International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:2, 2009

Inefficiency of Data Storing in Physical Memory

Kamaruddin Malik Mohamad, Sapiee Haji Jamel, Mustafa Mat Deris

***Abstract*—**Memory forensic is important in digital investigation. The forensic is based on the data stored in physical memory that involve memory management and processing time. However, the current forensic tools do not consider the efficiency in terms of storage management and the processing time. This paper shows the high redundancy of data found in the physical memory that cause inefficiency in processing time and memory management. The experiment is done using Borland C compiler on Windows XP with 512 MB of physical memory.

***Keywords*—**Digital Evidence, Memory Forensics.

I. Introduction

DIGITAL Forensic Science is defined as "The use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations" [1].

Memory forensic is important to fight against a more sophisticated attack from malicious softwares and viruses such as FU rootkit, Code Red or the SQL Slammer worms because they only resides in the memory without writing to the hard disk. "There are also other advantages of performing memory investigation. Let's suppose, that we need to recover a part of email or a part of a document lost after a word editor crash. Where are we going to look it for? Even a simple task of searching of strings in main memory is sometimes very useful and allows us to extract interesting information such as commands typed by an intruder". Nevertheless, a trusted toolkit need to be used to prevent known data to be written to the memory and data (evidence) in the memory to be overwritten [2].

Kamaruddin Malik bin Mohamad is with the Faculty of Information Technology and Multimedia, University of Tun Hussein Onn Malaysia, 806400 Parit Raja, Batu Pahat, Johor, Malaysia. (E-mail: malik@ uthm.edu.my).

Sapiee Haji Jamel is a senior lecturer with the Faculty of Information Technology and Multimedia, University of Tun Hussein Onn Malaysia, 806400 Parit Raja, Batu Pahat, Johor, Malaysia. (E-mail: sapiee@ uthm.edu.my).

Mustafa Mat Deris is the dean of Faculty of Information Technology and Multimedia, University of Tun Hussein Onn Malaysia, 806400 Parit Raja, Batu Pahat, Johor, Malaysia. (E-mail: mmustafa@ uthm.edu.my).

This paper shows that the value for a variable in C program is redundantly stored (more than once eventhough only one unique keyword is initialized into a variable) in the physical memory. From this point onwards, the term memory is used to denote physical memory or Random Access Memory (RAM). The memory image is acquired using free downloadable forensic tools called Helix live CD. The memory image (binary file) is then analyzed using another small free downloadable extraction tool called BinText. String search is done in BinText to count the number of occurences for each of the keywords used by the program from the memory. The keyword will be highlighted in BinText.

The rest of the paper is organized as follows. Section 2 describes Helix live CD, the tool used to acquire the data from the memory (memory image). Section 3 describes BinText, the other free forensic tool used to view the memory image. Section 4 discussed about the result and finally section 5 concludes this paper with future work of this research.

II. Helix Live CD

Helix Live CD is a customized version of Knoppix Live Linux CD [3]. This bootable live CD incorporates many tools and one of them is sdd tool (specialized dd tool) which is used to acquire memory image in this experiment. A screenshot of Helix Live CD memory acquisition is shown in Figure 1. A list of tools included in the live CD is shown in Table 1. Organizations that are using Helix live CD as the forensic tool in their Incident Response/Forensics Training are listed in Table 2.

Fig. 1. A screenshot from Helix live CD for capturing memory imagers.

TABLE I
TOOLS INCLUDED IN HELIX LIVE CD

Tool	Description
sleuthkit	Brian Carrier's replacement to TCT
autopsy	Web front-end to sleuthkit
mac-robber	TCT's graverobber written in C
fenris	debugging, tracing, decompiling
wipe	Secure file deletion
MAC_Grab	e-fense MAC time utility
AIR	Steve Gibson Forensic Acquisition Utility
foremost	Carve files based on header and footer
fatback	Analyze and recover deleted FAT files
md5deep	Recursive md5sum with db lookups
sha15deep	Recursive sha1sum with db lookups
dcfldd	dd replacement from the DCFL
sdd	Specialized dd w/better performance
PyFLAG	Forensic and Log Analysis GUI
Faust	Analyze elf binaries and bash scripts
e2recover	Recover deleted files in ext2 file systems
Pasco	Forensic tool for Internet Explorer Analysis
Galleta	Cookie analyzer for Internet Explorer
Rifiuti	"Recycle BIN" analyzer
Bmap	Detect & Recover data in used slackspace
Ftimes	A toolset for forensic data acquisition
chkrootkit	Look for rootkits
rkhunter	Rootkit hunter
ChaosReader	Trace tcpdump files and extract data
lshw	Hardware Lister
logsh	Log your terminal session (Borrowed from FIRE)
ClamAV	ClamAV Anti Virus Scanner
F-Prot	F-Prot Anti Virus Scanner
2 Hash	MD5 & SHA1 parallel hashing
glimpse	Indexing and query system
Outguess	Stego detection suite
Stegdetect	Stego detection suite
Regviewer	Windows Registry viewer
Chntpw	Change Windows passwords
Grepmail	Grep through mailboxes
logfinder	EFF logfinder utility
linen	EnCase Image Acquisition Tool
Retriever	Find pics/movies/docs/web-mail
Scalpel	Carve files based on header and footer

TABLE II
ORGANIZATIONS THAT USE HELIX LIVE CD FOR INCIDENT RESPONSE/FORENSICS TRAINING

Organization	Training
e-fense	Helix Incident Response & Computer Forensics
NW3C	Linux Forensics
SANS Track 508	System Forensics, Investigation and Response
InfoSec Institute	Computer Forensics Training
SEARCH	Basic Investigators Training

III. BINTEXT

BinText is a small free download software from Foundstone Inc [4]. A screenshot of BinText filtering screen is shown in Figure 2. It has the ability to extract binary file (the memory image file) to plain ASCII text or Unicode (double byte ANSI). The software also incorporate features to filter unwanted text to be displayed and with a simple string search option.

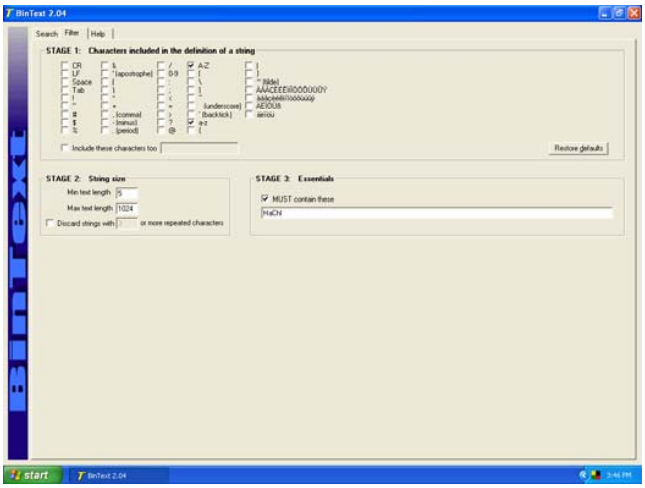

Fig. 2. A screenshot of BinText filtering screen

IV. RESULT AND DISCUSSION

The experiment is done on a Windows XP with 512MB of memory. The simple C program is written using Borland Turbo C for DOS version. The C program initializes and stores a unique value into a variable. The value for the variable will definitely be kept in the physical memory when the program is run. The algorithm is shown below.

```
begin
    variable declaration
    initialize a variable with a unique string
        e.g.“SoNgO”, “NuEvE”, ”HaChI” etc
end
```

Example 1. Small C Program Codes to initialize variable with unique string to the physical memory

The unique string must be used to initialize the variable because it would make the searching of the string from the physical memory much easier. This is to avoid the used of common string which may be written to the memory not by the program that is being tested. In this experiment, the word “SoNgO” (“songo” is a Malay Javanese word which means the number nine), “NuEvE” (“nueve” is a Spanish word for nine) and “HaChI” (which mean the number eight in Japanese). Not only unique and not-so-common strings are used, but also the capital and small letters are being used alternately to make it more easily to do the string search process.

The experiment is done according to the following steps:

1. Open the Turbo C in the Windows DOS mode;
2. Write the small C program; (initialize with the word "SoNgO")
3. Save, compile and run the program;
4. Close Turbo C and DOS mode;
5. Run Helix live CD and acquire memory image using built-in dd function;
6. Save the memory image.

Once the memory image is saved, the following experiment is done as described below:

1. Change the initialization value using the word "NuEvE"
2. Save, compile and run the program;
3. Close Turbo C and DOS mode;
4. Run Helix live CD and acquire memory image using built-in dd function;
5. Save the memory image.

Repeat the steps from 1 to 5 to run the program using different unique keywords. In this experiment, the program is run three times for 3 keywords.

Each of the memory images (in this experiment, there are 3 images to store 3 different keywords) will be analyzed using BinText (binary to text software converter) free software to search for a specific keyword. The occurrences of the searched keywords are tabulated in Table 3.

TABLE III THE KEYWORDS OCCURRENCE IN THE MEMORY

Keyword	**SoNgO**	**NuEvE**	**HaChI**
Occurrence	10	8	14

V. CONCLUSION

This experiment shows that a variable in C program is redundantly stored (more than once) in the physical memory. Many occurrences of the same variable would be inefficient in terms of storage and processing time taken. It also suggests that many artifacts or traces are left behind by a single variable in C program in the memory. We hope to come out with a new approach to reduce the number of occurrences of a C variable stored in the memory.

REFERENCES

[1] DFRWS Technical Report, (2001), A Road Map for Digital Forensic Research, Digital Forensic Research Workshop (DFRWS).
[2] Burdach, M. (2005), Digital Forensics of the Physical Memory, Warsaw University.
[3] http://www.e-fense.com/helix/; accessed on 09 Sep, 2008.
[4] http://www.foundstone.com/us/resources/proddesc/bintext.htm; accessed on 09 Sep, 2008.