ICIC 2016 A File

by Romi Fadillah Rahmat

Submission date: 24-Oct-2019 10:20AM (UTC+0700)
Submission ID: 1199223891
File name: ICIC_2016_07905757.pdf (736.98K)
Word count: 3982
Character count: 19019
A File Undelete with Aho-Corasick Algorithm
In File Recovery

Opim Salim Sitompul, Andrew Handoko, Romi Fadillah Rahmat
Department of Information Technology
Faculty of Computer Science and Information Technology
University of Samatera Utara
Medan, Indonesia
opim@usu.ac.id, andrew.handoko@rocketmail.com, romi.fadillah@usu.ac.id

Abstract—In this research, a file undelete method is proposed by which the file recovery system retrieved the file metadata through a parsing process from the master file table (MFT) attributes. Using the Aho-Corasick algorithm, the process is then continued with a filtering process in which keywords are matched with file names. The result obtained shows that the proposed method is able to perform recovery of files that have been deleted from the file system. The experiment is performed four times with various file conditions which had been overwritten 0%, 18.98%, 32.21% and 59.77% from their original size. The rate of the average file recovery success is 87.50% and the average time required is 0.32 second for string matching on file names.

Keywords: file undelete; file recovery; parsing process; Aho-Corasick algorithm.

I. INTRODUCTION

A file could be used as an authentic evidence in certain criminal cases. In the form of digital file (digital evidence), an evidence is all data stored or transmitted using a computer to support or to deny how a criminal act is happened or to show some important element of a criminal act used as a motive or an alibi [1]. In digital forensic, there are several types of procedure, such as file type identification [2] and file undelete [3]. Most of digital evidence criminalization is related to deleting digital evidence. Digital evidence can be easily eliminated by a criminal with deleting the file. In fact, file deletion could be done only by deleting a reference to the file from system table [4].

In some criminal cases such as corruption, digital evidence file could be a file that has already been deleted from file system and in fact it still could be recovered. Files which have been deleted from the file system could not be accessed using file manager. However, the file is still intact on the hard disk and could be recovered as long as the allocated spaces for that file have not been overwritten by other data, through deletion or hard disk wiping [1]. The file system could be recovered using a technique called file undelete.

In this research, implementations of file undelete and Aho-Corasick algorithm is proposed to recover deleted files. The Aho-Corasick algorithm has been applied to solve various problems such as signature-based anti-virus application [4], structural-to-syntactic matching for similar document [5], set matching in the field of Bioinformatics [6], searching text string on digital forensics [7], and text mining [8].

The rest of the paper will be organized as follows. Section 2 describes some previous research on file recovery. The proposed method is illustrated in Section 3, while section 4 will discuss the result. Finally, section 5 will conclude the research and give some recommendation for further research.

II. RELATED RESEARCH

Some previous research on deleted file recovery are using various methods such as Boyer-Moore algorithm [3], carving process approach to recover multimedia file [9], and forensic reconsctruction for mp3 file [10].

In 2007, a carving method using Boyer-Moore algorithm was proposed to recover deleted files [3]. Their results showed that the carving process requires some resource such as a long processing time and very large storage capacity. The carving process on 8GB target disk resulting more than 1.1 million files with a total of 250GB each, in addition to a very large amount of false positive. Furthermore, applying the Boyer-Moore algorithm was considered less optimal for matching process of file header and footer, which is O(nm).

Reference [10] had reconstructed MP3 file fragment using Variable Bit Rate (VBR). The proposed method is able to enhance successful finding of correct fragment of file to be reconstructed. The percentage of enhancement for high quality MP3 file is 49.20 - 69.42%, for medium quality file is 1.80 - 3.75%, and for low quality file is 41.2 - 100.00%. Successful enhancement in finding fragment from file will enhance the performance of carving process.

For the next research in 2011, a carving method was proposed for multimedia file [9]. The proposed method can recover multimedia files of types MP3, AVI, and WAV perfectly for continuously allocated files. Even if a file is
allocated discontinuously, the file can still be verified after the recovery process due to the multimedia files characteristics. Even though multimedia files saved in NTFS file system which undergo compression are more difficult to recover, but the compressed files can still be recovered using the carving method.

III. METHODOLOGY

A. Proposed Method

The general architecture of proposed method is illustrated in Fig. 1.

![Diagram](image)

Fig. 1. The general architecture of the proposed method

In Fig. 1 we proposed our methodology that can be described in several steps as follows:

Step 1: The first phase is to duplicate the content of storage media (disk imaging) in order to get an identical duplicate of the storage media. The storage media that has been used in this research is a 4GB secondary storage media with NTFS file system (the effective size is 90% from the storage media that is 3.60GB or media cluster size of 4KB (4096 bytes)). In this storage media, there are 56 files with varying file types with a total size of 3.54GB (3,797,409,792 bytes). Each file will undergo a process of calculating CRC-32 value which will be used as a comparative variable in verification process.

Step 2: Accessing and reading the Master File Table (MFT) record to find the records of deleted files. The MFT is parsed to extract the metadata record. In this stage, the cluster number containing the MFT will be read from the 0th boot sector, which is located on 0x30 offset and 8 bytes in length using little endian system. Each record in the MFT is accessed to read information of each file and directory in the storage media. Each existing record is then undergoing a parsing process that will separate each record based on MFT entry header, attribute header containing information about the type, size, and the name of the attribute as well as attribute containing the data.

Step 3: The parsing process on each record is performed in order to obtain useful metadata for the undelete process such as filename, logical cluster number occupied by the file, flag whether the file is deleted, and other information. Based on the metadata obtain, all files with deleted flag will be listed.

Step 4: In this phase, the identification of filename type based on filename extension and signature is conducted. The identification of filename extension is by looking at the filename extension of the file, and the identification of the signature is by occupying the first 32 bytes of the file. Using Table 1 as the reference for filename extension and signature then the next process is creating trie for the specified filename extension and signature. The following process is the classification of damage or undamaged file by using Aho-Corasick algorithm implemented for filename extension’s trie and signature’s trie. Then we use several use as follows:

- If the type of the file was successfully identified based on filename extension and signature, and both of them give a same result, then the file is in undamaged condition.
- If the type of the file was successfully identified based on filename extension and signature, but both of them give a different result, then the file is in damage condition.
- If the type of the file was unsuccessful to be identified based on filename extension, but successfully identified based on signature, then the file is in damage condition.
- If the type of the file was successfully identified based on filename extension, but unsuccessful to be identified based on signature, then the file is in unknown condition.

<table>
<thead>
<tr>
<th>No.</th>
<th>Filename Extension</th>
<th>Signature</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DOC, DOCX, PPT, PPTX, MSI, VSD, XLS, XLSX</td>
<td>E51C1363719E38D9C476E4B184A3A6400ED7F7777</td>
</tr>
<tr>
<td>2</td>
<td>TXT, CSV, HTML</td>
<td>50.40/30/40/40/00/05/00</td>
</tr>
</tbody>
</table>
Step 5: Accepting keyword from user in the form of string of characters, separating the accepted keyword into words delimited by spaces, and then transforming the keyword into an automaton which will be used by Aho-Corasick algorithm in filtering process to find files that matches with the given keywords. In this stage, user will give a keyword in order to find a deleted file. The keyword given by user will be divided into words using spaces as delimiter and then will be transformed into automata that will be used for string matching using Aho-Corasick algorithm. As an example, supposed a user gives a keyword such as "casey crime pdf". This string will be divided into 3 words based on the spaces, namely "casey", "crime", and "pdf". These words will be used to build an automaton as can be seen in Fig. 2.

Step 6: After the automaton has been built then functions for the state movement and identifier for state terminal will be added. Then the list of files with deleted flag will be filtered by matching the filename and the given keyword and deleting filenames that are not match with keywords in the list.

Step 7: Reconstructing file which match with the given keyword based on metadata from MFT. Metadata obtained from MFT record through parsing process is used to recover file. Important information to reconstruct file are filename, Logical Cluster Number (LCN), and allocated size. The reconstruction steps are as follows:

- A new blank file will be created with the same name as the original file and the same allocated space as the original file.
- LCN of the file that will be discovered will be opened and the content of the cluster will be read and saved into a buffer. The amount of data read from the cluster is the same as the allocated buffer size. The content of the buffer will be written into a new file in hexadecimal form. After the data have been written into the empty file, the buffer will be used again for the next file. The written process will be continued from the last offset of the data to be written. This process will be iterated until all cluster content occupied by the deleted file is moved into the new file.

Step 8: Verifying file that has been reconstructed using CRC-32 to make sure whether or not the recovered file is damage. The recovering process will continue by counting the CRC-32 value and then comparing to the CRC value of the file before it is deleted. The verification process will determine whether the file is successfully recovered without damage. If the CRC value is different with the original file, then the recovered file can be considered as in damage condition.

B. Aho-Corasick Algorithm

In our proposed method, Aho-Corasick Algorithm is used in phase 4 and 5 described before.

Aho-Corasick algorithm is string matching algorithm founded by Alfred V. Aho and Margaret J. Corasick (Aho & Corasick, 1975). This algorithm is dictionary adjustment algorithm which place an element in the set of unlimited string. Complexity of the algorithm is \( O(n + m \cdot z) \), where \( n \) is number of pattern, \( m \) is the length of text that used in the searching process, and \( z \) is number of matching output or number of pattern occurred.

Aho-Corasick Algorithm uses keyword trie foundations. Keyword trie with the set of pattern \( P \) is trie with \( K \) as its root of trie where:

1. Every edge \( K \) is named as one character.
2. Two edges which derived from one node \( v \) will get different alphabetical character.
3. For every \( X \in P \) consist of one node \( v \) with \( L(v) = X \).
4. Label \( L(v) \) for every node \( v \) is equivalent with \( X \in P \).

For the construction of keyword trie for pattern set \( P = \{ P_1, ..., P_k \} \) the rules are as follows:
1. Starts from inputting every character one by one from pattern set following its path. This character is named as character of $P_i$. $P_i$ here is the element of $i$ from pattern set $P$.

2. If path ends before $P_i$, then proceed to add new edge and node for every character $P_i$.

3. Save the identifier $i$ from $P_i$, in the node terminal of path.

The construction of keyword trie has the complexity $O(|P_1| + \ldots + |P_n|) = O(n)$.

IV. Result and Discussion

A. Result of undedelete process

There are 56 files used in this experiment with a total of 3,54GB (3,797,949,792 bytes). From the 56 files, 55 file (98.21%) are successfully recovered with a total size of 3,52GB (3,781,166,380 bytes) or 99.71% from total size of deleted files.

Total elapsed time or total time required in the process of recovering 55 file is 229,418 seconds. With the total recovered file size of 3,52GB and time required for recovery is 229,418 seconds. The rate of processing is 15,711 MB/s.

In this experiment, the number of files recovered is 55 files out of 56. According to the analysis performed, some factors which may cause failure to the recovery process are:

1) Damage on MFT entry containing information about Logical Cluster Number (LCN) occupied by the file. This will cause cluster number containing data from the file is not known and data from the file on that cluster couldn't be read.

2) Recovery is performed based on filename attached to the file which is saved as MFT attribute. The buffer size provided for full path filename (complete directory to save recovered file) is not enough.

B. Result of CRC-32 Verification

This section will describe result of comparing the CRC-32 of the file successfully recovered with CRC-32 of the original file in order to detect whether or not the recovered file is damage. Differences in CRC value show that there is an alteration on the file data that causing the differences between the content of the recovered file and the content of the original file. If the storage media is secondary and doesn’t filled with other data, there are 55 (98.21%) files from total files deleted can be recovered successfully without any damage.

C. Result of Aho-Corasick Algorithm

The result of implementing the Aho-Corasick algorithm for filtering process is described in Table II. The keyword is separated with spaces as delimiter (each keyword is written with underscore in Table II). The words obtained are then used to construct trie.

Only after the trie has been constructed than the searching process could be initiated. Apart from matching string containing alphabet, the Aho-Corasick algorithm could also be used to string containing special character such as "*" and characters from foreign language such as the following phrase: "非公開授権APK レンジャー" or "デジター". Matching is performed on 56 filenames that still readable from MFT record entry. Total time required for string matching with Aho-Corasick algorithm for 10 times are 3,198 seconds. Therefore, the average time for each matching process is 0.320 seconds.

D. Testing Results

In this section, we describe the testing results of our proposed method which is shown in tables below.

<table>
<thead>
<tr>
<th>No.</th>
<th>Keyword</th>
<th>Elapsed Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>mpg</td>
<td>0.312</td>
</tr>
<tr>
<td>2</td>
<td>pdf</td>
<td>0.327</td>
</tr>
<tr>
<td>3</td>
<td>supercol Handbook (shortcuts)</td>
<td>0.328</td>
</tr>
<tr>
<td>4</td>
<td>google se 32 320 1.5 14 2012 version</td>
<td>0.312</td>
</tr>
<tr>
<td>5</td>
<td>doc</td>
<td>0.312</td>
</tr>
<tr>
<td>6</td>
<td>pdf</td>
<td>0.312</td>
</tr>
<tr>
<td>7</td>
<td>jpg</td>
<td>0.312</td>
</tr>
<tr>
<td>8</td>
<td>txt</td>
<td>0.312</td>
</tr>
<tr>
<td>9</td>
<td>html</td>
<td>0.312</td>
</tr>
<tr>
<td>10</td>
<td>txt</td>
<td>0.312</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>3.198</td>
</tr>
</tbody>
</table>

Average Time: 0.320

The storage media condition is varied from storage media containing data which has not been overwritten by other data and storage media which some of its data has been partly overwritten.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data size added to media</td>
<td>1,245,450,240 bytes</td>
</tr>
<tr>
<td>Storage number of entry from old file in MFT</td>
<td>55 (98.21%)</td>
</tr>
<tr>
<td>Number of file successfully recovered</td>
<td>48 (98.71%)</td>
</tr>
<tr>
<td>Number of file recovered with damage</td>
<td>7 (12.59%)</td>
</tr>
<tr>
<td>Number of file fail to be recovered</td>
<td>1 (1.70%)</td>
</tr>
<tr>
<td>Total file size got damage</td>
<td>1,634,210,540 bytes</td>
</tr>
</tbody>
</table>

Testing is performed on 4GB storage media (effective size 3.60GB or 3,873,783,808 bytes) containing 56 files with varied types and sizes of 3.54GB (3,797,409,792 bytes). Result is described in Table III.
File entries on MFT that still can be read is 55 (98.21%). From 56 entries read, 48 files can be recovered successfully (85.71%). 7 files can be recovered with some damage (12.50%) with total file size of 1,631,210,540 bytes, and 1 file failed to be recovered (1.79%) because the file entry in MFT has been deleted.

Testing results in Table IV is a testing on storage media with data overwritten by 2,310,575,435 bytes or 59.77% of total media storage size.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data size added to media storage</td>
<td>2,310,575,435 bytes</td>
</tr>
<tr>
<td>Number of entry from old file in MFT</td>
<td>55 (98.21%)</td>
</tr>
<tr>
<td>Number of file successfully recovered</td>
<td>46 (82.14%)</td>
</tr>
<tr>
<td>Number of file recovered with damage</td>
<td>9 (16.07%)</td>
</tr>
<tr>
<td>Number of file failed to recover</td>
<td>1 (1.79%)</td>
</tr>
<tr>
<td>Total file size got damage</td>
<td>2,337,387,248 bytes</td>
</tr>
</tbody>
</table>

File entry on MFT that still can be read is 55 (98.21%). From 56 entries that can be read, 46 files are successfully recovered (82.14%). 9 files can be recovered with some damage (16.07%) with total size of 2,337,387,248 bytes for all files are damaged, and 1 file is failed to be recovered (1.79%) because the file entry has been deleted from MFT.

Even though a file entry can still be read from MFT, however it could not guarantee that this file can be perfectly recovered since the original cluster occupied by the file could be override by data from other files. If the undelete process is performed on this file, the result is a file which contain some data from another file. This file will have different CRC calculation from what it supposedly had so that this file could be considered as a damage file.

V. CONCLUSION AND FUTURE RESEARCH

From the implementation of the proposed method namely file undelete and the Aho-Corasick algorithm to recover files which are deleted from file system, it can be concluded that the proposed method gives a successful level of recovery file of 98.21% for data that have not been overwritten and the average successful level from four tests with level of data overwritten 9%, 18.98%, 32.21%, and 59.77% are 87.50%. Therefore, it can be concluded that the file undelete method proposed is able to recover files that have been deleted from file system so they could be reused. The string matching method with the Aho-Corasick algorithm is able to quickly match the string given with the average speed of record matching from 56 filenames for 0.32 seconds.

For further research, the method could be given additional features such as capability to detect whether a file got damage before the file is recovered. Another feature that could be considered is a capability to identify the file type based on the content characterized from every byte in the file so that the file could be opened and read in order to discover the file content. Lastly, a feature to detect whether the recovered file is a complete file that could be opened by a certain application or it is a fragment of another file.

REFERENCES

ICIC 2016 A File

ORIGINALITY REPORT

10% SIMILARITY INDEX 6% INTERNET SOURCES 7% PUBLICATIONS 5% STUDENT PAPERS

PRIMARY SOURCES

3. Submitted to Naval Postgraduate School

Hong-Rong Yang, Ming Xu, Ning Zheng. "An Improved Method for Ranking of Search Results Based on User Interest", 2008 IFIP International Conference on Network and Parallel Computing, 2008

Submitted to King's College

Submitted to University of Sunderland

Hong-Rong Yang, Ming Xu, Ning Zheng. "An Improved Method for Ranking of Search Results Based on User Interest", 2008 IFIP International Conference on Network and Parallel Computing, 2008

www.slideshare.net

cirg.cs.up.ac.za

www.ariadne-infrastructure.eu

Submitted to Deltak

primis.phmsa.dot.gov


Submitted to British University In Dubai