J. Internet Comput. Serv.
ISSN 1598-0170 (Print) / ISSN 2287-1136 (Online)
http://www.jics.or.kr

Block based Smart Carving System for Forgery Analysis and Fragmented File Identification

이 한 성[1] 이 형 우[1]*
Hanseong Lee Hyung-WooLee

요 약

디지털 범죄 수사의 전 단계에 걸쳐 획득된 자료가 증거 능력으로 인정 받을 수 있기 위해서는 법적/기술적 요구사항을 만족하여야 한다. 본 논문에서는 파일 시스템에서 기본적으로 제공하는 정보에 의존하지 않고, 저장장치 디스크 내부의 비할당 영역을 블록 단위로 스캔/검사하여 파일을 자동 복구하여 디지털 포렌식 증거 자료로 확보하는 메커니즘을 제시하였고 이를 직접 SW로 구현하였다. 제시한 기법은 분석 대상 시스템의 RAW 디스크 데이터에 대해 운영체제에서 제공하는 파일 시스템 관련 정보를 참조하지 않으면서 디스크 내에 저장된 각종 파일의 저장 포맷/파일 구조에 관한 정보를 토대로 512 바이트 블록 단위로 검사/분석하는 파일 카빙 과정을 구현하였으며, 저장 장치 내에 삭제되거나 손상된 파일을 지능적으로 복원하는 Smart Carving 메커니즘을 제시하였다. 구현한 기법을 이용할 경우 디지털 포렌식 분석 과정에서 시스템 내부에 저장된 파일에 대한 위변조 여부를 지능적으로 판별할 수 있는 블록 기반 스마트 카빙 기능을 제공한다.

☞ 주제어 : 파일 카빙, 스마트 카빙, 파일 시스템, 블록, 단편화, 위변조 분석, 디지털 포렌식.

ABSTRACT

In order for data obtained through all stages of digital crime investigation to be recognized as evidence capability, it must satisfy legal / technical requirements. In this paper, we propose a mechanism and implement software to provide digital forensic evidence by automatically recovering files by scanning / inspecting the unallocated area inside the storage disk block without relying on information provided by the file system. The proposed technique checks / analyzes the RAW disk data of the system under analysis in 512-byte block units based on information on the storage format / file structure of various files stored on the disk without referring to the file system-related information provided by the operating system. The file carving process was implemented, and a smart carving mechanism was proposed to intelligently restore deleted or damaged files in the storage device. As a result, we have provided a block based smart carving method to intelligently identify fragmented and damaged files in storage efficiently for forgery analysis on digital forensic investigation.

☞ keyword : File Carving, Smart Carving, File System, Block, Fragmentation, Forgery Analysis, Digital Forensics.

1. Introduction

File carving can be described as a process used in computer forensics to extract data from a disk drive or other external/internal storage device without the assistance of the file system that originality created the file. Therefore, file carving is a great method for recovering files and fragments of files when directory entries are corrupt or missing. This is especially used by digital forensics experts in criminal cases for recovering several evidences. As a result, it is a method that recovers files at unallocated space without any file information and is used to recover data and execute a digital forensic investigation. As a forensics technique that recovers files based merely on file structure and content and without any matching file system meta-data, file carving is most often used to recover files from the unallocated space in a drive. Unallocated space refers to the area of the drive which no longer holds any file information as indicated by the file system structures like the file table. In the case of damaged or missing file system structures, this may involve the whole

[1] Div. of Computer Engineering, Hanshin University, Gyeon-gi(Osan), 18101, Rep. of Korea.
* Corresponding author (hwlee@hs.ac.kr)
[Received 28 January 2020, Reviewed 26 February 2020(R2 20 April 2020), Accepted 6 May 2020]
☆ A preliminary version of this paper was presented at APIC-IST 2019.
☆ This work was partially supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF) (NRF-2017R1D1B03035040).

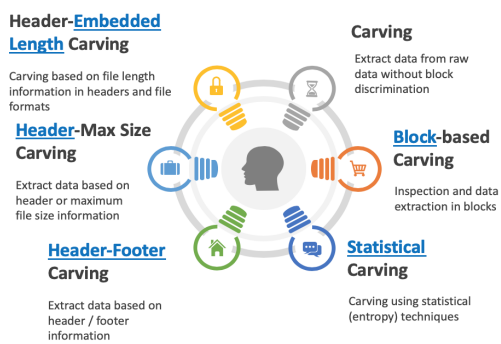
drive. File carving is the process of reconstructing files by scanning the raw bytes of the disk and reassembling them. Digital forensics is an important field of cybersecurity and digital crimes investigation. It entails applying file recovery methods to analyze data from storage media and extract hidden, deleted or overwritten files. The recovery process might have accompanied by cases of unallocated partitions of blocks or clusters and the absence of file system metadata. These cases entail advance recovery methods that have carving abilities. The file carving methods include different types of techniques to identify, validate and reassemble the file. In the process of digital crime investigation, it is necessary to dump memory and its process in order to acquire evidence of intrusion into computer system by hacker or malicious program, and it is necessary to dump memory and process, and to obtain volatile information such as network and its port information. In addition, static non-volatile information such as deleted and damaged file systems and application artifacts should be provided as evidence. Therefore, "*File Carving*" [1] process for damaged files should be performed along with recovery of deleted files in storage devices cooperated both with the signature analysis and cryptanalysis. In this paper, we propose a mechanism to scan and inspect unallocated areas in a storage device in case the file allocation information provided by the file system can not be used. The proposed technique checks / analyzes the RAW disk data of the system under analysis in 512-byte block units based on information on the storage format / file structure of various files stored on the disk without referring to the file system-related information provided by the operating system. The file carving process was implemented, and a smart carving mechanism was proposed to intelligently restore deleted or damaged files in the storage device. As a result, we provided a smart carving [2] method that intelligently restores deleted and damaged files in the storage device efficiently.

2. File Carving Mechanism

2.1 File Recovery and File Carving

All filesystems contain some "*metadata*" that describes its specific file system structure, for example the hierarchy of folders and files, with names for each and the physical address on the disk where the file is stored. File carving is the process of trying to recover files without this filesystem structure. This is done by analyzing the raw data usually looking for specific sequences of bytes in file headers or footers: these bytes used for file identification are named "magic numbers". For example, a jpeg image file begins with "0xFFD8" and ends with "0xFFD9" and a Java class file has as its first four bytes the hexadecimal value **CA FE BA BE** as a magic number.

In order to efficiently manage the data in the storage device, the file system records various information including the creation, modification and access time of the file in the form of metadata. Therefore, those metadata can be used during file recovery process. However, when we couldn't receive help or reference to internal metadata in the file system, we should have to restore all or some of the deleted files directly just depending on its unique information such as signature, logical file structure information on digital forensic [3]. In this case, file carving method should be applied. File carving technique can be divided into several detailed techniques as shown below Fig. 1. Unlike the data recovery method based on the information provided by the file system, the file carving method uses the format and the standard information of each digital file without the help of the file system[4].



(Figure 1) Classification of File Carving Technology

In order to successfully perform file carving in an environment where the metadata information provided by the file system can not be used, the file system must be checked in 512-byte block units. In order to do this, it is necessary to

precede the process of correctly identifying (file identification) each blocks on which part it corresponds to within the file as an AI assistant digital forensics[5].

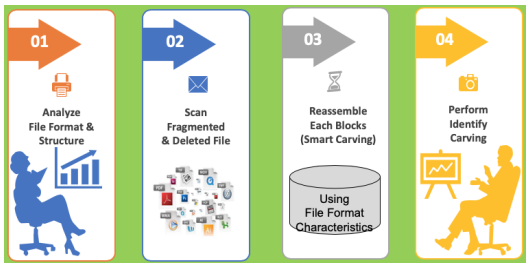
2.2 File Carving Categorizes〔6〕

Generally, there are three categories of file carving techniques. The categorization is depending on the complexity of the recovery cases as described in [7]. The three categorize are listed in low to high complex ascending order as follows:

- **Signature based carving**: it works by searching in dataset for the patterns that mark the beginning of a file (header) value (like FFD8 for JPEG files) and then looks for the first occurrence of the (footer) value (like FFD9 for JPEG files). All data clusters between the header and footer values are carved in an output file. Once the header value and end locations of a file are calculated, then all data clusters between these values are carved in an output file.
- **Structure based carving**: It is also sometimes referred as “Semantic carving” or “Deep carving”. File structure-based carving first identifies a certain level of information in a file format. This information is matched in the raw data set to identify the file. This type uses the information of the internal file structure to carve a few fragmentation cases by reducing false positives of carving fragmented file.
- **Content-based carving**: The main idea behind cluster content-based is to read each individual cluster in the dataset and then analyze its contents to find out some relationships between the clusters that belong to a particular file. It calculates metadata information like character counts or statistical information over the bytes of the clusters. These clusters are later reassembled to recover the original file.
- Based on these categories, **smart carving** is used to carve out files which is divided into many fragments. It can work on fragmented and non-fragmented data by using preprocessing for data clusters, classifying of cluster to various file types and reassembly the block in sequences that match their file type.

2.3 File Carving on Unallocated Block

If a file is deleted, the data and metadata remain, and if the data has not been overwritten with another file, you can recover the deleted or some damaged file by browsing the allocation area recorded in the metadata. Therefore, it is necessary to restore the file by applying the file carving method because it does not receive the help of the metadata. In this paper, we performed extraction and verification of semantic information and restoration of fragmented blocks using only the format and file structure information of each file without referring to metadata as shown in the figure below. As a result, a smart carving mechanism for analyzing the blocks in the unallocated area is proposed.



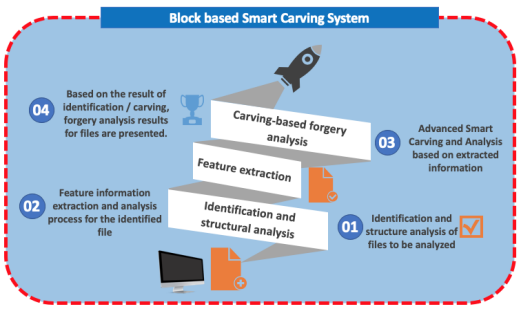
(Figure 2) File Identification based Smart Carving

3. Block Based Fragmented File Smart Carving Mechanism

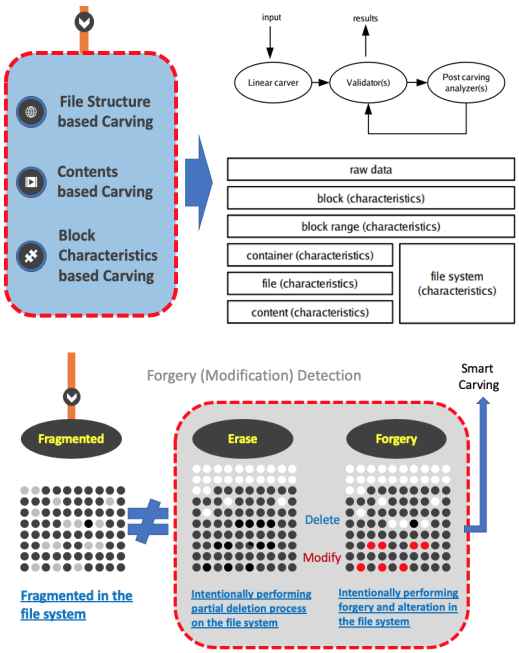
3.1 Block based File Carving

When a disk image is input, the process of inspecting each block is performed, and a process of identifying each file corresponding to each block is performed. Then, the file carving process is performed based on unique format information of each file. Most files are larger than the block size, so they are fragmented into several blocks within the disk. Therefore, we carried out complicated and difficult identification and smart carving process as below Fig. 3.

In order to distinguish the files generated by the SWs in the file system, most SWs embed their signature in various parts, such as the header and footer of the generated file. In addition, for quick execution, data is divided into several areas and the signatures are attached to each area. Therefore,



(Figure 3) Proposed Smart Carving Process



(Figure 4) Smart Carving Procedure for Block based Fragmented File

in the file carving process, the file format information unique to each file is utilized, and each block is identified by using the signature information recorded in the file. It is reassembled and restored to the original file.

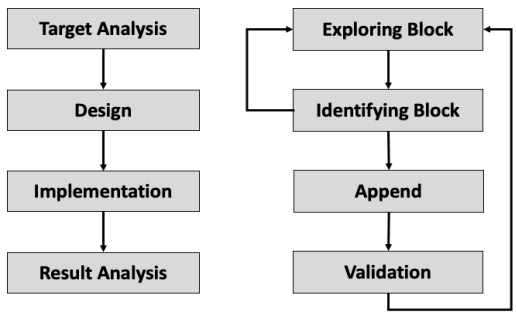
3.2 Fragmented File Smart Carving

In order to carve a specific file deleted in the storage device, the signature of the corresponding file must be scanned for all the areas of the storage device on a block-by-block basis. The magic number part recorded in the header part is searched in first, and all the blocks including the same signature as the file to be searched can be found. Finally, the file is reconstructed by recombining based on the association and semantic structure between the found blocks. However, it is necessary to limit the search target to the deleted file in the unallocated area because it may be a task that requires a very long time depending on the capacity to examine the entire area of the storage device. Therefore, it is possible to recover the damaged file faster than the existing technique by performing the search process on the signature information based on the characteristics of the unique format information of each file with respect to the unassigned area.

4. Proposed Smart Carving System Structure for Forgery Detection

4.1 Smart Carving Mechanism

In addition to the fact that data is overwritten and not recoverable, there is a major problem with file carving operations. If the process of creating / modifying / deleting a file is repeated, the file is fragmented, and large and small unallocated areas are formed in the middle of the block in the storage device. Therefore, when carving a fragmented file, it is possible to determine the block belonging to which file by analyzing the association with the data in the block.



(Figure 5) Proposed Smart Carving Process

However, if you do not get the help of metadata, if you have (1) data from another file in the middle of the file, or

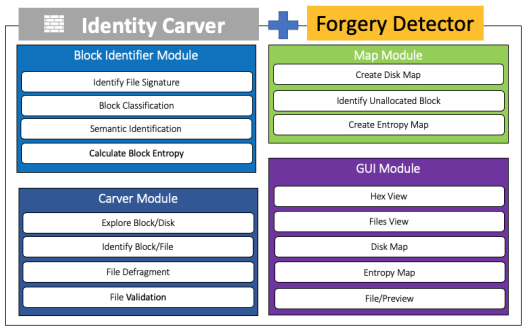
(2) the footer signature portion that indicates the end of the file is deleted, It may be difficult to specify. In order to solve this problem, in this study, we used (1) a method which can estimate / judge which block of a file the block obtained in the unallocated area corresponds to, based on the *file format information* inherent in the individual file. In addition, (2) the *entropy value* for each block is calculated, and it is correctly determined that each block is a block corresponding to a part of a file.

4.2 Design and Implementation

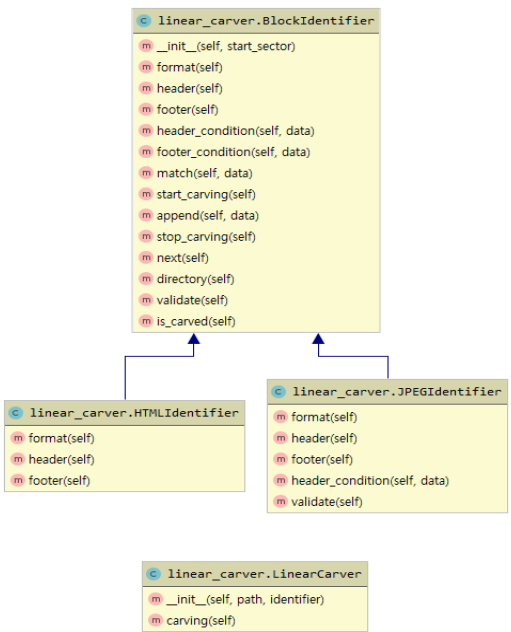
The software architecture, which provides file identification and smart carving functions within a disk image, consists of four modules: Block Identifier, Map, Carver, and GUI. Among these, Block Identifier as shown Figure 6. And Carver module are the core functions of the software.

The Block Identifier module identifies each block by providing file signature identification, block classification, and linguistic identification. The Carver module performs block and disk search, block and file identification, file combinations, and file verification. The Map module collects disk information, turns each block into a disk map, provides the ability to identify allocated and unallocated areas, and calculates the entropy of each block. The GUI module is a user interface that shows disk images and carving results. The GUI module provides a hex-view function to view the hex-values of the disk and a file view function to view a list of files extracted during the carving process. It also provides a preview of the disk map, entropy map, and stored files created in the Map module. The GUI module allows the user to perform disk image analysis, carving, and access to the files stored in the Identity Carver software.

Figure 7 shows a block diagram of some of the Block Identifier and Carver modules. The *BlockIdentifier* class is an abstract class that performs common functions for identifying signatures for a block. In this paper, we implement the block identifiers of HTML and JPEG files. In order to implement the smart carving technique for two identifiers, we inherited the BlockIdentifier abstract class and implemented it as *HTMLIdentifier and JPEGIdentifier* classes. *LinearCarver* class inherits BlockIdentifier class and uses HTMLIdentitifer and JPEGIdentifier class to get each block from disk image to identify blocks and perform carving.



(Figure 6) Module Diagram of IDentity Carver SW



(Figure 7) Class diagram of Block Identifier and Carver module

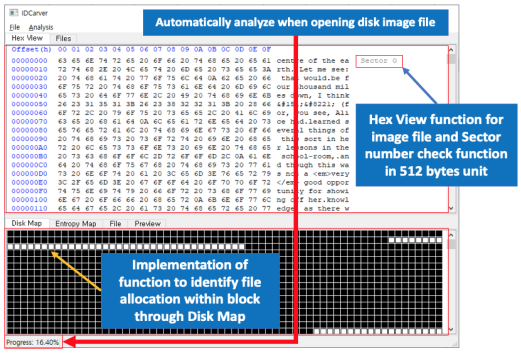
Figure 8 shows the class implementing the block-based HTML identification and JPEG smart carving modules. After inheriting the BlockIdentifier class, the detailed function is implemented by overriding the format, header, footer, and validate methods. The Format method corresponds to the file's extension. When we format a carved block into a file, we specify the extension by using the format method. The

header and footer methods return a list of Python to specify the specific signature within the block. File signatures are considered for scalability so that they can be added later. If the validate method finds a footer, it joins the block and verifies that the restored file is correct and returns the result. The JPEGIdentifier class implements forgery analysis and verification process by checking whether the archived file is normally opened using Python's image library, Python Imaging Library (PIL)[8]. In the case of JPEG block identifier, the header_condition method is added after strict condition on the header identifier. Through this class implementation, we implemented smart carving software that provides file identification and forgery verification function based on block.

```
class HTMLIdentifier(BlockIdentifier):
    def format(self):
        return "html"

    def header(self):
        return [b'\x3C\x68\x74\x6D\x6C']  # <html

    def footer(self):
        return [b'\x3C\x2F\x68\x74\x6D\x6C']  # </html


class JPEGIdentifier(BlockIdentifier):
    def format(self):
        return "jpg"

    def header(self):
        return b'\xff\xd8\xff'  # SOI

    def footer(self):
        return [b'\xff\xd9']    # EOI

    """
    Adjust the strict condition
    """
    def header_condition(self, data):
        return data[:3] == self.header()

    def validate(self):
        try:
            img = Image.open(self.full_path)
            img = None
        except:
            return False
        return True
```

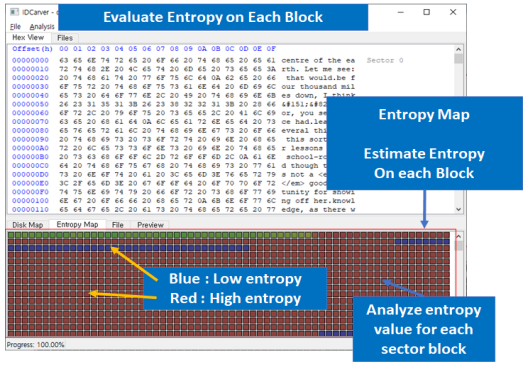
(Figure 8) Source code for the HTMLIdentifier and JPEGIdentifier classes

The file identification and smart carving function inside the disk image was implemented using the Python language on Windows 10 OS. As shown in the following Figure 9, (1) file carving function is performed based on block information based on file format information, (2) entropy is calculated for each block in order to perform efficient and intelligent carving process on fragmented files, We have provided a method to increase the accuracy and minimize errors in the process.



(Figure 9) Implementation of Block Entropy based Smart Carving SW

Figure 10 shows the result of implementing the entropy map in software. The entropy map is a visualized heat map of the entropy values of each block. Each block in the entropy map corresponds to one sector in storage. Blocks in blue indicate low entropy values, and red blocks indicate high entropy values. Green, which is halfway between blue and red, means the median of the total entropy values. By visualizing the entropy values of each block, digital forensics can identify contiguous blocks with similar entropy values and provide faster block-by-block analysis.

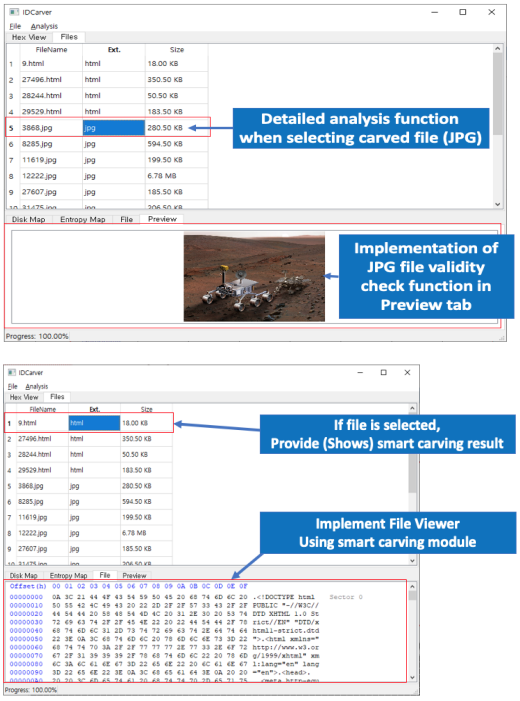


(Figure 10) Entropy Map of IDentity Carver SW

4.3 Experimental Results

The performance of the smart carving process is shown in

the figure below. In the process of re-merging fragmented files in block unit, it is confirmed that JPG file in disk image is extracted / restored automatically as shown in Figure 11.

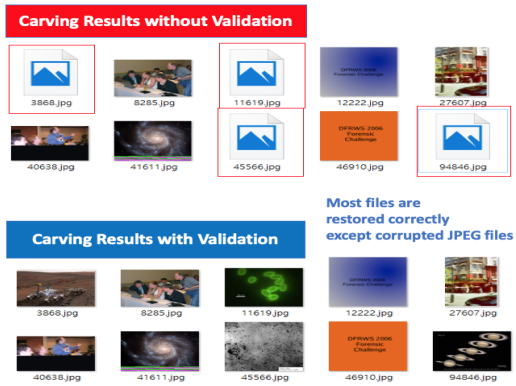


(Figure 11) Implementation of Validity Check Procedure on Smart Carving

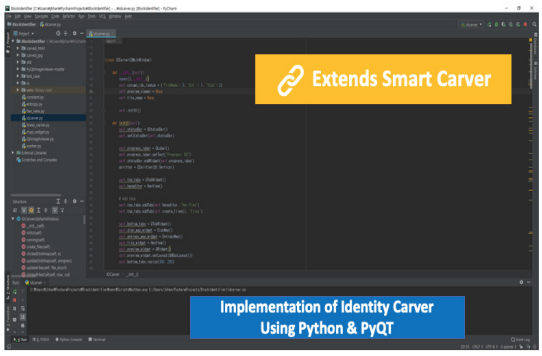
In addition, when the verification module is additionally applied to automatically check whether the restored file is restored through the smart carving method, it is possible to correctly identify each block in the html file and the JPG file as shown in the figure below, we could confirm that it was restored to the original file correctly.

4.4 Analysis and Discussion

Table 1 compares the features of Foremost [9], Scalpel [10], Bulk extractor [11] [12], and *IDentity Carver* software developed in this paper. The Foremost tool is an open source carving tool written in C. The tool supports Linux and Mac OS X except Windows. The Scalpel tool is an open source carving tool implemented in C ++. It is similar in function to Foremost tool and has the advantage of supporting Windows operating system and multi-threading function not supported by Foremost. Both Foremost and Scalpel tools can add a Recover File Type in the configuration file. So, in addition to the existing supported file types, other file types can be added and restored by the user in Figure 13.



(Figure 12) Smart Carving Result with (or) without Validation Procedure



(Figure 13) Implementation of Smart Carving System

Both tools will carve based on the file's header and footer, and the maximum file size can be specified depending on the file type. After the carving is complete, the restored files are stored on the system. There are three more supported data types for Scalpel except for C ++ source code in both software. The bulk extractor tool is an open source carving tool implemented in C ++. Among the four carving tools, there are many functional differences. One difference between the bulk extractor and the other three tools is that the Recover Data is different. Bulk extractor is a carving tool, but it

Table. 1. Feature Comparison by Carving Tools

Feature		Foremost [9]	Scalpel [10]	Bulk extractor [11] [12]	IDentity Carver
Source Code	Language	**C**	**C++**	**C++**	**Python**
	Open Source	○	○	○	○
OS Support	Linux	○	○	○	○
	Mac OS X	○	○	○	○
	Windows	×	○	○	○
Configuration Requirement	File Type (Header/Footer)	○	○	×	×
	Options	×	×	○	×
	Tuning Parameters	×	×	○	×
Supported Interface	CLI (Command Line Interface)	○	○	○	×
	GUI (Graphic User Interface)	×	×	○	○
Input Data	Raw Block	○	○	○	○
	Disk Image	○	○	○	○
	Raw Device	×	×	○	×
	Directory Files	×	×	○	×
Carving Method	Header and Footer Carving	○	○	○	○
	Limit Max Size	○	○	×	×
	Pattern Matching	×	×	○	×
	Validation	×	×	○	○
	Smart Carving	×	×	×	○
Output Files	Feature Files	×	×	○	×
	Recover Files	○	○	×	○
Additional Support	Multi-threading	×	○	○	×
	Plugins	×	×	○	×
	Disk Map	×	×	×	○
	Entropy Map	×	×	×	○
	Recover File Preview	×	×	×	○
	Forgery Analysis	×	×	×	○

extracts features of a specific file rather than a file. For example, an email has a feature of an email address, and a JPG image file has EXIF data. Instead of carving the file, it extracts the features and location of the file. And unlike other tools, you can select options when tuning and parameter tuning. You can select Raw Device or Directory Files as well as Raw Block and Disk Image. The bulk extractor uses both header and footer carving and pattern matching. Data that failed verification by verifying the carving result is not output. Each feature has a verification algorithm. Bulk extractors are plug-ins that extend the functionality of your program. And unlike traditional tools, Bulk extractor allows files to recover data such as email addresses, URLs, and credit card numbers. The *IDentity Carver* tool implemented in Python in this paper provides a cross-platform and GUI-based carving method using Qt. Carving is performed based on headers and footers, and additional verification can be performed. And what makes the *IDentity Carver* tool different from other carving tools provides improved verification, provides forgery and analysis for smart carving-based restored files, the ability to analyze disk images and entropy maps, and restore file previews. View function is supported on GUI. Two formats, JPG and HTML, are supported for restorable files, but in the future, software scalability is considered to be able to restore even more file formats.

5. Conclusions

In case of digital crime investigation, it is necessary to use file recovery and file carving process in order to acquire digital evidence from volatile and non-volatile information. Therefore, "*File Carving*" process should be performed along

with recovery of deleted files in storage devices. In detail, we used a method which can estimate / judge which block of a file the block obtained in the unallocated area based on the *file format information* inherent in the individual file. In addition, we used *entropy value* calculated for each block to determine each block correctly corresponding to a part of a file. As experimental results, we can adopt verification module to check whether the restored file is restored through the smart carving method. And, we could confirm that it was restored to the original file correctly. And we compare the function of the tool implemented in this paper with the existing carving tool and the restoration data type. As a result of comparison, the tools implemented in this paper, unlike the existing tools, provide block-based reconstruction and verification functions for disk images, and forgery analysis functions. In addition, it is expected that the implemented technique can be used for detecting anti-forensic techniques [14] and forgery analysis processes [15] [16] for Android apps using blockchain to detect malware [17]. Therefore, the smart carving mechanism suggested in this study can be used for digital forensics efficiently.

References

[1] A. Pal, Nasir D. Memon, "*The evolution of file carving,*" IEEE Signal Processing Magazine 26(2):59-71, 2009. https://doi.org/10.1109/MSP.2008.931081

[2] File Carving: Smart Carving, Wikipedia, Available May, 1, 2019, [Online] https://www.forensicswiki.org/wiki/File_Carving:SmartCarving.

[3] R. R. Ali, K. M. Mohamad, S. Jamel, S. K. A. Khalid, "*A Review of Digital Forensics Methods for JPEG File Carving*," Journal of Theoretical and Applied Information Technology, Vol.96. No 17, pp.5841-5856, 2018.

[4] R. K. Pahade, B. Singh, and U. Singh, "*A Survey on Multimedia File Carving*", International Journal of Computer Science & Engineering Survey, Vol.6, No.6. 2015.

[5] V. Ganesh, "*Artificial Intelligence* Applied to Computer Forensics", International Journal, 5(5), 2017.

[6] R. R. Ali, K. M. Mohamad, S. Jamel, S. K. A. Khalid, "A Review OF Digital Forensics Methods For JPEG File Carving," Journal of Theoretical and Applied Information Technology, Vol.96, No. 17, pp.5841- 5856, 15th Sep.2018.

[7] M. Nadeem Ashraf, "Forensic Multimedia File Carving", Master's Thesis, KTH, 2013.

[8] Pillow, Available January, 14, 2020, [Online] https://pillow.readthedocs.io/en/stable/.

[9] Foremost, Source Forge, Available January, 14, 2020, [Online] http://foremost.sourceforge.net/.

[10] Scalpel, GitHub, Available January, 14, 2020, [Online] https://github.com/sleuthkit/scalpel.

[11] Bulk extractor, GitHub, Available January, 14, 2020, [Online] https://github.com/simsong/bulk_extractor.

[12] Garfinkel, Simson L., "Digital media triage with bulk data analysis and bulk_extractor.", Computers & Security 32, 56-72, 2013.

[13] A. Singh, N. Jindal, and K. Singh, "*A review on digital image forensics*", International Conference on Signal Processing, page 12-6, 2016.

[14] S. Kim and H. Kim, "Fuzzy Expert System for Detecting Anti-Forensic Activities," Journal of Internet Computing and Services, vol. 12, no. 5, pp. 47-62, 2011.

[15] H. Lee and H. Lee, "Consortium Blockchain based Forgery Android APK Discrimination DApp using Hyperledger Composer," Journal of Internet Computing and Services, vol. 20, no. 5, pp. 9-18, 2019. https://doi.org/ 10.7472/jksii.2019.20.5.9.

[16] C. Park, "Hybrid copy-move-forgery detection algorithm fusing key point-based and block-based approaches," Journal of Internet Computing and Services, vol. 19, no. 4, pp. 7-13, 2018. https://doi.org/10.7472/jksii.2018.19.4.7.

[17] S. Kim and S. Lee, "Automatic Malware Detection Rule Generation and Verification System," Journal of Internet Computing and Services, vol. 20, no. 2, pp. 9-19, 2019. https://doi.org/10.7472/jksii.2019.20.2.9.

◐ 저 자 소 개 ◑

이 형 우(Hyung-Woo Lee)
1994년 2월 고려대학교 컴퓨터학과(이학사)
1996년 2월 고려대학교 대학원 컴퓨터학과(이학석사)
1999년 2월 고려대학교 대학원 컴퓨터학과(이학박사)
2003년 3월~현재 한신대학교 컴퓨터공학부 교수
관심분야 : 모바일/네트워크 보안, 디지털포렌식, 정보보호, etc.
E-mail : hwlee@hs.ac.kr

이 한 성(Han Seong Lee)
2017년 2월 한신대학교 컴퓨터공학부(공학사)
2018년 9월~현재 한신대학교 일반대학원 컴퓨터공학과 석사과정
관심분야 : 블록체인 기술, 모바일 보안, 디지털포렌식
E-mail : jkhanseong@naver.com