World Academy of Science, Engineering and Technology
International Journal of Electrical and Computer Engineering
Vol:1, No:8, 2007

High Speed Bitwise Search for Digital Forensic System

Hyungkeun Jee, Jooyoung Lee, and Dowon Hong

***Abstract*—**The most common forensic activity is searching a hard disk for string of data. Nowadays, investigators and analysts are increasingly experiencing large, even terabyte sized data sets when conducting digital investigations. Therefore consecutive searching can take weeks to complete successfully. There are two primary search methods: index-based search and bitwise search. Index-based searching is very fast after the initial indexing but initial indexing takes a long time. In this paper, we discuss a high speed bitwise search model for large-scale digital forensic investigations. We used pattern matching board, which is generally used for network security, to search for string and complex regular expressions. Our results indicate that in many cases, the use of pattern matching board can substantially increase the performance of digital forensic search tools.

***Keywords*—**Digital forensics, search, regular expression.

I. Introduction

TEXTUAL evidence is important to the vast majority of digital investigations. So searching the contents of an imaged drive are the most common forensic tasks performed by an examiner. A common search technique is to search for files based on their names or patterns in their names. Another common search technique is to search for files based on a keyword in their content. Digital forensic text string searches are designed to search every byte of the digital evidence, at the physical level, to locate specific text strings of interest to the investigation. Recent surveys of string searching can be found in [1,2,8]. However, the capacity of the storage device is drastically increasing by the day. Large-scale digital forensic investigations present a fundamental challenge. It is accommodating the computational needs of a large amount of data to be processed. Consecutive searching which searches based on previous search results can therefore take weeks to complete successfully. This problem could result in long processing times that can seriously hamper an investigation. In this paper, so we propose a high speed bitwise search model for large-scale digital forensic investigations.

Hyungkeun Jee is with the Electronics and Telecommunications Research Institute, 161 Gajeong-dong, Yuseong-gu, Daejeon, Korea (phone: +82-42-860-1674; fax: +82-42-860-5112; e-mail: hkjee@ etri.re.kr).

Jooyoung Lee is with the Electronics and Telecommunications Research Institute, 161 Gajeong-dong, Yuseong-gu, Daejeon, Korea (phone: +82-42-860-5849; fax: +82-42-860-5112; e-mail: joolee@ etri.re.kr).

Dowon Hong is with the Electronics and Telecommunications Research Institute, 161 Gajeong-dong, Yuseong-gu, Daejeon, Korea (phone: +82-42-860-6147; fax: +82-42-860-5112; e-mail: dwhong@ etri.re.kr).

The remainder of this paper is organized as follows: Chapter 2 provides brief overviews of some searching methodologies, chapter 3 describes our proposed searching system for large-scale digital evidence, and chapter 4 shows the experimental results. Finally chapter 5 presents conclusions and future work.

II. Related Work

Generally there are two primary methods to search for string: Index-based searching and bitwise searching. Each technique has merits and demerits, so many investigations call for a combination of searching techniques. The remainder of this section will discuss research in these fields, thereby laying the foundation for the text string search approach proposed in this paper.

A. Index-based Searching

Index-based searching is functionally similar to internet search engines in the sense that their basic process creates an index of keywords based on the contents of files and executes queries against inverted files. This method generally opens all files on a disk, searches them for repeating strings of printable characters, and creates a table of the repeated strings with pointers to the original content.

Index searching has two primary advantages over bitwise searching. Because it is file-based, index searching can utilize hooks into various file types to index their contents in its native format. This allows the proper searching of contents for applications like XLS files and PDF files, which store data in a modified format, and compressed files such as WinZip. Bitwise searches on these file types are ineffective as the data is not stored in a directly readable ASCII formation. Secondly, after the initial indexing, searching is extremely fast. Searching for tangential terms based on initial search results and browsing of indexed words for similar spellings is extremely effective with index searching. Therefore an index-based search may be used to provide quick, repeated searches with new terms on files copied from a shared drive.

But Index-based searching is not extensible to digital forensic text string searching for two reasons [5]. The first reason is the high startup costs of the index creation process. Initial indexing of one's file system can take days. The second reason is that digital forensic text string searches seek to find digital evidence independent of the file system. They endeavor to find data at the physical level including both unallocated and slack space, as opposed to the logical file system level.

B. Bitwise Searching

Bitwise searching looks for simple text strings or regular expression matches in any sectors on a drive including both unallocated and slack space [6]. Slack space occurs when the size of a file is not a multiple of a data unit size [4]. A file must allocate a full data unit, even if it needs only a small part of it, and the unused bytes in the last data unit are called slack space. For example, if a file is 612 byes, it needs to allocate a full 2,048-byte data unit, and the final 1,436 bytes would be slack space. Slack space is important because computers do not wipe the unused bytes, so the slack space contains data from previous files or from memory.

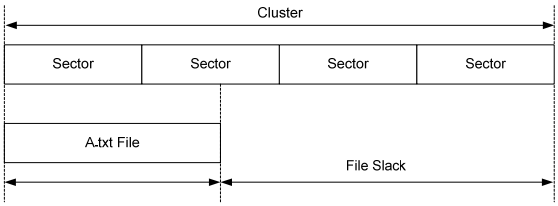

Fig. 1 Slack space of a 612-byte file in a 2048-byte cluster

It performs a full, regular expression-based search on the raw data, file specific or not. A full bitwise search may be more relevant if a hard disk is being searched for deleted files or residual fragments of their content and when searching for very complex regular expressions(for example, looking for all strings that match a credit card number or phone number). The ability to do regular expression searches enables the examiner to search for non-text (binary) values such as file headers as well as complex text terms. The criminal might change the extension of files to hide the files. But we can find all files of given type even if someone has changed their name by searching for files based on signatures in their header.

The two biggest limitations of bitwise searches are speed and non-standard file formats. A bitwise search can take multiple hours depending on the size of the drives or images searched. Consecutive searching (searches based on previous search results) can therefore take weeks to complete successfully. In addition to the speed concern, file formats such as XLS and PDF, compressed files will not be found by standard bitwise search.

C. Choice of Searching Methodology

Many investigations call for a combination of searching techniques, and the methodology applied to a particular case needs to be context specific. Factors affecting the choice and order of searching include [6]:

Awareness of suspect: If the suspect is aware that he is under investigation, file-based content may have been deleted, which leans toward bitwise searching. If the content is likely to still be present on the drive intact, index-based searching may be more effective.

Likely data format: If there is a chance that the content resides in PDF, XLS, HWP, index-based searching will be more through. A preliminary bitwise search for the header bytes from these file types and subsequent recovery of deleted files before the index-based search will combine both techniques for the maximum effectiveness.

Time constraints: For a single keyword or group of keywords, a bitwise search will be slightly faster in most circumstances due to the overhead created by indexing. If subsequent or real-time searches are expected, index-based searches will be faster overall.

Search complexity: When searching for very complex regular expressions (for example, looking for all strings that match a credit card number or phone number), bitwise search tools will have more success. Searches based on synonyms of words, phonetically similar words, and fuzzy spelling of words will be more successful with index-based searches.

III. Proposed Searching System

As mentioned above, a full bitwise searching has many merits for forensic investigation, but it has the two biggest limitations. The one is speed and the other is non-standard file formats. In order to solve these problems, this paper describes high-speed searching system for large scale device based on bitwise search using pattern matching board. Fig. 2 shows our proposed searching process.

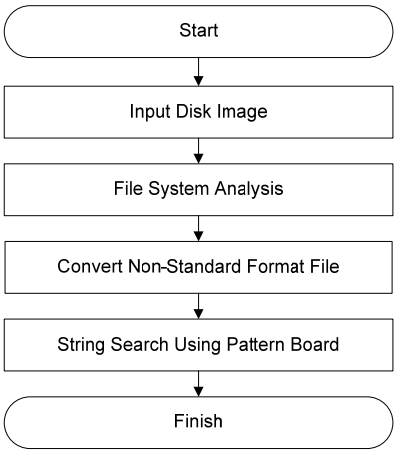

Fig. 2 Proposed searching process

A. File System Analysis

The file formats such as XLS, HWP and PDF files will not be found by standard bitwise search because the data of these file types is not stored in a directly readable ASCII formation. So we must convert these files into the readable plain text in order to search strings accurately. At first, the file system of the evidence image has to be analyzed to find out which files are stored in this image. File systems provide a mechanism for users to store data in a hierarchy of files and directories. A file system consists of structural and user data that are organized such that the computer knows where to find them. Generally the most of file systems manages files with metadata. The metadata contains the data that describe a file. It contains information, such as where the file content is stored, how big the file is, the times and dates when the file was last read from

or written to, and access control information. It does not contain the content of the file.

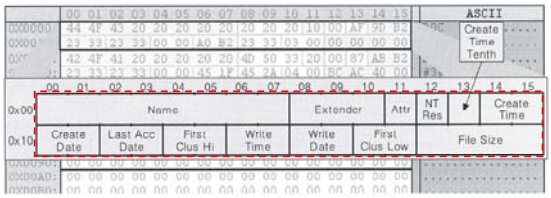

Fig. 3 Directory Entry of FAT file system

In case of FAT file system, directory entry is a data structure that is allocated for every file and directory. It is 32 bytes in size and contains the file's name, extender, attribute, size, starting cluster, and dates and times as shown in Fig. 3. Thus by analyzing the file system we can know the files stored in a disk image.

B. Convert Non-Standard File Formats

After analyzing file system, we select all non-standard format files on the disk image and convert these files into plain text using parser. Each parser exists for each document format. For example, PDF file uses PDFBox(0.7.3) library and MS-Office file uses POI(3.0) library. And these converted plain text files are stored in the evidence file storage server with corresponding image file before searching. Figure 4 shows the layout of evidence file storage server which stores disk images and plain text files of each image.

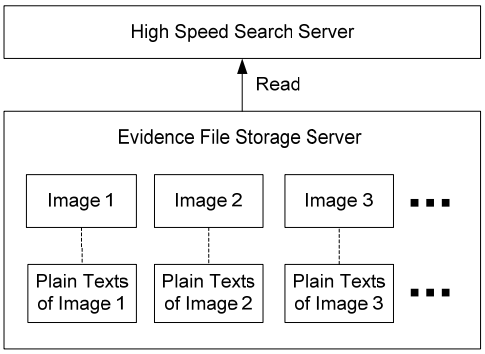

Fig. 4 Layout of Evidence File Storage Server

C. Bitwise Search using Pattern Board

One of the drawbacks of bitwise search on a full disk image is the high amount of searching time. Moreover in case of searching multiple keywords at the same time, it takes more time. To solve this search speed problem, we use pattern matching board to search keywords in the high speed search server. The pattern matching board which we use is CPX-3113 made by Tarari [9]. Originally this board is mainly used for network security such as Intrusion Detection/ Prevention System. This board supports the PCI-X interface to provide high levels of performance (at speeds up to 1 Gigabit per second). And up to 100K complex regular expression rules can be processed simultaneously.

Fig. 5 Pattern matching board

We search multiple keywords and regular expressions on a disk image and converted plain texts successively using this pattern matching board and return the physical location of matched patterns.

IV. Experimental Results

A. System and Test Data

We are finally ready to present some experiments with our proposed searching system that validate our work. Our High Speed Search Server is dual core Dell with a Pentium 2.8GHz, 4GB of RAM. And for high speed search, we use CPX-3113 pattern matching board. The test data is consisted of 1,512 files varying in size from 22KB to 3MB for a total of about 1GB of data with variable file formats.

B. Results

The results of comparing searching speed between software and pattern board are presented in Table I. We searched multiple keywords and regular expressions increasing number of keywords from 1 to 200 at the 1GB test disk image.

TABLE I
Comparison of Bitwise Searching Time

Number of Keywords	Bitwise searching time (sec)	
	Software	Pattern Board
1	4	4
10	61	42
50	385	104
100	602	121
200	1,026	133

According to the experimental result, the searching speed is similar with software and pattern board when we search with less than 10 keywords. However the searching speed of software is linearly increased with an increasing the number of keywords, whereas the searching speed using pattern matching board is a little bit increased. When we search with 200 keywords, the searching speed using pattern board is about 8 times faster than that of software.

V. Concluding Remarks

In this paper, we presented a high speed bitwise search model using pattern matching board for large-scale digital forensic investigations. Experimental results show that our

proposed method improves two biggest drawbacks of bitwise search, speed and non-standard file formats. However, Bitwise searches can fail due to file boundaries. On a heavily fragmented disk, many files are stored in non-contiguous sectors. Any text on the boundary of one of these sectors (that is, spanning two sectors) will not be accurately identified by most bitwise tools unless the sectors are contiguous. To prevent this case, we are considering some ways to reorder all sectors in order that all files are allocated consecutively before searching.

REFERENCES

[1] Beebe NL, Dietrich G., "A new process model for text string searching," *Research advances in digital forensics III*. Norwell: Springer, 2007, pp. 73-85.
[2] Baeza-Yates, R., *String searching algorithms. In information Retrieval: Algorithms and Data Structures*, Chap. 10, W. Frakes and R. Baeza-Yates, Eds., Prentice Hall, Englewood Cliffs, N.J.m 1992, pp. 219-240.
[3] E. Casey, *Handbook of Computer Crime Investigation: Forensic Tools and Technology*, Academic Press, San Diego, California, 2002.
[4] Brian Carrier, *File System Forensic Analysis*, Pearson Education, Inc., 2005.
[5] Nicole Lang Beebe, Jan Guynes Clark, "Digital forensic text string searching," *Proceedings of the 2007 digital forensics research workshop (DFRWS 2007)*, pp. 49-54.
[6] Chad Steel, *Windows Forensics*, Wiley Publishing, Inc. 2006.
[7] W. B. Frakes and R. Baeza-Yates, *Information Retrieval: Data Structure & Algorithms*, Prentice Hall, 1992.
[8] Kukich K., "Techniques for Automatically Correcting Words in Text," *ACM Computing Surveys*, vol. 24, No. 4, 1992, pp. 377-440.
[9] Tarari. Available: http://www.tarari.com