April 2014

Forensic File Carving Tool Test Assertions and Test Plan

Draft Version 1.0 for Public Comment

NIST
National Institute of
Standards and Technology
U.S. Department of Commerce

Abstract

This document defines test assertions and test cases for digital file carving forensic tools that extract and reconstruct files without examination of file system metadata. The specification is limited to tools that identify inaccessible (deleted or embedded) files from file data content. Such tools exploit the unique data signatures of certain file types to identify starting and ending data blocks of these file types. In addition, file system allocation policies often keep file data blocks contiguous and sequential. For such contiguous sequential block placement identification of starting and ending data blocks may be sufficient to carve complete files. In other non-contiguous or non-sequential block placement, file reconstruction by carving is problematic.

As this document evolves updated versions will be posted at http://www.cftt.nist.gov

TABLE OF CONTENTS

1 Introduction

There is a critical need in the law enforcement community to ensure the reliability of computer forensic tools. A capability is required to ensure that forensic software tools consistently produce accurate and objective results. The goal of the Computer Forensic Tool Testing (CFTT) project at the National Institute of Standards and Technology (NIST) is to establish a methodology for testing computer forensic software tools by development of general tool specifications, test procedures, test criteria, test sets, and test hardware. The results provide the information necessary for toolmakers to improve tools, for users to make informed choices about acquiring and using computer forensics tools, and for interested parties to understand the tools capabilities. Our approach for testing computer forensic tools is based on well-recognized international methodologies for conformance testing and quality testing. This project is further described at http://www.cftt.nist.gov/.

The CFTT program is a joint project of the Department of Homeland Security, the National Institute of Justice, and the NIST Law Enforcement Standards Office and Information Technology Laboratory. CFTT is supported by other organizations, including the Federal Bureau of Investigation, the U.S. Department of Defense Cyber Crime Center, U.S. Internal Revenue Service Criminal Investigation Division Electronic Crimes Program, U.S. Department of Homeland Security's Bureau of Immigration and Customs Enforcement, U.S. Customs and Border Protection and the U.S. Secret Service. The objective of the CFTT program is to provide measurable assurance to practitioners, researchers, and other applicable users that the tools used in computer forensics investigations provide accurate results. Accomplishing this requires the development of specifications and test methods for computer forensics tools and subsequent testing of specific tools against those specifications.

Frequently during a forensic examination, data is discovered on the target media that is not part of any active or visible file. Although this data can still be examined at the byte level (e.g., string searching), the higher-level information is not apparent. If the data associated with a particular file could be identified and examined in its usual presentation format for the given file type, e.g., as a picture or video, this could provide more complete information. An example of this would be where a graphics file, if carved from unallocated space, could be viewed—potentially providing more information than a simple string search. Many of the forensic tools used by investigators identify files that have been deleted and allow the operator to recover them by file carving. This allows the investigator to examine the carved file in the original format (e.g., a graphics file viewer).

A fundamental problem is that the potential uncertainty present in any recovery effort leads to a reduced level of confidence in the information recovered. Specifically with file carving, the data recovered may be commingled with data from other deleted files, allocated files, or even from non-allocated space.

2 Purpose

This document defines the test assertions and test cases for tools used within forensic investigations to carve files. That is reconstructing deleted or extracting embedded files based on file content.

These test assertions were derived from the tool requirements presented in *Forensic File Carving Tool Specification*. Additionally, as this document evolves, feedback will be incorporated from a variety of sources, and will be posted to the CFTT web site at http://www.cftt.nist.gov for comments.

The test assertions are described as general statements of conditions that can be checked after a test is executed. Each assertion appears in one or more test cases consisting of a test protocol and the expected test results. The test protocol specifies detailed procedures for setting up the test, executing the test, and measuring the test results.

3 Scope

The scope of the test assertions and test cases is limited to software that is used for file carving. The proper or improper use of a tool is not within the scope of this document.

The test assertions for file carving are high-level, and are based on the following assumptions.

- The tools are used in a forensically sound environment.
- The individuals using these tools adhere to forensic principles and have control over the environment in which the tools are used.
- The carving tool input is a file or set of files that might be produced by a forensic acquisition tool acquiring digital media such as secondary storage or volatile memory.
- The files used test input to carving tools were created in a process that places file data blocks in a manner similar to how end-user activity would locate file data blocks.

4 Definitions

Included here are definitions of terms used in this document. Although there may be commonly accepted definitions for some of the terms, the context of this document may require a specific meaning.

Carved file: A file created by a file carving tool purported to be one of the source files present in the search arena.

Data block: File system specific data allocation unit (block), usually a multiple of 512 bytes. Some file systems may use other terms to describe a *data block* such as, *cluster* in FAT file systems.

File carving: Reconstructing deleted files from unallocated storage or extracting embedded files from a container file, based on file content; file system metadata may be a secondary consideration or completely ignored.

File-footer signature: A data string that identifies the end of a file. The string must be unique for a given file type. The string may begin anywhere within a data block.

File-header signature: A data string that identifies the beginning of a file. The string must be unique for a given file type. The string usually begins on a data block boundary, but it may begin anywhere within a data block.

Padding block: A data block in the search arena used to separate source file data blocks.

Metadata: The associated periphery information or attributes that describe a file such as name, time-based metadata (creation, modification, and last accessed times), access rights, ownership, and location.

Search arena: An acquisition file to be searched, e.g., the file obtained by acquiring unallocated space from a secondary storage device or acquiring primary memory from a running system. The search arena is composed of source file data blocks and other non-source file data blocks. A given source file may be complete, incomplete, fragmented, contiguous, sequential or non-sequential.

Source file: One of several files used to construct the search arena. All or part of a source file might be used. A carving tool should return a carved file for each complete source file in the search arena. The carved file returned by the carving tool should be visually identical to the original source file.

Target: The data blocks of a source file contained within the search arena.

5 Background

This section provides the technical background needed to discuss file carving tools and functions. The first subsection lists the file types that are candidates for file carving. The second subsection discusses some file system metadata considerations for file carving. Subsection 3 covers the most common properties of file systems. If combined with unique file content (file type signatures) these form the basis for most file carving algorithms.

5.1 Carving Source File Types

File carving is widely used in digital investigations to extract information from unallocated storage. Usually file carving is applied to file types with a recognizable structure so that unallocated space can be scanned for file components that are reassembled into complete files. Under some conditions this is an easy task. If the file has easily identified beginning and ending content and is contiguously allocated then carving is simple. However, the reality of file fragmentation complicates the task considerably.

Categories of files that are common targets of file carving include:

- Still Picture: JPG, GIF, PNG, BMP & TIF
- Videos: MP4, AVI, MOV, 3GP, OGV & WMV
- Audio: MP3, WAV, AU & WMA
- Document: DOC, DOCX, XLS, XLSX, PDF, PPT & PPTX,
- WEB: HTML, SQLite & chat
- Archive: ZIP, RAR, 7Z, GZ & TAR
- Misc: exec, logs, etc.

5.2 File System Metadata Considerations

A file system is used to store data for access by a computer. The data is normally stored within a tree-like structured hierarchy of directories and files. File system *metadata* contains information to describe and locate every file within a given file system. Some *metadata* resides in directory entries, but additional *metadata* may reside in special files (e.g., NTFS $MFT) or other locations (e.g., UNIX i-nodes).

When a file or directory is deleted, normally the associated *metadata* entry is flagged as being no longer active. However, in most file systems, neither the metadata associated with the file nor the actual content is completely removed.

File carving identifies deleted files by examination of data blocks present in the search arena to identify the beginning of a file. In general, file carving assumes that the file system metadata is overwritten, corrupt or otherwise unusable. However, some tools use knowledge of particular file system metadata to assist the carving process. For example, if a block of data is identified as an i-node or MFT fragment, that data can be excluded from consideration when carving a JPG file.

Metadata based deleted file recovery is addressed in the documents posted at the following link: http://www.cftt.nist.gov/DeletedFileRecovery.htm.

5.3 File System Properties

File systems are designed to allow an operating system to have access to secondary storage in a manner that is both efficient and timely. In the past, storage devices have been expensive, and slow (when compared to Random Access Memory). Accessing the hard drive efficiently, although implemented differently in each file system, tends to have some side effects that can be exploited to carve deleted files. Two of the key properties are contiguous writes and the conservative nature of file system activity.

File systems use contiguous writes if possible. Most operating systems write data to the drive in a contiguous set of data blocks or sectors if available. A given data file, provided it is not modified after being written to the disk, tends to have all the data in sequentially accessible sectors. This speeds up both the write and read processes, since the heads on the drive do not need to move to different areas on the disk to write or read data. This plays a role in data recovery, in that data from a given file, even deleted, has a high likelihood of being grouped together on the disk in contiguous data blocks.

File systems are conservative: this characteristic implies that, to be fast and efficient, file systems perform many activities with minimal changes or overhead. In the case of file deletion, in most situations, only a *logical deletion* is performed—meaning that the actual data is not erased, but the metadata that indexes the information is changed, flagged or removed.

For the most part, these common file system block allocation policies assist in the recovery of data on the drive, regardless of the type of file system the data resides on.

The behavior of a carving tool is dependent on the layout of data in the search arena. Files can be completely recovered if at least three conditions are present:

1. There is a uniquely identifiable start data block.
2. The file is contiguously and sequentially allocated.
3. There is a uniquely identifiable final data block.

Several problems may occur in practice that file carving tools might be required to deal with:

- Files begin on sector boundaries, so file-header signatures for non-embedded files are aligned on sector boundaries. However, an embedded file may begin anywhere within a data block and is not sector aligned.

- Not all file types have a uniquely identifiable final data block and may require tools to guess where the end of the file is located.

- If a complete source file is present in the search arena, but the file is fragmented then the carving tool needs to be capable of identifying all file fragments and assembling the fragments in the correct order. This is not an easy task and may not be possible is many cases.

- If a source file is incomplete within the search arena then it may be possible to assemble the first or last part a file from the available data, but this may not be possible is many cases.

These problems suggest some factors to consider in developing a carving tool test strategy:

- Placement of padding blocks data between source file data blocks in the search arena,
- Fragmentation of source files data blocks within the search arena,
- Order of source file fragments,
- Alignment of file-header and file footer signatures, and
- Source file completeness in the search arena.

6 Test Assertions

This section lists test assertions for file carving.

FC-CA-01. The tool shall report all accessible metadata entries for deleted files.

FC-CA-02. The tool shall return one carved file for each supported file header signature from a source file that is present in the search arena.

FC-CA-03. A carved file shall only contain data blocks from the search arena.

FC-CA-04. All data blocks in a carved file shall originate in a single source file.

FC-CA-05. The file type of a carved file shall match the file type of its contents.

FC-CA-06. The tool shall return carved files in a state that conforms to a valid file of the carved file type.

FC-CA-07. The tool shall ignore padding between targets.

FC-CA-08. The tool shall ignore padding between file fragments.

FC-CA-09. The tool shall reorder non-sequential file fragments.

FC-CA-10. If a target is incomplete, the carved file shall contain all source file blocks present in the search arena.

7 Measuring Conformance to Test Assertions

Two methods are used to measure conformance to test assertions. One method is used to evaluate the utility of the carved files and the other method is used to evaluate the relevance and completeness of the carved files.

7.1 Evaluating Utility of the Carved File

Evaluating file utility falls under the test assertion FC-CA-06. The returned carved file must conform to a valid file of the designated file type. For a graphic file, e.g., TIF, JPG, etc., the file must be viewable. For an audio file there must be a recognizable sound stream when the file is played. In addition to being viewable, audible, etc., the carved file may require further classification and evaluation by a human being. Some carved files may be incomplete or contain data from more than one source. Part of the human evaluation is to classify imperfectly carved files for usability. Three broad categories emerge based on the amount of information discernable for the carved file: *minor defects* (most file information is apparent to the human observer), *major defects* (most file information is lost to the human observer), and *unusable* (no significant information is discernable to the human observer). The central criterion for classification is the amount of file information that can be discerned.

For example, for graphic files the criterion is the fraction of visual elements that are observable. Minor defect examples include:

- A few missing scan lines from the display
- Corrupted color map that still allows the subject of the picture to be recognized.
- A few segments of the picture rearranged out of order when viewed.

Major defect examples include:

- Only small fraction of picture recognized
- Most of the displayed image a single color with no visible details.

7.2 Evaluating Relevance and Completeness

The relevance and completeness are evaluated by analysis of each carved file. The source files used to create the acquisition image files are known along with the layout of the acquisition image files. Sector hashes of the carved files can be used to identify the origin of each block of a returned carved file. This allows evaluating the conformance of the tested tool to the other test assertions.

Precision and recall are used in information retrieval to measure the effectiveness of a query to identify relevant documents from a document set. These measures can be adapted to measure the effectiveness of a carving tool to identify target data within an acquisition file. Calculation of precision and recall needs to have the following items defined:

- Set of documents to be searched
- Search query to specify relevance criteria
- Set of documents returned by the query
- Set of relevant documents returned by the query

For file carving the corresponding sets can be defined as sector sized (512 byte) data blocks:

- Acquisition file (set of data sectors)
- Tool carves acquisition for specified file type

- Set of sectors from the acquisition file returned in carved files
- Subset of returned sectors originating in files of the specified type

If we make the following definitions we can define precision and recall for file carving:

Ret: number of unique sectors returned as part of a carved file.

Rel: number of unique sectors in the acquisition file belonging to a source file of the file type being carved.

RnR: Number unique returned sectors that are also relevant to the file type being carved.

Precision: RnR/Ret

Recall: *RnR/Rel*

F score: The F score is the harmonic mean of precision and recall.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

It should be noted that these measures just indicate the effectiveness of the tool to return the desired data. They do not measure usability of the carved files. When the usability classification of the carved files does not match the F score, further investigation is indicated. For example, a high F score for a set of files that has a low usability score indicates a tool carving almost all the relevant data, but failing to ensure that the returned files are valid.

8 Test Cases

The test cases are general and can be applied to any file carving tool.

The test assertions derive the following test cases:

FC-01. Sector aligned contiguous files with no padding data between files. (No padding)
FC-02. Byte aligned contiguous files with no padding data between files. (Byte shifted)
FC-03. Contiguous files with padding data between files. (Cluster padded)
FC-04. Sequential fragmented files. (Fragmented in order)
FC-05. Non-sequential fragmented files. (Fragmented out of order)
FC-06. Intermixed fragments. (Braided pair)
FC-07. Partial files. (Incomplete)

The test data for each test case is constructed as a single acquisition image created from a pool of source files as follows:

File Carving Test Images	
Image Name	**Content**
No padding	All source files are adjusted so that the file length is a multiple of whole

File Carving Test Images	
Image Name	**Content**
	sectors. The adjusted files are concatenated together into one file that serves as the acquisition image. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data.
Byte shifted	All source files have varying length strings of bytes inserted before the first byte of the file. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data. The files are then adjusted so that the file length is a multiple of whole sectors. The adjusted files are concatenated together into one file that serves as the acquisition image.
Cluster padded	All source files are adjusted so that the file length is a multiple of whole sectors. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data. The adjusted files are concatenated together into one file that serves as the acquisition image. After each file is added to the acquisition image, some blocks of padding data are appended before the next source file is added. The length of data varies, but is always the size of a file cluster, i.e., 1, 2, 4, 8, 16, 32, or 64 sectors in length.
Fragmented in order	All source files are adjusted so that the file length is a multiple of whole sectors. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data. The adjusted files are split into several fragments. The fragments are concatenated together into one file that serves as the acquisition image. After each fragment is added to the acquisition image, some blocks of padding data are appended before the next source file is added. The length of data varies, but is always the size of a file cluster, i.e., 1, 2, 4, 8, 16, 32, or 64 sectors in length.
Fragmented out of order	All source files are adjusted so that the file length is a multiple of whole sectors. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data. The adjusted files are split into several fragments. The fragments are concatenated together into one file that serves as the acquisition image. The fragments are systematically rearranged so that the fragments are not in logical order. After each fragment is added to the acquisition image, some blocks of padding data are appended before the next source file is added. The length of data varies, but is always the size of a file cluster, i.e., 1, 2, 4, 8, 16, 32, or 64 sectors in length.
Braided pair	Source files are grouped into pairs, adjusted so that the file length is a multiple of whole sectors, and fragmented into two pieces each. The fragments are assembled such that the fragments alternate between to two source files of the pair. Cluster padding is inserted between fragments.
Incomplete	All source files are adjusted so that the file length is a multiple of whole sectors. The file length is adjusted by adding null bytes to the end of the file to pad out the last sector of data. The adjusted files are split into several fragments. Fragments from each source file are concatenated

File Carving Test Images	
Image Name	**Content**
	together such that one fragment is missing from each source file in the final acquisition image file. After each fragment is added to the acquisition image, some blocks of padding data are appended before the next source file is added. The length of data varies, but is always the size of a file cluster, i.e., 1, 2, 4, 8, 16, 32, or 64 sectors in length.

Figure 1 shows the relationships of the test images to each other.

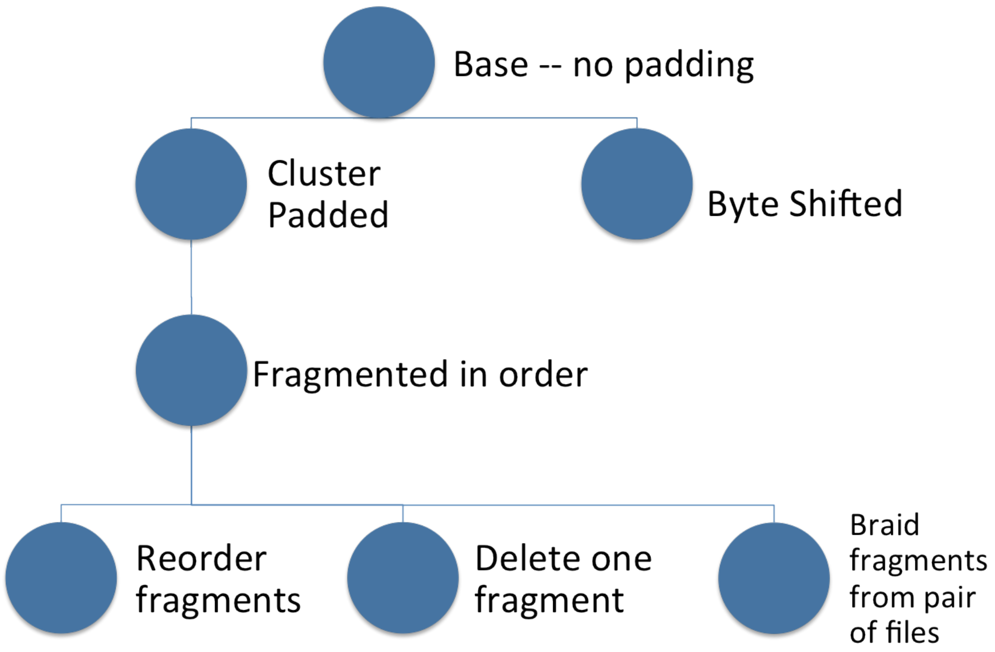

Figure 1Test Image Relationships

9 References (Informative)

It is important to note that these references are primarily informative.

Carrier, (2003). "File System Analysis Techniques: Sleuth Kit Reference Document." Available at http://www.sleuthkit.org/sleuthkit/docs/ref_fs.html.

Crane, (1999). "Linux Ext2fs Undeletion mini-HOWTO." Available at http://www.tldp.org/HOWTO/Ext2fs-Undeletion.html.

Erdelsky, (1993). "A Description of the DOS File System." Available at http://www.alumni.caltech.edu/~pje/dosfiles.html.

Himmer, (2000). "File Systems HOWTO." Available at http://www.faqs.org/docs/Linux-HOWTO/Filesystems-HOWTO.html.

Microsoft, (2004). “Description of the FAT32 File System.” Available at http://support.microsoft.com/default.aspx?scid=http://support.microsoft.com:80/support/kb/articles/q154/9/97.asp&NoWebContent=1.

NIST, (2004). “General Test Methodology for Computer Forensic Tools,” Available at http://www.cftt.nist.gov/.

Anandabrata Pal and Nasir Memon. (2009, March) www.smartcarver.com. [Online]. www.smartcarver.com/technology/research/pubs/ieee-spm-2009.pdf

Antonio Merola. (2008, November) www.sans.org. [Online]. http://www.sans.org/reading_room/whitepapers/forensics/data-carving-concepts_32969

Brian Carrier, Eoghan Casey, and Venema Wietse. DFRWS 2006 Forensics Challenge File Image Layout. [Online]. http://dfrws.org/2006/challenge/layout.shtml

Brian Carrier, Eoghan Casey, and Venema Wietst. DFRWS 2007 Forensics Challenge. [Online]. http://dfrws.org/2007/challenge/layout.shtml

Nicholas A. Mikus. Basic Data Carving Test #1. [Online]. http://dftt.sourceforge.net/test11/index.html

Nicholas A. Mikus. Basic Data Carving Test #2. [Online]. http://dftt.sourceforge.net/test12/index.html

S.J.J. Kloet, "Measuring and improving the quality of file carving methods," Department of Mathematics and Computer Science, Eindhoven University of Technology, Almere, Master's Thesis 2007.

Simson Garfinkel, Paul Farrell, Vassil Roussev, and George Dinolt, "Bringing science to digital forensics with standardized forensic corpora," in DFRWS, Montreal, 2009, pp. 2-11.

S. Garfinkel, "Carving Contiguous and Fragmented Files with Fast Object Validation," in Proceedings of Digital Forensic Research Workshop (DFRWS), Pittsburg, 2007, pp. 2-12.

G. Richard Golden III and Vassil Roussev, "Scalpel: A Frugal, High Performance File Carver," in Proceedings of Digital Forenwsics Workshop (DFRWS), New Orleans, 2005, pp. 1-10. [Online]. roussev.net/pdf/2005-DFRWS--scalpel.pdf

Ahmed Patel, Mustafa Mat Deris Kamaruddin Malik Mohamad, "Carving JPEG Images and Thumbnails Using Image Pattern Matching," in 2011 IEEE Symposium on Computers & Informatics , Kuala Lumpur, 2011, pp. 78-83.

Anabadrata Pal, Husrev T Sencar, and Nasir Memon, "Detecting file fragmentation point using sequential hypothesis testing," in Proceedings of the Digital Forensic Research Workshop (DFRWS), Baltimore, 2008, pp. 2-13.

Husrev T Sencar and Nasir Memon, "Identification and recovery of JPEG files with missing fragments," in DFRWS, pp. 88-98.

Kamaruddin Malik Mohamad, Ahmed Patel, Tutut Herawan, and Mustafa Mat Deris, "myKarve: JPEG Image and Thumbnail Carver," Journal of Digital Forensic Practice, vol. 3, no. 2-4, pp. 74-97, January 2010.

Simson L Garfinkel, Aleatha Parker-Wood, Daniel Huynh, and James Migletz, "An Automated Solution to the Multiuser Carved Data Ascription Problem," IEEE Transactions on Information Forensics and Security, vol. 5, no. 4, pp. 868-882, December 2010.