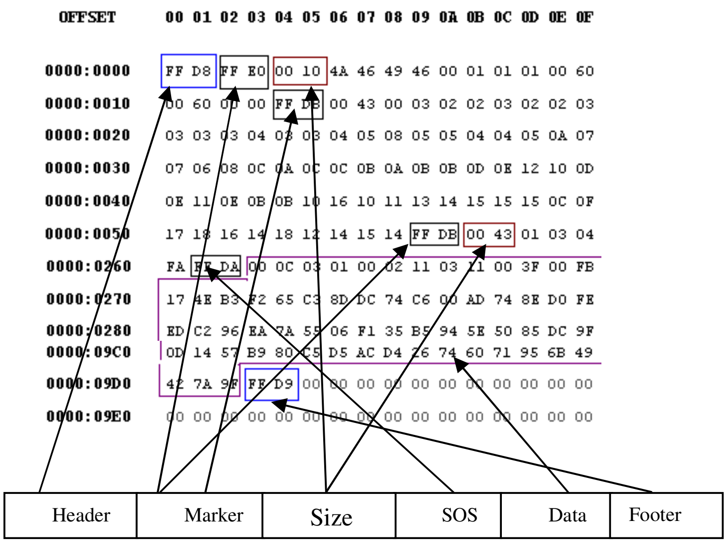
File	Header signature	Footer signature/ Method of carving
jpeg	FFD8	FFD9
gif	47494638	003B
png	89504E470D0A1A0A	49454E44
html	3C48544D4C3E	3C2F68746D6C3E
pdf	25504446	2525454F46
doc	D0CF11E0A1B11AE1	File structure based carving
ppt	”	”
excel	”	”
thumbs.db	”	”
zip	504B0304	”
bmp	424D	File size is embedded in the header
avi	52494646	”
dat	”	”
mp4	66747970	File structure based carving
mov	”	”
3gp	”	”
wmv	3026B2758E66CF11	”

Table 2. Jpeg segments

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 the compressed data

4.1 Carving JPEG Files

JPEG stands for Joint Photographic Experts Group, which is a standardization committee. It also stands for the compression algorithm that was invented by this committee. JPEG compressed images are often stored in a file format called JFIF (JPEG File Interchange Format). JPEG data structures are composed of segments (as shown in Table 2, that are marked by identifiers [4]. According to new JPEG [5] specifications, the new formats allow for multiple headers, footers and even nested images, to support thumbnails. Digital cameras often utilize the Application (APP) segment marker "0xffe1" to signify that they include more meta-data than the standard JFIF. The JPEG extraction algorithm need not search for footer from start of the header. Instead, it has to jump from the marker to marker until Start of Scan (SOS) marker.



```
OFFSET      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

0000:0000   FF D8 FF E0 00 10 4A 46 49 46 00 01 01 01 00 60
0000:0010   00 60 00 00 FF DB 00 43 00 03 02 02 03 02 02 03
0000:0020   03 03 03 04 03 03 04 05 08 05 05 04 04 05 0A 07
0000:0030   07 06 08 0C 0A 0C 0C 0B 0A 0B 0B 0D 0E 12 10 0D
0000:0040   0E 11 0E 0B 0B 10 16 10 11 13 14 15 15 15 0C 0F
0000:0050   17 18 16 14 18 12 14 15 14 FF DB 00 43 01 03 04
0000:0260   FA FF DA 00 0C 03 01 00 02 11 03 11 00 3F 00 FB
0000:0270   17 4E B3 F2 65 C3 8D DC 74 C6 00 AD 74 8E D0 FE
0000:0280   ED C2 96 EA 7A 55 06 F1 35 B5 94 5E 50 85 DC 9F
0000:09C0   0D 14 57 B9 80 C5 D5 AC D4 26 74 60 71 95 6B 49
0000:09D0   42 7A 9F FF D9 00 00 00 00 00 00 00 00 00 00 00
0000:09E0   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Fig. 3. Jpeg file format

The distance between any two consecutive markers is stored immediately after the first marker and that is two bytes in length. If the file is a valid JPEG then the last marker parsed will be the SOS marker, which signifies the beginning of the actual image data. Once this marker is reached then, our algorithm looks for the "0xffd9" marker, i.e., end of the image. This method of extraction increases the accuracy of extraction as well as the speed as entire headers are skipped instead of being processed by the searching algorithm. Figure 3 shows the data representation of jpeg file format with important markers.

5 Header/Embedded Length Carving

This method of carving files is used when a file has a distinct header and its length (file size) is stored in its first few bytes. Files that fall in this category are bmp, pdf, dat, avi, etc. This is one of the fastest carving methods used by our carving tool.

5.1 Carving PDF Files

PDF is a file format used to represent a document in a manner independent of the application software, hardware, and operating system used to create it [6]. A PDF file contains a PDF document and other supporting data. It is basically a binary file that also uses ASCII tags as delimiters to describe the header and trailer data structures in a Standard Generalized Markup Language (SGML) inspired fashion. A PDF file can have more than one footer signature. So, determining which footer actually represents the end of the file is a problem. As a result Kornblum and Kendall developed a REVERSE search mechanism [7] to find the last footer. The REVERSE method essentially looked for the last footer in the buffer and associated it with the given header.

As buffer size grows, the algorithms performance degrades. Further research of the PDF file specification revealed that a PDF contains multiple footers only if it has been "linearized". A linearized [6] PDF file is one that has been organized in a special way to enable efficient incremental access in a networked environment. The primary goal of the linearized PDF organization is to enhance viewing performance. Linearization is independent of PDF version number and can be applied to any PDF file version 1.1 or greater.

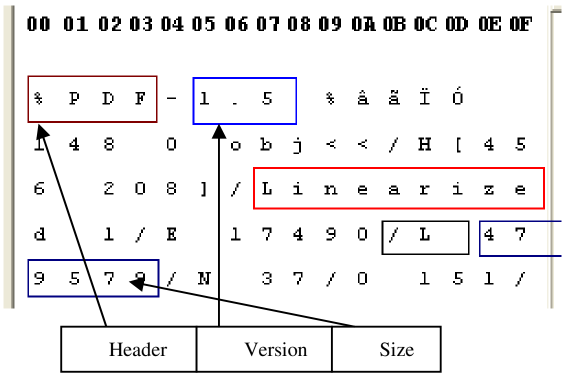
Algorithm to extract pdf Files

```
Step1. Look for the header signature (%PDF)
Step2. Check for the version number [file offset 6-8]
Step3. If version no. > 1.1 go to Step4
       Else go to Step6
Step4. Search for the string "Linearized" in first few
        bytes of the file
Step5. If it finds the string in Step 4, then length of the
        file is preceded by a "/L " character sequence as
        shown in fig.4
       Carved file size = embedded length;// 479579
       Else go to step6.
```

Step6. Use search algorithms to find footer signature
(%%EOF). Searching will be continued till
Carved file size<=User specified file size.

Note: - User specified file size is set by the user of the application to limit number of bytes to be carved. This is useful when a file does not have a defined footer signature.

The proposed algorithm is fast due to the fact that file size is often found within the first 100 bytes and no more file processing is necessary to extract linearized pdf files. In case a file is not linearized, a straightforward Boyer Moore search is performed to know the end of the file. Figure 4 shows the linearized pdf file header format.



```
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

%  P  D  F  -  1  .  5     %  â  ã  Ï  Ó
1  4  8     0     o  b  j  <  <  /  H  [  4  5
6     2  0  8  ]  /  L  i  n  e  a  r  i  z  e
d     1  /  E     1  7  4  9  0  /  L     4  7
9  5  7  9  /  N     3  7  /  0     1  5  1  /
```

Fig. 4. Linearized pdf file header

6 File Structure Based Carving

This method of carving files is used when the internal structure of file is known. Files, which use this method of carving, are zip, compound document files (MS Document files, MS Power Point files, MS Excel files, etc) and video files (mp4, mov, 3gp, wmv). Compound document files are also called OLE (object linking and embedding) files. Compound document files work similar to real file systems. They contain a number of independent data *streams* (like files in a file system) that are organized in a hierarchy of *storages* (like sub directories in a file system) [12]. An example of file structure based carving along with the algorithm is explained the following section.

6.1 Carving Zip Files

Zip files often contain multiple embedded files of varying formats. ZIP archives are a standard format for compressing and storing multiple files [16]. Files are structured in an incremental fashion, followed by a "central directory structure" as shown in Figure 5 [9]. Each file contained within the zip has its own valid ZIP header with its compressed and uncompressed data sizes stored within it (Figure 6). This information can be exploited to increase the speed of the extraction of ZIP files.

The Zip file carving algorithm works incrementally by parsing each local file header (Figure 6). The next local file header offset is the summation of the compressed file size, file name length, extra length and the size of the data structure itself (30 bytes). The iteration will be continued until the beginning of the central directory (Figure 7). Then the Boyer Moore search is performed to locate the end of the central directory record (Figure 8). To find the end of zip file, the algorithm reads the length of the comment field at location 20 from the end of central directory.

```
[Local file header 1]
   [File data 1]
   [Data descriptor 1]
   .
   .
   .
   [Local file header n]
   [File data n]
   [Data descriptor n]
   [Central directory]
   [Zip64 end of central directory record]
   [Zip64 end of central directory locator]
   [End of central directory record]
```

Fig. 5. ZIP file format [9]

Local file header signature	**4ytes (0x504B0304)**
Version needed to extract	2 bytes
General purpose bit flag	2 bytes
Compression method	2 bytes
Last mod file time	2 bytes
Last mod file date	2 bytes
CRC-32	4 bytes
Compressed size	**4 bytes** [offset: 18]
Uncompressed size	4 bytes
File name length	2 bytes [offset: 26]
Extra field length	**2 bytes** [offset: 28]
File name	variable size
Extra field	variable size

Fig. 6. Local file header [9]

Central file header signature	**4 bytes (0x504b0102)**
Version made by	2 bytes
Version needed to extract	2 bytes
General purpose bit flag	2 bytes
Compression method	2 bytes
Last mod file time	2 bytes
Last mod file date	2 bytes
CRC-32	4 bytes
Compressed size	4 bytes
Uncompressed size	4 bytes
File name length	2 bytes
Extra field length	2 bytes
File comment length	2 bytes
Disk number start	2 bytes
Internal file attributes	2 bytes
External file attributes	4 bytes
Relative offset of local header	4 bytes
File name	(variable size)
Extra field	(variable size)
File comment	(variable size)

Fig. 7. Central directory record structure [9]

End of central dir signature	**4 bytes (0x504b0506)**
Number of this disk	2 bytes
Number of the disk with the start of the central directory	2 bytes
Total number of entries in the central directory on this disk	2 bytes
Total number of entries in the central directory	2 bytes
Size of the central directory	4 bytes
Offset of start of central directory with respect to the starting disk number	4 bytes
Zip file comment length	**2 bytes** [offset: 20]
ZIP file comment	(variable size)

Fig. 8. End of central directory record structure [9]

Algorithm to extract zip files

Step1. Look for the header signature (**0x504B0304**)
Step2. Calculate the offset of next file header.
(Next file header offset=compressed file size+file name length+extra length+30)

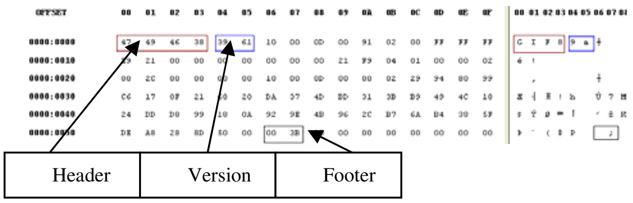
```
Step3. If content at the offset found in step2 is not
          equal to beginning of the central directory
          (0x504b0102) Go to Step2.
          Else
          Go to Step4.
Step4. Use search algorithms to find end of the
          directory record (0x504b0506)
Step5. Get Zip file comment length at offset 20 from
          the end of "end of the directory".
Step6. Offset of "end of the directory" and Zip file
          comment length gives end of the Zip file.
```

7 Carving with Validation

This method of carving files is used when a file needs to be validated using a file type specific validator. Files that use this validation method are gif, bmp, pdf, etc. This method is used in combination with one of the other methods.

7.1 Carving GIF Files

GIF was the first image format introduced for the needs of the World Wide Web back in 1987. Standing for Graphics Interchange Format, it represents a bitmap (graphics) file format, which is based on the 2D raster data type and which supports a wide range of resolutions [11]. There are two common versions of this format, 87a and 89a revision [10]. This format has remained unchanged for the last decade and thus has proven to be a rather easy file to extract. It is one of the few that has a defined header and footer. To carve a GIF file, one needs to acqure the file offsets of header and footer. Before carving data between these two points, a test is performed to determine if it is in fact a valid GIF file by looking for the versions "7a" or "9a". This kind of validation minimizes false positives while carving.



```
OFFSET      00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
0000:0000   47 49 46 38 39 61 10 00 0D 00 91 02 00 FF FF FF   G I F 8 9 a
0000:0010   E9 21 00 00 00 00 00 00 00 21 F9 04 01 00 00 02
0000:0020   00 2C 00 00 00 00 10 00 0D 00 00 02 29 94 80 99
0000:0030   C6 17 0F 21 60 20 DA 37 4D BD 31 3D D9 49 4C 10
0000:0040   24 DD D0 99 10 0A 92 9E 4D 96 2C D7 6A D4 30 5F
0000:0050   DE A8 28 8D 50 00 00 3B    00 00 00 00 00 00 00
```

Fig. 9. GIF file format

8 Header/Maximum File Size Carving

This method of carving files is used when a file has a distinct header and does not a have footer signature (or file that has footer, gets corrupted or overwritten). This is also useful in carving files where the embedded length of the file gets overwritten.

In such a case, the carving file size is limited by the user specified file size. This method is very useful to get part of the file when the file footer gets corrupted, file is fragmented, or embedded file length is overwritten.

Note

1. For the file that does not have a standard footer and does not store file size in the header, the carved file size will be equal to the user specified file size.
2. For the file that has a standard footer and does not store the file size in the header, the carved file size will be:
    a. Less than the user specified file size, if the file footer is found before the user specified file size limit is reached during the search process.
    b. Equal to the user specified file size, otherwise.

9 Results

The proposed approach of data carving was applied on a number of digital images of varying sizes. Because of file fragments and part of file corruption due to over writing, the percentage of false positives observed was approximately 13%. The following table shows the statistics of carved files using this approach.

Table 3. Results

Digital Image Size	**100% carved files**	**Partially carved files**	**% of correctly carved files**
2GB	579	63	89.2
4GB	1745	198	88.7
8GB	4740	571	87.9
10GB	5090	721	85.9
20GB	13020	1511	88.4
Average % of correctly carved files = 87			

10 Conclusion

All carving methods defined in this paper work fine for files that have defined header and footer (or embedded file length) signatures. Unfortunately, not all file types have a standard footer signature, so determining the end of the file can be difficult. Carving sometimes may be time consuming, because only one carving method may not be sufficient for carving files like GIF, bmp, pdf etc. If a disk or digital media contains the file system, the proposed approach for data carving will be the best to carve files from lost clusters, unallocated clusters, slack space and a known file. This paper emphasizes on minimizing the time and search of carving files from digital media that does or does not support a file system.

Acknowledgement

The presented work is a part of National Cyber Forensics Resource Center initiative, funded by Department of Information Technology (DIT), Ministry of Communications and Information Technology (MCIT), Govt. of India. The goal of this initiative is to establish C-DAC, Trivandrum as one of the center of excellence for research and development in Cyber Forensics technology. We would like to thank Shri. Ramani. B, Adl. Derector, Shri. Thomas K.L, Jt. Director, Shri. Balan C, Dy. Director, C-DAC, Trivandrum, India, for providing constructive comments and help in improving the contents of this paper.

References

1. Statistical Disk Cluster Classification for File Carving, Cor J. Veenman. Intelligent System Lab, Computer Science Institute, University of Amsterdam, Amsterdam
2. Richard, G.G., Roussev, V.: Next-generation digital forensics. Communications of the ACM 49(2), 76–80 (2006)
3. Boyer, R.S., Moore, J.S.: A Fast String Searching Algorithm. Communications of the Association for Computing Machinery 20(10), 762–772 (1977)
4. Hamilton, E.: JPEG File Interchange Format, Version1.02.1 (September 1992)
5. Joint Photographic Experts Group, JPEG 2000 Specification (2004), `http://www.jpeg.org/jpeg2000/` (last visited February 2009)
6. Adobe Systems Incorporated, Portable Document Format Reference Manual Version 1.3, March 11 (1999)
7. Naval Postgraduate School Thesis, Monterey, California, Nicholas Mikus (March 2005)
8. Digital Imaging Group, DIG2000 file format proposal, Appendix A (October 1998)
9. PKWARE Inc.: .ZIP File Format Specification Version: 6.2.0 (June 2004)
10. CompuServe Incorporated, Graphics Interchange Format (sm) (July 1990)
11. `http://www.ntchosting.com/multimedia/gif-graphics-interchange-format.html` (June 2009)
12. Sun Microsystems. OpenOffice, `http://www.openoffice.org/` (last visited: December 2009)
13. Wouters, W.: BMP Format (February 1997)
14. `http://www.forensicswiki.org` (last visited: March 2010)
15. `http://www.webopedia.com` (last visited: March 2010)
16. `http://www.pkware.com/documents/casestudies/` (last visited: January 2010)