Advanced File Carving Approaches for Multimedia Files*

Rainer Poisel†, Simon Tjoa, and Paul Tavolato
Insitute of IT Security Research
St. Poelten University of Applied Sciences
St. Poelten, Austria
{rainer.poisel,simon.tjoa,paul.tavolato}@fhstp.ac.at

Abstract

File carving is a recovery technique that recovers files based on information about their structure and content without matching file system information. As files can be recovered from their content and/or file structure this technique is indispensable during digital forensics investigations. So far many approaches for the recovery of digital images have been proposed. The main contribution of this paper is a discussion of existing and new approaches for the recovery of multimedia files. After a short discussion of relevant multimedia file formats we present an overview of the current state-of-the-art in file carving. In the main part we focus on the implementation of a file carver for fragmented multimedia files. Finally, we summarize our findings and give an outlook with regard to post-processing files that have been recovered successfully.

Keywords: Forensics, multimedia, carving, recovery, fragmented.

1 Introduction

In the last decades the value of information vastly gained importance. Therefore the term "information society" has been introduced to refer to nowadays society. The creation, manipulation, distribution and usage of information have become fundamental activities in fields such as economy, politics and culture [2]. As a consequence, the usage, development and deployment of electronic devices as well as the exchange of information has steadily increased in recent years. According to Kryder's Law, the capacity of magnetic storage media such as hard disks is growing faster even than processor speed [3]. While, according to Moore's Law, processor speed is doubling every 18 months, storage capacity has increased 50 million times since the introduction of the disk drive in 1956. In 2008, the number of computers in use has surpassed the one billion mark [4]. The number of mobile phones worldwide is even higher: with 3.9 billion devices in 2009 and 4.2 billion devices in 2010 this number has still increased by 7% in just one year [5].

The numbers mentioned before are also reflected in crime related to IT infrastructures and computers in general. In 2011, Germany's Federal Criminal Police Office showed in their most recent report (2010), that the total number of crime committed using computers has increased (by 12.6%) while the detection rate has decreased (by 1.7%) again [6]. These developments can be traced back to the increased penetration rate with computing devices and the huge amount of data that has to be processed in each case. Recent ascertainments by the Francophone Association for Digital Investigation (AFSIN) have shown that on average, each suspect in a criminal case owns five hard disks, 140 CDs or DVDs and 4 memory cards and USB sticks. Analyzing business cases, the amount of data is even higher with up to 31 hard disks and 14 terabytes for a single case [7].

Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications, volume: 2, number: 4, pp. 42-58

*This paper is an extended version of the work originally presented at the 6th International Conference on Availability, Reliability and Security (ARES'11), Vienna, Austria, August 2011 [1].

†Corresponding author: Fachhochschule St. Polten GmbH, Matthias Corvinus-Straße 15, A-3100 St. Polten, Vienna, Austria, Tel: +432742313228-637, Email: rainer.poisel@fhstp.ac.at

During digital investigations various types of media have to be analyzed. Relevant data can be found on different storage and networking devices and computer memory. Different types of data such as e-mails, electronic documents, system log and multimedia files have to be analyzed. Within this paper we focus on the recovery of multimedia files which are stored on either storage devices or computer memory using the file carving approach. File carving is a recovery technique which merely considers the contents and structures of files instead of file system structures or other meta-data which is used to organize data on storage media. Table 1 summarizes the file carving terminology. It is used throughout this document.

Term	Definition
Block	The size of the smallest data unit that can be written to storage media. It refers to either the sector or the cluster size.
Header	Header blocks contain the starting point of a file.
Footer	Footers contain the ending point of a file.
Fragment	One block or a sequence of blocks that belong to one file. One file can be built from different fragments which are not sequentially connected to each other. The distance between different fragments of one file is unknown, further it is possible that fragments do not exist anymore because they have been overwritten.
Base-fragment	The first fragment of a file. It contains the header (if available for the filetype investigated).
Fragmentation point	The last block of a file before fragmentation occurs. As a file can consist of multiple fragments it is possible that there exist multiple fragmentation points.
Fragmentation area	Consecutive blocks which are grouped into a set and which contain the fragmentation point.

Table 1: File carving terminology based on Pal et al. [8]

The main contribution of this paper is a presentation of the state of the art of file carving as well as a presentation of current developments in this field. Mainly this paper focuses on the recovery of fragmented multimedia files. The recovery process is computationally complex, therefore we also discuss different optimizations to improve the overall performance of such file carvers.

As video files can be seen as the prime example for multimedia files, we refer to these file types when we do not explicitly distinguish between file types that only contain one kind of media (visual or auditive). The following section discusses formats of multimedia files that are currently used on the Internet.

1.1 Relevant Formats of Multimedia Files

This paper discusses the content-based recovery of multimedia files from various types of storage media. By “multimedia” files we refer to files that contain both auditive and visual information, such as digital movies.

Container files are files that may contain different files of different file types. The format of the container file describes how the different contents (subfiles) are arranged within such a file. Multimedia files basically consist of header or metadata, as well as visual, auditive, and text-based information [9]. Therefore it can also be refered to as a “compound document”. Video data is compressed by a so called

"codec" program which reduces the required amount of data in a (usually) lossy fashion. With support for HTML5, web browsers support the playback of video files natively [10]. According to a recent study [11] more than 50% of nowaday's web browsers support the playback of HTML5 videos. Since 2009 there was an increase of 66%. Therefore, in our study, we focused on the recovery of video formats supported by the HTML5 draft standard.

According to the HTML5 draft standard [10] and documentation of major web browsers [12], the following video formats are supported by web browsers:

- Ogg containers with Theora video streams,
- MPEG-4 containers with H.264 video streams and
- WebM containers with VP8 video streams.

Web browsers store the contents of surfed web pages in a cache for faster access [13]. As with HTML5 multimedia files are part of web pages contents, it can be expected, that such files can also be found in the cache of web browsers. Cohen [14] described the investigation of a browser's cache using the PyFlag framework. A browser's cache is typically organized in regular files on a file system and therefore it is possible to acquire content from visited web pages with tools from this field as well.

2 Related Work

As recovery strategies are key components for disaster recovery, forensics and e-discovery the need for improvements in these fields have been highlighted by Pal and Memon [15] in their research paper. Classical approaches for the recovery of files from corrupted hard disks or storage media in general are often based on file system meta-data [16]. Digital investigators and emergency personnel consider content based recovery techniques in case required file system information is not available.

Fragmentation complicates content based file recovery as it shuffles the constituent parts of the file which should be recovered. Pal et al. [8] gave different reasons why fragmentation occurs on storage media. Flash based devices such as solid state disks utilize wear-leveling algorithms to extend their lifetime [17]. These algorithms are implemented in the storage controller of the storage device. Storage cells are used in an even fashion, but it is difficult to determine the correct sequence of blocks as most wear-leveling algorithms are proprietary and therefore unknown to the digital investigator.

Tools that focus on the recovery based on file system structures have been presented by Carrier [16]. His findings have been compiled into a suite of small programs "The Sleuth Kit" [18]. Each of these programs has its dedicated purpose, such as calculating file/directory listings or storage units of specific files.

One of the first file carvers which has been mentioned in scientific research papers is "Scalpel: A Frugal, High Performance File Carver" [19]. It only supports the recovery of unfragmented files by defining signatures for the beginning and end of file types that should be recovered. Data between these defined signatures is extracted and represents the recoverd files. Garfinkel [20] extended the signature based approach by considering meta-data in files intended for the recovery. Some file headers contain information about the file length. In case a file format does not have a distinctive signature for the end of a file this information can be interpreted to extract the files' contents. Further, Garfinkel [20] proposed the "Bifragment Gap Carving" approach for files that are fragmented into exactly two fragments.

After determining the starting and end points of a file, the data between these two points is tested for its validity. If the test fails a gap which is excluded from the validation procedure is introduced. The boundaries of this gap are modified until the validation succeeds. As a subset of recovering files,

Garfinkel [20] discussed the recovery and validation of objects. Especially for container formats such as the Portable Document Format (PDF) it is possible to extract meaningful information from within such a file [21]. In this context the purpose of a validator is to ensure that data recovered by file carving can be decoded successfully. However, certain decoders which have been used as validators do not necessarily generate an error if they encounter invalid information. Further, it is possible to successfully decode syntactically correct, but semantically invalid data. Therefore the validity of investigated material has to be ensured by further analysis, e. g. semantic validation.

Cohen [22] considered the recovery of fragmented files and described the carving process as being equivalent to estimating a mapping function between bytes copied from an image of storage media to the recovered file. Files could be recovered using a generator that produces all possible mapping functions. Results of these mapping functions are evaluated for their validity. The downside of this approach is the vast number of combinations (see "Reassembly" later in this paper). Cohen also discussed another critical component of a file carving system: the discriminator. This component can tell if a recovered file is corrupt or likely to be correct. It's result is fed back into the mapping function generator. This way the recovery process can be improved by excluding mappings that are incorrect before they are evaluated for their correctness. Cohen [22] described mapping function generators for the PDF and ZIP file formats.

Further improvements have been proposed by Pal and Memon [15]. For the recovery of fragmented files they introduced the SmartCarving architecture which is shown in Figure 1 extended by a post-processing step. This approach is not limited to the number of pieces into which a file is fragmented by the underlying file system or storage algorithm. Nevertheless, the process of recovering files from their fragments can be optimized as files infrequently fragment into more than three fragments [20]. So only a small number of all possible permutations of constituent fragments has to be tested for their correct order in case fragments have been identified successfully. Three steps have been identified and described [15]: preprocessing, collation and reassembly. In Figure 1 rectangles (e. g. F_1, H_1, ...) show file fragments that belong to a specific file and their index denotes their position within that file. Recovered files are displayed as sheets of paper on the right. These recovered files can be handed over to the post-processing step to be analyzed for their relevance in the currently investigated case. In the following the three steps of the smart carving technique are described.

Preprocessing : In case file system meta-data can be interpreted it is used to remove known and/or allocated clusters from the file carving analysis. Thus it is possible to reduce the amount of data that has to be analyzed. Carrier describes methods to determine regular as well as deleted files from storage media based on remaining file system information [16]. In practice, tools such as "The Sleuth Kit" [18] are used to extract unallocated disk space from an image.

Collation : Blocks that cannot be assigned to files by the help of information used in the preprocessing step are categorized according to their file type. This allows to further reduce the number of fragments that have to be considered for each file intended for recovery. For the identification of file types several techniques have been discussed: matching keywords and signatures has been proposed by Pal et al. [8]. Certain file formats such as HTML are characterized by specific tags or sequences of bytes. Hence the blocks that contain these tags are assumed to be of that specific file type. As an additional deciding feature it is possible to distinguish different file types of adjacent blocks by comparing their information entropy [15]. Other approaches comprise statistical analysis of fragment data [23], others consider the "Normalized Compression Distance" [24]. As there are many file formats with similar characteristics Roussev and Garfinkel [25] proposed the usage of specialized approaches depending on the file type that should be recovered.

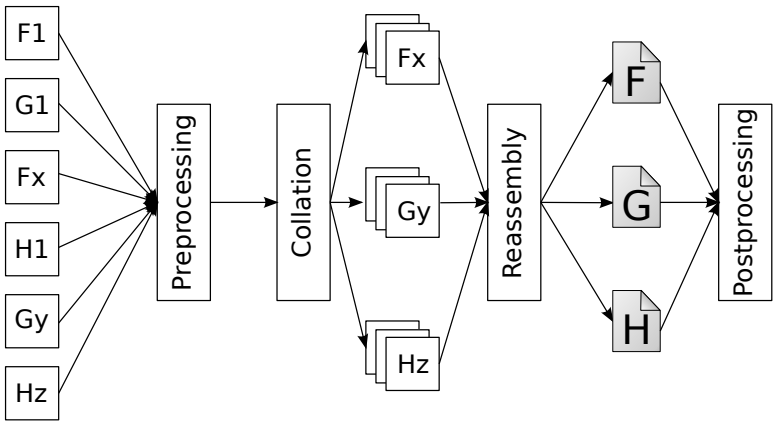

Figure 1: Extended "SmartCarving" architecture derived from Pal and Memon [15]

Reassembly : In this step the fragmentation points of unrecovered files are determined. Further, during this process fragments are re-ordered and merged in order to obtain the original files. Up to now the reassembly of fragments has been optimized for JPEG-files. For this purpose several approaches have been proposed.

Pal and Memon [26] proposed the application of the Parallel Unique Path (PUP) algorithm to reorder fragments. Their approach was not optimized to the way how file systems actually fragment files. Therefore they and others proposed the addition of "Sequential Hypothesis Testing" (SHT) to the PUP algorithm in a later paper [8]. SHT tests adjacent blocks from a storage medium to be in sequence. A weighting function determines if the investigated blocks which are tested in sequence are part of the fragment in question. Based on the results of the weighting function the outcome of the hypothesis (SHT) can be determined: the blocks belong in sequence to the fragment (H_1) or they do not belong in sequence to the fragment (H_0). Once the fragmentation point is reached the PUP algorithm is applied to determine the starting point of the next fragment. Garfinkel [20] showed that the gap between file fragments is rarely larger than 80 blocks. Based on that fact the PUP algorithm has further been optimized [8]. The extension is known as "close region sweep" and tries to decode a few blocks in an area around header fragments. Optimized approaches for finding the fragmentation points of JPEG files has been discussed by Karresand ([27] and [28]). Yoo et al. [29] discussed the recovery of multimedia files using the file carving approach from compressed NTFS partitions.

Metz and Mora [30] presented an open-source implementation of the smart carving architecture as part of their submission to the DFRWS 2006 challenge. Their tool "ReviveIt" can be configured to support the recovery of text-based file formats such as the Portable Document Format (PDF). An overview of the possibilities for post-processing multimedia files in the field of digital forensics has been proposed by Poisel and Tjoa [31].

3 File Carving of Multimedia Files

In this section we discuss existing considerations of former file carving approaches and how these concepts are integrated in our file carver for the recovery of multimedia files. It is structured into the generation of test-data as well as the constituent parts of a Smart Carver which has been extended by post processing procedures.

3.1 Automated Generation of Test Data

The purpose of a file carver for fragmented files is to be able to recover files if they are not stored in sequence of consecutive blocks on storage media. Data between the start and the end of a file incorrectly represent a reconstruction of the file if it was fragmented [8].

Moch and Freiling [32] followed an approach that generates images that contain a complete Linux-based operating system including some user-created information (e. g. the person creating the image surfs arbitrary sites on the Internet using a web browser). Therefore this approach results in images that are more similar to real-world scenarios than the approach chosen for our project.

As part of our project a Python script has been developed [1] for the automatic generation of file system images that can be processed with the file carver proposed in this paper. Several parameters affect the fragmentation of files stored in the generated file system image: size of the image, used file system, list of additional files that are copied to the generated image, the files that should be tested for their fragmentation behavior, size and number of randomly generated files, and the number of iterations the file system is filled up with random files. During each run created files are randomly deleted until a predetermined amount of space is emptied. After that, files chosen for their inclusion are copied to the generated image.

3.2 Preprocessing

During the preprocessing step digital evidence is processed to obtain the original data by decompressing or decrypting it. Hard drives which originally have been encrypted using Microsoft's BitLocker technology [15] are a typical example for this scenario. To be able to process stored data, the disk has to be decrypted first. Recently, attacks have been presented to obtain the encryption key by analyzing images of physical memory from such machines [33].

Further, during the preprocessing step data that can be referenced through existing file system information or structures from investigated memory can be excluded from the analysis process. Only unreferenced contents (e. g. slack space or unallocated space) are forwarded to further processing steps to keep the amount of data that has to be analyzed at a minimum. For this purpose suitable open-source software such as "The Sleuth Kit" (TSK; based on [16]) [18] is available.

Some file systems such as NTFS, FAT [34] and EXT [35] pool multiple sectors (512 bytes) into clusters. The used cluster size can be determined by interpreting the file system's metadata during the preprocessing step. This information can be utilized to optimize the process of determining the fragmentation points as well as for the reassembly process as there is more data available for file type identification. In case file system information should be ignored or in case it is not available, the block size is set to the sector size (512 bytes).

To improve the performance of the preprocessing step, different tools can run at the same time on different computation nodes. Results gained from the different nodes have to be merged so that only unreferenced data remains for further processing steps (collating and reassembly).

3.3 Collation

After sorting out irrelevant blocks (e. g. by only retaining unallocated blocks) the file type of the remaining blocks is identified [15]. File types represent specific file formats such as MPEG, the Microsoft Office format, or HTML files. Sorting out blocks from relevant file types determines those blocks that can be worked on in later processing steps. Therefore it further reduces the amount of blocks that have to be considered during the recovery process. Several techniques have been developed for the classification of filetypes from byte streams. One fact that has to be considered is the small fragment size as it occurs during file type identification of packets when analyzing network traffic. Because of the small number

of available bytes it is likely that the wrong file type is determined during this process. Further problems arise when analyzing compound file types, such as PDF files.

Identifying the file type can be performed using two models [36]: type-x recognition models and type-all recognition models. Where the first refers to models which allow to determine fragments of a specific file type, the latter refers to models which classify clusters or blocks into pre-defined file types. Type-x recognition models have been found more useful as they allow the false positive rate to be adjusted per file type. In the following sections, well-known techniques within the context of file carving are presented.

Signature-based classification is well-known from the field of file type identification from whole files using the "file" command on UNIX systems [37]. This tool is based on the "libmagic" library which in turn accesses a database of byte sequences, so called signatures, which are characteristic for certain file types. Open-Source tools such as "scalpel" [19] carve files based on their signature. Further, metadata can be considered during the recovery. It contains information like the file size or parameters which determine how the data has been encoded and can be used to verify the carving results. Tools such as PhotoRec [38] consider this information. The file carver for the recovery of multimedia files uses this approach to determine the beginning of such files. The number of video files that should be recovered also determines the number of parallel execution paths for the reassembly part.

Feature-based classification has been described by Roussev [25]. Depending on the searched file type classification results can be improved by considering properties which are characteristic only for that file type. Roussev described that it was difficult to determine the filetype of compressed data (zlib) as it is similar to random or encoded data. On the other side, it was possible to determine the filetype of several multimedia files such as MP3 or JPG files from fragments. For the multimedia file carver this approach is chosen for the distinction between JPG and H.264 video files. Tests have shown that some of the marker codes defined in the JPEG standard [39] only occur rarely in H.264 video files.

Normalized Compression Distance determines the file type by comparing the information entropy of reference data and the blocks in question. Axelsson [24] described the result of the NCD as a measure of how distant data vectors are. The data vectors are handed over to the compression algorithm both individually and concatenated. In Equation 1 $C()$ refers to the compression algorithm, x and y refer to the individual data vectors and x,y refer to the concatenated data vectors. The smaller the result, the more similar the data vectors in question are.

$$NCD(x,y) = \frac{C(x,y) - \min(C(x), C(y))}{\max(C(x), C(y))} \tag{1}$$

For the multimedia file carver "zlib" has been chosen as a compressor. The NCD approach has been used to distinguish blocks of text-based file types from blocks with compressed data.

Statistical approaches have been used to distinguish files from text-based formats from files of binary formats, such as compressed files like digital images or multimedia files. Shannon [40] classified files based on their amount of printable ASCII characters as well as their information entropy.

Some statistical approaches consist of a training as well as a classification phase. Li discussed the usage of "fileprints" which represent "all members of a the same file type by a set of statistical 1-gram models" [41]. The file type of a file can be identified without parsing the data. Therefore this method has been optimized to be ressource efficient so that it can be used in networking appliances with high data throughput. Li et al. state in their work that only "one single descriptive model that accurately

represents all members of a single file type class" [41] may not be enough. Hence the results of previous approaches can be poor with a low number of exemplars. Furthermore the authors criticized that with other approaches the data has been normalized. This method seems adequate for files with different information entropy. Accuracy for classifying the file type has been between 62.7% and 100% depending on the amount of data investigated: the lower the truncation size, the better the results.

Veenman improved the classification performance by exploiting "the statistics of a restricted number of neighboring clusters" [36]. Preceeding and consecutive clusters were used to establish a context which resulted in more reliable statistics. One disadvantage is that it is possible to include blocks of other file types into the calculations. With his experiments Veenman showed, that the classification results depend on the classified file type.

Calhoun and Coles [42] combined two statistical approaches to determine the file type of file fragments: type prediction with the linear discriminant and type prediction with longest common substrings and subsequences. Linear discriminant prediction is based on the combination of various statistical properties such as information entropy, mean byte value or standard deviation of the byte frequencies. The latter approach, longest common substrings, is based on determining the longest substring that is common between the fragments in question. Two test sets have been created to validate the approaches: one consisted of 896 byte long fragments from files which had the first 128 bytes removed and the other consisted of 512 byte long fragments from files which had the first 512 bytes removed. The results showed that the accuracy strongly depends on the investigated file types, the number of bytes removed from fragments and the statistical properties chosen. The best result could be achieved for the distinction between fragments from JPG and PDF files.

Conti et al. presented a taxonomy of the different file types of binary fragments. It is "based on common types of source media and common ways this media may be encoded, encrypted, or compressed" [43]. Several databases of media types such as the MIME (Multipurpose Internet Mail Extensions, RFC 2046 [44]) have been reviewed to obtain a comprehensive catalog. Based on this catalog the different characteristic visual patterns of file fragments have been discussed. The usage of this approach requires a training phase within which visualizations for the different file types are generated. The file type of fragments can be determined using comparison algorithms known from the field of digital image processing.

The collation process can be parallelized as it allows to process different areas of the investigated image independently. After distributing the classification process to several processing nodes (e. g. cores of a multi-core processor) the results have to be aggregated to be handed over to the fragment reassembly.

3.4 Reassembly

As a first step in the reassembly process, after the file types of the blocks from the analyzed image are determined, subsequent blocks of similar file type are grouped into so called fragments. The reassembly process then refers to finding the right order of these fragments. The sequential order of fragments can be determined by ordering fragments based on the similarity of their boundaries. Within this section the complexity of two ordering approaches is compared. Further, different approaches for determining the togetherness between fragments are presented.

Grouping There is no algorithm for determing the file types of fragments completely accurately. It is possible that certain blocks are excluded from the recovery process because their file type has been determined wrongly (false negative). On the other hand it is unlikely that a file system fragments its managed files into many small fragments with sizes of only a few sectors [20]. Therefore blocks of the same data type which are close to each other are merged into fragments. The grouping parameters

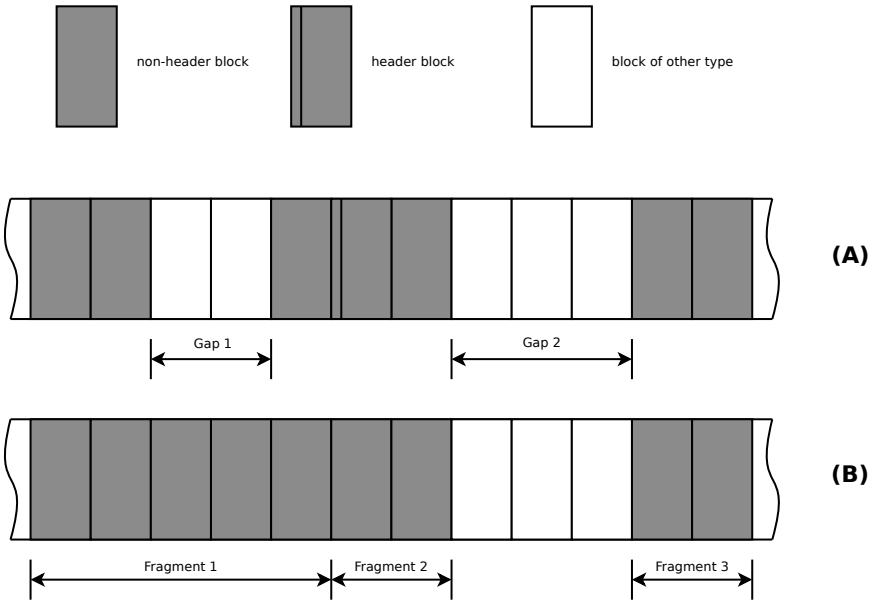

Figure 2: Summary of fragments

depend on the characteristics of the underlying file system. Our file carver considers two parameters: "block-gap" and the minimum fragment size.

The "block-gap" determines the number of blocks that are counted to the surrounding fragments, even if they are of different file type. An example is shown in Figure 2A. "Gap 1" is less or equal the "block-gap" (predetermined to be two blocks). It is therefore counted to Fragment 1 (see Figure 2B). Gap 2 is bigger than the "block-gap". Fragments before and after this gap are counted to different fragments (Fragment 2 and Fragment 3). In contrast to the scenario described before, fragments are broken up at the position where header blocks are identified. This scenario is shown in Figure 2B between Fragment 1 and Fragment 2.

The minimum fragment size refers to the minimum number of blocks that make a fragment. In Figure 2B Fragment 3 would be excluded from further analysis in case the minimum fragment size would have been predetermined to be two blocks. Fragment 2 would still be in for the reassembly process as it is a header fragment containing information in its header which could be necessary to decode other non-header fragments which meet the requirements for fragments.

Determining the reassembly order In the next step the reassembly order of fragments is determined. A simple approach is the comparison of all header fragments with all non-header fragments. In this case all permutations have to be calculated and the weight between fragments is not determined. Every permutation that can be decoded is valid. This simple approach has already been discussed by Garfinkel [20] and it has several disadvantages. It is possible that a decoder is able to decode wrong permutations. This fact depends on the error resistance of the decoder. It is possible that a header and a non-header fragment from different multimedia files can be decoded because their internal structures which describe the video format fit syntactically together.

Further, depending on the total number of header and non-header fragments in the investigated image, the total number of combinations can be tremendous. Equation 2 shows the formula to calculate the number of permutations with a given number of header (H) and non-header (F) fragments. To calculate the complexity of this approach, the number of header and non-header fragments is simplified as n

(Equation 3). The complexity of this approach is then factorial (Equation 4).

$$Num_P = H * \sum_{n=1}^{F} \frac{F!}{n!} \tag{2}$$

$$\left.\begin{matrix} H \\ F \end{matrix}\right\} n \tag{3}$$

$$Num_P = n * \sum_{i=1}^{n} \frac{n!}{i!} = n * n! \sum_{i=1}^{n} \frac{1}{i!} \cong n * n! * k; 1 \leq k \leq e \Rightarrow O(n!) \tag{4}$$

Therefore graph-based reassembly algorithms have been developed. The complexity and implementation considerations of different graph-based assembly approaches have been discussed in detail by Pal and Memon [26]. Figure 3 shows the algorithm which has been chosen for the reassembly of multimedia files: it is based on the "Greedy Parallel Unique Path" (Greedy PUP) approach.

The following steps give an example of this procedure. In the first steps (A and B) the header and non-header fragments data structures are sorted. Header fragments are assigned to be the "heads" of the files which are intended for recovery. Sorting the fragments allows to manage all fragments in one data structure. In the next step (C), based on calculated weights between remaining fragments, only the best matching result (H_1F_2) is counted to the final reassembly order. The head of the first file "Head 1" is now changed to non-header fragment F2 (D). Then the same procedure takes place again. Only "heads" are compared to non-header fragments, the best match is found between non-header fragments F2 and F3. In the last run (E), the best match is found between "Head 2" and non-header fragment F1. Step (F) shows the final reassembly order.

The total number of comparisons can be calculated as shown in Equation 5. H refers to the number of header and F to the number of non-header fragments. As a performance improvement we recommend to cache previous comparison results for future use.

$$Num_G = \sum_{n=1}^{F} H * n \tag{5}$$

According to Pal and Memon [15] the Smartcarving procedure consists of three steps with different complexity: preprocessing, collation and reassembly. During preprocessing all blocks are sequentially checked for their relevance therefore the complexity results to $O(n)$. In contrast to our approach, weights are determined beforehand between all fragments. Therefore the corresponding complexity rises to $O(n^2)$. Sorting (which is $O(n \log n)$) these weights n times takes $O(n^2 \log n)$. The overall complexity is dominated by the most complex step and can be calculated as presented by Pal and Memon [26]. This is shown in Equation 6.

$$Num_G = O(n) + O(n^2) + O(n^2 \log n) \cong O(n^2 \log n) \tag{6}$$

Weighting The purpose of the comparisons during the reassembly process is the determination of weights which describe how well fragments fit together. The weights serve as a basis for determining the reassembly order with the help of graph-based algorithms with weighted edges such as the Greedy PUP algorithm [26]. The process and algorithm of calculating the weights depends on the file format of the files which should be recovered by the file carver. Fragments from multimedia files can be decoded in case the movie file header is available for the decoder. The movie file header contains general information such as encoding options, color information or image dimensions which is necessary to initialize decoders properly. In case of our multimedia file carver we tried to decode non-header fragments with

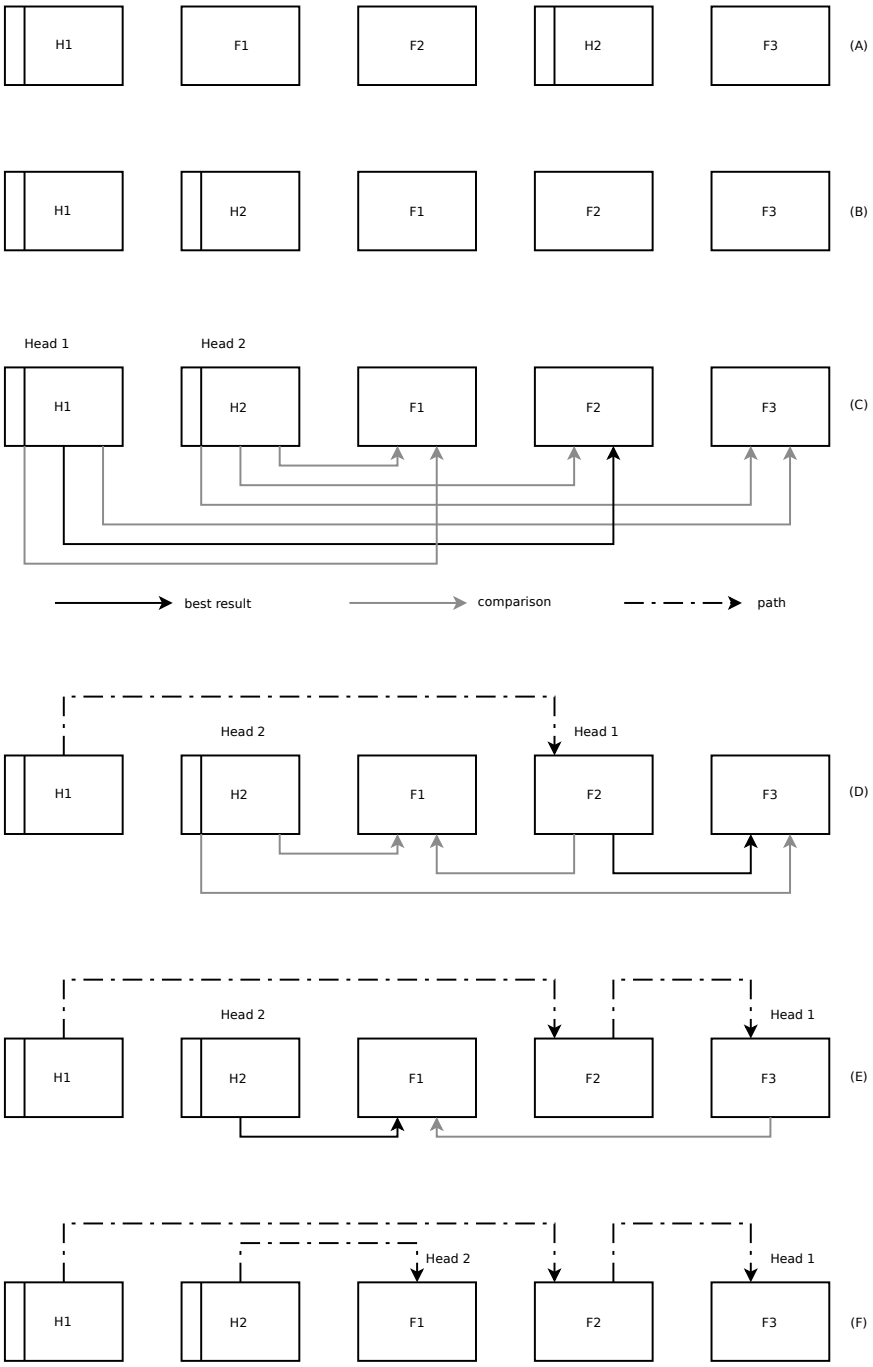

Figure 3: Greedy Parallel Unique Path as presented by Pal and Memon [26]

header-information from available header fragments until the decoder returned the first decoded frame. In case of missing header information, some video formats, such as H.264 [45], provide standard headers which are recommended for typical day to day scenarios.

Several different approaches for recovering digital images and calculating weights between identified fragments have been evaluated for their applicability and implemented in existing file carving programs. Weighting algorithms presented depend on characteristics of the JPEG file format ([28], [46], and [47]) such as the Define Huffman Table (DHT) segments and restart markers. The following characteristics

are considered when matching fragments from multimedia files with our file carver:

- Image resolution: the dimensions (height and width) of frames of fragments from multimedia files are compared. If they do not match frames are definitely from a different file as movie files never change their dimensions during playback.
- Histogram intersection: is a comparably (in terms of computational resources) cheap approach to determine, how similar images are. Pixels of frames at the boundaries of fragments are evaluated regarding their color value. For each color value there is a counter. The counters of color values of the different pictures are compared with each other. This requires the definition of a threshold to determine if counting values and thus the compared frames shall be considered as matching or not. This approach has shown unsatisfactory results as relative frames from fragments, that do not contain key-frames, have to be calculated from a common reference frame which results in indistinguishable outcomes. In case of missing key-frames, "ffmpeg" [48] uses frames that only consist of grey pixels (RGBA value 0x808080FF). This can be compared to calculating differential pictures: frames from different fragments are compared by calculating differential pictures, e. g. by subtracting color values for each single pixel. In case of similar pictures, the differential picture is a pure black picture. The more different images are, the brighter is their difference picture. Figure 4 shows a real world example from our file carver. In (A) and (B) a frame that has been extracted from the end of a fragment is subtracted from frames from the beginning of other fragments. In this case the togetherness of frames from different fragments could be determined correctly as the difference of scenario (A) results in a almost only black picture.

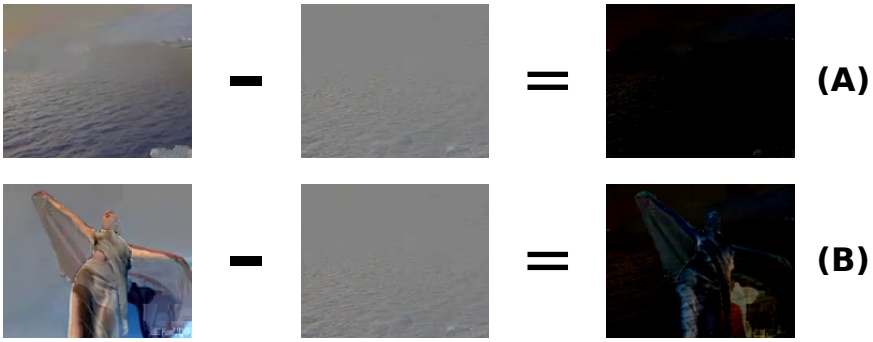

Figure 4: The difference of frames from different movie files.

The following characteristics are considered for future implementations of our file carver. These approaches, again, evaluate characteristics that are optimized for determining weights between fragments from different multimedia files:

- Datarate per fragment: tools such as "ffmpeg" [48] allow to decode fragments from movie files as long as the file header is available. The datarate per file fragment can be calculated by dividing the size of the file fragment by the number of fragments that could be extracted from it. This approach is expected to work only suboptimal under certain circumstances. In case of cuts within a movie it is possible that the datarate per file fragment changes abruptly.
- Frequency analysis: key-frames are transformed to the frequency domain before their comparison. After their transformation the frames are compared by using normalizing algorithms such as the "Zero Mean Normalized Sum of Absolute Differences" [49]. The difference to the histogram

intersection approach is, that it allows to detect structural differences (e. g. edges or contours) between the different fragments [50].

- Motion vectors: Modern video codecs store relative information between the different frames of a movie, the so called motion vectors [45]. Except for scene changes, the differences of these motion vectors between subsequent frames can be expected to be of only subtle difference. Tools such as "mplayer" [51] allow for extracting this information from MPEG visual scenes. The deviation of frames from different multimedia file fragments can be calculated based on the difference of their motion vectors.

- Combinations of previous approaches: different approaches presented before can be combined to obtain results which reflect the togetherness of file fragments with high probability. In this case a weighting of used weighting algorithms has to be established.

It is possible that the amount of recovered files is too high to be processed by humans. Therefore, in the following section, approaches for post-processing recovered files are presented.

3.5 Post-processing

Post-processing recovered multimedia files is performed for several reasons. Different cases such as child pornography, financial crime or the evaluation of moving pictures from surveillance cameras demand machine supported processing of data.

Deselaers et al. [52] presented a method for the classification of images into different levels of pornographic content. Originally it was intended for filtering network traffic from pornographic images. The approach was based on the bag-of-visual-words (BOVW) approach. Images are presented as a histogram of visual words. The vocabulary of this classifier is learned during the training phase from a task-specific database. This approach has been extended by Jansohn et al. [53] to support the classification of multimedia files based on both content analysis from key-frames (using BOVW and skin-tone detection algorithms) and the analysis of motions shown in these files. The motion analysis has been performed with two different approaches: periodicity detection and motion histograms. The authors concluded that repetitive motion detection deserves more research because of its robustness.

Another approach of classifying content is "fuzzy hashing". Several publications ([54], [55], [56]) have shown advances in this field recently. Hashsums are calculated independent of the carrier medium (e. g. PNG or JPG files). Thus it is possible to detect images even if they are stored in a different format (e. g. recoding) on a suspect's computer. The hashsums of known images are stored in a database during the training phase to be detected during the classification phase.

4 Conclusion and Outlook

In this paper we presented the current state-of-the-art in file carving. Furthermore, we discussed the applicability of techniques for the recovery of digital images to the field of multimedia files. Our findings have been implemented as a prototype of a multimedia file carver. It can be downloaded as open-source software from our homepage[1].

Currently the SmartCarving approach gives the best recovery results for fragmented files. It consists of three consecutive steps: preprocessing, collation and reassembly. During the first step, existing file system structures are evaluated to exclude data areas that are referenced through them. In the second

[1] `http://www.digitalforensics.at`

phase, unallocated or unreferenced data blocks are classified according to their file type to find the fragments of files that should be recovered using the file carver. In the last step, the reassembly, identified fragments are assembled in their original order.

In our paper we presented suitable approaches for all three steps. During the preprocessing step we discussed the application of tools that are able to interpret file system structures. During the collation phase, the Normalized Compression Distance (NCD) is utilized to distinguish blocks from text-based file formats from blocks containing data with high information entropy. Our reassembly algorithm is based on the Greedy Parallel Unique Path (PUP) algorithm known from the field of graph theory. It is an algorithm with weighted edges. The reassembly order is determined by calculating the best matching fragments in several runs. We also presented different characteristics that can be utilized to determine weights between different fragments. Mainly these characteristics are related to the field of digital image processing.

Further, we suggested the introduction of a fourth step, the post-processing step. Within this step recovered files are checked for their relevance in the related investigated case. We proposed the application of techniques from the fields of content classification as well as fuzzy hashing. Techniques from the latter field allow the detection of multimedia content even if it has been recoded into another format.

In the field of file carving open questions remain for the identification of file types of fragments as well as for weighting functions which determine how well fragments fit together. Current research projects deal with file type identification of fragments using algorithms from the field of computer vision. In future projects we want to integrate techniques that consider contextual information around file fragments (e. g. by implementing the SHT-PUP approach). Also, more research has to be conducted for the reassembly of text-based documents, such as e-mails or office documents.

Another open research issue is the performance of file carvers for the recovery of fragmented files. With the introduction of cloud computing setups, computation time has become an increasingly dynamic thing. In case more resources are needed, additional machines can be integrated into the currently used setup. The main challenge for the deployment of file carving techniques to cloud infrastractures is the data exchange between participating nodes. In the future we want to evaluate different data exchange models to make file carving applicable to todays harddisks with capacities in the multi terabyte range.

5 Acknowledgements

We would like to thank Vasileios Miskos for his research in the field of file fragment classification as well as for his support during the development of our multimedia file carver as partner in our extreme programming teams.

References

[1] R. Poisel and S. Tjoa, "Roadmap to approaches for carving of fragmented multimedia files," in *Proc. of the 4th International Workshop on Digital Forensics (WSDF'11), Vienna, Austria*. IEEE, August 2011, pp. 752–757.

[2] J. Beniger, *The Control Revolution: Technological and Economic Origins of the Information Society*. Cambridge: Harvard University Press, 1989.

[3] C. Walter. (2005, July) Kryder's law. http://www.scientificamerican.com/article.cfm?id=kryders-law. Scientific American. [Online; Status: October 15^{th} 2011].

[4] Gartner/Reuters, "Computers in use pass 1 billion mark: Gartner," http://www.reuters.com/article/2008/06/23/us-computers-statistics-idUSL2324525420080623, June 2008, [Online; Status: March 29^{th} 2011].

[5] BBC News, "Over 5 billion mobile phone connections worldwide," http://www.bbc.co.uk/news/10569081, July 2010, [Online; Status: April 15^{th} 2011].

[6] German Federal Ministry of the Interior, "Polizeiliche Kriminalstatistik 2010," http://www.bmi.bund.de/SharedDocs/Downloads/DE/Broschueren/2011/PKS2010.html, May 2010, [Online; Status: October 6th 2011].

[7] McAfee Avert Labs, "French Authorities Talk Up Digital Investigations," http://process-info.org/news/security-news-blogs/archive/2010/10/french-authorities-talk-up-digital-investigations, October 2010, [Online; Status: March 29th 2011].

[8] A. Pal, H. T. Sencar, and N. D. Memon, "Detecting file fragmentation point using sequential hypothesis testing," *Digital Investigation*, vol. 5, no. Supplement 1, pp. S2 – S13, 2008.

[9] ISO, *ISO/IEC 14496-12:2008: Information technology — Coding of audio-visual objects — Part 12: ISO base media file format*. Geneva, Switzerland: International Organization for Standardization, 2008. [Online]. Available: http://standards.iso.org/ittf/PubliclyAvailableStandards/c051533_ISO_IEC_14496-12_2008.zip

[10] W3C. (2011, August) HTML5 - A vocabulary and associated APIs for HTML and XHTML. http://dev.w3.org/html5/spec/Overview.html. W3C. [Online; Status: August 9th 2011].

[11] Zencoder, "Over 50% of web users now support html5 video," http://videojs.com/2011/01/html5-video-statistics/, January 2011, [Online; Status: August 9th 2011].

[12] S. Pfeiffer, *The Definitive Guide to HTML5 Video*. Apress, December 2010.

[13] S. V. Nagaraj, *Web Caching And Its Applications (Kluwer International Series in Engineering and Computer Science)*. Norwell, MA, USA: Kluwer Academic Publishers, 2004.

[14] M. I. Cohen, "PyFlag – an advanced network forensic framework," *Digital Investigation*, vol. 5, Supplement, pp. S112—-S120, September 2008.

[15] A. Pal and N. D. Memon, "The evolution of file carving," *IEEE Signal Processing Magazine*, vol. 26, no. 2, pp. 59–71, March 2009.

[16] B. Carrier, *File System Forensic Analysis*. Addison-Wesley Professional, 2005.

[17] M. Breeuwsma, M. D. Jongh, C. Klaver, R. V. D. Knijff, and M. Roeloffs, "Forensic data recovery from flash memory," *Small Scale Digital Device Forensics Journal*, vol. 1, no. 1, pp. 1–17, June 2007.

[18] B. Carrier, "The Sleuth Kit (TSK)," http://www.sleuthkit.org/sleuthkit/, [Online; Status: April 15th 2011].

[19] G. G. Richard and V. Roussev, "Scalpel: A frugal, high performance file carver," in *Proc. of the 5th Annual Digital Forensic Research Workshop (DFRWS'05), New Orleans, Los Angeles, USA*, August 2005, pp. 1–10.

[20] S. L. Garfinkel, "Carving contiguous and fragmented files with fast object validation," *Digital Investigation*, vol. 4, no. Supplement 1, pp. 2–12, 2007.

[21] A. Castiglione, A. D. Santis, and C. Soriente, "Security and privacy issues in the portable document format," *Journal of Systems and Software*, vol. 83, no. 10, pp. 1813–1822, 2010.

[22] M. I. Cohen, "Advanced carving techniques," *Digital Investigation*, vol. 4, no. 3-4, pp. 119–128, 2007.

[23] M. Karresand and N. Shahmehri, "File type identification of data fragments by their binary structure," in *Proc. of the 2006 IEEE Information Assurance Workshop (IAW'06), West Point, New York, USA*. IEEE, June 2006, pp. 140–147.

[24] S. Axelsson, "Using normalized compression distance for classifying file fragments," in *Proc. of the 2010 International Conference on Availability, Reliability and Security (ARES'10), Krakow, Poland*. IEEE, February 2010, pp. 641–646.

[25] V. Roussev and S. L. Garfinkel, "File fragment classification-the case for specialized approaches," in *Proc. of the 4th IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'09), Berkeley, California, USA*. IEEE, May 2009, pp. 3–14.

[26] A. Pal and N. D. Memon, "Automated reassembly of file fragmented images using greedy algorithms," *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 385–393, 2006.

[27] M. Karresand, "Completing the picture: Fragments and back again," Master's thesis, Linkoepings universitet, 2008. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-11752

[28] M. Karresand and N. Shahmehri, "Reassembly of fragmented jpeg images containing restart markers," in *Proc. of the 2008 European Conference on Computer Network Defense (EC2ND'08), Dublin, Ireland*. IEEE, December 2008, pp. 25–32.

[29] B. Yoo, J. Park, S. Lim, J. Bang, and S. Lee, "A study on multimedia file carving method," *Multimedia Tools*

and Applications, pp. 1–19, 2011. [Online]. Available: http://dx.doi.org/10.1007/s11042-010-0704-y

[30] J. Metz and R. J. Mora. (2006, July) In analysis of 2006 dfrws forensic carving challenge. http://sandbox.dfrws.org/2006/mora/dfrws2006.pdf.

[31] R. Poisel and S. Tjoa, "Forensics investigations of multimedia data: A review of the state-of-the-art," in *Proc. of the 6th International Conference on IT Security Incident Management & IT Forensics (IMF'11), Stuttgart, Germany*. IEEE, May 2011, pp. 48–61.

[32] C. Moch and F. C. Freiling, "The forensic image generator generator (Forensig2)," in *Proc. of the 5th International Conference on IT Security Incident Management and IT Forensics (IMF'09), Stuttgart, Germany*. IEEE, September 2009, pp. 78–93.

[33] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten, "Lest we remember: cold-boot attacks on encryption keys," *Communications of the ACM*, vol. 52, pp. 91–98, May 2009.

[34] Microsoft. Default cluster size for ntfs, fat, and exfat. http://support.microsoft.com/kb/140365. [Online; Status April 12th 2011].

[35] A. Mathur, M. Cao, S. Bhattacharya, A. Dilger, A. Tomas, and L. Vivier, "The new ext4 filesystem: current status and future plans," in *Proc. of the 2007 Linux Symposium, Ottawa, Canada*, June 2007.

[36] C. J. Veenman, "Statistical disk cluster classification for file carving," in *Proc. of the 3rd International Symposium on Information Assurance and Security (IAS'07), Manchester, UK*. IEEE, August 2007, pp. 393–398.

[37] "file(1)," ftp://ftp.astron.com/pub/file/, [Online; Status: October 24th 2011].

[38] "PhotoRec," http://www.cgsecurity.org/wiki/PhotoRec, [Online; Status: April 15th 2011].

[39] J. Jeong, *The JPEG standard*. Hightstown, NJ, USA: McGraw-Hill, Inc., 1997. [Online]. Available: http://portal.acm.org/citation.cfm?id=275869.275888

[40] M. M. Shannon, "Forensic relative strength scoring: ASCII and entropy scoring," *International Journal of Digital Evidence*, vol. 2, no. 4, 2004.

[41] W. Li, K. Wang, S. J. Stolfo, and B. Herzog, "Fileprints: Identifying file types by n-gram analysis," in *Proc. of the 6th Systems, Man and Cybernetics: Information Assurance Workshop (IAW'05), West Point, New York, USA*. IEEE, June 2005, pp. 64–71.

[42] W. C. Calhoun and D. Coles, "Predicting the types of file fragments," *Digital Investigation*, vol. 5, Supplement, pp. S14–S20, September 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1742287608000273

[43] G. Conti, S. Bratus, A. Shubina, A. Lichtenberg, R. Ragsdale, R. Perez-Alemany, B. Sangster, and M. Supan. (2010, July) A visual study of primitive binary fragment types. White Paper, Black Hat USA 2010. [Online]. Available: http://www.rumint.org/gregconti/publications/taxonomy-bh.pdf

[44] N. Freed and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types," RFC 2046 (Draft Standard), Internet Engineering Task Force, November 1996, updated by RFCs 2646, 3798, 5147. [Online]. Available: http://www.ietf.org/rfc/rfc2046.txt

[45] I. E. G. Richardson, *The H.264 Advanced Video Compression Standard*, 2nd, Ed. Wiley, 2010.

[46] K. Mohamad and M. Deris, "Fragmentation point detection of JPEG images at DHT using validator," in *Proc. of the 1st International Conference on Future Generation Information Technology (FGIT'09), Jeju Island, Korea, LNCS*, vol. 5899. Springer-Verlag, December 2009, pp. 173–180.

[47] H. T. Sencar and N. D. Memon, "Identification and recovery of jpeg files with missing fragments," *Digital Investigation*, vol. 6, no. Supplement 1, pp. S88–S98, 2009.

[48] "FFmpeg," http://www.ffmpeg.org/, [Online; Status: April 20th 2011].

[49] P. A. T. Gockel, R. Dillmann, *Computer Vision*, , Ed. Elektor-Verlag, June 2007. [Online]. Available: http://amazon.com/o/ASIN/3895761656/

[50] W. Burger and M. J. Burge, *Principles of Digital Image Processing - Core Algorithms*, I. Mackie, Ed. Springer-Verlag, London Limited, 2009.

[51] "MPlayer," http://www.mplayerhq.hu/, [Online; Status: November 1st 2011].

[52] T. Deselaers, L. Pimenidis, and H. Ney, "Bag-of-visual-words models for adult image classification and filtering," in *Proc. of International Conference on Pattern Recognition (ICPR'08), Tampa, Florida, USA*.

IEEE, December 2008, pp. 1–4.

[53] C. Jansohn, A. Ulges, and T. M. Breuel, "Detecting pornographic video content by combining image features with motion information," in *Proc. of the 17th ACM international conference on Multimedia (MM'09), Beijing, China*. ACM, October 2009, pp. 601–604.

[54] V. Monga, "Perceptually based methods for robust image hashing," Ph.D. dissertation, The University of Texas at Austin, August 2005.

[55] X. X. Guo and H. Dimitrios, "Content based image hashing via wavelet and radon transform," in *Proc. of the 8th Pacific Rim Conference on Multimedia (PCM'07), Hongkong, China, LNCS*, vol. 4810. Springer-Verlag, 2007, pp. 755–764.

[56] C. Zauner, "Implementation and benchmarking of perceptual image hash functions," Master's thesis, University of Applied Sciences Upper Austria, 2010.

Rainer Poisel received a master's degree in computer science management from the Technical University of Vienna as well as a master's degree in telecommunications from St. Pölten University of applied sciences. He has been working as a scientific researcher in the field of information security since 2007. During his last research project he developed a file carver for the recovery of multimedia files from various storage media. Besides the publication of various papers for different security conferences such as D-A-CH Security, IEEE SIBIRCON, DeepSec and Chaos Computer Club he holds certifications for AccessData (ACE, PSA) and Cisco (CCNA) products.

Simon Tjoa received a master's degree in business informatics from the University of Vienna and is currently working on his PhD in the field of security. He has been working full time in the area of information security since 2006. At present he is research scientist and lecturer at the St. Pölten University of Applied Sciences. His research interests include digital forensics, business continuity management and business process security. He participates as reviewer in various program committees in the field of security such as D-A-CH Security, ARES conference (International Conference on Availability, Reliability and Security) or the IFIP ISM conference (Twelfth Annual IFIP Workshop on Information Security Management). He furthermore is in the organizing committee of security related international workshops such as the International Workshop on Visualization and Information Security Management or the International Workshop on Organizational Security Aspects. He is currently the secretary of IEEE SMC Austria and holds professional security certifications such as AMBCI, ACE, CISA, CISM.

Paul Tavolato received his master's degree and PhD from the University of Technology in Vienna. During his professional career he has worked on research projects and on industrial software development projects in the academic area as well as in private enterprises. Currently he is a research scientist and lecturer at the IT Security Department of the St. Pölten University of Applied Sciences. His research interests include software engineering, malware analysis, and theoretical questions in these fields.